**Case Study: Employee Management System using Spring Core Module**

# 🎯 Objective

To build a simple Employee Management System demonstrating the core features of the Spring Core Module, especially Inversion of Control (IoC) and Dependency Injection using XML-based configuration and Annotation-based configuration.

# ✅ Requirements

- Manage employee records (add, list)  
- Use interfaces and services to follow decoupled architecture  
- Apply Spring’s Dependency Injection to manage the service layer  
- Configure beans using Spring XML and Annotation-based configuration

# 📁 Project Structure

com.company.ems  
├── App.java  
├── config  
│ └── AppConfig.java  
├── model  
│ └── Employee.java  
├── dao  
│ └── EmployeeDAO.java (Interface)  
│ └── EmployeeDAOImpl.java (Class)  
├── service  
│ └── EmployeeService.java (Interface)  
│ └── EmployeeServiceImpl.java (Class)

# 🚀 5. App.java (Main Application)

package com.company.ems;  
  
import com.company.ems.model.Employee;  
import com.company.ems.service.EmployeeService;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**public class App {** public static void main(String[] args) {  
 ApplicationContext context = new ClassPathXmlApplicationContext("ems-beans.xml");  
   
 EmployeeService empService = (EmployeeService) context.getBean("employeeService");  
  
 empService.addEmployee(new Employee(1, "John"));  
 empService.addEmployee(new Employee(2, "Alice"));  
  
 empService.listEmployees().forEach(emp -> System.out.println(emp.getId() + ": " + emp.getName()));  
 }  
}