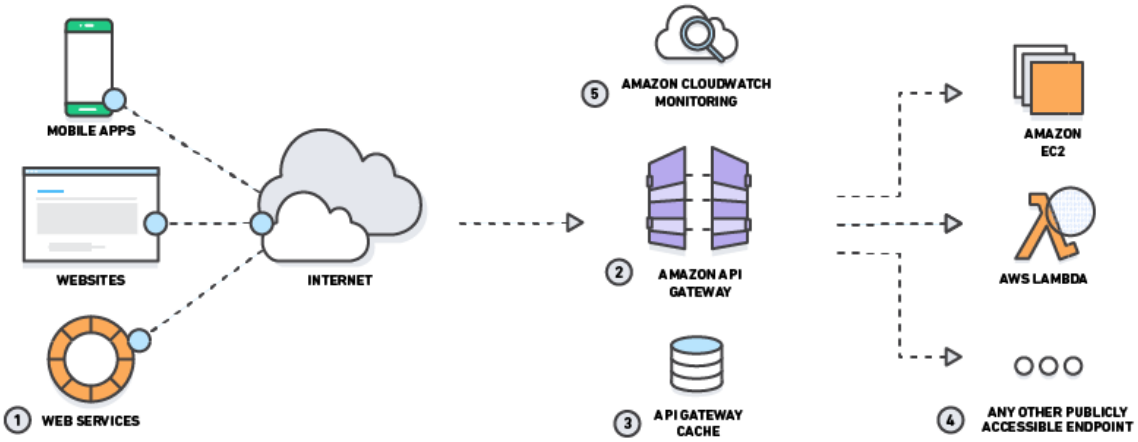
Amazon API Gateway

Amazon API Gateway is an awesome service to use as an HTTP frontend. You can use it for building serverless applications, integrating with legacy applications, or proxying HTTP requests directly to other AWS services

What is AWS API Gateway

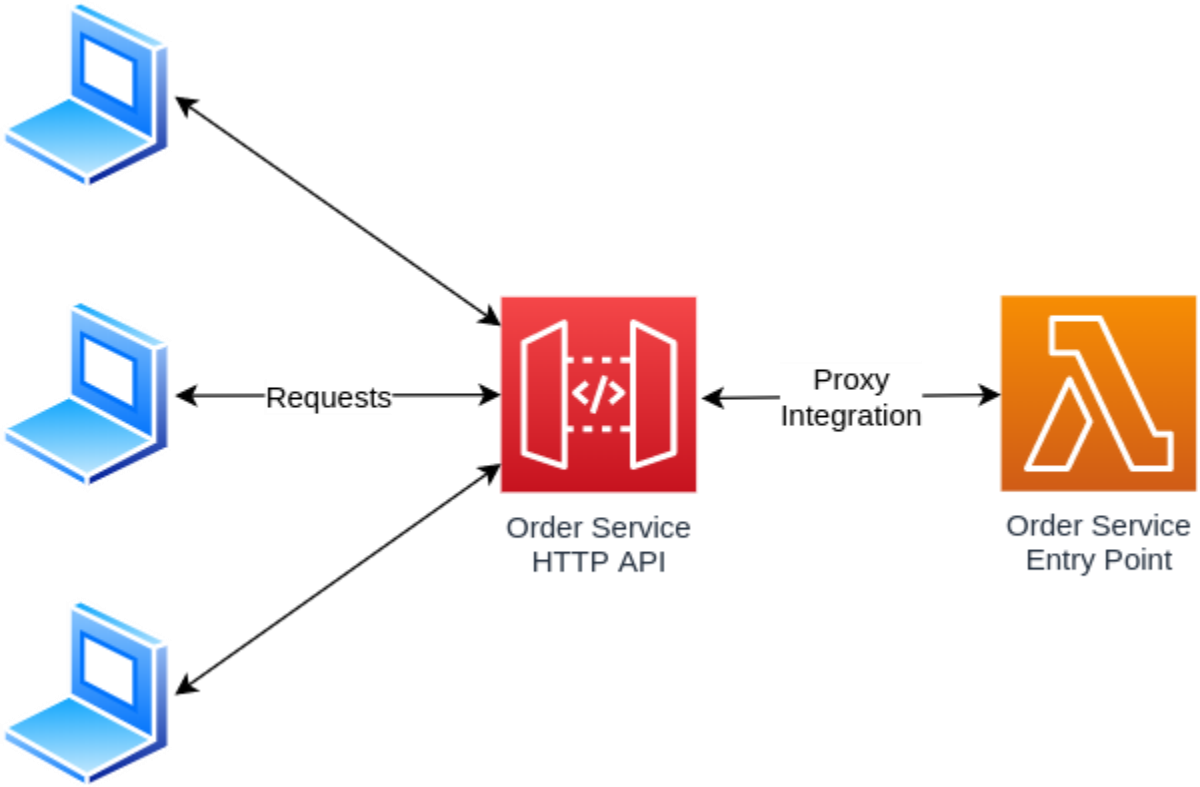
- AWS API Gateway may be a fully managed service that creates it easy for developers to publish, maintain, monitor, and secure APIs at any scale
- It handles all of the tasks involved in **accepting** and **processing** up to many thousands of concurrent **API calls**, including traffic management, authorization, access control, monitoring, and API version management.
- It has no minimum fees or startup costs and charges just for the API calls received and therefore the amount of knowledge transferred out.
- API Gateway also acts as a proxy to the configured backend operations.
- It can scale automatically to handle the quantity of traffic the API receives
- API Gateway exposes HTTPS endpoints only for all the APIs created. It does not support unencrypted (HTTP) endpoints
- APIs built on Amazon API Gateway can accept any payloads sent over HTTP with typical data formats including JSON, XML, query string parameters, and request headers.
- AWS API Gateway can communicate to multiple backends services
    - Lambda functions
    - AWS Step functions state machines
    - HTTP endpoints through Elastic Beanstalk, ELB, or EC2 servers
    - **Non-AWS hosted HTTP-based operations accessible via the public Internet**



API Gateway use cases

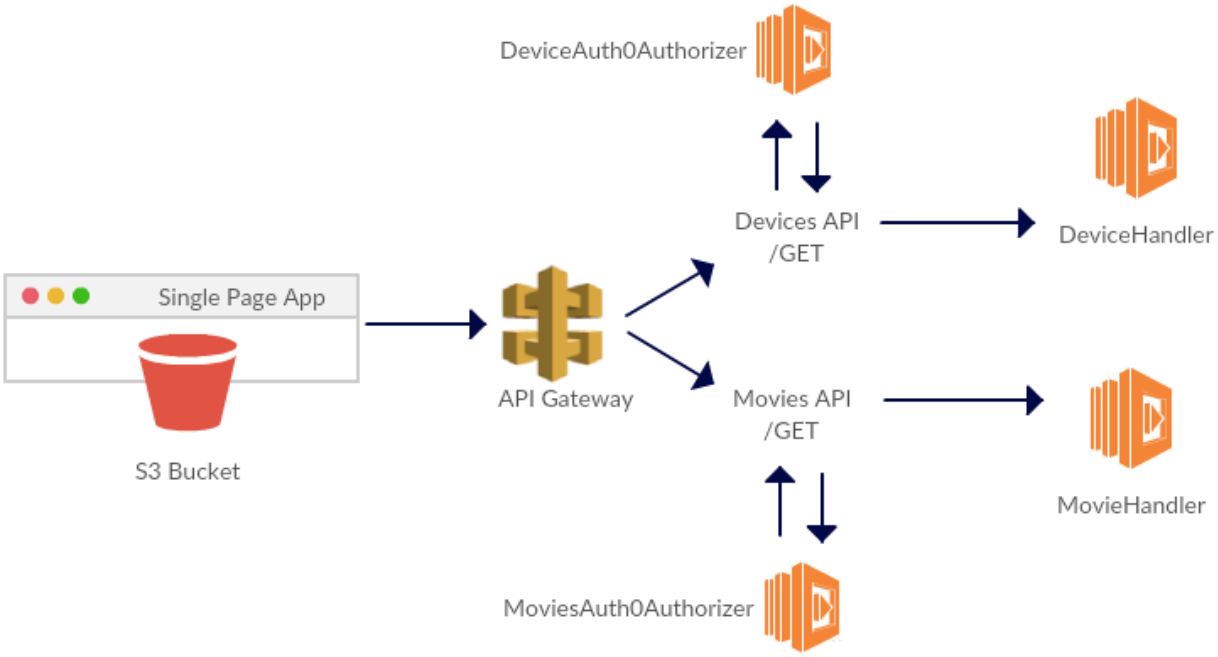
• Use API Gateway to create HTTP APIs
• Use API Gateway to create REST APIs
• Use API Gateway to create WebSocket APIs
• Who uses API Gateway?

1. **Use API Gateway to create HTTP APIs**



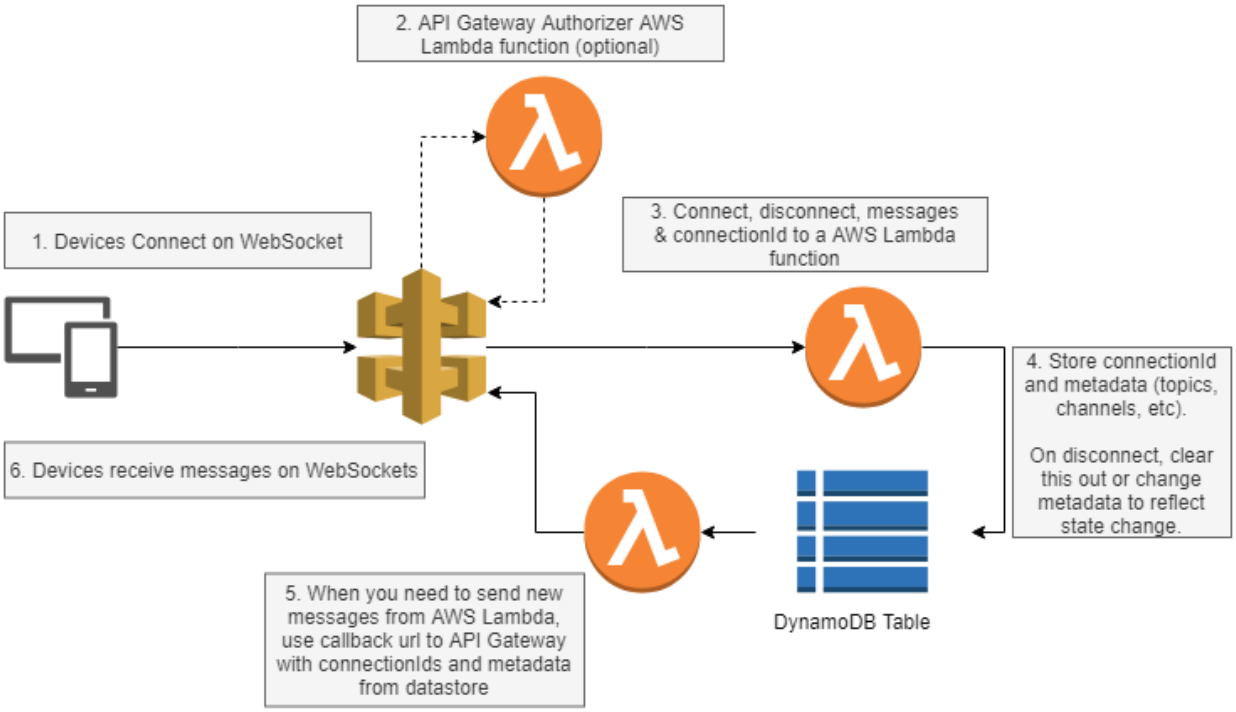
HTTP APIs help to enable you to create **RESTful** APIs with lower latency and lower cost than REST APIs. Users can use HTTP APIs to send requests to AWS Lambda functions or to any publicly routable HTTP endpoint.
For example, Users can create an HTTP API that integrates with a Lambda function on the backend. When a client calls your API, Amazon API Gateway sends the request to the Lambda function and returns the function's response to the client.

2. **Use API Gateway to create REST APIs**



An API Gateway **REST API** is made up of two things resources and methods. A resource is basically a logical entity that an app can access through a resource path. A method corresponds to a REST API request that is submitted by the user or client of your API and the response is returned to the user.
For example, /incomes could be the path of a resource representing the total income of the app user. A resource can have one or more operations that are defined by appropriate HTTP verbs such as **GET, POST, PUT, PATCH**, and **DELETE**. The combination of a resource path and an operation identifies a method of the API. Consider an example, a POST /incomes method could add an income earned by the caller, and a GET/expenses method could query the reported expenses incurred by the caller.

3. **Use API Gateway to create WebSocket APIs**



In WebSocket API, the client/user and the server can both send communicate with each other at any time. Backend servers can easily push data to connected users and devices, avoiding the need to implement complex polling mechanisms that are difficult to manage and make. **For example**, you could build a serverless application using an API Gateway WebSocket API and AWS Lambda to send and receive messages to and from individual clients or groups of users in a chat room. Or the user could invoke backend services such as AWS Lambda, Amazon Kinesis, or an HTTP endpoint based on message content.

4. **Who uses API Gateway?**

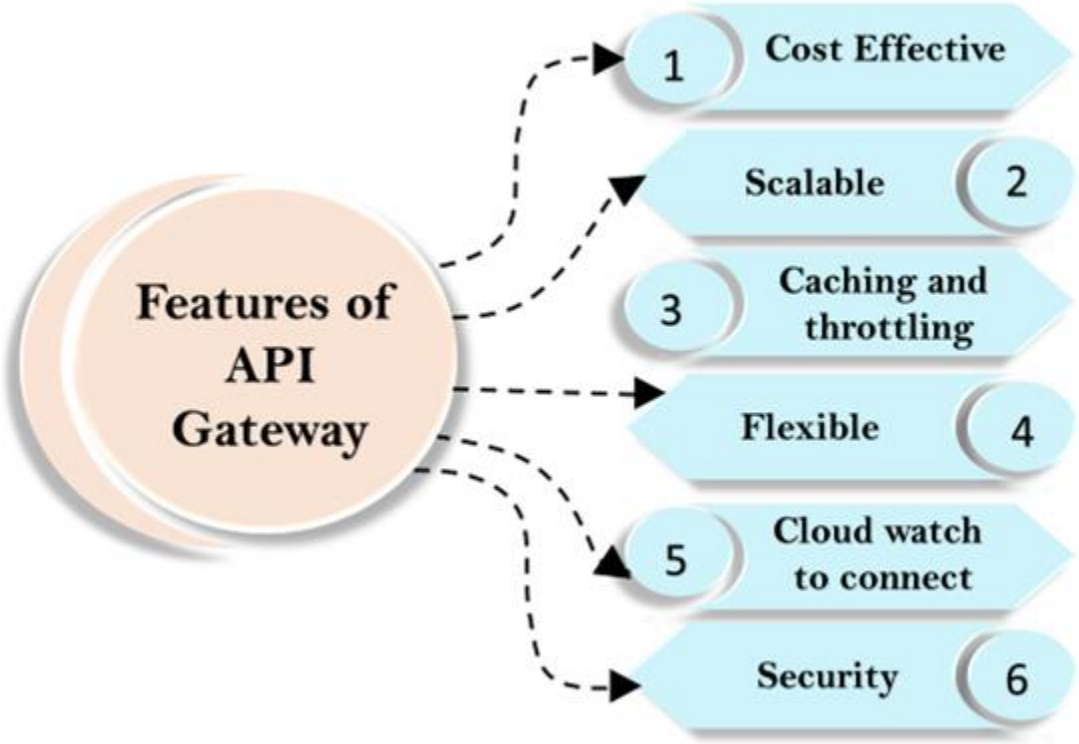
There are two types of developers who use API Gateway: **API developers** and **app developers**.

An API developer creates and deploys an API to enable the required functionality in Amazon API Gateway. The API developer must be an IAM user in the AWS account that owns the API or the root account, user.

An app developer builds a functioning application to call AWS services by invoking a WebSocket or REST API created by the developer in API Gateway.

The app developer is the customer of the API developer. The app developer doesn't need to have an AWS account, provided that the API either doesn't require IAM permissions or supports the authorization of users through third-party federated identity providers like Amazon Cognito user pool identity federation. Such identity providers include Amazon Cognito user pools, Facebook, and Google.

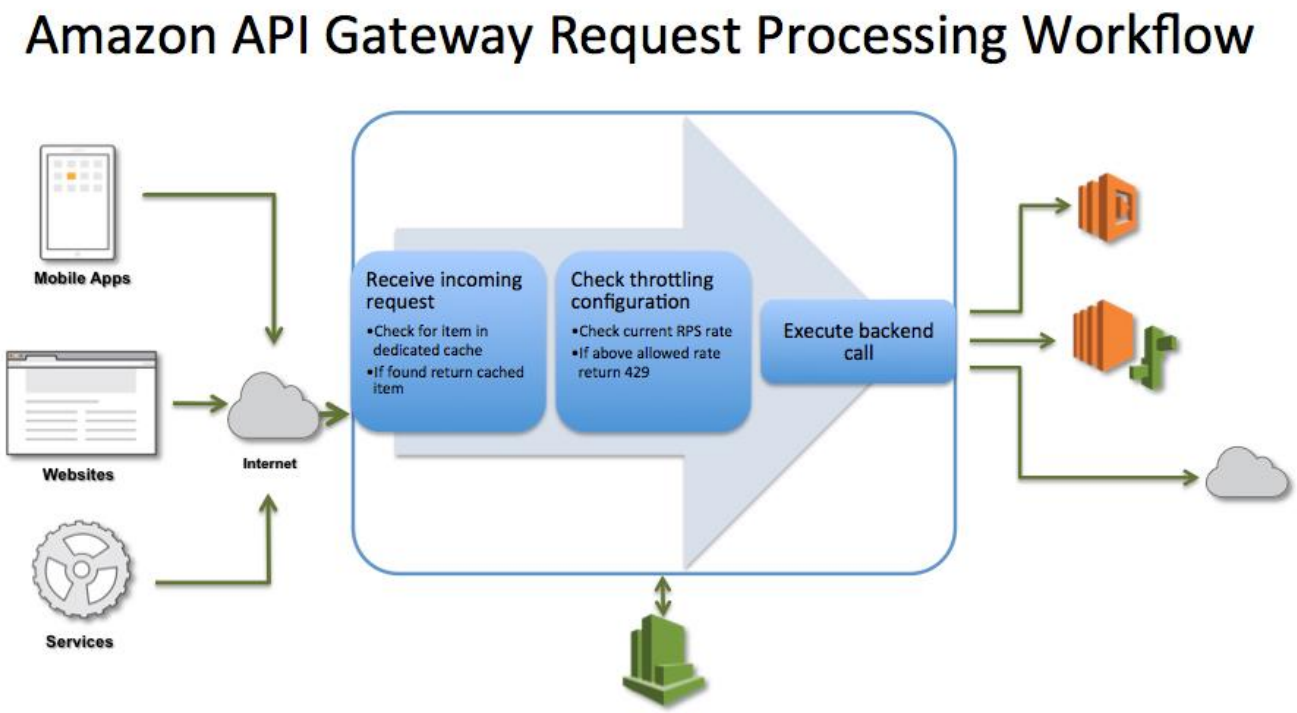
Features of API Gateway



- **Cost-Effective**
  It is very low cost and efficient as an API Gateway provides a tiered pricing model for API requests. The price of an Amazon API request is as low as $1.51 per million requests, you can also decrease the costs by decreasing the number of requests.
- **Scalable**
  Users do not have to worry about having EC2 service or Autoscaling groups responding to API requests. An API Gateway scales automatically.
- **Caching and Throttling**
  Caching is the most important feature of Amazon API Gateway. Caching is used to cache the endpoint's responses which improve the latency of requests to your API. It is also a primary factor that determines the price of the service.
  You can also prevent security risks to your API Gateway. If you want to prevent flooding with the fraud API calls to API Gateway, you can configure a throttled service that can throttle requests to prevent attacks.
- **Flexible**
  To execute the API Gateway, you do not have to launch an EC2 instance or set up the Gateway software. API Gateway can be implemented in a few minutes through the AWS Management Console.
- **CloudWatch to Connect**
  An Amazon API Gateway is integrated with the AWS CloudWatch service which is a monitoring service. This tool is used to monitor the metrics of incoming API calls, latency, and errors.
- **Security**
  You can authorize access to your APIs. API Gateway is used to verify incoming requests by executing various authorization options such as Lambda function and Identity Access Management service (IAM). An IAM is integrated with a gateway that provides tools such as

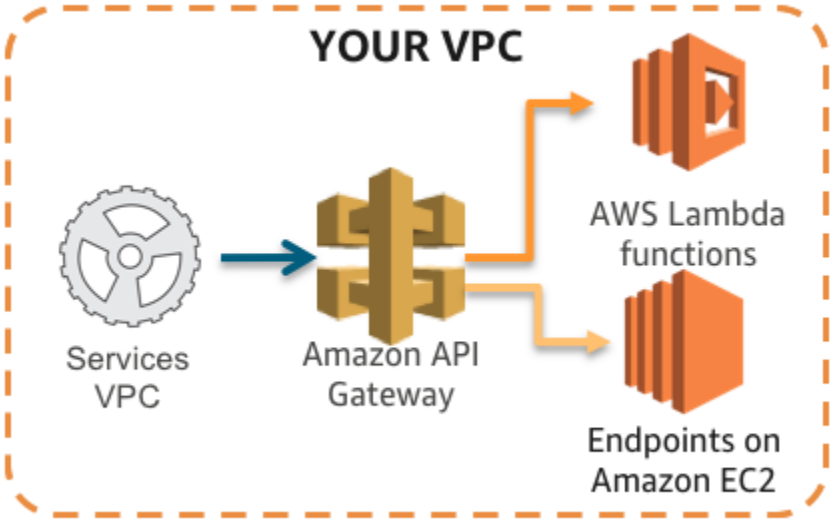
AWS credentials, i.e., access and secret keys to access an API. A Lambda function is used to verify tokens, and if tokens are successfully verified, then access to an API will be granted.

API Gateway Throttling and Caching



- **Throttling**
  API Gateway provides throttling at multiple levels including global and by-service calls and limits can be set for standard rates and bursts. It tracks the number of requests per second. Any requests over the limit will receive a 429 HTTP response. It ensures that API traffic is controlled to help the backend services maintain performance and availability.
- **Caching**
  API Gateway provides API result caching by provisioning an API Gateway cache and specifying its size in gigabytes. It helps improve performance and reduces the traffic sent to the back end.
    - API Gateway handles the request in the following manner
        - Consider an example, If caching is not enabled and throttling limits have not been applied, then all requests pass through to the backend service until the account-level throttling limits are reached.
        - If throttling limits are specified, the API Gateway will shed the necessary amount of requests and send only the defined limit to the back-end
        - If a cache is configured, the API Gateway will return a cached response for duplicate requests for a customizable time, but only if under configured throttling limits

API Endpoint Types



- **Edge-optimized API endpoint:**
  The hostname of an API Gateway API that is deployed to the specified region while using an AWS CloudFront distribution to facilitate client access typically from across AWS regions. API requests are routed to the nearest CloudFront Point of Presence.
- **Regional API endpoint:**
  The hostname of an API that is deployed to the specified region and intended to serve clients, such as EC2 instances, in the same AWS region. API requests are targeted directly to the region-specific API Gateway without going through any CloudFront distribution.
  Users can apply latency-based routing on regional endpoints to deploy an API to multiple regions using the same regional API endpoint configuration, set the same custom domain name for each deployed API, and configure latency-based DNS records in Route 53 to route client requests to the region that has the lowest latency.
- **Private API endpoint**:
  It allows a user to securely access private API resources inside a VPC. Private APIs are isolated from the public Internet, and they can only be accessed using VPC endpoints for API Gateway that have been granted access.

Pricing

Amazon API Gateway supports the following flavors of the API Gateway:

- REST APIs
- HTTP APIs
- Websocket APIs

Amazon API Gateway Free Tier

API Gateway provides a free tier that is only available to new customers and is available for 12 months after the initial sign-up date. After this time period, users pay the regular rate for the services.

The free tier benefits are as follows:

- **REST API**: **1 million** API calls received
- **HTTP API**: **1 million** API calls received
- **Websocket API**: **1 million** messages and **750,000** connection minutes

Now, we will look at the pricing for the different types of API Gateway services:

HTTP APIs

HTTP APIs are charged based on the basis of the number of API calls made. For e.g. below mentioned is the pricing in US East (N. Virginia)

Number of requests(per month)	Price(per million)
First 300 million	$1.00
300+ million	$0.90

Important: API Gateway billing is always pro-rated. If you make less than 300 million API requests, you will be charged exactly for the number of API requests made.
Also Read: Our blog post on **Amazon DynamoDB**. Click here

REST APIs

REST APIs are also charged based on the number of API calls made. For e.g. the below mentioned is the current pricing in US East (N.Virginia):

Number of requests(per month)	Price(per million)
First 333 million	$3.50
Next 667 million	$2.80

Websocket APIs

Websocket APIs are charged based on the number of messages sent and received by users as well as the total number of connection minutes. Messages are metered in 32 KB increments.

The below-mentioned shows the current pricing for messages in US East (N.Virginia):

Number of requests(per month)	Price(per million)
First 1 billion	$1.00
Over 1 million	$0.80

Connection Minutes are charged at **$0.25** per million connection minutes

Additional Charges

REST APIs provide the ability to provision a dedicated cache to improve performance. The pricing for the Cache is based on the amount of memory provisioned by the user.

Cache Memory Size (GB)	Price per Hour
0.5	$0.02
1.6	$0.038
6.1	$0.20
13.5	$0.25
28.4	$0.50
58.2	$1.00
118.0	$1.90
237.0	$3.80

There can be other additional charges depending on how API Gateway is being used such as:

- Data transfer
- Lambda
- Cloudwatch