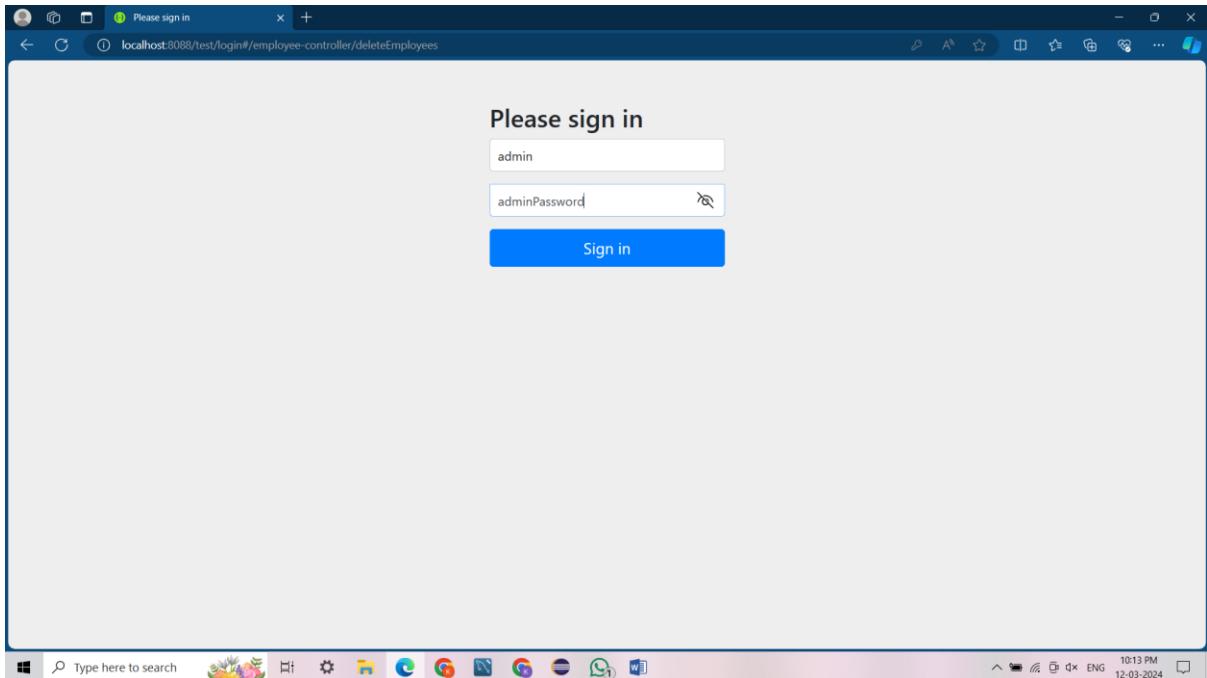
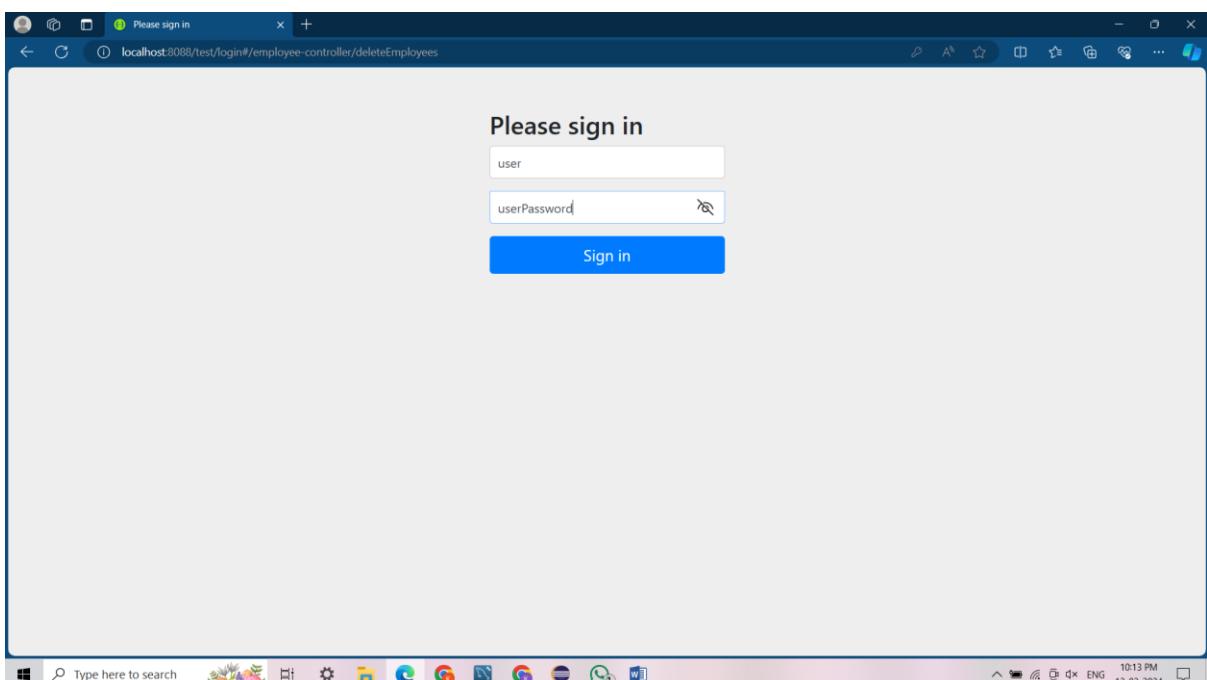


LOGINPAGE

Admin Login:



User Login:



Front Page:

The screenshot shows the Swagger UI interface for an OpenAPI definition. The URL in the browser is `localhost:8088/test/swagger-ui/index.html?continue`. The main title is "OpenAPI definition v0 OAS3". Below it, there's a "Servers" dropdown set to `http://localhost:8088/test - Generated server url`. The API is organized under the "employee-controller" section. It contains the following endpoints:

- PUT /updateEmployees** (orange background)
- POST /addNewEmployees** (green background)
- GET /sortByFirstName** (light blue background)
- GET /searchByFirstname** (light blue background)
- GET /getAllEmployees** (light blue background)
- DELETE /deleteEmployees** (red background)

The status bar at the bottom shows "Type here to search" and various system icons.

The screenshot shows the Swagger UI interface for an OpenAPI definition. The URL in the browser is `localhost:8088/test/swagger-ui/index.html?continue`. The main title is "OpenAPI definition v0 OAS3". Below it, there's a "Servers" dropdown set to `http://localhost:8088/test - Generated server url`. The API is organized under the "api-controller" section. It contains the following endpoints:

- GET /getAllEmployees** (orange background)
- DELETE /deleteEmployees** (red background)
- POST /api/updateEmployee** (green background)
- POST /api/deleteEmployee** (green background)
- POST /api/addNewEmployee** (green background)
- GET /api/getAllEmployee** (light blue background)
- GET /api/sortByFirstName** (light blue background)
- GET /api/searchByFirstname** (light blue background)
- GET /api/CheckLogin** (light blue background)
- GET /EmployeeHome** (light blue background)

Below the endpoints, there's a "Schemas" section with a single entry: "Employee". The status bar at the bottom shows "Type here to search" and various system icons.

EMPLOYEE

Employee without using Token

Create Employee:

Add Employee page

POST /addNewEmployees

Parameters

Name	Description
id * required integer(\$int32) (query)	1
firstname * required string (query)	Kim
lastname * required string (query)	Seokjin
email * required string (query)	kim@gmail.com

Responses

Curl

```
curl -X 'POST' \
'http://localhost:8088/test/addNewEmployees?id=1&firstname=Kim&lastname=Seokjin&email=kim%40gmail.com' \
-H 'accept: */*' \
-d ''
```

Employee Added

POST /addNewEmployees

Parameters

Name	Description
email * required string (query)	kim@gmail.com

Responses

Curl

```
curl -X 'POST' \
'http://localhost:8088/test/addNewEmployees?id=1&firstname=Kim&lastname=Seokjin&email=kim%40gmail.com' \
-H 'accept: */*' \
-d ''
```

Request URL

```
http://localhost:8088/test/addNewEmployees?id=1&firstname=Kim&lastname=Seokjin&email=kim%40gmail.com
```

Server response

Code Details

200

Response body

```
Employee added successfully
```

Response headers

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-length: 14
content-type: text/plain;charset=UTF-8
date: Tue, 12 Mar 2024 16:07:02 GMT
expires: 0
keep-alive: timeout=60
```

In database Added Employee

The screenshot shows the MySQL Workbench interface. In the Navigator pane, under the SCHEMAS section, a new database named 'GradedAssignment' is visible. Inside this database, there are several tables: 'car', 'chennai01', 'chennai_servlet', 'e_commerce', 'employee', 'employees', 'example', 'gl', and 'user_authorities'. A 'Views' folder is also present. The 'Information' pane shows 'No object selected'. The central Query Grid displays the following employee data:

ID	Email	Firstname	Lastname
1	kim@gmail.com	Kim	Seokjin
2	Min@gmail.com	Min	Yoonki
3	Jung@gmail.com	Jung	Hoseok
4	kim@gmail.com	Kim	Joorie

The SQL Editor pane contains the following SQL code:

```
1 • create database GradedAssignment;
2 • use GradedAssignment;
3 • show tables;
4 • select * from employee;
5 • select * from app_user;
6 • select * from userAuthorities;
```

The status bar at the bottom right indicates the time as 09:39 PM and the date as 12-03-2024.

Read Employee:

The screenshot shows a Swagger UI interface for a REST API. The URL is `localhost:8088/test/swagger-ui/index.html?continue#/employee-controller/getAllEmployees`. The method is set to `GET /getAllEmployees`. The 'Parameters' section shows 'No parameters'. Below it is an 'Execute' button. The 'Responses' section includes a 'Curl' command and a 'Request URL' input field containing `http://localhost:8088/test/getAllEmployees`. The 'Server response' section shows a 'Code' tab and a 'Details' tab. Under the 'Code' tab, the status code is 200, and the 'Response body' is displayed as JSON:

```
[{"id": 1, "firstname": "Kim", "lastname": "Seokjin", "email": "kim@gmail.com"}, {"id": 2, "firstname": "Min", "lastname": "Yoonki", "email": "Min@gmail.com"}]
```

The status bar at the bottom right indicates the time as 09:42 PM and the date as 12-03-2024.

Swagger UI

Request URL: <http://localhost:8088/test/swagger-ui/index.html?continue#/employee-controller/getAllEmployees>

-H accept: */*

Server response

Code Details

200 Response body

```
{
  "id": 3,
  "firstname": "Jung",
  "lastname": "Hoseok",
  "email": "Jung@gmail.com"
},
{
  "id": 5,
  "firstname": "Kim",
  "lastname": "Naejion",
  "email": "kim@gmail.com"
},
{
  "id": 6,
  "firstname": "Park",
  "lastname": "Jimin",
  "email": "park@gmail.com"
},
{
  "id": 6,
  "firstname": "Kim",
  "lastname": "Taehyung",
  "email": "kim@gmail.com"
},
{
  "id": 7,
  "firstname": "Jeon",
  "lastname": "Jungkook",
  "email": "jeon@gmail.com"
}
```

Response headers

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-type: application/json
date: Tue, 12 Mar 2024 16:12:01 GMT
expires: 0
keep-alive: timeout=60
pragma: no-cache
transfer-encoding: chunked
```

Download

In database

MySQL Workbench

Local instance MySQL >

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- car
- chennai01
- chennai_servlet
- e_commerce
- employee
- employees
- example
- gl
- GradedAssignment
- Tables
- Views
- Stored Procedures

Administration Schemas

No object selected

Query 1 SQL File 2* SQL File 4* SQL File 5* SQL File 6* SQL File 7*

```
1 • create database GradedAssignment;
2 • use GradedAssignment;
3 • show tables;
4 • select * from employee;
5 • select * from app_user;
6 • select * from userAuthorities;
```

Result Grid

	id	email	firstname	lastname
1	1	kin@gmail.com	Kim	Seokjin
2	2	Min@gmail.com	Min	Yoonki
3	3	Jung@gmail.com	Jung	Hoseok
4	4	kim@gmail.com	Kim	Nampon
5	5	park@gmail.com	Park	Jimin
6	6	kin@gmail.com	Kim	Taehyung
7	7	Jung@gmail.com	Jeon	Jungkook

employee 6 x

Action Output

#	Time	Action	Message	Duration / Fetch
6	21:35:18	select * from employee LIMIT 0, 50000	7 row(s) returned	0.000 sec / 0.000 sec
7	21:36:12	select * from employee LIMIT 0, 50000	0 row(s) returned	0.000 sec / 0.000 sec
8	21:39:08	select * from employee LIMIT 0, 50000	4 row(s) returned	0.000 sec / 0.000 sec
9	21:40:14	select * from employee LIMIT 0, 50000	4 row(s) returned	0.016 sec / 0.000 sec
10	21:42:20	select * from employee LIMIT 0, 50000	7 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

Update Employee:

The screenshot shows the Swagger UI interface for an 'updateEmployees' endpoint. The form fields are filled with the following values:

- firstname: Kim
- lastname: Namjoon
- email: kim@gmail.com

Below the form is an 'Execute' button and a 'Clear' button. Under the 'Responses' section, there is a 'Curl' code block and a 'Request URL' block. The 'Server response' section shows a status of 200 with the message "Employee updated successfully".

In database update employee

The screenshot shows MySQL Workbench with the following SQL query in the Query Editor:

```
1 • create database GradedAssignment;
2 • use GradedAssignment;
3 • show tables;
4 • select * from employee;
5 • select * from app_user;
6 • select * from userAuthorities;
```

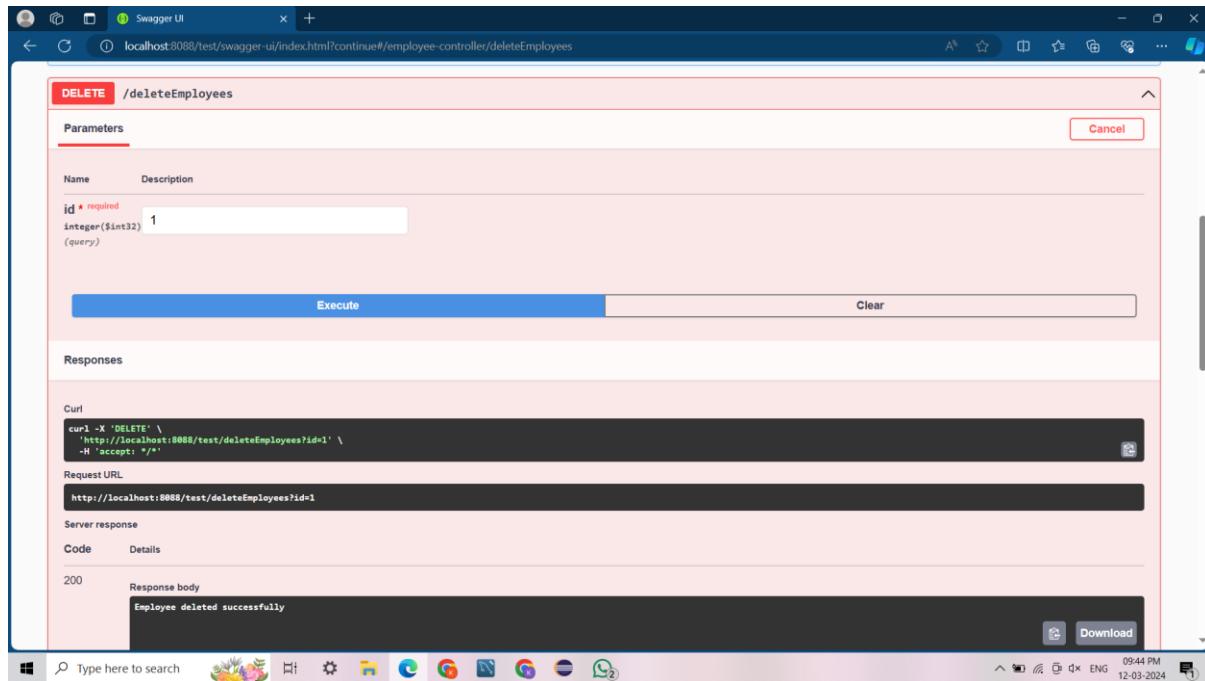
The Results Grid displays the following data from the 'employee' table:

	id	email	firstname	lastname
1	1	kim@gmail.com	Kim	Seokjin
2	2	Min@gmail.com	Min	Yoonki
3	3	Jung@gmail.com	Jung	Hoseok
4	4	kim@gmail.com	Kim	Namjoon

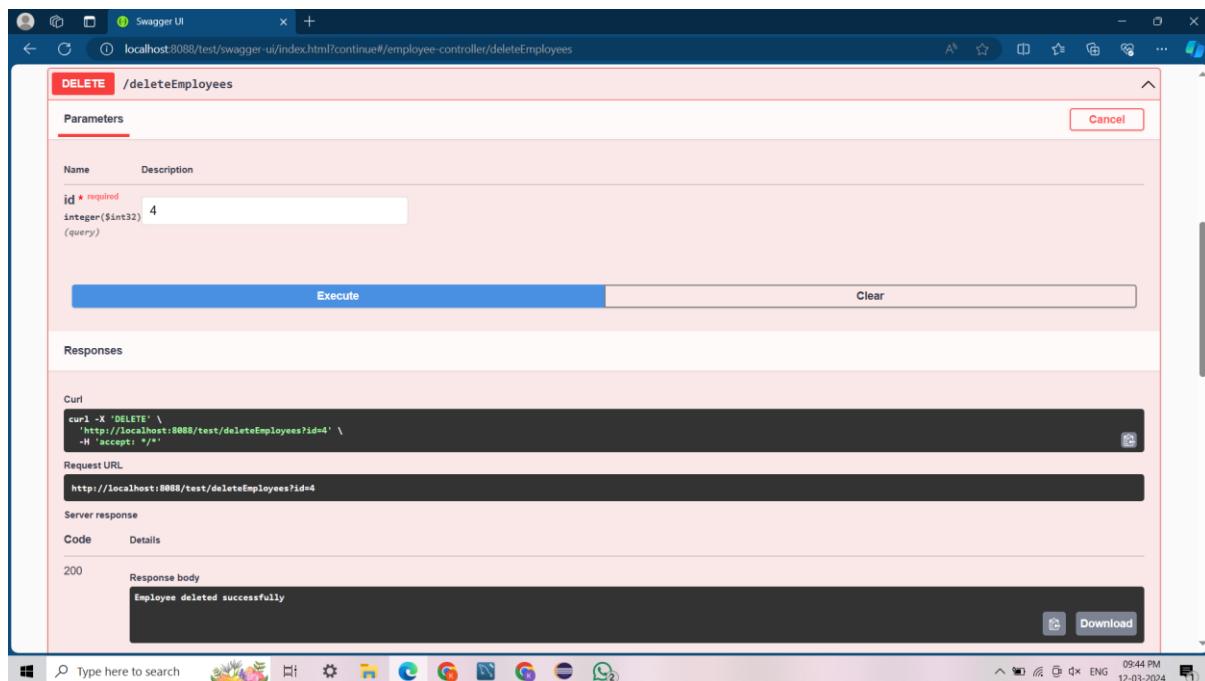
The Output pane shows the execution history of the query:

Action	Time	Action	Message	Duration / Fetch
5	21:08:14	select * from employee LIMIT 0, 50000	12 row(s) returned	0.000 sec / 0.000 sec
6	21:35:18	select * from employee LIMIT 0, 50000	7 row(s) returned	0.000 sec / 0.000 sec
7	21:36:12	select * from employee LIMIT 0, 50000	0 row(s) returned	0.000 sec / 0.000 sec
8	21:39:08	select * from employee LIMIT 0, 50000	4 row(s) returned	0.000 sec / 0.000 sec
9	21:40:14	select * from employee LIMIT 0, 50000	4 row(s) returned	0.016 sec / 0.000 sec

Delete Employee:

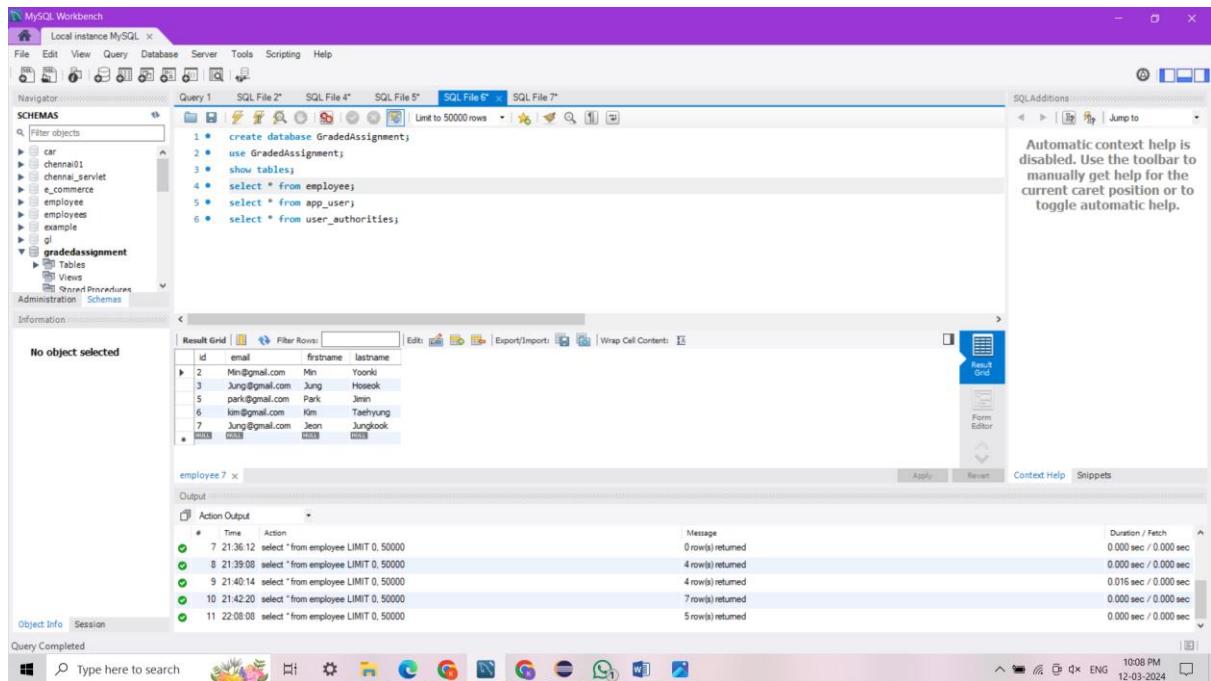


The screenshot shows the Swagger UI interface for a DELETE request to the endpoint `/deleteEmployees`. The **Parameters** section contains a single required parameter `id` with a value of `1`. Below the parameters are two buttons: **Execute** and **Clear**. The **Responses** section displays the server response code `200` and the response body `Employee deleted successfully`. At the bottom of the window, there is a toolbar with various icons and a status bar indicating the date and time.

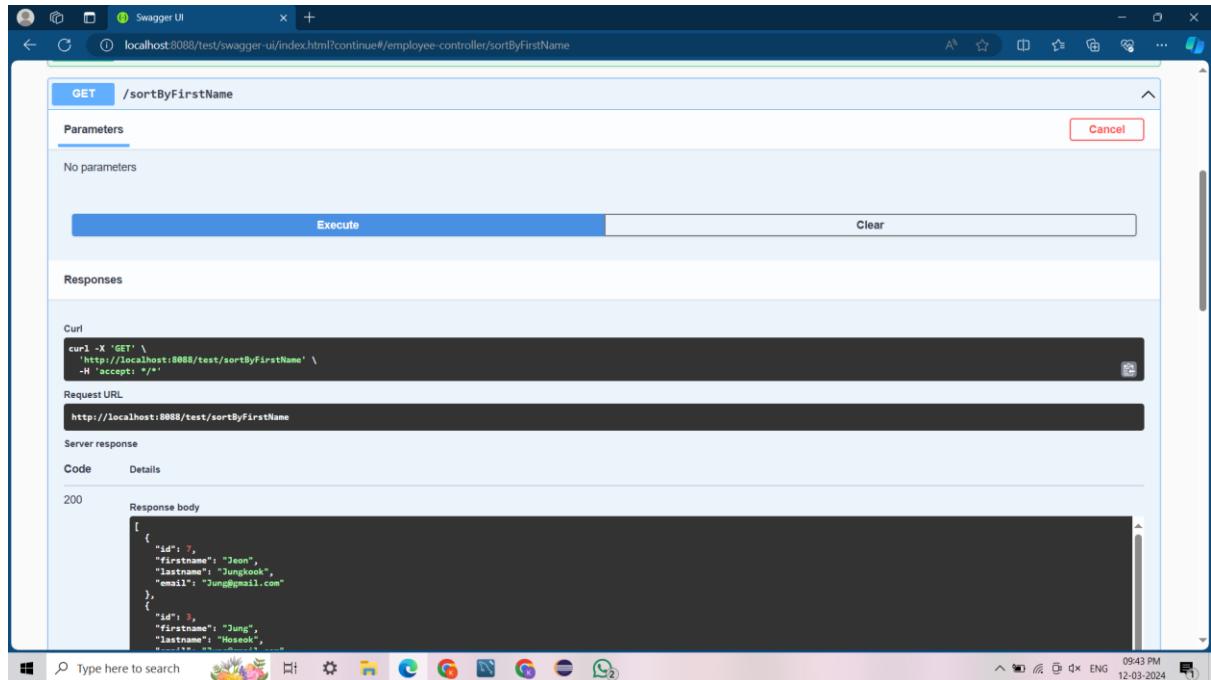


This screenshot is identical to the one above, showing the same DELETE request to `/deleteEmployees` with `id=1`. The only difference is the value of the `id` parameter, which has been changed to `4`. The rest of the interface, including the **Responses** section and the status bar, remains the same.

In database delete employee



Sort Employee:



Swagger UI

localhost:8088/test/swagger-ui/index.html?continue#/employee-controller/sortByFirstName

```
curl -X 'GET' \
  'http://localhost:8088/test/sortByFirstName' \
  -H 'accept: */*'
```

Request URL

<http://localhost:8088/test/sortByFirstName>

Server response

Code Details

200

Response body

```
[{"id": 3, "firstname": "Jung", "lastname": "Hoseok", "email": "Jung@gmail.com"}, {"id": 1, "firstname": "Kim", "lastname": "Seokjin", "email": "kim@gmail.com"}, {"id": 4, "firstname": "Kim", "lastname": "Naejion", "email": "kim@gmail.com"}, {"id": 6, "firstname": "Kim", "lastname": "Taehyung", "email": "kim@gmail.com"}, {"id": 2, "firstname": "Min", "lastname": "Yoonki", "email": "Min@gmail.com"}]
```

Download

Response header

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-type: application/json
date: Tue,12 Mar 2024 16:13:33 GMT
expires: 0
```

Type here to search

09:43 PM 12-03-2024

Swagger UI

localhost:8088/test/swagger-ui/index.html?continue#/employee-controller/sortByFirstName

```
curl -X 'GET' \
  'http://localhost:8088/test/sortByFirstName' \
  -H 'accept: */*'
```

Request URL

<http://localhost:8088/test/sortByFirstName>

Server response

Code Details

200

Response body

```
[{"id": 3, "firstname": "Jung", "lastname": "Hoseok", "email": "Jung@gmail.com"}, {"id": 4, "firstname": "Kim", "lastname": "Naejin", "email": "kim@gmail.com"}, {"id": 6, "firstname": "Kim", "lastname": "Taehyung", "email": "kim@gmail.com"}, {"id": 2, "firstname": "Min", "lastname": "Yoonki", "email": "Min@gmail.com"}, {"id": 5, "firstname": "Park", "lastname": "Jinmin", "email": "park@gmail.com"}]
```

Download

Response headers

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-type: application/json
date: Tue,12 Mar 2024 16:13:33 GMT
expires: 0
```

Type here to search

09:43 PM 12-03-2024

Search Employee:

The screenshot shows the Swagger UI interface for a 'searchByFirstname' endpoint. In the 'Parameters' section, a 'firstname' parameter is set to 'kim'. Below it, there are 'Execute' and 'Clear' buttons. The 'Responses' section includes a 'Curl' command and a 'Request URL' of 'http://localhost:8088/test/searchByFirstname?firstname=kim'. The 'Server response' section shows a 200 status code with a JSON response body containing three employee records. The records have IDs 2, 4, and 6, with first names 'Kim', 'Kim', and 'Taehyung' respectively, and last names 'Seokjin', 'Namejoun', and 'Taehyung', and emails 'kim@gmail.com'.

```
curl -X 'GET' \
'http://localhost:8088/test/searchByFirstname?firstname=kim' \
-H 'accept: */*'
```

```
http://localhost:8088/test/searchByFirstname?firstname=kim
```

```
[{"id": 2, "firstname": "Kim", "lastname": "Seokjin", "email": "kim@gmail.com"}, {"id": 4, "firstname": "Kim", "lastname": "Namejoun", "email": "kim@gmail.com"}, {"id": 6, "firstname": "Kim", "lastname": "Taehyung", "email": "kim@gmail.com"}]
```

This screenshot shows the same 'searchByFirstname' endpoint in the Swagger UI. The 'Responses' section includes a 'Curl' command and a 'Request URL' of 'http://localhost:8088/test/searchByFirstname?firstname=kim'. The 'Server response' section shows a 200 status code with a JSON response body containing three employee records. Below the response body, the 'Response headers' section displays the following HTTP headers:

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-type: application/json
date: Fri, 09 Mar 2024 16:14:12 GHT
expires: 0
keep-alive: timeout=60
pragma: no-cache
transfer-encoding: chunked
x-content-type-prefix: nosniff
x-frame-options: DENY
```

EMPLOYEE - API

Employee using Token

Login Check Generating Token:

For Admin:

The screenshot shows the Swagger UI interface for the 'CheckLogin' API endpoint. The 'Parameters' section has two fields: 'name' (string, required) with value 'admin' and 'password' (string, required) with value 'adminPassword'. Below the parameters are 'Execute' and 'Clear' buttons. The 'Responses' section includes a 'Curl' command and a 'Request URL' field containing 'http://localhost:8088/test/api/CheckLogin?name=admin&password=adminPassword'. The 'Server response' section shows a 200 status code with a 'Response body' field containing a large JSON object. The Windows taskbar at the bottom shows various application icons.

This screenshot is similar to the one above, showing the 'CheckLogin' API endpoint for an admin. The parameters 'name' (admin) and 'password' (adminPassword) are set. The 'Responses' section shows a 200 status code with a 'Response body' field containing a very large JSON object, indicated by three horizontal ellipsis dots. Below the response body is a 'Download' button. The Windows taskbar at the bottom is visible.

For User:

The screenshot shows the Swagger UI interface for a GET request to the endpoint `/api/CheckLogin`. The request requires two parameters: `name` (with value `user`) and `password` (with value `userPassword`). Below the parameters, there are "Execute" and "Clear" buttons. The "Responses" section is collapsed. On the left, there are sections for "Curl" (containing a command-line example), "Request URL" (containing the full URL), and "Server response". The "Server response" section shows a status code of 200 and a "Response body" field containing the placeholder text "Response body". The bottom of the screen shows a taskbar with various icons and system status.

This screenshot shows the same Swagger UI interface as the previous one, but the "Server response" section is expanded. It displays a complex JSON response body consisting of a single line of encoded data. Below the response body, the "Response headers" section is expanded, showing the following header information:

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-length: 182
content-type: text/plain;charset=UTF-8
date: Tue, 12 Mar 2024 16:57:14 GMT
```

The rest of the interface remains consistent with the first screenshot, including the "Curl" and "Request URL" sections.

Add Employee:

Adding Employee

POST /api1/addNewEmployee

Parameters

Name	Description
id * required	integer(\$int32) (query)
firstname * required	string (query)
lastname * required	string (query)
email * required	string (query)
token * required	string (query)

Execute Clear

Responses

Curl

```
curl -X POST \
http://localhost:8088/test/api/addNewEmployee?id=4&firstname=Jung&lastname=Hoseok&email=jung%40gmail.com&token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxliwiaWF0IjoxNzEwMjYyNjY3LCJzdWIiOiJhZG1pbisImV4cC16MzA3ODQ3MjkwOTAwMjQ0.QUiwOkmmUi5wujpusUPBXq95c6xIOZr
-d ''
```

POST /api1/addNewEmployee

Parameters

Name	Description
email * required	string (query)
token * required	string (query)

Responses

Curl

```
curl -X POST \
http://localhost:8088/test/api/addNewEmployee?id=4&firstname=Jung&lastname=Hoseok&email=jung%40gmail.com&token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxliwiaWF0IjoxNzEwMjYyNjY3LCJzdWIiOiJhZG1pbisImV4cC16MzA3ODQ3MjkwOTAwMjQ0.QUiwOkmmUi5wujpusUPBXq95c6xIOZr
-d ''
```

Request URL

```
http://localhost:8088/test/api/addNewEmployee?id=4&firstname=Jung&lastname=Hoseok&email=jung%40gmail.com&token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxliwiaWF0IjoxNzEwMjYyNjY3LCJzdWIiOiJhZG1pbisImV4cC16MzA3ODQ3MjkwOTAwMjQ0.QUiwOkmmUi5wujpusUPBXq95c6xIOZr
```

Server response

Code Details

200 Response body

```
Employee Added Successfully
```

Download

Response headers

In database add employee

The screenshot shows the MySQL Workbench interface. In the top-left, the Navigator pane displays the schema structure under 'Schemas'. A query window titled 'Query 1' contains the following SQL code:

```
1 • create database GradedAssignment;
2 • use GradedAssignment;
3 • show tables;
4 • select * from employee;
5 • select * from app_user;
6 • select * from userAuthorities;
```

The results grid shows the 'employee' table with 8 rows of data:

	id	email	firstname	lastname
1	1	Kim@gmail.com	Kim	Seokjin
2	2	Min@gmail.com	Min	Yeonil
3	3	Jung@gmail.com	Jung	Hoseok
4	4	Jung@gmail.com	Jung	Hoseok
5	5	park@gmail.com	Park	Jimin
6	6	kim@gmail.com	Kim	Taehyung
7	7	Jung@gmail.com	Jean	Jungkook

The bottom section shows the 'Output' tab with a table of action logs:

Action	Time	Message	Duration / Fetch
select * from employee LIMIT 0, 50000	8 21:39:08	4 row(s) returned	0.000 sec / 0.000 sec
select * from employee LIMIT 0, 50000	9 21:40:14	4 row(s) returned	0.016 sec / 0.000 sec
select * from employee LIMIT 0, 50000	10 21:42:20	7 row(s) returned	0.000 sec / 0.000 sec
select * from employee LIMIT 0, 50000	11 22:08:08	5 row(s) returned	0.000 sec / 0.000 sec
select * from employee LIMIT 0, 50000	12 22:35:11	7 row(s) returned	0.000 sec / 0.000 sec

Read Employee:

The screenshot shows the Swagger UI interface for a 'GET /getAllEmployee' endpoint. The 'Parameters' section includes a required parameter 'token' with the value 'eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxIiwidF0IjoxNzEwMjYyNgY3LCJzdWIiOiJhZGlpbiIsImV4cCI6MzA3ODQ3Mjg4MDYwOTAwMH0.QIIwOkmmUI9wulpusUPBXq9SzExIOZrPFFYeDDXs9w8'. The 'Responses' section shows the expected response body:

```
[{"id": 1, "firstname": "Kim", "lastname": "Seokjin"}, {"id": 2, "firstname": "Min", "lastname": "Yeonil"}, {"id": 3, "firstname": "Jung", "lastname": "Hoseok"}, {"id": 4, "firstname": "Jung", "lastname": "Hoseok"}, {"id": 5, "firstname": "Park", "lastname": "Jimin"}, {"id": 6, "firstname": "Kim", "lastname": "Taehyung"}, {"id": 7, "firstname": "Jung", "lastname": "Jungkook"}]
```

localhost:8088/test/swagger-ui/index.html?continue#/api-controller/getAllEmployee

200

Response body

```
[{"id": 1, "firstname": "Jung", "lastname": "Hoseok", "email": "Jung@gmail.com"}, {"id": 4, "firstname": "Jung", "lastname": "Hoseok", "email": "jung@gmail.com"}, {"id": 5, "firstname": "Park", "lastname": "Jinmin", "email": "park@gmail.com"}, {"id": 6, "firstname": "Kim", "lastname": "Taehyung", "email": "kim@gmail.com"}, {"id": 7, "firstname": "Jeon", "lastname": "Jungkook", "email": "jeon@gmail.com"}]
```

Download

Response headers

```
cache-control: no-cache,no-store,max-age=0,must-revalidate  
connection: keep-alive  
content-type: application/json  
date: Tue,12 Mar 2024 17:08:19 GMT  
expires:  
keepalive: timeout=60  
pragma: no-cache  
transfer-encoding: chunked  
x-content-type-options: nosniff  
x-frame-options: DENY  
x-xss-protection: 0
```

Responses

Code	Description
200	

No links

Update Employee:

localhost:8088/test/swagger-ui/index.html?continue#/api-controller/updateEmployee

POST /api/updateEmployee

Parameters

Name	Description
id * required	integer(\$int32) (query) 1
firstname * required	string (query) Kim
lastname * required	string (query) jin
email * required	string (query) kim@gmail.com
token * required	string (query) eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxliwiaWF

Execute

Responses

Code	Description
200	

localhost:8088/test/swagger-ui/index.html?continue#/api-controller/updateEmployee

kim@gmail.com

token * required
string
(query)

eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxliwiaWF
(query)

Execute **Clear**

Responses

Curl

```
curl -X 'POST' \
'http://localhost:8088/test/api/updateEmployee?id=1&firstname=Kim&lastname=jin&email=kim%4@gmail.com&token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxliwiaWF0IjoaNzEwMjYyNjY3LCzdwIiOjZhZG1pbilsImV4cCIEhA30DQ3MjgwD'
-H 'accept: /*' \
-d ''
```

Request URL

```
http://localhost:8088/test/api/updateEmployee?
id=1&firstname=Kim&lastname=jin&email=kim%4@gmail.com&token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxliwiaWF0IjoaNzEwMjYyNjY3LCzdwIiOjZhZG1pbilsImV4cCIEhA30DQ3MjgwD
00Xs&d
```

Server response

Code	Details
200	Response body Employee Updated Successfully <div style="text-align: right;">Download</div> Response headers cache-control: no-cache,no-store,max-age=0,must-revalidate connection: keep-alive

In database update employee

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Local instance MySQL >

Navigator

SCHEMAS

- car
- chennai1
- chennai_servlet
- e_commerce
- employee
- employees
- example
- gl
- GradedAssignment
- Tables
- Views
- Stored Procedures

Administration Schemas

No object selected

Query 1 SQL File 2* SQL File 4* SQL File 5* SpringBookAssignment SQL File 7*

```
1 • create database GradedAssignment;
2 • use GradedAssignment;
3 • show tables;
4 • select * from employee;
5 • select * from app_user;
6 • select * from userAuthorities;
```

Result Grid | Filter Rows: Limit to 50000 rows | Edit: | Export/Import: | Wrap Cell Contents: |

	id	email	firstname	lastname
1	1	kim@gmail.com	Kim	Jin
2	2	Min@gmail.com	Min	Yoonki
3	3	Jung@gmail.com	Jung	Hoseok
4	4	jung@gmail.com	Jung	Hoseok
5	5	park@gmail.com	Park	Jimin
6	6	kim@gmail.com	Kim	Taehyung
7	7	Jung@gmail.com	Jeon	Jungkook
8	8			
9	9			

employee 9 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
9	21:40:14	select * from employee LIMIT 0, 50000	4 row(s) returned	0.016 sec / 0.000 sec
10	21:42:20	select * from employee LIMIT 0, 50000	7 row(s) returned	0.000 sec / 0.000 sec
11	22:08:08	select * from employee LIMIT 0, 50000	5 row(s) returned	0.000 sec / 0.000 sec
12	22:35:11	select * from employee LIMIT 0, 50000	7 row(s) returned	0.000 sec / 0.000 sec
13	22:43:27	select * from employee LIMIT 0, 50000	7 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

Query Completed

Delete Employee:

The screenshot shows the Swagger UI interface for a POST request to the endpoint `/api/deleteEmployee`. The request requires two parameters: `id` (integer, required) and `token` (string, required). The `id` field is set to 7, and the `token` field contains a JWT token: `eyJhbGciOiJIUzI1NiJ9eyJpc3MiOiIxIiwiaWF0IjoxNzEwMjY3LCJzdWIiOiJhZG1pbmIsImV4cCI6IzA3ODQ3Mjg#IDYwOTAw#H0.QIIwOkmmUI9vu1pusUPBXq95zGxIOZrPFFYvDDX-d`. Below the parameters are buttons for "Execute" and "Clear". The "Responses" section is collapsed. The "Curl" section shows the command to run the request from the terminal. The "Request URL" section shows the full URL: `http://localhost:8088/test/api/deleteEmployee?id=7&token=eyJhbGciOiJIUzI1NiJ9eyJpc3MiOiIxIiwiaWF0IjoxNzEwMjY3LCJzdWIiOiJhZG1pbmIsImV4cCI6IzA3ODQ3Mjg#IDYwOTAw#H0.QIIwOkmmUI9vu1pusUPBXq95zGxIOZrPFFYvDDX-d`. The "Server response" section is collapsed. The "Code" and "Details" tabs are visible at the bottom.

The screenshot shows the same Swagger UI interface after the POST request has been executed. The "Responses" section is expanded, showing a successful response (200 OK) with the message "Employee Deleted Successfully". Below the message are "Response headers" with values: `cache-control: no-cache,no-store,max-age=0,must-revalidate`, `connection: keep-alive`, and `content-length: 28`. There are "Download" and "Copy" buttons next to the response message. The "Curl" and "Request URL" sections are identical to the previous screenshot. The "Server response" section is collapsed. The "Code" and "Details" tabs are visible at the bottom.

In database delete employee

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the database is set to "Local instance MySQL". The left sidebar lists various schemas and tables, including "gradedassignment" which contains the "employee" table. The main pane displays a SQL query:

```
1 • create database GradedAssignment;
2 • use GradedAssignment;
3 • show tables;
4 • select * from employee;
5 • select * from app_user;
6 • select * from userAuthorities;
```

Below the query, the "Result Grid" shows the "employee" table with 6 rows of data:

	id	email	firstname	lastname
▶	1	kim@gmail.com	Kim	Jin
	2	Min@gmail.com	Min	Yeonki
	3	Jung@gmail.com	Jung	Hoseok
	4	jung@gmail.com	Jung	Hoseok
	5	park@gmail.com	Park	Jimin
	6	kim@gmail.com	Kim	TaeHyung

The "Output" pane shows the execution history of the query:

#	Time	Action	Message	Duration / Fetch
10	21:42:20	select * from employee LIMIT 0, 50000	7row(s) returned	0.000 sec / 0.000 sec
11	22:08:08	select * from employee LIMIT 0, 50000	5row(s) returned	0.000 sec / 0.000 sec
12	22:35:11	select * from employee LIMIT 0, 50000	7row(s) returned	0.000 sec / 0.000 sec
13	22:43:27	select * from employee LIMIT 0, 50000	7row(s) returned	0.015 sec / 0.000 sec
14	23:04:28	select * from employee LIMIT 0, 50000	6row(s) returned	0.000 sec / 0.000 sec

The status bar at the bottom indicates "Query Completed" and the date and time "12-03-2024 11:04 PM".

Sort Employee:

The screenshot shows a browser window displaying the Swagger UI for an API endpoint. The URL is "localhost:8088/test/swagger-ui/index.html?continue#/api-controller/sortByFirstName_1". The endpoint is a GET request to "/api/sortByFirstName".

The "Parameters" section shows a required parameter "token" with the value "eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiIxIiwiaWF0IjoxNzEwMjYyNjY3LCJzdWIiOjIzMjZGipbiIsImV4cCI6IzA3ODQ3IjgwMDYwOTAwMH0.QIIwOkmmU19wulpusUPBxq5SzGxIOZrPFYyEDXs9w8".

The "Responses" section includes "Curl" and "Request URL" examples, and a "Server response" section showing a JSON response body:

```
[{"id": 3, "firstname": "Jung"}]
```

The status bar at the bottom indicates "11:06 PM 12-03-2024".

localhost:8088/test/swagger-ui/index.html?continue#/api-controller/sortByFirstName_1

Curl

```
curl -X 'GET' \
'http://localhost:8088/test/api/sortByFirstName?token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MIoIxIiwidF0IjoxNzEwMjYyNjY3LCJzdWIiO1JhZG1pbiliIwV4cCIGHsA30DQ3IjgwMDYwOTAwMHB.QIIuOkmmUI9vuIpusUPBXq9SzGxIOZrPFFYeDOxs9w&H' \
-H 'accept: */*'
```

Request URL

```
http://localhost:8088/test/api/sortByFirstName?token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MIoIxIiwidF0IjoxNzEwMjYyNjY3LCJzdWIiO1JhZG1pbiliIwV4cCIGHsA30DQ3IjgwMDYwOTAwMHB.QIIuOkmmUI9vuIpusUPBXq9SzGxIOZrPFFYeDOxs9w&H
```

Server response

Code Details

200 Response body

```
[{"id": 3, "firstname": "Jung", "lastname": "Hoseok", "email": "Jung@gmail.com"}, {"id": 4, "firstname": "Jung", "lastname": "Hoseok", "email": "jung@gmail.com"}, {"id": 5, "firstname": "Kim", "lastname": "jin", "email": "kim@gmail.com"}, {"id": 6, "firstname": "Kim", "lastname": "Taehyung", "email": "kim@gmail.com"}, {"id": 2, "firstname": "Min", "lastname": "Yoonki", "email": "Min@gmail.com"}]
```

Response headers

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
```

localhost:8088/test/swagger-ui/index.html?continue#/api-controller/sortByFirstName_1

Curl

```
curl -X 'GET' \
'http://localhost:8088/test/api/sortByFirstName?token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MIoIxIiwidF0IjoxNzEwMjYyNjY3LCJzdWIiO1JhZG1pbiliIwV4cCIGHsA30DQ3IjgwMDYwOTAwMHB.QIIuOkmmUI9vuIpusUPBXq9SzGxIOZrPFFYeDOxs9w&H' \
-H 'accept: */*'
```

Request URL

```
http://localhost:8088/test/api/sortByFirstName?token=eyJhbGciOiJIUzI1NiJ9.eyJpc3MIoIxIiwidF0IjoxNzEwMjYyNjY3LCJzdWIiO1JhZG1pbiliIwV4cCIGHsA30DQ3IjgwMDYwOTAwMHB.QIIuOkmmUI9vuIpusUPBXq9SzGxIOZrPFFYeDOxs9w&H
```

Server response

Code Details

200 Response body

```
[{"id": 3, "firstname": "Jung", "lastname": "Hoseok", "email": "Jung@gmail.com"}, {"id": 1, "firstname": "Kim", "lastname": "jin", "email": "kim@gmail.com"}, {"id": 6, "firstname": "Kim", "lastname": "Taehyung", "email": "kim@gmail.com"}, {"id": 2, "firstname": "Min", "lastname": "Yoonki", "email": "Min@gmail.com"}, {"id": 5, "firstname": "Park", "lastname": "Jisun", "email": "park@gmail.com"}]
```

Response headers

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
```

Search Employee:

