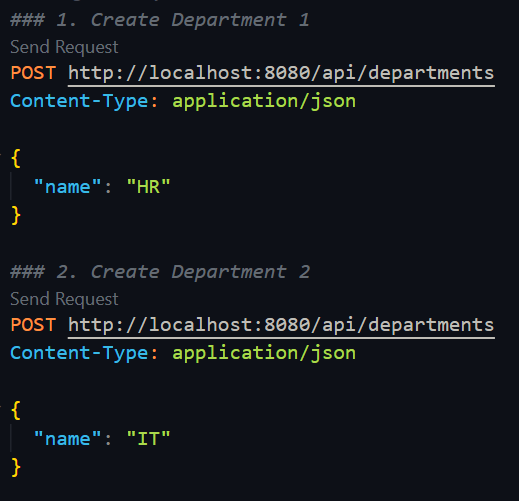
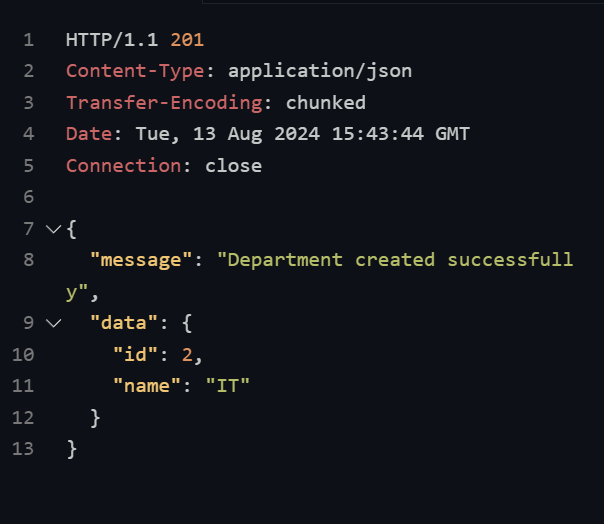
**Exercise 5: Employee Management System - Defining Query Methods**

* **Defining Query Methods:**
* **Keywords in Method Names:** You can use keywords in method names to define custom queries. For example, methods like *findByEmail*, *findByDepartmentName*, or *countByDepartmentId* can be created based on the keywords.
* **@Query Annotation:** This annotation allows you to define custom queries directly within the repository interface. For instance, using @Query("*SELECT e FROM Employee e WHERE e.email = :email*") allows retrieving an employee by their email.
* **Named Queries:**
* **@NamedQuery and @NamedQueries:** These annotations are used at the entity level to define reusable queries. For example, a named query Employee.findByLastName can be defined in the Employee entity to retrieve employees by their last name. The advantage is that the query is reusable across the application.
* **The Output of the code:** To execute all the test cases with the REST Client, you simply need to open the `test-request.http` file in your code editor (e.g., Visual Studio Code). Then, place your cursor on the desired HTTP request in the file and click on the "Send Request" button that appears above the line or use the keyboard shortcut (e.g., `Ctrl + Alt + R` on VS Code) to execute the request and view the response directly within the editor.

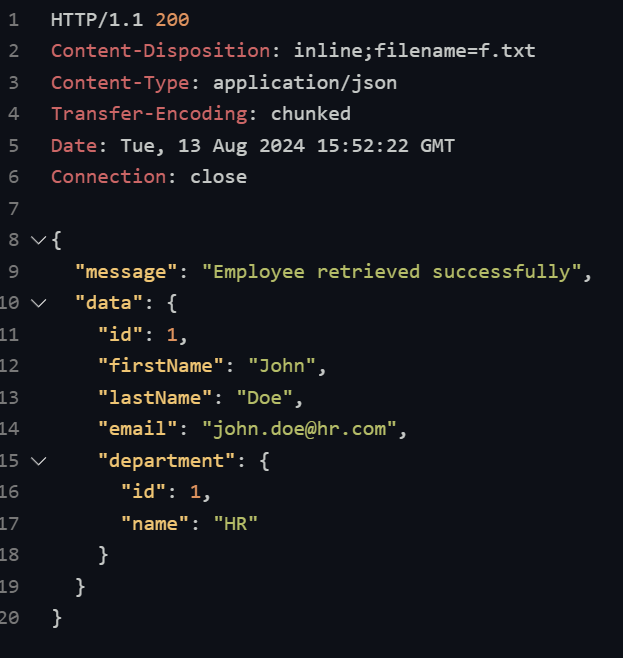
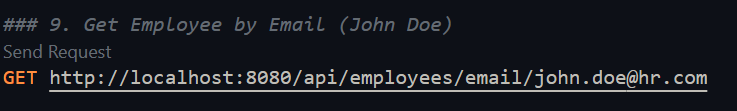
1. **Create Two Departments:** create two departments: **HR** and **Engineering**.

****



1. **Add Employees to Departments:** This step will add five employees in the 2 dept.

In this way add employees in the database

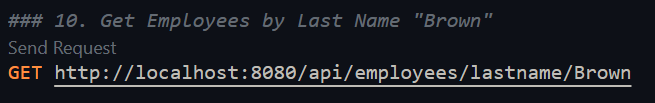
1. **Retrieve Employee by Email :** This step checks whether the query for retrieving

an employee by email is functioning correctly.

Execution code and the output response

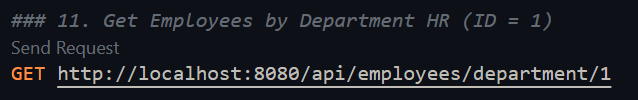
1. **Retrieve Employees by Last Name Using Named Query:** Fetch employees by their last name using the *NamedQuery*.

This step will verify that employees with the same last name can be retrieved.

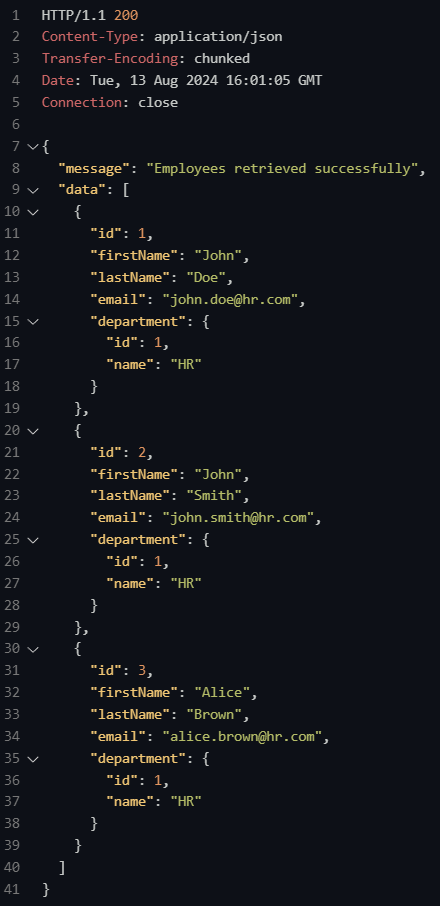
This is the execution code and the output response to Retrieve Employees by the last Name

There are Two persons with *Brown* Last name so it gives two employees response.

1. **Retrieve Employees by Department ID Using Named Query**: Fetch employees by department ID using the *NamedQuery.* This step will confirm that employees belonging to a specific department can be retrieved.

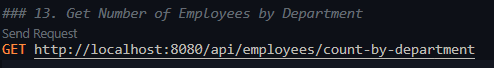


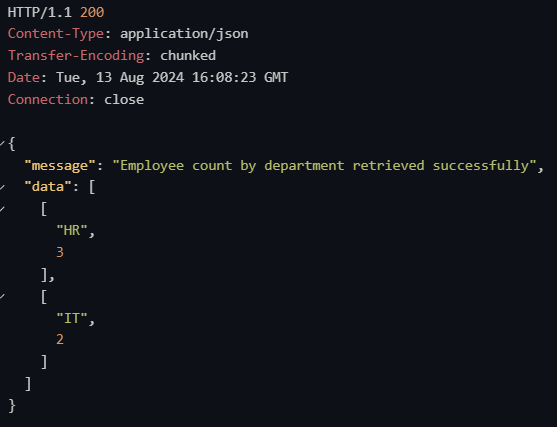
This will return the whole employee details which dept. Id is 1.

this is the whole employee details which are in the *HR* department

1. **Count Employees by Department:**

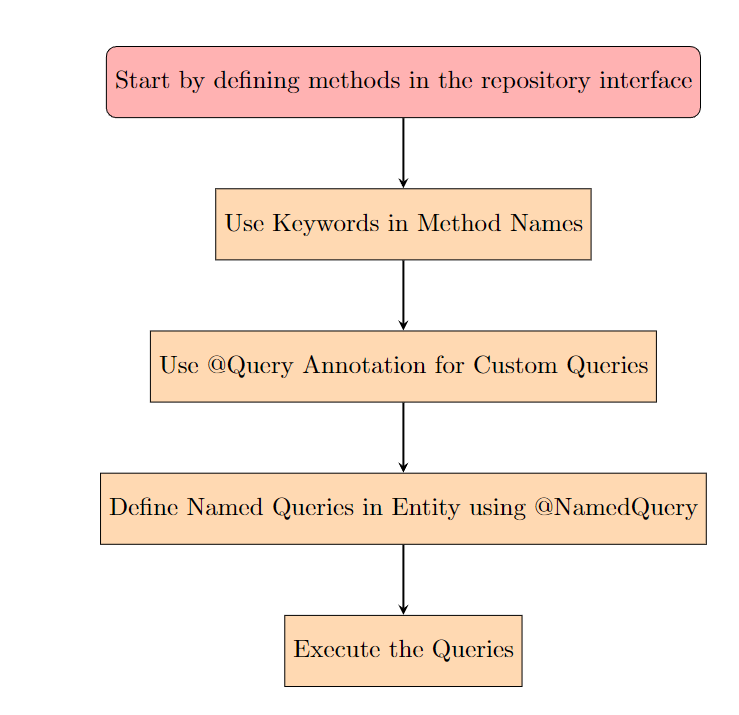
This step checks if the count query is working correctly to count the employees in the HR and Engineering departments.

 This is the Execution code and the output response



* **The Flow Chart of the Program :**

1. Start by defining methods in the repository interface.
2. Use keywords in method names for simple queries or use the ***@Query*** annotation for custom queries.
3. If you need reusable queries, define them in the entity using ***@NamedQuery.***
4. The process ensures your repository is enhanced with flexible querying capabilities.
5. Each method or query is tied back to the corresponding entity or method to retrieve specific data.

The GitHub link of the program - [link](https://github.com/Hyperstrom/Aniket-Pal_5017587/tree/main/WEEK-3/Exercise-5)