

# Technology Spotlight

---

# GraphQL



## STEP 1

# Sign up for a free tier account in AWS

AWS offers an assortment of free services for all individuals that sign up. It is important to note that not all services are free and that it is almost disturbingly easy to incur charges that can add up quickly.

This guide will instruct you on how to tear down the services that are created here.

Create an account here: <https://aws.amazon.com/free/>

**aws** Contact Us Support English My Account Sign in [Create an AWS Account](#)

Products Solutions Pricing Documentation Learn Partner Network AWS Marketplace Customer Enablement Events Explore More

AWS Free Tier Overview FAQs Terms and Conditions

## AWS Free Tier

Gain free, hands-on experience with the AWS platform, products, and services

[Learn more about AWS Free Tier](#)

[Create a Free Account](#)

**FEATURED**

**Startups get up to \$100,000 in AWS credits**

AWS Activate provides eligible startups with a host of resources, including free AWS credits to spend on AWS services, and AWS Support.

[Sign up for Activate Today](#)

### Types Of Offers

Explore more than 100 products and start building on AWS using the Free Tier. Three different types of free offers are available depending on the product used. Click icon below to explore our offers.

**Free trials**

Short-term free trial offers start from the date you activate a particular service

**12 months free**

Enjoy these offers for 12 months following your initial sign-up date to AWS

**Always free**

These free tier offers do not expire and are available to all AWS customers

### Explore Top Product Categories

Compute

Database

Storage

Containers

Web & Mobile Apps

Serverless

Machine Learning

### Free Tier details

Filter by: [Clear all filters](#)

Tier Type

- Featured
- 12 Months Free
- Always Free
- Trials

COMPUTE

Free Tier 12 MONTHS FREE

Amazon EC2

**750 Hours**

STORAGE

Free Tier 12 MONTHS FREE

Amazon S3

**5 GB**

DATABASE

Free Tier 12 MONTHS FREE

Amazon RDS

**750 Hours**

## STEP 2

# Design a data model

Take a subject that you know a lot about or are interested in and write down the elements of its information.

Topics that AIM attendees chose to model included recipes, wines, skydiving, music, and others.

Once you have a topic you can look at the elements and properties of your topic to narrow down what to model.

- **Music:** songs have album, duration, genre, label, artist
- **Recipe:** title, ingredients, steps
- **Beer Tasting:** brand, type, location, notes

Think about your data model and start a spreadsheet with 3 columns: field, type, single/list.

Here is an example of a modeling spreadsheet with a model for beer tasting notes:

Field Name	Field Datatype
brewery	String
name	String
containerSize	float
style	String
rating	Int

And then give a title to your whole dataset. For instance:

- MusicCollection
- FavoriteRecipes
- BeerTasting

## STEP 3

# Plug your model into AWS AppSync

Once you know the information you want to track, you are ready to use AWS AppSync to build this into a working API that includes Data Storage, a Web API, Security, and resiliency with zero code.

This will set up

- A GraphQL API with a public URL
- The service connection from the API into AWS
- A DynamoDB that stores that data in the format that you specify in the setup

Go to AWS AppSync and log in if you need.

<https://us-west-2.console.aws.amazon.com/appsync/>

Click **Create API**, select "Create with wizard," and click **Start**.

This will bring you to the page to create your model.

- Enter the name that you chose in step 2 for The Model Name.
- There will be 2 prepopulated model fields. Remove all fields and click "Add Field."
- Enter the Model Name and ID as the name of the first field.
- Add Fields and enter the Names and Types from your spreadsheet.
- Leave List checkbox empty – this is an advanced topic for a future tutorial.

Here is a set of model fields that reflects the "BeerTasting" model from the spreadsheet in the above example.

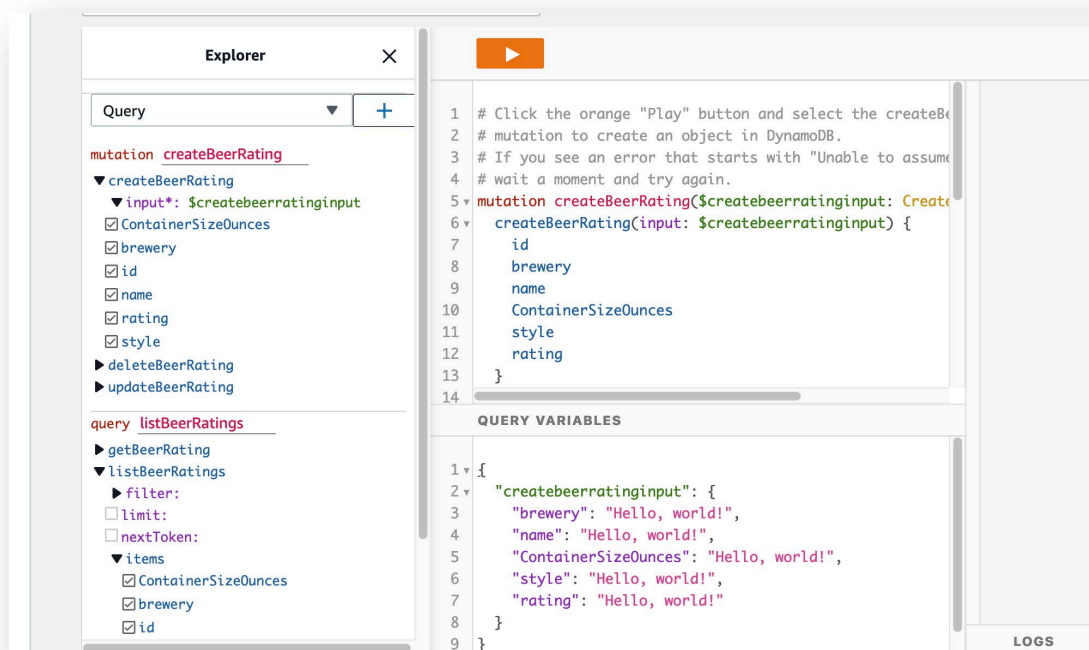
**Configure model fields**  
Models have fields. Fields have a name and type.

Name	Type	List	Required	
id	ID	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Remove field
brewery	String	<input type="checkbox"/>	<input type="checkbox"/>	Remove field
name	String	<input type="checkbox"/>	<input type="checkbox"/>	Remove field
ContainerSizeOunces	String	<input type="checkbox"/>	<input type="checkbox"/>	Remove field
style	String	<input type="checkbox"/>	<input type="checkbox"/>	Remove field
rating	String	<input type="checkbox"/>	<input type="checkbox"/>	Remove field
Add field				

- Click **Create** to go to the next page.
- Enter the name with API or App added to it for the name of your API (BeerTastingAPI).
- Click **Create** to start the creation of your API.

Once the page refreshes to show Queries, congratulations! You have built an API!

Here is a section of the page that appeared when we created the “BeerTasting” API:



Read the prepopulated values on the Queries page carefully and notice the elements that were taken directly from your data model, you should see the field names you created and the necessary methods to query and mutate data.

## STEP 4

# Interact with your API using Apollo GraphQL Sandbox

To connect to your API from outside of AWS you will need:

- The API location URL
- The API Key

Both of which are available in your AppSync API's settings.

From your App in AWS Console, you can go to the left navigation pane and click Settings. This will bring you to the page containing the API URL and the API Key that you will need to connect.

### API Details

API URL  
https://[REDACTED]appsync-api.us-west-2.amazonaws.com/graphql

API ID  
xn7qmrvttrae7dftvmt3tulfvm

Primary auth mode  
API KEY  
[REDACTED]

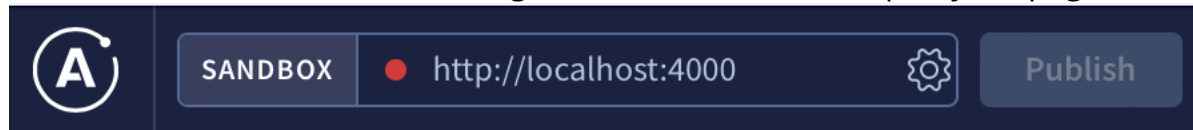
API keys by default expire 7 days after creation.

Copy these values onto a document and save that for use in Apollo.

Go to the Apollo GraphQL Sandbox.

<https://studio.apollographql.com/sandbox/>

You should see the sandbox configuration section at the top of your page:



In the SANDBOX field, replace `http://localhost:400` with your API URL



Click the Gear and enter a Shared Header

#### Shared headers

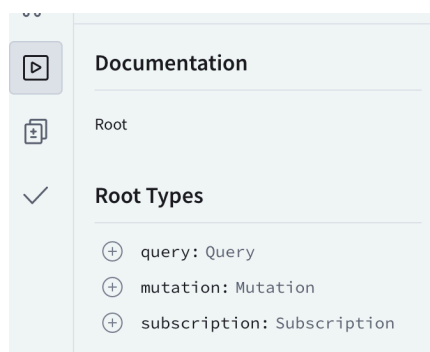
header key	value
------------	-------

The header key will be “x-api-key” and the value will be the API KEY that you saved from Settings.

When you save this, you will see the Documentation pane on the left update to show the fields along with queries and mutations.


### Create Records in your API

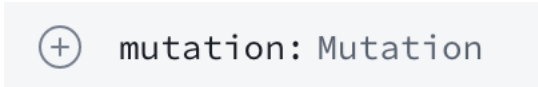
In the left navigation, click on Root to ensure you are at the root of your API. The screen will look like this when it is correct:




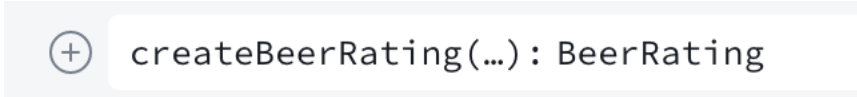
Clear all text from the Operations and Variables panes in the center of your screen. (This Apollo Sandbox tool works perfectly when starting from a blank screen but is not well suited to edit operation or variable data. Start all activity from a clean window)

Screengrabs below will be for the BeerTasting API. Replace anything that you see about beer with your API Name.

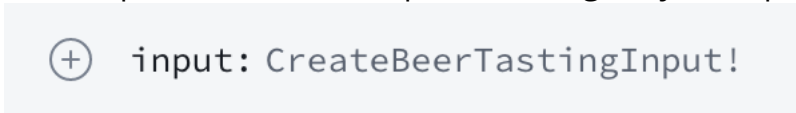
Click the plus  next to mutation to start a mutation.




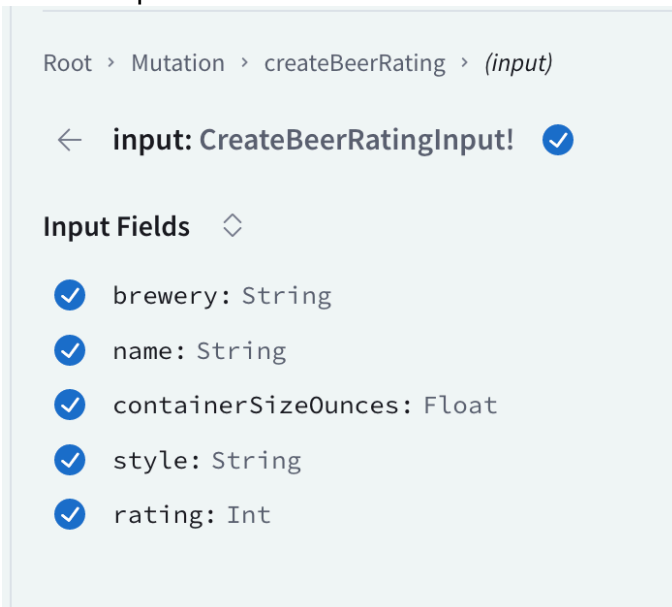
Click the plus  next to create to make it a create action



Click on plus  next to input: to configure your inputs



Click the plus  next to each fieldname to select them.





Click the Arrow to go back and select at least one field.

← **input: CreateBeerTastingInput!** ✓

Here you see that the API will return the ID, brewery, name, and rating

**Arguments**

- ✓ input: CreateBeerRatingInput!

**Fields** ◇ - ...

- ✓ id: ID!
- ✓ brewery: String
- ✓ name: String
- ⊕ containerSizeOunces: Float
- ⊕ style: String
- ✓ rating: Int

Enter the data for a record in the Variables pane by replacing nulls with real values.

Here is an example. Notice that strings have quotation marks around them, floats have decimal points and integers (ints) are whole numbers without decimals.

**Variables** Headers

```

1  {
2    "input": {}
3    "brewery": "cloudburst",
4    "name": "unicorn rainbow",
5    "ContainerSizeOunces": 12.0,
6    "style": "hazy pale",
7    "rating": 9
8  }
9  }
```

In this configuration, the Operation pane will show the fact that it is a mutation, a reference to the input, and a list of the fields to be returned to your screen when a successful operation is complete.

If there are no red underlines or red X icons on your screen click

▶ Mutation


You should see a response in the right-hand pane with STATUS 200 and a data payload indicating that you have created a record.

Create a few more records with variations in your Variables. You are creating a new record in your API each time you click Mutation and get back that 200 response.

## Select Records in your API

Now you are ready to read back what you have done.

- Clear out the Operation and Variables panes to start clean.
- Click the arrow to get back to Root Types.

You will then use the plus  to select:

Query

ListyourAPIName

Items [yourAPIName]

And then select the fields you want to return.

Once you have a query with at least one field and no error messages, the button will highlight, indicating that you are ready to return data to your screen.

▶ Query

This will be a list of all the records you have entered in the earlier section.

## You did it!

There is a lot more to explore, but you have done all the basic operations of defining, creating and interacting with a GraphQL API in AWS AppSync.

- ✓ You defined a data model.
- ✓ You entered that data model into AppSync and created a working API from it.
- ✓ You used the Apollo Sandbox to add records to your API.
- ✓ You used the Apollo Sandbox to read records from your API.

## STEP 5

# Cleaning Up

The tutorial here does create persistent resources that can have ongoing effects in your AWS account.

These resources include:

- An AWS AppSync API
- A DynamoDB table
- An IAM Policy
- An IAM Role

The AppSync API and the Dynamo DB can incur ongoing charges and the IAM Policy and role can leave unnecessary permissions in your account, which create an unnecessary security exposure.

Go to each service in the AWS console and locate and delete the resources that were created.

The resource names should contain the name of your API and should then be easy to identify.

Delete your API from AppSync

<https://console.aws.amazon.com/appsync/>

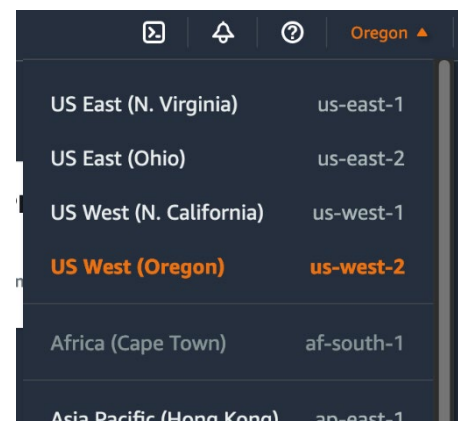
Delete your table from DynamoDB

<https://console.aws.amazon.com/dynamodbv2>

Delete your Policy and Role from Identity and Access Management

<https://console.aws.amazon.com/iamv2>

The AppSync and DynamoDB are region-specific and may not show if your region setting changed in the console. If you do not see your resources, change the region in the upper right corner of the console to see your resources.



## Take it to the next level

We have only scratched the surface of what GraphQL and AppSync can offer in a robust API-provided data interface.

Other subjects very close to this field include: GraphQL data subscriptions, GraphQL query management, and DynamoDB features including cost management, performance allocation, data integration and other rich aspects of data and APIs in the AWS cloud.

GraphQL is a well-established and expressive standard for data interfaces that can be useful for both small and large projects. Amazon Web Services' AppSync is a mature, robust and easy-to-use implementation of GraphQL that allows a full backend service to be implemented with practically no coding. I hope that these techniques give you insight into the use of GraphQL and cloud services to serve the needs of your projects both today and tomorrow.

## About the Author



### Patrick Taylor

National Technology Evangelist

Patrick Taylor is a National Technology Evangelist at AIM and holds over twenty years of experience in technology leadership. He helps our clients solve their hardest technology problems by providing reference architectures, high quality tools, and industry-leading techniques to software development. On a day to day, Patrick builds teams, designs cloud architectures, and drives velocity and quality. His background spans all levels across software organizations, from individual contributor, to software director, to architect.