

Spring Data JPA with Hibernate Part 1

(1) Create an Employee Entity which contains following fields

Name

Id

Age

Location

```
@Data
@AllArgsConstructor
@NoArgsConstructor
@Entity
@Table(name = "employee")
public class Employee {
    @Id
    @GeneratedValue
    private int id;
    private String name;
    private int age;
    private String location;
}
```

(2) Set up EmployeeRepository with Spring Data JPA

```
1 package com.ttn.bootcamp.springDataJPA.employee.respository;
2
3 import com.ttn.bootcamp.springDataJPA.employee.entities.Employee;
4 import org.springframework.data.repository.CrudRepository;
5
6 public interface EmployeeRepository extends CrudRepository<Employee, Integer> {
7
8
9 }
```

(3) Perform Create Operation on Entity using Spring Data JPA

```
@Override
public Employee save(Employee employee) {
    return employeeRepository.save(employee);
}

@Override
public List<Employee> saveList(List<Employee> employees) {
    return (List<Employee>) employeeRepository.saveAll(employees);
}
```

```
@PostMapping(path = "/add")
public Employee add(@RequestBody Employee employee) {
    return employeeService.save(employee);
}

@PostMapping(path = "/addList")
public List<Employee> addList(@RequestBody List<Employee> employees) {
    return employeeService.saveList(employees);
}
```

Added one employee details

The screenshot shows a REST client interface with a POST request to `http://localhost:9191/employee/add`. The request body is a JSON object representing an employee: `{ "name": "Alex", "age": 25, "location": "London" }`. The response status is 200 OK. The response body is a JSON object: `{ "id": 2, "name": "Alex", "age": 25, "location": "London" }`.

```
POST http://localhost:9191/employee/add
```

Authorization Headers (1) Body Pre-request Script Tests

form-data x-www-form-urlencoded raw binary JSON (application/json)

```
1 {
2   "name": "Alex",
3   "age": 25,
4   "location": "London"
5 }
6
```

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview JSON

```
1 {
2   "id": 2,
3   "name": "Alex",
4   "age": 25,
5   "location": "London"
6 }
```

Added list of employees details

The screenshot shows a REST client interface with a POST request to `http://localhost:9191/employee/addList`. The request body is a JSON array of two employee objects: `[{ "name": "Ben", "age": 34, "location": "New York" }, { "name": "CaLay", "age": 27, "location": "London" }]`. The response status is 200 OK. The response body is a JSON array of two employee objects with their IDs: `[{ "id": 3, "name": "Ben", "age": 34, "location": "New York" }, { "id": 4, "name": "CaLay", "age": 27, "location": "London" }]`.

```
POST http://localhost:9191/employee/addList
```

form-data x-www-form-urlencoded raw binary JSON (application/json)

```
1 [
2   {
3     "name": "Ben",
4     "age": 34,
5     "location": "New York"
6   },
7   {
8     "name": "CaLay",
9     "age": 27,
10    "location": "London"
11  }
12 ]
13
14
```

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview JSON

```
1 [
2   {
3     "id": 3,
4     "name": "Ben",
5     "age": 34,
6     "location": "New York"
7   },
8   {
9     "id": 4,
10    "name": "CaLay",
11    "age": 27,
12    "location": "London"
13  }
14 ]
```

(4) Perform Update Operation on Entity using Spring Data JPA

```
@Override
public Employee update(Employee employee) {
    Employee existingEmployee = employeeRepository.findById(employee.getId()).orElse( other: null);
    existingEmployee.setName(employee.getName());
    existingEmployee.setAge(employee.getAge());
    existingEmployee.setLocation(employee.getLocation());
    return employeeRepository.save(existingEmployee);
}
```

Updating employee with id 1

GET Params

Authorization Headers (1) Body Pre-request Script Tests

Type

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview JSON

```
1- [
2-   {
3-     "id": 1,
4-     "name": "baban",
5-     "age": 25,
6-     "location": "Delhi"
7-   },
8- ]
```

Updated by put request

PUT Params

Authorization Headers (1) Body Pre-request Script Tests

☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary JSON (application/json)

```
1- [
2-   {
3-     "id": 1,
4-     "name": "John",
5-     "age": 34,
6-     "location": "Delhi"
7-   },
8- ]
```

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview JSON

```
1- [
2-   {
3-     "id": 1,
4-     "name": "John",
5-     "age": 34,
6-     "location": "Delhi"
7-   },
8- ]
```

GET ▼ http://localhost:9191/employee Params Send ▼

Authorization Headers (1) Body Pre-request Script Tests

Type No Auth ▼

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview JSON ≡

```

1 {
2   {
3     "id": 1,
4     "name": "John",
5     "age": 34,
6     "location": "Delhi"
7   },
8   {
9     "id": 2,
10    "name": "Alex",
11    "age": 25,
12    "location": "London"
13  },
14  {
15    "id": 3,
16    "name": "Ben",
17    "age": 34,
18    "location": "New York"
19  },
20  {
21    "id": 4,
22    "name": "Calay",
23    "age": 27,
24    "location": "London"
25  }
26 }

```

(5) Perform Delete Operation on Entity using Spring Data JPA

```

@Override
public String delete(Integer id) {
    employeeRepository.deleteById(id);
    return "Employee removed" + id;
}

```

DELETE ▼ http://localhost:9191/employee/delete/1 Params Send ▼

Key	Value
New key	Value

Authorization Headers (1) Body ● Pre-request Script Tests

Type No Auth ▼

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview Text ≡

```

1 Employee removed1

```

(5) Perform Read Operation on Entity using Spring Data JPA

```
@Override
public List<Employee> findAll() {
    return (List<Employee>) employeeRepository.findAll();
}

@Override
public Optional<Employee> findById(Integer id) {
    return employeeRepository.findById(id);
}
```

The screenshot shows a REST client interface. At the top, a GET request is configured for the URL `http://localhost:9191/employee/get/2`. Below the request bar, the 'Body' tab is selected, displaying a JSON response in 'Pretty' format. The response is a JSON object with the following fields: `id` (2), `name` ("Alex"), `age` (25), and `location` ("London"). The status bar at the bottom right indicates a successful response with status 200 OK.

```
1 {
2   "id": 2,
3   "name": "Alex",
4   "age": 25,
5   "location": "London"
6 }
```

(6) Get the total count of the number of Employees

```
@Override
public String count() {
    int count= (int) employeeRepository.count();
    return "Total count of Employee is "+count;
}
```

http http http http http http http http http http http New Tab http http No Environment

GET http://localhost:9191/employee/count Params Send

Authorization Headers (1) Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview Text

```
1 Total count of Employee is 3
```

(7) Implement Pagination and Sorting on the bases of Employee Age

```
public List<Employee> getPageSortedOnAge(Integer pageNo, Integer pageSize, String sortBy) {
    Pageable paging = PageRequest.of(pageNo, pageSize, Sort.by(Sort.Order.asc(sortBy)));
    Page<Employee> pagedResult = employeePageRepository.findAll(paging);

    if (pagedResult.hasContent()) {
        return pagedResult.getContent();
    } else {
        return new ArrayList<Employee>();
    }
}
```

GET http://localhost:9191/employee/sortedOnAge?pageNo=0&pageSize=5&age=age Params Send

Key	Value
<input checked="" type="checkbox"/> pageNo	0
<input checked="" type="checkbox"/> pageSize	5
<input checked="" type="checkbox"/> age	age
New key	Value

Authorization Headers (1) Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview JSON

```
1 {
2   {
3     "id": 7,
4     "name": "Anu",
5     "age": 23,
6     "location": "Uttarakhand"
7   },
8   {
9     "id": 2,
10    "name": "Alex",
11    "age": 25,
12    "location": "London"
13  },
14  {
15    "id": 28,
16    "name": "Alex",
17    "age": 25,
18    "location": "London"
19  },
20  {
21    "id": 5,
22    "name": "Alex",
23    "age": 25,
24    "location": "London"
25  },
26  {
27    "id": 8,
28    "name": "Panku",
29    "age": 25,
30    "location": "Rudraprayag"
31  }
32 }
```

8 Create and use finder to find Employee by Name

```
public List<Employee> findByName(String name) {  
    return employeeRepository.findByName(name);  
}
```

GET http://localhost:9191/employee/findByName?name=panku

Key	Value
name	panku

Authorization: No Auth

Body: {
 "id": 8,
 "name": "Panku",
 "age": 25,
 "location": "Rudraprayag"
}

Status: 200 OK

(9) Create and use finder to find Employees starting with A character

```
public List<Employee> findByNameLikeEmployee() {  
    return employeeRepository.findByNameLike("A%");  
}
```


GET ▼ http://localhost:9191/employee/like Params Send ▼

Key	Value
New key	Value

Authorization Headers (1) Body Pre-request Script Tests

Type No Auth ▼

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview JSON ▼ ≡

```

1 [
2   {
3     "id": 2,
4     "name": "Alex",
5     "age": 25,
6     "location": "London"
7   },
8   {
9     "id": 5,
10    "name": "Alex",
11    "age": 25,
12    "location": "London"
13  },
14  {
15    "id": 7,
16    "name": "Anu",
17    "age": 23,
18    "location": "Uttarakhand"
19  },
20  {
21    "id": 28,
22    "name": "Alex",
23    "age": 25,
24    "location": "London"
25  }
26 ]

```

(10) Create and use finder to find Employees Between the age of 28 to 32

```

public List<Employee> findByAgeBetweenEmployee(int age1, int age2) {
    return employeeRepository.findByAgeBetween(age1, age2);
}

```

GET ▼ http://localhost:9191/employee/findByAgeBetween?Age1=28&Age2=32 Params Send ▼ Save

Key	Value
<input checked="" type="checkbox"/> Age1	28
<input checked="" type="checkbox"/> Age2	32
New key	Value

Authorization Headers (1) Body Pre-request Script Tests

Type No Auth ▼

Body Cookies Headers (5) Test Results Status: 200 OK Time: 7

Pretty Raw Preview JSON ▼ ≡

```

1 [
2   {
3     "id": 9,
4     "name": "Park",
5     "age": 28,
6     "location": "New York"
7   },
8   {
9     "id": 30,
10    "name": "harry",
11    "age": 28,
12    "location": "uk"
13  }
14 ]

```