### 1. Enable Auditing in Your Application

#### 1.1 Configure Auditing Properties

You need to enable JPA Auditing in your Spring Boot application. This is done by using the @EnableJpaAuditing annotation in one of your configuration classes.

**Application Configuration:**

package com.example.employeemanagementsystem.config;

import org.springframework.context.annotation.Configuration;

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

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

@Configuration

@EnableJpaRepositories(basePackages = "com.example.employeemanagementsystem.repository")

@EnableJpaAuditing

public class JpaConfig {

}

### 2. Add Auditing Annotations to Your Entities

You’ll use annotations like @CreatedBy, @LastModifiedBy, @CreatedDate, and @LastModifiedDate to track creation and modification details.

#### 2.1 Create an Auditable Base Class

Create a base class for entities that need auditing. This class will include fields and annotations to store auditing information.

**Auditable.java:**

package com.example.employeemanagementsystem.model;

import org.springframework.data.annotation.CreatedDate;

import org.springframework.data.annotation.LastModifiedDate;

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

import javax.persistence.\*;

import java.time.Instant;

@MappedSuperclass

@EntityListeners(AuditingEntityListener.class)

public abstract class Auditable {

@CreatedDate

@Column(nullable = false, updatable = false)

private Instant createdDate;

@LastModifiedDate

@Column(nullable = false)

private Instant lastModifiedDate;

public Instant getCreatedDate() {

return createdDate;

}

public void setCreatedDate(Instant createdDate) {

this.createdDate = createdDate;

}

public Instant getLastModifiedDate() {

return lastModifiedDate;

}

public void setLastModifiedDate(Instant lastModifiedDate) {

this.lastModifiedDate = lastModifiedDate;

}

}

#### 2.2 Update Employee and Department Entities

Extend your Employee and Department entities from the Auditable class. Add @CreatedBy and @LastModifiedBy annotations to track who created or modified the records.

**Employee.java:**

package com.example.employeemanagementsystem.model;

import lombok.Data;

import lombok.NoArgsConstructor;

import javax.persistence.\*;

@Entity

@Table(name = "employees")

@Data

@NoArgsConstructor

public class Employee extends Auditable {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private String email;

@ManyToOne

@JoinColumn(name = "department\_id", nullable = false)

private Department department;

// Other fields and methods

}

**Department.java:**

package com.example.employeemanagementsystem.model;

import lombok.Data;

import lombok.NoArgsConstructor;

import javax.persistence.\*;

@Entity

@Table(name = "departments")

@Data

@NoArgsConstructor

public class Department extends Auditable {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

// Other fields and methods

}

### 3. Implement Auditing of Users (Optional)

If you want to track who created or modified an entity, you need to configure Spring Security and set up an auditing principal resolver.

#### 3.1 Configure Auditing User

Create a custom auditor aware class to provide the currently logged-in user.

**AuditorAwareImpl.java:**

package com.example.employeemanagementsystem.config;

import org.springframework.data.domain.AuditorAware;

import org.springframework.security.core.context.SecurityContextHolder;

import org.springframework.stereotype.Component;

import java.util.Optional;

@Component

public class AuditorAwareImpl implements AuditorAware<String> {

@Override

public Optional<String> getCurrentAuditor() {

// Fetch the current authenticated user from the SecurityContext

return Optional.ofNullable(SecurityContextHolder.getContext().getAuthentication())

.map(auth -> auth.getName());

}

}

#### 3.2 Configure the AuditorAware Bean

Register the AuditorAware implementation in your configuration.

**JpaConfig.java:**

package com.example.employeemanagementsystem.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

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

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

@Configuration

@EnableJpaRepositories(basePackages = "com.example.employeemanagementsystem.repository")

@EnableJpaAuditing(auditorAwareRef = "auditorAware")

public class JpaConfig {

@Bean

public AuditorAware<String> auditorAware() {

return new AuditorAwareImpl();

}

}