Spring Boot and React Practice Exam

Spring Boot Tasks

Task 1: Spring Boot Application Setup with Oracle Database

Objective: Create a new Spring Boot application configured to connect to an Oracle database.

Requirements:

1. Use Spring Initializr (https://start.spring.io/) to generate a Spring Boot project with dependencies for Spring Web, Spring Data JPA, and Oracle JDBC Driver.
2. Configure `application.properties` to connect to your Oracle database. Provide placeholders for the database URL, username, and password.
3. Demonstrate a successful connection to the database by running the application.

Task 2: Implementing a RESTful Controller for Employee Management

Objective: Develop a RESTful controller in Spring Boot to manage employee entities.

Requirements:

1. Define an `Employee` entity with at least `id`, `name`, and `department` fields.
2. Create an `EmployeeRepository` interface that extends `JpaRepository<Employee, Long>`.
3. Implement an `EmployeeController` class with CRUD operations:
    - `GET /employees` to list all employees.
    - `POST /employees` to create a new employee.
    - `PUT /employees/{id}` to update an existing employee.
    - `DELETE /employees/{id}` to delete an employee.

React Tasks

Task 3: React Components for Employee Management

Objective: Design and describe React components for an employee management application.

Requirements:

1. Sketch or describe the component hierarchy for the application, including components for displaying a list of employees, individual employee details, and a filter for searching employees by name.
2. Detail the props and state each component will manage or receive.

Task 4: Implementing Employee Listing with Filters in React

Objective: Implement a filter feature in a React application for listing employees.

Requirements:

1. Create a `Filter` component that allows users to input a name to filter the list of employees.

2. In the `App` component, integrate the `Filter` component and implement logic to filter the displayed list of employees based on the user's input.

Caching Implementation Task for Spring Boot Application

Task: Implement Caching for Employee List Retrieval and Invalidate Cache on Modification

Objective: Enhance the employee management Spring Boot application by adding caching functionality to the employee list retrieval feature. Additionally, ensure the cache is invalidated appropriately whenever an employee record is updated or deleted, to maintain data consistency.

Requirements

1. **Caching Setup**:
   - Add caching support to your Spring Boot application by including the necessary cache dependencies.
   - Enable caching in your application by annotating your main application class with `@EnableCaching`.
2. **Employee List Caching**:
   - Implement caching for the method that retrieves the list of employees. Use the `@Cacheable` annotation with a unique cache name (e.g., `"employeesCache"`) to cache the method's result.
3. **Cache Invalidation on Employee Update/Delete**:
   - Ensure the cache is invalidated properly when an employee is updated or deleted. Use the `@CacheEvict` annotation on the corresponding update and delete methods, targeting the same cache used for the employee list. Specify `allEntries = true` to clear all entries from the cache upon any update or delete operation.

Oracle Database Task

Task 5: HR Schema Table Creation

Objective: Extend an Oracle Database HR schema by creating a new table.

Requirements:

1. Write an SQL statement to create a new table named `EMPLOYEES_EXTENDED` based on the existing `EMPLOYEES` table. Include all existing columns and add a new column `EMAIL_ADDRESS` of type VARCHAR2(100).