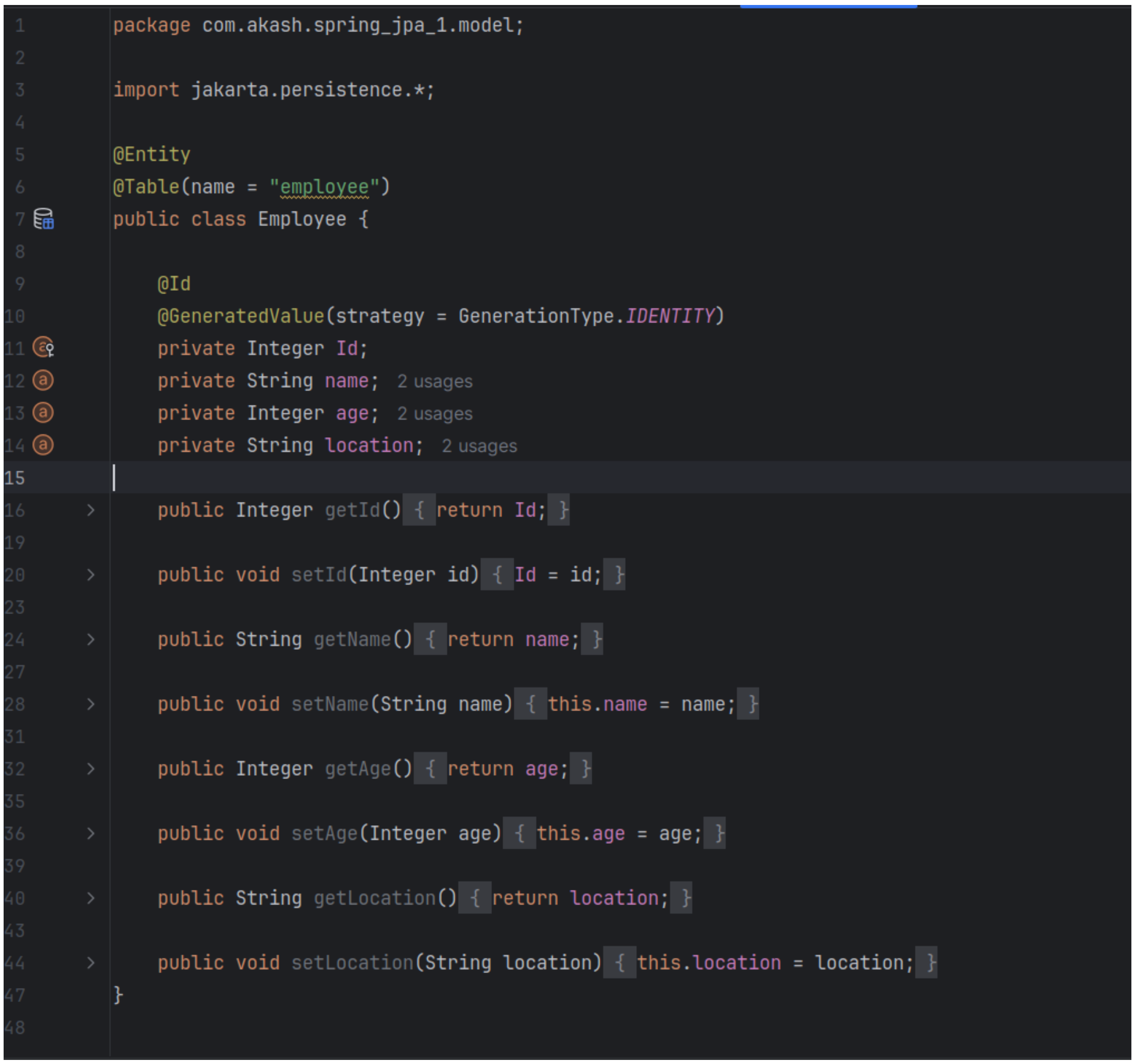
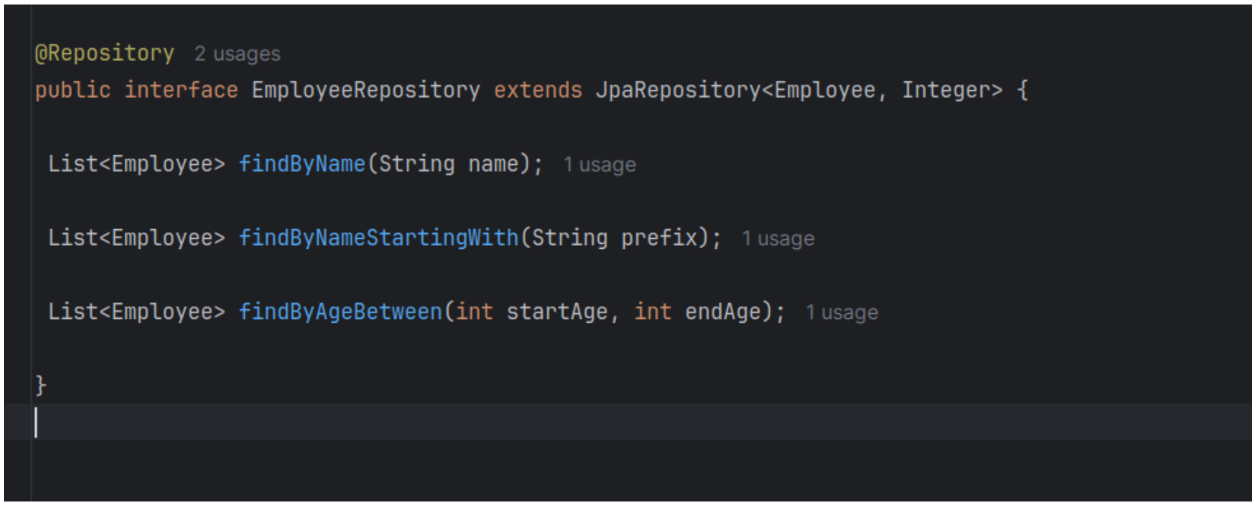
Assignment - Spring Data JPA - Part 1

Q1)Create an Employee Entity which contains the following fields: Name, Id, Age, Location



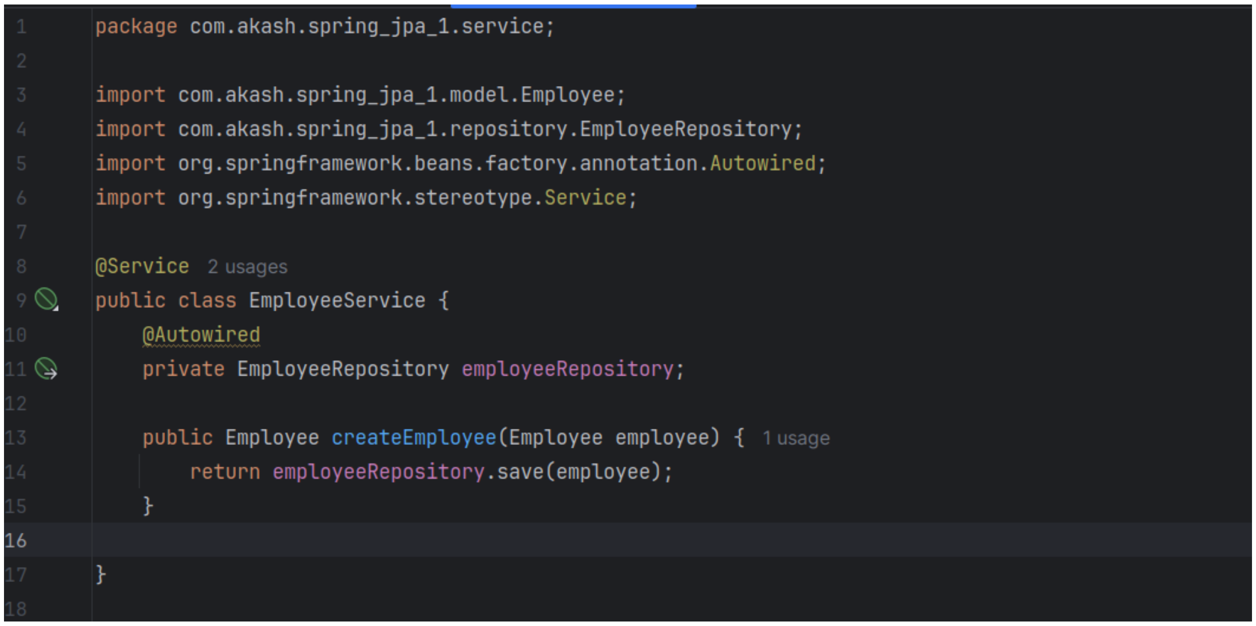
```java
package com.akash.spring_jpa_1.model;

import jakarta.persistence.*;

@Entity
@Table(name = "employee")
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer Id;
    private String name;  2 usages
    private Integer age;  2 usages
    private String location;  2 usages

    public Integer getId() { return Id; }

    public void setId(Integer id) { Id = id; }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public Integer getAge() { return age; }

    public void setAge(Integer age) { this.age = age; }

    public String getLocation() { return location; }

    public void setLocation(String location) { this.location = location; }
}
```

Q2)Set up EmployeeRepository with Spring Data JPA

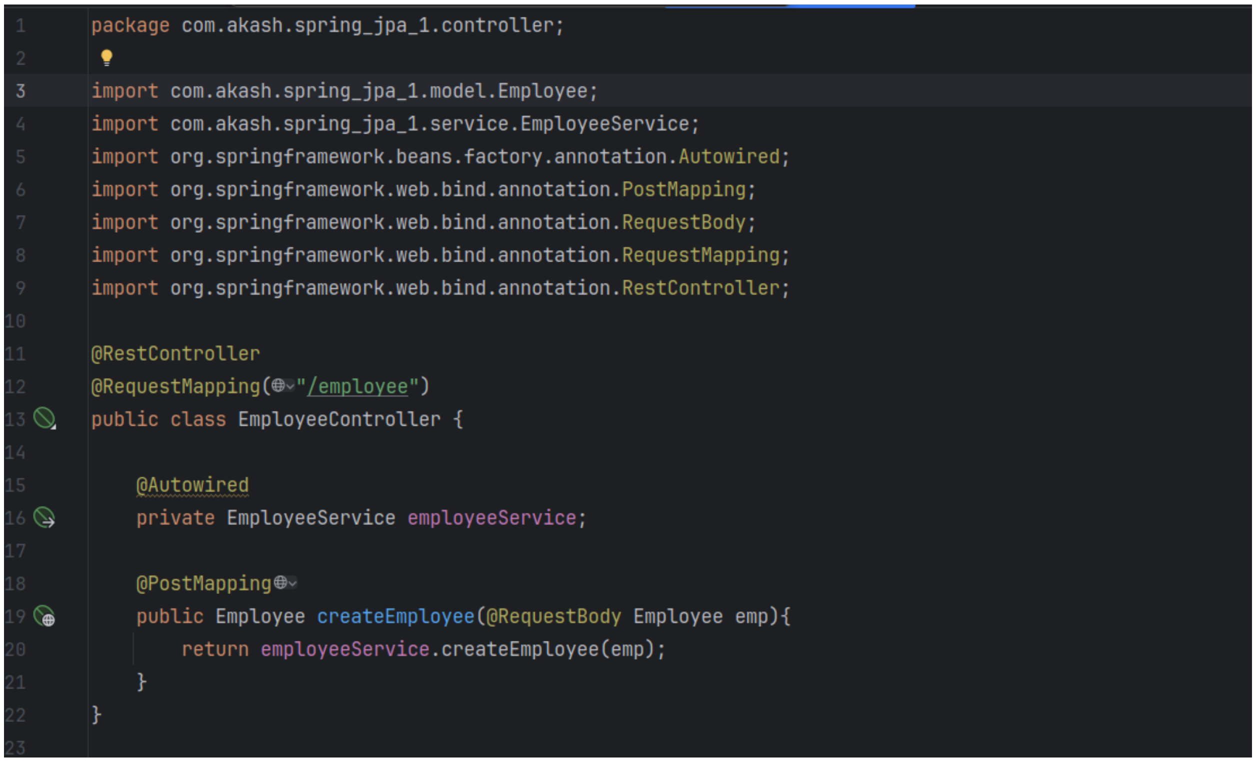


```java
@Repository  2 usages
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

 List<Employee> findByName(String name);  1 usage

 List<Employee> findByNameStartingWith(String prefix);  1 usage

 List<Employee> findByAgeBetween(int startAge, int endAge);  1 usage

}
```

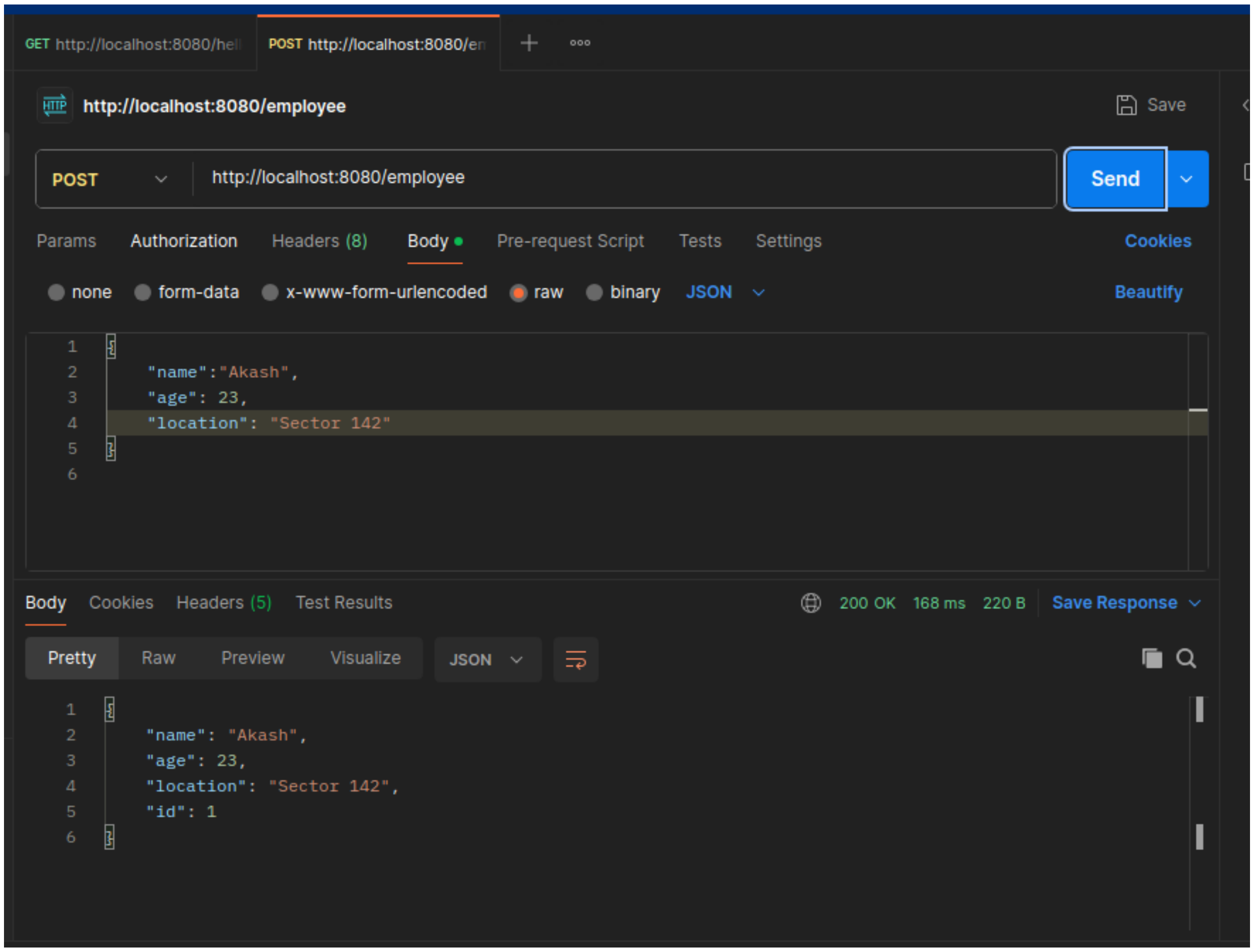
Q3)Perform Create Operation on Entity using Spring Data JPA



```java
package com.akash.spring_jpa_1.service;

import com.akash.spring_jpa_1.model.Employee;
import com.akash.spring_jpa_1.repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service  2 usages
public class EmployeeService {
    @Autowired
    private EmployeeRepository employeeRepository;

    public Employee createEmployee(Employee employee) {  1 usage
        return employeeRepository.save(employee);
    }

}
```



```java
package com.akash.spring_jpa_1.controller;

import com.akash.spring_jpa_1.model.Employee;
import com.akash.spring_jpa_1.service.EmployeeService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("/employee")
public class EmployeeController {

    @Autowired
    private EmployeeService employeeService;

    @PostMapping
    public Employee createEmployee(@RequestBody Employee emp){
        return employeeService.createEmployee(emp);
    }
}
```



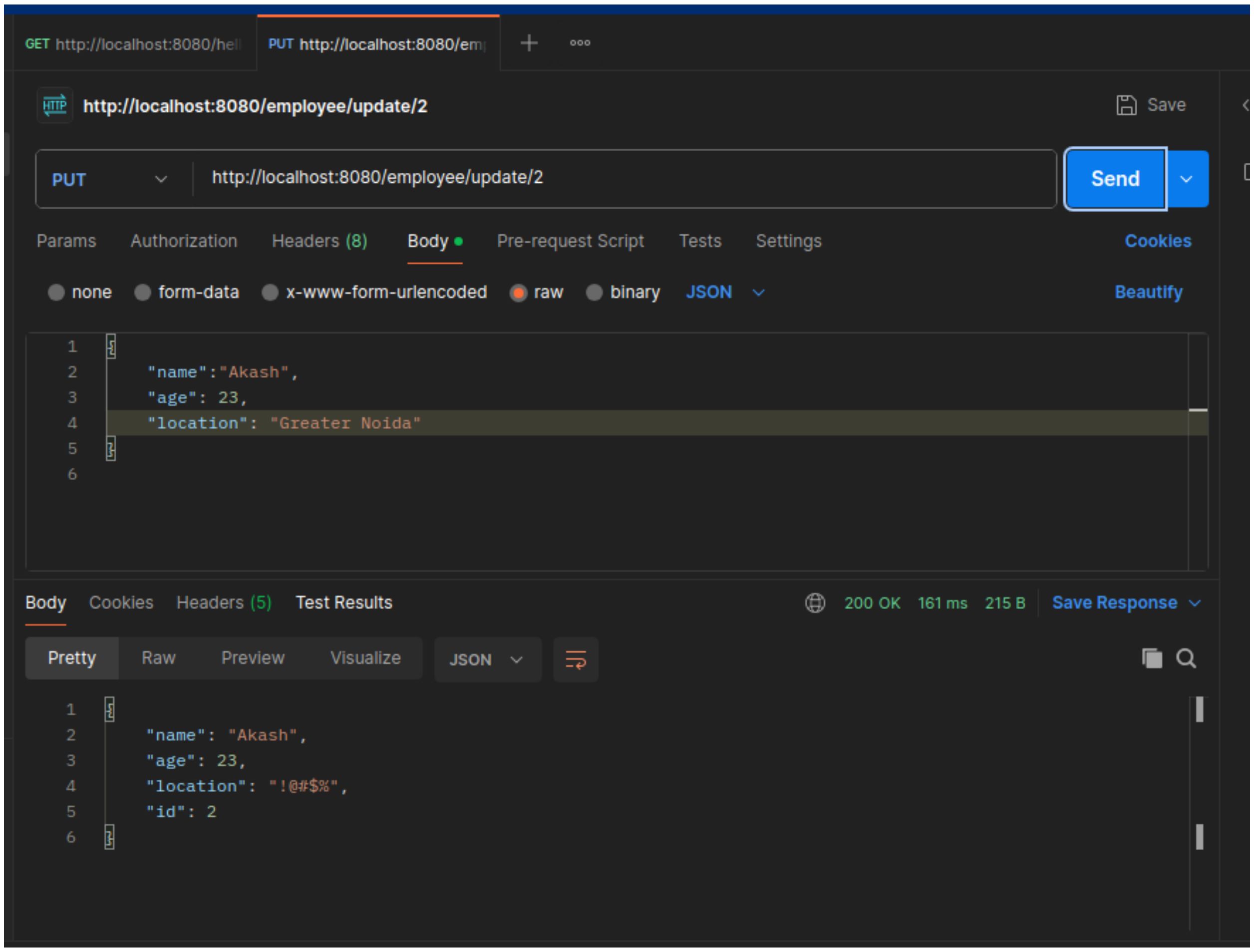
POST http://localhost:8080/employee

```
{
    "name":"Akash",
    "age": 23,
    "location": "Sector 142"
}
```

200 OK 168 ms 220 B

```
{
    "name": "Akash",
    "age": 23,
    "location": "Sector 142",
    "id": 1
}
```

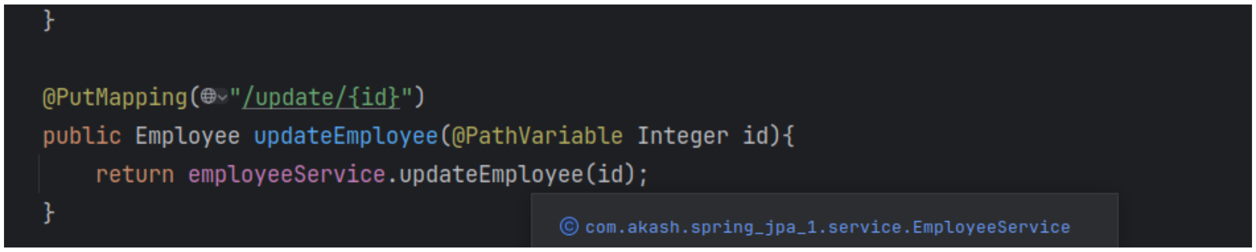
Q4)Perform Update Operation on Entity using Spring Data JPA

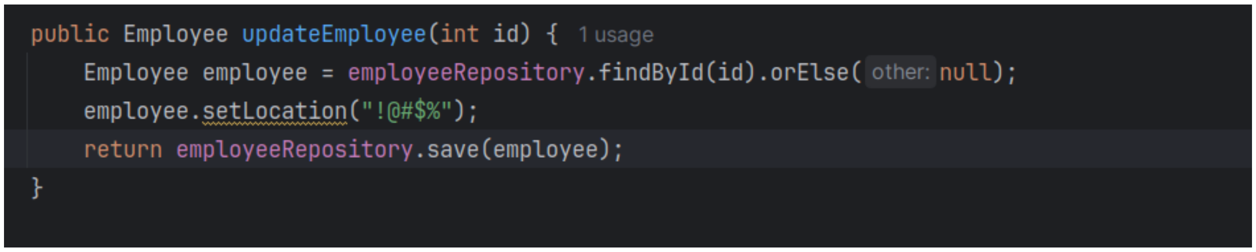

PUT http://localhost:8080/employee/update/2

```
{
    "name":"Akash",
    "age": 23,
    "location": "Greater Noida"
}
```

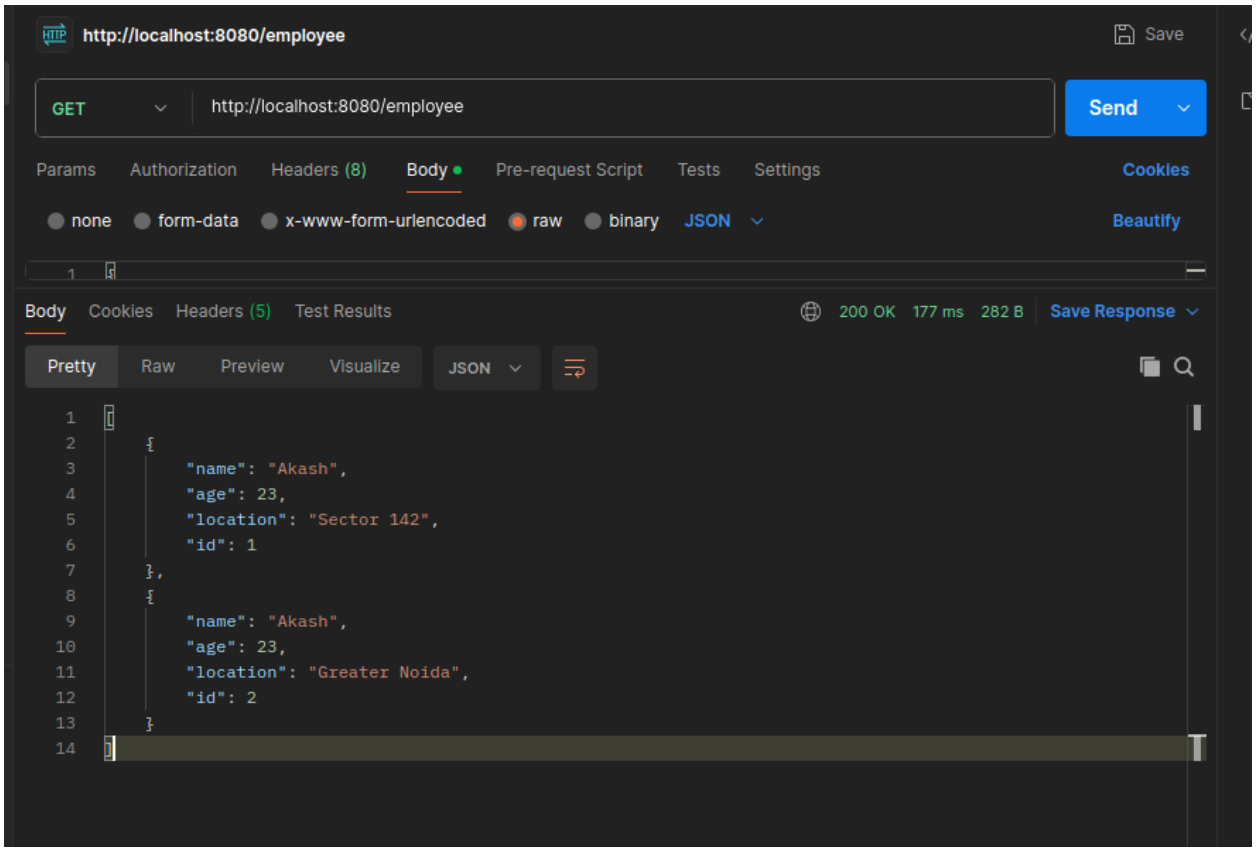
Response: 200 OK 161 ms 215 B

```
{
    "name": "Akash",
    "age": 23,
    "location": "!@#$%",
    "id": 2
}
```


```
}

@PutMapping("/update/{id}")
public Employee updateEmployee(@PathVariable Integer id){
    return employeeService.updateEmployee(id);
}
```

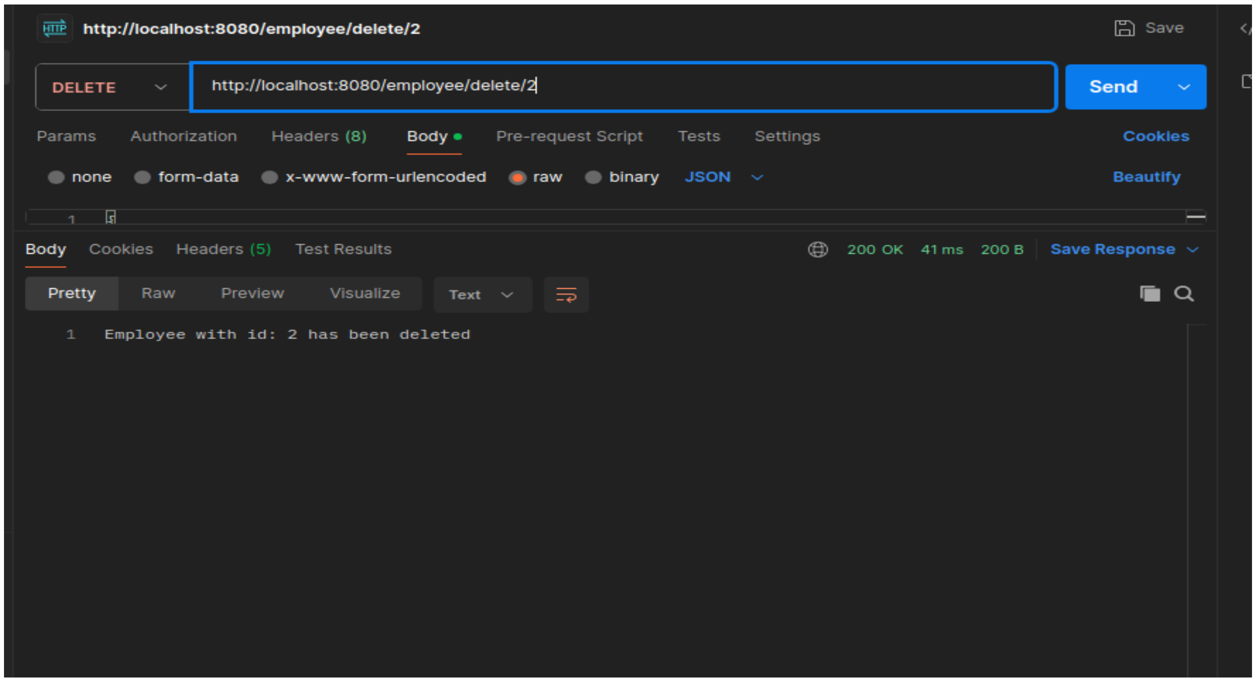

```
public Employee updateEmployee(int id) {  1 usage
    Employee employee = employeeRepository.findById(id).orElse( other: null);
    employee.setLocation("!@#$%");
    return employeeRepository.save(employee);
}
```

Q5)Perform Delete Operation on Entity using Spring Data JPA



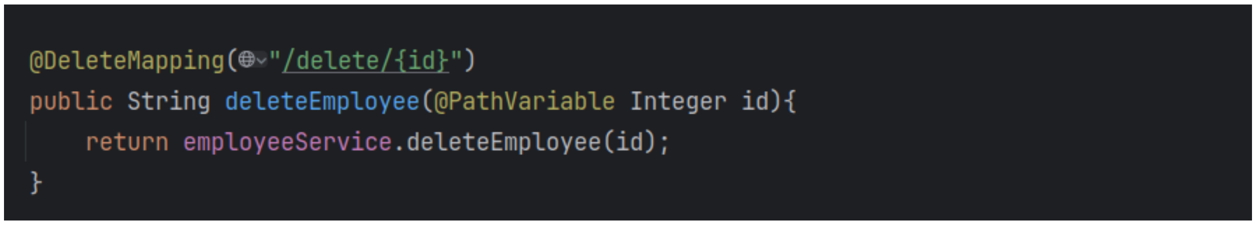
```
GET http://localhost:8080/employee
```

```
[
    {
        "name": "Akash",
        "age": 23,
        "location": "Sector 142",
        "id": 1
    },
    {
        "name": "Akash",
        "age": 23,
        "location": "Greater Noida",
        "id": 2
    }
]
```

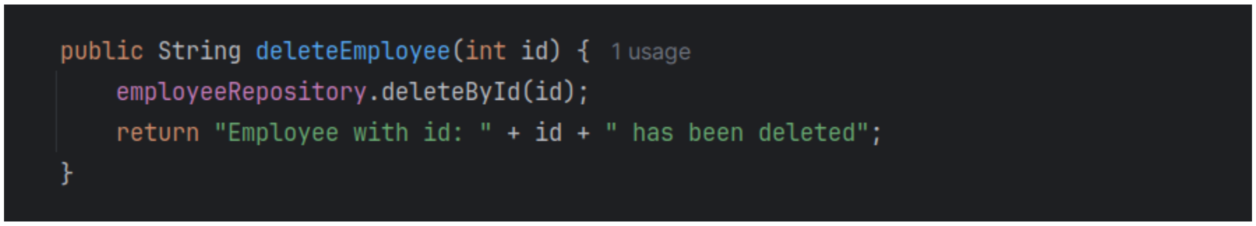


```
DELETE http://localhost:8080/employee/delete/2
```

```
Employee with id: 2 has been deleted
```

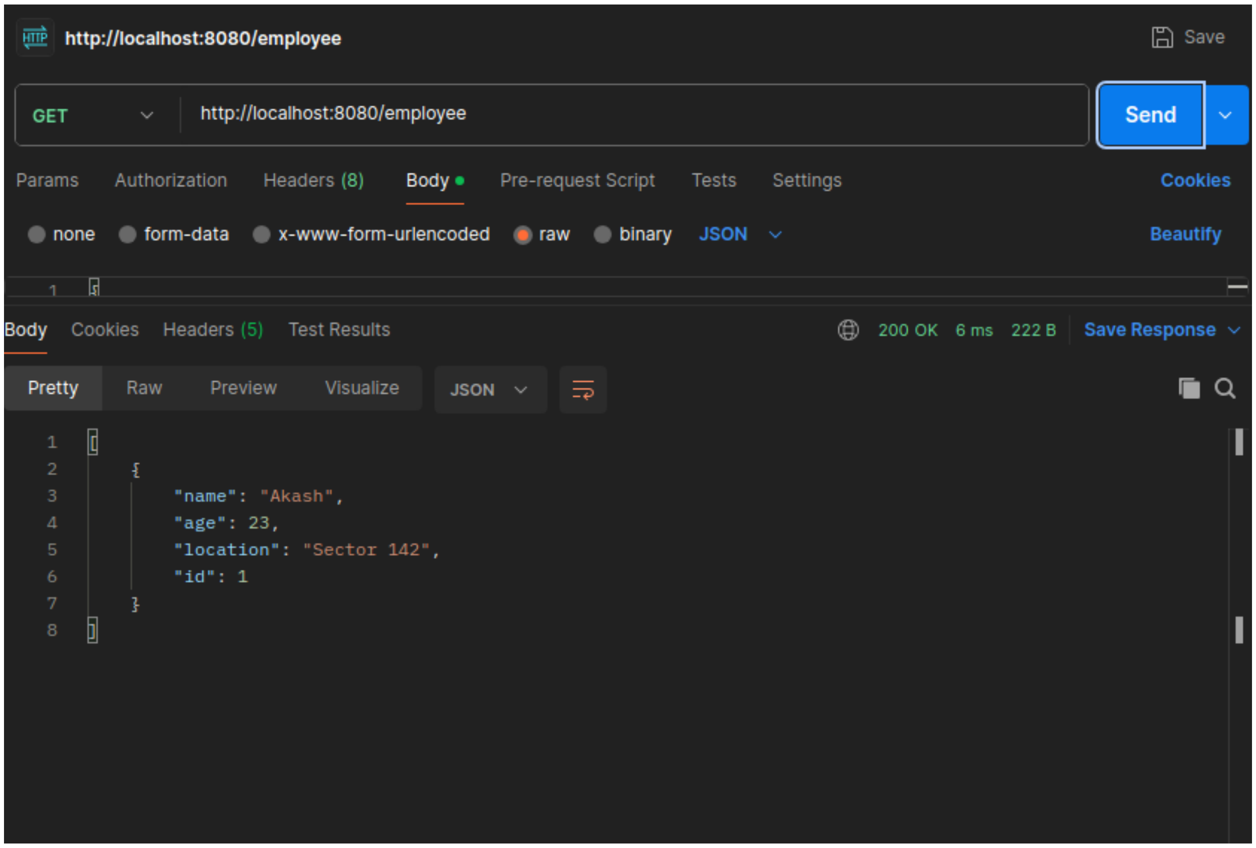


```java
@DeleteMapping("/delete/{id}")
public String deleteEmployee(@PathVariable Integer id){
    return employeeService.deleteEmployee(id);
}
```



```java
public String deleteEmployee(int id) {  1 usage
    employeeRepository.deleteById(id);
    return "Employee with id: " + id + " has been deleted";
}
```

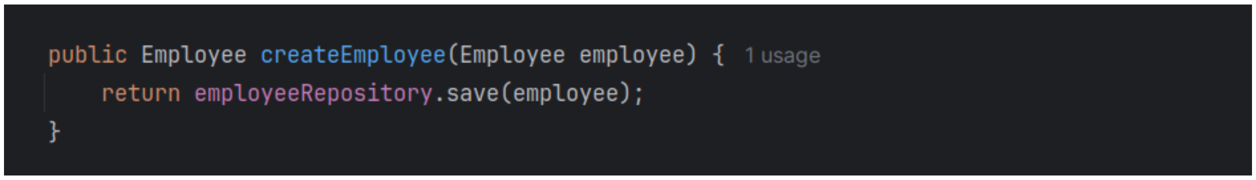
Q6)Perform Read Operation on Entity using Spring Data JPA



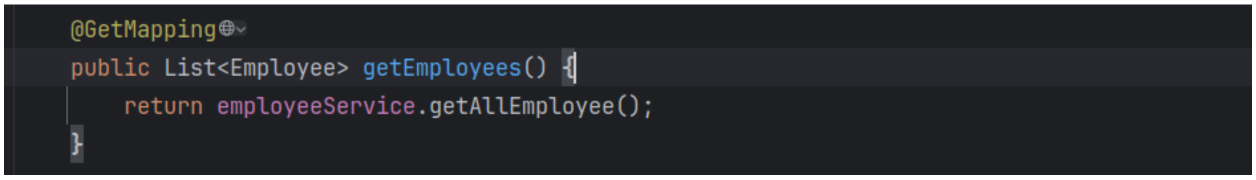
GET http://localhost:8080/employee

200 OK 6 ms 222 B

```
[
    {
        "name": "Akash",
        "age": 23,
        "location": "Sector 142",
        "id": 1
    }
]
```

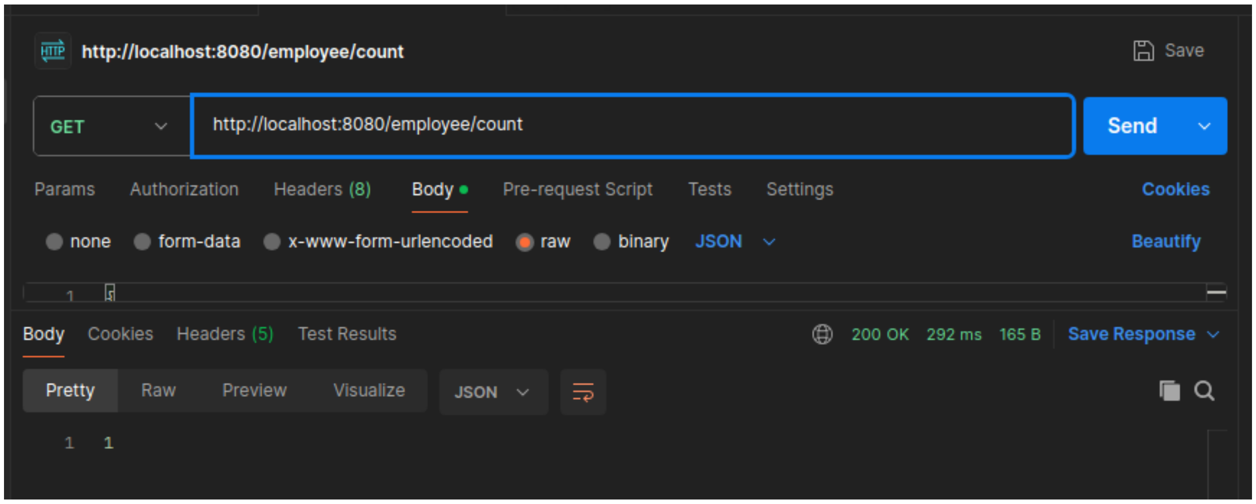


```
public Employee createEmployee(Employee employee) {  1 usage
    return employeeRepository.save(employee);
}
```



```
@GetMapping
public List<Employee> getEmployees() {
    return employeeService.getAllEmployee();
}
```

Q7)Get the total count of the number of Employees



http://localhost:8080/employee/count

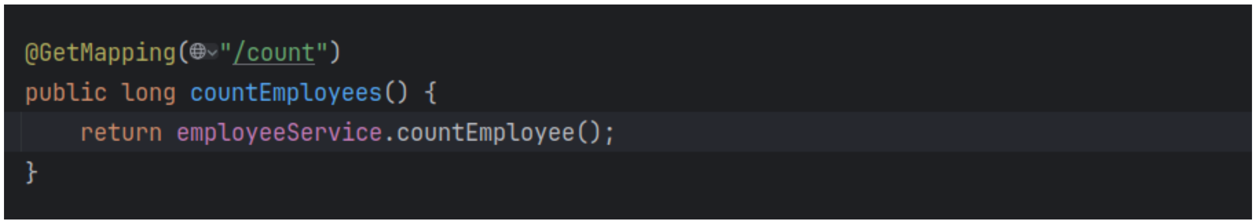
GET http://localhost:8080/employee/count

Body: none | form-data | x-www-form-urlencoded | [✓ raw] | binary — JSON

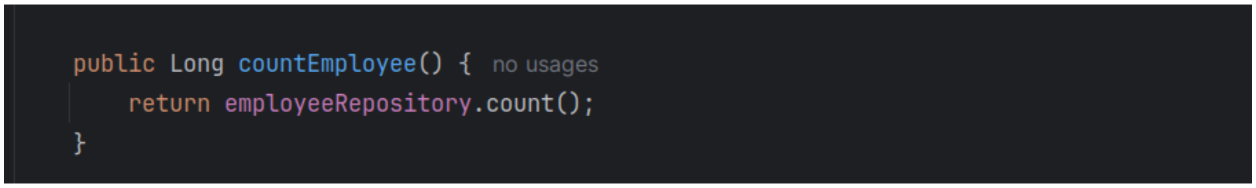
200 OK 292 ms 165 B

Body — Pretty — JSON

```
1
```

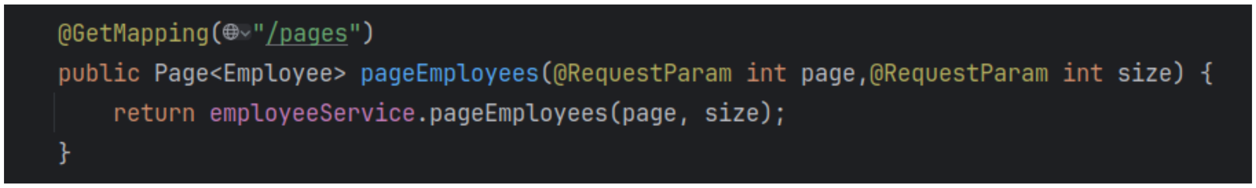


```
@GetMapping("/count")
public long countEmployees() {
    return employeeService.countEmployee();
}
```

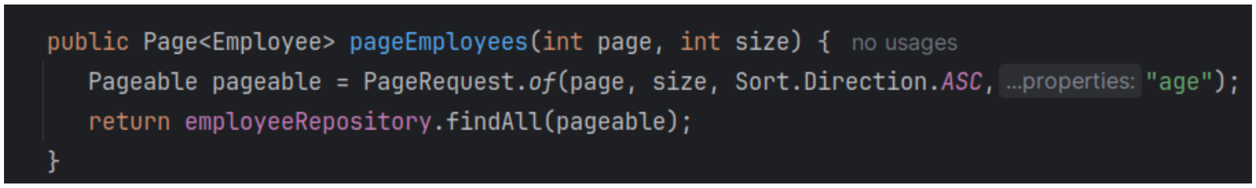


```
    public Long countEmployee() {  no usages
        return employeeRepository.count();
    }
```

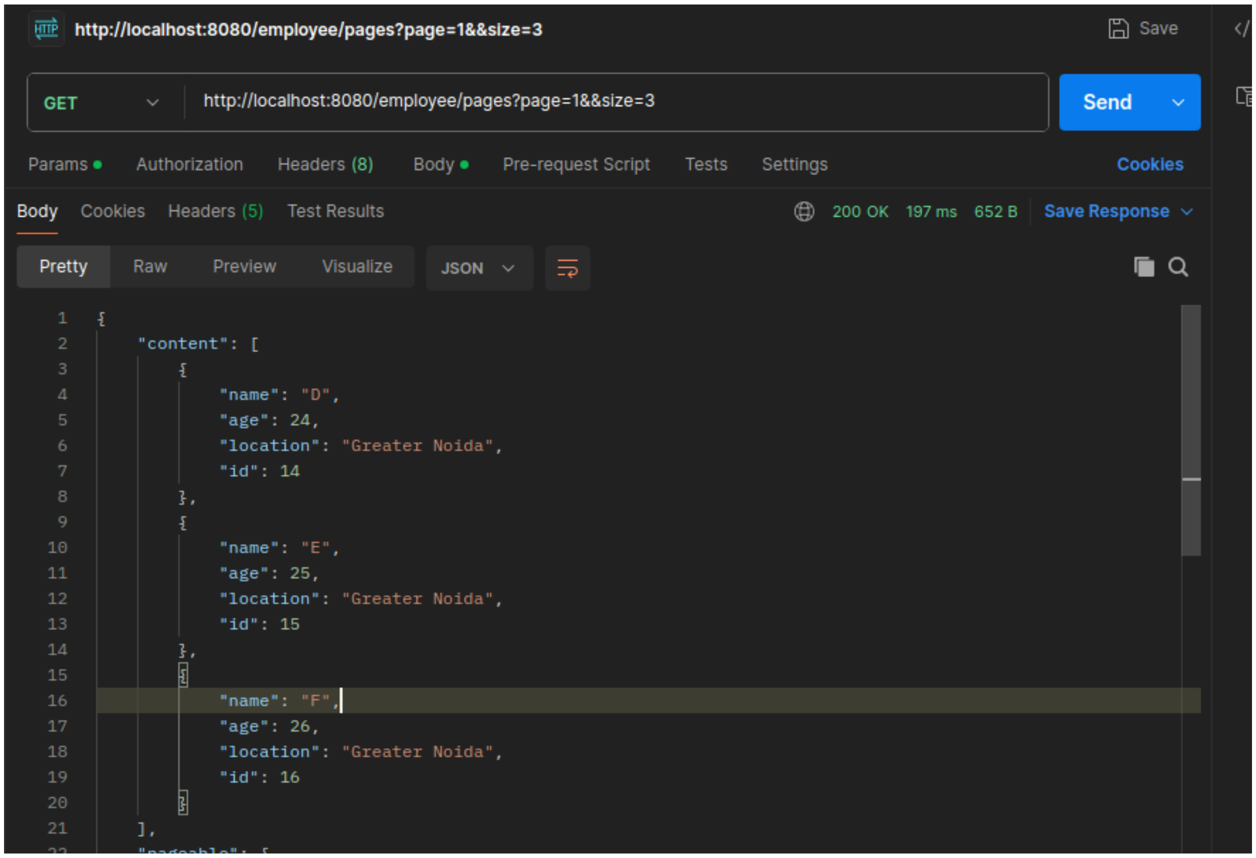
Q8)Implement Pagination and Sorting on the bases of Employee Age



```
@GetMapping("/pages")
public Page<Employee> pageEmployees(@RequestParam int page,@RequestParam int size) {
    return employeeService.pageEmployees(page, size);
}
```



```
public Page<Employee> pageEmployees(int page, int size) {  no usages
    Pageable pageable = PageRequest.of(page, size, Sort.Direction.ASC, ...properties: "age");
    return employeeRepository.findAll(pageable);
}
```

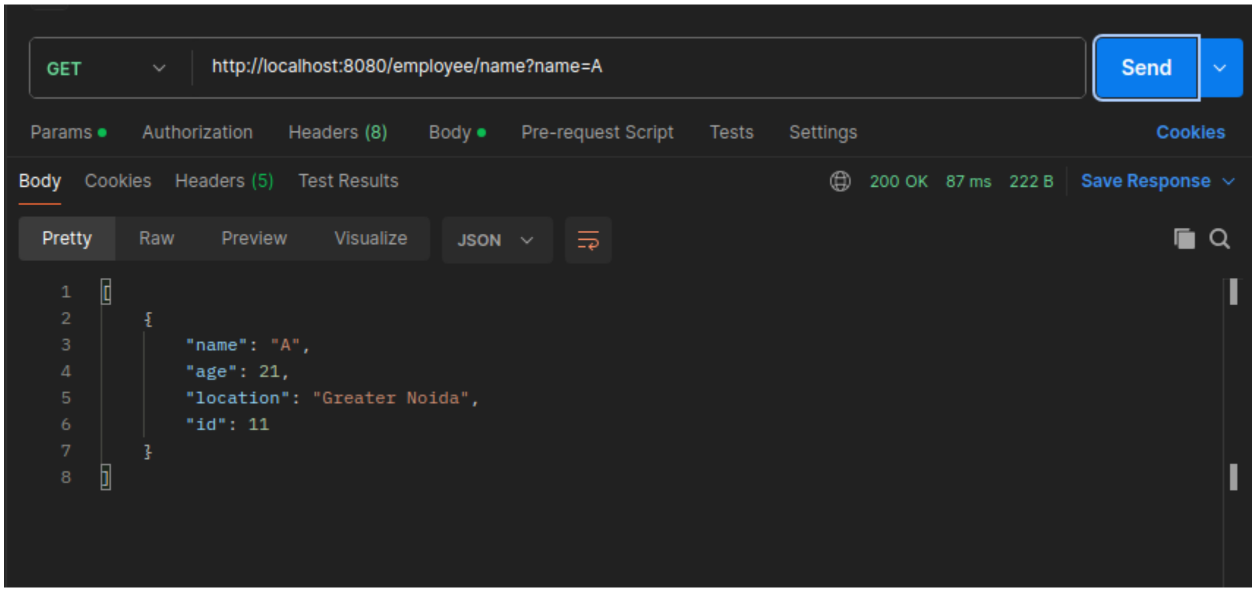


GET http://localhost:8080/employee/pages?page=1&&size=3

200 OK 197 ms 652 B

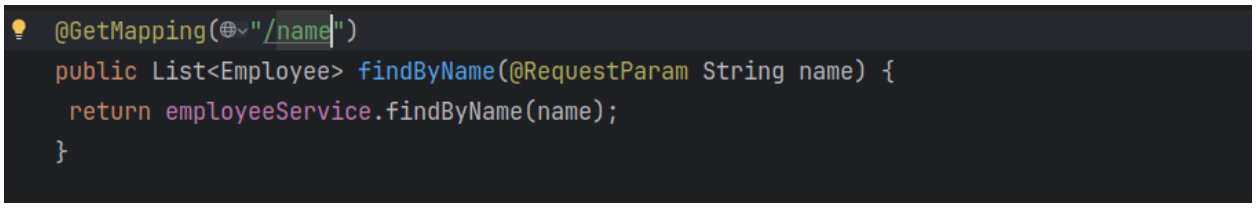
```
{
    "content": [
        {
            "name": "D",
            "age": 24,
            "location": "Greater Noida",
            "id": 14
        },
        {
            "name": "E",
            "age": 25,
            "location": "Greater Noida",
            "id": 15
        },
        {
            "name": "F",
            "age": 26,
            "location": "Greater Noida",
            "id": 16
        }
    ],
```

Q9)Create and use finder to find Employee by Name

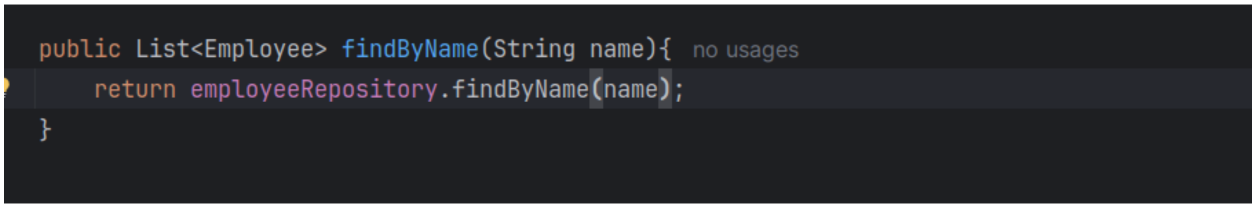


http://localhost:8080/employee/name?name=A

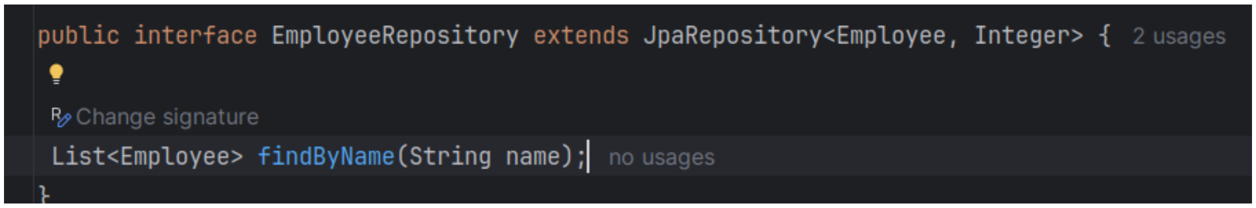
```
[
    {
        "name": "A",
        "age": 21,
        "location": "Greater Noida",
        "id": 11
    }
]
```



```
@GetMapping("/name")
public List<Employee> findByName(@RequestParam String name) {
 return employeeService.findByName(name);
}
```

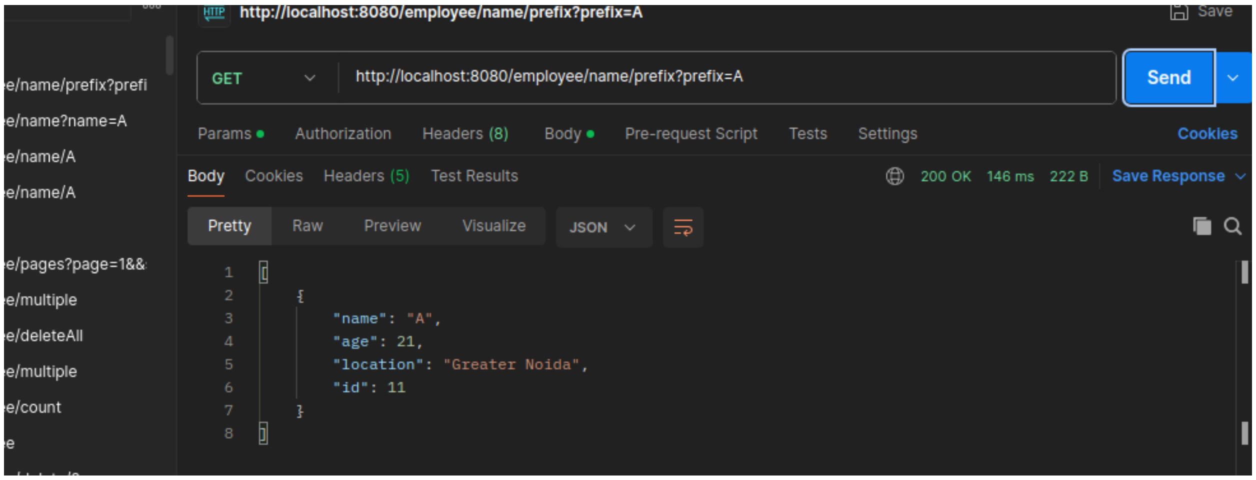


```
public List<Employee> findByName(String name){  no usages
    return employeeRepository.findByName(name);
}
```



```
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {  2 usages

 Change signature
 List<Employee> findByName(String name);  no usages
}
```

Q10)Create and use finder to find Employees starting with A character

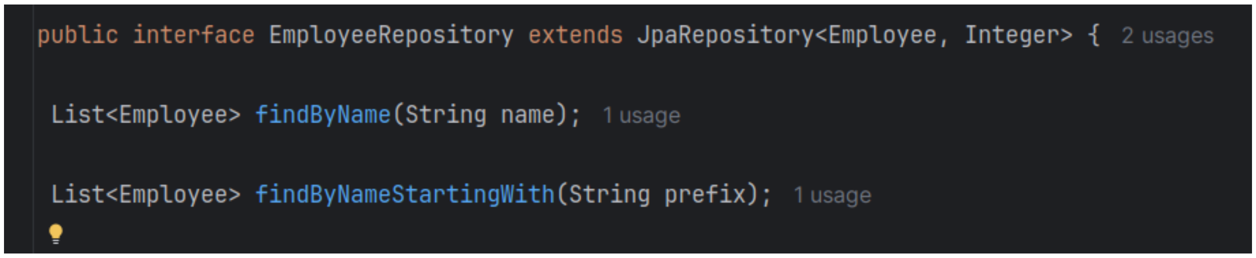


http://localhost:8080/employee/name/prefix?prefix=A

GET http://localhost:8080/employee/name/prefix?prefix=A

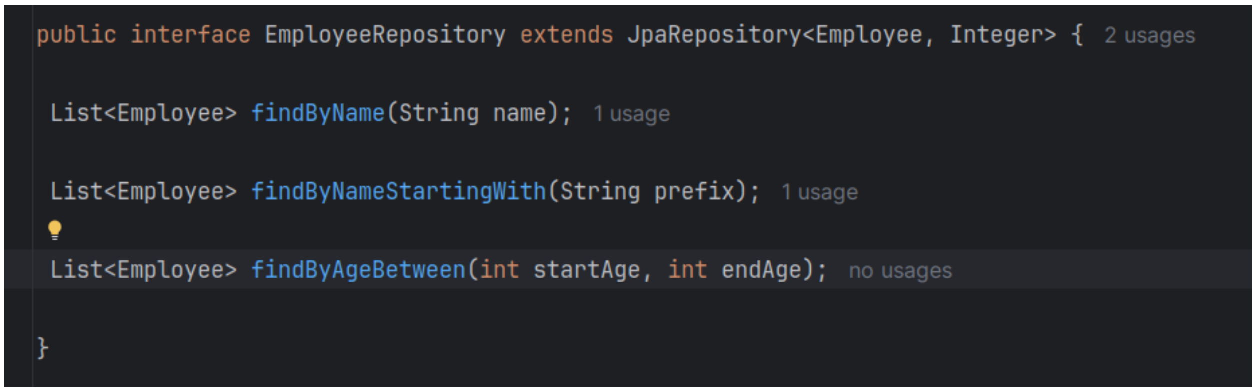
200 OK 146 ms 222 B

```
[
    {
        "name": "A",
        "age": 21,
        "location": "Greater Noida",
        "id": 11
    }
]
```

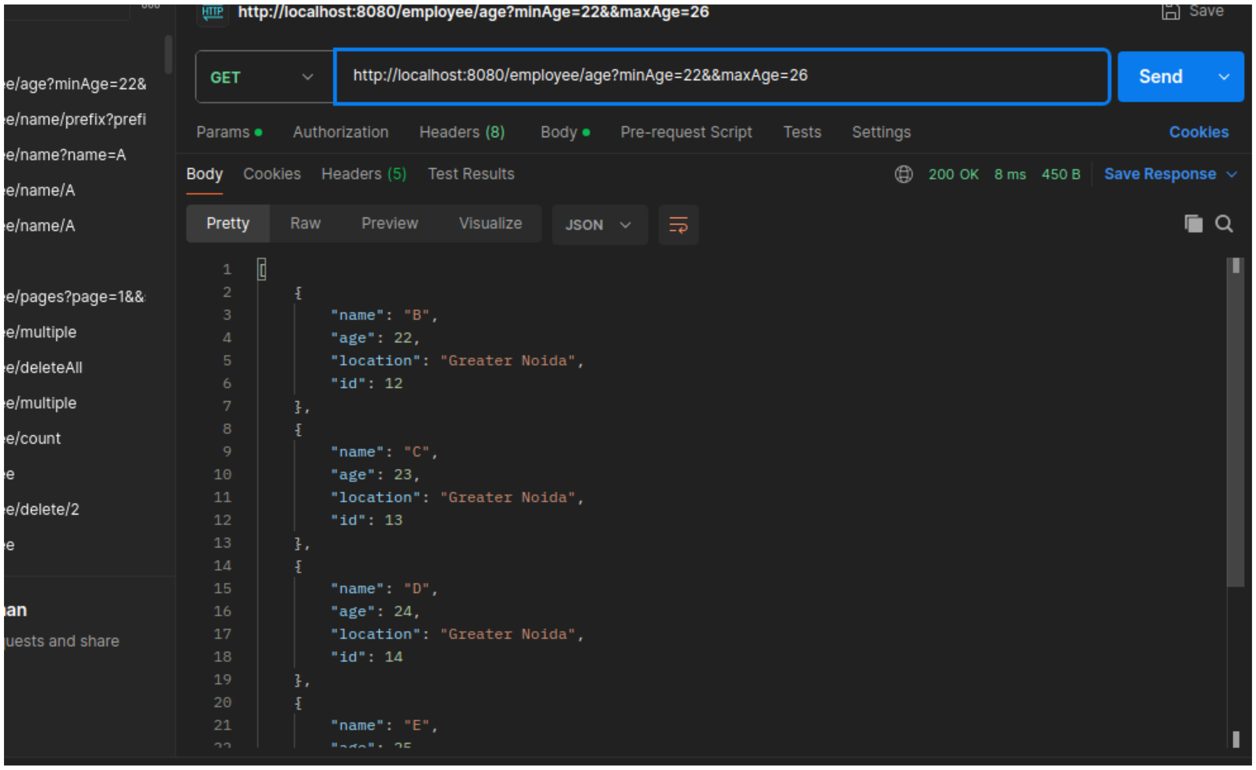


```
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {  2 usages

 List<Employee> findByName(String name);  1 usage

 List<Employee> findByNameStartingWith(String prefix);  1 usage
```

Q11)Create and use finder to find Employees Between the age of 28 to 32



```
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {  2 usages

  List<Employee> findByName(String name);  1 usage

  List<Employee> findByNameStartingWith(String prefix);  1 usage

  List<Employee> findByAgeBetween(int startAge, int endAge);  no usages

}
```



http://localhost:8080/employee/age?minAge=22&&maxAge=26

GET http://localhost:8080/employee/age?minAge=22&&maxAge=26

200 OK 8 ms 450 B

```
[
    {
        "name": "B",
        "age": 22,
        "location": "Greater Noida",
        "id": 12
    },
    {
        "name": "C",
        "age": 23,
        "location": "Greater Noida",
        "id": 13
    },
    {
        "name": "D",
        "age": 24,
        "location": "Greater Noida",
        "id": 14
    },
    {
        "name": "E",
        "age": 25
```