4. **Problem Statement: Employee Attendance Tracker**
   Develop an application to track employee attendance within an organization. The system should allow employees to mark attendance and managers to view attendance reports.

   Answer :

```
package com.example.attendance;


import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.format.annotation.DateTimeFormat;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.core.userdetails.User;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.provisioning.InMemoryUserDetailsManager;

import org.springframework.security.web.SecurityFilterChain;

import org.springframework.stereotype.Repository;

import org.springframework.web.bind.annotation.*;


import javax.persistence.*;

import javax.validation.constraints.NotBlank;

import java.time.LocalDate;

import java.util.List;


@SpringBootApplication
```

```
public class AttendanceApplication {
    public static void main(String[] args) {
        SpringApplication.run(AttendanceApplication.class, args);
    }
}

@Entity
class Employee {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long employeeId;

    @NotBlank
    private String name;

    private String department;
    private String designation;

    // Getters and Setters
}

@Entity
class Attendance {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

```
    private Long attendanceId;

    @ManyToOne
    @JoinColumn(name = "employee_id")
    private Employee employee;

    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private LocalDate date;

    private String status; // Present or Absent

    // Getters and Setters
}

@Repository
interface EmployeeRepository extends JpaRepository<Employee, Long> {}

@Repository
interface AttendanceRepository extends JpaRepository<Attendance, Long> {
    List<Attendance> findByEmployeeEmployeeId(Long employeeId);

    @Query("SELECT a FROM Attendance a WHERE a.employee.department = :department")
    List<Attendance> findByDepartment(String department);
}
```

```
@RestController
@RequestMapping("/api")
class AttendanceController {

    private final EmployeeRepository employeeRepo;
    private final AttendanceRepository attendanceRepo;

    public AttendanceController(EmployeeRepository employeeRepo, AttendanceRepository
attendanceRepo) {
        this.employeeRepo = employeeRepo;
        this.attendanceRepo = attendanceRepo;
    }

    @PostMapping("/employees")
    public Employee addEmployee(@RequestBody Employee employee) {
        return employeeRepo.save(employee);
    }

    @PostMapping("/attendance")
    public Attendance markAttendance(@RequestBody Attendance attendance) {
        return attendanceRepo.save(attendance);
    }

    @GetMapping("/attendance/employee/{id}")
    public List<Attendance> getAttendanceByEmployee(@PathVariable Long id) {
        return attendanceRepo.findByEmployeeEmployeeId(id);
```

```
    }

    @GetMapping("/attendance/department/{dept}")
    public List<Attendance> getAttendanceByDepartment(@PathVariable String dept) {
        return attendanceRepo.findByDepartment(dept);
    }
}

@EnableWebSecurity
class SecurityConfig {
    @Bean
    public InMemoryUserDetailsManager userDetailsService() {
        UserDetails employee = User.withDefaultPasswordEncoder()
                .username("employee")
                .password("pass")
                .roles("EMPLOYEE")
                .build();
        UserDetails manager = User.withDefaultPasswordEncoder()
                .username("manager")
                .password("pass")
                .roles("MANAGER")
                .build();
        return new InMemoryUserDetailsManager(employee, manager);
    }
```

```
    @Bean
    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
        http.csrf().disable()
            .authorizeRequests()
            .antMatchers("/api/attendance").hasRole("EMPLOYEE")
            .antMatchers("/api/attendance/**").hasRole("MANAGER")
            .antMatchers("/api/employees/**").hasRole("MANAGER")
            .and().httpBasic();
        return http.build();
    }
}
```