
AWS API Gateway

2024-03

Renato Matos
renato.matos79@gmail.com

A Representational State Transfer (REST) API is a **type of web service architecture that allows different systems to communicate with each other over the internet** broken down into key components:

Resources: In a REST API, resources represent the data entities or objects that clients can interact with. Each resource is identified by a unique Uniform Resource Identifier (URI), commonly known as an endpoint. For example, `/users` might represent a collection of user resources, and `/users/{id}` might represent a specific user identified by their ID.

HTTP Methods: RESTful APIs use standard HTTP methods to perform operations on resources. The most commonly used HTTP methods in REST APIs are:

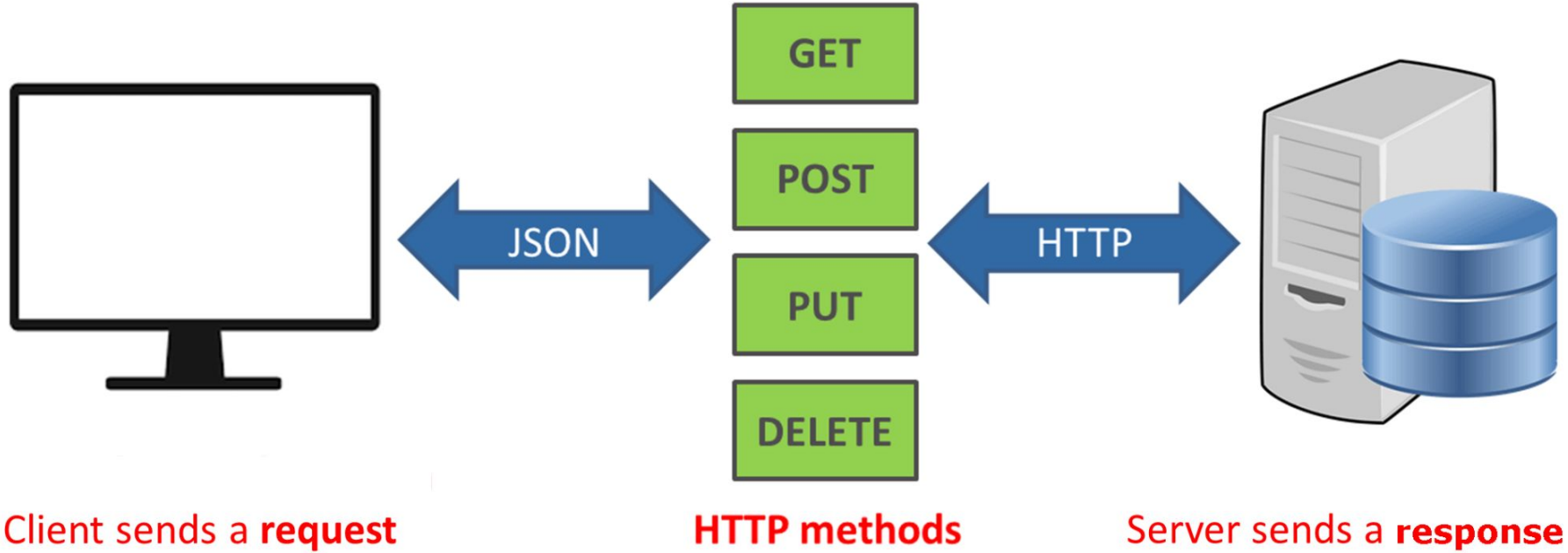
- GET: Retrieve a representation of a resource.
- POST: Create a new resource.
- PUT: Update an existing resource.
- DELETE: Delete a resource.
- PATCH: Partially update a resource.

These HTTP methods map to CRUD (Create, Read, Update, Delete) operations on resources.

Statelessness: REST APIs are stateless, meaning that each request from a client to the server must contain all the information necessary to understand and process the request. The server does not maintain any client state between requests.

Representation: Resources in a REST API are typically represented using different media types such as JSON (JavaScript Object Notation) or XML (eXtensible Markup Language). Clients and servers communicate by exchanging representations of resources, which contain data in a structured format.

Rest API



API Gateway



Build and Deploy APIs Quickly: API Gateway allows you to quickly create RESTful APIs or WebSocket APIs without the need for infrastructure setup or management

Scalability and High Availability: API Gateway automatically scales to handle any amount of traffic, ensuring high availability and reliability of your APIs

Integration with AWS Services: AWS Lambda, AWS S3, AWS DynamoDB, AWS Cognito, and more

Security and Access Control: API Gateway provides features for securing your APIs, including authentication, authorization, and encryption. You can use AWS Identity and Access Management (IAM), OAuth 2.0, and Amazon Cognito to control access to your APIs and protect sensitive data

Monitoring and Logging: API Gateway offers built-in monitoring and logging capabilities that allow you to track API usage, performance metrics, and errors in real-time

Cost-effective Pricing Model: API Gateway offers a pay-as-you-go pricing model, where you only pay for the API requests and data transfer out of the service.

Support for API Versioning and Lifecycle Management: API Gateway supports API versioning and lifecycle management, allowing you to easily manage multiple versions of your APIs, roll back changes, and deprecate outdated versions without disrupting existing clients.

Developer-Friendly Features: API Gateway provides features such as API documentation, SDK generation, and testing tools that make it easier for developers to consume and integrate with your APIs.

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

HTTP API:

- HTTP API is a new type of API Gateway introduced by AWS, designed to offer low-latency, low-cost API integrations for HTTP-based APIs.
- It provides a lightweight and cost-effective solution for building APIs with simple use cases.
- HTTP APIs support core API Gateway features such as request/response transformation, CORS (Cross-Origin Resource Sharing), JWT (JSON Web Tokens) authorizers, and rate limiting.
- HTTP APIs do not support all features available in REST APIs, but they are optimized for serverless architectures and simple API use cases.

REST API:

- REST API is the traditional API Gateway offered by AWS, which supports a broader range of features and customization options compared to HTTP APIs.
- It follows the principles of Representational State Transfer (REST) architecture, allowing clients to interact with resources using standard HTTP methods and representations.
- REST APIs support more advanced features such as custom domain names, API keys, IAM (Identity and Access Management) authorization, request/response validation, and stage variables.
- REST APIs are suitable for building complex APIs with multiple endpoints, advanced security requirements, and integration with various AWS services.

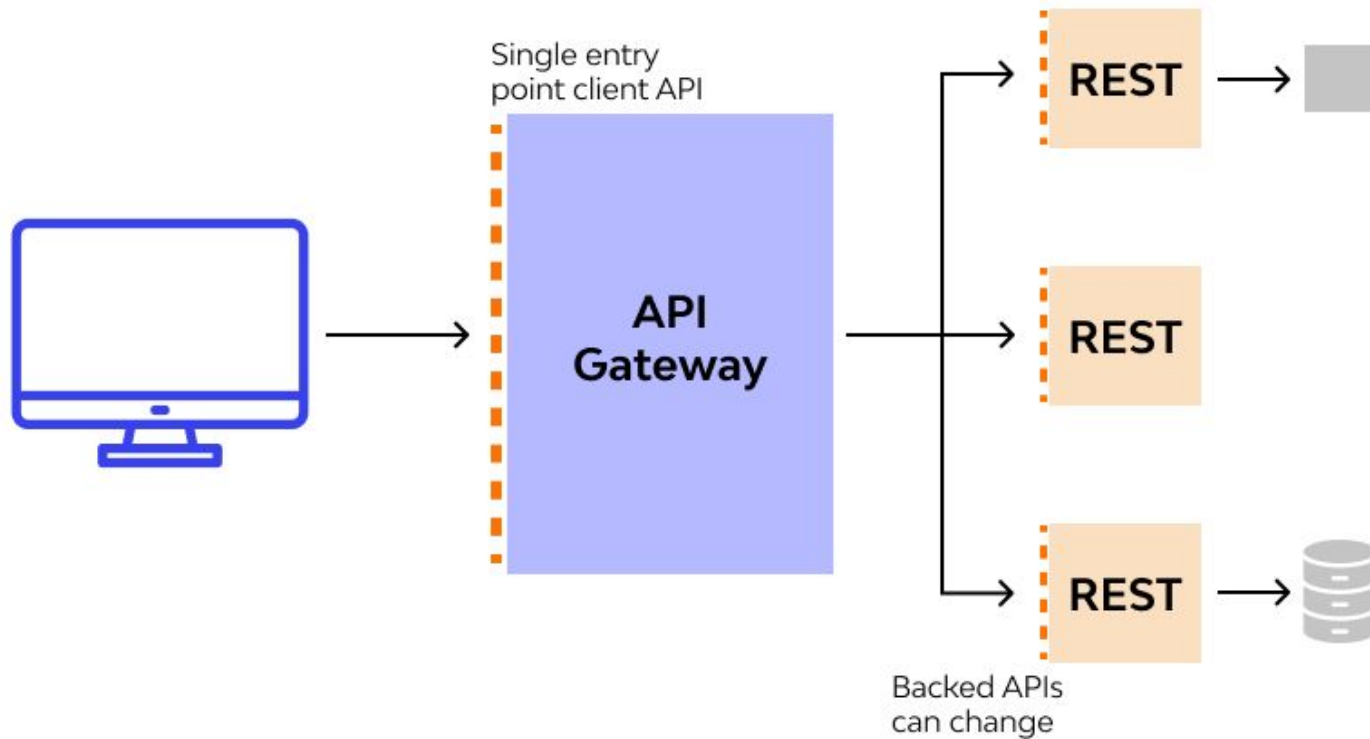
REST API Private:

- REST API Private is a variation of REST API that allows you to expose your API privately within a VPC (Virtual Private Cloud) without exposing it to the public internet.
- It is suitable for scenarios where you want to restrict access to your API to specific VPC resources or securely integrate with internal systems.
- REST API Private supports all the features available in REST APIs, including custom domain names, authorization mechanisms, and request/response handling.
- It provides enhanced security and isolation by leveraging private integration with AWS resources or using AWS PrivateLink to access resources within a VPC.

WebSocket APIs:

- WebSocket APIs are used for real-time, bidirectional communication between clients and servers over a single, long-lived connection.
- Unlike HTTP APIs and REST APIs, which use request/response model, WebSocket APIs enable full-duplex communication, allowing both clients and servers to send messages asynchronously.
- WebSocket APIs support features such as route management, message transformation, authorization, and integration with AWS Lambda for serverless processing.
- They are suitable for building applications that require real-time updates, such as chat applications, gaming platforms, or collaborative editing tools.

API Gateway



API Gateway



01. Building our first API

02.Exploring the flow

API Gateway



API Gateway > APIs > Resources - PetStore (1k9ggr35e1)

Resources

API actions ▼

Deploy API

Create resource



/

GET

/pets

GET

POST

/petId

GET

/breeds

GET

OPTIONS

/pets - GET - Method execution

Update documentation

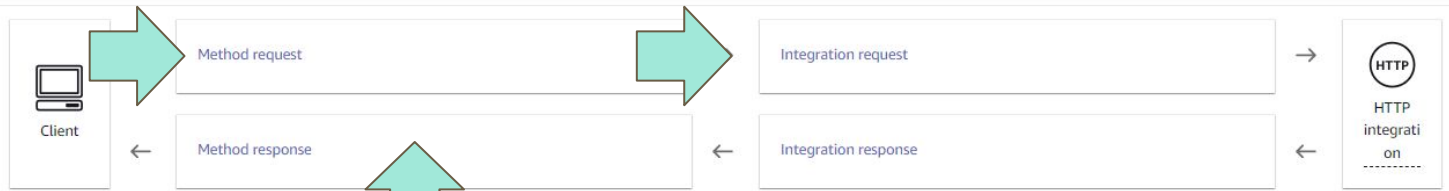
Delete

ARN

arn:aws:execute-api:us-east-1:992382569486:1k9ggr35e1/*/GET/pets

Resource ID

5rslzt



Method request

Integration request

Integration response

Method response

Test

API Gateway > APIs > Resources - PetStore (1k9ggr35e1) > Edit method request

Edit method request

Method request settings

Authorization

None

Request validator

None

☐ API key required

Operation name - optional

GetPets

► URL query string parameters

► HTTP request headers

► Request body

Cancel

Save

API Gateway > APIs > Resources - PetStore (1k9ggr35e1) > Edit integration request

Edit integration request

Method details

Integration type

☐ Lambda function

Integrate your API with a Lambda function.



☒ HTTP

Integrate with an existing HTTP endpoint.



☐ Mock

Generate a response based on API Gateway mappings and transformations.



☐ AWS service

Integrate with an AWS Service.



☐ VPC link

Integrate with a resource that isn't accessible over the public internet.



☒ HTTP proxy integration

Send the request to your HTTP endpoint without customizing the integration request or integration response.

HTTP method

GET

Endpoint URL

http://petstore.execute-api.us-east-1.amazonaws.com/petstore/pets

Content handling [Learn more](#)

Passthrough

☒ Default timeout

The default timeout is 29 seconds.

[API Gateway](#) > [APIs](#) > [Resources - PetStore \(1k9ggr35e1\)](#) > [Edit integration response](#)

Edit integration response [Info](#)

Response details

HTTP status regex [Info](#)

Method response status code

Content handling [Learn more](#) 

Header mappings [Info](#)

Response header

Mapping value

Access-Control-Allow-Origin

► Mapping templates

Cancel

Save

[API Gateway](#) > [APIs](#) > [Resources - PetStore \(1k9ggr35e1\)](#) > [Edit method response](#)

Edit method response [Info](#)

Response details

HTTP status code

Header name

Header name

Remove

Add header

Response body

Content type

Model

Remove

Add model

Cancel

Save

03.Providing mock data

04.Deploying to DEV env

API Gateway



aws

Services

Search

[Alt+S]

Certificate Manager

API Gateway

🔍

🔔

?

⚙️

N. Virginia

vfm

API Gateway

×

APIs

Custom domain names

VPC links

▼ API: PetStore

Resources

Stages

Authorizers

Gateway responses

Models

Resource policy

Documentation

Dashboard

API settings

Usage plans

API keys

Client certificates

+

dev

Logs and tracing

Info

Edit

CloudWatch logs

⊖ Inactive

Detailed metrics

⊖ Inactive

X-Ray tracing

⊖ Inactive

Custom access logging

⊖ Inactive

Stage variables

Deployment history

Documentation history

Canary

Tags

Deployments (5)

Info

Change active deployment

🔍

Find deployments

<

1

>

⚙️

	Deployment date	Status	Description	Deployment ID
<input type="radio"/>	March 19, 2024, 16:40 (UTC+00:00)	✔️ Active	enabling cors to globo.com	abso1l
<input type="radio"/>	March 19, 2024, 16:35 (UTC+00:00)	-	setting cors	ou3ixp
<input type="radio"/>	March 19, 2024, 16:33 (UTC+00:00)	-	setting cors	bykmfs
<input type="radio"/>	March 19, 2024, 16:26 (UTC+00:00)	-	updating breeds	q5a9x0
<input type="radio"/>	March 19, 2024, 15:36 (UTC+00:00)	-	dev environment	mjs67d

05.Using lambda function to replace the mock content

06.Enabling CORS

Paste this code below using **console browser** and press **RETURN**, after that run the code typing **main()**

```
function main() {  
    console.log("requesting breeds")  
    $.ajax({  
        url: "https://1k9ggr35e1.execute-api.us-east-1.amazonaws.com/dev/pets/breeds",  
        success: function(data) {  
            console.log(data)  
        }  
    })  
}
```

07.Enabling IAM Authentication

[API Gateway](#) > [APIs](#) > Resources - PetStore (1k9ggr35e1)

Edit method request

Resources

Create resource

- [-] /
 - GET
- [-] /pets
 - GET
 - POST
- [-] /{petId}
 - GET
- [-] /breeds
 - GET

/pets/breeds

ARN

arn:aws:execute-api:us-east-1:992382569486:1k9ggr35e1/*/GET/pets/breeds

Method request settings

Authorization

AWS IAM



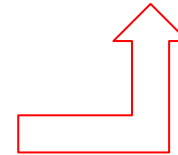
Client



Method request



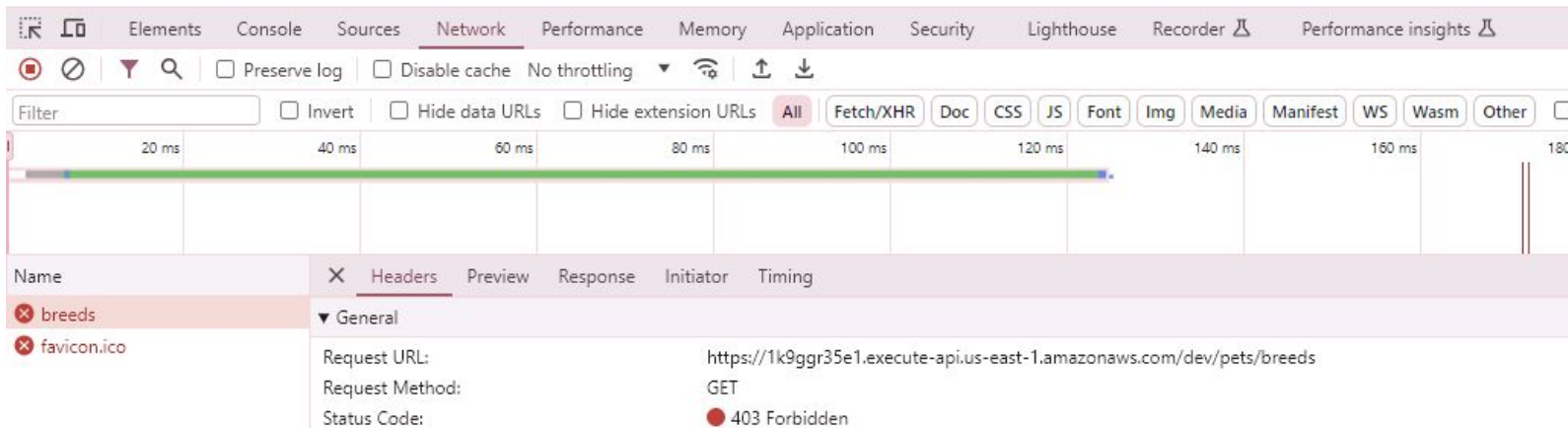
Method response



API Gateway



```
{"message": "Missing Authentication Token"}
```



API Gateway



The screenshot shows a REST client interface with the following components:

- URL Bar:** Displays the URL `https://1k9ggr35e1.execute-api.us-east-1.amazonaws.com/dev/pets/breeds`. The method is set to `GET`.
- Authorization:** The 'Type' is set to 'AWS Signature'. A warning box states: "Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables." The 'AccessKey' is `AKIA6ODU4JQHKBW2WUH` and the 'SecretKey' is `w48reyvUsUELUPM7S3fcjszBzr/QfmEt+NV...`.
- Response:** The 'Body' tab is active, showing a JSON response in 'Pretty' format:

```
1 {
2   "statusCode": 200,
3   "body": "[{"breed": \"Fluffywoof\", \"origin\": \"Toy Land\", \"size\": \"Small\", \"temperament\": [\"Playful\", \"Affectionate\", \"Energetic\"]}, {\"breed\": \"Golden Floof\", \"origin\": \"Sunny Meadows\", \"size\": \"Medium\", \"temperament\": [\"Friendly\", \"Loyal\", \"Intelligent\"]}, {\"breed\": \"Snugglepup\", \"origin\": \"Cozy Corner\", \"size\": \"Small\", \"temperament\": [\"Cuddly\", \"Gentle\", \"Sociable\"]}, {\"breed\": \"Thunderhound\", \"origin\": \"Stormy Mountains\", \"size\": \"Large\", \"temperament\": [\"Brave\", \"Strong\", \"Protective\"]}, {\"breed\": \"Spritelytail\", \"origin\": \"Enchanted Forest\", \"size\": \"Medium\", \"temperament\": [\"Playful\", \"Curious\", \"Agile\"]}]"
4 }
```

08.Resource Policy

API Gateway



Services

Search [Alt+S]

N. Virginia vformat

Certificate Manager

API Gateway

API Gateway

APIs

Custom domain names

VPC links

▼ API: PetStore

Resources

Stages

Authorizers

Gateway responses

Models

Resource policy

Documentation

Dashboard

API settings

✔ Successfully updated resource policy for 'PetStore'.

API Gateway > APIs > PetStore (1k9ggr35e1) > Resource policy

Resource policy Info

Use resource policies to configure access control to this API. You must redeploy your API for changes to this policy to take effect.

Policy details

Create policy

No resource policy

This API does not have a resource policy.

Create policy


```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "execute-api:Invoke",
      "Resource": "arn:aws:execute-api:us-east-1:992382569486:1k9ggr35e1/*/*/*"
    },
    {
      "Effect": "Deny",
      "Principal": "*",
      "Action": "execute-api:Invoke",
      "Resource": "arn:aws:execute-api:us-east-1:992382569486:1k9ggr35e1/*/*/*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": "89.180.236.241"
        }
      }
    }
  ]
}
```

aws

Services

Search

[Alt+S]

Certificate Manager

API Gateway

N. Virginia

vfmt

API Gateway

×

APIs

Custom domain names

VPC links

▼ API: PetStore

Resources

Stages

Authorizers

Gateway responses

Models

Resource policy

Documentation

Dashboard

API settings

Usage plans

API keys

Client certificates

Settings

API Gateway > APIs > PetStore (1k9ggr35e1) > Resource policy

Resource policy [Info](#)

Use resource policies to configure access control to this API. You must redeploy your API for changes to this policy to take effect.

Policy details

Edit

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Principal": "*",
7       "Action": "execute-api:Invoke",
8       "Resource": "arn:aws:execute-api:us-east-1:992382569486:1k9ggr35e1/*/*/*"
9     },
10    {
11      "Effect": "Deny",
12      "Principal": "*",
13      "Action": "execute-api:Invoke",
14      "Resource": "arn:aws:execute-api:us-east-1:992382569486:1k9ggr35e1/*/*/*",
15      "Condition": {
16        "NotIpAddress": {
17          "aws:SourceIp": "89.180.236.141"
18        }
19      }
20    }
21  ]
22 }
```

API Gateway



Postman interface showing a REST client request configuration and response.

Request Configuration:

- Method: GET
- URL: `https://1k9ggr35e1.execute-api.us-east-1.amazonaws.com/dev/pets/breeds`
- Environment: No environment

Params: Authorization, Headers (10), Body, Pre-request Script, Tests, Settings

Query Params:

Key	Value	Description
Key	Value	Description

Body: Cookies, Headers (8), Test Results

Status: 200 OK Time: 142 ms Size: 1.05 KB Save as example

Response Body (JSON):

```
{
  "statusCode": 200,
  "body": [
    {
      "breed": "Fluffywoof",
      "origin": "Toy Land",
      "size": "Small",
      "temperament": ["Playful", "Affectionate", "Energetic"]
    },
    {
      "breed": "Golden Floof",
      "origin": "Sunny Meadows",
      "size": "Medium",
      "temperament": ["Friendly", "Loyal", "Intelligent"]
    },
    {
      "breed": "Snugglepup",
      "origin": "Cozy Corner",
      "size": "Small",
      "temperament": ["Cuddly", "Gentle", "Sociable"]
    },
    {
      "breed": "Thunderhound",
      "origin": "Stormy Mountains",
      "size": "Large",
      "temperament": ["Brave", "Strong", "Protective"]
    },
    {
      "breed": "Spritelytail",
      "origin": "Enchanted Forest",
      "size": "Medium",
      "temperament": ["Playful", "Curious", "Agile"]
    }
  ]
}
```

API Gateway



Postman interface showing an API request to `https://1k9ggr35e1.execute-api.us-east-1.amazonaws.com/dev/pets/breeds` using the GET method. The response is a 403 Forbidden status with a message indicating the user is not authorized to perform the `execute-api:Invoke` action on the resource `arn:aws:execute-api:us-east-1:*****9486:1k9ggr35e1/dev/GET/pets/breeds` with an explicit deny.

The interface includes a sidebar with Collections, Environments, and History. The main area shows the request details, including the URL, method, and response body. A large red arrow points to the response body.

Query Params

Key	Value	Description
Key	Value	Description

Body

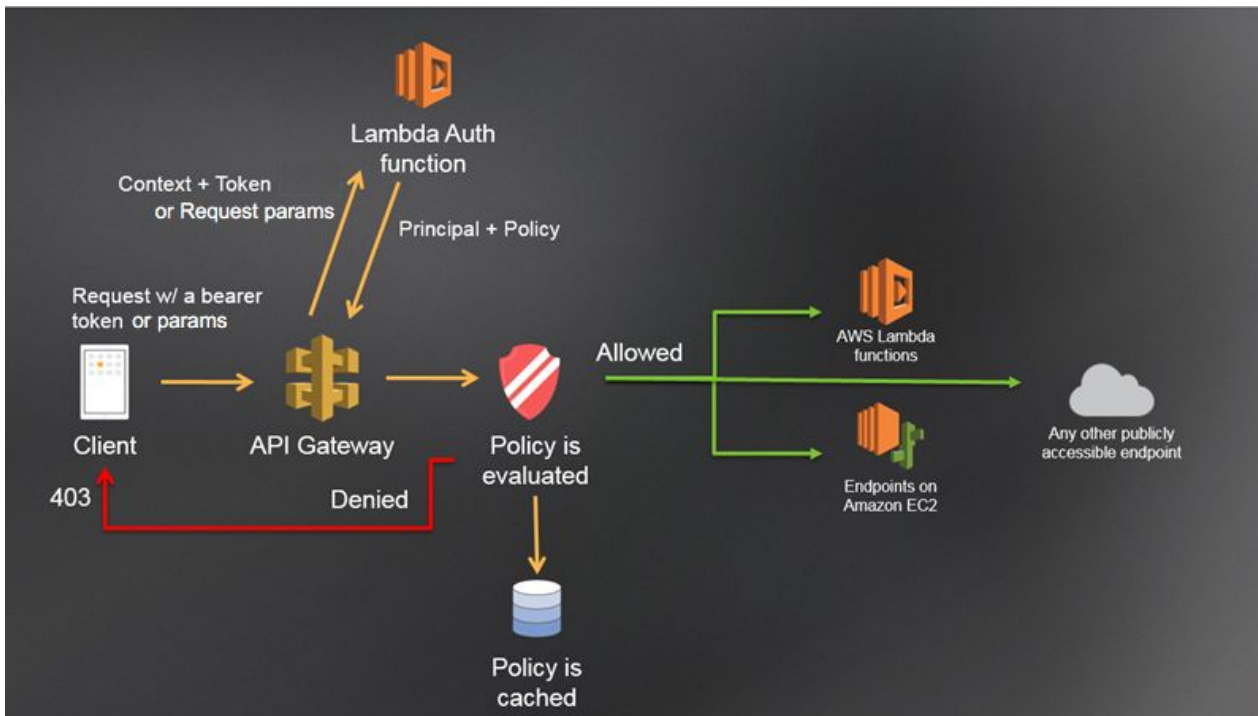
Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "User: arn:aws:iam::992382569486:user/api-user is not authorized to perform: execute-api:Invoke on resource:
3     arn:aws:execute-api:us-east-1:*****9486:1k9ggr35e1/dev/GET/pets/breeds with an explicit deny"
```

09.Lambda Authorizers

API Gateway

A *Lambda authorizer* (formerly known as a *custom authorizer*) is an **API Gateway feature** that uses a **Lambda function** to control access to your API.








Token-Based Authorizer:

- Token-based authorizers are used when clients provide tokens (such as JWT - JSON Web Tokens) as part of the request, typically in the Authorization header.
- The Lambda authorizer extracts the token from the request, validates its authenticity (e.g., verifies the signature, expiration), and optionally checks the token against an external identity provider (e.g., AWS Cognito, OAuth provider).
- If the token is valid and authorized, the authorizer returns an IAM policy allowing access to the requested resource.
- Token-based authorizers are **commonly used for authentication purposes, where clients provide tokens to prove their identity and access rights.**

Request-Based Authorizer:

- Request-based authorizers are used when clients provide additional information in the request payload, headers, or query parameters to determine access control.
- The Lambda authorizer receives the entire request and examines its content to determine whether the requester is authorized to access the resource.
- Based on the information provided in the request, the authorizer constructs an IAM policy allowing or denying access to the requested resource.
- Request-based authorizers are more flexible and can accommodate various authentication and authorization mechanisms beyond token-based authentication. **For example, they can perform custom logic based on request attributes, user attributes, or external data sources.**

  [Alt+S]

 Certificate Manager  API Gateway  Lambda

Lambda > Functions > Create function

Create function [Info](#)

Choose one of the following options to create your function.

☒ Author from scratch
Start with a simple Hello World example.

☐ Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.


☐ Container image
Select a container image to deploy


Basic information

Function name
Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.





Architecture [Info](#)
Choose the instruction set architecture you want for your function code.

☒ x86_64 ☐ arm64

Permissions [Info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

API Gateway



Lambda > Functions > authorizer

authorizer

Throttle Copy ARN Actions

Function overview Info

Export to Application Composer Download

Diagram Template

authorizer

Layers (0)

API Gateway

+ Add trigger

+ Add destination

Description
-

Last modified
1 minute ago

Function ARN
arn:aws:lambda:us-east-1:992382569486:function:authorizer

Function URL Info
-

Code Test Monitor Configuration Aliases Versions

Code source Info

Upload from

File Edit Find View Go Tools Window Test Deploy Changes not deployed

Go to Anything (Ctrl-P)

index.mjs Environment Var Execution results

Environment

authorizer - / index.mjs

1

API Gateway



Code Test Monitor Configuration Aliases Versions

Code source Info

Upload from ▼

File Edit Find View Go Tools Window Test Deploy Changes not deployed

Go to Anything (Ctrl-P)

Environment

authorizer -/

index.mjs

index.mjs

Environment Var X

Execution results X

```
1 export async function handler (event, context ) {
2
3   console.log('Received event:', JSON.stringify(event));
4
5   // from header, extracts the authorization code and source IP
6   const authorization = event.headers['Authorization']
7   const sourceIp = event.requestContext.identity.sourceIp ?? ''
8   const authCode = getCredentialContent(authorization) ?? ''
9
10  try {
11    console.log("authorization: ", authorization)
12    console.log("sourceIp: ", sourceIp)
13    console.log("authCode: ", authCode)
14
15    // auth code must be available
16    if (authCode === '') {
17      throw new Error("Validation Error (authCode)")
18    }
19
20    if (authCode !== 'AKIA6ODU4JQHKBW2WJH') {
21      return authorizerResponse('Deny', sourceIp, authCode)
22    }
23
24    // here we are able to validate the IP
25    // build your function to checkIP
26    // const ipAllowed = await checkIP(sourceIp)
27    // if(!ipAllowed){
28    //   return authorizerResponse('Deny', sourceIp, authCode)
29    // }
30
31    return authorizerResponse('Allow', sourceIp, authCode)
32  } catch (err) {
33    console.log(err)
34    return authorizerResponse('Deny', sourceIp, authCode)
35  }
36 }
37 }
```

7:64 JavaScript Spaces: 2

```
export async function handler (event, context ) {
  // from header, extracts the authorization code and source IP
  const authorization = event.headers['Authorization']
  const sourceIp = event.requestContext.identity.sourceIp ?? ''
  const authCode = getCredentialContent(authorization) ?? ''
  try {
    // auth code must be available
    if (authCode === '') {
      throw new Error("Validation Error (authCode)")
    }
    // forcing authCode to be a valid user 'AKIA6ODU4JQHKB JW2WUH'
    if (authCode !== 'AKIA6ODU4JQHKB JW2WUH') {
      return authorizerResponse('Deny', sourceIp, authCode)
    }

    return authorizerResponse('Allow', sourceIp, authCode)

  } catch (err) {
    console.log(err)
    return authorizerResponse('Deny', sourceIp, authCode)
  }
};
```

API Gateway

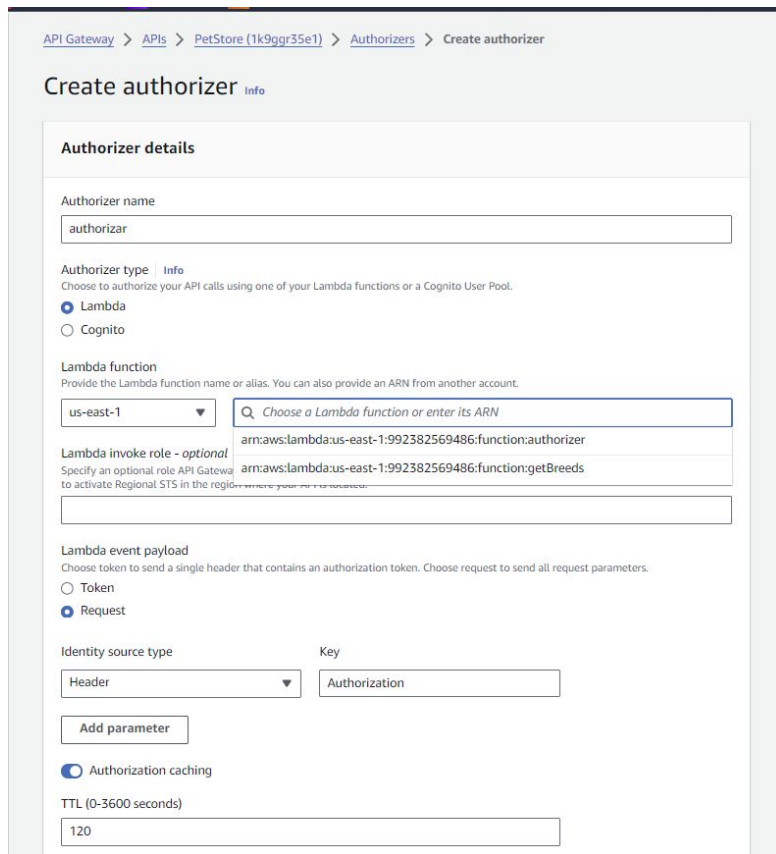
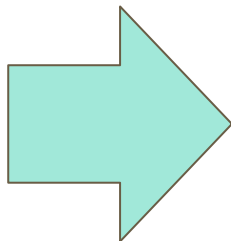
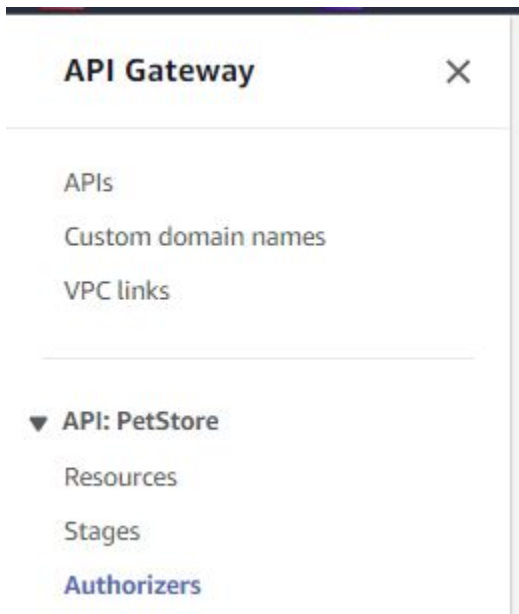


```
// Using REGEX extracts the credential content
function getCredentialContent(header) {
  // Use regular expression to extract the credential content
  const match = header.match(/Credential=([^\s]+)/);

  // Check if the match is found
  if (match && match.length > 1) {
    return match[1];
  } else {
    return null
  }
}
```

```
// build the final Policy (Allow)
function authorizerResponse (effect, sourceIp, authCode) {
  const response = {
    principalId: sourceIp,
    context: {
      "sourceIp": sourceIp,
      "authCode": authCode
    },
    policyDocument: {
      Version: '2012-10-17',
      Statement: [
        {
          Action: 'execute-api:Invoke',
          Effect: effect,
          Resource: '*'
        },
      ],
    },
  }
  return response
}
```

API Gateway



The screenshot shows the 'Create authorizer' page in the AWS API Gateway console. The breadcrumb trail at the top reads: 'API Gateway > APIs > PetStore (1k9ggr35e1) > Authorizers > Create authorizer'. The page title is 'Create authorizer' with an 'Info' link. The form is divided into several sections:

- Authorizer details**
 - Authorizer name:** A text input field containing 'authorizar'.
 - Authorizer type:** A section with an 'Info' link. It states 'Choose to authorize your API calls using one of your Lambda functions or a Cognito User Pool.' There are two radio buttons: 'Lambda' (selected) and 'Cognito'.
 - Lambda function:** A section with the instruction 'Provide the Lambda function name or alias. You can also provide an ARN from another account.' It includes a dropdown menu set to 'us-east-1' and a search input field containing 'arn:aws:lambda:us-east-1:992382569486:function:authorizer'.
 - Lambda invoke role - optional:** A section with the instruction 'Specify an optional role API Gateway to activate Regional STS in the region where your API is located.' It includes a text input field containing 'arn:aws:lambda:us-east-1:992382569486:function:getBreeds'.
- Lambda event payload**
 - A section with the instruction 'Choose token to send a single header that contains an authorization token. Choose request to send all request parameters.' It has two radio buttons: 'Token' and 'Request' (selected).
 - Identity source type:** A dropdown menu set to 'Header'.
 - Key:** A text input field containing 'Authorization'.
 - Add parameter:** A button.
 - Authorization caching:** A section with a checked radio button and the label 'Authorization caching'.
 - TTL (0-3600 seconds):** A text input field containing '120'.

API Gateway

[API Gateway](#) > [APIs](#) > Resources - PetStore (1k9ggr35e1)

Resources

Create resource



/

GET



/pets

GET

POST



/petId

GET




/breeds

GET

OPTIONS

/pets/breeds - GET - Method execution

ARN

 `arn:aws:execute-api:us-east-1:992382569486:1k9ggr35e1/*/GET/pets/breeds`



[Method request](#) | [Integration request](#) | [Integration response](#) | [Method response](#) | [Test](#)

[API Gateway](#) > [APIs](#) > [Resources - PetStore \(1k9ggr35e1\)](#) > Edit method request

Edit method request

Method request settings

Authorization

authorizer	▲
None	
AWS IAM	
Request authorizers	
authorizer	✓

Operation name - optional

GetPets

► URL query string parameters

► HTTP request headers

► Request body

Cancel

Save

API Gateway



<https://1k9ggr35e1.execute-api.us-east-1.amazonaws.com/dev/pets/breeds>SaveEdit

GET <https://1k9ggr35e1.execute-api.us-east-1.amazonaws.com/dev/pets/breeds> Send

Params

Authorization •

Headers (10)

Body

Pre-request Script

Tests

Settings

Cookies

Type

AWS Signature

The authorization header will be automatically generated when you send the request. Learn more about [AWS Signature](#) authorization.

Add authorization data to

Request Headers

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#).

AccessKey

AKIA6ODU4JQHKBW2WUH

SecretKey

w48reyvUsUELUPM7S3fcjszBzr/QfmEt+NV ...

Advanced configuration

Postman auto-generates default values for some of these fields unless a value is specified.

AWS Region

e.g. us-east-1

Service Name

e.g. s3

Session Token

Session Token

Body

Cookies

Headers (8)

Test Results

Status: 200 OK

Time: 245 ms

Size: 1.06 KB

Save as example

Pretty

Raw

Preview

Visualize

JSON

```
{
  "statusCode": 200,
  "body": "[{"breed": \"Fluffywoof\", \"origin\": \"Toy Land\", \"size\": \"Small\", \"temperament\": [\"Playful\", \"Affectionate\", \"Energetic\"]}, {\"breed\": \"Golden Floof\", \"origin\": \"Sunny Meadows\", \"size\": \"Medium\", \"temperament\": [\"Friendly\", \"Loyal\", \"Intelligent\"]}, {\"breed\": \"Snugglepup\", \"origin\": \"Cozy Corner\", \"size\": \"Small\", \"temperament\": [\"Cuddly\", \"Gentle\", \"Sociable\"]}, {\"breed\": \"Thunderhound\", \"origin\": \"Stormy Mountains\", \"size\": \"Large\", \"temperament\": [\"Brave\", \"Strong\", \"Protective\"]}, {\"breed\": \"Spritelytail\", \"origin\": \"Enchanted Forest\", \"size\": \"Medium\", \"temperament\": [\"Playful\", \"Curious\", \"Agile\"]}]\""]
```


API Gateway



<https://1k9ggr35e1.execute-api.us-east-1.amazonaws.com/dev/pets/breeds> Save Edit Send

GET <https://1k9ggr35e1.execute-api.us-east-1.amazonaws.com/dev/pets/breeds>

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

Type

AWS Signature

The authorization header will be automatically generated when you send the request. Learn more about [AWS Signature](#) authorization.

Add authorization data to

Request Headers

ⓘ Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about [variables](#).

AccessKey

AKIA6ODU4JQHKBW2WUH999

SecretKey

w48reyvUsUELUPM7S3fcjszBzr/QfmEt+NV...

Advanced configuration

Postman auto-generates default values for some of these fields unless a value is specified.

AWS Region

e.g. us-east-1

Service Name

e.g. s3

Session Token

Session Token

Body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize JSON

1 {

2 "message": "User is not authorized to access this resource with an explicit deny"

3 }

Status: 403 Forbidden Time: 219 ms Size: 547 B Save as example

10.Custom Domain

API Gateway

The screenshot shows the AWS Route 53 console. The left sidebar contains navigation links: Dashboard, Hosted zones, Health checks, IP-based routing, and Traffic flow. The main content area is titled 'Route 53 Dashboard' and includes a 'DNS management' section with a description: 'A hosted zone tells Route 53 how to respond to DNS queries for a domain such as example.com.' Below this description is a 'Create hosted zone' button, which is highlighted by a large green arrow pointing towards it. At the bottom of the dashboard, there is a 'Register domain' section.

[Route 53](#) > [Hosted zones](#) > [Create hosted zone](#)

Create hosted zone [Info](#)

Hosted zone configuration

A hosted zone is a container that holds information about how you want to route traffic for a domain, such as example.com, and its subdomains.

Domain name [Info](#)

This is the name of the domain that you want to route traffic for.

Valid characters: a-z, 0-9, ! " # \$ % & ' () * + , - . / : ; < = > ? @ [\] ^ _ ` { | } . ~

Description - optional [Info](#)

This value lets you distinguish hosted zones that have the same name.

The description can have up to 256 characters. 0/256

Type [Info](#)

The type indicates whether you want to route traffic on the internet or in an Amazon VPC.

☒ Public hosted zone

A public hosted zone determines how traffic is routed on the internet.

☐ Private hosted zone

A private hosted zone determines how traffic is routed within an Amazon VPC.

Tags [Info](#)

Apply tags to hosted zones to help organize and identify them.

No tags associated with the resource.

You can add up to 50 more tags.

[Cancel](#)

[Create hosted zone](#)

API Gateway



[Route 53](#) > [Hosted zones](#) > [bjss-aws.pt](#)

Public [bjss-aws.pt](#) [Info](#)

Delete zone

Test record

Configure query logging

► Hosted zone details

Edit hosted zone

[Records \(2\)](#)

[DNSSEC signing](#)

[Hosted zone tags \(0\)](#)

Records (2) [Info](#)



Delete record

Import zone file

Create record

Automatic mode is the current search behavior optimized for best filter results. To change modes go to [settings](#).

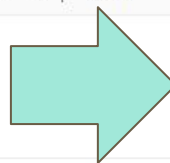
Type ▼

Routing pol... ▼

Alias ▼

< 1 >

<input type="checkbox"/>	Record name	Type ▼	Routin... ▼	Differ... ▼	Alias ▼	Value/Route traffic to ▼	TTL (s)
<input type="checkbox"/>	bjss-aws.pt	NS	Simple	-		ns-1549.awsdns-01.co.uk. ns-95.awsdns-11.com. ns-1301.awsdns-34.org. ns-708.awsdns-24.net.	17280
<input type="checkbox"/>	bjss-aws.pt	SOA	Simple	-	No	ns-1549.awsdns-01.co.uk. a...	900



API Gateway



PAINEL DE CONTROLO

bjss-aws.pt

[Voltar](#)



[Voltar à página de gestão](#)

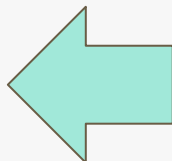
Este domínio tem o DNS:

ns-1549.awsdns-01.co.uk

ns-95.awsdns-11.com

ns-1301.awsdns-34.org

ns-708.awsdns-24.net



Renato Matos

Código de Cliente: RM5118-AMPT

[Gestão da conta, facturas e pagamentos »](#)

OS SEUS PRODUTOS

PRODUTOS POR ACTIVAR

DOMÍNIOS E PRODUTOS

Domínios

bjss-aws.pt

Security, Identity, & Compliance

AWS Certificate Manager

Easily provision, manage, deploy,
and renew SSL/TLS certificates

AWS Certificate Manager > Certificates > Request certificate

Request certificate

Certificate type [Info](#)

ACM certificates can be used to establish secure communications access across the internet or within an internal network. Choose the type of certificate for ACM to provide.

☒ Request a public certificate

Request a public SSL/TLS certificate from Amazon. By default, public certificates are trusted by browsers and operating systems.

☐ Request a private certificate

No private CAs available for issuance.

Requesting a private certificate requires the creation of a private certificate authority (CA). To create a private CA, visit [AWS Private Certificate Authority](#)

Cancel

Next

New ACM managed certificate

Request a public certificate from Amazon or a private certificate from your organization's certificate authority (CA).

[Request a certificate](#)

Import certificates that you obtained

API Gateway



AWS Certificate Manager > Certificates > Request certificate > Request public certificate

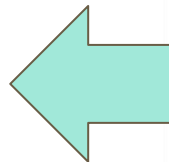
Request public certificate

Domain names

Provide one or more domain names for your certificate.

Fully qualified domain name [Info](#)

You can add additional names to this certificate. For example, if you're requesting a certificate for "www.example.com", you might want to add the name "example.com" so that customers can reach your site by either name.



AWS Certificate Manager > Certificates

Certificates (1)

< 1 > ⚙

<input type="checkbox"/>	Certificate ID	Domain name	Type	Status	In use	Renewal eligibility	Key algorithm
<input type="checkbox"/>	781d795d-8816-4972-9a46-e7223dab7ba4	*.bjss-aws.pt	Amazon Issued	⌚ Pending validation	No	Ineligible	RSA 2048



API Gateway



AWS Certificate Manager > Certificates > 781d795d-8816-4972-9a46-e7223dab7ba4

781d795d-8816-4972-9a46-e7223dab7ba4

Delete

Certificate status

Identifier

781d795d-8816-4972-9a46-e7223dab7ba4

Status

Pending validation [Info](#)

ARN

arn:aws:acm:us-east-1:992382569486:certificate/781d795d-8816-4972-9a46-e7223dab7ba4

Type

Amazon Issued

Domains (2)

Create records in Route 53

Export to CSV

< 1 >

Domain	Status	Renewal status	Type	CNAME name	CNAME value
*.bjss-aws.pt	Pending validation	-	CNAME	_3bfc79bb2a0fca01316c0a06a46e3ba9.bjss-aws.pt.	_155d507151c9d8e2c536dc954585d528.mhbtspdnt.acm-validations.aws.
bjss-aws.pt	Pending validation	-	CNAME	_3bfc79bb2a0fca01316c0a06a46e3ba9.bjss-aws.pt.	_155d507151c9d8e2c536dc954585d528.mhbtspdnt.acm-validations.aws.



<https://github.com/renatomatos79/bjss-aws-api-gateway>
renato.matos79@gmail.com

