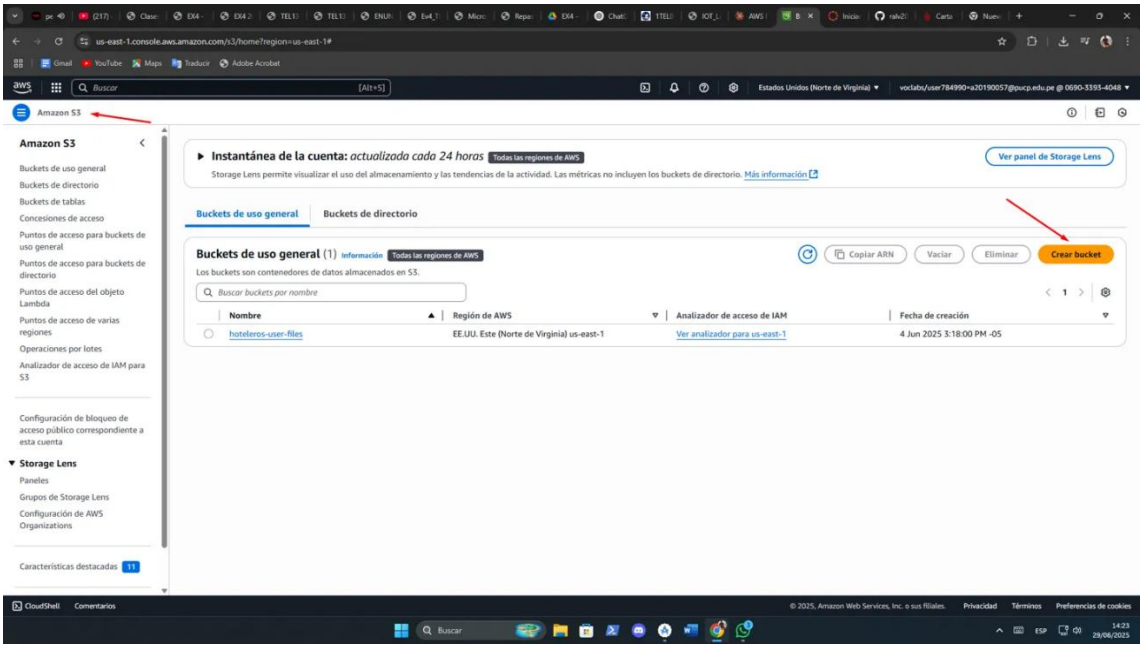
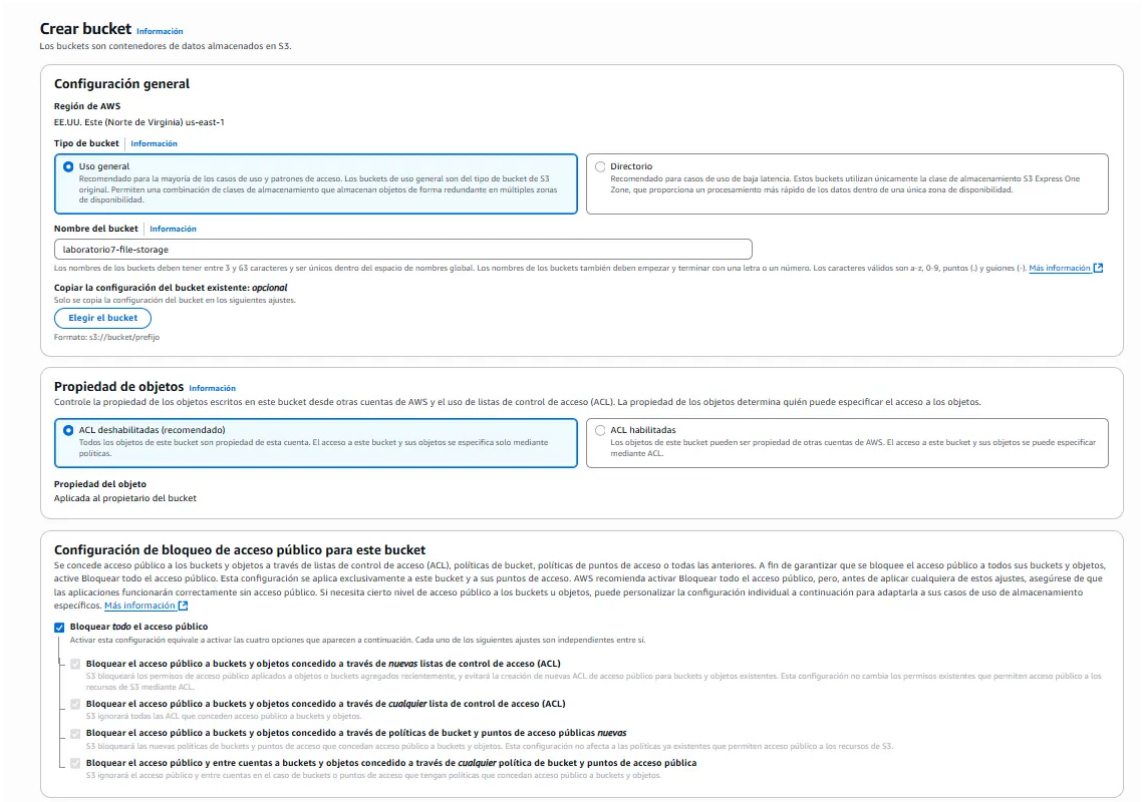
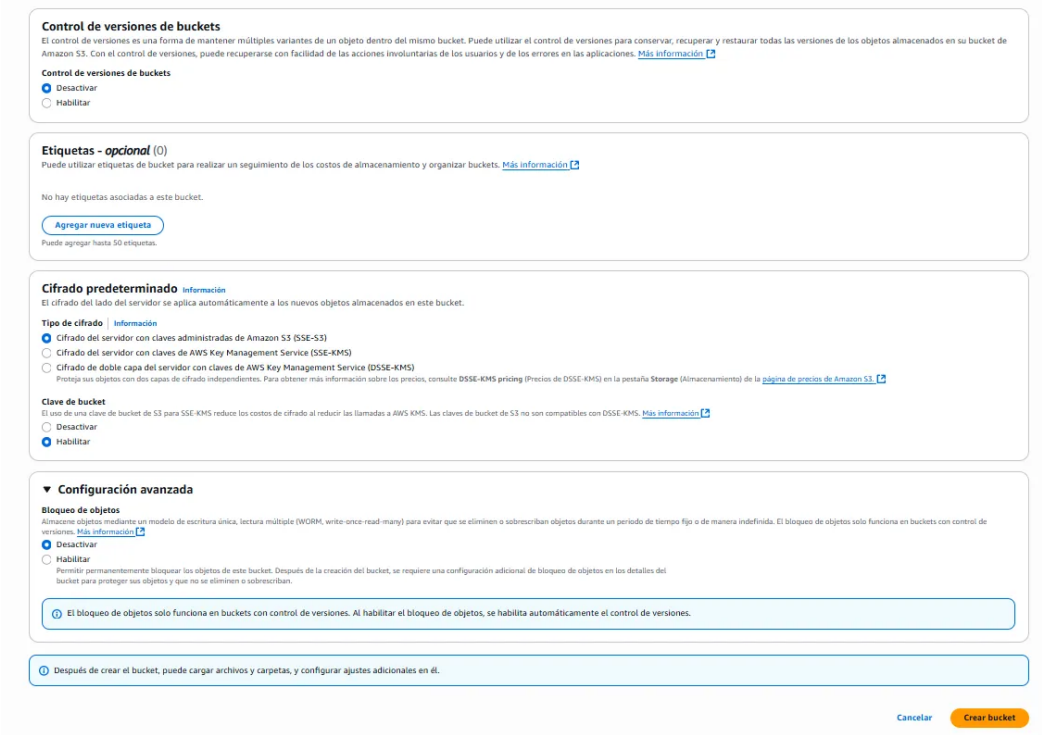
PRUEBAS DE CONFIGURACIÓN AWS LAMBDA Y AWS S3 - LABORATORIO 7 – RICARDO ALVARADO 20190057

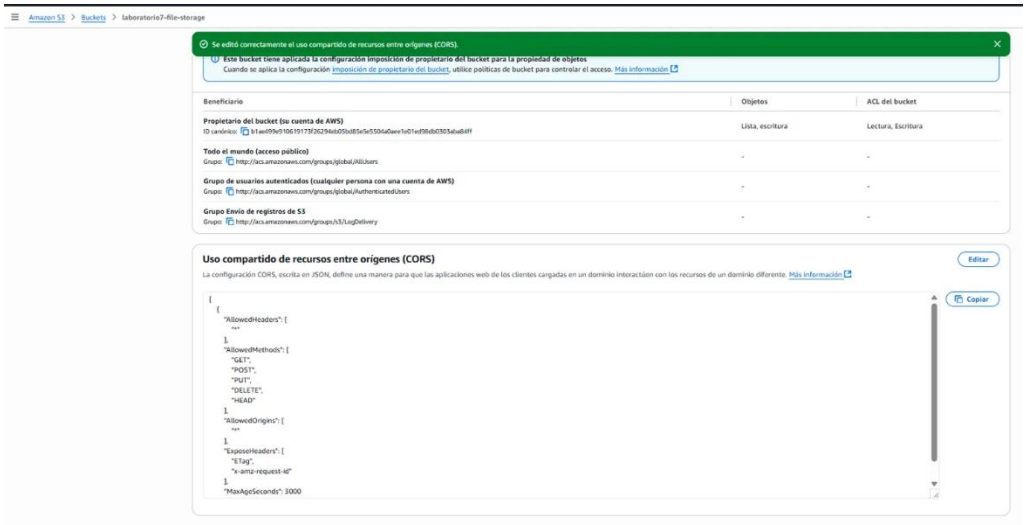
Paso 1:

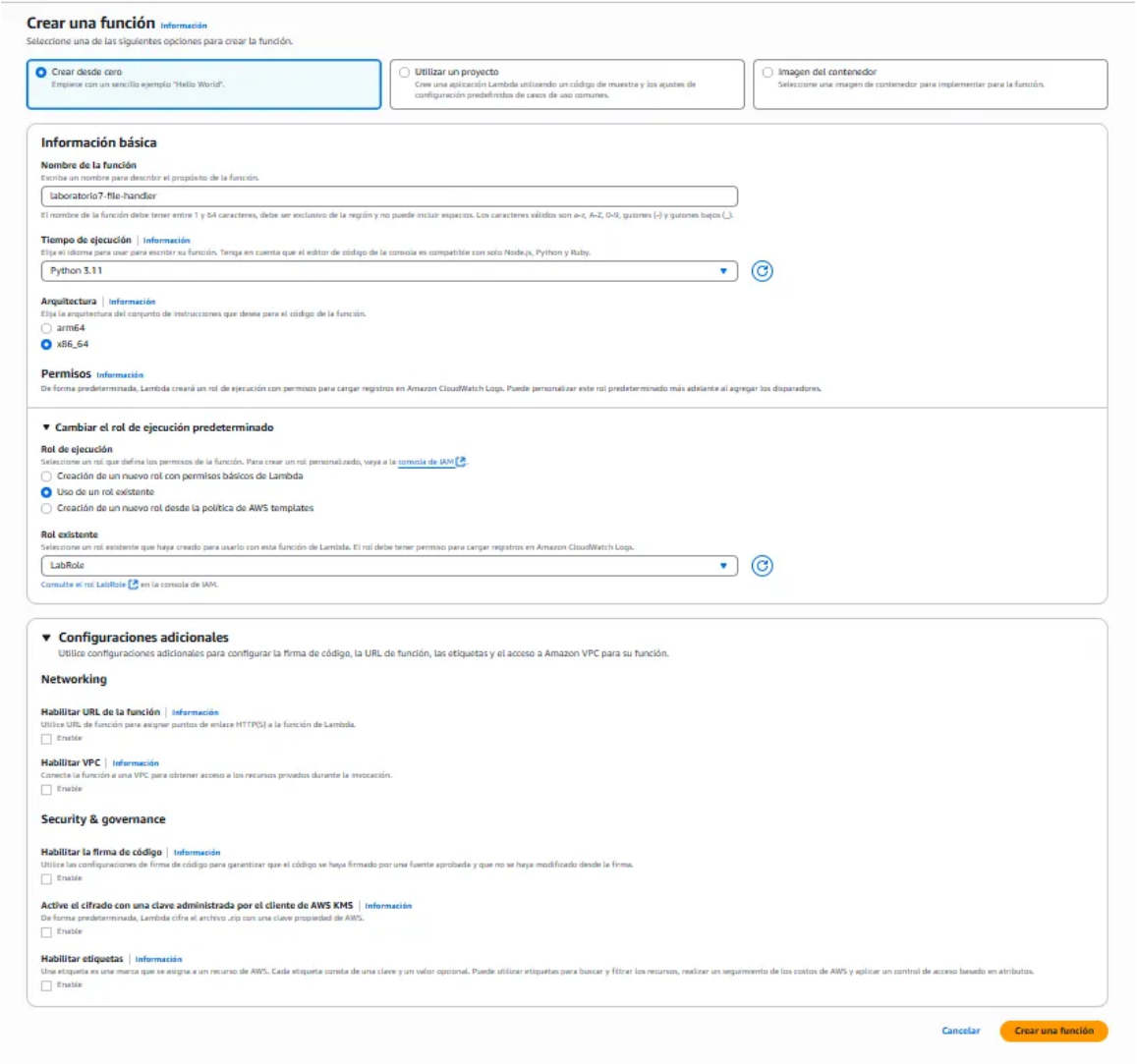




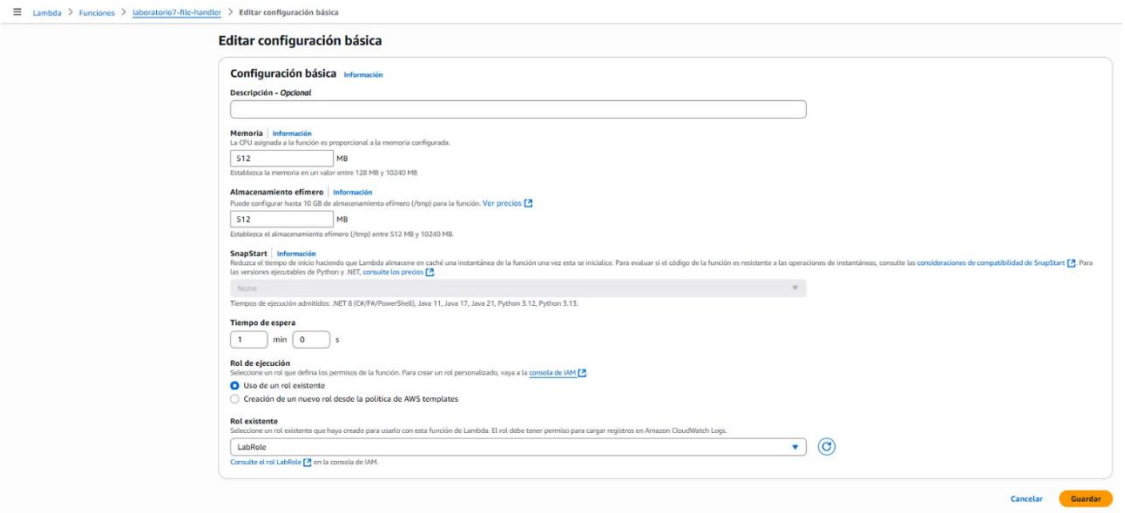

Control de versiones de buckets
El control de versiones es una forma de mantener múltiples variantes de un objeto dentro del mismo bucket. Puede utilizar el control de versiones para conservar, recuperar y restaurar todas las versiones de los objetos almacenados en su bucket de Amazon S3. Con el control de versiones, puede recuperarse con facilidad de las acciones involuntarias de los usuarios y de los errores en las aplicaciones. Más información
Control de versiones de buckets
Desactivar
Habilitar
Etiquetas - opcional (0)
Puede utilizar etiquetas de bucket para realizar un seguimiento de los costos de almacenamiento y organizar buckets. Más información
No hay etiquetas asociadas a este bucket.
Agregar nueva etiqueta
Puede agregar hasta 50 etiquetas.
Cifrado predeterminado Información
El cifrado del lado del servidor se aplica automáticamente a los nuevos objetos almacenados en este bucket.
Tipo de cifrado | Información
Cifrado del servidor con claves administradas de Amazon S3 (SSE-S3)
Cifrado del servidor con claves de AWS Key Management Service (SSE-KMS)
Cifrado de doble capa del servidor con claves de AWS Key Management Service (DSSE-KMS)
Proteja sus objetos con dos capas de cifrado independientes. Para obtener más información sobre los precios, consulte DSSE-KMS pricing (Precios de DSSE-KMS) en la pestaña Storage (Almacenamiento) de la página de precios de Amazon S3.
Clave de bucket
El uso de una clave de bucket de S3 para SSE-KMS reduce los costos de cifrado al reducir las llamadas a AWS KMS. Las claves de bucket de S3 no son compatibles con DSSE-KMS. Más información
Desactivar
Habilitar
Configuración avanzada
Bloqueo de objetos
Almacene objetos mediante un modelo de escritura única, lectura múltiple (WORM, write-once-read-many) para evitar que se eliminen o sobrescriban objetos durante un periodo de tiempo fijo o de manera indefinida. El bloqueo de objetos solo funciona en buckets con control de versiones. Más información
Desactivar
Habilitar
Permitir permanentemente bloquear los objetos de este bucket. Después de la creación del bucket, se requiere una configuración adicional de bloqueo de objetos en los detalles del bucket para proteger sus objetos y que no se eliminen o sobrescriban.
El bloqueo de objetos solo funciona en buckets con control de versiones. Al habilitar el bloqueo de objetos, se habilita automáticamente el control de versiones.
Después de crear el bucket, puede cargar archivos y carpetas, y configurar ajustes adicionales en él.
Cancelar
Crear bucket

Paso 2:

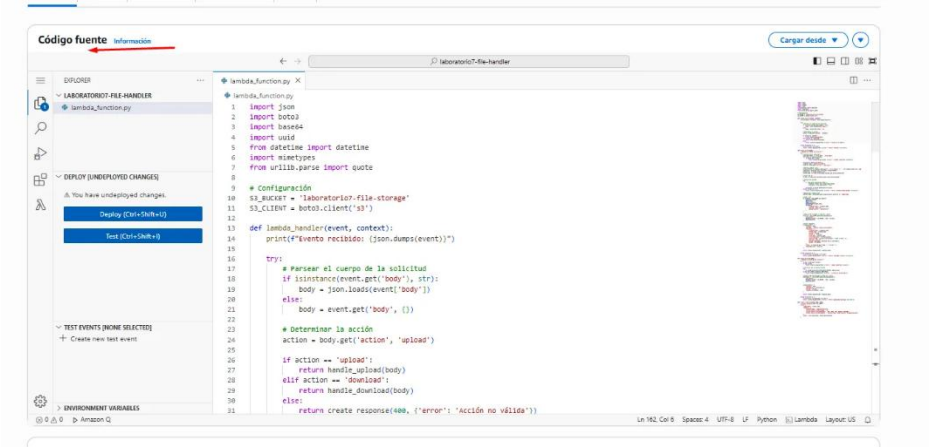

Amazon S3 > Buckets > laboratorio7-file-storage
Se editó correctamente el uso compartido de recursos entre orígenes (CORS).
Este bucket tiene aplicada la configuración imposición de propietario del bucket para la propiedad de objetos
Cuando se aplica la configuración imposición de propietario del bucket, utilice políticas de bucket para controlar el acceso. Más información
Beneficiario
Objetos
ACL del bucket
Propietario del bucket (su cuenta de AWS)
ID canónico:
Lista, escritura
Lectura, Escritura
Todo el mundo (acceso público)
Grupo: http://acs.amazonaws.com/groups/global/AllUsers
Grupo de usuarios autenticados (cualquier persona con una cuenta de AWS)
Grupo: http://acs.amazonaws.com/groups/global/AuthenticatedUsers
Grupo Envío de registros de S3
Grupo: http://acs.amazonaws.com/groups/s3/LogDelivery
Uso compartido de recursos entre orígenes (CORS)
Editar
La configuración CORS, escrita en JSON, define una manera para que las aplicaciones web de los clientes cargadas en un dominio interactúen con los recursos de un dominio diferente. Más información
Copiar
[
{
"AllowedHeaders": [
"*"
],
"AllowedMethods": [
"GET",
"POST",
"PUT",
"DELETE",
"HEAD"
],
"AllowedOrigins": [
"*"
],
"ExposeHeaders": [
"ETag",
"x-amz-request-id"
],
"MaxAgeSeconds": 3000


Crear una función Información
Seleccione una de las siguientes opciones para crear la función.
Crear desde cero
Empiece con un sencillo ejemplo "Hello World".
Utilizar un proyecto
Cree una aplicación Lambda utilizando un código de muestra y los ajustes de configuración predefinidos de casos de uso comunes.
Imagen del contenedor
Seleccione una imagen de contenedor para implementar para la función.
Información básica
Nombre de la función
Escriba un nombre para describir el propósito de la función.
laboratorio7-file-handler
El nombre de la función debe tener entre 1 y 64 caracteres, debe ser exclusivo de la región y no puede incluir espacios. Los caracteres válidos son a-z, A-Z, 0-9, guiones (-) y guiones bajos (_).
Tiempo de ejecución | Información
Elija el idioma para usar para escribir su función. Tenga en cuenta que el editor de código de la consola es compatible con solo Node.js, Python y Ruby.
Python 3.11
Arquitectura | Información
Elija la arquitectura del conjunto de instrucciones que desea para el código de la función.
arm64
x86_64
Permisos Información
De forma predeterminada, Lambda creará un rol de ejecución con permisos para cargar registros en Amazon CloudWatch Logs. Puede personalizar este rol predeterminado más adelante al agregar los disparadores.
Cambiar el rol de ejecución predeterminado
Rol de ejecución
Seleccione un rol que defina los permisos de la función. Para crear un rol personalizado, vaya a la consola de IAM.
Creación de un nuevo rol con permisos básicos de Lambda
Uso de un rol existente
Creación de un nuevo rol desde la política de AWS templates
Rol existente
Seleccione un rol existente que haya creado para usarlo con esta función de Lambda. El rol debe tener permiso para cargar registros en Amazon CloudWatch Logs.
LabRole
Consulte el rol LabRole en la consola de IAM.
Configuraciones adicionales
Utilice configuraciones adicionales para configurar la firma de código, la URL de función, las etiquetas y el acceso a Amazon VPC para su función.
Networking
Habilitar URL de la función | Información
Utilice URL de función para asignar puntos de enlace HTTP(S) a la función de Lambda.
Enable
Habilitar VPC | Información
Conecte la función a una VPC para obtener acceso a los recursos privados durante la invocación.
Enable
Security & governance
Habilitar la firma de código | Información
Utilice las configuraciones de firma de código para garantizar que el código se haya firmado por una fuente aprobada y que no se haya modificado desde la firma.
Enable
Active el cifrado con una clave administrada por el cliente de AWS KMS | Información
De forma predeterminada, Lambda cifra el archivo .zip con una clave propiedad de AWS.
Enable
Habilitar etiquetas | Información
Una etiqueta es una marca que se asigna a un recurso de AWS. Cada etiqueta consta de una clave y un valor opcional. Puede utilizar etiquetas para buscar y filtrar los recursos, realizar un seguimiento de los costos de AWS y aplicar un control de acceso basado en atributos.
Enable
Cancelar
Crear una función

Paso 3:


Lambda > Funciones > laboratorio7-file-handler > Editar configuración básica
Editar configuración básica
Configuración básica Información
Descripción - Opcional
Memoria | Información
La CPU asignada a la función es proporcional a la memoria configurada.
512
MB
Establezca la memoria en un valor entre 128 MB y 10240 MB.
Almacenamiento efímero | Información
Puede configurar hasta 10 GB de almacenamiento efímero (/tmp) para la función. Ver precios
512
MB
Establezca el almacenamiento efímero (/tmp) entre 512 MB y 10240 MB.
SnapStart | Información
Reduzca el tiempo de inicio haciendo que Lambda almacene en caché una instantánea de la función una vez esta se inicialice. Para evaluar si el código de la función es resistente a las operaciones de instantáneas, consulte las consideraciones de compatibilidad de SnapStart. Para las versiones ejecutables de Python y .NET, consulte los precios
None
Tiempos de ejecución admitidos: .NET 8 (C#/F#/PowerShell), Java 11, Java 17, Java 21, Python 3.12, Python 3.13.
Tiempo de espera
1
min
0
s
Rol de ejecución
Seleccione un rol que defina los permisos de la función. Para crear un rol personalizado, vaya a la consola de IAM
Uso de un rol existente
Creación de un nuevo rol desde la política de AWS templates
Rol existente
Seleccione un rol existente que haya creado para usarlo con esta función de Lambda. El rol debe tener permiso para cargar registros en Amazon CloudWatch Logs.
LabRole
Consulte el rol LabRole en la consola de IAM.
Cancelar
Guardar

Paso 4:



Función implementada:

```python
import json
import boto3
import base64
import uuid
from datetime import datetime
import mimetypes
from urllib.parse import quote

# Configuración
S3_BUCKET = 'laboratorio7-file-storage'
S3_CLIENT = boto3.client('s3')

def lambda_handler(event, context):
    print(f"Evento recibido: {json.dumps(event)}")

    try:
        # Parsear el cuerpo de la solicitud
        if isinstance(event.get('body'), str):
            body = json.loads(event['body'])
        else:
            body = event.get('body', {})

        # Determinar la acción
        action = body.get('action', 'upload')

        if action == 'upload':
            return handle_upload(body)
        elif action == 'download':
```

```
            return handle_download(body)
        else:
            return create_response(400, {'error': 'Acción
no válida'})

    except Exception as e:
        print(f"Error: {str(e)}")
        return create_response(500, {'error': f'Error
interno: {str(e)}'})

def handle_upload(body):
    """Maneja la subida de archivos"""
    try:
        # Validar datos requeridos
        required_fields = ['file_data', 'file_name']
        for field in required_fields:
            if field not in body:
                return create_response(400, {'error':
f'Campo requerido: {field}'})

        file_data = body['file_data']
        original_name = body['file_name']
        user_id = body.get('user_id', 'lab7_user')

        # Generar nombre único
        file_extension = original_name.split('.')[-
1].lower() if '.' in original_name else 'jpg'
        timestamp =
datetime.now().strftime('%Y%m%d_%H%M%S')
        unique_id = str(uuid.uuid4())[:8]
        stored_name =
f"lab7_{timestamp}_{unique_id}.{file_extension}"

        # Crear key S3
        s3_key =
f"laboratorio7/photos/{user_id}/{stored_name}"

        # Decodificar Base64
```

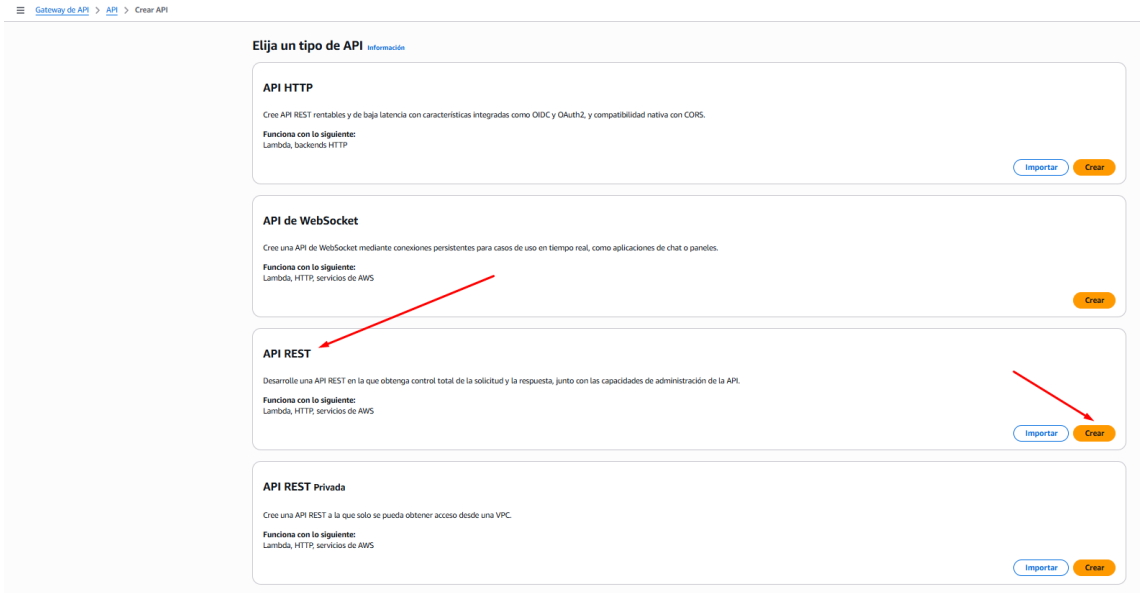
```
        try:
            if file_data.startswith('data:'):
                # Remover header 
                file_data = file_data.split(',')[1]

            file_bytes = base64.b64decode(file_data)
        except Exception as e:
            return create_response(400, {'error': f'Error
decodificando Base64: {str(e)}'})

        # Determinar content type
        content_type =
mimetypes.guess_type(original_name)[0] or 'image/jpeg'

        # Subir a S3
        s3_response = S3_CLIENT.put_object(
            Bucket=S3_BUCKET,
            Key=s3_key,
            Body=file_bytes,
            ContentType=content_type,
            Metadata={
                'original-name': original_name,
                'user-id': user_id,
                'upload-source': 'laboratorio7'
            }
        )

        # Generar URL firmada (válida por 1 hora)
        file_url = S3_CLIENT.generate_presigned_url(
            'get_object',
            Params={'Bucket': S3_BUCKET, 'Key': s3_key},
            ExpiresIn=2592000
        )

        # Crear respuesta
        response_data = {
            'success': True,
            'message': 'Archivo subido exitosamente',
```

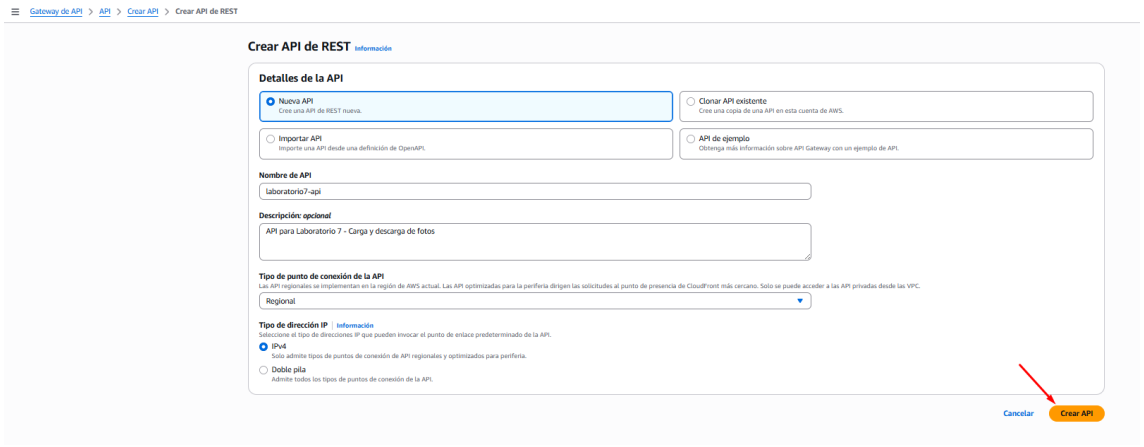
```
            'file_info': {
                'original_name': original_name,
                'stored_name': stored_name,
                's3_key': s3_key,
                'file_url': file_url,
                'file_type': content_type,
                'file_size_bytes': len(file_bytes),
                'file_size_mb': round(len(file_bytes) /
(1024 * 1024), 2),
                'user_id': user_id,
                'upload_timestamp':
datetime.now().isoformat(),
                'bucket': S3_BUCKET
            },
            'etag': s3_response.get('ETag',
'').strip('"'),
            'download_link': file_url
        }

        return create_response(200, response_data)

    except Exception as e:
        print(f"Error en upload: {str(e)}")
        return create_response(500, {'error': f'Error
subiendo archivo: {str(e)}'})

def handle_download(body):
    """Maneja la descarga de archivos"""
    try:
        s3_key = body.get('s3_key')
        if not s3_key:
            return create_response(400, {'error': 'Campo
requerido: s3_key'})

        # Verificar que el archivo existe
        try:
            S3_CLIENT.head_object(Bucket=S3_BUCKET,
Key=s3_key)
```

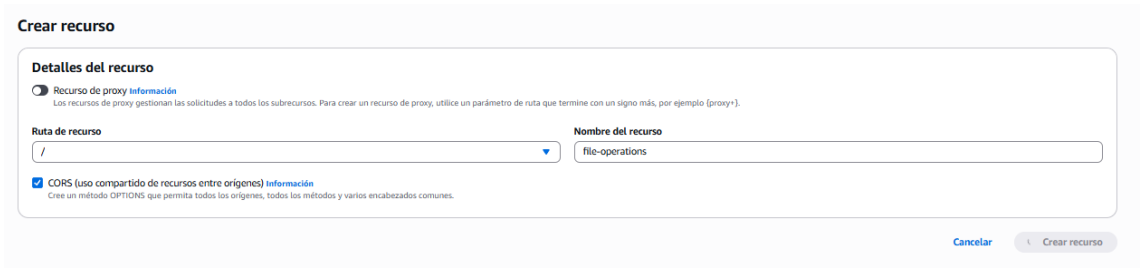
```
        except S3_CLIENT.exceptions.NoSuchKey:
            return create_response(404, {'error':
'Archivo no encontrado'})

        # Generar URL de descarga (válida por 1 hora)
        download_url = S3_CLIENT.generate_presigned_url(
            'get_object',
            Params={'Bucket': S3_BUCKET, 'Key': s3_key},
            ExpiresIn=2592000
        )

        response_data = {
            'success': True,
            'download_url': download_url,
            's3_key': s3_key,
            'expires_in_seconds': 2592000
        }

        return create_response(200, response_data)

    except Exception as e:
        print(f"Error en download: {str(e)}")
        return create_response(500, {'error': f'Error
generando descarga: {str(e)}'})

def create_response(status_code, body):
    """Crear respuesta HTTP con CORS"""
    return {
        'statusCode': status_code,
        'headers': {
            'Content-Type': 'application/json',
            'Access-Control-Allow-Origin': '*',
            'Access-Control-Allow-Methods': 'GET, POST,
PUT, DELETE, OPTIONS',
            'Access-Control-Allow-Headers': 'Content-
Type, Authorization, X-Requested-With'
        },
        'body': json.dumps(body, ensure_ascii=False)
```

```
}
```

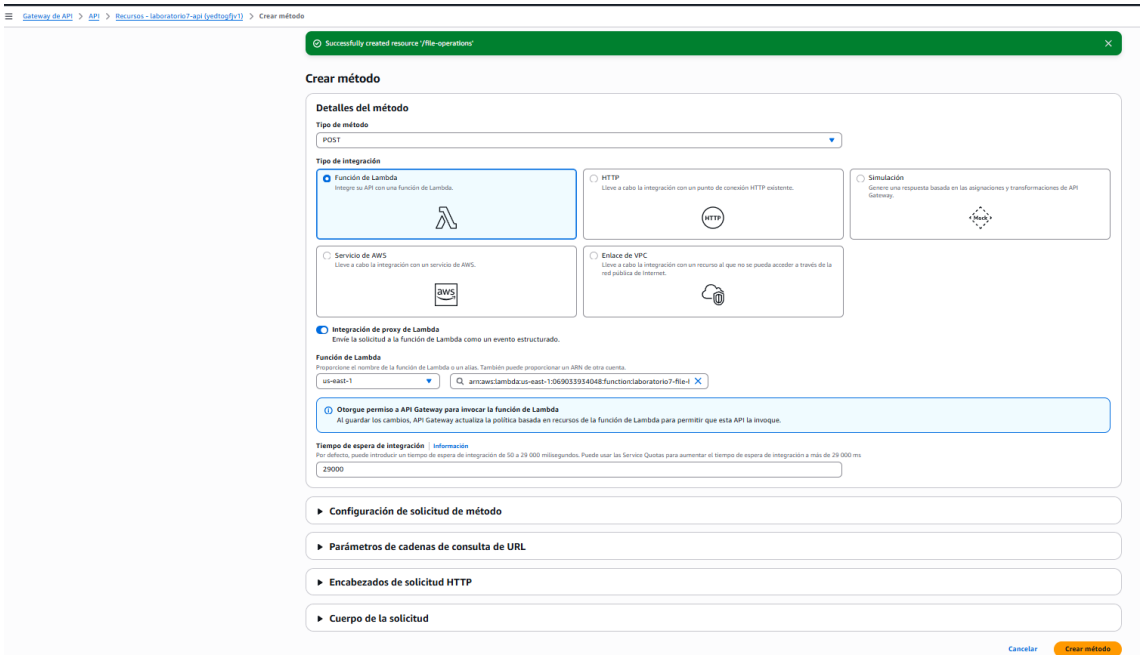
Paso 5:


Gateway de API > API > Crear API
Elija un tipo de API Información
API HTTP
Cree API REST rentables y de baja latencia con características integradas como OIDC y OAuth2, y compatibilidad nativa con CORS.
Funciona con lo siguiente:
Lambda, backends HTTP
Importar
Crear
API de WebSocket
Cree una API de WebSocket mediante conexiones persistentes para casos de uso en tiempo real, como aplicaciones de chat o paneles.
Funciona con lo siguiente:
Lambda, HTTP, servicios de AWS
Crear
API REST
Desarrolle una API REST en la que obtenga control total de la solicitud y la respuesta, junto con las capacidades de administración de la API.
Funciona con lo siguiente:
Lambda, HTTP, servicios de AWS
Importar
Crear
API REST Privada
Cree una API REST a la que solo se pueda obtener acceso desde una VPC.
Funciona con lo siguiente:
Lambda, HTTP, servicios de AWS
Importar
Crear

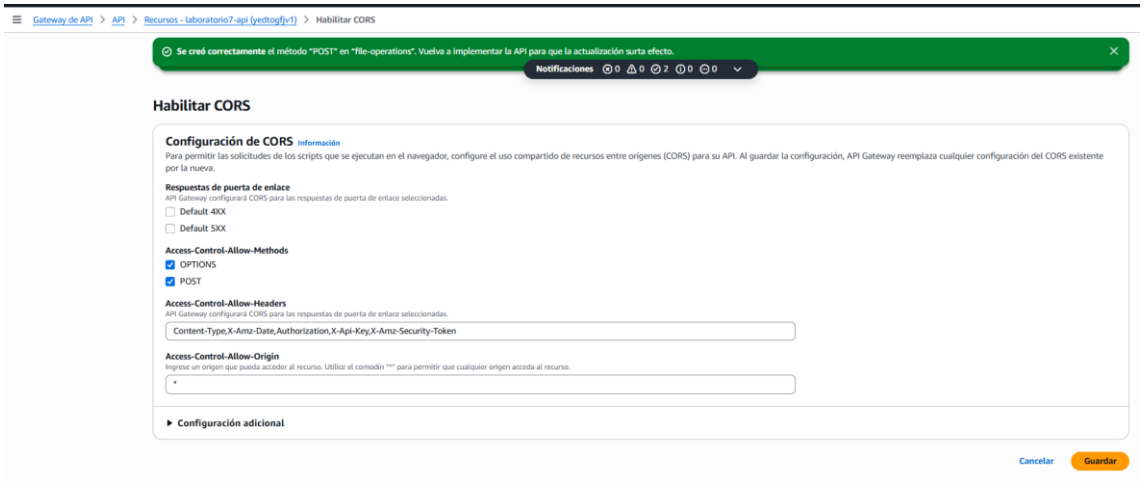

Gateway de API > API > Crear API > Crear API de REST
Crear API de REST Información
Detalles de la API
Nueva API
Cree una API de REST nueva.
Clonar API existente
Cree una copia de una API en esta cuenta de AWS.
Importar API
Importe una API desde una definición de OpenAPI.
API de ejemplo
Obtenga más información sobre API Gateway con un ejemplo de API.
Nombre de API
laboratorio7-api
Descripción: opcional
API para Laboratorio 7 - Carga y descarga de fotos
Tipo de punto de conexión de la API
Las API regionales se implementan en la región de AWS actual. Las API optimizadas para la periferia dirigen las solicitudes al punto de presencia de CloudFront más cercano. Solo se puede acceder a las API privadas desde las VPC.
Regional
Tipo de dirección IP | Información
Seleccione el tipo de direcciones IP que pueden invocar el punto de enlace predeterminado de la API.
IPv4
Solo admite tipos de puntos de conexión de API regionales y optimizados para periferia.
Doble pila
Admite todos los tipos de puntos de conexión de la API.
Cancelar
Crear API


Crear recurso
Detalles del recurso
Recurso de proxy Información
Los recursos de proxy gestionan las solicitudes a todos los subrecursos. Para crear un recurso de proxy, utilice un parámetro de ruta que termine con un signo más, por ejemplo {proxy+}.
Ruta de recurso
/
Nombre del recurso
file-operations
CORS (uso compartido de recursos entre orígenes) Información
Cree un método OPTIONS que permita todos los orígenes, todos los métodos y varios encabezados comunes.
Cancelar
Crear recurso

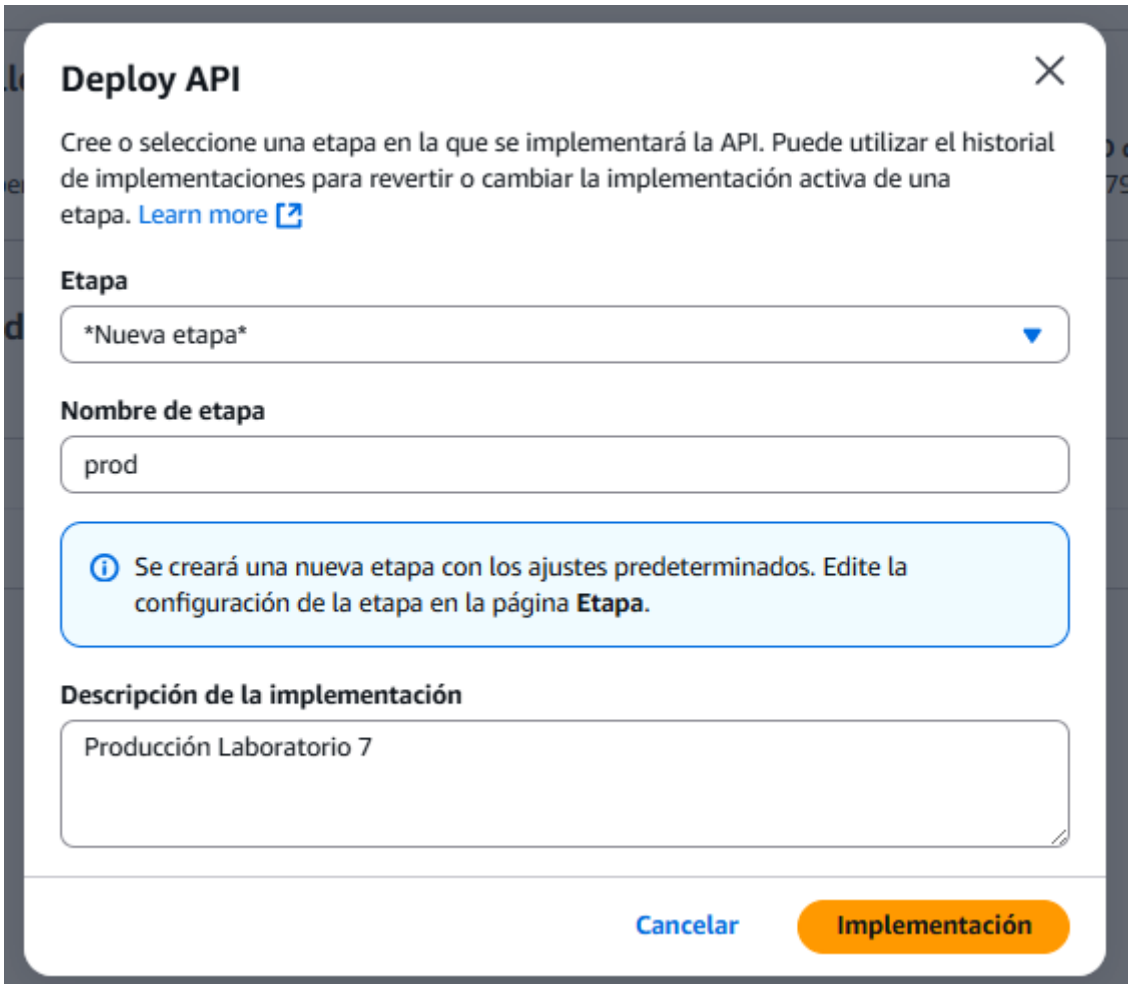
Paso 6:


Gateway de API > API > Recursos - laboratorio7-api (yedtogfjv1) > Crear método
Successfully created resource '/file-operations'
Crear método
Detalles del método
Tipo de método
POST
Tipo de integración
Función de Lambda
Integre su API con una función de Lambda.
HTTP
Lleve a cabo la integración con un punto de conexión HTTP existente.
Simulación
Genere una respuesta basada en las asignaciones y transformaciones de API Gateway.
Servicio de AWS
Lleve a cabo la integración con un servicio de AWS.
Enlace de VPC
Lleve a cabo la integración con un recurso al que no se pueda acceder a través de la red pública de Internet.
Integración de proxy de Lambda
Envíe la solicitud a la función de Lambda como un evento estructurado.
Función de Lambda
Proporcione el nombre de la función de Lambda o un alias. También puede proporcionar un ARN de otra cuenta.
us-east-1
arn:aws:lambda:us-east-1:069033934048:function:laboratorio7-file-
Otorgue permiso a API Gateway para invocar la función de Lambda
Al guardar los cambios, API Gateway actualiza la política basada en recursos de la función de Lambda para permitir que esta API la invoque.
Tiempo de espera de integración | Información
Por defecto, puede introducir un tiempo de espera de integración de 50 a 29 000 milisegundos. Puede usar las Service Quotas para aumentar el tiempo de espera de integración a más de 29 000 ms
29000
Configuración de solicitud de método
Parámetros de cadenas de consulta de URL
Encabezados de solicitud HTTP
Cuerpo de la solicitud
Cancelar
Crear método

Paso 7:

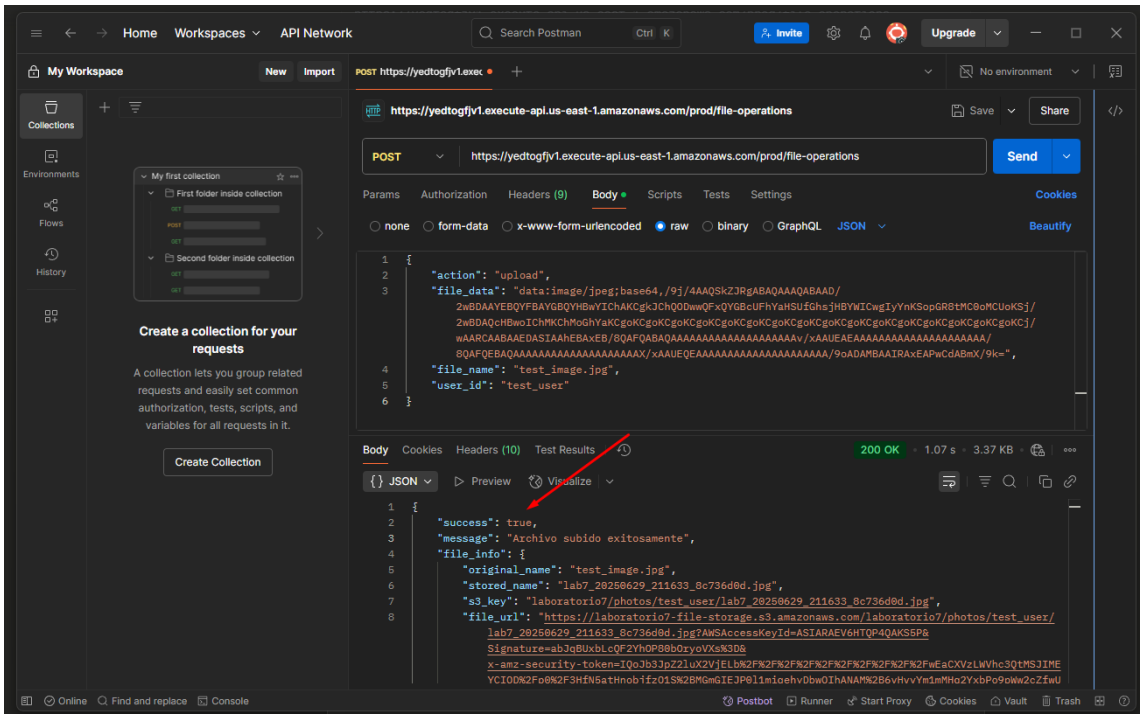

Gateway de API > API > Recursos - laboratorio7-api (yedtogfjv1) > Habilitar CORS
Se creó correctamente el método "POST" en "file-operations". Vuelva a implementar la API para que la actualización surta efecto.
Notificaciones
Habilitar CORS
Configuración de CORS Información
Para permitir las solicitudes de los scripts que se ejecutan en el navegador, configure el uso compartido de recursos entre orígenes (CORS) para su API. Al guardar la configuración, API Gateway reemplaza cualquier configuración del CORS existente por la nueva.
Respuestas de puerta de enlace
API Gateway configurará CORS para las respuestas de puerta de enlace seleccionadas.
Default 4XX
Default 5XX
Access-Control-Allow-Methods
OPTIONS
POST
Access-Control-Allow-Headers
API Gateway configurará CORS para las respuestas de puerta de enlace seleccionadas.
Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token
Access-Control-Allow-Origin
Ingrese un origen que pueda acceder al recurso. Utilice el comodín "*" para permitir que cualquier origen acceda al recurso.
*
Configuración adicional
Cancelar
Guardar

Paso 8:



Paso 9:

Verificación con Postman:



URL completa del endpoint:
https://yedtogfjv1.execute-api.us-east-1.amazonaws.com/prod/file-operations