Employee Management System using SpringBoot/Mysql

This project is an Employee Management System built using Spring Boot and MySQL. To verify that the application works correctly, I tested every operation of the API using Postman. Each operation is also reflected correctly in the Spring Boot console, which confirms that the backend and the database are connected properly.

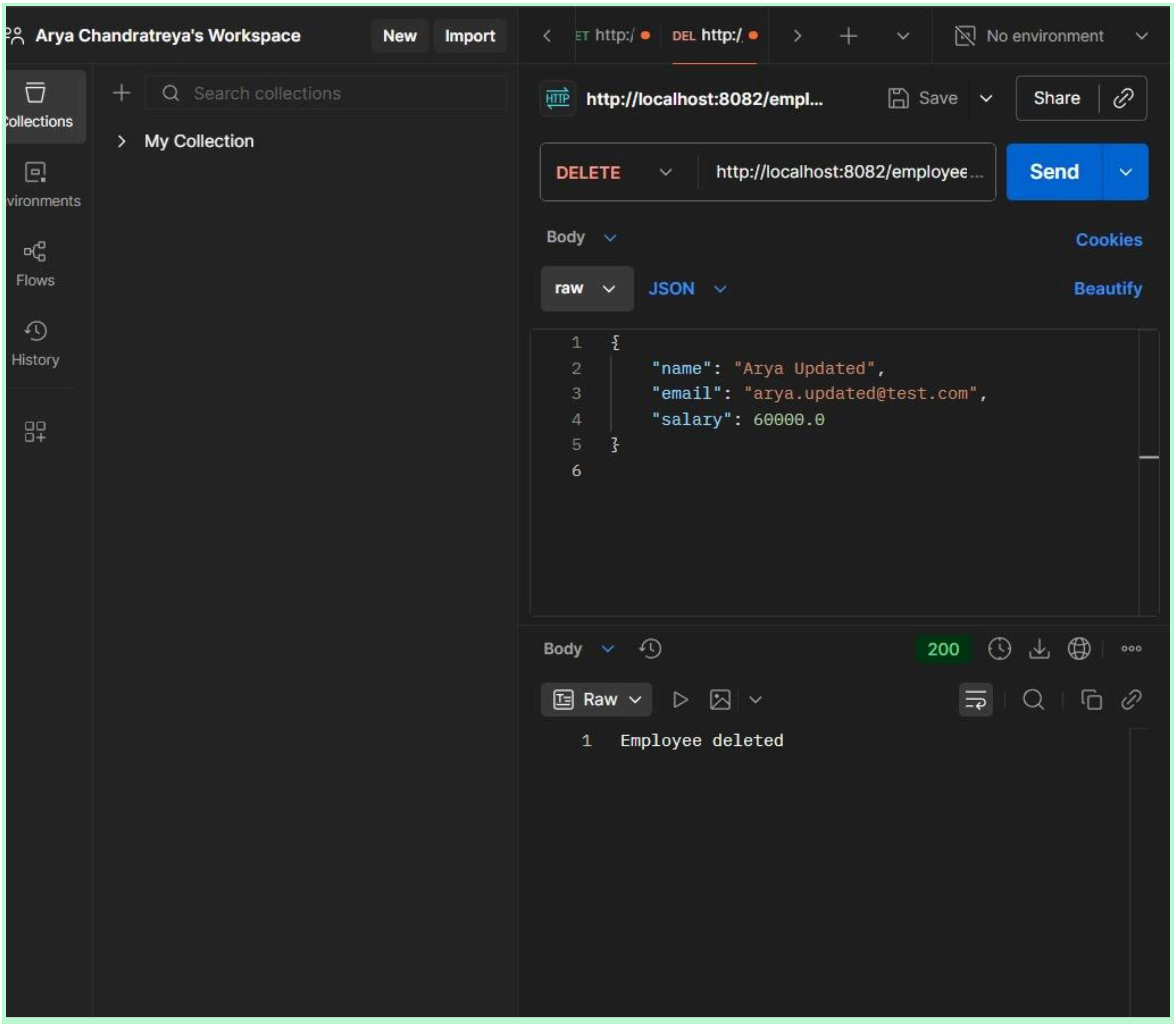
I first tested the create operation by sending a POST request to add a new employee. The API accepted the data, saved it into the database, and responded with the employee details including the generated ID. The console also showed the Hibernate queries running successfully.

Next, I tested the read operations. Using the GET request for a specific employee ID, the correct data was returned exactly as stored in the database. I also tested the GET request that returns all employees, and it displayed the full list of records, showing that the retrieval logic is working perfectly.

For the update operation, I sent a PUT request with a modified name, email, and salary. The employee details were updated in the database, and the changes were visible when retrieving the same employee again. This confirms that the update flow is functioning correctly.

Finally, I tested the delete operation. After sending a DELETE request for a specific ID, the record was removed successfully. A follow-up check showed that the employee no longer existed, proving that deletion works as expected.

All the screenshots included below show the actual Postman responses during testing. Every operation behaved correctly and consistently. This confirms that the CRUD features in the application are working end-to-end with proper database interaction.
Also the results were visible in STS in which we created our spring project.



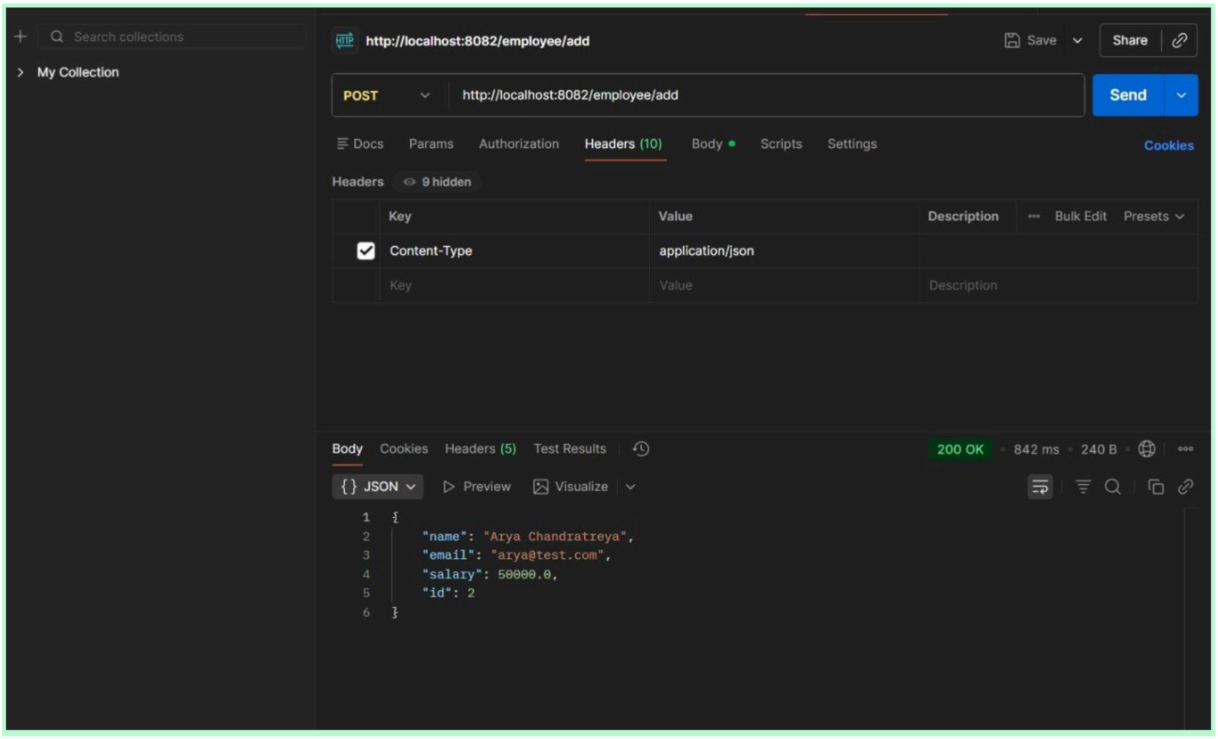
http://localhost:8082/empl...

DELETE http://localhost:8082/employee...

```
{
    "name": "Arya Updated",
    "email": "arya.updated@test.com",
    "salary": 60000.0
}
```

200

```
Employee deleted
```



http://localhost:8082/employee/add

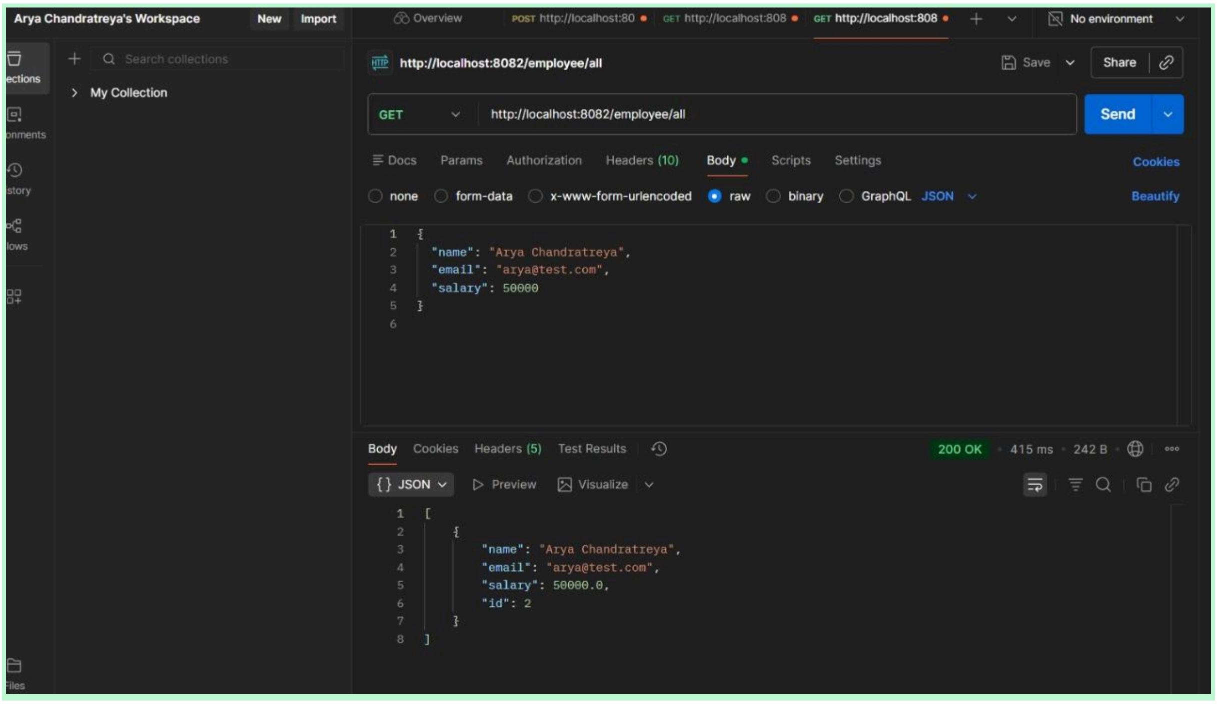
POST http://localhost:8082/employee/add

	Key	Value	Description
✓	Content-Type	application/json	
	Key	Value	Description

200 OK • 842 ms • 240 B

{} JSON

```
{
    "name": "Arya Chandratreya",
    "email": "arya@test.com",
    "salary": 50000.0,
    "id": 2
}
```



http://localhost:8082/employee/all

GET http://localhost:8082/employee/all

none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

```
{
  "name": "Arya Chandratreya",
  "email": "arya@test.com",
  "salary": 50000
}
```

200 OK • 415 ms • 242 B

{} JSON

```
[
    {
        "name": "Arya Chandratreya",
        "email": "arya@test.com",
        "salary": 50000.0,
        "id": 2
    }
]
```