

API Driven Development

What is API Driven Development?

API Driven Development is a process that allows developers to focus on API design before writing code. The idea is, before you start writing your software, you first design and agree on what the API will be. You focus time up front putting thoughtful design into this API, and the first artifact out of the process is the API design. This API design could be captured in a document, or it could be built out as a Mock API.

Creating the API before you create the code that backs the API provides benefits to both the developers creating the backend code, as well as the clients who will be consuming the API.

Faster Application Development

Both the backend API Code and Client code can be written in parallel once the API design is created. Integrating and merging the two also becomes easier as the API is the contract upon which they both agreed. If the contract didn't change, the release of the actual backend code should go smoothly.

APIs Encourage Modular Architectures

By designing the API up front, you know what functionality should exist in the code and therefore the scope of the program is well defined. This encourages modular and distributed architectures.

Uncover API Incompatibilities Quickly

By designing and releasing the API first, you are allowing the consumers of the API to discover incompatible integrations quickly, instead of having integration be the last step for release.

Your App will be Ready to Connect to the World

Through hosting your code with an API, you can then integrate it with other systems easily.

Documentation

By following API Driven Development you have already documented the API and its features before you implement it. Tools exist in the market that allow you to visualize your API.

To read about Microservices and API Gateway on AWS click here:

<https://docs.aws.amazon.com/whitepapers/latest/microservices-on-aws/api-implementation.html> 

To read about API Management in general on AWS click here:

<https://aws.amazon.com/api-gateway/api-management/> 

Amazon API Gateway

What is Amazon API Gateway?


Amazon API Gateway is an AWS service for creating, publishing, maintaining, monitoring, and securing REST, HTTP, and WebSocket APIs at any scale. API developers can create APIs that access AWS or other web services, as well as data stored in the AWS Cloud. As an API Gateway API developer, you can create APIs for use in your own client applications. Or you can make your APIs available to third-party app developers. API Gateway creates RESTful APIs that:

Are HTTP-based.

Enable stateless client-server communication.

Implement standard HTTP methods such as GET, POST, PUT, PATCH, and DELETE.

Read more about Amazon API Gateway at:

<https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html> 

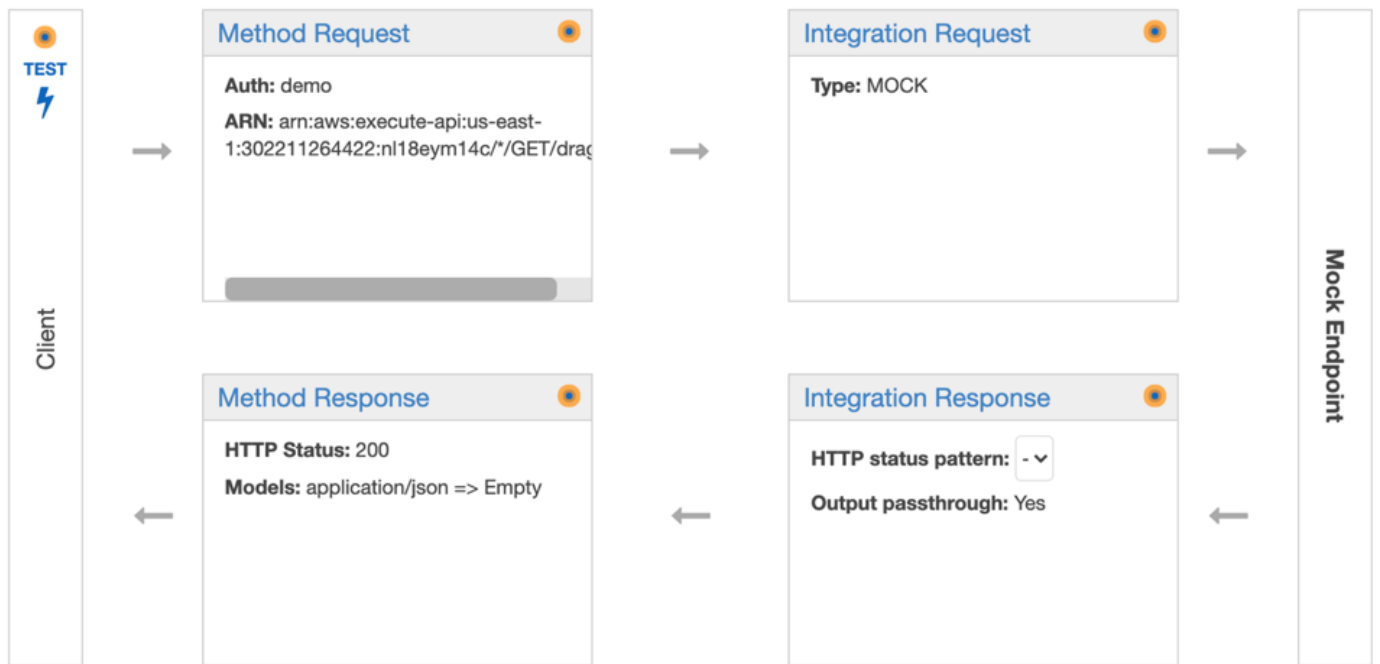
API Gateway REST APIs

API Gateway supports multiple types of APIs, in this course we will be focusing on REST APIs.

A REST API in API Gateway is a collection of resources and methods that are integrated with backend HTTP endpoints, Lambda functions, or other AWS services. You can use API Gateway features to help you with all aspects of the API lifecycle, from creation through monitoring your production APIs.

API Gateway REST APIs use a request/response model where a client sends a request to a service and the service responds back synchronously. This kind of model is suitable for many different kinds of applications that depend on synchronous communication.

API Gateway REST APIs have multiple configurable pieces:



The diagram above shows what steps a client call makes to the API to access back-end resources. A Method resource is integrated with an Integration resource. Both consist of a request and one or more responses. The method request takes the client input, and optionally validates it if configured - either against JSON Schema models or it checks the required request parameters in the URI, query string, and headers of an incoming request are included and non-blank.

If the validation fails, API Gateway immediately fails the request, returns a 400 error response to the caller, and publishes the validation results in CloudWatch Logs. This reduces unnecessary calls to the backend.

Then, the client input is passed to the back end through the integration request. The integration request is where you configure what backend resource the API will be passing the client input to. This is also where you perform any mappings or data transformations potentially using VTL Mappings.

The request then gets passed to the backend resource.

A method response returns the output from the back end to the client through an integration response. You can configure data mappings on the response from the backend at the integration request level. An integration response is an HTTP response encapsulating the backend response. You can map backend responses to specific HTTP codes.

A method request is embodied in a Method resource, whereas an integration request is embodied in an Integration resource. On the other hand, a method response is represented by a MethodResponse resource, whereas an integration response is represented by an IntegrationResponse resource.

Read more about Developing REST APIs with API Gateway at:

<https://docs.aws.amazon.com/apigateway/latest/developerguide/rest-api-develop.html>