Low-Level Design Document (LLDD)

Xyz Employee Management System

1. Introduction

- The **Xyz Employee Management System (EMS)** is designed to manage employees of XYZ Corporation across three categories: **Full-Time**, **Contractor**, and **Intern**.
- The system supports the following core operations:
- Adding employees (random or specific type)
- Removing (resigning) employees
- Searching employees by ID or name
- Converting employee types (Intern/Contractor → Full-Time)
- Managing leave details (for Full-Time employees)
- Maintaining employee status transitions (Active, Inactive, Resigned)
- The system uses a **custom Extended Doubly Linked List (EDLL)** for managing employee collections without relying on STL libraries.

2. Core Entities (Classes & Interfaces)

- The system consists of the following core components:
- **2.1 XyzEmployeeIF (Interface)**
    - Defines a contract that all employee types must follow.
    - Responsibilities:
- Provide getter methods for common attributes
- Enforce implementation of printDetails()
- Ensure abstraction and consistency across employee types
- **2.2 XyzEmployee (Abstract Base Class)**
    - Represents common properties shared across all employee types.
    - Common Attributes:
- Employee ID
- Name
- Gender
- Status
- Employee Type
- Date of Birth
- Date of Joining
- Date of Leaving
- Responsibilities:
- Store shared employee information
- Provide common getter methods
- Declare printDetails() as a virtual function
- Enable polymorphism
    - This class implements XyzEmployeeIF.
- **2.3 XyzFullTimeEmployee (Derived Class)**
    - Inherits from XyzEmployee.
    - Additional Attribute:

- Leaves Availed (0–22)
- Responsibilities:
- Track leave usage
- Calculate remaining leaves
- Override printDetails() to include leave details
- **2.4 XyzContractorEmployee (Derived Class)**
    - Inherits from XyzEmployee.
    - Additional Attribute:
- External Agency
- Responsibilities:
- Store contractor agency details
- Automatically compute Date of Leaving as (DOJ + 1 year)
- Override printDetails() to include agency information
- **2.5 XyzInternEmployee (Derived Class)**
    - Inherits from XyzEmployee.
    - Additional Attributes:
- College
- Branch
- Responsibilities:
- Store intern academic information
- Automatically compute Date of Leaving as (DOJ + 6 months)
- Override printDetails() to include college and branch

3. Data Structures

3.1 EDLL<T> (Extended Doubly Linked List)

A custom-built doubly linked list implementation.

Features:

- Insert at front
- Insert at rear
- Insert at middle (based on index)
- Remove from front
- Remove from rear
- Remove from middle (based on index)
- Traverse list

3.2 Node<T>

Each node contains:

- Data (T)
- Pointer to previous node
- Pointer to next node

Relationship:

- **Composition between EDLL and Node**
    - Node lifecycle depends entirely on EDLL.
    - When EDLL is destroyed, all Nodes are destroyed.

4. Manager Class

XyzEmployeeManager

Acts as the central controller of the system.

Data Members:

- EDLL<XyzEmployee*> activeEmployees
- EDLL<XyzEmployee*> resignedEmployees

Responsibilities:

- t provides operations to add new employees, remove or process resignations of existing ones, search for employees by ID or name, and convert employees from one type to another (e.g., from contractor to full-time).

5. Enumerations

Enumerations are used to enforce controlled values and improve type safety.

- EmployeeStatus
  - ACTIVE
  - INACTIVE
  - RESIGNED
- EmployeeType
  - FULLTIME
  - CONTRACTOR
  - INTERN
- Agency
  - AVENGERS
  - JUSTICE_LEAGUE
  - X_MEN
- College
  - IIT_DELHI
  - IIT_MUMBAI
  - IIT_KANPUR
  - IIT_HYDERABAD
  - NIT_WARANGAL
  - NIT_TIRUCHI
  - IIIT_HYDERABAD
- Branch
  - CSE
  - CSIT
  - ECE

6. Relationships in UML

The system defines the following relationships:

6.1 Inheritance

- XyzFullTimeEmployee → XyzEmployee
- XyzContractorEmployee → XyzEmployee
- XyzInternEmployee → XyzEmployee

6.2 Interface Implementation

- XyzEmployee implements XyzEmployeeIF

6.3 Composition

- XyzEmployeeManager → EDLL
- EDLL → Node

6.4 Association

- XyzEmployee → Enums (Status, Type, Gender)
- XyzContractorEmployee → Agency
- XyzInternEmployee → College, Branch

7. Behavioral Explanation

- **7.1 Adding an Employee**

1. Manager generates employee details.
2. Creates appropriate derived object.
3. If status = RESIGNED → store in resigned list.
4. Else → store in activeEmployees list.

- **7.2 Removing an Employee**

1. Search employee in active list.
2. Change status to RESIGNED.
3. Populate Date of Leaving.
4. Remove from active list.
5. Insert into resigned list.

- **7.3 Searching an Employee**
- Traverse active list.
- If not found, traverse resigned list.
- Match by ID or name.
- **7.4 Converting Employee Type**
- Example: Intern → Full-Time

1. Locate Intern object.
2. Create new Full-Time object.
3. Copy common attributes.
4. Remove Intern from list.
5. Insert new Full-Time employee.
6. Delete old Intern object (memory cleanup).

- **7.5 Adding Leaves to All Full-Time Employees**
    - Traverse active list.
    - For each Full-Time employee:
    - Increase leave count
    - Ensure total does not exceed 22