Hands-on Lab - Querying with GraphQL

Skills Network

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

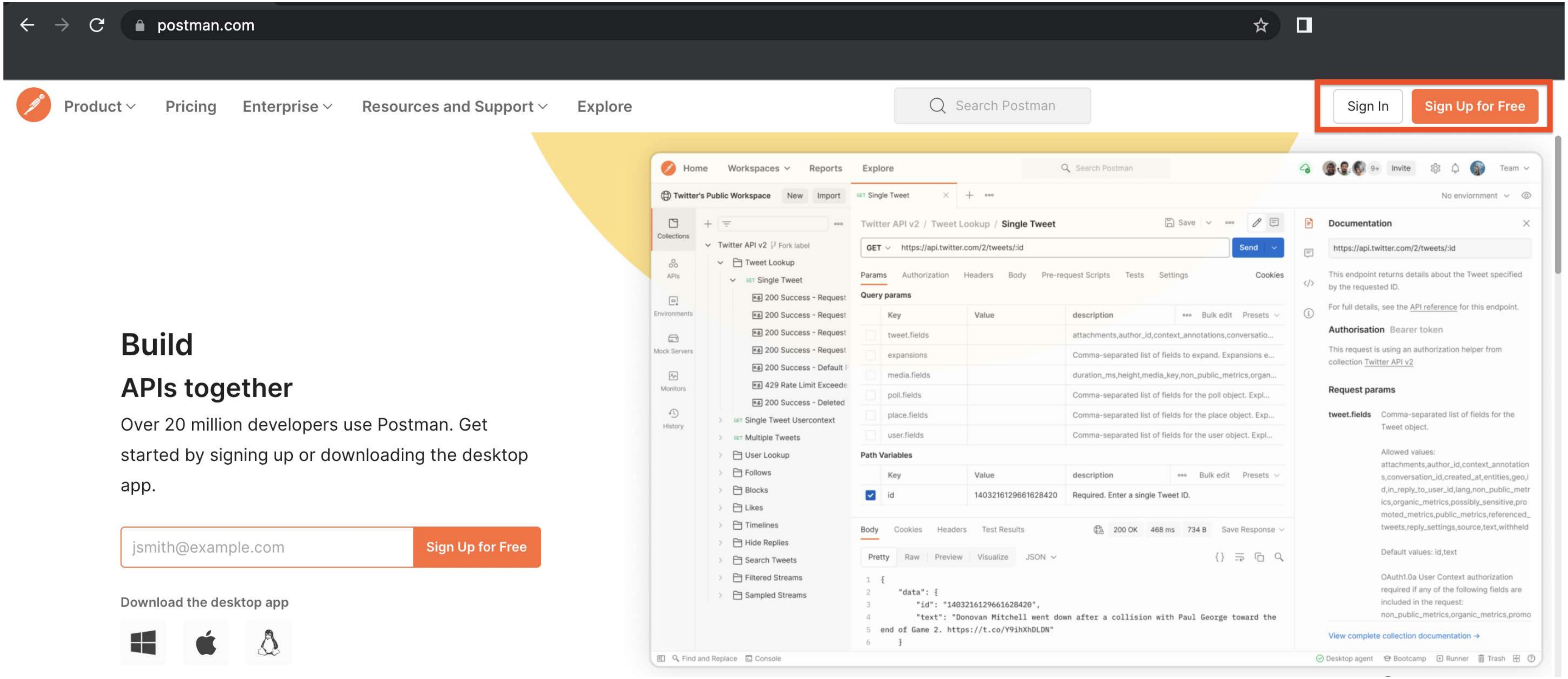
Pre-requisites

- 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



Build

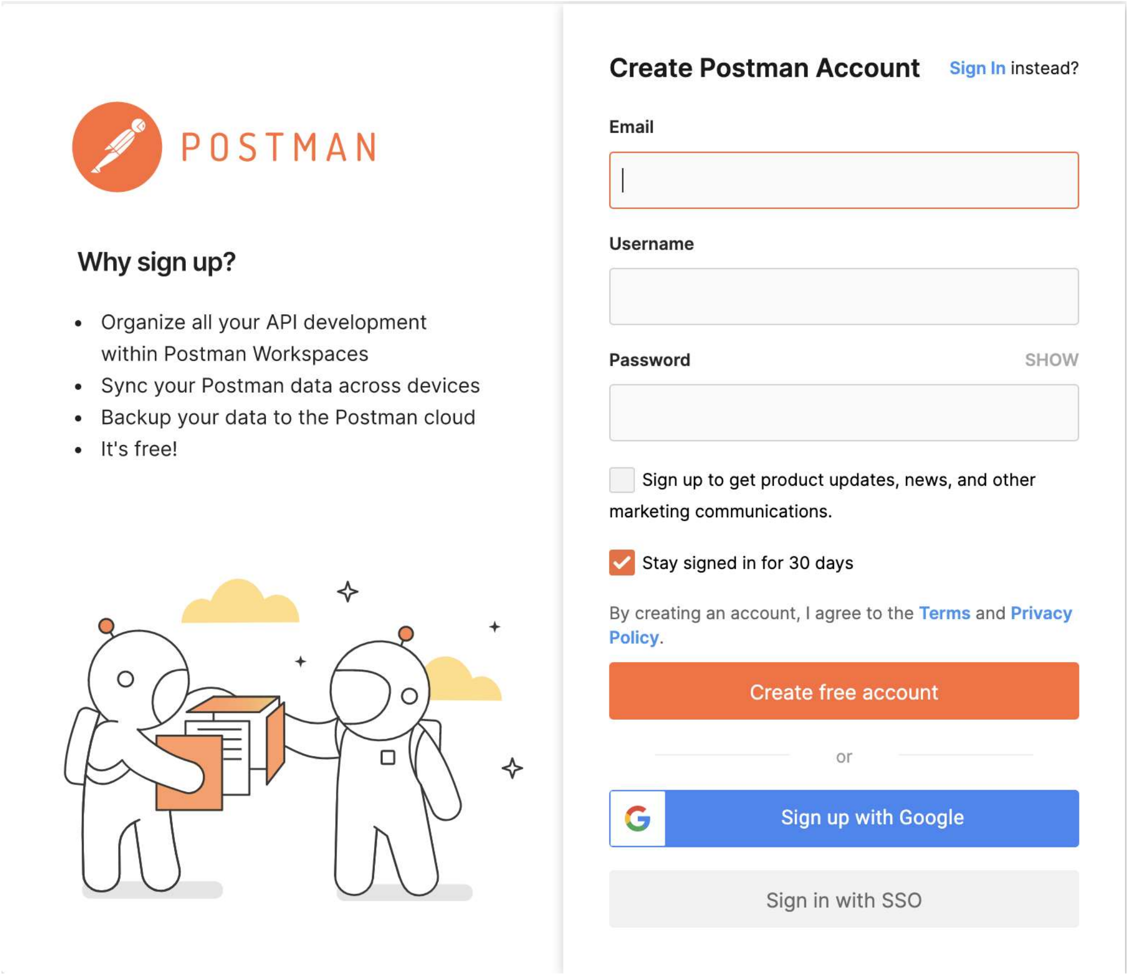
APIs together

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
Home
Workspaces
Reports
Explore
Twitter's Public Workspace
Twitter API v2 / Tweet Lookup / Single Tweet
GET
https://api.twitter.com/2/tweets/:id
Send
Documentation

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.



POSTMAN

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!

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.

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.

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

Copied!

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

```
cd jmgdo-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**.

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

Copied!

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

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

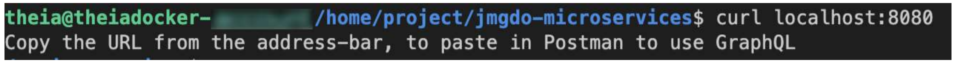
Copied!

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

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

```
curl localhost:8080
```

Copied!

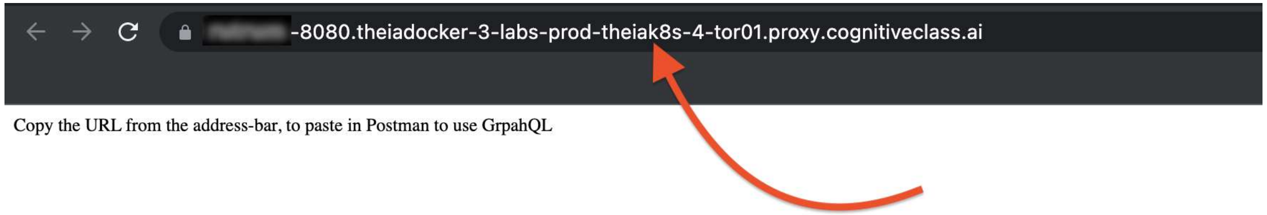


```
theia@theiadocker-[redacted]/home/project/jmgdo-microservices$ curl localhost:8080
Copy the URL from the address-bar, to paste in Postman to use GraphQL
```

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.



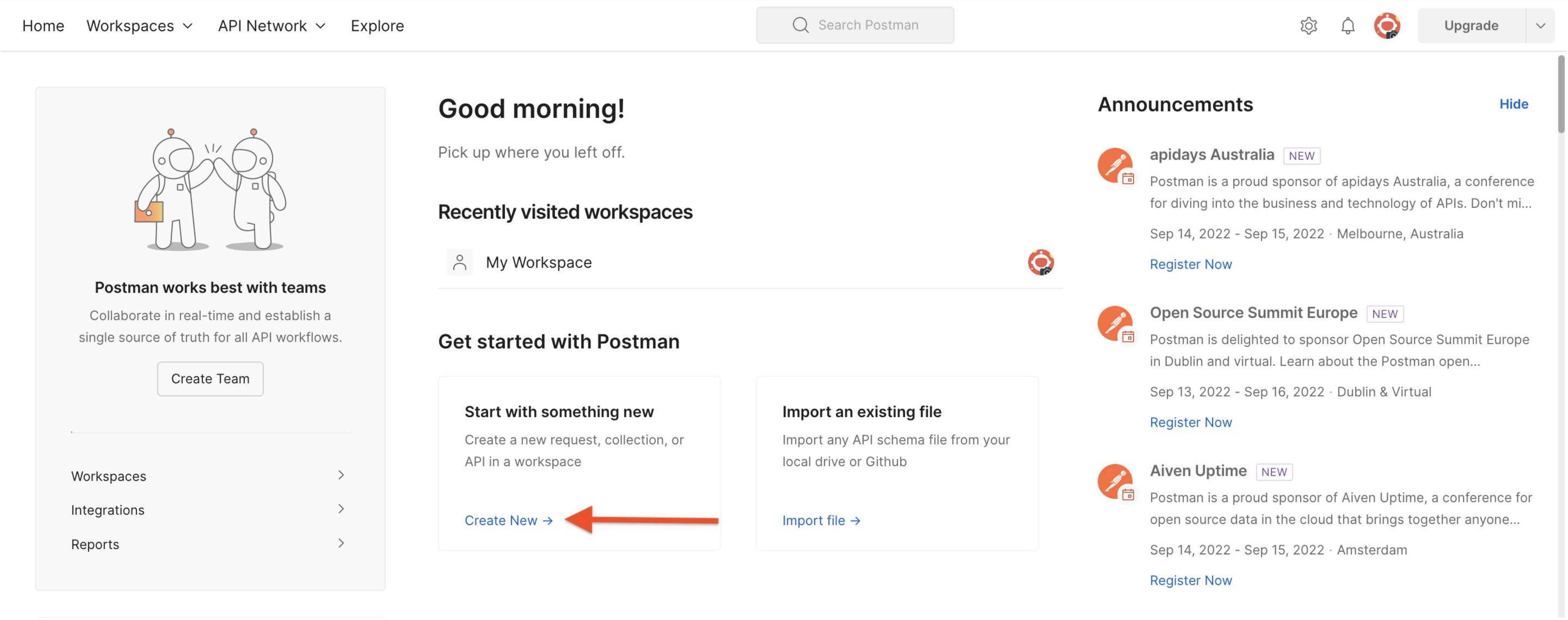
[redacted]-8080.theiadocker-3-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai

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.

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

Import an existing file

Import any API schema file from your local drive or Github

Announcements

apidays Australia NEW

Postman is a proud sponsor of apidays Australia, a conference for diving into the business and technology of APIs. Don't mi...

Sep 14, 2022 - Sep 15, 2022 · Melbourne, Australia

Open Source Summit Europe NEW

Postman is delighted to sponsor Open Source Summit Europe in Dublin and virtual. Learn about the Postman open...

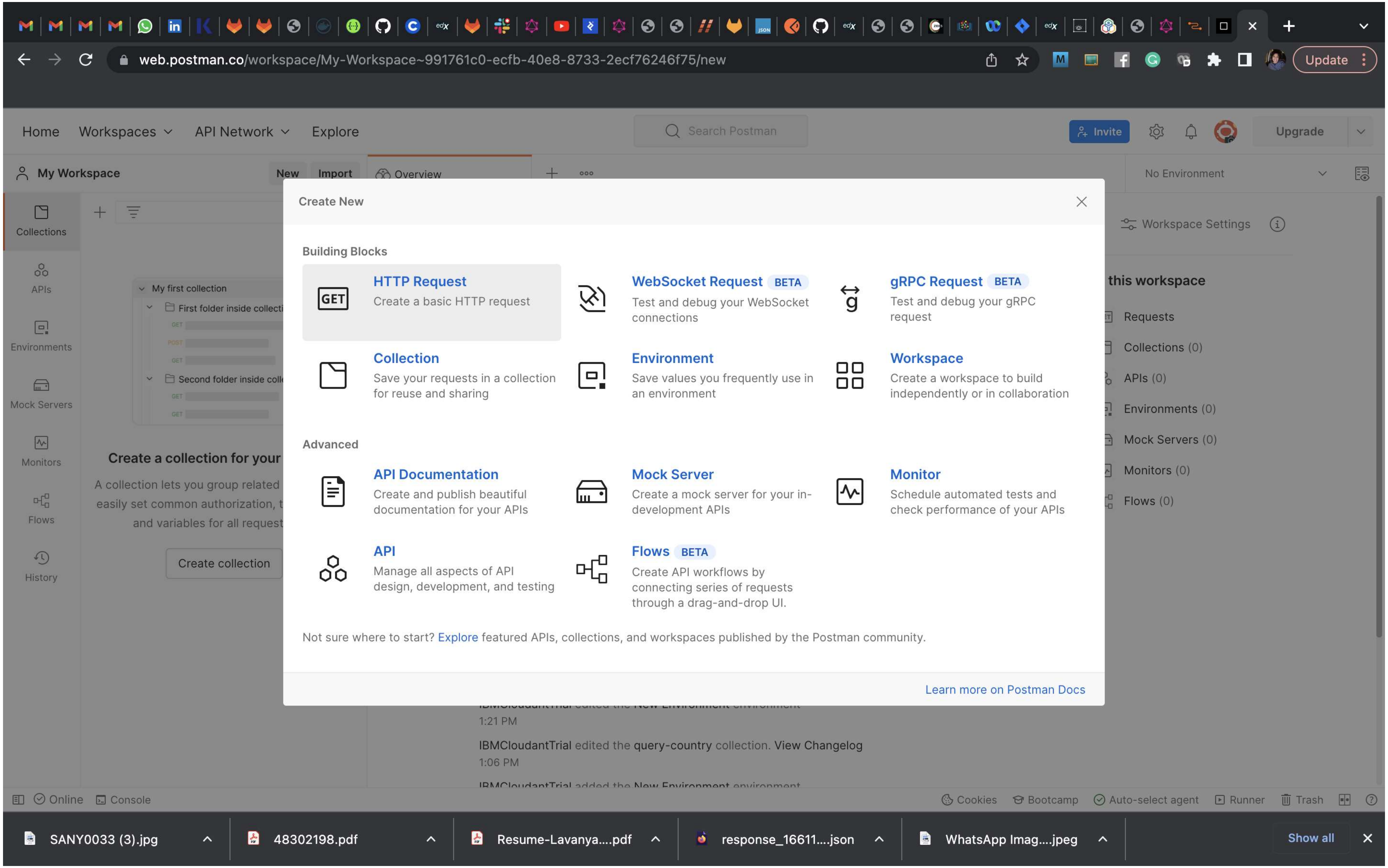
Sep 13, 2022 - Sep 16, 2022 · Dublin & Virtual

Aiven Uptime NEW

Postman is a proud sponsor of Aiven Uptime, a conference for open source data in the cloud that brings together anyone...

Sep 14, 2022 - Sep 15, 2022 · Amsterdam

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



web.postman.co/workspace/My-Workspace~991761c0-ecfb-40e8-8733-2ecf76246f75/new

Create New

Building Blocks

HTTP Request Create a basic HTTP request

WebSocket Request BETA Test and debug your WebSocket connections

gRPC Request BETA Test and debug your gRPC request

Collection Save your requests in a collection for reuse and sharing

Environment Save values you frequently use in an environment

Workspace Create a workspace to build independently or in collaboration

Advanced

API Documentation Create and publish beautiful documentation for your APIs

Mock Server Create a mock server for your in-development APIs

Monitor Schedule automated tests and check performance of your APIs

API Manage all aspects of API design, development, and testing

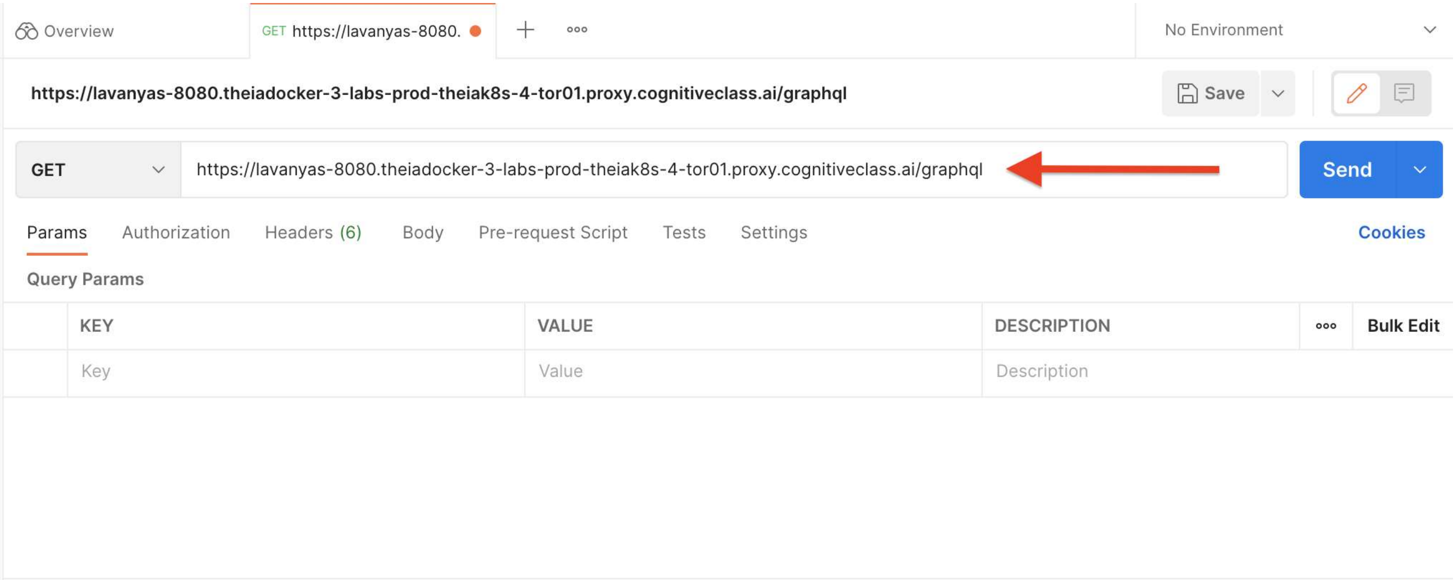
Flows BETA Create API workflows by connecting series of requests through a drag-and-drop UI.

Not sure where to start? Explore featured APIs, collections, and workspaces published by the Postman community.

Learn more on Postman Docs

IBMCloudantTrial edited the query-country collection. View Changelog
1:06 PM

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**.



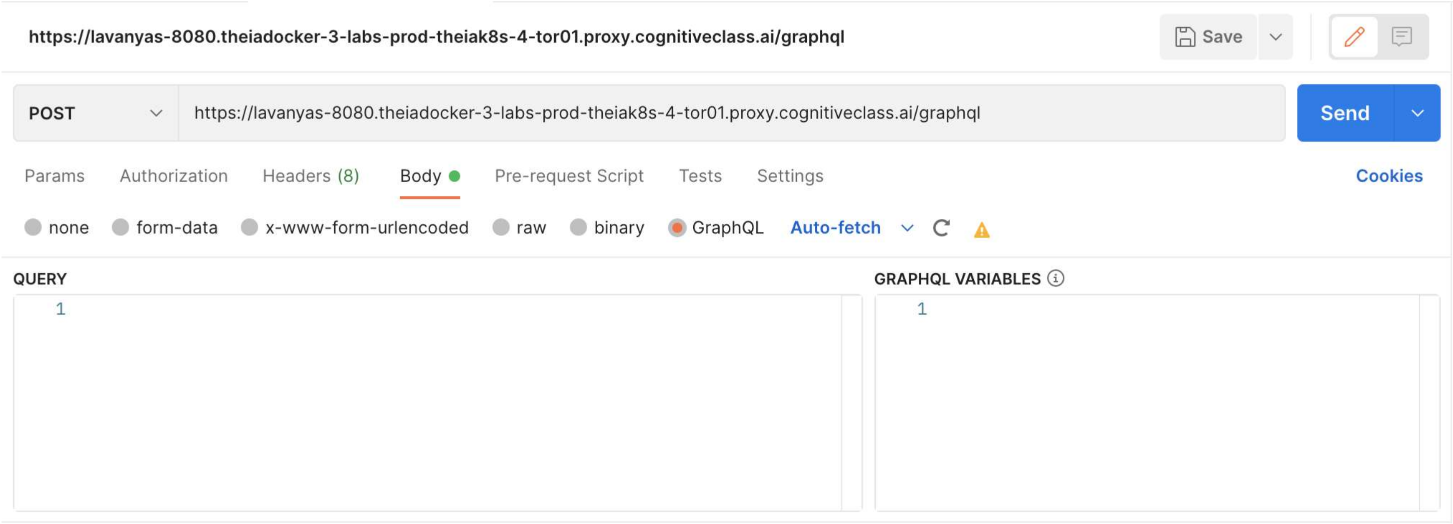
https://lavanyas-8080.theiadocker-3-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/graphql

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

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

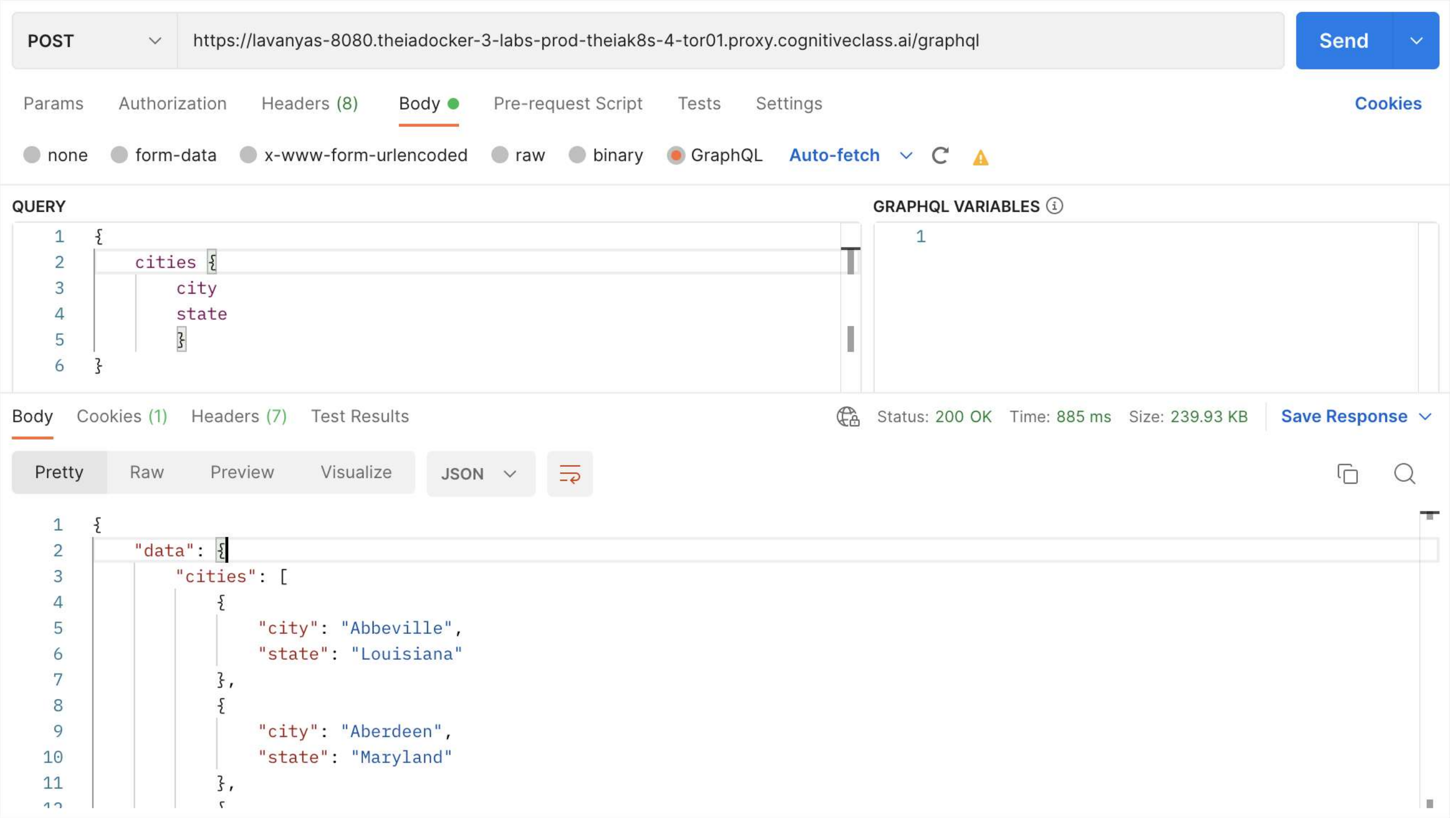
Body: ○ none ○ form-data ○ x-www-form-urlencoded ○ raw ○ binary ● GraphQL

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
2
3
4
5
6
```

```
{
    cities {
        city
        state
        }
}
```

Copied!



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

QUERY

```
{
    cities {
        city
        state
        }
}
```

Status: 200 OK Time: 885 ms Size: 239.93 KB

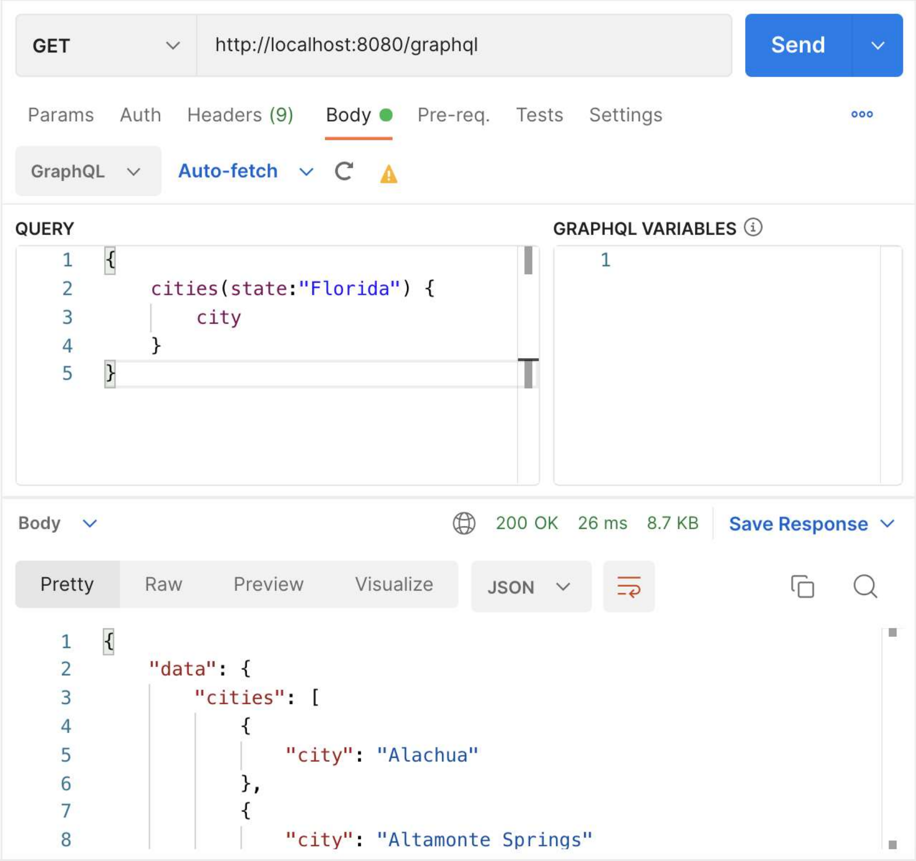
```
{
    "data": {
        "cities": [
            {
                "city": "Abbeville",
                "state": "Louisiana"
            },
            {
                "city": "Aberdeen",
                "state": "Maryland"
            },
```

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.

```
{
    cities(state:"Florida") {
        city
    }
}
```

Copied!



```
http://localhost:8080/graphql
```

```
{
    cities(state:"Florida") {
        city
    }
}
```

```
{
    "data": {
        "cities": [
            {
                "city": "Alachua"
            },
            {
                "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

```
{
    cities(state:"Ohio") {
        city
        state
    }
}
```

Copied!

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