

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.

DELETE OPERATION

The screenshot displays a REST client interface within a workspace named "Arya Chandratreya's Workspace". The interface is divided into several sections:

- Left Sidebar:** Contains navigation options: "collections", "My Collection", "environments", "Flows", and "History".
- Top Bar:** Shows the workspace name, "New" and "Import" buttons, and a breadcrumb trail: "< HTTP http://... > DEL http://...".
- URL Bar:** The active endpoint is "http://localhost:8082/empl...".
- Method and Path:** The HTTP method is set to "DELETE" and the path is "http://localhost:8082/employee...". A "Send" button is present.
- Body:** The request body is in "JSON" format, containing the following data:

```
1 {  
2   "name": "Arya Updated",  
3   "email": "arya.updated@test.com",  
4   "salary": 60000.0  
5 }  
6
```
- Response:** The status is "200". The response body is displayed in "Raw" format:

```
1 Employee deleted
```

ADD DATA

My Collection

Search collections

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

Send

Docs Params Authorization Headers (10) Body Scripts Settings Cookies

Headers 9 hidden

	Key	Value	Description		Bulk Edit	Presets
<input checked="" type="checkbox"/>	Content-Type	application/json				
	Key	Value	Description			

Body Cookies Headers (5) Test Results

200 OK • 842 ms • 240 B •

JSON Preview Visualize

```
1 {
2   "name": "Arya Chandratreya",
3   "email": "arya@test.com",
4   "salary": 50000.0,
5   "id": 2
6 }
```

GET ALL

HTTP http://localhost:8082/employee/all

Save Share

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

Send

Docs Params Authorization Headers (10) Body Scripts Settings Cookies

Query Params

	Key	Value	Description		Bulk Edit
	Key	Value	Description		

Body Cookies Headers (5) Test Results

200 OK • 484 ms • 396 B •

JSON Preview Visualize

```
4   "email": "arya@test.com",
5   "salary": 50000.0,
6   "id": 2
7 },
8 {
9   "name": "Arya Chandratreya",
10  "email": "arya@test.com",
11  "salary": 50000.0,
12  "id": 3
13 },
14 {
15  "name": "Arya Chandratreya"
```

UPDATE DATA

HTTP

http://localhost:8082/employee/update/1

Save

Share

</>

PUT

http://localhost:8082/employee/update/2

Send

>

Docs

Params

Authorization

Headers (10)

Body

Scripts

Settings

Cookies

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

☐ GraphQL

JSON

Beautify

```
1 {
2   "name": "Arya Updated",
3   "email": "arya.updated@test.com",
4   "salary": 60000.0
5 }
6
```

Body

Cookies

Headers (5)

Test Results

200 OK

116 ms

243 B

🌐

⋮

{ } JSON

Preview

Visualize

⌵

↩

☰

🔍

📄

🔗

```
1 {
2   "name": "Arya Updated",
3   "email": "arya.updated@test.com",
4   "salary": 60000.0,
5   "id": 2
6 }
```

🔄

GET DATA

HTTP

http://localhost:8082/employee/get/1

Save

Share

</>

GET

http://localhost:8082/employee/get/2

Send

↻

Docs

Params

Authorization

Headers (10)

Body

Scripts

Settings

Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (5)

Test Results

200 OK

15 ms

240 B

🌐

⋮

{ } JSON

▶ Preview

🖼 Visualize

⌵

1 {

2 " name": "Arya Chandratreya",

3 " email": "arya@test.com",

4 " salary": 50000.0,

5 " id": 2

6 }