

## **Part 02 - Amazon API Gateway**

# API Driven Development

An API is an interface.

An interface essentially defines how you can interact with a resource.

An interface has defined actions, defined inputs, and defined outputs.

OUTPUT

INPUT



The beauty of this interface is that it **abstracts** what is going on  
in the kitchen from the consumer.

Interfaces allow you flexibility when it comes to programming.

You can define how you want your clients or consumers to interact with your service.

You define the **actions**, **inputs**, and **outputs**.

From there, you can implement it however you want behind the scenes,  
and as long as you are fulfilling your end of the bargain and  
the outputs are what they should be given the input,  
no one using the interface needs to know or care about the implementation.

One way we use APIs is to **expose services** to one another.

If I have a program that is composed of five different microservices,  
they would all communicate via a technology-agnostic communication protocol.

Like using **RESTful HTTP-based calls**.

As long as the service can speak HTTP, it doesn't matter what each service is implemented in.

Maybe one service is best written in Python,  
but another service is best written in C++.

Doesn't matter because you are enforcing the use of interfaces through your APIs.

Note: When the API doesn't change, everything works nicely.

Once your APIs are defined and shared with clients,  
it's important that you do what you can to maintain **backwards compatibility** with changes.

This is why it is so important to make sure you are spending time designing your APIs up front and exploring what the actions, inputs, and outputs should be.

Following the practice of **designing the API first**  
is called **API-driven development**,  
where the first artifact created out of building a new service is the API.

Once the API is created, the front-end, or clients, using the API and the actual implementation of the API can be built in parallel.

At the end, when both pieces are done, they should be able to communicate seamlessly as long as the back-end implementation adhere to the API that was first designed and it didn't get changed.

To wrap it up here,  
building APIs is like introducing a contract on  
how you allow others to interact with your code.

# What is API Gateway?

Let's talk about the tools that help to manage and take advantage of APIs.



Amazon API Gateway

**API Gateway** is an AWS service  
for creating, publishing, maintaining, monitoring, and securing  
REST, HTTP, and WebSocket APIs at any scale.

App developers can create APIs that access  
AWS or other web services as well as data stored in the AWS Cloud.

While an API developer can gain the ability to create and deploy an API,  
enabling the necessary functionality in API Gateway,  
  
an app developer can build a functioning application  
to call AWS services by invoking the created APIs.

While we'll go more into detail on the use of API Gateway later,  
let's talk more about what the service provides.

This service can be used to create **HTTP**, **REST** and **WebSocket** APIs.

**WebSocket APIs** operate using lower-level protocols based on sockets and ports and requires the use of IP addresses and port information.

It is bidirectional, stateful, vertically scalable, and is ideally used for real-time scenarios such as a chat application as part of a game.

**REST** (Representational State Transfer) is an architectural style for designing networked applications that utilize the HTTP protocol.

Unlike WebSocket APIs which rely on lower-level protocols, REST operates over standard web protocols and is **stateless**.

It focuses on **resources**, which are uniquely identified by URLs, and uses standard HTTP methods (GET, POST, PUT, DELETE) to perform actions on these resources.

This makes it a robust choice for applications that primarily involve creating, retrieving, updating, and deleting data.

RESTful services are widely used for tasks like fetching web pages, accessing web APIs, and manipulating data in web and mobile applications.

**HTTP** operates on a request-response model.

A client sends an HTTP request to a server,  
which processes the request and sends back an HTTP response.

We focus on REST APIs,  
which are fast, stateless, standard, horizontally scalable and dependable.

## Other features provided by API Gateway:

- Canary deployments for safety and rolling out changes
- AWS CloudTrail integration for logging
- Amazon CloudWatch integration for monitoring
- Support for custom domain names to throttling of requests
- Direct integration with other AWS services  
including AWS WAF, AWS X-Ray and AWS Lambda.

A **method** represents a client-facing interface by which the client calls the API to access backend resources and refers to the particular HTTP verb used with a request.

Note: The most common HTTP verbs are:

- GET: Used to retrieve data from a server.
- POST: Used to send data to be processed by a server.
- PUT: Used to update a resource on the server.
- DELETE: Used to remove a resource from the server.
- And others like PATCH, OPTIONS, etc.

API Gateway provides multiple **endpoint types** that you can utilize.

An API endpoint type refers to the host name of the API.

It can be **edge** optimized, **regional** or **private**  
depending on where the majority of your API traffic originates.

**Edge optimized** API endpoints are best for geographically distributed clients.

Requests are routed to the nearest **Amazon CloudFront point of presence**.

This is the default type for API Gateway REST APIs.

A **regional API** endpoint is intended for clients in the same region.

When a client running an Amazon EC2 instance calls an API in the same region or when an API is intended to serve a small number of clients with high demands, a regional API endpoint reduces connection overhead.

**Private API** endpoints are a great way to provide a client secure access to resources inside of an Amazon virtual private cloud.

Private APIs are isolated from the public internet and they are only accessed using VPC endpoints for API Gateway that have been granted access.

# Dragon API: API Gateway Terminology

In this section, we're going to build the first method of our dragon API.

To start, we'll be going through the console  
and exploring some of the different pieces you can configure, as well as what they mean.

AWS Management Console X +

console.aws.amazon.com/console/home?region=us-east-1

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD GitHub - aws-sam...

aws Services Resource Groups

buildingmodernapps @ 3022-1... N. Virginia Support

# AWS Management Console

**AWS services**

**Find Services**  
You can enter names, keywords or acronyms.

▼ Recently visited services

 API Gateway  S3  CloudWatch

 Cloud9  IAM

▶ All services

**Build a solution**  
Get started with simple wizards and automated workflows.

[Launch a virtual machine](#) [Build a web app](#) [Build using virtual servers](#)

Stay connected to your AWS resources on-the-go

 Download the AWS Console Mobile App to your iOS or Android mobile device. [Learn more](#)

**Explore AWS**

**Amazon Redshift RA3 Nodes**  
Scale your compute and storage independently and lower your costs.  
[Learn more](#)

**S3 Intelligent-Tiering**  
Optimize costs automatically with Amazon S3.  
[Get started](#)

API Gateway

console.aws.amazon.com/apigateway/main/apis?region=us-east-1

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

Services Resource Groups

API Gateway X Networking & Content Delivery

# Amazon API Gateway

create, maintain, and secure APIs at any scale

Amazon API Gateway helps developers to create and manage APIs to back-end systems running on Amazon EC2, AWS Lambda, or any publicly addressable web service. With Amazon API Gateway, you can generate custom client SDKs for your APIs, to connect your back-end systems to mobile, web, and server applications or services.

### Choose an API type

HTTP API

Build low-latency and cost-effective REST APIs with built-in features such as OIDC and OAuth2, and native CORS support.

Works with the following:  
Lambda, HTTP backends

Import Build

Feedback English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms

You can build an HTTP API.

This is designed to offer REST-base HTTP API's at low latency and low cost.

HTTP API's are used to proxy backend resources and are supposed to be simple and fast.

This is great for when you want API Gateway to  
simply take in a request, authorize it and pass it on to the  
backend resource like a Lambda function or an HTTP endpoint.

API Gateway

console.aws.amazon.com/apigateway/main/apis?region=us-east-1

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

Services Resource Groups

buildingmodernapps @ 3022-1... N. Virginia Support

API Gateway APIs Custom domain names VPC links

### Choose an API type

**HTTP API**

Build low-latency and cost-effective REST APIs with built-in features such as OIDC and OAuth2, and native CORS support.

Works with the following:  
Lambda, HTTP backends

[Import](#) [Build](#)

**WebSocket API**

Build a WebSocket API using persistent connections for real-time use cases such as chat applications or dashboards.

Works with the following:  
Lambda, HTTP, AWS Services

[Build](#)

**REST API**

WebSockets unlike HTTP, is a stateful communications protocol.

WebSocket API's allow for support for applications  
that need real time data or real time communication.

API Gateway

console.aws.amazon.com/apigateway/main/apis?region=us-east-1

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

aws Services Resource Groups

API Gateway

Import Build

WebSocket API

Build a WebSocket API using persistent connections for real-time use cases such as chat applications or dashboards.

Works with the following:  
Lambda, HTTP, AWS Services

Build

REST API

Develop a REST API where you gain complete control over the request and response along with API management capabilities.

Works with the following:  
Lambda, HTTP, AWS Services

Import Build

REST API Private

REST API's with API Gateway uses HTTPS and is stateless.

This is very similar to HTTP API's, but it offers some different functionality.

REST API's allow you to have full control over  
the response and requests between your client and API Gateway.

You can apply what are called **models** and **mappings**  
to validate and transform requests and responses.

We will talk about this more in upcoming sections.

API Gateway

console.aws.amazon.com/apigateway/main/apis?region=us-east-1

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

aws Services Resource Groups

API Gateway APIs Custom domain names VPC links

Works with the following:  
Lambda, HTTP, AWS Services

Build

REST API

Develop a REST API where you gain complete control over the request and response along with API management capabilities.

Works with the following:  
Lambda, HTTP, AWS Services

Import Build

REST API Private

Create a REST API that is only accessible from within a VPC.

Works with the following:  
Lambda, HTTP, AWS Services

Import Build

Private API is the same thing as a normal REST API,  
but it allows to set up an API that can only be accessed from a VPC, creating a private API.

This is useful for internal APIs  
that you do not want to expose to outside clients.

API Gateway

console.aws.amazon.com/apigateway/main/apis?region=us-east-1

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

aWS Services Resource Groups

cases such as chat applications or dashboards.

API Gateway APIs Custom domain names VPC links

Build

REST API

Develop a REST API where you gain complete control over the request and response along with API management capabilities.

Works with the following:  
Lambda, HTTP, AWS Services

Import Build

REST API Private

Create a REST API that is only accessible from within a VPC.

Works with the following:  
Lambda, HTTP, AWS Services

Import Build

We're going to select the public REST API,  
because we have some use cases that require  
we validate and transform incoming requests before they reach the backend.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/create

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

aws Services Resource Groups

Amazon API Gateway APIs > Create

## Create your first API

Welcome to Amazon API Gateway. To create your first API, we have pre-populated the import form with a Pet Store API defined using Swagger 2.0. To get started, close this modal and select Import in the Create API form.

In Amazon API Gateway, a REST API refers to a collection of resources.

REST  WebSocket

New API  Import from Swagger or Open API 3  Example API

**OK**

## Create new API

Example API

Learn about the service by importing an example API and turning on hints throughout the console.

```
1  {
2    "swagger": "2.0",
3    "info": {
4      "description": "Your first API with Amazon API Gateway. This is a sample API that integrates via HTTP with our demo Pet Store endpoints",
5      "title": "PetStore"
6    },
7    "schemes": [
8      "https"
9    ],
10   "paths": {
11     "/": {
12       "get": {
13         "tags": [
14           "pets"
15         ],
16         "description": "PetStore HTML web page containing API usage information",
17         "summary": "List all pets"
18       }
19     }
20   }
21 }
```

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/create

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

aws Services Resource Groups

Amazon API Gateway APIs > Create Show all hints

## Choose the protocol

Select whether you would like to create a REST API or a WebSocket API.

REST  WebSocket

## Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

New API  Import from Swagger or Open API 3  Example API

## Settings

Choose a friendly name and description for your API.

**API name\*** My API

**Description**

**Endpoint Type** Regional

\* Required

API Gateway X + console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/create

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

AWS Services Resource Groups star

Amazon API Gateway APIs > Create Show all hints ?

## Choose the protocol

Select whether you would like to create a REST API or a WebSocket API.

REST  WebSocket

## Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

New API  Import from Swagger or Open API 3  Example API

## Settings

Choose a friendly name and description for your API.

API name\*

Description

Endpoint Type  ⓘ

\* Required

Create API 

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/dhqjpep0vi

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

AWS Services Resource Groups

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > / (dhqjpep0vi)

Show all hints ?

APIs Resources Actions / Methods

No methods defined for the resource.

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

On the left hand side is the Navigation, for everything you can configure for an API.

We're going to talk about most of these things,  
but for now though, we're just going to keep it simple.

I want to create an API that allows me  
to submit a request and responds with a 200 OK  
but doesn't actually hit any real backend.

To do that, I need to first create a resource.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/dhqjpep0vi/create

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam... M

Services Resource Groups

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > / (dhqjpep0vi)

Show all hints ?

APIs

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Actions / Methods

No methods defined for the resource.

RESOURCE ACTIONS

- Create Method
- Create Resource**
- Enable CORS
- Edit Resource Documentation

API ACTIONS

- Deploy API
- Import API
- Edit API Documentation
- Delete API**

The screenshot shows the AWS API Gateway console interface. The left sidebar lists various API management features like APIs, Stages, and Models. The main area shows an API named 'dragons' with a single resource path '/'. A context menu is open over this resource, with the 'Create Resource' option being highlighted. The menu also includes options for creating methods, enabling CORS, and editing documentation. Below the menu, a message states 'No methods defined for the resource.' The top navigation bar includes links for AWS services like Onboarding, Training, and AWS Resources, along with user-specific information like GitHub links and profile icons.

A **resource** is an abstract concept  
that allows you to expose a “*thing*” to be consumed by a client.

For our example, we will be building out the dragons API.  
So the resource is the dragon data.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/dhqjpep0vi/create

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam... M M buildingmodernapps @ 3022-1... N. Virginia Support

Services Resource Groups

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > / (dhqjpep0vi) > Create Show all hints ?

New Child Resource

Use this page to create a new child resource for your resource.

Configure as proxy resource

Resource Name\* dragons

Resource Path\* / dragons

You can add path parameters using brackets. For example, the resource path `{username}` represents a path parameter called 'username'. Configuring `/{proxy+}` as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to `/foo`. To handle requests to `/`, add a new ANY method on the `/` resource.

Enable API Gateway CORS

\* Required

Cancel Create Resource

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Enable CORS on the resource.

This will enable cross origin resource sharing for this API,  
and this will be used for future use.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33

Services Resource Groups

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33) Show all hints

APIs Custom Domain Names VPC Links

**API: dragons**

**Resources**

- Stages
- Authorizers
- Gateway Responses
- Models
- Resource Policy
- Documentation
- Settings
- Usage Plans
- API Keys
- Client Certificates

**/dragons Methods**

**OPTIONS**

**Mock Endpoint**

Authorization: None  
API Key: Not required

This screenshot shows the AWS API Gateway console interface. The left sidebar is collapsed, and the main area displays the configuration for the '/dragons' resource under the 'dragons' API. The 'Actions' dropdown is open, and the 'OPTIONS' method is selected. The configuration pane shows that 'Mock Endpoint' is active, with 'Authorization' set to 'None' and 'API Key' set to 'Not required'. The URL for this method is listed as '/dragons'.

Resources in API's have **methods** that allow you to interact or submit actions to a resource.

In our dragon example, we will be using HTTP methods like GET and POST.

But you can use whatever HTTP methods you like in your own APIs.

Let's create a method.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33

Services ▾ Resource Groups ▾

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33) Show all hints

Actions ▾ /dragons Methods

RESOURCE ACTIONS

- Create Method
- Create Resource
- Enable CORS
- Edit Resource Documentation
- Delete Resource

None  
Not required

API ACTIONS

- Deploy API
- Import API
- Edit API Documentation
- Delete API

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Client Certificates

The screenshot shows the AWS API Gateway console interface. The left sidebar lists various API management components like APIs, Stages, and Settings. The main navigation path is 'APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33)'. A context menu is open over the path '/dragons (eqdk33)', specifically under the 'Actions' dropdown. The menu is divided into 'RESOURCE ACTIONS' and 'API ACTIONS'. Under 'RESOURCE ACTIONS', there are five items: 'Create Method', 'Create Resource', 'Enable CORS', 'Edit Resource Documentation', and 'Delete Resource'. The 'Delete Resource' item is currently selected and highlighted in red. Under 'API ACTIONS', there are four items: 'Deploy API', 'Import API', 'Edit API Documentation', and 'Delete API'. Below the menu, a note states 'None' and 'Not required'.

The screenshot shows the AWS API Gateway console interface. The left sidebar lists various API management components: APIs, Custom Domain Names, VPC Links, API: dragons (selected), Resources, Stages, Authorizers, Gateway Responses, Models, Resource Policy, Documentation, Settings, Usage Plans, API Keys, and Client Certificates. The main content area shows the 'APIs' section with 'dragons (it9pqb5zm7)' selected. Under 'Resources', the path '/dragons' is shown, and under it, the specific resource '/dragons (eqdk33)'. The 'Actions' dropdown menu is open over the 'OPTIONS' method, with 'GET' highlighted. The right panel displays the 'OPTIONS' method configuration, which includes 'Mock Endpoint' settings and specifies 'None' for Authorization and 'Not required' for API Key.

Select the GET method here

The screenshot shows the AWS API Gateway console interface. The top navigation bar includes links for Apps, Onboarding, Training, AWS Sandbox, Travel, AWS Resources, DynamoDB, Containers, Workshops, Imported, CD, workshops, GitHub - aws-sam..., and buildingmodernapps @ 3022-1... N. Virginia. The main content area displays the 'APIs' section, specifically the 'dragons (it9pqb5zm7)' API under 'Resources'. A sub-menu for '/dragons' is open, showing the 'Actions' dropdown with 'OPTIONS' selected. The 'OPTIONS' method configuration screen is visible, featuring fields for 'Mock Endpoint', 'Authorization' (set to 'None'), and 'API Key' (set to 'Not required'). The left sidebar lists various API management components: APIs, Custom Domain Names, VPC Links, API: dragons (selected), Resources (Stages, Authorizers, Gateway Responses, Models, Resource Policy, Documentation, Settings), Usage Plans, API Keys, and Client Certificates.

Click the check mark. This brings up a screen for us to configure the integration type.

The screenshot shows the AWS API Gateway console. The URL in the browser is `console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET`. The top navigation bar includes links for Apps, Onboarding, Training, AWS Sandbox, Travel, AWS Resources, DynamoDB, Containers, Workshops, Imported, CD, workshops, GitHub - aws-sam..., and buildingmodernapps @ 3022-1... N. Virginia. The main navigation bar shows the path: Services > Resource Groups > Amazon API Gateway > APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33) > GET. The left sidebar lists various API management features: APIs, Custom Domain Names, VPC Links, API: dragons (selected), Resources, Stages, Authorizers, Gateway Responses, Models, Resource Policy, Documentation, Settings, Usage Plans, API Keys, and Client Certificates.

The main content area is titled "/dragons - GET - Setup". It displays the configuration for the GET method on the /dragons resource. The "Integration type" is set to "Lambda Function" (radio button selected). Other options include "HTTP", "Mock", "AWS Service", and "VPC Link". A checkbox for "Use Lambda Proxy integration" is present, which is unchecked. Below it, the "Lambda Region" is set to "us-east-1" and the "Lambda Function" field is empty. A note states: "You do not have any Lambda Functions in us-east-1. Create a Lambda Function in your current account, or provide an Lambda Function ARN with Cross-Account Access." At the bottom, there is a checkbox for "Use Default Timeout" which is checked.

This is where you select what you want your API to sit in front of, and how you want your API to integrate with that backend.

You can select

- **Lambda function**, where API Gateway is purely a **proxy** for that Lambda function.
- **HTTP**, where you can paste an endpoint that already exists and you want API Gateway to front that.
- **Mock**, where you don't front anything real and instead, you just stub out the API.
- **AWS service**, where you can select another AWS service for API Gateway to front (which we will go into detail on this later).
- **VPC link**, which allows you to expose resources inside of a VPC.

We're following an API driven development process  
and want to design and build the API first.

So we will select Mock as the integration type.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

Services Resource Groups

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33) > GET

Show all hints ?

APIs Resources Actions / /dragons - GET - Setup

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Choose the integration point for your new method.

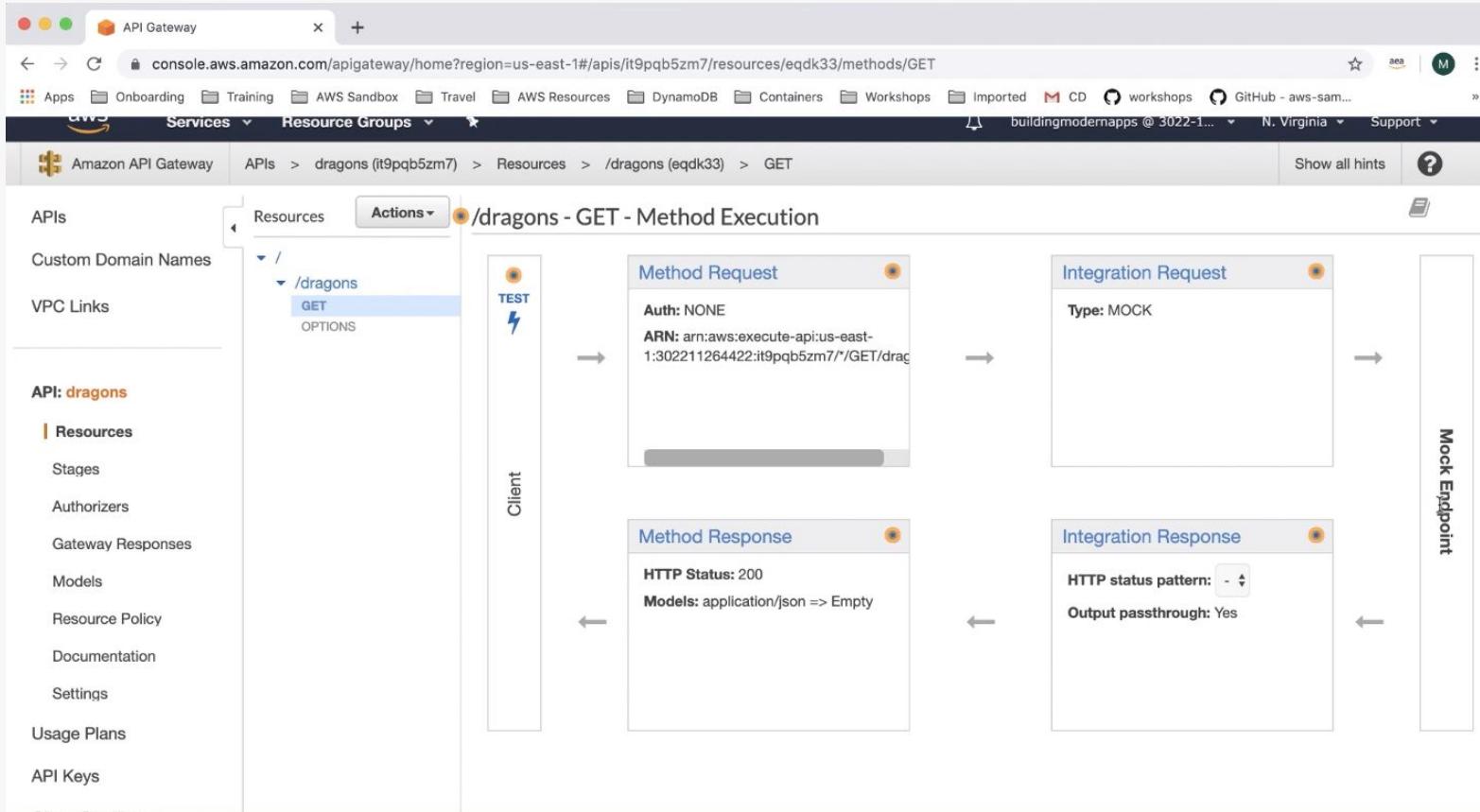
Integration type

- Lambda Function
- HTTP
- Mock
- AWS Service
- VPC Link

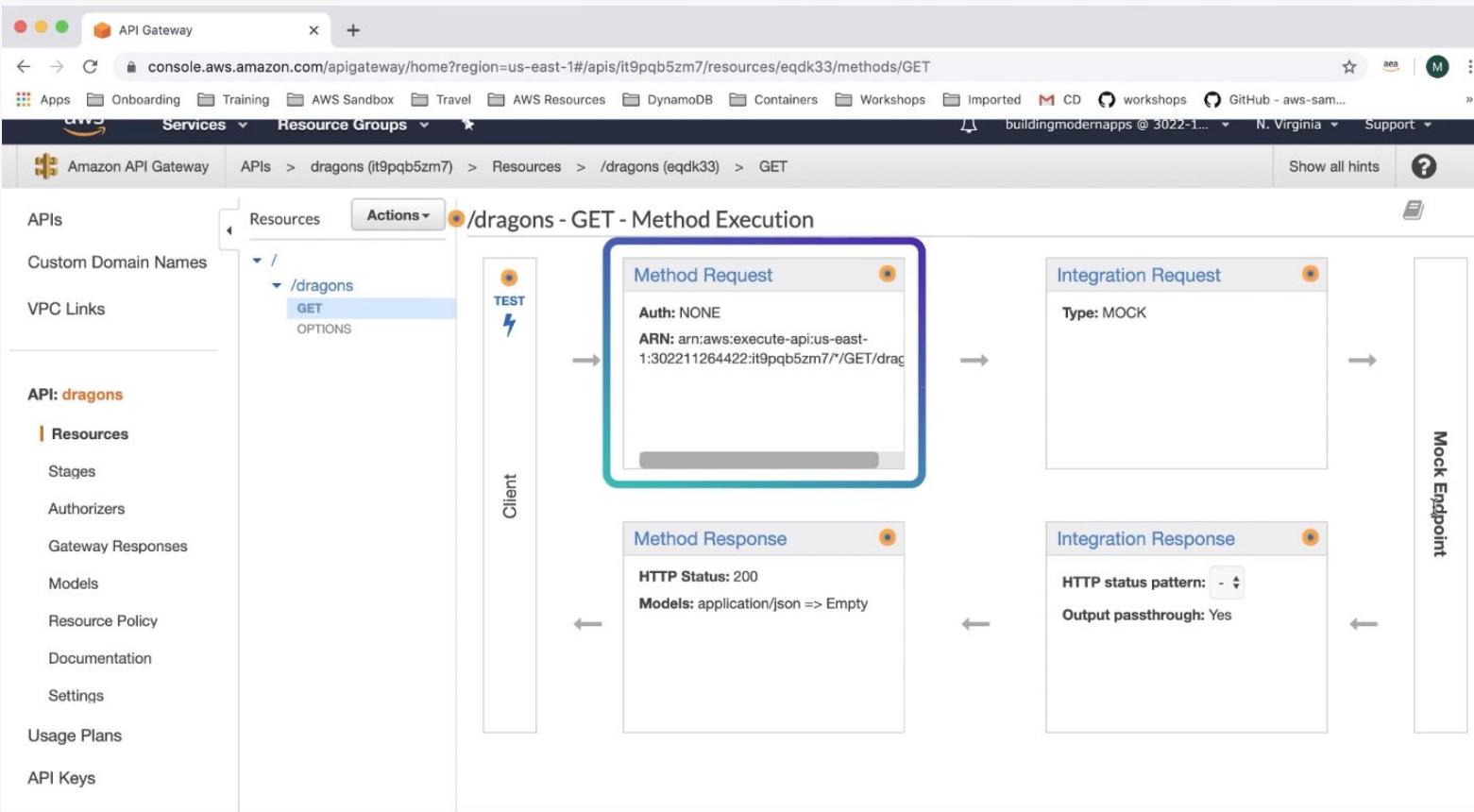
Save

The Mock integration type allows you to set up the API and interact with it without ever needing to actually hit another service for the backend.

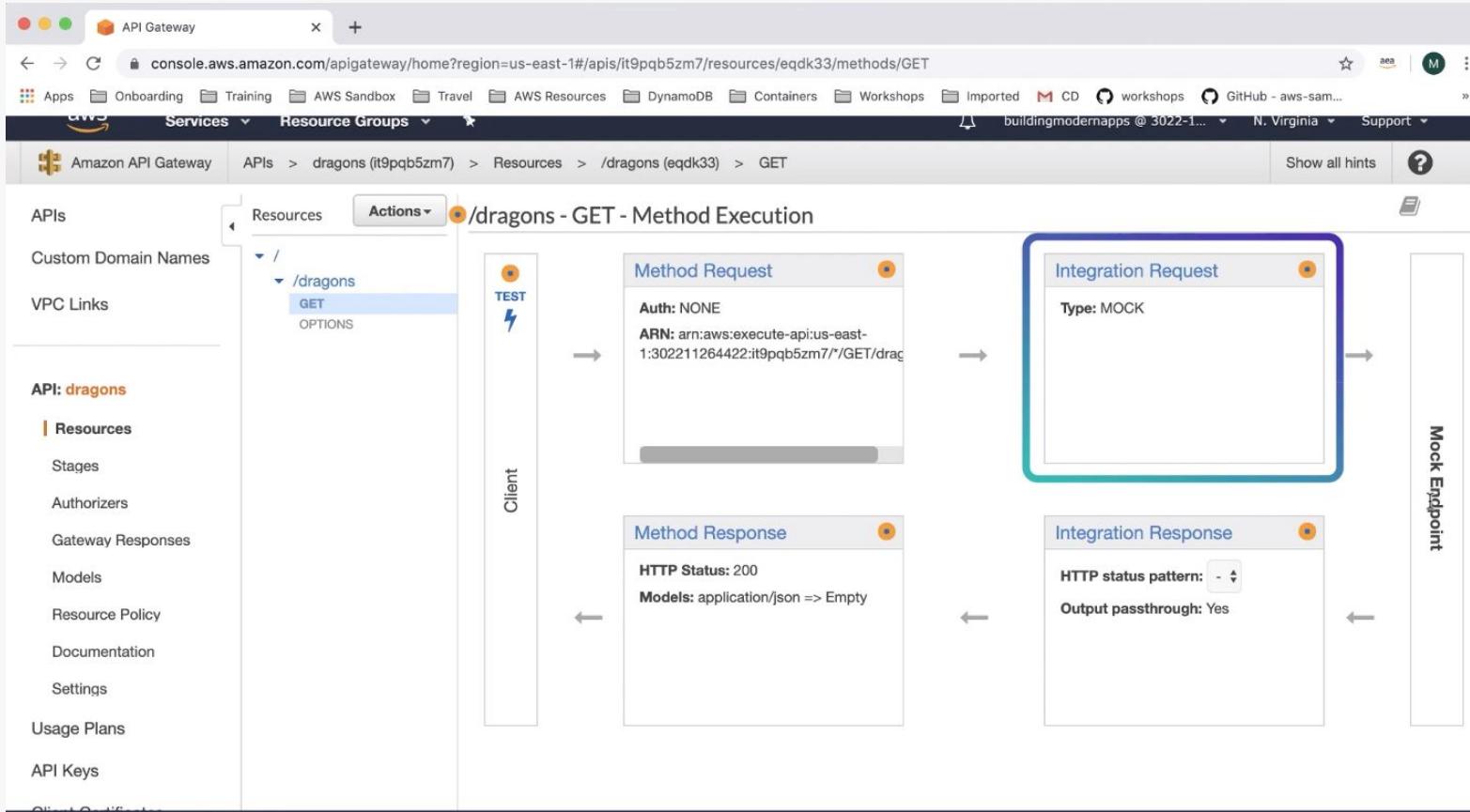
We can hardcode the responses we expect and test it before building and integrating with a real backend.



Now we are brought to this screen, which shows the flow of the request and the response with API Gateway.



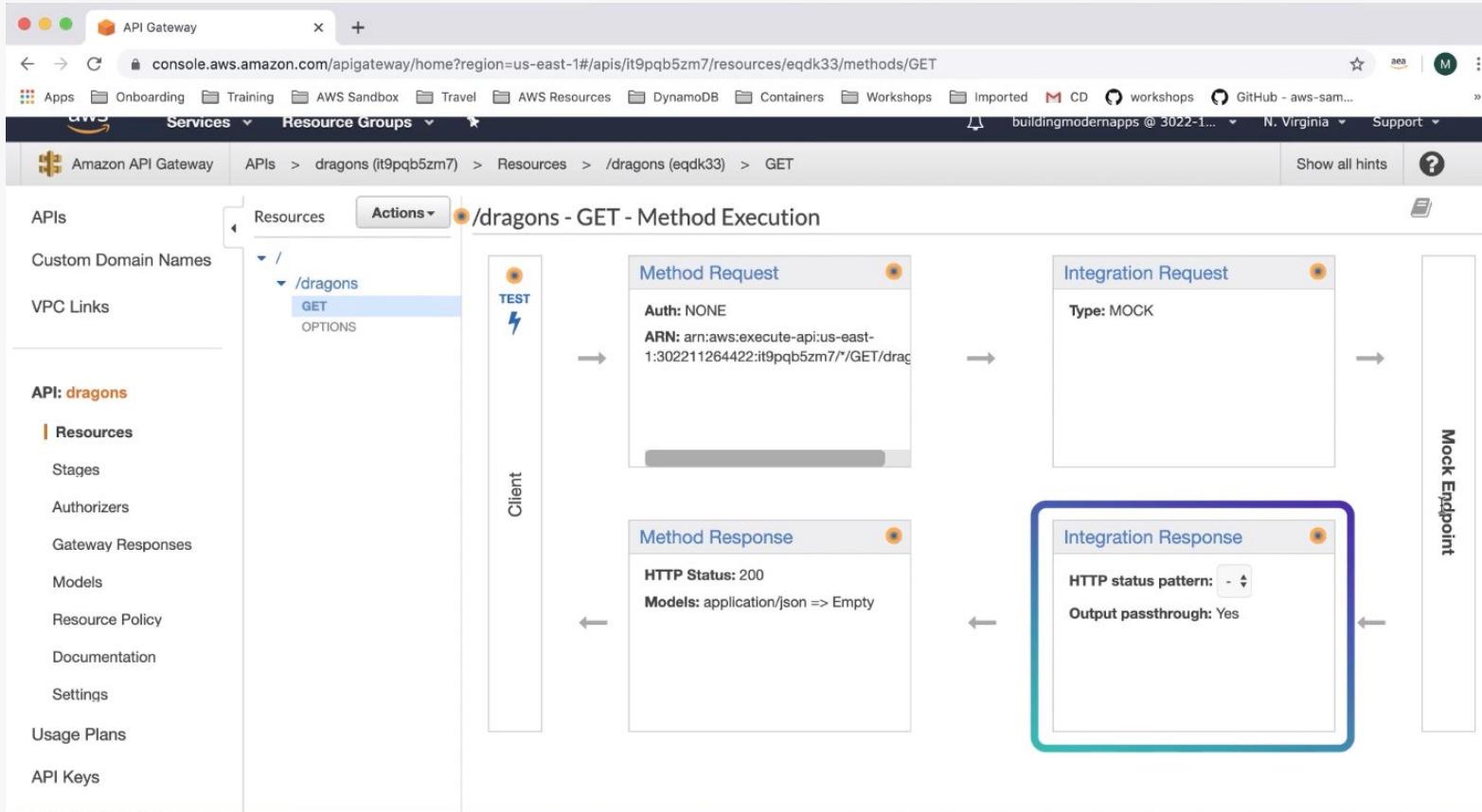
Method request is the first step for API Gateway to accept a request. This is where you can apply authorization and data payload validation.



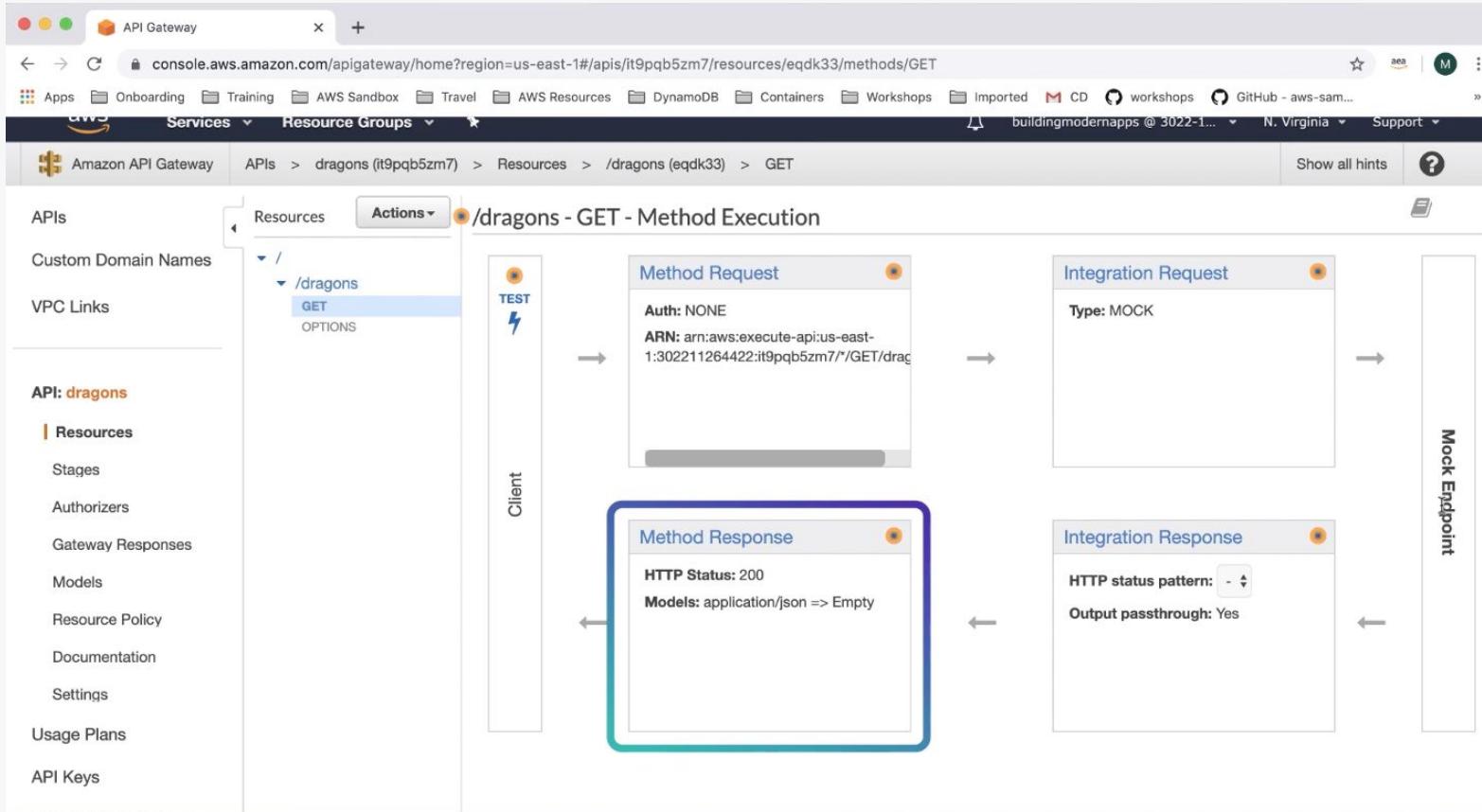
Next, it is passed to the integration request. This is where you configure what backend service you are fronting with API Gateway, as well as applying any sort of data transformations that you may require.

Once the request has passed the validation and the authorization and the transformation steps, it will then send the request to the backend service.

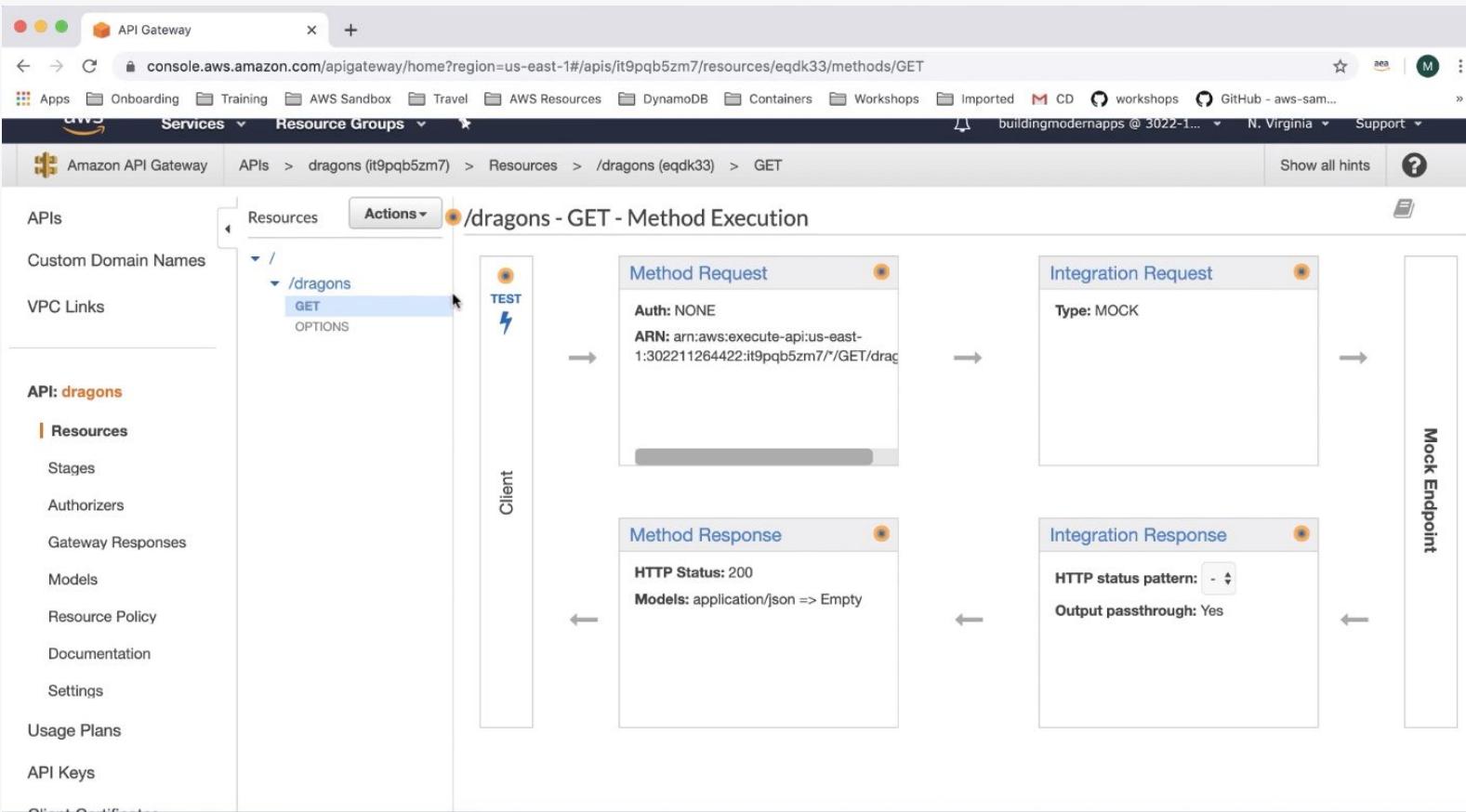
In our case, since this is a mock endpoint,  
it will take the request, do nothing and then send a response.



Integration response is an HTTP response encapsulating the backend response. You can configure how your backend service responses map to HTTP responses and apply data transformations at this step.



Method responses are similar to method requests. They're responsible for validating and fitting responses to models.



We can actually go ahead and just click on the test.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

**API: dragons**

**Resources**

Stages  
Authorizers  
Gateway Responses  
Models  
Resource Policy  
Documentation  
Settings

Usage Plans  
API Keys  
Client Certificates  
Settings

{myPathParam} in a resource path.

**Query Strings**

{dragons}

param1=value1&param2=value2

**Headers**

{dragons}

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. e.g.  
Accept:application/json.

**Stage Variables**

No stage variables exist for this method.

**Client Certificate**

No client certificates have been generated.

**Request Body**

Request Body is not supported for GET methods.

**Test**

Leave all blank, and click test

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

### API: dragons

#### Resources

- Stages
- Authorizers
- Gateway Responses
- Models
- Resource Policy
- Documentation
- Settings

#### Usage Plans

#### API Keys

#### Client Certificates

#### Settings

#### Query Strings

{dragons}

```
param1=value1&param2=value2
```

#### Headers

{dragons}

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg.  
Accept:application/json.

#### Stage Variables

No stage variables exist for this method.

#### Client Certificate

No client certificates have been generated.

#### Request Body

Request Body is not supported for GET methods.

 Test

#### Response Body

no data

#### Response Headers

```
{"Content-Type": "application/json"}
```

#### Logs

```
Execution log for request f6968935-3c8f-40c7-815c-f59a75b57fde
Fri May 15 21:44:28 UTC 2020 : Starting execution for request: f6968935-3c8f-40c7-815c-f59a75b57fde
Fri May 15 21:44:28 UTC 2020 : HTTP Method: GET, Resource Path: /dragons
Fri May 15 21:44:28 UTC 2020 : Method request path: {}
Fri May 15 21:44:28 UTC 2020 : Method request query string: {}
Fri May 15 21:44:28 UTC 2020 : Method request headers: {}
Fri May 15 21:44:28 UTC 2020 : Method request body before transformations:
Fri May 15 21:44:28 UTC 2020 : Method response body after transformations:
Fri May 15 21:44:28 UTC 2020 : Method response headers: {Content-Type=application/json}
Fri May 15 21:44:28 UTC 2020 : Successfully completed execution
Fri May 15 21:44:28 UTC 2020 : Method completed with status: 200
```

Now, we have an API for our dragon resource with the GET method.  
Again, this is mocked up. It's not doing anything real yet in the background,  
but we are going to continue to Mock up our dragon API,  
and then we will back it with the Lambda functions later on in the course.

# Models and Mapping

API Gateway has a lot of great features that allow you to offload some of the burden that might otherwise be on your back end services.

For example, it's oftentimes the back end service that is checking the incoming payload for required fields, ensuring data is not null, or checking that the data is formatted in a particular way, in order for the code to run properly.

API Gateway REST APIs provide a way to validate incoming requests against **models**.

And data can be transformed in shape by using API Gateway **mappings**.

Let's start with models

**Models** in API Gateway define the structure or shape of the payload of the request.

Models are created using JSON schemas,  
which allow you to define the properties of the payload and their types.

When a request comes in, it will be validated against that schema.

And if API Gateway sees that the data cannot be fit into the JSON schema, then API Gateway will return a 400-error response code to the client.

This frees up your back end  
from having to do the sort of basic data validation.

Let's take a look at an example.

```
{  
  "dragonName": "Frank",  
  "description": "This dragon is brand  
                 new, we don't know  
                 much about it yet.",  
  "family": "purple",  
  "city": "Seattle",  
  "country": "USA",  
  "state": "WA",  
  "neighborhood": "Downtown",  
  "reportingPhoneNumber": "15555555555",  
  "confirmationRequired": false  
}
```

This is what a JSON payload for reporting a new dragon would look like.

{

```
"dragonName": "Frank",  
"description": "This dragon is brand  
new, we don't know  
much about it yet.",
```

```
"family": "purple",  
"city": "Seattle",  
"country": "USA",  
"state": "WA",  
"neighborhood": "Downtown",  
"reportingPhoneNumber": "15555555555",  
"confirmationRequired": false
```

```
{  
  "$schema": "http://json-schema.org/draft-07/schema#",  
  "title": "Dragon",  
  "type": "object",  
  "properties": {  
    "dragonName": {  
      "type": "string"  
    },  
    "description": {  
      "type": "string"  
    },  
    "family": {  
      "type": "string"  
    },  
    "city": {  
      "type": "string"  
    },  
    "country": {  
      "type": "string"  
    },  
    "state": {  
      "type": "string"  
    },  
    "neighborhood": {  
      "type": "string"  
    },  
    "reportingPhoneNumber": {  
      "type": "string"  
    },  
    "confirmationRequired": {  
      "type": "boolean"  
    }  
  }  
}
```

And this is what the JSON schema for that payload would look like.  
This schema is used to do request validation.

```
{  
  "$schema": "http://json-schema.org/draft-07/schema#",  
  "title": "Person Contact Information",  
  "type": "object",  
  "properties": {  
    "firstName": {  
      "type": "string",  
      "description": "The first name of the person."  
    },  
    "lastName": {  
      "type": "string",  
      "description": "The last name of the person."  
    },  
    "email": {  
      "type": "string",  
      "format": "email",  
      "description": "The email address of the person."  
    },  
    "phone": {  
      "type": "string",  
      "pattern": "^\d{3}-\d{3}-\d{4}$",  
      "description": "The phone number of the person in the format XXX-XXX-XXXX."  
    }  
  },  
  "required": ["firstName", "lastName", "email"]  
}
```

A Contact Information JSON schema using the JSON Schema Draft 7 standard.

Models can be applied to both method requests and method responses.

Note that each method for resource like GET, POST, etc. could have different models.  
So these get applied at the method level, not at the resource level.

Now, let's talk about **Mapping**.

It's fairly common to run into situations where your back end services are expecting incoming data in a different format than the client is sending, or vice versa.

Instead of having to change your code to support the clients data structure,  
wouldn't it be nice if API Gateway could handle that for you?

Luckily, it can. Mappings does this.

**Mappings** are applied to the integration request  
and integration response of your API.

Mappings are written in Velocity Template Language, or VTL.

If you already have defined a model for the method,  
API Gateway can generate a VTL blueprint for the mapping which you can then modify.

A mapping template assumes the data coming in as a JSON object by default, and mappings do support other data types like XML.

Mappings support conditional statements,  
can inject new parameters into the payload,  
can hardcode values, which is needed for mocking,  
map data in complex structures,  
and can even reference data made available at runtime,  
such as context and stage variables which we will cover later.

# Creating a GET API with Mock Integration

In the last section, we created a REST API for the dragon resource and setting it as a mock endpoint.

But, there isn't any data to provide a response.

In this section, we're going to continue that example by adding a method for response with a mock integration in API Gateway.

A mock integration enables your API to return a response for a request directly, without the need for a resource on the backend.

This is a way to develop the API independently from the other parts of your distributed application.



Services

Resource Groups

Amazon API Gateway

APIs &gt; dragons (nl18eym14c) &gt; Resources &gt; /dragons (o5zw4d) &gt; GET



buildingmodernapps @ 3022-1...

APIs

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Dashboard

Settings

Usage Plans

API Keys

Client Certificates

Settings

Resources

Actions

## /dragons - GET - Method Execution

/

dragons

GET

OPTIONS

POST



Client

## Method Request

Auth: demo

ARN: arn:aws:execute-api:us-east-1:302211264422:nl18eym14c/"GET/dragons



## Method Response

HTTP Status: 200

Models: application/json =&gt; Empty



## Integration Request

Type: MOCK



## Integration Response

HTTP status pattern: -

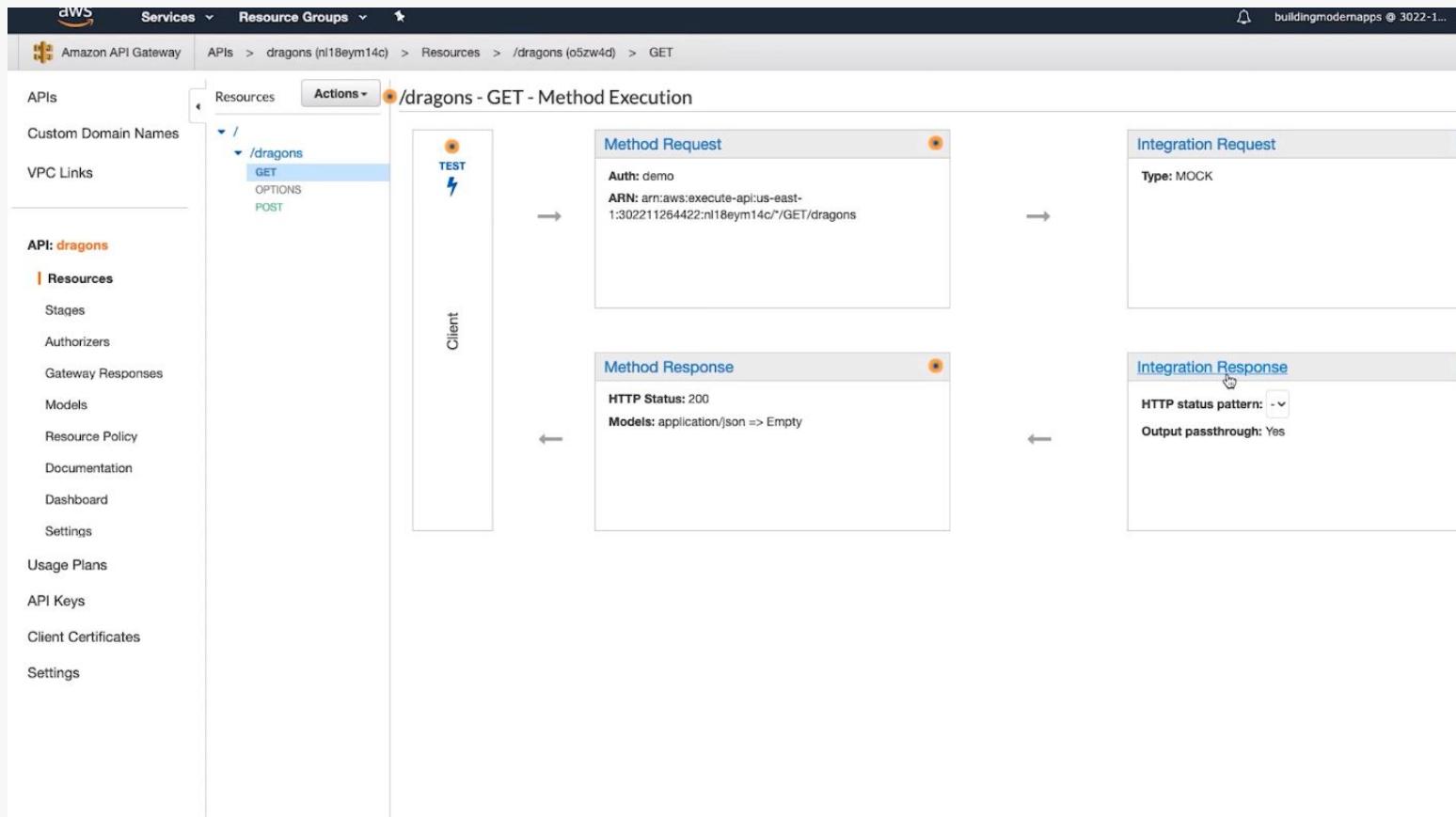
Output passthrough: Yes



Having a mock endpoint return a blank response,  
isn't very helpful for testing.

So, we are going to be mocking what a real backend would do.

So now that we have an empty mock backend established,  
I want to set up the response for the integration.



Go into the integration response

Amazon API Gateway APIs > dragons (nl18eym14c) > Resources > /dragons (o5zw4d) > GET

buildingmoderndevs@3022-1...

APIs / Custom Domain Names VPC Links

API: dragons

Resources Stages Authorizers Gateway Responses Models Resource Policy Documentation Dashboard Settings Usage Plans API Keys Client Certificates Settings

Actions < Method Execution /dragons - GET - Integration Response

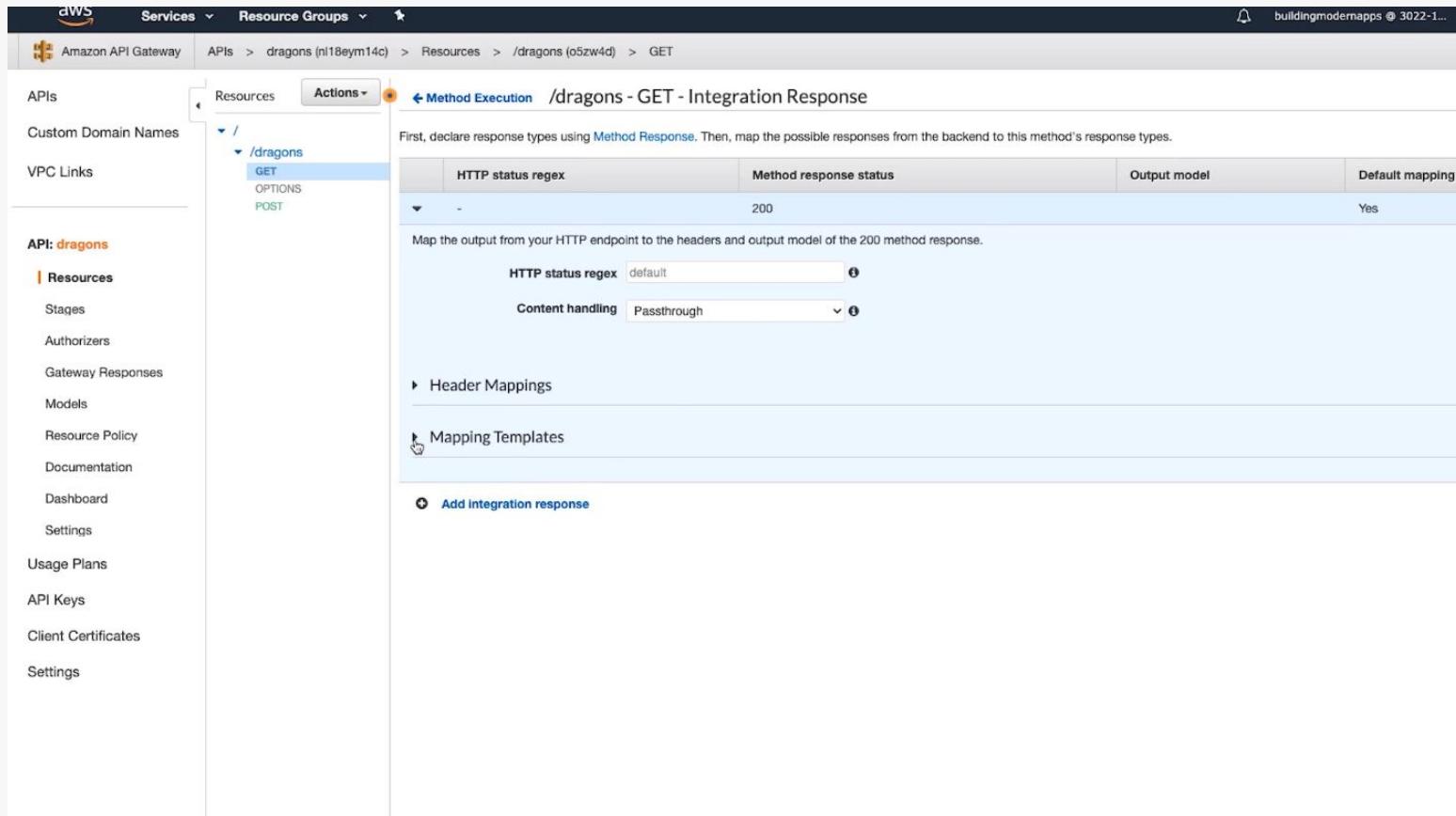
First, declare response types using [Method Response](#). Then, map the possible responses from the backend to this method's response types.

	HTTP status regex	Method response status	Output model	Default mapping
-		200		Yes

Map the output from your HTTP endpoint to the headers and output model of the 200 method response.

HTTP status regex: default Content handling: Passthrough

Header Mappings Mapping Templates Add integration response



Expand the 200 method response status that we see already there.

Screenshot of the AWS API Gateway console showing the configuration of a GET method for the '/dragons' resource.

The left sidebar shows the navigation path: APIs > dragons (nl18eym14c) > Resources > /dragons > GET.

The main panel title is "Method Execution /dragons - GET - Integration Response".

Instructions: "First, declare response types using Method Response. Then, map the possible responses from the backend to this method's response types."

HTTP status regex	Method response status	Output model	Default mapping
-	200	-	Yes

Configuration details:

- HTTP status regex: default
- Content handling: Passthrough

Expansion sections:

- Header Mappings
- Mapping Templates
  - Content-Type
    - application/json
  - Add mapping template
- Add integration response

Select application/json that already exists under the content type.

Screenshot of the AWS API Gateway console showing the configuration of a GET method for the '/dragons' resource.

The left sidebar shows the navigation path: Services > Resource Groups > APIs > dragons (nl18eym14c) > Resources > /dragons > GET.

The main panel title is "Method Execution /dragons - GET - Integration Response".

Table headers:

	HTTP status regex	Method response status	Output model	Default mapping
--	-------------------	------------------------	--------------	-----------------

Table data:

		200		Yes
--	--	-----	--	-----

Content handling: Passthrough

Header Mappings:

Mapping Templates:

Content-Type: application/json

Generate template: [ ]

```
1 - [
2   "description_str": "From the northern fire tribe, Atlas was born from",
3   "father_in_combat": "his father in combat. He is fearless and does not fear battle.",
4   "dragon_name_str": "Atlas",
5   "family_str": "red",
6   "location_city_str": "anchorage",
7   "location_country_str": "usa",
8   "location_neighborhood_str": "w fireweed ln",
9   "location_state_str": "alaska"
10  },
11  [
12    "description_str": "Protheus is a wise and ancient dragon that serves",
13    "the_sky_world": "the sky world. He uses his power to calm those near him.",
14    "dragon_name_str": "Protheus",
15    "family_str": "blue",
16    "location_city_str": "brandon",
17    "location_country_str": "usa",
18    "location_neighborhood_str": "e morgan st",
19    "location_state_str": "florida"
20  ]
```

Put your hard-coded response. The data is static, but it gives the appearance of a working API.

Screenshot of the AWS API Gateway Method Execution page for the /dragons - GET method.

The left sidebar shows the API structure:

- APIs
- Custom Domain Names
- VPC Links
- API: dragons**
  - Resources**
  - Stages
  - Authorizers
  - Gateway Responses
  - Models
  - Resource Policy
  - Documentation
  - Dashboard
  - Settings
  - Usage Plans
  - API Keys
  - Client Certificates
  - Settings

The main content area shows the method configuration and test results:

- Path:** /dragons
- Request:** /dragons
- Status:** 200
- Latency:** 26 ms
- Response Body:**

```
[  
  {  
    "description_str": "From the northern fire tribe, Atlas was born from a fallen father in combat. He is fearless and does not fear battle.",  
    "dragon_name_str": "Atlas",  
    "family_str": "red",  
    "location_city_str": "anchorage",  
    "location_country_str": "usa",  
    "location_neighborhood_str": "w fireweed ln",  
    "location_state_str": "alaska"  
  },  
  {  
    "description_str": "Protheus is a wise and ancient dragon that serves as a council in the sky world. He uses his power to calm those near him.",  
    "dragon_name_str": "Protheus",  
    "family_str": "blue",  
    "location_city_str": "brandon",  
    "location_country_str": "usa",  
    "location_neighborhood_str": "e morgan st",  
    "location_state_str": "florida"  
  },  
  {  
    "description_str": "Xanya is the fire tribe's banished general. She has been wandering ever since.",  
    "dragon_name_str": "Xanya",  
    "family_str": "red",  
    "location_city_str": "las vegas",  
    "location_country_str": "usa",  
    "location_neighborhood_str": "e clark ave",  
    "location_state_str": "nevada"  
  }]
```
- Query Strings:** [dragons] param1=value1&param2=value2
- Headers:** [dragons] Use a colon (:) to separate header name and value, and new lines to declare multiple headers. e.g. Accept:application/json.
- Stage Variables:** No stage variables exist for this method.
- Client Certificate:** No client certificates have been generated.
- Request Body:** Request Body is not supported for GET methods.

A blue "Test" button is located at the bottom center of the main content area.

And after clicking test, we see the hard-coded data is coming back in the side pane.

What you'll see next is the addition of the functionality for the query parameters, that enable querying the data by dragon family and dragon name.

# Dragon API: Using Mappings

Now that you have an API created for your dragon data and a basic GET method that returns some hard coded data what we need to do now is ensure that our mocked API can account for the fact that when we submit a GET request to our API, we could be trying to list all dragons, list dragons by family, or list dragons by name.

One method will be handling all three of these use cases.

When you submit a request,  
you will include a query parameter on the request, like this:

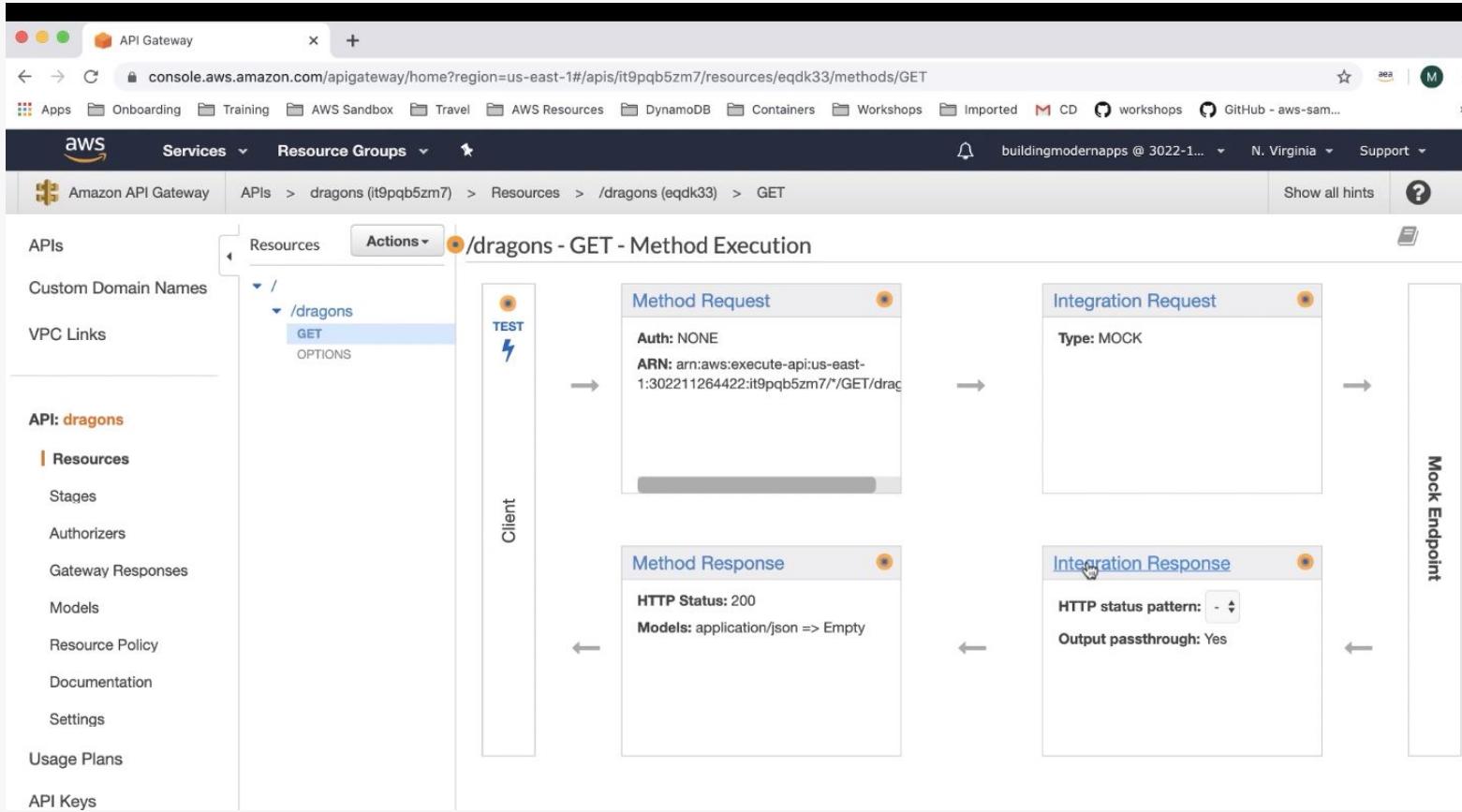
/dragon?dragonName=Atlas

/dragon?family=red

We must modify the mock endpoint to respond with different data  
if one of these query parameters is present.

To make that happen, what I need to take advantage of is the mappings that were applied to the integration response.

I'm going to modify the integration response to check the query parameters using conditionals in VTL.



Click on the Integration Response.

The screenshot shows the AWS API Gateway console with the URL `console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET`. The left sidebar is titled 'Resources' and lists various API components. The main configuration screen shows the following settings:

- HTTP status regex:** default
- Content handling:** Passthrough

Under the 'Mapping Templates' section, there is a table for 'Content-Type':

Content-Type
application/json

A button labeled '+ Add mapping template' is available.

The 'application/json' mapping template content is as follows:

```
1  {
2    "description_str": "From the northern fire tribe, Atlas was born from the ashes of his fallen father in combat. He is fearless and does not fear battle.",
3    "dragon_name_str": "Atlas",
4    "family_str": "red",
5    "location_city_str": "anchorage",
6    "location_country_str": "usa",
7    "location_neighborhood_str": "w fireweed ln",
8    "location_state_str": "alaska"
9  },
10  {
11    "description_str": "Proteus is a wise and ancient dragon that serves on the grand council in the sky world. He uses his power to calm those near him.",
12    "dragon_name_str": "Proteus",
13    "family_str": "blue",
14    "location_city_str": "brandon".
15 }
```

At the bottom right of the mapping template editor are 'Cancel' and 'Save' buttons.

Then navigate to the existing mapping, which is under Mapping Templates, click on application/json.

The screenshot shows the AWS API Gateway console with the URL `console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET`. The page displays a JSON mapping template for a GET request. The template contains four entries, each representing a dragon with its description, name, family, and location details. The entries are numbered 1 through 37. The last entry (line 37) is a placeholder object with two properties: 'description\_str' and 'location\_state\_str'. At the bottom right, there are 'Cancel' and 'Save' buttons.

```
1  {
2    "description_str": "From the northern fire tribe, Atlas was born from the ashes of his fallen father in combat. He is fearless and does not fear battle.",
3    "dragon_name_str": "Atlas",
4    "family_str": "red",
5    "location_city_str": "anchorage",
6    "location_country_str": "usa",
7    "location_neighborhood_str": "w fireweed ln",
8    "location_state_str": "alaska"
9  },
10  {
11    "description_str": "Protheus is a wise and ancient dragon that serves on the grand council in the sky world. He uses his power to calm those near him.",
12    "dragon_name_str": "Protheus",
13    "family_str": "blue",
14    "location_city_str": "brandon",
15    "location_country_str": "usa",
16    "location_neighborhood_str": "e morgan st",
17    "location_state_str": "florida"
18  },
19  {
20    "description_str": "Xanya is the fire tribe's banished general. She broke ranks and has been wandering ever since.",
21    "dragon_name_str": "Xanya",
22    "family_str": "red",
23    "location_city_str": "las vegas",
24    "location_country_str": "usa",
25    "location_neighborhood_str": "e clark ave",
26    "location_state_str": "nevada"
27  },
28  {
29    "description_str": "Eislex flies with the fire sprites. He protects them and is their guardian.",
30    "dragon_name_str": "Eislex",
31    "family_str": "red",
32    "location_city_str": "st. cloud",
33    "location_country_str": "usa",
34    "location_neighborhood_str": "breckenridge ave",
35    "location_state_str": "minnesota"
36  },
37  { "description_str": "", "location_state_str": "" }
```

This is the existing mapping we have right now.

Mappings are written in VTL.

This means we can use conditionals in the mapping.

The screenshot shows the AWS API Gateway Lambda function editor. The URL in the browser is `console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET`. The page displays a code editor with a template for a Lambda function. The code uses AWS Lambda's \$input.params conditional logic to return different dragon details based on the input parameters.

```
1+ [
2+   #if( $input.params('family') == "red" )
3+   {
4+     "description_str": "Xanya is the fire tribe's banished general. She broke ranks and has been wandering ever since.",
5+     "dragon_name_str": "Xanya",
6+     "family_str": "red",
7+     "location_city_str": "las vegas",
8+     "location_country_str": "usa",
9+     "location_neighborhood_str": "e clark ave",
10+    "location_state_str": "nevada"
11+  },
12+  {
13+    "description_str": "Eislex flies with the fire sprites. He protects them and is their guardian.",
14+    "dragon_name_str": "Eislex",
15+    "family_str": "red",
16+    "location_city_str": "st. cloud",
17+    "location_country_str": "usa",
18+    "location_neighborhood_str": "breckenridge ave",
19+    "location_state_str": "minnesota"
20+  },
21+  #elseif( $input.params('family') == "blue" )
22+  {
23+    "description_str": "Protheus is a wise and ancient dragon that serves on the grand council in the sky world. He uses his power to calm those near him.",
24+    "dragon_name_str": "Protheus",
25+    "family_str": "blue",
26+    "location_city_str": "brandon",
27+    "location_country_str": "usa",
28+    "location_neighborhood_str": "e morgan st",
29+    "location_state_str": "florida"
30+  },
31+  #elseif( $input.params('dragonName') == "Atlas" )
32+  {
33+    "description_str": "From the northern fire tribe, Atlas was born from the ashes of his fallen father in combat. He is fearless and does not fear battle.",
34+    "dragon_name_str": "Atlas",
35+    "family_str": "red",
36+    "location_city_str": "anchorage",
37+    "location_country_str": "usa",
38+    "location_neighborhood_str": "w fireweed ln",
39+    "location_state_str": "alaska"
40+
41+
```

At the bottom right of the code editor are two buttons: "Cancel" and "Save".

We added some conditionals and it is checking against this `$input.params`.

So what is this **\$input**?

API Gateway provides variables that start with a \$ sign  
that give you access to payload and context information in your mappings.

dragon is the \$input here:

/dragon?dragonName=Atlas

/dragon?family=red

Generate template: 

```
1+ [  
2+     #if( $input.params('family') == "red" )  
3+     {  
4+         "description_str":"Xanya is the fire tribe's banished general. She broke ranks and has been wandering ever since.",  
5+         "dragon_name_str":"Xanya",  
6+         "family_str":"red",  
7+         "location_city_str":"las vegas",  
8+         "location_country_str":"usa",  
9+         "location_neighborhood_str":"e clark ave",  
10+        "location_state_str":"nevada"  
11+    }, {  
12+        "description_str":"Eislex flies with the fire sprites. He protects them and is their guardian.",  
13+        "dragon_name_str":"Eislex",  
14+        "family_str":"red",  
15+        "location_city_str":"st. cloud",  
16+        "location_country_str":"usa",  
17+        "location_neighborhood_str":"breckenridge ave",  
18+        "location_state_str":"minnesota"    }  
19+    #elseif( $input.params('family') == "blue" )  
20+    {  
21+        "description_str":"Protheus is a wise and ancient dragon that serves on the grand council in the sky world. He uses his power to calm those near him.",  
22+        "dragon_name_str":"Protheus",  
23+        "family_str":"blue",  
24+        "location_city_str":"brandon",  
25+        "location_country_str":"usa",  
26+        "location_neighborhood_str":"e morgan st",  
27+        "location_state_str":"florida"  
28+    }  
29+    #elseif( $input.params('dragonName') == "Atlas" )  
30+    {  
31+        "description_str":"From the northern fire tribe, Atlas was born from the ashes of his fallen father in combat. He is fearless and does not fear battle.",  
32+        "dragon_name_str":"Atlas",  
33+        "family_str":"red",  
34+        "location_city_str":"anchorage",  
35+        "location_country_str":"usa",  
36+        "location_neighborhood_str":"w fireweed ln",  
37+        "location_state_str":"alaska"  
38+    }  
39+
```

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET

Custom Domain Names

VPC Links

**API: dragons**

**Resources**

- Stages
- Authorizers
- Gateway Responses
- Models
- Resource Policy
- Documentation
- Settings
- Usage Plans
- API Keys
- Client Certificates
- Settings

/

dragons

GET

OPTIONS

Path

No path parameters exist for this resource. You can define path parameters by using the syntax `{myPathParam}` in a resource path.

Query Strings

`{dragons}`

dragonName=Atlas

Headers

`{dragons}`

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. e.g.  
Accept:application/json.

Stage Variables

No stage variables exist for this method.

Client Certificate

No client certificates have been generated.

Request Body

add in some query parameters to the test

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD GitHub - aws-sam...

**API: dragons**

**Resources**

- Stages
- Authorizers
- Gateway Responses
- Models
- Resource Policy
- Documentation
- Settings
- Usage Plans
- API Keys
- Client Certificates
- Settings

**OPTIONS**

No path parameters exist for this resource. You can define path parameters by using the syntax `{myPathParam}` in a resource path.

**Query Strings**

`{dragons}`

dragonName=Atlas

**Headers**

`{dragons}`

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg.  
Accept:application/json.

**Stage Variables**

No [stage variables](#) exist for this method.

**Client Certificate**

No client certificates have been generated.

**Request Body**

Request Body is not supported for GET methods.

**Status: 200**

Latency: 41 ms

**Response Body**

```
[  
  {  
    "description_str": "From the northern fire tribe, Atla  
s was born from the ashes of his fallen father in combat.  
He is fearless and does not fear battle.",  
    "dragon_name_str": "Atlas",  
    "family_str": "red",  
    "location_city_str": "anchorage",  
    "location_country_str": "usa",  
    "location_neighborhood_str": "w fireweed ln",  
    "location_state_str": "alaska"  
  }  
]
```

**Response Headers**

```
{"Content-Type": "application/json"}
```

**Logs**

```
Execution log for request 886f714a-be7b-4c22-822f-9753d859e3bc  
Fri May 15 21:58:29 UTC 2020 : Starting execution for requ  
est: 886f714a-be7b-4c22-822f-9753d859e3bc  
Fri May 15 21:58:29 UTC 2020 : HTTP Method: GET, Resource  
Path: /dragons  
Fri May 15 21:58:29 UTC 2020 : Method request path: {}  
Fri May 15 21:58:29 UTC 2020 : Method request query strin
```

we got back just the Atlas dragon.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/GET

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

OPTIONS

**API: dragons**

**Resources**

- Stages
- Authorizers
- Gateway Responses
- Models
- Resource Policy
- Documentation
- Settings
- Usage Plans
- API Keys
- Client Certificates
- Settings

No path parameters exist for this resource. You can define path parameters by using the syntax `{myPathParam}` in a resource path.

**Query Strings**

`{dragons}`

`family=red`

**Headers**

`{dragons}`

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg.  
Accept:application/json

**Stage Variables**

No stage variables exist for this method.

**Client Certificate**

No client certificates have been generated.

**Request Body**

Request Body is not supported for GET methods.

**Status: 200**

**Latency: 40 ms**

**Response Body**

```
[  
  {  
    "description_str": "Xanya is the fire tribe's banished general. She broke ranks and has been wandering ever since.",  
    "dragon_name_str": "Xanya",  
    "family_str": "red",  
    "location_city_str": "las vegas",  
    "location_country_str": "usa",  
    "location_neighborhood_str": "e clark ave",  
    "location_state_str": "nevada"  
  },  
  {  
    "description_str": "Eislex flies with the fire sprite. He protects them and is their guardian.",  
    "dragon_name_str": "Eislex",  
    "family_str": "red",  
    "location_city_str": "st. cloud",  
    "location_country_str": "usa",  
    "location_neighborhood_str": "breckenridge ave",  
    "location_state_str": "minnesota"  
  }  
]
```

**Response Headers**

```
{"Content-Type": "application/json"}
```

family=red

This API is responding as if there is a backend  
but there isn't.

This method is now fully mocked up and we can move on to the next method.

# DragonAPI: Using Models

Let's continue to build out our dragon API  
and move on to adding the method that will handle the reporting a new dragon.

First, add a new POST method to the dragons resource.

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam... buildingmodernapps @ 3022-1... N. Virginia Support

aws Services Resource Groups

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33) Show all hints ?

APIs Custom Domain Names VPC Links

**API: dragons**

**Resources**

Stages Authorizers Gateway Responses Models Resource Policy Documentation Settings Usage Plans API Keys

**Actions** /dragons Methods

**RESOURCE ACTIONS**

- Create Method
- Create Resource
- Enable CORS
- Edit Resource Documentation
- Delete Resource

**API ACTIONS**

- Deploy API
- Import API
- Edit API Documentation
- Delete API

**OPTIONS**

**Mock Endpoint**

Authorization None API Key Not required

This screenshot shows the AWS API Gateway console interface. The left sidebar lists various API management features like Stages, Authorizers, and Resource Policies. The main area shows an API named 'dragons' with a single resource at the path '/dragons'. A context menu is open over this resource, with the 'Actions' tab selected. Under 'RESOURCE ACTIONS', the 'Create Method' option is highlighted. The 'API ACTIONS' section contains options like Deploy API and Import API. To the right, a panel titled 'OPTIONS' displays settings for a 'Mock Endpoint', showing 'None' for Authorization and 'Not required' for API Key.

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam... M :

warn buildingmodernapps @ 3022-1... N. Virginia Support

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33) Show all hints ?

APIs Resources Actions / /dragons Methods

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Feedback English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

**/dragons**

GET Mock Endpoint

Authorization None

API Key Not required

OPTIONS Mock Endpoint

Authorization None

API Key Not required

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/POST

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam... buildingmodernapps @ 3022-1... N. Virginia Support

AWS Services Resource Groups

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33) > POST Show all hints ?

APIs

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Resources

Actions

/dragons - POST - Setup

Choose the integration point for your new method.

Integration type

- Lambda Function 
- HTTP 
- Mock 
- AWS Service 
- VPC Link 

Save

Feedback

English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Privacy Policy

Terms of Use

← → ⌂ console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/POST ⌂    M ⌂

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam... ⌂

aws Services Resource Groups ⌂

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33) > POST Show all hints ⌂

APIs Resources Actions / /dragons - POST - Method Execution

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

TEST /dragons - POST - Method Execution

Method Request

Auth: NONE

ARN: arn:aws:execute-api:us-east-1:302211264422:it9pqb5zm7/\*/POST/dragons

Integration Request

Type: MOCK

Client

Method Response

HTTP Status: 200

Models: application/json => Empty

Integration Response

HTTP status pattern: -

Output passthrough: Yes

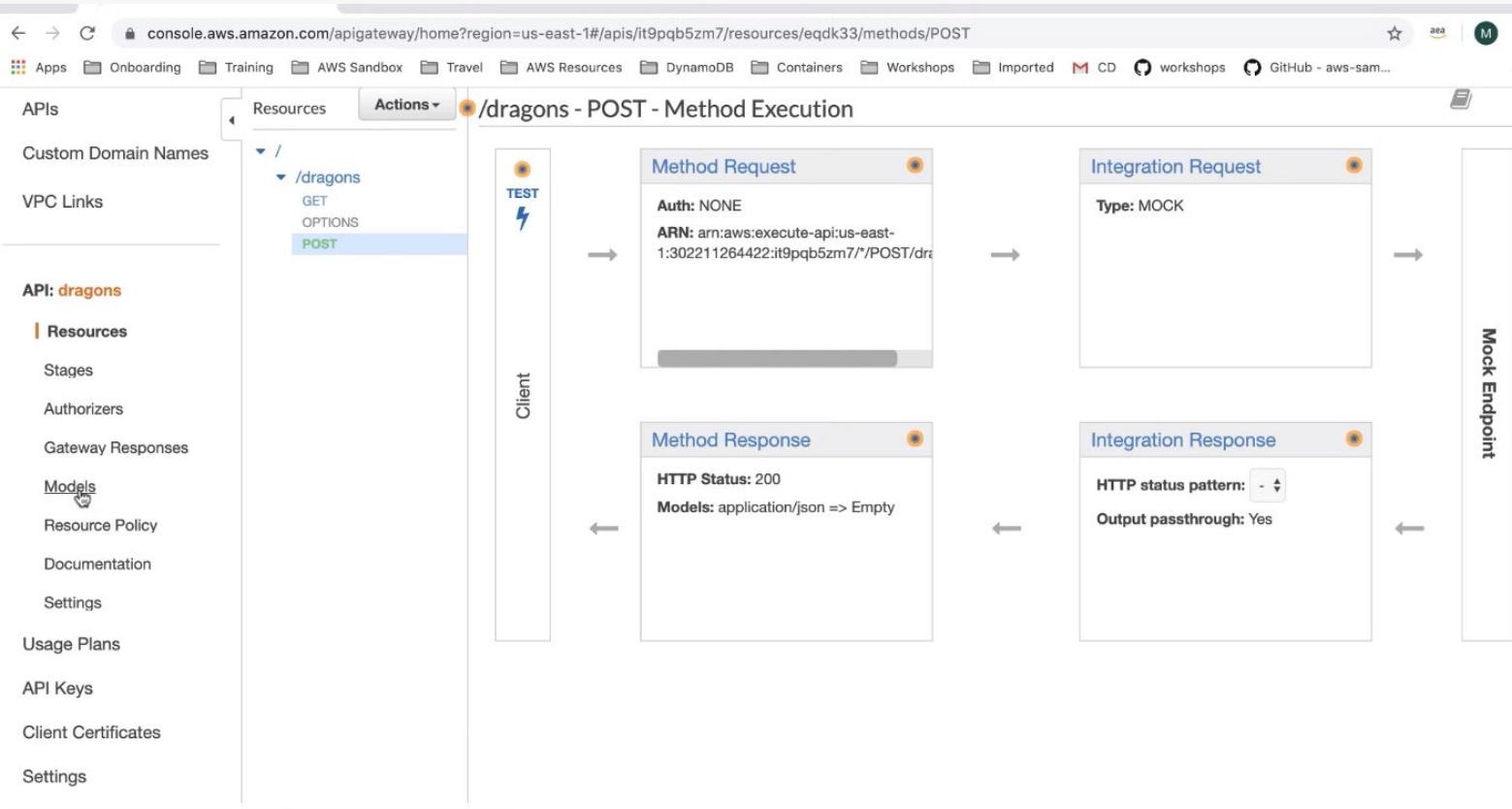
Mock Endpoint

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

This method is going to be used for reporting new dragons  
so there will be a payload on the request that has the new dragons information.

One of the features API Gateway has  
is to validate incoming requests based on models.

Let's use that feature with this request.



We first need to create a model. Click on the models section in the left-hand side.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/models

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

APIs

Custom Domain Names

VPC Links

API: dragons

- Resources
- Stages
- Authorizers
- Gateway Responses
- Models**
- Resource Policy
- Documentation
- Settings
- Usage Plans
- API Keys
- Client Certificates
- Settings

Select a model

Empty Error

Click on Create

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/models/create

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

APIs

Custom Domain Names

VPC Links

API: dragons

- Resources
- Stages
- Authorizers
- Gateway Responses
- Models**
- Resource Policy
- Documentation
- Settings
- Usage Plans
- API Keys
- Client Certificates
- Settings

New Model

Provide a name, content type, and a schema for your model. Models use JSON schema.

Model name\* My Model

Content type\* eg. application/json

Model description

Model schema\*

1

\* Required

Cancel Create

Feedback English (US)

© 2008–2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Service

Models in API Gateway are JSON schemas.

The screenshot shows the AWS API Gateway console at <https://console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/models/create>. The left sidebar is collapsed, and the main area is titled "New Model". It asks for a "Model name\*" (My Model), "Content type\*" (application/json), and a "Model description" (empty). Below is the "Model schema\*" section containing a JSON schema for a "Dragon" object:

```
1  {  
2      "$schema": "http://json-schema.org/draft-07/schema#",  
3      "title": "Dragon",  
4      "type": "object",  
5      "properties": {  
6          "dragonName": {  
7              "type": "string"  
8          },  
9          "description": {  
10             "type": "string"  
11         },  
12         "family": {  
13             "type": "string"  
14         },  
15         "city": {  
16             "type": "string"  
17         },  
18         "country": {  
19             "type": "string"  
20         }  
21     }  
22 }
```

At the bottom, there's a note "\* Required" and buttons for "Cancel" and "Create model". The footer includes links for Feedback, English (US), Privacy Policy, and Terms of Use.

I paste in a pre-written JSON schema for the incoming dragon data that we expect to come in for the POST request.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/models/create

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Models > Create Show all hints

APIs Models Create

Empty Error

## New Model

Provide a name, content type, and a schema for your model. Models use JSON schema.

Model name\* dragon

Content type\* application/json

Model description

### Model schema\*

```
1  {
2    "$schema": "http://json-schema.org/draft-07/schema#",
3    "title": "Dragon",
4    "type": "object",
5    "properties": {
6      "dragonName": {
7        "type": "string"
8      },
9      "description": {
10        "type": "string"
11      },
12      "family": {
13        "type": "string"
14      },
15      "city": {
16        "type": "string"
17      },
18      "country": {
19        "type": "string"
20      }
21    }
22  }
```

\* Required

Cancel Create model

Name this dragon, give it the content type application/JSON, and then click create model.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/models/dragon

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

APIs Models Create Delete Model

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Client Certificates

Settings

Update Model

Make changes to your model in the form below. Models are declared using JSON schema.

Model name dragon

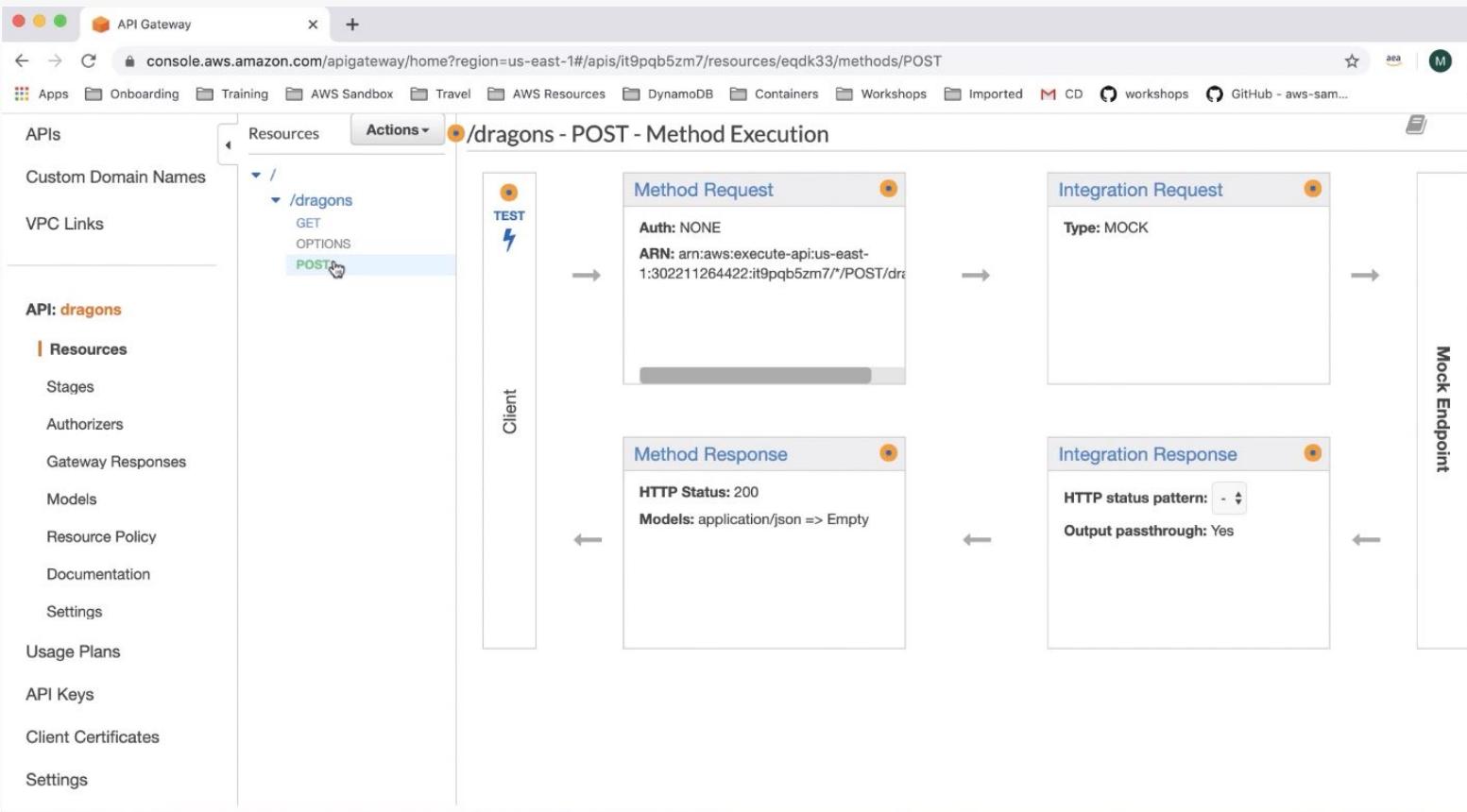
Content type application/json

Model description

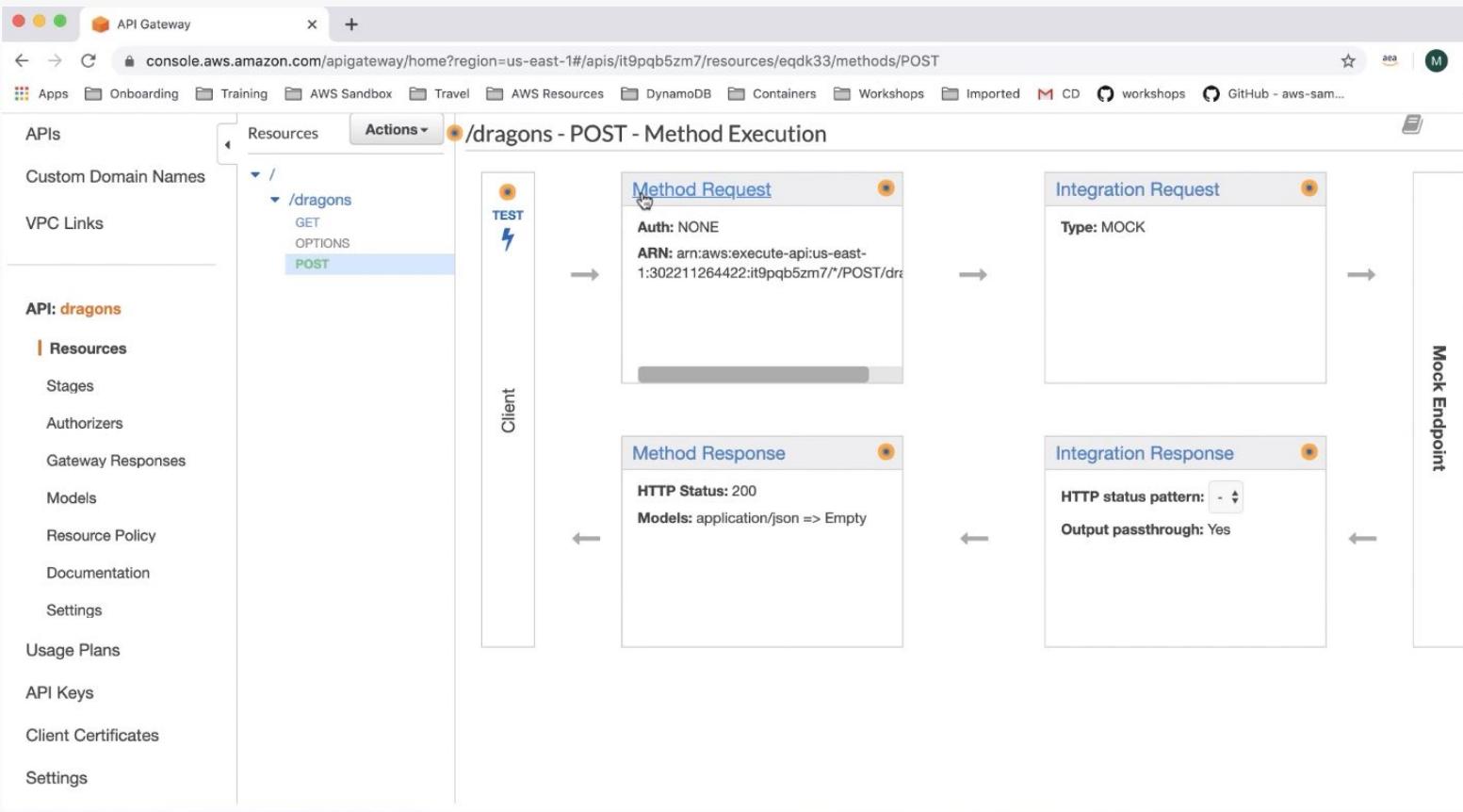
Model schema\*

```
1  {
2      "$schema": "http://json-schema.org/draft-07/schema#",
3      "title": "Dragon",
4      "type": "object",
5      "properties": {
6          "dragonName": {
7              "type": "string"
8          },
9          "description": {
10             "type": "string"
11         },
12         "family": {
13             "type": "string"
14         },
15         "city": {
16             "type": "string"
17         },
18         "country": {
19             "type": "string"
20     }
21 }
```

Based on this model,  
if the input deviates from the structure that we created,  
I want API Gateway to reject the request before it ever hits my backend.



Navigate back to the resources and then POST method



The place you apply request validation is on the method request.

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/POST

APIs    Resources    Actions

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Client Certificates

Settings

Method Execution /dragons - POST - Method Request

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization NONE

Request Validator NONE

API Key Required false

URL Query String Parameters

HTTP Request Headers

Request Body

Content type      Model name

No Models

Add model

SDK Settings

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/POST

APIs    Resources    Actions

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Client Certificates

Settings

Method Execution /dragons - POST - Method Request

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization NONE

Request Validator NONE

API Key Required false

URL Query String Parameters

HTTP Request Headers

Request Body

Content type	Model name
application/json	dragon

SDK Settings

Screenshot of the AWS API Gateway console showing the configuration of a POST method for the '/dragons' resource. The 'Authorization' section is set to 'NONE'. A dropdown menu for 'Request Validator' is open, with 'Validate body' selected. Other options shown are 'Validate body, query string parameters, and headers' and 'Validate query string parameters and headers'. The 'Content type' is set to 'application/json' and the 'Model name' is 'dragon'. The 'SDK Settings' section is also visible.

APIs

Custom Domain Names

VPC Links

**API: dragons**

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Client Certificates

Settings

Resources

Actions

Method Execution /dragons - POST - Method Request

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization NONE

Request Validator ✓ NONE

- Validate body
- API Key Required Validate body, query string parameters, and headers
- Validate query string parameters and headers

URL Query String Parameters

HTTP Request Headers

Request Body

Content type	Model name
application/json	dragon

Add model

SDK Settings

to actually validate this, select the request validator, select the validate body and then click save.

API Gateway x +

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/POST

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

APIs Resources Actions Method Execution /dragons - POST - Method Request

Custom Domain Names

VPC Links

API: dragons

Resources

- Stages
- Authorizers
- Gateway Responses
- Models
- Resource Policy
- Documentation
- Settings

Usage Plans

API Keys

Client Certificates

Settings

Provide information about this method's authorization settings and the parameters it can receive.

Settings

Authorization NONE edit info

Request Validator Validate body edit info

API Key Required false edit

URL Query String Parameters edit info

HTTP Request Headers

Request Body edit model

Content type	Model name	
application/json	dragon	<span>edit</span>

Add model

SDK Settings

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/POST

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

Documentation

Accept:application/json.

Settings

Usage Plans

API Keys

Client Certificates

Settings

Stage Variables

No [stage variables](#) exist for this method.

Client Certificate

No client certificates have been generated.

Request Body

1

Test

The screenshot shows the AWS API Gateway console with the URL `console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/POST`. The left sidebar contains links for Gateway Responses, Models, Resource Policy, Documentation, Settings, Usage Plans, API Keys, Client Certificates, and Settings. The main content area is titled "Headers" and contains a section for "dragons" with a note: "Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.". Below this is a "Stage Variables" section stating "No stage variables exist for this method." There is also a "Client Certificate" section stating "No client certificates have been generated." Under "Request Body", there is a text input field containing the number "1". To the right, under "Response Headers", is a JSON object: `{"x-amzn-ErrorType": "BadRequestException"}`. Below this is a "Logs" section displaying an execution log for a POST request to the /dragons endpoint:

```
Execution log for request 1629434d-b656-4ca3-a372-44485a474478
Fri May 15 22:07:52 UTC 2020 : Starting execution for request: 1629434d-b656-4ca3-a372-44485a474478
Fri May 15 22:07:52 UTC 2020 : HTTP Method: POST, Resource Path: /dragons
Fri May 15 22:07:52 UTC 2020 : Method request path: {}
Fri May 15 22:07:52 UTC 2020 : Method request query string: {}
Fri May 15 22:07:52 UTC 2020 : Method request headers: {}
Fri May 15 22:07:52 UTC 2020 : Method request body before transformations:
Fri May 15 22:07:52 UTC 2020 : Request body does not match model schema for content type application/json: [Unknown error parsing request body]
Fri May 15 22:07:52 UTC 2020 : Method completed with status: 400
```

This error is what we would expect here because this is an empty request body

API Gateway

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33/methods/POST

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

### Gateway Responses

Models  
Resource Policy  
Documentation  
Settings  
Usage Plans  
API Keys  
Client Certificates  
Settings

### Headers

{dragons}

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg.  
Accept:application/json.

### Stage Variables

No stage variables exist for this method.

### Client Certificate

No client certificates have been generated.

### Request Body

```
1+ {  
2     "dragonName": "Frank",  
3     "description": "This dragon is brand  
        new, we don't know much about it  
        yet.",  
4     "family": "purple",  
5     "city": "Seattle",  
6     "country": "USA",  
7     "state": "WA",  
8     "neighborhood": "Downtown",  
9     "reportingPhoneNumber":  
        "15555555555",  
10    "confirmationRequired": false  
11 }
```

### Logs

```
Execution log for request 923e6680-3d05-4b93-9f2a-92be053a0bb7  
Fri May 15 22:09:06 UTC 2020 : Starting execution for request: 923e6680-3d05-4b93-9f2a-92be053a0bb7  
Fri May 15 22:09:06 UTC 2020 : HTTP Method: POST, Resource Path: /dragons  
Fri May 15 22:09:06 UTC 2020 : Method request path: {}  
Fri May 15 22:09:06 UTC 2020 : Method request query string: {}  
Fri May 15 22:09:06 UTC 2020 : Method request headers: {}  
Fri May 15 22:09:06 UTC 2020 : Method request body before transformations:  
    "dragonName": "Frank",  
    "description": "This dragon is brand new, we don't know much about it yet.",  
    "family": "purple",  
    "city": "Seattle",  
    "country": "USA",  
    "state": "WA",  
    "neighborhood": "Downtown",  
    "reportingPhoneNumber": "15555555555",  
    "confirmationRequired": false  
Fri May 15 22:09:06 UTC 2020 : Request validation succeeded for content type application/json  
Fri May 15 22:09:06 UTC 2020 : Method response body after transformations:  
Fri May 15 22:09:06 UTC 2020 : Method response headers: {Content-Type=application/json}  
Fri May 15 22:09:06 UTC 2020 : Successfully completed exec
```

Successfully completed the execution with a sample request body.

API Gateway is now validating the incoming request against a model and it is responding with the GET method with those mappings that we already had created.

That means that this API is totally mocked up.

So now we can deploy it and interact with it as if it was our real API.

# Publish API

After creating your API, you must **deploy** it to make it **callable** by your users.

To deploy an API, you create an **API deployment** and associate it with a **stage**.

A **stage** is a logical reference to a lifecycle state of your API  
(for example, dev, prod, beta, v2).

API stages are identified by the **API ID** and **stage name**.

They're included in the URL that you use to invoke the API.

Every time you update an API,  
you must redeploy the API to an existing stage or to a new stage.

Updating an API includes modifying routes, methods, integrations, authorizers,  
and anything else other than stage settings.

Our API is built and mocked, so now let's deploy it to a stage.

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/it9pqb5zm7/resources/eqdk33

Apps Onboarding Training AWS Sandbox Travel AWS Resources DynamoDB Containers Workshops Imported CD workshops GitHub - aws-sam...

aws Services Resource Groups

Amazon API Gateway APIs > dragons (it9pqb5zm7) > Resources > /dragons (eqdk33)

Show all hints ?

APIs Resources Actions /dragons Methods

RESOURCES ACTIONS

- Create Method
- Create Resource
- Enable CORS
- Edit Resource Documentation
- Delete Resource

API ACTIONS

- Deploy API
- Import API
- Edit API Documentation
- Delete API

OPTIONS Mock Endpoint

Authorization None

API Key Not required

None

Not required

None

Not required

API Key Not required

API: dragons

Resources

- Stages
- Authorizers
- Gateway Responses
- Models
- Resource Policy
- Documentation
- Settings
- Usage Plans
- API Keys

https://console.aws.amazon.com/apigateway/home?region=us-east-1#

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Click actions and the click deploy API.



Services

Resource Groups



buildingmodernapps @ 3022-1...

N. Virginia

Support

Amazon API Gateway

APIs &gt; dragons (it9pqb5zn)

APIs

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Resources

Actions

/

/dragons

GET  
OPTIONS  
POST

## Deploy API

Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage

✓ [New Stage]

Deployment description

Cancel

Deploy

POST

Mock Endpoint

Authorization None

API Key Not required

create a new deployment stage



Services

Resource Groups



buildingmodernapps @ 3022-1...

N. Virginia

Support

Amazon API Gateway

APIs &gt; dragons (it9pb5zn)

APIs

Custom Domain Names

VPC Links

API: dragons

Resources

Stages

Authorizers

Gateway Responses

Models

Resource Policy

Documentation

Settings

Usage Plans

API Keys

Resources

Actions

/

/dragons

GET  
OPTIONS  
POST

## Deploy API

Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage

[New Stage]

Stage name\*

test

Stage description

Deployment description

Cancel

Deploy

Authorization: None

API Key Not required

Screenshot of the AWS API Gateway Stage Editor for the 'test' stage of the 'dragons' API.

The top navigation bar shows the AWS logo, Services dropdown, Resource Groups dropdown, and user information (buildingmodernapps @ 3022-1... N. Virginia Support). The left sidebar lists API resources: APIs, Custom Domain Names, VPC Links, and detailed sections for the selected API: dragons (Resources, Stages, Authorizers, Gateway Responses, Models, Resource Policy, Documentation, Dashboard, Settings, Usage Plans).

The main content area is titled "test Stage Editor". It displays the "Invoke URL" (https://it9pqb5zm7.execute-api.us-east-1.amazonaws.com/test) and includes tabs for Settings, Logs/Tracing, Stage Variables, SDK Generation, Export, Deployment History, Documentation History, and Canary.

The "Settings" tab is active, showing "Cache Settings" with an "Enable API cache" checkbox (unchecked). It also shows "Default Method Throttling" settings: "Enable throttling" (checked), "Rate: 10000 requests per second", and "Burst: 5000 requests".

Below these settings is a section for "Web Application Firewall (WAF)" with a "Learn more" link. A dropdown menu for selecting a Web ACL is shown, with "Web ACL" and "Create WAF ACL" options.

It gives you an invoke URL at the top. This is the endpoint for your API.

Now, we have our API deployed,  
and we have an active endpoint.

Now, let's test it out.

I don't want to use the test features built into the console, but instead.

Let's actually hit this endpoint  
from an external entity using a tool called **Postman**.

(You could use cURL on the command line as well)

Postman is not an AWS tool, but rather a tool used very widely in the developer community.

The Postman API client allows you to configure requests, invoke APIs and then view the result.

This is just to show you how to hit your endpoint  
using a service that is outside of the AWS environment,  
proving that your API is deployed and ready to go.

New Import Runner +

My Workspace Invite Upgrade

Filter History Collections APIs

Save Responses

You haven't sent any requests

Any request you send in this workspace will appear here.

Show me how

No Environment

Open Launchpad

For you

Learn how to monitor a collection | Start

Bootcamp Build Browse

The screenshot shows the Postman application interface. At the top, there's a navigation bar with 'New', 'Import', 'Runner', and other workspace-related options. Below the navigation is a search bar labeled 'Filter' and tabs for 'History', 'Collections', and 'APIs'. A 'Save Responses' toggle switch is also present. The main content area is titled 'You haven't sent any requests' and features a large circular icon with a pen nib and a play button, suggesting a video tutorial. Below this is a 'Show me how' button. Further down, there's a 'For you' section with a 'Start' button and a link to learn about monitoring collections. The bottom of the screen has a toolbar with icons for 'Bootcamp', 'Build', and 'Browse', along with other small icons.

Sign into the Postman app

New Import Runner

My Workspace Invite Upgrade

Filter History Collections APIs

Save Responses

You haven't sent any requests

Any request you send in this workspace will appear here.

Show me how

GET list-dragons

No Environment

Comments 0 Examples 0

GET https://l19pqb5zm7.execute-api.us-east-1.amazonaws.com/test/dragons

Sending...

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	***	Bulk Edit
Key	Value	Description	***	Bulk Edit

Response

Sending request...

Cancel

Hit Send to get a response

For you

Learn how to monitor a collection | Start

Bootcamp Build Browse

Filter History Collections APIs

New Import Runner + My Workspace Invite Upgrade

Filter History Collections APIs Save Responses Clear all Today https://it9pqb5zm7.execute-api.us-east-1.amazonaws.com/test/dragons

GET list-dragons No Environment Comments 0 Examples 0

GET https://it9pqb5zm7.execute-api.us-east-1.amazonaws.com/test/dragons Send Save

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 59ms Size: 1.81 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "description_str": "From the northern fire tribe, Atlas was born from the ashes of his fallen father in combat. He is fearless and does not fear battle.",
4     "dragon_name_str": "Atlas",
5     "family_str": "red",
6     "location_city_str": "anchorage",
7     "location_country_str": "usa",
8     "location_neighborhood_str": "w fireweed ln",
9     "location_state_str": "alaska"
10   },
11   {
12     "description_str": "Protheus is a wise and ancient dragon that serves on the grand council in the sky world. He uses his power to calm those near him.",
13     "dragon_name_str": "Protheus",
14     "family_str": "blue",
15     "location_city_str": "brandon",
16     "location_country_str": "usa",
```

Bootcamp Build Browse

New Import Runner +

My Workspace Invite Upgrade

Filter History Collections APIs

Save Responses Clear all

Today

https://it9pqb5zm7.execute-api.us-east-1.amazonaws.com/test/dragons?family=blue

GET https://it9pqb5zm7.execute-api.us-east-1.amazonaws.com/test/dragons?family=red

GET https://it9pqb5zm7.execute-api.us-east-1.amazonaws.com/test/dragons

GET list-dragons

No Environment

Comments 0 Examples 0

GET https://it9pqb5zm7.execute-api.us-east-1.amazonaws.com/test/dragons?family=blue

Send Save

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> family	blue			
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 76ms Size: 685 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [ ]  
2 {  
3     "description_str": "Protheus is a wise and ancient dragon that serves on the grand council in the sky world. He  
4         uses his power to calm those near him.",  
5     "dragon_name_str": "Protheus",  
6     "family_str": "blue",  
7     "location_city_str": "brandon",  
8     "location_country_str": "usa",  
9     "location_neighborhood_str": "e morgan st",  
10    "location_state_str": "florida"  
11 }
```

Bootcamp Build Browse

test with query parameter

To download and try out Postman for yourself click here:  
<https://www.postman.com/downloads>

## **Exercise 2: Amazon API Gateway**

In this lab, you will continue to build the Dragons application.

First, you will build the REST API that's used to list and add dragons.

You will create the API by using mock integrations.

You can use mock integrations to create a testable API  
before you write any code for your backend services.

After the API is created, you will deploy an updated version of the web application.

The web application contains the frontend logic to access  
the GET and POST methods of your /dragons resource.

Start from where you stopped at the end of Lab 1,  
and continue to build the application.

## Exercise 2: Amazon API Gateway

Upload a doc with a screenshot for each completed Task as a Lab report in Moodle.

## **Task: Deleting all lab resources**

### **1. Delete the API Gateway DragonsApp API**

- Open the Amazon API Gateway dashboard.
- In the navigation pane, choose APIs.
- Delete the DragonsApp API and confirm the deletion.

### **2. Delete the S3 bucket for the Dragons application.**

- Open the Amazon Simple Storage Service (Amazon S3) dashboard.
- Delete the bucket that ends with -dragons-app and confirm the deletion. You must empty the bucket before you delete it.

### **3. Delete the AWS Cloud9 development environment for this project.**

- Open the AWS Cloud9 dashboard.
- Delete the Python-DevEnv environment and confirm the deletion.