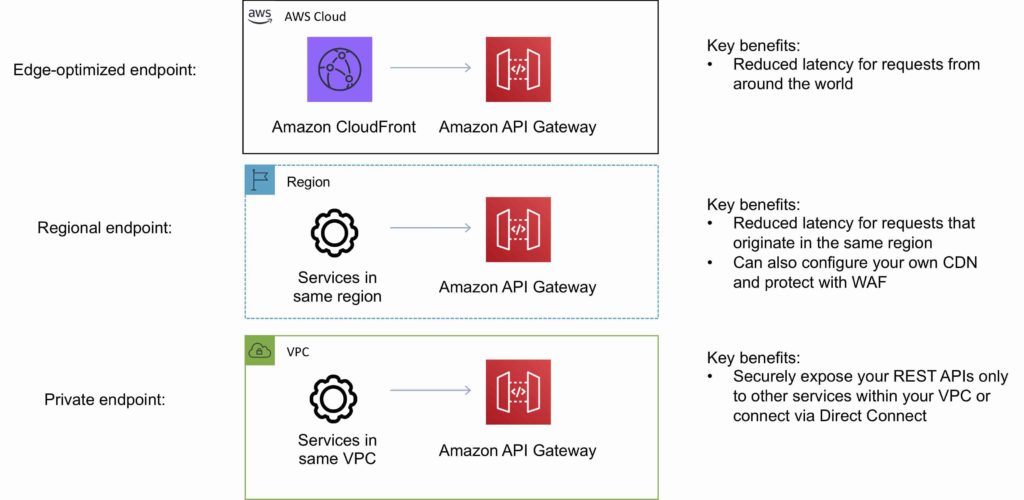
API Gateway

* API Gateway is a collection of resources and methods that are integrated with back-end HTTP endpoints, Lambda functions or other AWS services
* API GW is a fully managed service that makes it easy for developers to publish, maintain, monitor and secure APIs at any scale
* API GW provides devs with a simple flexible, fully managed, pay-as-you-go service that handles all aspects of creating and operating robust APIs for application back ends
* API GW handles all of the tasks involved in accepting and processing up to hundreds of thousands of concurrent API calls
* API calls include traffic management, authorization and access control, monitoring and API version management
* Together with Lambda, API GW forms the app-facing part of the AWS serverless infrastructure
* Back-end services include EC2, Lambda, or any web application(private or public endpoints)
* CF is used as the public endpoint for API GW
* Supports API keys and Usage Plans for user identification, throttling or quota management
* Using CF behind the scenes, custom domains, and SNI are supported
* Can be published as products and monetized on the AWS Marketplace
* Collections can be deployed in stages
* Permissions to invoke a method are granted using IAM roles and policies or API GW custom authorizers
* An API can present a certificate to be authenticated by the back-end
* All of the APIs created with API GW expose HTTPS endpoints only(does not support unencrypted endpoints)
* By default API GW assigns an internal domain that automatically uses the API GW certificates
* When configuring your APIs to run under a custom domain name you can provide your own certificate
* Supported data formats include JSON, XML, query string parameters, and request headers
* Can enable Cross Origin Resource Sharing(CORS) for multiple domain use with Javascript/AJAX
  + Can be used to enable requests from domains other than the APIs domain
  + Allows the sharing of resources between different domains
  + The method(GET, PUT, POST etc) for which you will enable CORS must be available in the API GW API before you enable CORS
  + If CORS is not enabled and an PI resource received requests from another domain the request will be locked
  + Enable CORS on the APIs resources using the selected methods under the API GW
* Data types used with API GW
  + Any payload sent over HTTP(always encrypted over HTTPS)
  + Data formats include JSON, XML, query string parameters and request headers
  + You can declare any content type for your APIs responses, and then use the transofrm templates to change the back-end response into your desired format
* You can add caching to API calls by provisioning an API GW cache and specifying its size in gigabytes
* **Endpoints**
  + An API endpoint type refers to the hostname of the API
  + The API endpoint type cannot be edge-optimized, regional, or private depending on where the majority of your API traffic originates from
  + **Edge-Optimized Endpoint:**
    - An edge-optimized API endpoint is best for geographically distributed clients. API requests are routed to the nearest CloudFront Point of Presence(POP). This is the default endpoint type for API GW REST APIs
    - Edge-optimized APIs capitalize the names of HTTP headers(for example, Cookie)
    - CF sorts HTTP cookies in natural order by cookie name before forwarding the request to your origin. For more information about the way CF processes cookies, see Caching Content Based on Cookies
    - Any custom domain name that you use for an edge-optimized API applies across all regions
  + **Regional Endpoint**
    - A regional API endpoint is intended for clients in the same region
    - When a client running on an 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 reduces connection overhead
    - For a regional API, any custom domain name that you can use is specific to the region where the API is deployed
    - If you deploy a regional API in multiple regions, it can have the same custom domain name in all regions
    - You can use custom domains together with Amazon Route 53 to perform tasks such as latency-based routing
    - Regional API endpoints pass all header names through as-is
  + **Private Endpoint:**
    - A private API endpoint is an API endpoint that can only be accessed from your Amazon Virtual Private Cloud(VPC) using an interface VPC endpoint, which is an endpoint network interface(ENI) that you create in your VPC
    - Private API endpoints pass all header names through as-is
  + 
* **Additional Features and Benefits**
  + API GW provides several features that assist with creating and managing APIS
    - **Metering –** Define plans that meter and restrict third-party developer access to APIs
    - **Security –** API GW provides multiple tools to authorize access to APIs and control service operation access
    - **Resiliency -**  Manage traffic with throttling so that backend operations can withstand traffic spikes
    - **Operations Monitoring -**  API GW provides metrics dashboard to monitor calls to services
    - **Lifecycle Management -** Operate multiple API versions and multiple stages for each version simultaneously so that existing applications can continue to call previous versions after new API versions are published
  + API GW provides robust, secure and scalable access to backend APIs and hosts multiple versions and release stages for your APIS
  + You can create and distribute API Keys to developers
  + Option to use AWS Sig-v4 to authorize access to APIs
  + You can throttle and monitor requests to protect your backend
  + API GW allows you to maintain a cache to store API responses
  + SDK Generation for iOS, Android, and JavaScript
  + Reduced latency and distributed denial of service protection through the use of CloudFront
  + Request/response data transofrmation and API mocking
  + Provides Swagger support
  + Resiliency through throttling rules base on the number of requests per second for each HTTP method(GET, PUT)
  + Throttling can be configured at multiple levels including Global and Service Call
  + A cache can be created and specified in gigabytes (not enabled by default)
  + Caches are provisioned for a specific stage of your APIs
  + Caching features include customizable keys and time-to-live in seconds for your API data which enhances response times and reduces load on back-end services
  + API GW can scale to any level of traffic received by an API
* **Logging and Monitoring**
  + The API GW logs(near real time) back-end performance metrics such as API calls, latency and error rates to CW
  + You can monitor through the API GW dashboard(REST API) allowing you to visually monitor calls to the servics
  + API GW also meters utilization by third-party developers and the data is available in the API GW console through APIs
  + API GW is integrated with CT to give a full autidatble history of the changes to your REST APIs
  + All API calls made to the API GW APIs to create, modify, delete or deploy REST APIs are logged to CT
* **Charges**
  + With API GW you only pay when your APIs are in use
  + There are no minimum fees or upfront commitments
  + You pay only for the API calls you receive and the amount of data transferred out
  + There are no data transfer out charges for Privatte APIs(however AWS PL charges apply when using Private APIs in API GW)
  + API GW also provides optional data caching charged at an hourly rate that varies based on cache size you select
  + The API GW free tier includes one million API calls per month for up to 12 months