

Hands-on Lab - Querying with GraphQL



Estimated Time: 60 minutes

In this lab, you will become familiar with using GraphQL to query the APIs served through a GraphQL service running on a server using Postman.

Learning Objectives:

After completing this exercise, you should be able to perform the following tasks:

- Use GraphQL with Postman to access GraphQL Service
- Structure queries to access APIs to get selective information that you need

Pre-requisites

- You must be familiar with Docker applications and commands.
- You must have a good understanding of API.
- You must be familiar with using Postman.

Task 1 - Sign-up for Postman

If you already have a Postman account, you can skip this task.

1. On your browser, go to www.postman.com

←

→

↺

🔒

postman.com

Product

▼

Pricing

Enterprise

▼

Resources and Support

▼

Explore

Home

Workspaces

▼

Reg

Twitter's Public Workspace

New

In

+

≡

Collections

+

≡

APIs

+

≡

Environments

+

≡

Mock Servers

+

≡

Monitors

+

≡

History

+

≡

Twitter API v2

Fork label

▼

Tweet Lookup

▼

GET Single Tweet

200 Success - R

200 Success - R

200 Success - R

200 Success - R

200 Success - D

429 Rate Limit E

200 Success - D

GET Single Tweet Usercont

GET Multiple Tweets

User Lookup

Follows

Blocks

Likes

Timelines

Hide Replies

Search Tweets

Filtered Streams

Sampled Streams

Build APIs together

Over 20 million developers use Postman. Get started by signing up or downloading the desktop app.

jsmith@example.com

Sign Up for Free

Download the desktop app

Windows

Mac

Linux

2. Click **Sign Up for Free**.
3. Enter your email and choose a password, confirm it, and create a free account. You can alternatively use your Google account to sign in too. You may notice that **Stay signed in for 30 days** is selected by default. It is highly recommended that you uncheck that option if you are not using a personal computer for this lab.

about:blank

1/8



Why sign up?

- Organize all your API development within Postman Workspaces
- Sync your Postman data across devices
- Backup your data to the Postman cloud
- It's free!



You have successfully created a Postman account.

Task 2 - Run a GraphQL service

1. Open a terminal window by using the top menu in the IDE: **Terminal > New Terminal**.

2. In the terminal, clone the repository which has the GraphQL service code ready by pasting the following command. The repository that you clone has code that will run a GraphQL service where we can query details on US cities and states.

1. 1

```
1. git clone https://github.com/ibm-developer-skills-network/jmgo-microservices.git
```

Copied!

3. Change the working directory to **jmgo-microservices/graphql_example** by running the following command.

1. 1

```
1. cd jmgo-microservices/graphql_example
```

Copied!

4. Build the application by running the following command. This will build the docker application using the docker file in the current directory and tag it **graphqlservice**.

1. 1

```
1. docker build . -t graphqlservice
```

Copied!

5. Run the application that you just built on port 8080, by running the following command on the terminal.

1. 1

```
1. docker run -dp 8080:4000 graphqlservice
```

Copied!

Create Postman Account [Sign In](#) instead?

Email

Username

Password

[SHOW](#)

☐ Sign up to get product updates, news, and other marketing communications.

☒ Stay signed in for 30 days

By creating an account, I agree to the [Terms](#) and [Privacy Policy](#).

Create free account

or



Sign up with Google

Sign in with SSO

You will get a hex code that indicates the application has started.

6. To confirm that the service is running, execute the following command.

1. 1

1. curl localhost:8080

Copied!

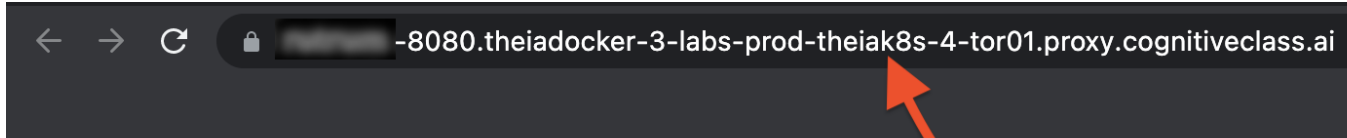
```
theia@theiadocker- /home/project/jmgdo-microservices$ curl localhost:8080
```

Copy the URL from the address-bar, to paste in Postman to use GraphQL

If the output is as seen in the image above, your service is up and running.

7. Click **Launch Application** from the top menu and enter port number *8080*.

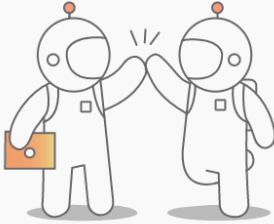
8. This will launch a browser page. Copy the URL that you see in the address bar.



Copy the URL from the address-bar, to paste in Postman to use GrpahQL

Task 3 - Run GraphQL queries on Postman

1. Go to www.postman.com and sign in to Postman. Please note that the home page may look different for every person depending on the day of the year and the region you are signing in from.
2. On the homepage, click **Create new**.



Postman works best with teams

Collaborate in real-time and establish a single source of truth for all API workflows.

Create Team

Workspaces >


Integrations >

Reports >

Good morning!

Pick up where you left off.

Recently visited workspaces



My Workspace

Get started with Postman

Start with something new

Create a new request, collection, or API in a workspace

Create New →

Im

Im

loc

Im



3. This brings up the set of options you can use with Postman. Choose **HttpRequest**.

The screenshot shows the Postman web interface. The top navigation bar includes 'Home', 'Workspaces', 'API Network', and 'Explore'. The main area displays 'My Workspace' with a sidebar on the left containing icons for Collections, APIs, Environments, Mock Servers, Monitors, Flows, and History. A 'Create New' modal is open in the center, listing various options:

- Building Blocks**
 - HTTP Request**: Create a basic HTTP request (icon: GET)
 - Collection**: Save your requests in a collection for reuse and sharing (icon: folder)
 - WebSocket Re**: Test and debug your connections (icon: handshake)
 - Environment**: Save values you use for an environment (icon: document)
- Advanced**
 - API Documentation**: Create and publish beautiful documentation for your APIs (icon: document)
 - Mock Server**: Create a mock server for development APIs (icon: server rack)
 - API**: Manage all aspects of API design, development, and testing (icon: hexagons)
 - Flows BETA**: Create API workflows connecting series through a drag-and-drop (icon: flowchart)

At the bottom of the modal, it says: 'Not sure where to start? [Explore](#) featured APIs, collections, and workspaces page'.

The background interface shows a workspace with a collection named 'My first collection' containing two folders: 'First folder inside collection' and 'Second folder inside collection'. Each folder contains a 'GET' request. A 'Create collection for your workspace' button is visible in the background.

4. In the space provided for the **GET** request URL paste the URL you copied from step 8 of the previous task and suffix it with **/graphql**.

4/18/23, 9:30 AM

about:blank

Overview

GET https://lavanyas-8080.

https://lavanyas-8080.theiadocker-3-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/graphql

GET

https://lavanyas-8080.theiadocker-3-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/graphql

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

	KEY	VALUE	DESCRIPTION
	Key	Value	Description

5. Choose **Body** and **GraphQL** as shown in the image below to start entering your query.

https://lavanyas-8080.theiadocker-3-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/graphql

POST

https://lavanyas-8080.theiadocker-3-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/graphql

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

Auto-fetch

QUERY

GRAPHQL VARIABLES

1

1

6. You will first query for all the cities in the US. What you can get is the city name and the state they are in. Paste the following in the **Query** tab and click **Send**.

```
1. 1
2. 2
3. 3
4. 4
5. 5
6. 6

1. {
2.   cities {
3.     city
4.     state
5.   }
6. }
```

Copied!

POST

https://lavanyas-8080.theiadocker-3-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/graphql

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

Auto-fetch

QUERY

```
1 {
2   cities {
3     city
4     state
5   }
6 }
```

GRAPHQL VARIABLES ⓘ
1

Body

Cookies (1)

Headers (7)

Test Results

Status: 200 OK

Time: 885ms

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "data": {
3     "cities": [
4       {
5         "city": "Abbeville",
6         "state": "Louisiana"
7       },
8       {
9         "city": "Aberdeen",
10        "state": "Maryland"
11      },
12    ]
13  }
14 }
```

As you can see, it lists all the 5980 cities in the US in the output tab.

7. You can now try to retrieve just the names of the cities in a particular state. To do this, you need to pass the state as a parameter. You will now request the state name in the return value as you are querying only for one state.

```
1. 1
2. 2
3. 3
4. 4
5. 5
6.
7. {
8.   cities(state:"Florida") {
9.     city
10.  }
11. }
```

Copied!

GET

http://localhost:8080/graphql

Send

Params

Auth

Headers (9)

Body

Pre-req.

Tests

Settings

GraphQL

Auto-fetch

QUERY

```
1 {
2   cities(state:"Florida") {
3     city
4   }
5 }
```

GRAPHQL VARIABLES

1

Body

200 OK 26 ms 8.7 KB

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "data": {
3     "cities": [
4       {
5         "city": "Alachua"
6       },
7       {
8         "city": "Altamonte Springs"
```

As you can see, only the names of the cities in Florida are returned.

Additional Task:

- 1. Construct a query to return all the cities in the state of "Ohio", along with the state name.
- [Click here for the solution](#)
- Congratulations! You have learned to construct queries using GraphQL.

Tutorial details

Author: Lavanaya T S

Contributors: Pallavi Rai

Change Log

Date	Version	Changed by	Change Description
2022-08-23	1.0	Lavanaya T S	Initial version created
2022-11/25	1.1	Steve Hord	QA pass edits

(C) IBM Corporation 2023. All rights reserved.