**Exercise 7: Employee Management System - Enabling Entity Auditing**

To enable entity auditing in your Employee Management System using Spring Data JPA, you can follow these steps:

1. **Add Dependencies**

Ensure your pom.xml has the necessary dependencies:

<dependencies>

<!-- Other dependencies -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

</dependencies>

1. **Enable Auditing in the Application**

You need to enable auditing in your Spring Boot application. To do this, annotate your main application class with @EnableJpaAuditing.

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.data.jpa.repository.config.EnableJpaAuditing;

@SpringBootApplication @EnableJpaAuditing

public class EmployeeManagementApplication { public static void main(String[] args) {

SpringApplication.run(EmployeeManagementApplication.class, args);

}

}

1. **Create Auditable Entity Base Class**

Create a base class that other entities can inherit from. This class will include common auditing fields.

import org.springframework.data.annotation.CreatedDate;

import org.springframework.data.annotation.LastModifiedDate;

import org.springframework.data.jpa.domain.support.AuditingEntityListener;

import javax.persistence.EntityListeners;

import javax.persistence.MappedSuperclass; import java.time.LocalDateTime;

@MappedSuperclass

@EntityListeners(AuditingEntityListener.class) public abstract class Auditable {

@CreatedDate

private LocalDateTime createdDate;

@LastModifiedDate

private LocalDateTime lastModifiedDate;

// Getters and Setters

}

1. **Add Auditing Fields to Entities**

Now, modify your Employee and Department entities to extend the Auditable base class and add additional fields like @CreatedBy and @LastModifiedBy.

For Employee:

import org.springframework.data.annotation.CreatedBy;

import org.springframework.data.annotation.LastModifiedBy; import javax.persistence.Entity;

@Entity

public class Employee extends Auditable {

private String name;

private String department; private double salary;

@CreatedBy

private String createdBy;

@LastModifiedBy

private String lastModifiedBy;

// Getters and Setters

}

For Department:

import org.springframework.data.annotation.CreatedBy;

import org.springframework.data.annotation.LastModifiedBy; import javax.persistence.Entity;

@Entity

public class Department extends Auditable { private String name;

@CreatedBy

private String createdBy; @LastModifiedBy

private String lastModifiedBy;

// Getters and Setters

}

1. **Configure AuditorAware Implementation**

You need to implement the AuditorAware interface to specify the current user or system performing the operation.

import org.springframework.context.annotation.Configuration; import org.springframework.data.domain.AuditorAware;

import java.util.Optional; @Configuration

public class AuditorAwareImpl implements AuditorAware<String> {

@Override

public Optional<String> getCurrentAuditor() {

// Here you can fetch the current logged-in user, for now returning a fixed value return Optional.of("admin"); // Replace with the actual user fetching logic

}

}

1. **Register AuditorAware Bean**

Register the AuditorAware implementation as a bean in your configuration.

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration; import org.springframework.data.domain.AuditorAware;

@Configuration

public class AuditConfig {

@Bean

public AuditorAware<String> auditorProvider() { return new AuditorAwareImpl();

}

}

1. **Database Changes**

Make sure your database schema supports these auditing fields. If you are using an ORM tool like Hibernate, the schema will be updated automatically if hibernate.ddl-auto is set to update or create.

1. **Testing the Auditing**
   * Create or Update Entities: When you create or modify an Employee or Department, the fields createdDate, lastModifiedDate, createdBy, and lastModifiedBy should be automatically populated.
   * Verify Data: Check the database to ensure that these fields are being correctly updated.