

CREATE EXAMPLES (Post)

The screenshot displays a REST client interface with a top tab bar containing various HTTP methods and endpoints. The active tab is 'POST Create S' for the endpoint 'http://localhost:5000/api/createShift'. The request body is a JSON object with the following fields: employeeID (1), shiftStartTime (2024-11-05 08:00:00), shiftEndTime (2024-11-05 16:00:00), clockInTime (2024-11-05 08:05:00), clockOutTime (null), and teamID (3, marked as optional). The response section shows a '200 OK' status with a response time of 8 ms and a size of 501 B. The response body is a JSON object with a success message and a data object containing the created shift's details.

POST Create S | POST Add Empl | GET View Empl | GET View All S | GET Shifts For | PUT Update a | PUT Update ai | DEL Delete Sh | DEL Delete Err | + | No environment

CSC_425_Project / Create Shift | Save | Share

POST | http://localhost:5000/api/createShift | Send

Params | Authorization | Headers (9) | Body | Scripts | Tests | Settings | Cookies | Beautify

none | form-data | x-www-form-urlencoded | raw | binary | GraphQL | JSON

```
1 {
2   "employeeID": 1,
3   "shiftStartTime": "2024-11-05 08:00:00",
4   "shiftEndTime": "2024-11-05 16:00:00",
5   "clockInTime": "2024-11-05 08:05:00",
6   "clockOutTime": null,
7   "teamID": 3 // Optional
8 }
```

Body | Cookies | Headers (8) | Test Results | 200 OK | 8 ms | 501 B | Save Response

Pretty | Raw | Preview | Visualize | JSON

```
1 {
2   "message": "Shift created and assigned successfully",
3   "data": {
4     "shiftID": 19,
5     "employeeID": 1,
6     "teamID": 3,
7     "shiftStartTime": "2024-11-05 08:00:00",
8     "shiftEndTime": "2024-11-05 16:00:00",
9     "clockInTime": "2024-11-05 08:05:00",
10    "clockOutTime": null
11  }
12 }
```

Figure 1. Response from the createShift endpoint.

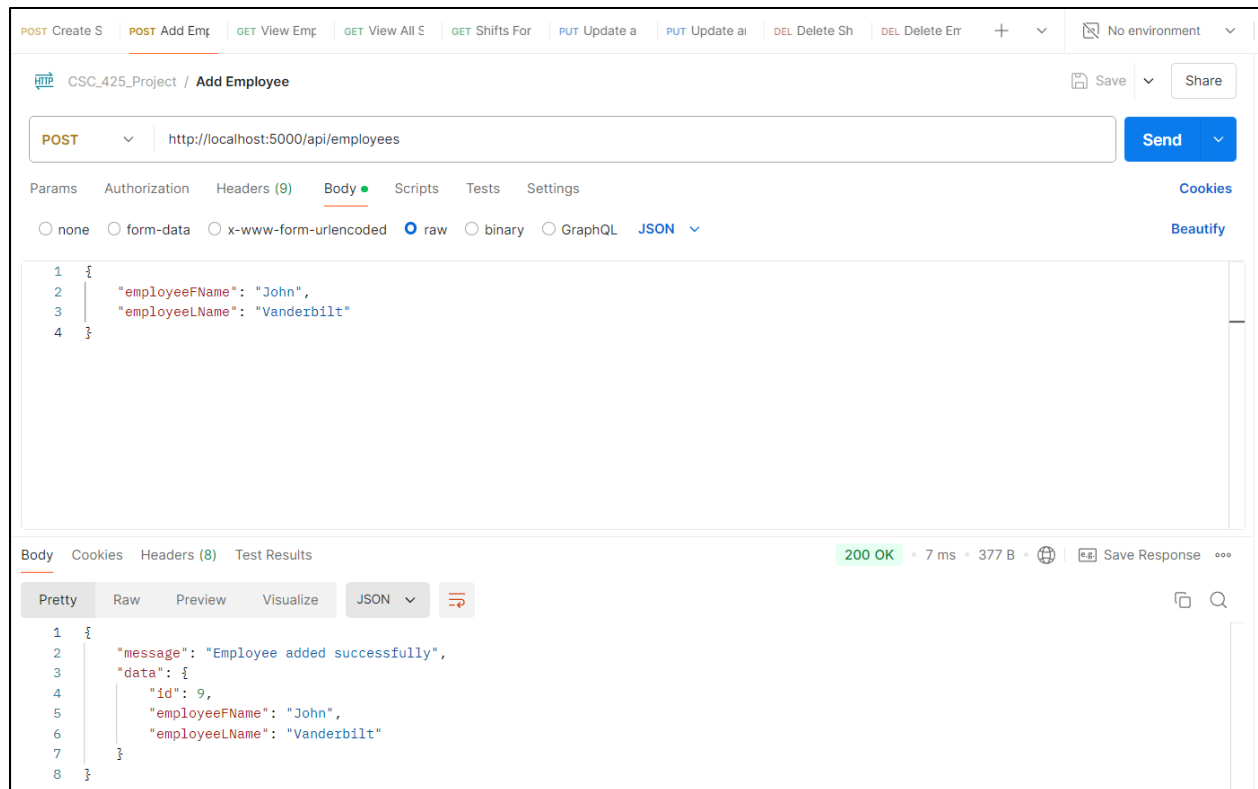


Figure 2. Response from the employees endpoint with post request

READ EXAMPLES (Get)

The screenshot shows a REST client interface with a top bar containing various HTTP methods and their corresponding endpoints. The main area displays a GET request to `http://localhost:5000/api/employees`. Below the request bar, there are tabs for Params, Authorization, Headers (7), Body, Scripts, Tests, and Settings. The Params tab is active, showing a table for Query Params with columns Key, Value, and Description. The Body tab is also active, showing the response body in JSON format. The response status is 200 OK, with a response time of 3 ms and a size of 808 B. The JSON response is as follows:

```
1 {
2   "data": [
3     {
4       "id": 1,
5       "employeeFName": "John",
6       "employeeLName": "Smith"
7     },
8     {
9       "id": 2,
10      "employeeFName": "David",
11      "employeeLName": "Smith"
12    },
13    {
14      "id": 3,
15      "employeeFName": "John",
16      "employeeLName": "Smith"
17    }
18  ]
19 }
```

Figure 3. Response from the employees endpoint as get request

POST Create SPOST Add EmrGET View EmrGET View All SGET Shifts ForPUT Update aPUT Update aDEL Delete ShDEL Delete Err+No environment

CSC_425_Project / View All ShiftsSaveShare

GEThttp://localhost:5000/api/shiftsSend

ParamsAuthorizationHeaders (7)BodyScriptsTestsSettingsCookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

BodyCookiesHeaders (8)Test Results200 OK • 3 ms • 4.64 KBSave Response

PrettyRawPreviewVisualizeJSON

```
1 {
2   "data": [
3     {
4       "shiftID": 1,
5       "employeeID": 1,
6       "employeeFName": "John",
7       "employeeLName": "Smith",
8       "shiftStartTime": "2024-10-31 08:00:00",
9       "shiftEndTime": "2024-10-31 16:00:00",
10      "clockInTime": "2024-10-31 08:01:23",
11      "clockOutTime": "",
12      "teamID": 1,
13      "teamName": "Team 1"
14    }
15  ]
16 }
```

Figure 4. Response from the shifts endpoint as get request.

POST Create S POST Add Em GET View Em GET View All GET Shifts For PUT Update a PUT Update ai DEL Delete Sh DEL Delete Err + No environment

CSC_425_Project / Shifts For Single Employee Save Share

GET http://localhost:5000/api/shifts?employeeID=1 Send

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Query Params

<input checked="" type="checkbox"/> Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/> employeeID	1		
Key	Value	Description	

Body Cookies Headers (8) Test Results 200 OK 3 ms 2.12 KB Save Response

Pretty Raw Preview Visualize JSON

```

12  "teamID": 1,
13  "teamName": "Team 1"
14  },
15  {
16    "shiftID": 8,
17    "employeeID": 1,
18    "employeeFName": "John",
19    "employeeLName": "Smith",
20    "shiftStartTime": "2024-09-31 06:00:00",
21    "shiftEndTime": "2024-09-31 18:00:00",
22    "clockInTime": "",
23    "clockOutTime": "",
24    "teamID": 2,
25    "teamName": "Team 2"
26  },
27  {
28    "shiftID": 11,
29    "employeeID": 1,
30    "employeeFName": "John",
31    "employeeLName": "Smith",
32    "shiftStartTime": "2024-07-13 08:00:00",

```

Figure 5. Response from the shifts endpoint with employee specified.

UPDATE EXAMPLES (Put)

The screenshot shows a Postman interface for a PUT request. The request is sent to `http://localhost:5000/api/shifts/1` with a JSON body containing shift details. The response is a 200 OK status with a success message.

Request:

- Method: PUT
- URL: `http://localhost:5000/api/shifts/1`
- Body (JSON):

```
{  "shiftStartTime": "2024-11-06 09:00:00",  "clockInTime": "2024-11-06 09:05:00"}
```

Response:

- Status: 200 OK
- Time: 7 ms
- Size: 307 B
- Body (JSON):

```
{  "message": "Shift updated successfully"}
```

Figure 6. Response from the `/shifts/{id}` endpoint as a Put request.

The screenshot shows a Postman interface for a PUT request. The request is sent to `http://localhost:5000/api/employees/1` with a JSON body containing employee details. The response is a 200 OK status with a success message and the updated employee data.

Request:

- Method: PUT
- URL: `http://localhost:5000/api/employees/1`
- Body (JSON):

```
{  "employeeLName": "Postman"}
```

Response:

- Status: 200 OK
- Time: 7 ms
- Size: 352 B
- Body (JSON):

```
{  "message": "Employee updated successfully",  "data": {    "id": 9,    "employeeLName": "Postman"  }}
```

Figure 7. Response from /employees/{id} endpoint as a Put request.

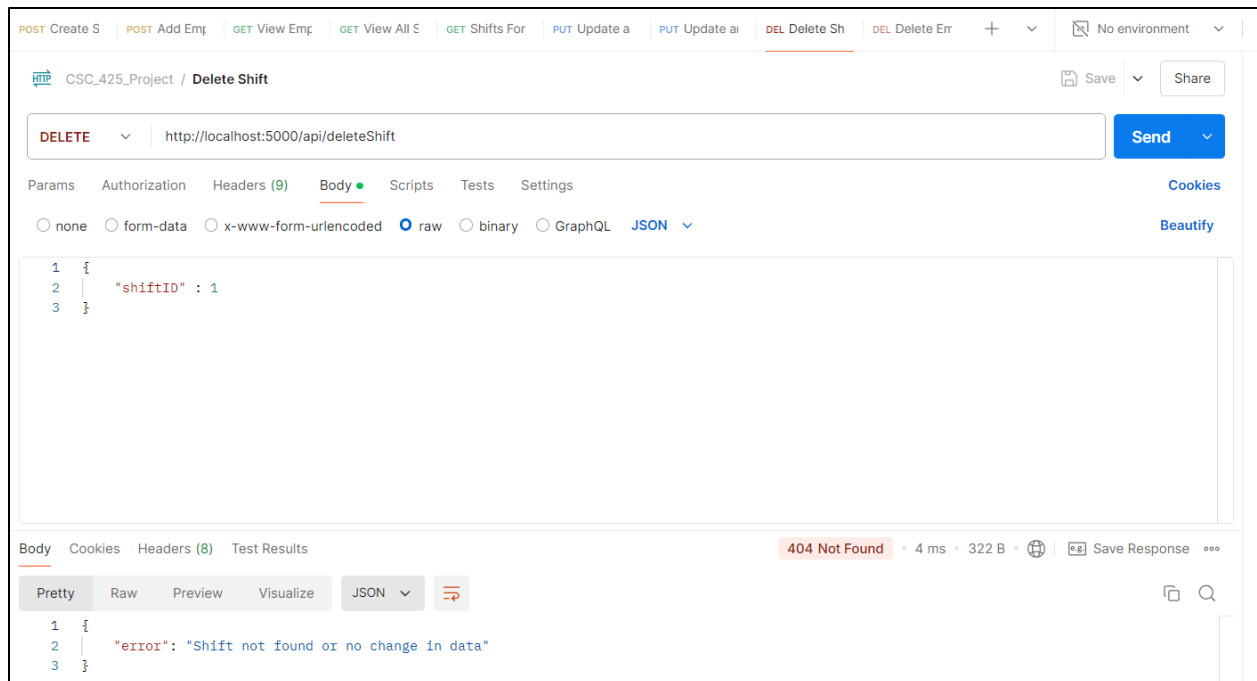
DELETE EXAMPLES (Delete)

The screenshot displays a REST client interface for a project named "CSC_425_Project". The active tab is "Delete Shift", which is a DELETE request to the endpoint `http://localhost:5000/api/deleteShift`. The request body is set to "raw" and contains the following JSON:

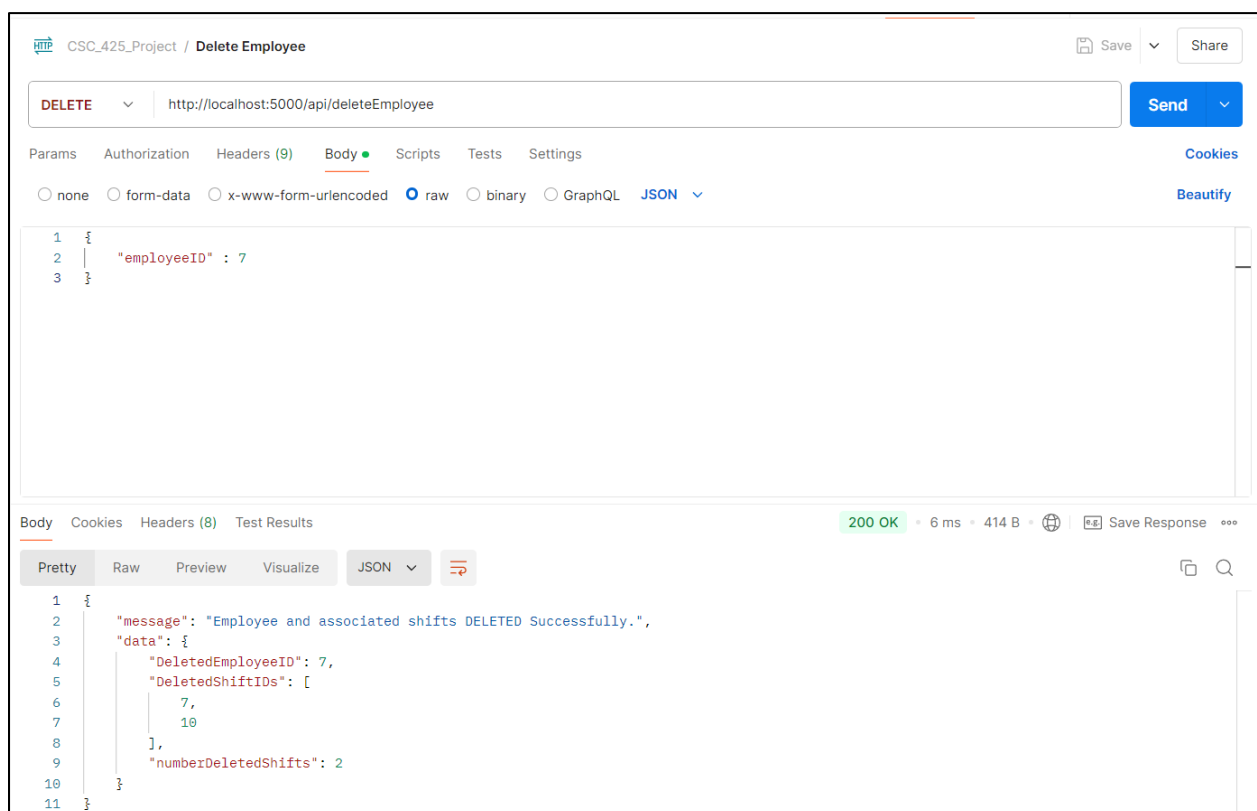
```
1 {
2   "shiftID" : 1
3 }
```

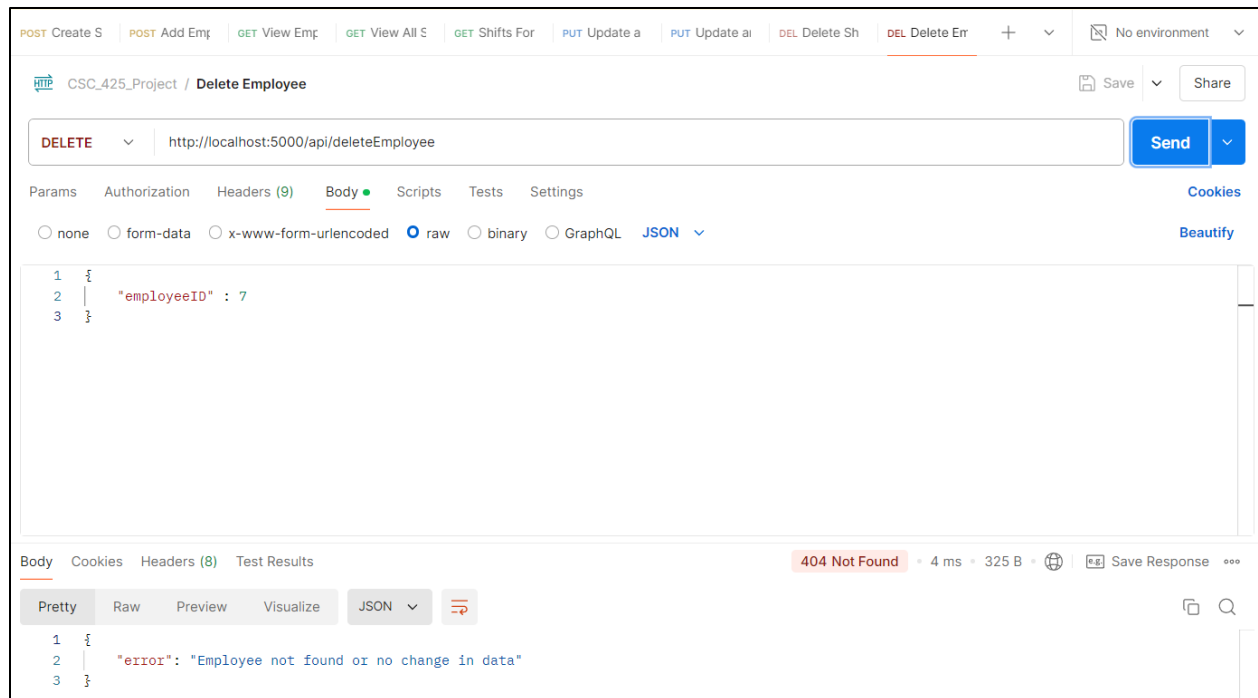
The response status is "200 OK" with a response time of 10 ms and a size of 399 B. The response body is shown in "Pretty" JSON format:

```
1 {
2   "message": "Shift DELETED successfully",
3   "data": {
4     "DeletedShiftID": 1,
5     "numberDeletesAssociativeTable": 1,
6     "numberDeletesShiftsTable": 1
7   }
8 }
```



Figures 8 and 9. Response from the /deleteShift endpoint with valid shift and invalid shift





Figures 10 and 11. Response from the /deleteEmployee endpoint with valid and invalid IDs.