# AWS API Gateway

***CS516 – Cloud Computing***

***Computer Science Department***

***Maharishi International University***

# Maharishi International University - Fairfield, Iowa

# Content

- Amazon API Gateway benefits and how it works
- REST, HTTP, WebSocket APIs
- CORS
- Custom domain
- Caching
- Throttling
- Quotas

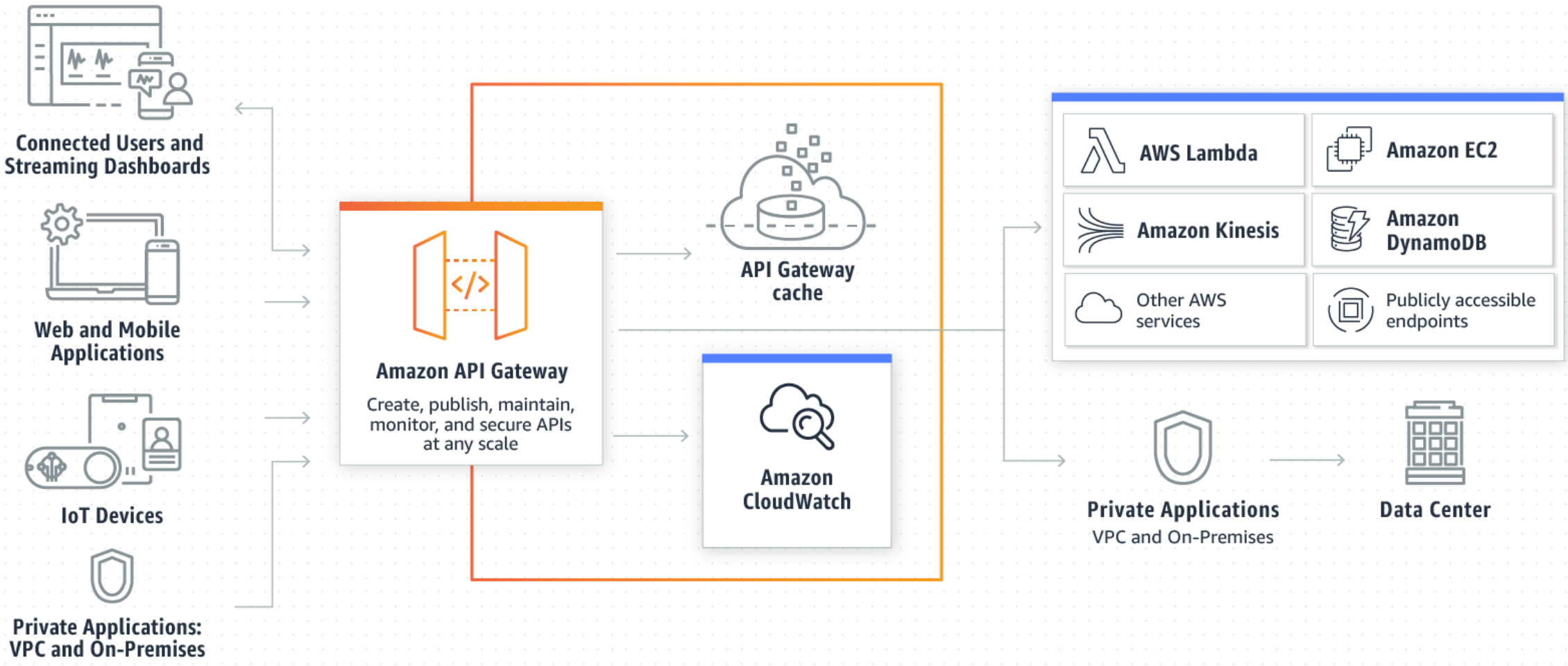
# Amazon API Gateway

Amazon API Gateway is an AWS service for creating, publishing, maintaining, monitoring, and securing APIs at any scale.

There are 3 types of APIs:

- REST
- HTTP
- WebSocket

# How it works


Connected Users and Streaming Dashboards
Web and Mobile Applications
IoT Devices
Private Applications: VPC and On-Premises
Amazon API Gateway
Create, publish, maintain, monitor, and secure APIs at any scale
API Gateway cache
Amazon CloudWatch
AWS Lambda
Amazon EC2
Amazon Kinesis
Amazon DynamoDB
Other AWS services
Publicly accessible endpoints
Private Applications
VPC and On-Premises
Data Center

# Amazon API Gateway benefits

**Resiliency** – It manages traffic to your backend systems by allowing you to set throttling rules. It handles all traffics so you can focus on business code rather than maintaining infrastructure. You can also setup cache in front of the API.

**Easy API Creation and Deployment** – It can execute AWS Lambda code in your account, start AWS Step Functions, or make calls to EC2, or other web services outside of AWS with publicly accessible HTTP endpoints.

**API Operations Monitoring** – It provides a dashboard to visually monitor calls to the services such as API calls, latency, and error rates through CloudWatch.

# Amazon API Gateway benefits

**AWS Authorization** – It helps you leverage signature version 4 authentication. You can use IAM, Lambda, JWT token.

**API Keys for Third-Party Developers** – It helps you manage the ecosystem of third-party developers accessing your APIs. You can create API keys. You can also define plans that set throttling and request quota limits for each individual API key.

**SDK Generation** – It can generate client SDKs for a number of platforms which you can use to quickly test new APIs from your applications and distribute SDKs to third-party developers.

**API Lifecycle Management** – API Gateway lets you run multiple versions of the same API simultaneously so that applications can continue to call previous API versions even after the latest versions are published.

# Rest API

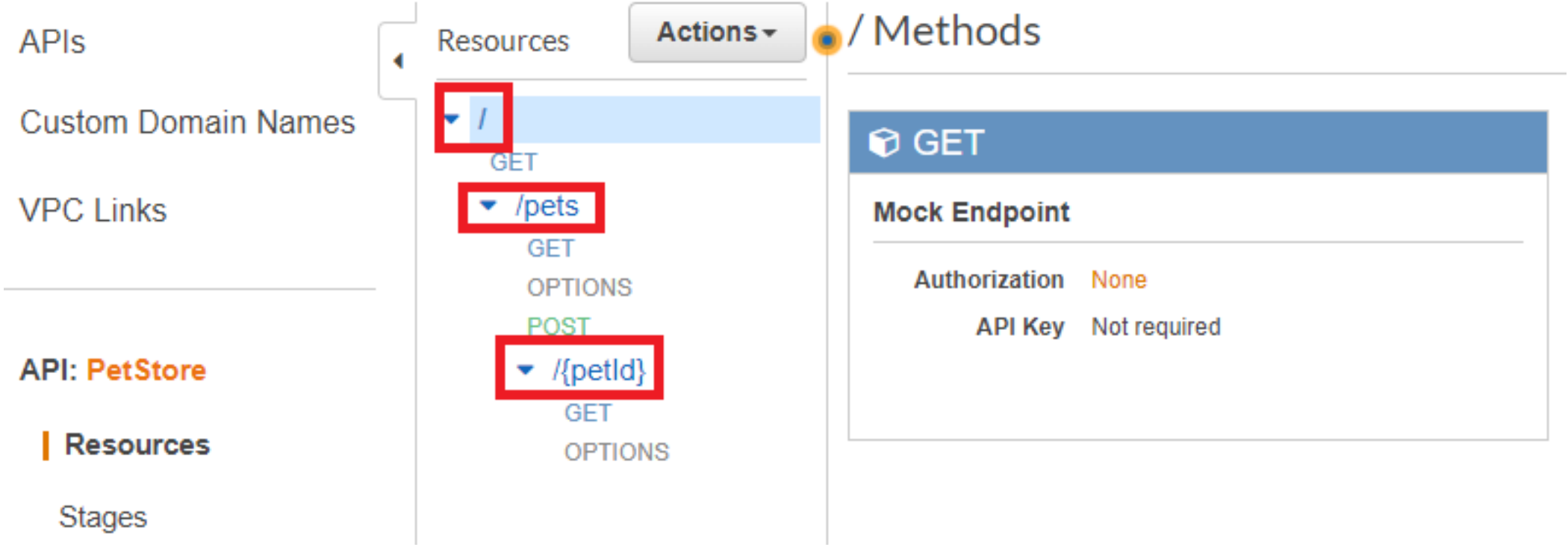
Representational state transfer (REST) is a software architectural style which uses a subset of HTTP.

A Web service that follows these guidelines is called RESTful. Such a Web service must provide its **web resources** in a textual representation and allow them to be read and modified with a **stateless** protocol and a **predefined set of operations**.

There is a contract between the service provider and consumer.

# Rest API

Resources are in red. Each resource can have multiple http methods.



# HTTP API

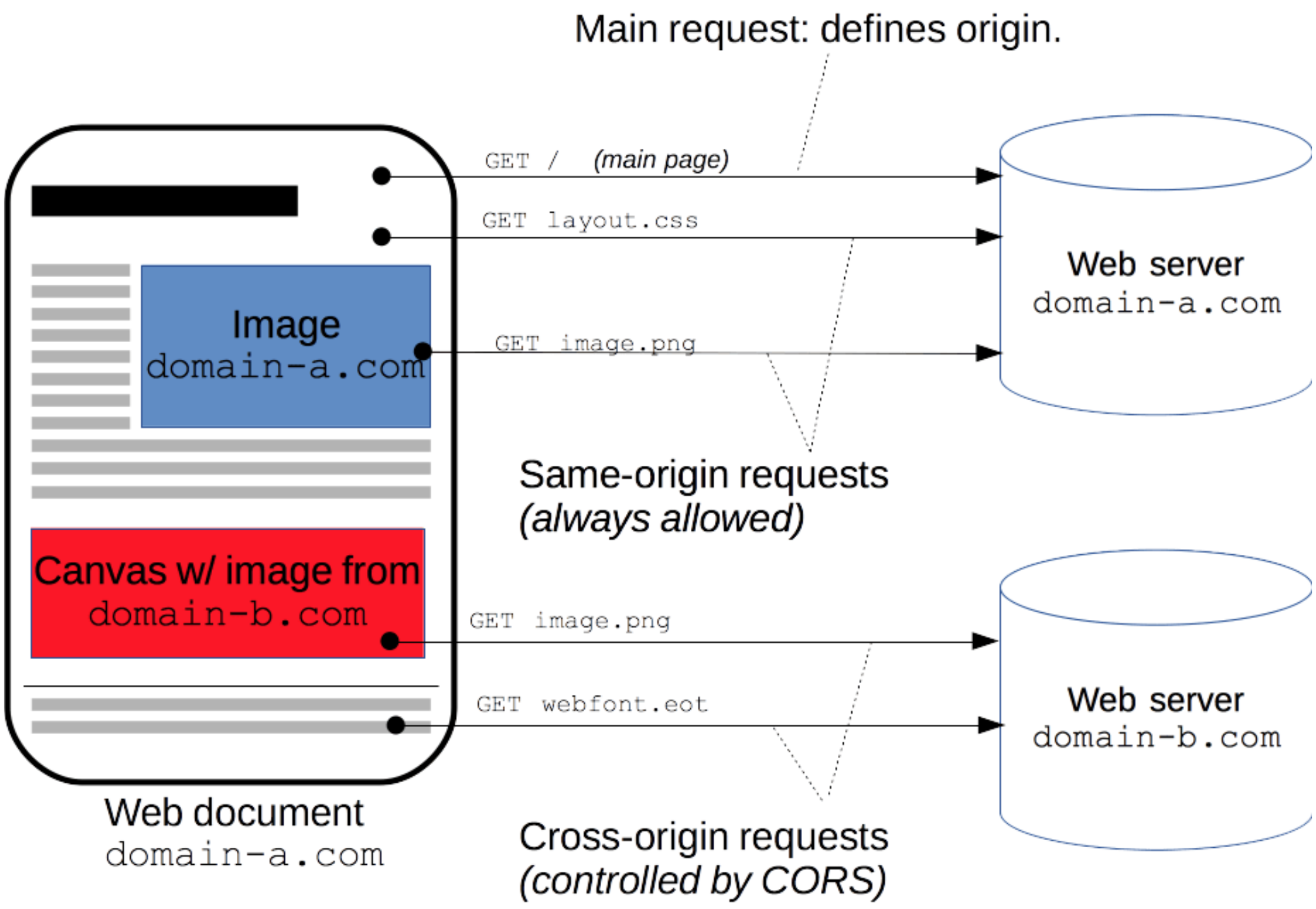
Build low-latency and cost-effective REST APIs with built-in features such as OIDC and OAuth2, and native CORS support.

For more info: Choosing between HTTP APIs and REST APIs

# What is CORS?

CORS (Cross-Origin Resource Sharing) is a security mechanism on web browsers that allows other origins (domains) load its resources. For example, if you are accessing from your *localhost* to the API behind AWS API gateway, it won't allow the request unless you enable CORS.

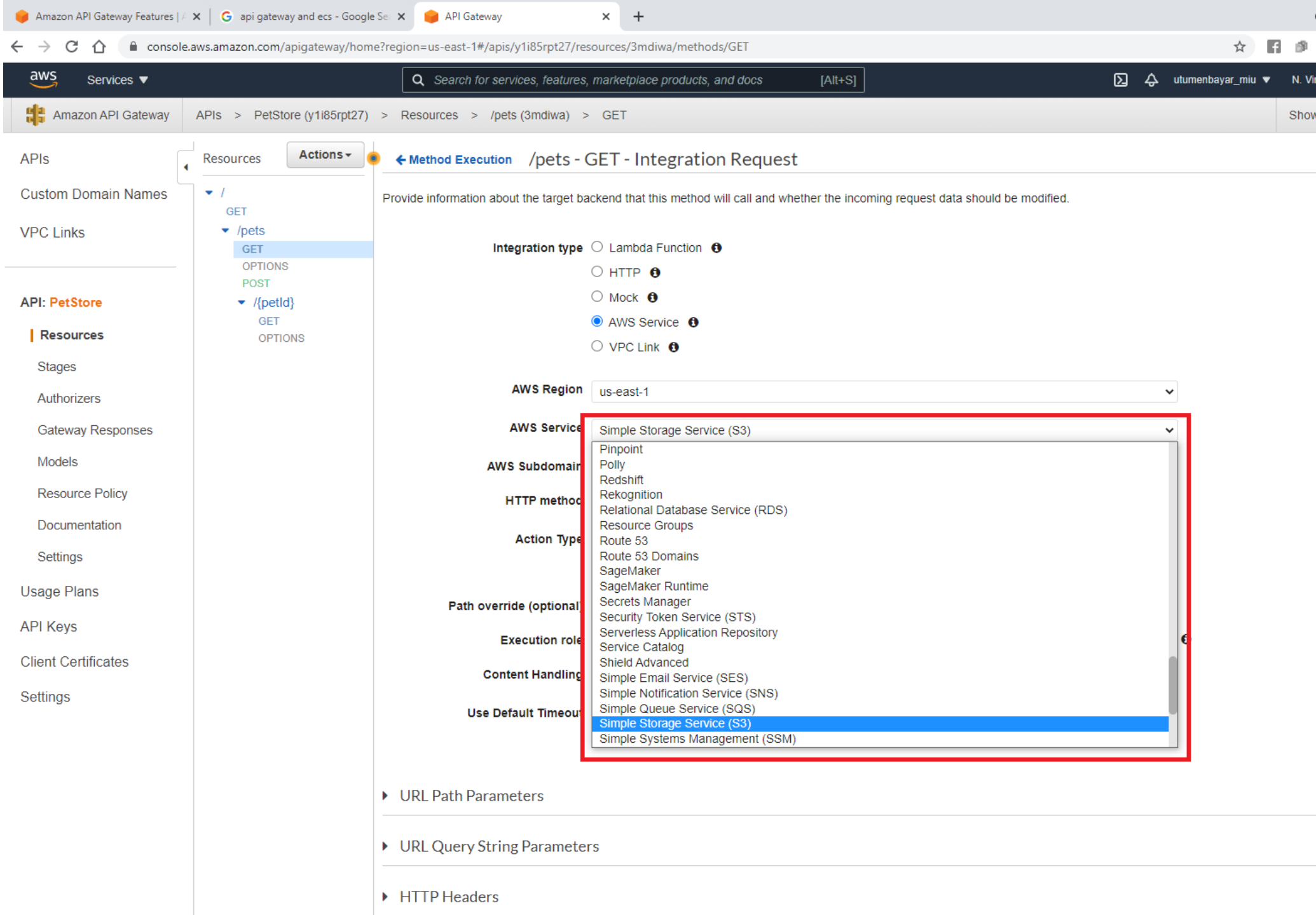
Browsers make an implicit call with OPTIONS method before sending the actual request. If the web server (in this case AWS API Gateway) enabled CORS, the request will be sent. Otherwise, it throws CORS error.


Main request: defines origin.
GET / (main page)
GET layout.css
GET image.png
Image
domain-a.com
Web server
domain-a.com
Same-origin requests
(always allowed)
Canvas w/ image from
domain-b.com
GET image.png
GET webfont.eot
Web server
domain-b.com
Web document
domain-a.com
Cross-origin requests
(controlled by CORS)

# WebSocket API

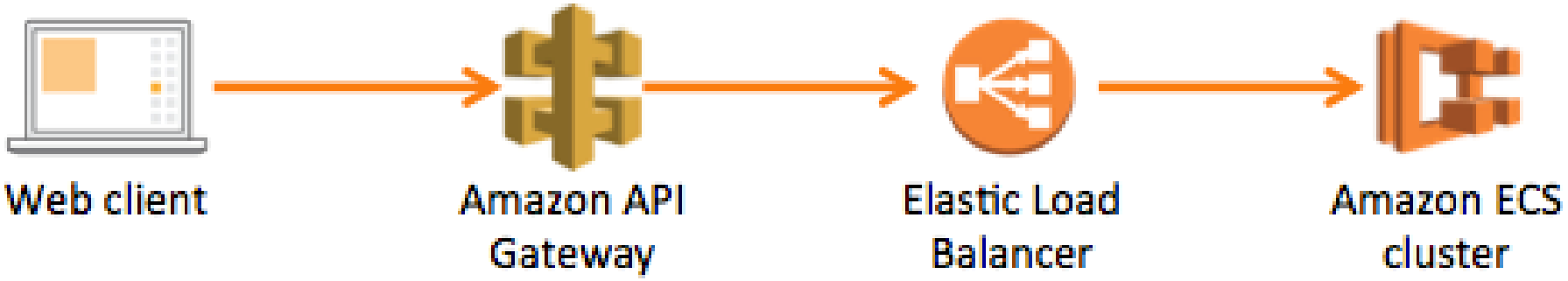
The WebSocket API is an advanced technology that makes it possible to open a **two-way** interactive communication **session** (persistent connections) between the user's browser and a server for real-time use cases such as chat applications or dashboards.

You can run other AWS services behind API Gateway



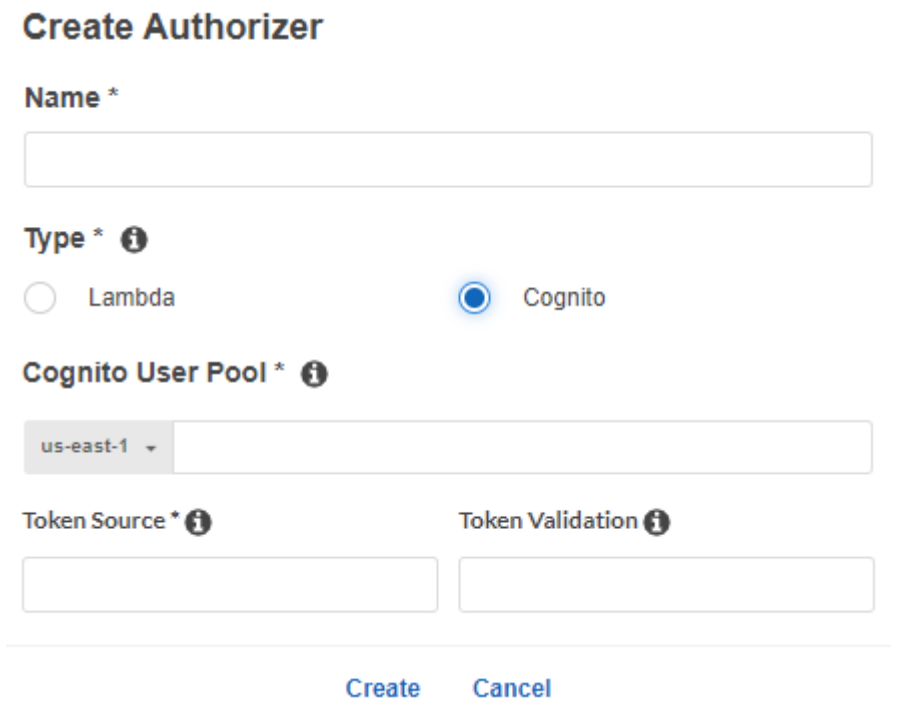
# HTTP proxy mode

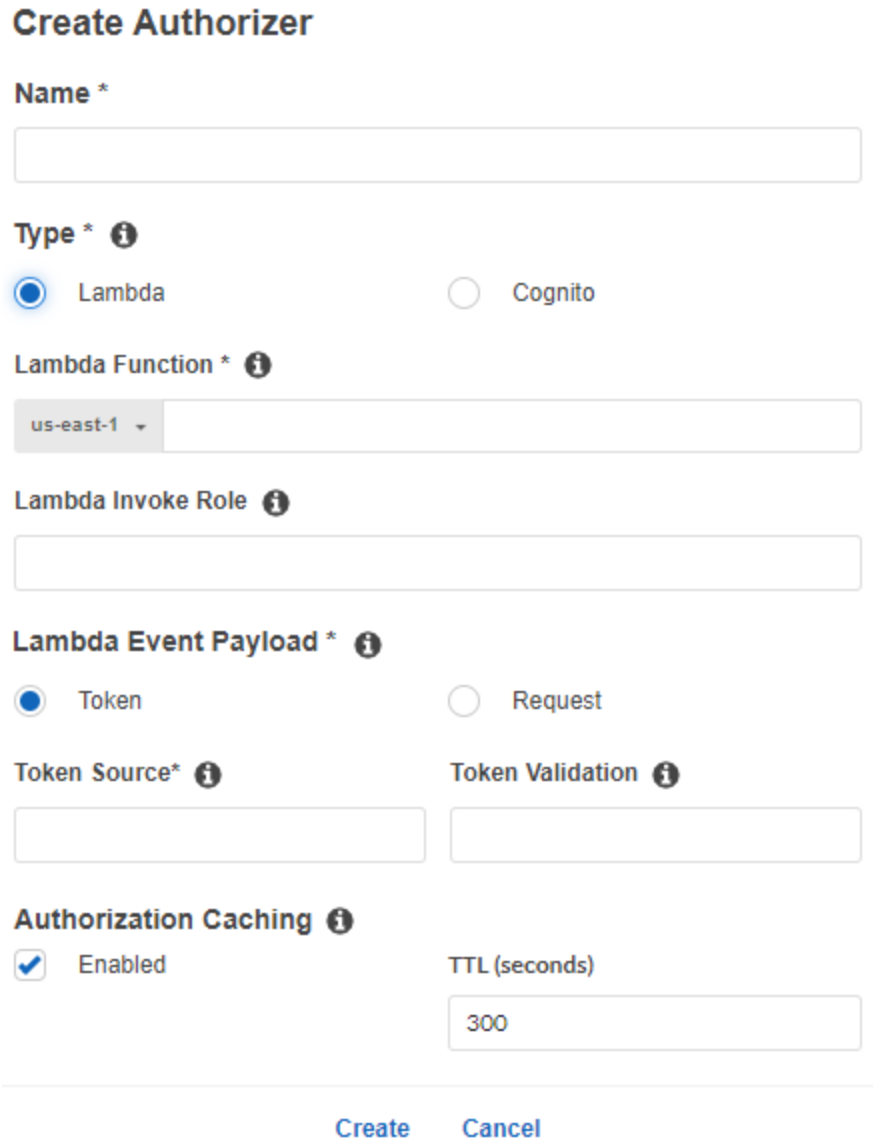
Amazon API Gateway can make proxy calls to any **publicly accessible** endpoint; for example, an Elastic Load Balancer endpoint in front of a microservice that is deployed on Amazon EC2 or ECS.



# Authorization

Authorizers enable you to control access to your APIs using Amazon Cognito User Pools or a Lambda function



# Custom domain

Custom domain names are simpler and more intuitive URLs that you can provide to your API users.

Default: https://api-id.execute-api.region.amazonaws.com/stage

Custom: https://api.example.com/myservice or https://myservice.api.example.com/

After a custom domain name is created in API Gateway, you must create a record in AWS Route53 or your DNS provider.

Certificate you can issue with Amazon Certificate Manager (ACM).

# Caching

You can enable API caching in Amazon API Gateway to cache your endpoint's responses. With caching, you can reduce the number of calls made to your endpoint and also improve the latency of requests to your API.

When you enable caching, it caches responses from your endpoint for a specified time-to-live (TTL) period, in seconds.

- By default, TTL is 300 seconds (5 minutes)
- The maximum TTL is 3600 seconds (1 hour)
- The maximum size of a response that can be cached is 1 MB

# Protect

To prevent your API from being overwhelmed by too many requests, Amazon API Gateway throttles requests to your API using the token bucket algorithm.

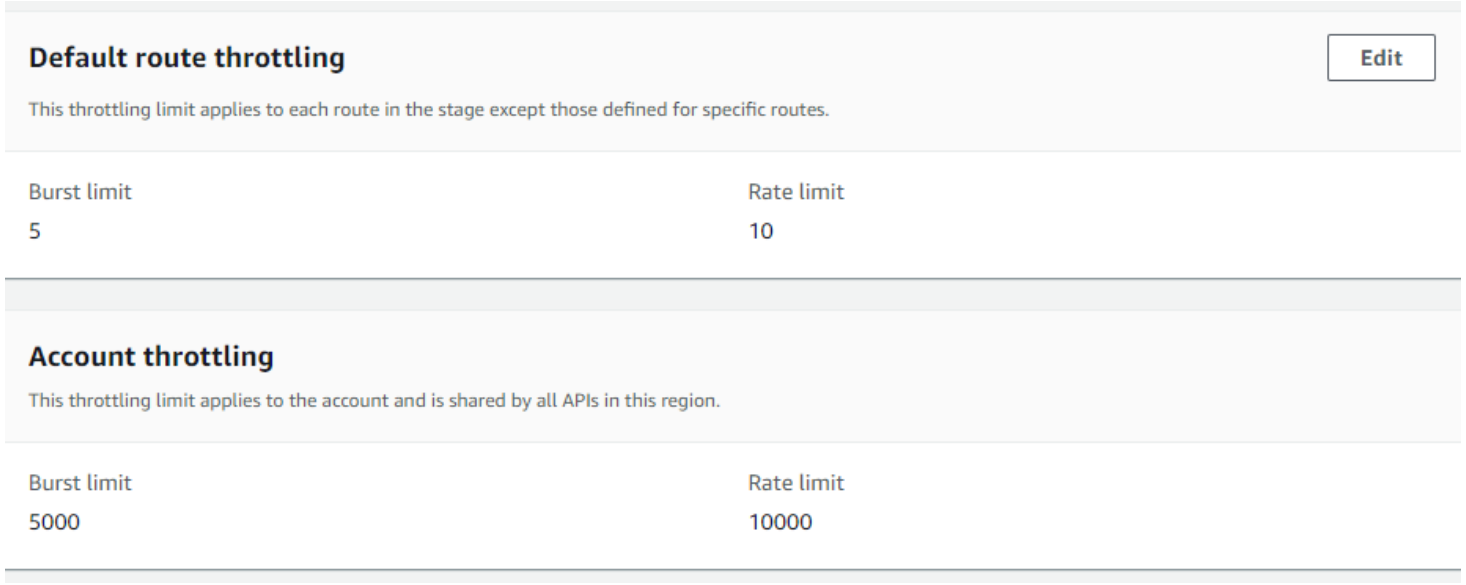
It also limits the burst (the maximum number of concurrent requests) across all APIs within an AWS account, per Region. If it reaches the limit, clients get *429 Too Many Requests*.

# Throttling

Route-level throttling limits for a specific stage or for individual routes in your API. It doesn’t exceed account limit.

The rate limit is per second. If a client submits 10 requests **(rate limit) evenly in one second**, it will work fine.

But if it submits all of them **in the first millisecond**, it takes the first 5 requests **(burst limit)**, and the rest gets 429 Too Many requests.



# Quotas

| Resource or operation | Default quota | Can be increased |
|---|---|---|
| Routes per API | 300 | Yes |
| Integrations per API | 300 | No |
| Maximum integration timeout | 30 seconds | No |
| Stages per API | 10 | Yes |
| Tags per stage | 50 | No |
| Total combined size of request line and header values | 10240 bytes | No |
| Payload size | 10 MB | No |
| Custom domains per account per Region | 120 | Yes |
| Access log template size | 3 KB | No |
| Amazon CloudWatch Logs log entry | 1 MB | No |
| Authorizers per API | 10 | Yes |
| Audiences per authorizer | 50 | No |
| Scopes per route | 10 | No |
| Timeout for JSON Web Key Set endpoint | 1500 ms | No |
| Response size from JSON Web Key Set endpoint | 150000 bytes | No |
| Timeout for OpenID Connect discovery endpoint | 1500 ms | No |
| VPC links per account per Region | 10 | Yes |
| Subnets per VPC link | 10 | Yes |
| Stage variables per stage | 100 | No |