

# JobPortalManualTesting

## Database Init:

### Jobs collection

```
> db.jobs.find()
{ "_id" : ObjectId("6550dc0d7072111a62043920"), "title" : "Software Engineer", "description" : "Job description...", "active" : true, "company" : "CompanyA", "salary" : "8000", "location" : "UK" }
{ "_id" : ObjectId("6550dc0d7072111a62043921"), "title" : "Data Scientist", "description" : "Job description...", "active" : true, "company" : "CompanyB" }
```

### Users collection

```
> db.users.find()
{ "_id" : ObjectId("6550dc0d7072111a62043922"), "email" : "user1@example.com", "firstName" : "John", "lastName" : "Doe", "password" : "user1password", "role" : "user", "phoneNumber" : "88861888", "location" : "Singapore", "education" : [ { "school" : "National University of Singapore", "degree" : "Bachelor of Science in Computer Science" }, { "school" : "Nanyang University", "degree" : "Master of Business Administration" } ], "skills" : [ "Java", "Python", "TypeScript" ] }
{ "_id" : ObjectId("6550dc0d7072111a62043923"), "email" : "user2@example.com", "firstName" : "Jane", "lastName" : "Doe", "password" : "user2password", "phoneNumber" : "88861888", "role" : "user", "education" : [ { "school" : "University of Cambridge", "degree" : "Bachelor of Science in Finance" }, { "school" : "University of Edinburgh", "degree" : "Master of Arts" } ], "skills" : [ "Java", "Python", "TypeScript" ] }
```

### Applications collection

```
> db.applications.find()
{ "_id" : ObjectId("6550dced7072111a62043924"), "jobId" : ObjectId("6550dced7072111a62043920")
, "userId" : ObjectId("6550dced7072111a62043922"), "applicationStatus" : "Pending", "createdAt"
: ISODate("2013-05-16T00:00:00Z") }
{ "_id" : ObjectId("6550dced7072111a62043925"), "jobId" : ObjectId("6550dced7072111a62043921")
, "userId" : ObjectId("6550dced7072111a62043923"), "applicationStatus" : "Accepted", "createdAt"
: ISODate("2023-02-16T00:00:00Z") }
{ "_id" : ObjectId("6550dced7072111a62043926"), "jobId" : ObjectId("6550dced7072111a62043920")
, "userId" : ObjectId("6550dced7072111a62043923"), "applicationStatus" : "Accepted", "createdAt"
: ISODate("2023-02-16T00:00:00Z") }
```

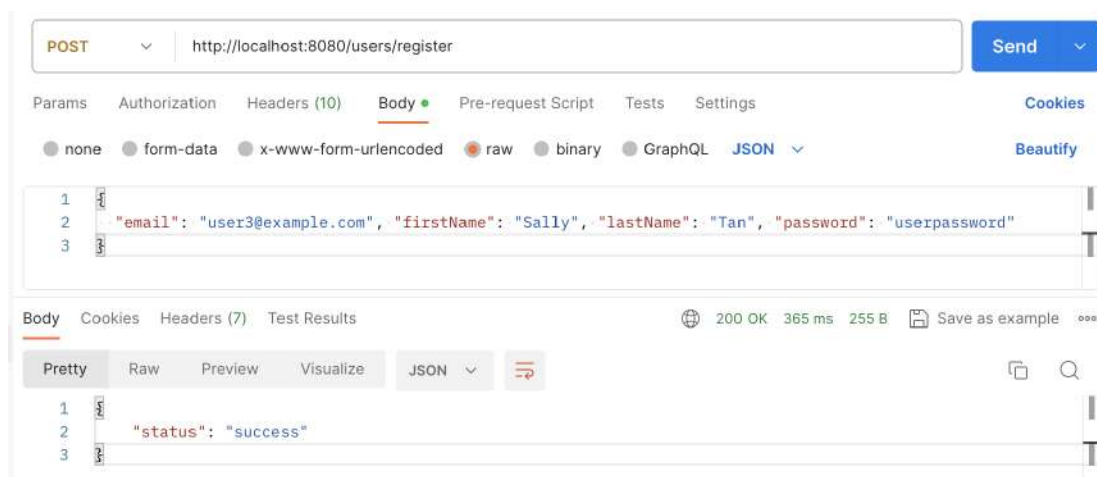
## ▼ User

### ▼ user register

POST <http://localhost:8080/users/register> for user

POST <http://localhost:8080/admin/register> for admin

#### 1. Success 200



users collection updated: user3@example as a user has been inserted

```
> db.users.find()
{ "_id" : ObjectId("6550dced7072111a62043922"), "email" : "user1@example.com", "firstName" : "John", "lastName" : "Doe", "password" : "user1password", "role" : "user", "phoneNumber" : "88861888", "location" : "Singapore", "education" : [ { "school" : "National University of Singapore", "degree" : "Bachelor of Science in Computer Science" }, { "school" : "Nanyang University", "degree" : "Master of Business Administration" } ], "skills" : [ "Java", "Python", "TypeScript" ] }
{ "_id" : ObjectId("6550dced7072111a62043923"), "email" : "user2@example.com", "firstName" : "Jane", "lastName" : "Doe", "password" : "user2password", "phoneNumber" : "88861888", "role" : "user", "education" : [ { "school" : "University of Cambridge", "degree" : "Bachelor of Science in Finance" }, { "school" : "University of Edinburgh", "degree" : "Master of Arts" } ], "skills" : [ "Java", "Python", "TypeScript" ] }
{ "_id" : ObjectId("6550e2f37072111a62043927"), "email" : "user3@example.com", "firstName" : "Sally", "lastName" : "Tan", "password" : "$2a$10$Rmmen30c8moyLTbGzA2/8ewJ7ocSb0yBTc07MB22xSA7iV.1oUuK2", "role" : "user" }
```

## admin register

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/admin/register`. The request body is a JSON object: `{ "email": "admin1@company.com", "firstName": "Ryan", "lastName": "Tan", "password": "userpassword" }`. The response status is 200 OK, and the response body is `{ "status": "success" }`.

users collection updated: admin1@example as a admin user has been inserted

```
> db.users.find()
{ "_id" : ObjectId("6550fba6a8db1a264fbffbb5"), "email" : "user1@example.com", "firstName" : "John", "lastName" : "Doe", "password" : "user1password", "role" : "user", "phoneNumber" : "88861888", "location" : "Singapore", "education" : [ { "school" : "National University of Singapore", "degree" : "Bachelor of Science in Computer Science" }, { "school" : "Nanyang University", "degree" : "Master of Business Administration" } ], "skills" : [ "Java", "Python", "TypeScript" ] }
{ "_id" : ObjectId("6550fba6a8db1a264fbffbb6"), "email" : "user2@example.com", "firstName" : "Jane", "lastName" : "Doe", "password" : "user2password", "phoneNumber" : "88861888", "role" : "user", "education" : [ { "school" : "University of Cambridge", "degree" : "Bachelor of Science in Finance" }, { "school" : "University of Edinburgh", "degree" : "Master of Arts" } ], "skills" : [ "Java", "Python", "TypeScript" ] }
{ "_id" : ObjectId("6550fd85a8db1a264fbffbb7"), "email" : "admin1@company.com", "firstName" : "Ryan", "lastName" : "Tan", "password" : "$2a$10$Y9./y.SdvsmSJNLV0DdLr.wftaE2xqeXayaEMpUaEz0oSjLnt9apK", "role" : "admin" }
```

## 2. 409 Conflict response: user has already registered

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/users/register`. The request body is a JSON object: `{"email": "user3@example.com", "firstName": "Sally", "lastName": "Tan", "password": "userpassword"}`. The response is a 409 Conflict with a status of 113 ms and 298 B. The response body is a JSON object: `{"error_code": 409, "error_message": "email already exists"}`.

```
POST http://localhost:8080/users/register

{"email": "user3@example.com", "firstName": "Sally", "lastName": "Tan", "password": "userpassword"}

409 Conflict 113 ms 298 B

{"error_code": 409, "error_message": "email already exists"}
```

## 3. 400 BAD\_REQUEST: missing parameters

### a. missing password

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/users/register`. The request body is a JSON object: `{"email": "user4@example", "lastname": "Grace", "firstName": "Tan"}`. The response is a 400 Bad Request with a status of 18 ms and 297 B. The response body is a JSON object: `{"error_code": 400, "error_message": "missing password"}`.

```
POST http://localhost:8080/users/register

{"email": "user4@example", "lastname": "Grace", "firstName": "Tan"}

400 Bad Request 18 ms 297 B

{"error_code": 400, "error_message": "missing password"}
```

## b. missing firstName

POST http://localhost:8080/users/register

Params Authorization Headers (10) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL **JSON** ▾

```
1 {
2   "email": "use4@example.com", "lastName": "Tan", "password": "userpassword"
3 }
```

Body Cookies Headers (7) Test Results 400 Bad Request 211 ms

Pretty Raw Preview Visualize **JSON** ▾

```
1 {
2   "error_code": 400,
3   "error_message": "missing parameters or invalid parameter"
4 }
```

## c. missing lastName

POST http://localhost:8080/users/register

Params Authorization Headers (10) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL **JSON** ▾

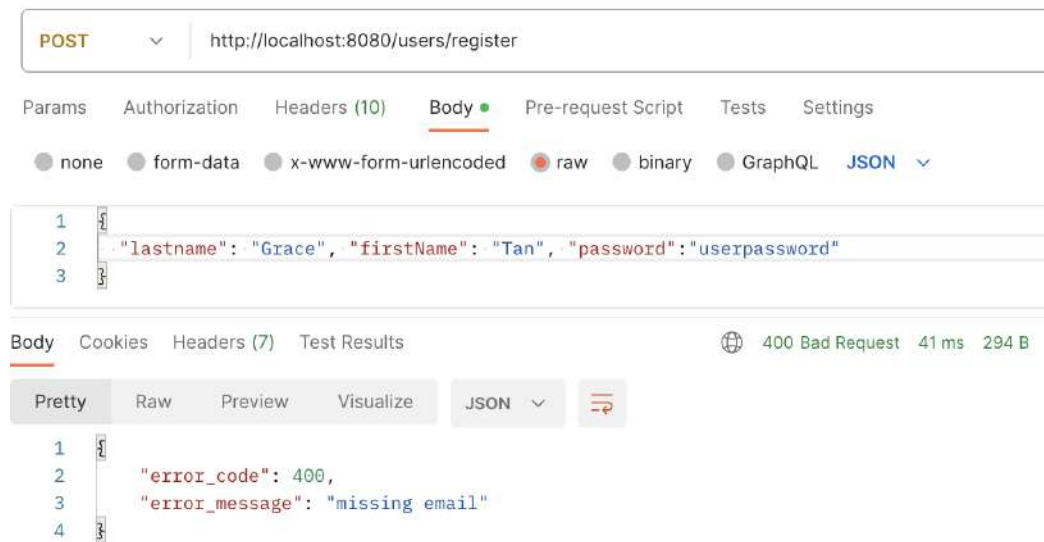
```
1 {
2   "email": "use4@example.com", "firstName": "Tan", "password": "userpassword"
3 }
```

Body Cookies Headers (7) Test Results 400 Bad Request 113 ms

Pretty Raw Preview Visualize **JSON** ▾

```
1 {
2   "error_code": 400,
3   "error_message": "missing parameters or invalid parameter"
4 }
```

#### d. missing email



### ▼ user login

POST `http://localhost:8080/users/login` for user

POST `http://localhost:8080/admin/login` for admin

#### 1. Success 200

user login



POST http://localhost:8080/users/login Send

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

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

```

1
2  {"email": "user3@example.com", "password": "userpassword"}
3

```

Body Cookies Headers (7) Test Results 200 OK 124 ms 463 B Save as example

Pretty Raw Preview Visualize JSON

```

1
2  {"status": "success",
3    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwYX1sb2FkIjoiaW1MGYxOTBhbjViNmUyMzU5MDhhIiwicm9sZSI6InVzZXIiLCJpYXQ10jE2OTk4MDM10TksImV4cCI6MTcwMDIzNTU5OX0.0bQ4fyf34R0s5uFIbKs8ejZriTsd1cfnhbtELjoufDQ"}
4

```

admin user login

POST http://localhost:8080/admin/login Send

Params Authorization Headers (11) Body Pre-request Script Tests Settings Cookies

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

```

1
2  {"email": "admin1@company.com", "password": "userpassword"}
3

```

Body Cookies Headers (7) Test Results 200 OK 180 ms 464 B Save as example

Pretty Raw Preview Visualize JSON

```

1
2  {"status": "success",
3    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwYX1sb2FkIjoiaW1MGZkODVhOGRiMWEyNjRmYmZmYmJhIiwicm9sZSI6ImFkbWluIiwiaWF0IjoxNjk5MDA2ODQzLjJleHAiOjE3MDAyMzg4NDN9.ZX3fLE6Ik9P3RVk6IpzVO0parIX6A3wsMczLE0GV3b0"}
4

```

## 2. 400 BAD\_REQUEST: missing parameters or invalid parameters

### a. missing password

POST http://localhost:8080/users/login

Params Authorization Headers (10) Body Pre-request Script Tests Settings

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

```
1  
2 "email": "user3@example.com", "password": ""  
3
```

Body Cookies Headers (7) Test Results 400 Bad Request 30 ms 307 B

Pretty Raw Preview Visualize JSON

```
1  
2 "error_code": 400,  
3 "error_message": "invalid Email or password."  
4
```

b. missing password

POST http://localhost:8080/users/login

Params Authorization Headers (10) Body Pre-request Script Tests Settings

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

```
1  
2 "email": "", "password": "userpassword"  
3
```

Body Cookies Headers (7) Test Results 400 Bad Request 30 ms

Pretty Raw Preview Visualize JSON

```
1  
2 "error_code": 400,  
3 "error_message": "invalid Email or password."  
4
```

c. user doesn't exist



POST ▼ http://localhost:8080/users/login

Params Authorization ● Headers (10) **Body** ● Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL **JSON** ▼

```
1 {
2   "email": "user100@example.com", "password": "userpassword"
3 }
```

Body Cookies Headers (7) Test Results 🌐 400 Bad Request 24 ms

Pretty Raw Preview Visualize **JSON** ▼ ≡

```
1 {
2   "error_code": 400,
3   "error_message": "this user does not exist"
4 }
```

d. incorrect password

POST ▼ http://localhost:8080/users/login

Params Authorization ● Headers (10) **Body** ● Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL **JSON** ▼

```
1 {
2   "email": "user3@example.com", "password": "wrongpassword"
3 }
```

Body Cookies Headers (7) Test Results 🌐 400 Bad Request 105 ms

Pretty Raw Preview Visualize **JSON** ▼ ≡

```
1 {
2   "error_code": 400,
3   "error_message": "incorrect password"
4 }
```

## ▼ User get profile

GET http://localhost:8080/users/profile

1. Success 200

GET http://localhost:8080/users/profile

Params Authorization Headers (12) **Body** Pre-request Script Tests Settings

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

```
1 {
2   "userId": "6550f6bf198353250cd2ced6"
3 }
4
```

Body Cookies Headers (7) Test Results 200 OK 14 ms 423 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "_id": "6550f6bf198353250cd2ced6",
3   "email": "user3@example.com",
4   "firstName": "Sally",
5   "lastName": "Tan",
6   "password": "$2a$10$fd7sxjpR1nkHpZdET3ea9.BNWs0mfN50HY6hEdFlqsdZfD4oA6bFG",
7   "role": "user"
8 }
```

## 2. 400 BAD\_REQUEST: missing parameters or invalid parameters

### a. missing userId

GET http://localhost:8080/users/profile

Params Authorization Headers (12) **Body** Pre-request Script Tests Settings

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

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

Body Cookies Headers (7) Test Results 400 Bad Request

Pretty Raw Preview Visualize JSON

```
1 {
2   "error_code": 400,
3   "error_message": "missing user id"
4 }
```

### b. invalid userId

GET http://localhost:8080/users/profile

Params Authorization Headers (12) **Body** Pre-request Script Tests Settings

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

```
1 {
2   "userId": "6725183668126"
3 }
```

Body Cookies Headers (7) Test Results 400 Bad Request 6 ms

Pretty Raw Preview Visualize JSON

```
1 {
2   "error_code": 400,
3   "error_message": "invalid user id"
4 }
```

### c. user not found

GET http://localhost:8080/users/profile

Params Authorization Headers (12) **Body** Pre-request Script Tests Settings

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

```
1 {
2   "userId": "6550f62e882f6e24ca282716"
3 }
```

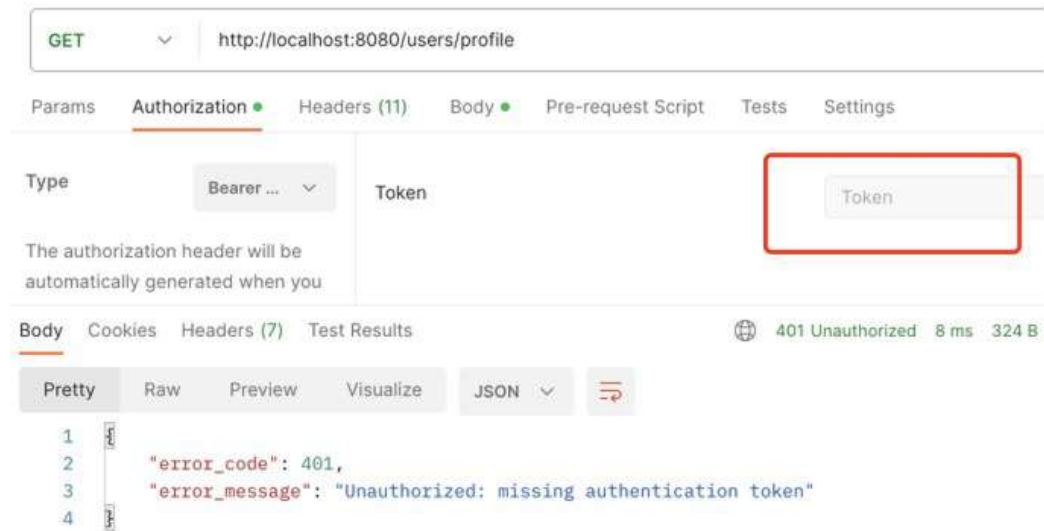
Body Cookies Headers (7) Test Results 400 Bad Request 6 ms

Pretty Raw Preview Visualize JSON

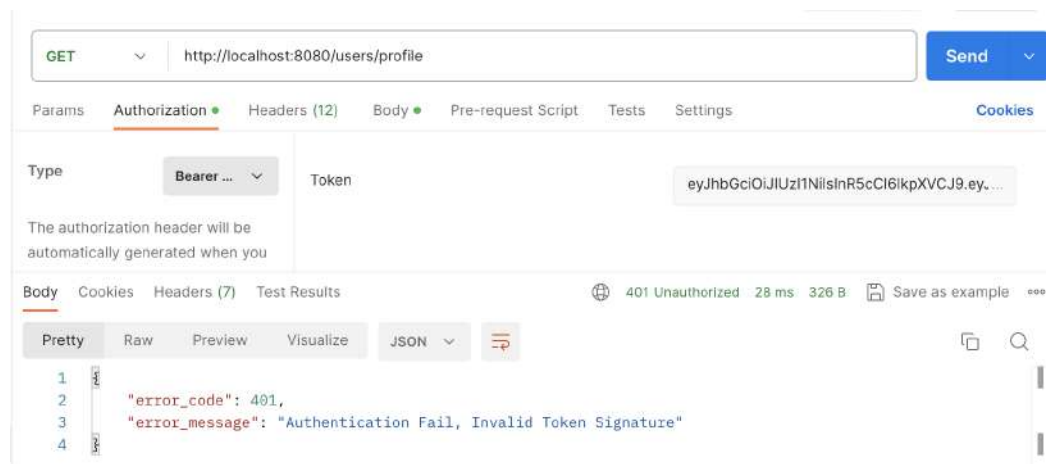
```
1 {
2   "error_code": 400,
3   "error_message": "User Not Found"
4 }
```

## 3. 401 UNAUTHORIZED: missing token

a. send request without token



b. send request with invalid token

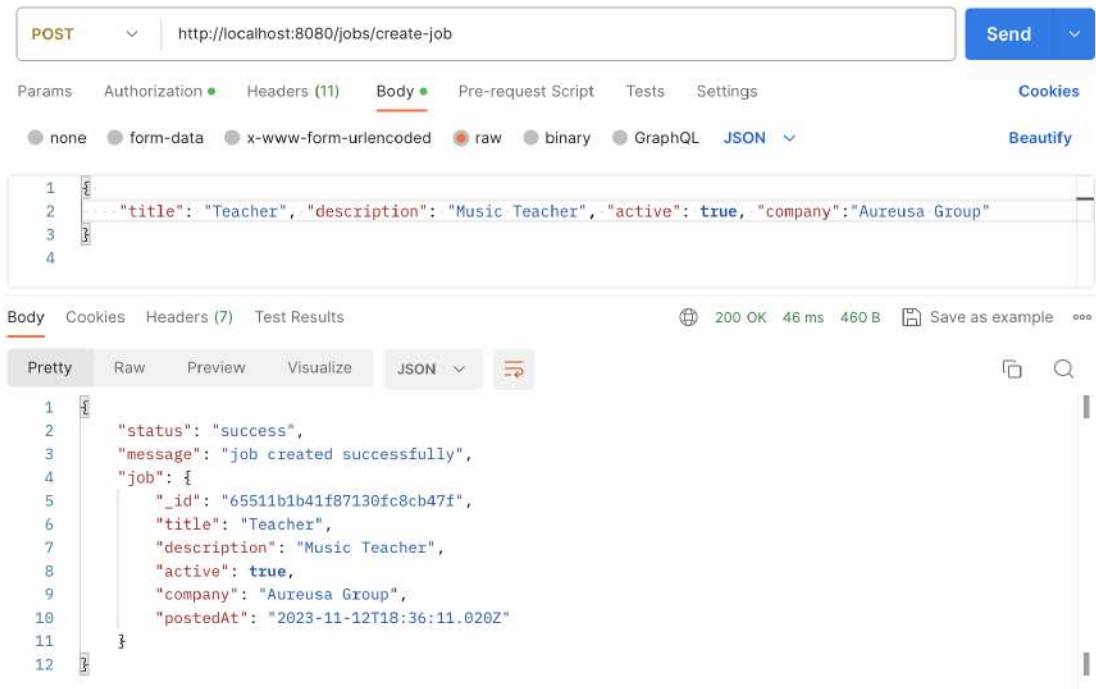


## ▼ Job

## ▼ Admin user create a job

POST <http://localhost:8080/jobs/create-job>

### 1. **Success 200** : admin create a job



The screenshot displays a REST client interface with the following details:

- Method:** POST
- URL:** <http://localhost:8080/jobs/create-job>
- Body:** A JSON object: `{ "title": "Teacher", "description": "Music Teacher", "active": true, "company": "Aureusa Group" }`
- Response:** A 200 OK status with a response body: `{ "status": "success", "message": "job created successfully", "job": { "_id": "65511b1b41f87130fc8cb47f", "title": "Teacher", "description": "Music Teacher", "active": true, "company": "Aureusa Group", "postedAt": "2023-11-12T18:36:11.020Z" } } }`

### 2. **400 BAD\_REQUEST**: missing parameters or invalid parameters

POST http://localhost:8080/jobs/create-job

Params Authorization Headers (11) Body Pre-request Script Tests Settings

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

```
1 {
2   "description": "Music Teacher", "active": true, "company": "Aureusa Group"
3 }
4
```

Body Cookies Headers (7) Test Results 400 Bad Request 15 ms

Pretty Raw Preview Visualize JSON

```
1 {
2   "error_code": 400,
3   "error_message": "Missing Parameters or Invalid Parameter"
4 }
```

POST http://localhost:8080/jobs/create-job

Params Authorization Headers (11) Body Pre-request Script Tests Settings

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

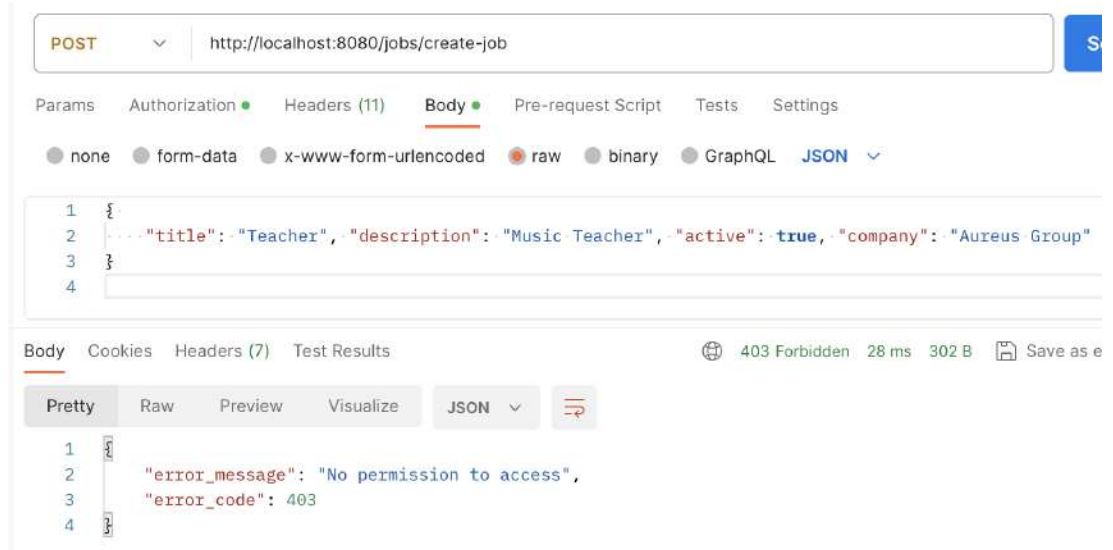
```
1 {
2   "title": "Teacher", "description": "Music Teacher", "active": true
3 }
4
```

Body Cookies Headers (7) Test Results 400 Bad Request 32 ms

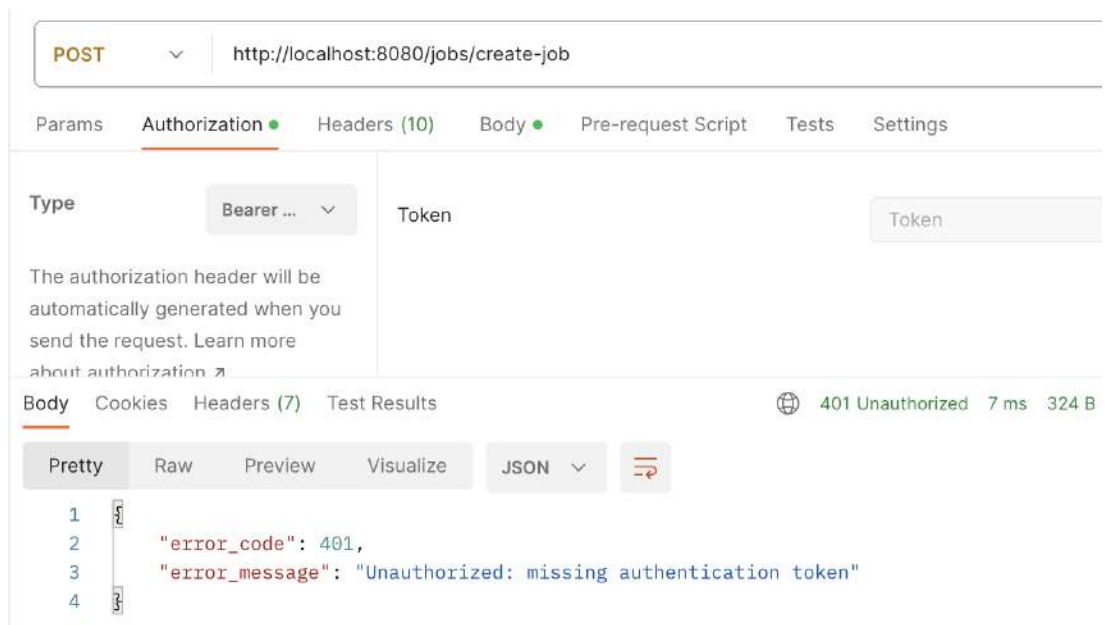
Pretty Raw Preview Visualize JSON

```
1 {
2   "error_code": 400,
3   "error_message": "Missing Parameters or Invalid Parameter"
4 }
```

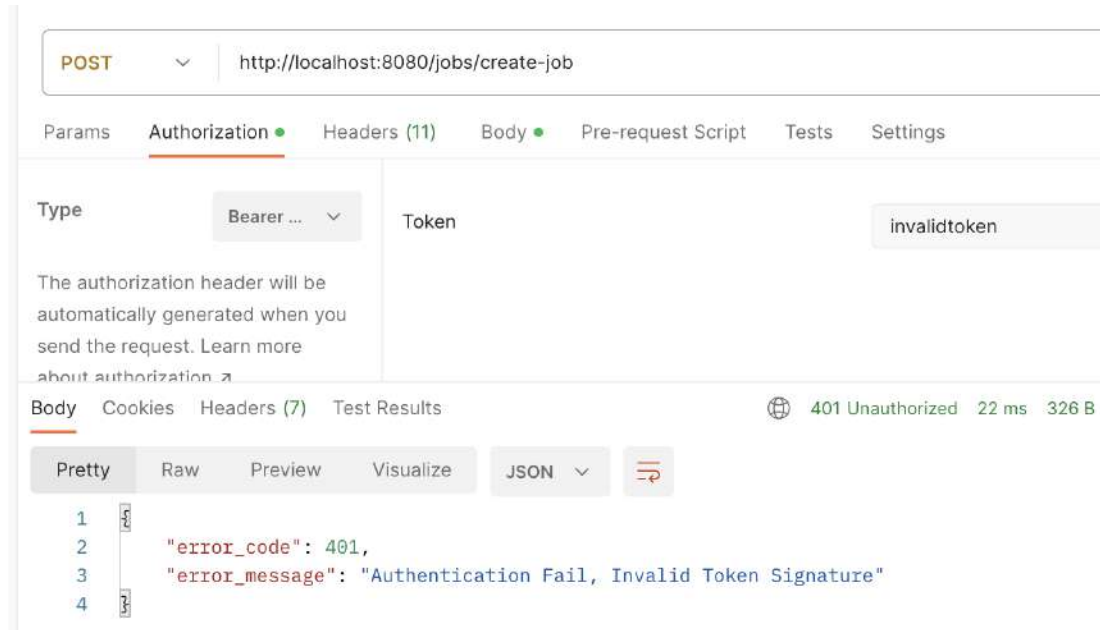
### 3. 403 FORBIDDEN: user is not allowed to create jobs



#### 4. **401 UNAUTHORIZED**: invalid token or missing token



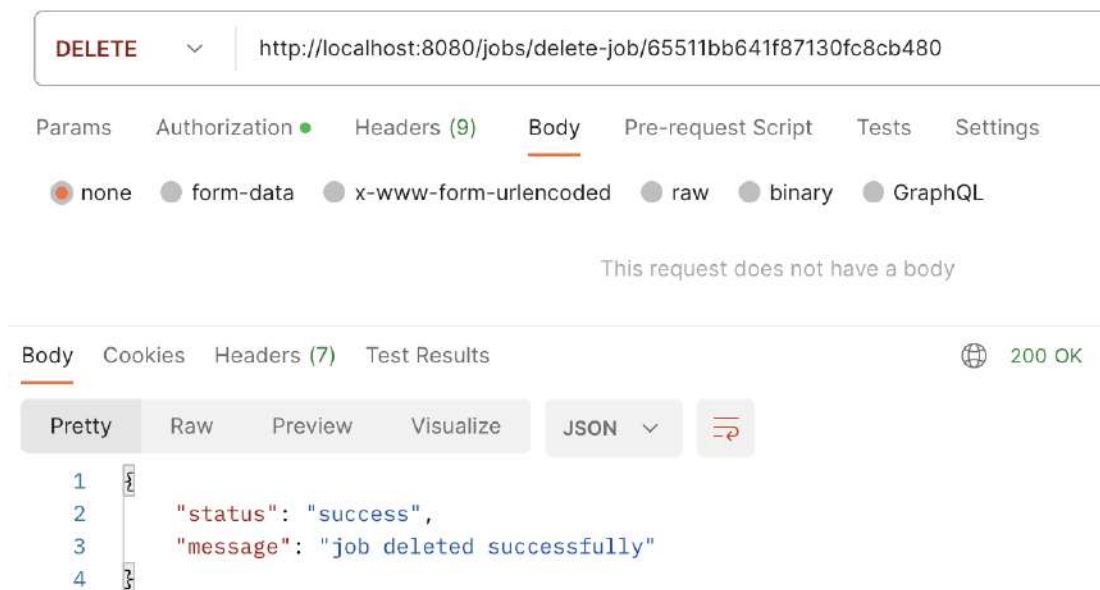




## ▼ Admin user delete a job

DELETE <http://localhost:8080/jobs/delete-job/:id>

### 1. Success 200 : admin delete a job



## 2. 400 BAD\_REQUEST

### a. invalid jobId

DELETE ▼ | http://localhost:8080/jobs/delete-job/12345678

Params Authorization ● Headers (9) **Body** Pre-request Script Tests Settings

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

This request does not have a body

Body Cookies Headers (7) Test Results 🌐 400 Bad Request

Pretty Raw Preview Visualize JSON ▼ 🔍

```
1 {
2   "error_code": 400,
3   "error_message": "invalid job id"
4 }
```

### b. jobId doesn't exist

DELETE ▼ | http://localhost:8080/jobs/delete-job/65511bb641f87130fc8cb480

Params Authorization ● Headers (9) **Body** Pre-request Script Tests Settings

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

This request does not have a body

Body Cookies Headers (7) Test Results 🌐 400 Bad Request

Pretty Raw Preview Visualize JSON ▼ 🔍

```
1 {
2   "error_code": 400,
3   "error_message": "this job does not exist"
4 }
```

### 3. **401 UNAUTHORIZED**: invalid token or missing token

The screenshot shows a REST client interface with the following details:

- Method:** DELETE
- URL:** http://localhost:8080/jobs/delete-job/65511bb641f87130fc8cb480
- Authorization:** Bearer ... (Token field is empty)
- Status:** 401 Unauthorized (4 ms)
- Body:** Pretty view showing JSON:

```
1 {
2   "error_code": 401,
3   "error_message": "Unauthorized: missing authentication token"
4 }
```

### 4. **403 FORBIDDEN**: user is not allowed to create jobs

The screenshot shows a REST client interface with the following details:

- Method:** DELETE
- URL:** http://localhost:8080/jobs/delete-job/65511bb641f87130fc8cb480
- Authorization:** Bearer ... (Token field contains: eyJhbGciOiJIUzI1NiIsInR...
- Status:** 403 Forbidden (15 ms, 302 B)
- Body:** Pretty view showing JSON:

```
1 {
2   "error_code": 403,
3   "error_message": "No permission to access"
4 }
```

## ▼ User get job profile

GET `http://localhost:8080/jobs/job-data:id`

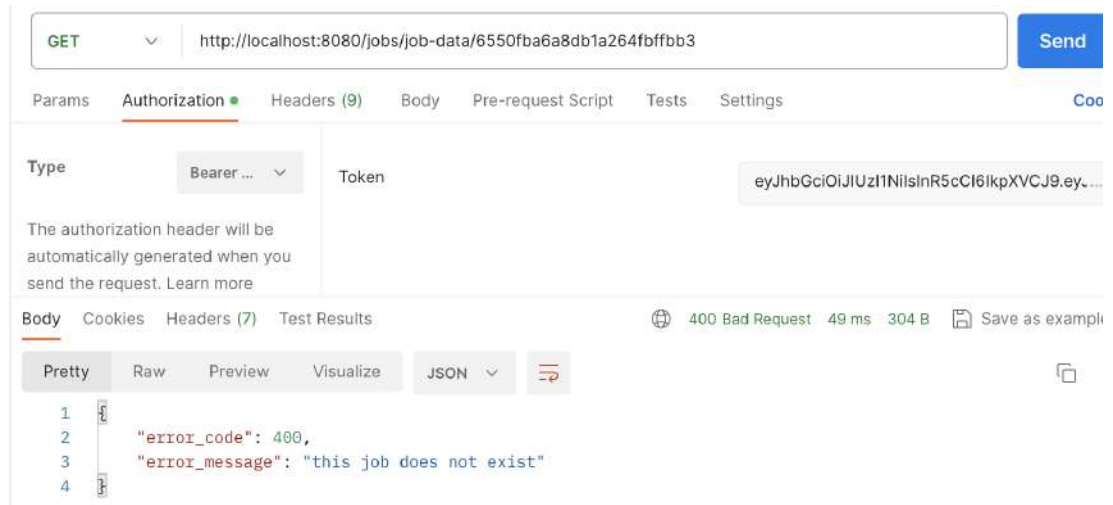
### 1. **Success 200** : user view job details

The screenshot shows a REST client interface with the following details:

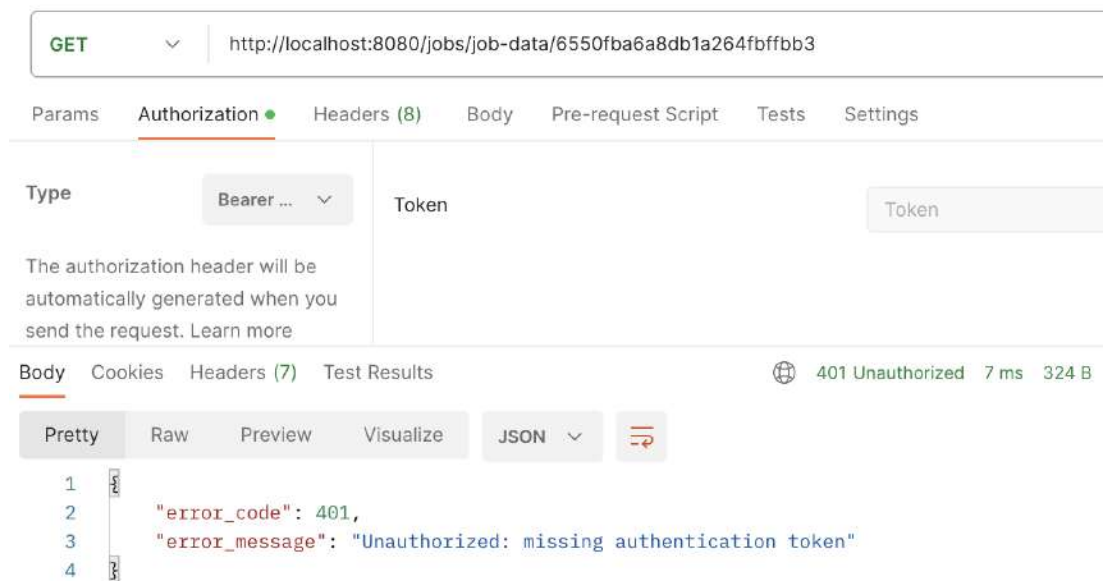
- Method:** GET
- URL:** `http://localhost:8080/jobs/job-data/655100f3ee14b527a4e17a1f`
- Authorization:** Bearer token `eyJhbGciOiJIUzI`
- Body:** JSON response showing job details:

```
1 {
2   "_id": "655100f3ee14b527a4e17a1f",
3   "title": "Data Scientist",
4   "description": "Job description...",
5   "active": true,
6   "company": "CompanyB"
7 }
```
- Status:** 200 OK, 32 ms

### 2. **400 BAD\_REQUEST**: invalid jobId, user requests to view a non existing job



### 3. **401 UNAUTHORIZED:** missing token



### 4. **403 FORBIDDEN:** user is not allowed to take this action

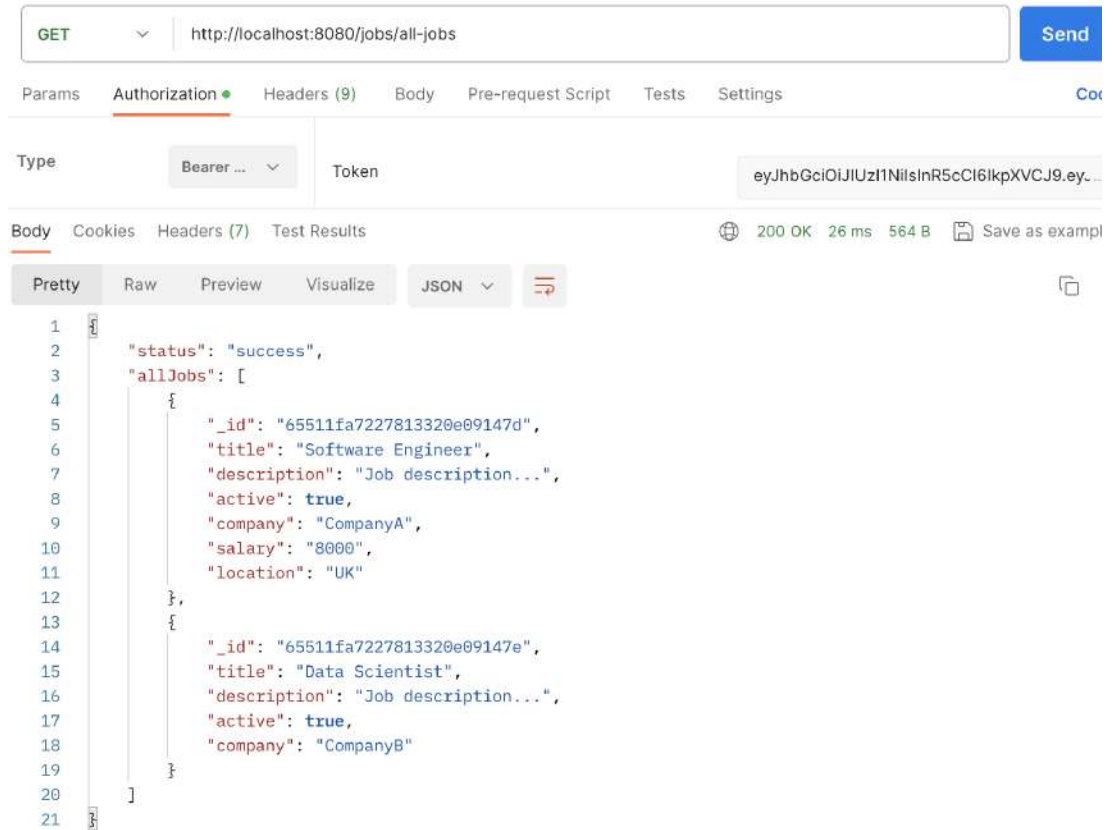
admin wants to view job details



## ▼ Admin User or User get the jobs list

GET `http://localhost:8080/jobs/all-jobs`

1. **Success 200** : admin get all jobs



## ▼ JobApplication

### ▼ user apply for a new job

POST <http://localhost:8080/applications/create-application>

#### 1. Success 200



POST http://localhost:8080/applications/create-application

Params Authorization Headers (11) Body Pre-request Script Tests Settings

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

```
1 {
2   "jobId": "655144233b62444d11c7f1b0", "userId": "655144463b62444d11c7f1b6"
3 }
4
```

Body Cookies Headers (7) Test Results 200 OK 37 ms 466 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "status": "success",
3   "message": "application send",
4   "job": {
5     "_id": "655146d42513574db004c7d5",
6     "jobId": "655144233b62444d11c7f1b0",
7     "userId": "655144463b62444d11c7f1b6",
8     "applicationStatus": "applied",
9     "createdAt": "2023-11-12T21:42:44.594Z"
10  }
11 }
```

## 2. 409 Conflict response: this user has already applied this job

POST http://localhost:8080/applications/create-application

Params Authorization Headers (11) Body Pre-request Script Tests Settings

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

```
1 {
2   "jobId": "655144233b62444d11c7f1b0", "userId": "655144463b62444d11c7f1b6"
3 }
4
```

Body Cookies Headers (7) Test Results 409 Conflict 17 ms 293 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "error_code": 409,
3   "error_message": "already applied"
4 }
```

### 3. 400 BAD\_REQUEST: missing parameters

#### a. missing userId

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/applications/create-application`. The request body is a JSON object with a `jobId` field. The response is a 400 Bad Request error with the message "missing userId".

**Request:**

```
POST http://localhost:8080/applications/create-application
```

**Body:**

```
{  "jobId": "655144233b62444d11c7f1b0"}
```

**Response:**

```
{  "error_code": 400,  "error_message": "missing userId"}
```

#### b. missing jobId

POST http://localhost:8080/applications/create-application

Params Authorization Headers (11) Body Pre-request Script Tests Settings

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

```
1 {
2   ... "userId": "655144463b62444d11c7f1b6"
3 }
4
```

Body Cookies Headers (7) Test Results 400 Bad Request 6 ms

Pretty Raw Preview Visualize JSON

```
1 {
2   "error_code": 400,
3   "error_message": "missing jobId"
4 }
```

c. invalid userId

POST http://localhost:8080/applications/create-application

Params Authorization Headers (11) Body Pre-request Script Tests Settings

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

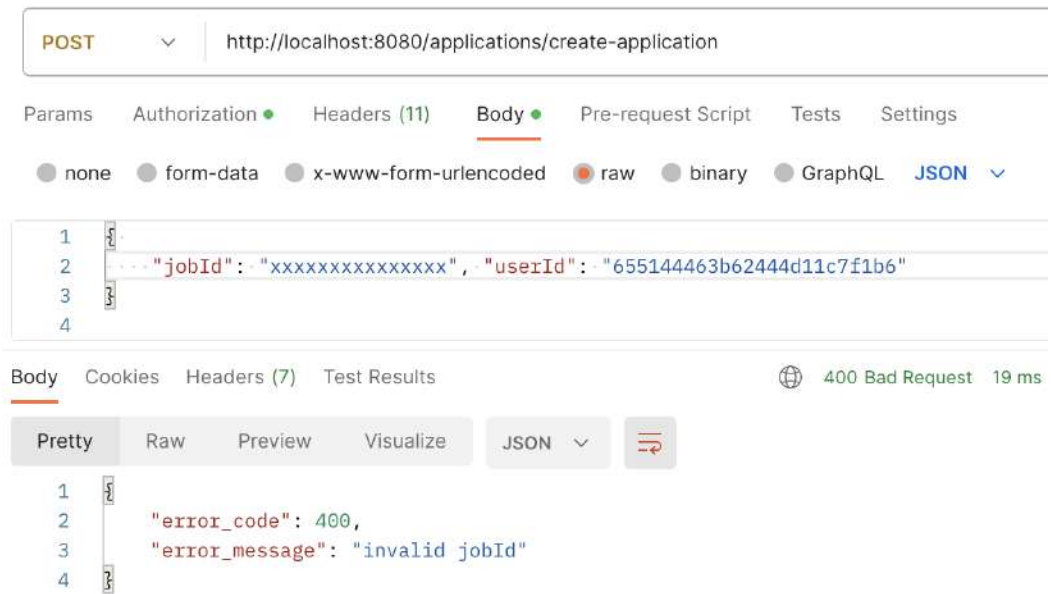
```
1 {
2   ... "jobId": "655144233b62444d11c7f1b0", "userId": "123wefwerwerwwwccccc"
3 }
4
```

Body Cookies Headers (7) Test Results 400 Bad Request 8 ms

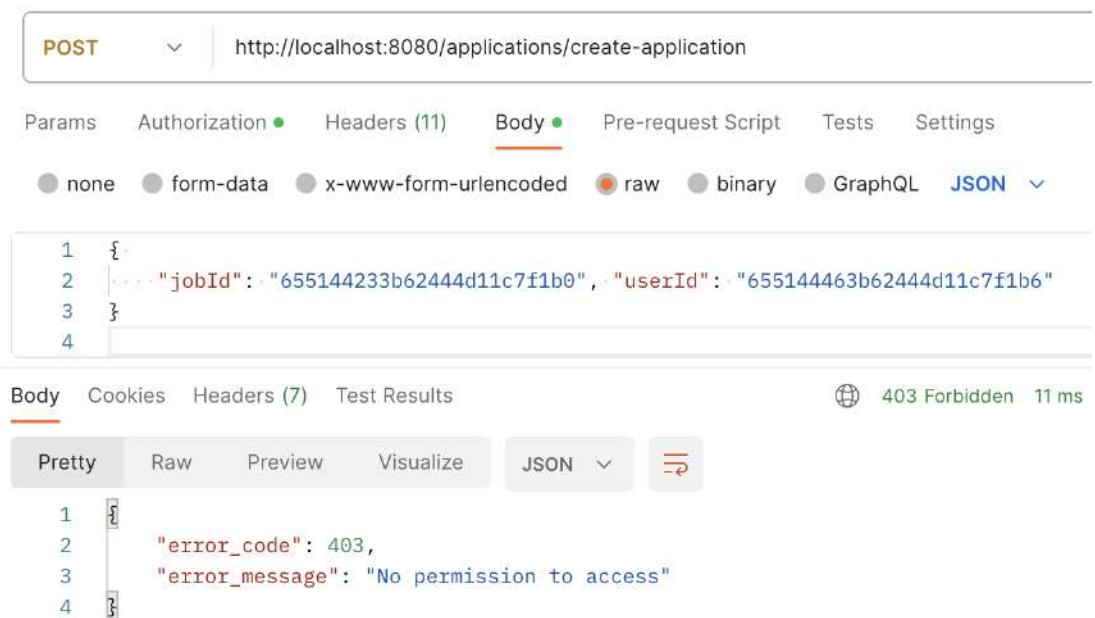
Pretty Raw Preview Visualize JSON

```
1 {
2   "error_code": 400,
3   "error_message": "invalid userId"
4 }
```

d. invalid jobId



#### 4. **403 FORBIDDEN:** admin user is not allowed to apply for jobs



### ▼ user view all applications

GET <http://localhost:8080/applications/user-applications>

1. **Success 200**

Users has applied for jobs


**GET** <http://localhost:8080/applications/user-applications>

Params Authorization Headers (11) Body Pre-request Script

☐ none
 ☐ form-data
 ☐ x-www-form-urlencoded
 ☒ raw
 ☐ binary

```
1 }
2 "userId": "655144463b62444d11c7f1b6"
```

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON 

```
1  {
2    "userApplications": [
3      {
4        "_id": "6551449b3b62444d11c7f1b9",
5        "jobInfo": {
6          "jobId": "655144233b62444d11c7f1af",
7          "title": "Software Engineer",
8          "location": "UK",
9          "company": "CompanyA"
10       },
11       "applicationStatus": "applied",
12       "createdAt": "2023-11-12T21:33:15.686Z"
13     },
14     {
15       "_id": "655144ed3b62444d11c7f1ba",
16       "jobInfo": {
17         "jobId": "655144233b62444d11c7f1b0",
18         "title": "Data Scientist",
19         "company": "CompanyB"
20       },
21       "applicationStatus": "applied",
22       "createdAt": "2023-11-12T21:34:37.990Z"
23     }
24   ],
25   "message": "get user's applications"
```

Users hasn't applied for jobs yet

GET http://localhost:8080/applications/user-applications

Params Authorization Headers (11) Body Pre-request Script Tests Settings

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

```
1 {
2   "userId": "6551444c3b62444d11c7f1b7"
3 }
4
```

Body Cookies Headers (7) Test Results 200 OK

Pretty Raw Preview Visualize JSON

```
1 {
2   "userApplications": null,
3   "message": "this user hasn't applied for any jobs"
4 }
```

## 2. 403 FORBIDDEN: admin user is not allowed to view applied jobs

GET http://localhost:8080/applications/user-applications

Params Authorization Headers (11) Body Pre-request Script Tests Settings

Type Bearer ... Token eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXLT...

The authorization header will be automatically generated when you send the request. Learn more

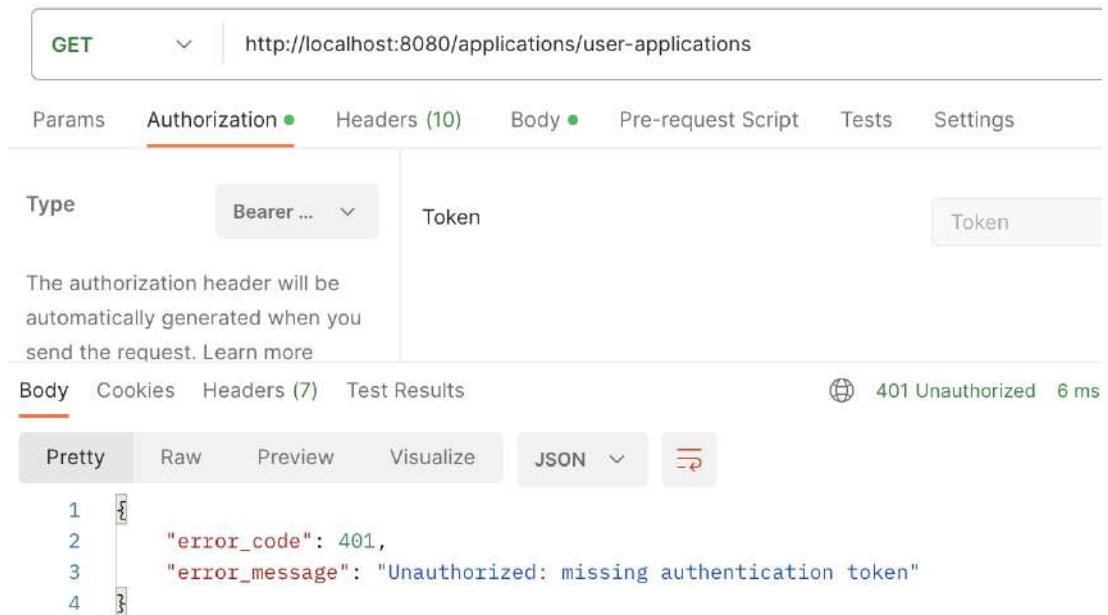
Body Cookies Headers (7) Test Results 403 Forbidden 7 ms 302 B

Pretty Raw Preview Visualize JSON


```
1 {
2   "error_code": 403,
3   "error_message": "No permission to access"
4 }
```











#### 4. **401 UNAUTHORIZED**: invalid token or missing token




#### 5. **400 BAD\_REQUEST**: invalid userId



GET  http://localhost:8080/applications/user-applications

Params Authorization  Headers (11) Body  Pre-request Script Tests Settings

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

```
1 {  
2   "userId": "1234hdhihw"  
3 }  
4
```

**Body** Cookies Headers (7) Test Results  400 Bad Request

Pretty Raw Preview Visualize **JSON**  

```
1 {  
2   "error_code": 400,  
3   "error_message": "invalid userId"  
4 }
```