

PASOS PARA LA IMPLEMENTACIÓN DEL PROYECTO COOKBOOK

1. Creamos el bucket s3 donde estará el frontend de nuestra web.

Create bucket [info](#)

Buckets are containers for data stored in S3.

General configuration

AWS Region
US East (N. Virginia) us-east-1

Bucket type [info](#)

☒ **General purpose**
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory**
Recommended for low latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [info](#)
recetas-frontend-immune
Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). [Learn More](#)

Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.
[Choose bucket](#)
Format: <id>/<bucket>/<profile>

Object Ownership [info](#)
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☐ **ACLs disabled (recommended)**
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☒ **ACLs enabled**
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership
☒ **Bucket owner preferred**
If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

☐ **Object writer**
The object writer remains the object owner.

☒ If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. [Learn more](#)

En la configuración, permitimos el acceso público y lo habilitamos la opción de hostear sitio web estático, colocando el archivo frontend/index.html como index document y frontend/error.html como backend. Además, configuramos la política para que se pueda hacer un getObject desde cualquier punto. (Con esto, se podrá acceder al frontend a través de internet)

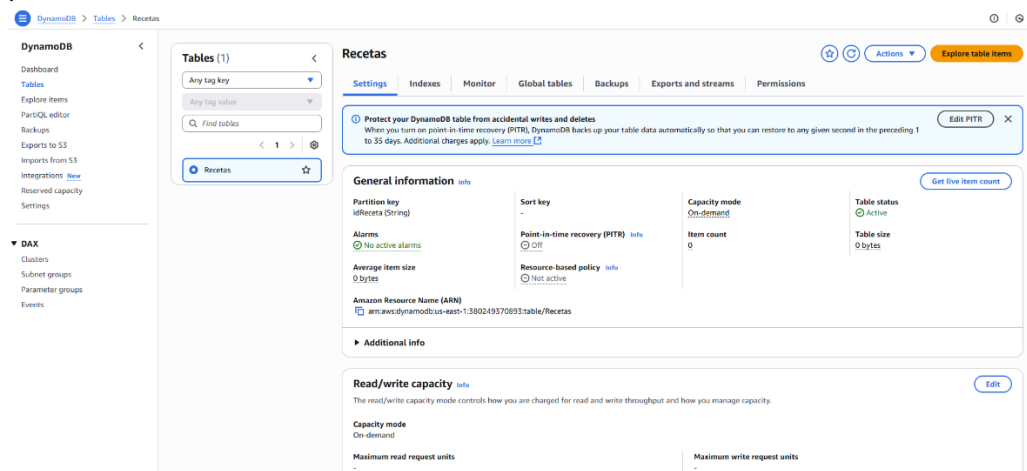
Bucket policy [Edit](#) [Delete](#)

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

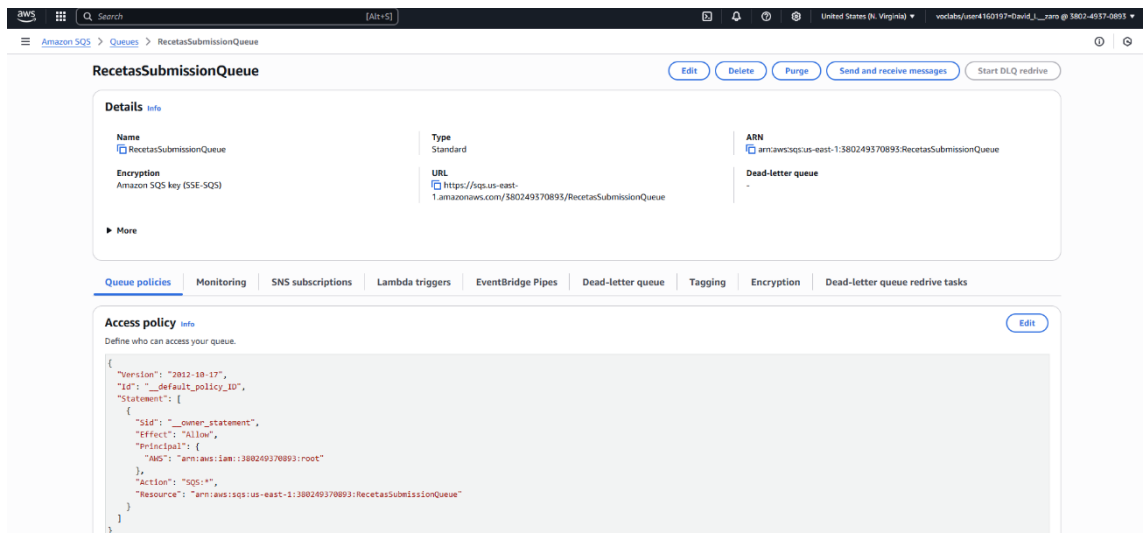
[Copy](#)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::recetas-frontend-immune/*"
    }
  ]
}
```

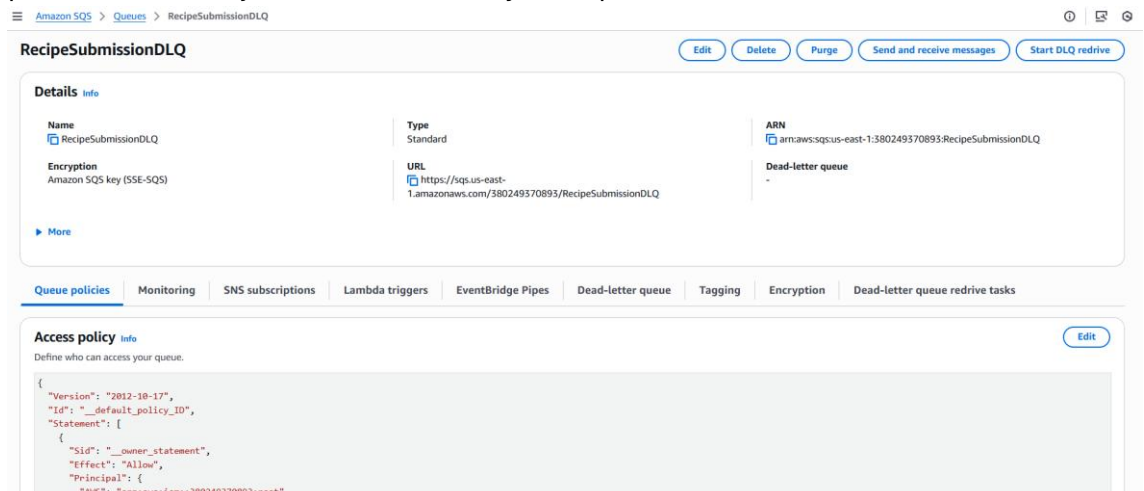
2. Creamos la tabla de DynamoDB donde se almacenarán los datos. La clave primaria será un idReceta.



3. Creación de la cola RecipeSubmissionQueue con la que desacoplaremos el procesamiento de los mensajes.



4. Creación de la cola RecipeSubmissionDLQ y la configuramos como Dead-Letter queue de la cola principal. Con esto conseguimos que, si se produce algún error al procesar el mensaje, este se almacene y no se pierda.



5. Creación de un topic y una subscripción a dicho topic para poder recibir mensajes en caso de que haya mensajes en la DLQ, lo que significará que el proceso ha fallado.

Mensaje_Fallo [Edit](#) [Delete](#) [Publish message](#)

Details

Name Mensaje_Fallo	Display name -
ARN arn:aws:sns:us-east-1:380249370893:Mensaje_Fallo	Topic owner 380249370893
Type Standard	

[Subscriptions](#) | [Access policy](#) | [Data protection policy](#) | [Delivery policy \(HTTP/S\)](#) | [Delivery status logging](#) | [Encryption](#) | [Tags](#) | [Integrations](#)

Subscriptions (1) [Edit](#) [Delete](#) [Request confirmation](#) [Confirm subscription](#) [Create subscription](#)

ID	Endpoint	Status	Protocol
7db5dc16-4e16-43b2-8ea9-e386e3e2aa07	david.lazaro@alumni.immune.institute	Confirmed	EMAIL

6. Creación de una alarma en CloudWatch que nos avise cuando haya algún mensaje en la DLQ.

Select metric [×](#)

ApproximateNumberOfMessagesVisible [🔗](#) 1h 3h 12h 1d 3d 1w Custom UTC timezone Line [🔄](#) [📄](#)

Count

2
1.5
1

13:30 13:45 14:00 14:15 14:30 14:45 15:00 15:15 15:30 15:45 16:00 16:15

■ ApproximateNumberOfMessagesVisible

[Browse](#) | [Multi source query](#) | [Graphed metrics \(1\)](#) | [Options](#) | [Source](#) = [Add math](#) [Add query](#)

<input type="checkbox"/>	RecipeSubmissionDLQ	NumberofMessagesDeleted 🔗	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	NumberofMessagesReceived 🔗	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	ApproximateNumberOfMessagesDelayedInQuietGroups	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	ApproximateNumberOfMessagesDelayed 🔗	No alarms
<input checked="" type="checkbox"/>	RecipeSubmissionDLQ	ApproximateNumberOfMessagesVisible 🔗	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	ApproximateNumberOfMessagesNotVisibleInQuietGroups	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	ApproximateNumberOfNoisyGroups	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	ApproximateNumberOfMessagesNotVisible 🔗	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	ApproximateAgeOfOldestMessage 🔗	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	ApproximateNumberOfMessagesVisibleInQuietGroups	No alarms
<input type="checkbox"/>	RecipeSubmissionDLQ	ApproximateAgeOfOldestMessageInQuietGroups	No alarms

[CloudWatch](#) > [Alarms](#) > [Create alarm](#) Cancel [Select metric](#)

Alarm recommendations available
Turn on Recommendations to pre-populate the wizard with the recommended alarms.

Step 1
Specify metric and conditions

Step 2
Configure actions

Step 3
Add alarm details

Step 4
Preview and create

Notification

Alarm state trigger
Define the alarm state that will trigger this action.

☒ In alarm
The metric or expression is outside of the defined threshold.

☐ OK
The metric or expression is within the defined threshold.

☐ Insufficient data
The alarm has just started or not enough data is available.

Send a notification to the following SNS topic
Define the SNS (Simple Notification Service) topic that will receive the notification.

☒ Select an existing SNS topic

☐ Create new topic

☐ Use topic ARN to notify other accounts

Send a notification to...

×

Only topics belonging to this account are listed here. All persons and applications subscribed to the selected topic will receive notifications.

Email (endpoints)
david.lazaro@alumni.immune.institute - [View in SNS Console](#)

[Add notification](#)

7. Creación de la lambda submitRecetaLambda que será la encargada de recibir las peticiones PUT y POST. Configuración de las variables de entorno necesarias en el código.

The image displays two screenshots of the AWS Lambda console interface.

Top Screenshot: Create function

- 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 for your function.)
- Basic information**
 - Function name:** submitRecetaLambda
 - Runtime:** Python 3.11
 - Architecture:** x86_64
 - Permissions:** Use an existing role (LabRole)

Bottom Screenshot: submitRecetaLambda

- Function overview:** Shows the function name, layers, and a diagram.
- Configuration:**
 - Environment variables (1):**

Key	Value
SQS_QUEUE_URL	https://sqs.us-east-1.amazonaws.com/380249370893/RecetasSubmissionQueue

- Creación de la lambda procesarRecetaLambda que será encargada de recibir los mensajes de la cola RecipeSubmissionQueue y ejecutar la acción requerida contra la base de datos. Se le añade dicha cola como trigger.

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 for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.
processorRecetasLambda
Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

Runtime info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.13

Architecture info
Choose the instruction set architecture you want for your function code.
☐ arm64
☒ x86_64

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.

Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).
☐ Create a new role with basic Lambda permissions
☒ Use an existing role
☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
LabRole
[View the LabRole role](#) on the IAM console.

Additional configurations
Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.

Add trigger

Trigger configuration info
SQS
aws event-source-mapping polling queue

SQS queue
Choose or enter the ARN of an SQS queue.
arn:aws:sqs:us-east-1:380249370893:recetasSubmissionQueue

Event poller configuration

☒ **Activate trigger**
Select to activate the trigger now. Keep unchecked to create the trigger in a deactivated state for testing (recommended).

☐ **Enable metrics**
Monitor your event source with metrics. You can view those metrics in CloudWatch console. Enabling this feature incurs additional costs. [Learn more](#)

Batch size - optional
The maximum number of records in each batch to send to the function.
10
The maximum is 10,000 for standard queues and 10 for FIFO queues.

Batch window - optional
The maximum amount of time to gather records before invoking the function, in seconds.
0
When the batch size is greater than 10, set the batch window to at least 1 second.

Maximum concurrency - optional
The maximum number of concurrent function instances that the SQS event source can invoke.
Specify a value between 2 and 1000. To deactivate, leave the box empty.

Report batch item failures - optional
Allow your function to return a partial successful response for a batch of records.
☐

Filter criteria - optional
Define the filtering criteria to determine whether or not to process an event. Each filter must be in a valid JSON format in filter rule syntax. [Learn more](#)

9. Creación de las lambdas getRecetasLambda y eliminarRecetaLambda que serán las encargadas de recibir los métodos GET y DELETE respectivamente.

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 for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.
getRecetasLambda
Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

Runtime info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.13

Architecture info
Choose the instruction set architecture you want for your function code.
☐ arm64
☒ x86_64

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.

Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).
☐ Create a new role with basic Lambda permissions
☒ Use an existing role
☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
LabRole
[View the LabRole role](#) on the IAM console.

Create function

Choose one of the following options to create your function.

- ☒ **Author from scratch**
Start with a simple Node.js 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 for your function.

Basic information

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

Runtime
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.12

Architecture
Choose the instruction set architecture you want for your function code.
☒ x86_64 ☐ ARM64

Permissions
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.

Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

- ☐ Create a new role with basic Lambda permissions
- ☒ Use an existing role
- ☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. This role must have permission to upload logs to Amazon CloudWatch Logs.
Lambda

10. Configuración de las rutas en la API Gateway. Desde la ruta /recetas se podrán obtener todas las recetas con una método GET y añadir una nueva con el método POST.

Desde el endpoint /recetas/{idReceta} podremos modificar recetas a partir de su id con el método PUT, obtener los detalles de una receta en particular con el método GET o borrar una receta con el método DELETE. En cada endpoint se configura la lambda correspondiente.

API Gateway

Resources - RecetasAPI (vrtprb76)

API Gateway

- APIs
- Custom domain names
- Domain name access associations
- VPC links

▼ API: RecetasAPI

- Resources
- Stages
- Authorizers
- Gateway responses
- Models
- Resource policy
- Documentation
- Dashboard
- API settings

Usage plans

- API keys
- Client certificates
- Settings

Resources

Create resource

- /
- /recetas
 - GET
 - POST
 - /recetas/{idReceta}
 - DELETE
 - GET
 - PUT

/recetas - POST - Method execution

ARN: arn:aws:execute-api:us-east-1:380245370893:rtprb76/POST/recetas

Resource ID: api1rf

Client → Method request → Integration request → Lambda integration

← Method response ← Integration response

Method request settings

Authorization: NONE

API key required: False

Request validator: None

SDK operation name: Generated based on method and path

Request paths (0)

No request paths defined

Se debe habilitar el CORS para poder enviar elementos desde el frontend y habilitar el Lambda proxy integration para poder enviar el evento con todas las cabeceras a la lambda.

API Gateway > APIs > Resources - RecetasAPI [endpoint:75] > Enable CORS

Successfully created method DELETE in [idReceta]. Redeploy your API for the update to take effect.

Enable CORS

CORS settings [Info](#)

To allow requests from scripts running in the browser, configure cross-origin sharing (CORS) for your API. When you save your configuration, API Gateway replaces any existing CORS settings with your new configuration.

Gateway responses

API Gateway will configure CORS for the selected gateway responses.

☐ Default: 40X

☐ Default: 5XX

Access-Control-Allow-Methods

☒ GET

☐ OPTIONS

☒ POST

Access-Control-Allow-Headers

API Gateway will configure CORS for the selected gateway responses.

Access-Control-Allow-Origin

Enter an origin that can access the resource. Use a wildcard "*" to allow any origin to access the resource.

► Additional settings

[Cancel](#) [Save](#)

Method request | **Integration request** | Integration response | Method response | Test

Integration request settings

Integration type [Info](#)

Lambda

Lambda proxy integration [Info](#)

True

Paths

idReceta

Region

us-east-1

Lambda function

[getRecetasLambda](#)

Timeout

Default (29 seconds)

[Edit](#)

11. Desplegamos la API en el entorno de desarrollo

API Gateway > APIs > Resources - RecetasAPI [endpoint:75]

Successfully enabled CORS

API actions > [Deploy API](#)

Resources

[Create resource](#)

☒ / Recetas

☐ / {idReceta}

Methods

☒ DELETE

☐ GET

☐ OPTIONS

☐ PUT

Deploy API

Create or select a stage where your API will be deployed. You can use the deployment history to revert or change the active deployment for a stage. [Learn more](#)

Stage

New stage

Stage name

dev

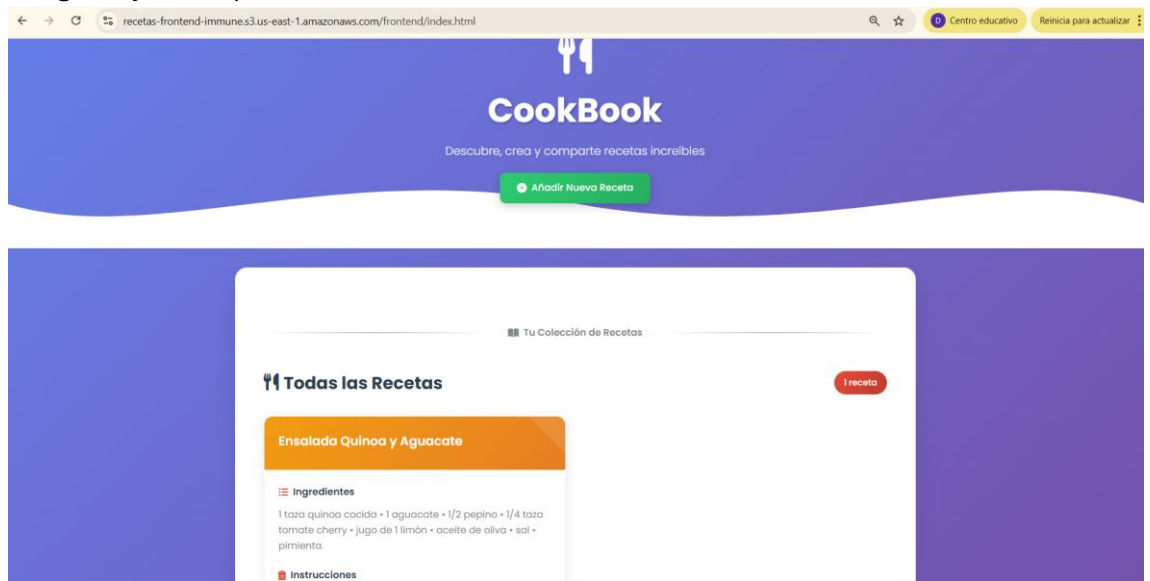
☒ A new stage will be created with the default settings. Edit your stage settings on the [Stage](#) page.

Deployment description

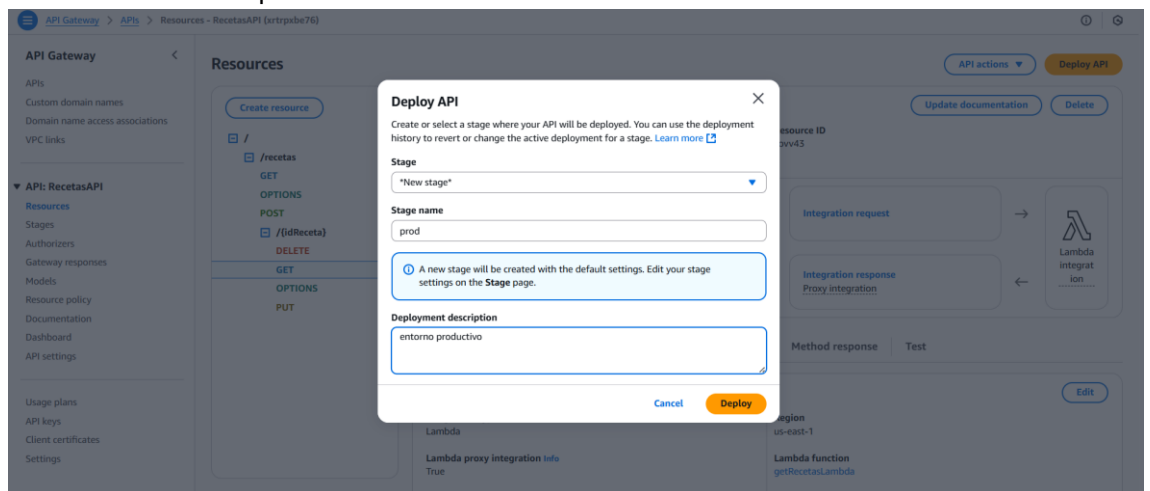
Entorno de desarrollo

[Cancel](#) [Deploy](#)

12. Programación de los archivos del frontend para que muestren una interfaz gráfica amigable y sea capaz de enviar las llamadas al backend.



13. Se realizan las pruebas necesarias y cuando está todo comprobado se despliega la API en el entorno productivo.



Con esta configuración, accediendo a la url del archivo index.html desde la web podremos ver el frontend e interactuar con el backend a través de la interfaz.

Se adjunta el link de la web:

<https://recetas-frontend-immune.s3.us-east-1.amazonaws.com/frontend/index.html>