
Documento Específico de la Célula 3

Image Handler para Imágenes de Gadgets – Acme Corp

Proyecto: Arquitecturas Serverless en AWS
Célula: 3
Integrantes: Alejandro Granados, Rodrigo Pulido
Versión: 1.0
Organización: Universidad La Salle – Ingeniería
Arquitectura Asignada: Image Handler serverless
Caso de uso: Gestión de imágenes de productos (“gadgets”) para Acme

---

Índice

14. Criterios de Evaluación
15. Control de Cambios

---

1. Introducción

La Célula 3 es responsable de construir un sistema **serverless para manejar imágenes de gadgets** pertenecientes al catálogo de productos de **Acme Corp**. Esta arquitectura permite administrar fotografías de productos, procesar versiones optimizadas y generar variaciones (thumbnails, preview, versiones web-friendly), utilizando servicios administrados de AWS.

Este documento formal describe los lineamientos técnicos y organizacionales aplicables, siguiendo el **Documento Maestro**, el **Well-Architected Framework** y las políticas institucionales de Ciberseguridad.

---

2. Objetivo del Módulo

Diseñar e implementar un servicio serverless que permita:

- Recibir imágenes de gadgets de Acme
- Procesarlas (resize, optimización, estandarización)
- Clasificarlas mediante metadatos
- Guardarlas en repositorios cifrados
- Exponer APIs seguras para consulta y administración
- Integrarse con el catálogo de productos

---

3. Alcance

Incluye:

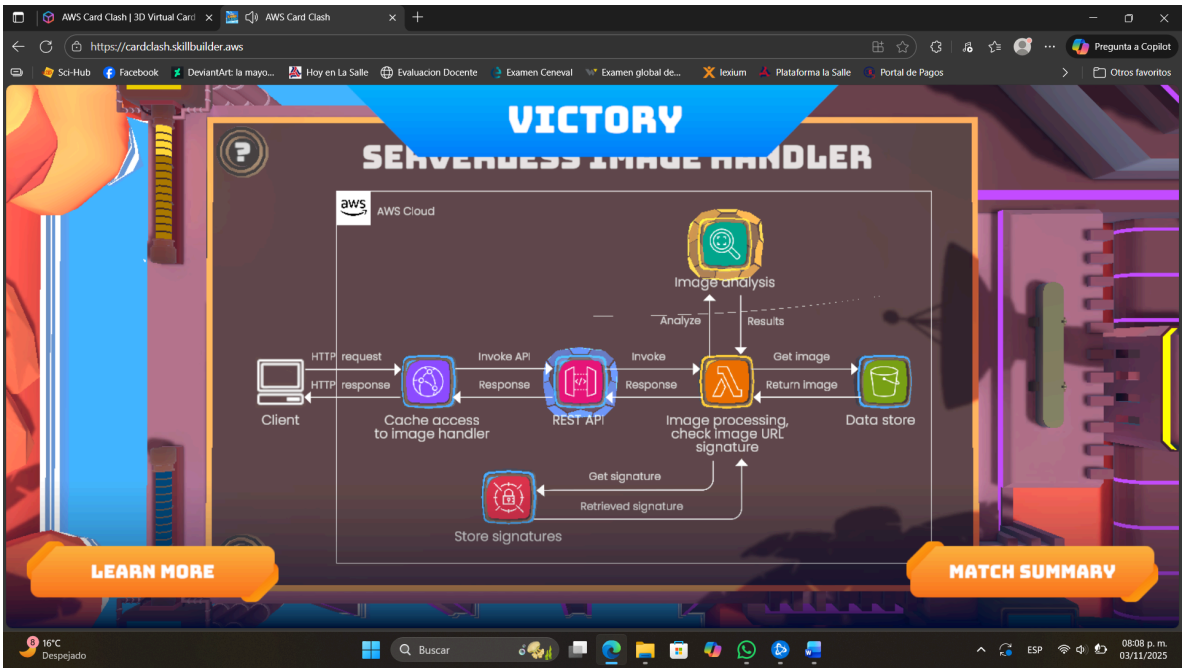
- API para subir imágenes
- Procesamiento automático mediante triggers
- Generación de múltiples versiones de cada imagen
- Registro de metadatos en DynamoDB
- Exposición de endpoints autenticados con Cognito
- Pipeline CI/CD completo
- Despliegue en sandbox → pre-producción → producción

No incluye:

- Administración de usuarios de negocio
- Backoffice visual (solo APIs)
- Procesamiento pesado no soportado por Lambda

---

4. Arquitectura General



El flujo principal es:

1. Cliente autenticado solicita una **URL de carga** para subir una imagen
2. Imagen se sube a **Amazon S3 (bucket de entrada)**
3. Evento de S3 dispara una **Lambda de procesamiento**
4. Lambda genera:
    - Versión original validada
    - Thumbnail
    - Versión optimizada
5. Guarda las versiones en un **bucket S3 procesado**
6. Registra metadatos en **DynamoDB**
7. API Gateway permite:
    - Listar imágenes
    - Consultar metadatos
    - Obtener URLs firmadas
8. Todo está protegido con Cognito y cifrado con KMS

---

5. Componentes AWS Utilizados

- **Amazon S3**
    - Bucket de entrada (`acme-gadgets-raw`)
    - Bucket procesado (`acme-gadgets-processed`)
- **AWS Lambda (Python)**
    - Procesamiento y optimización de imágenes

- **Amazon DynamoDB**
    - Tabla `GadgetImages`
    - PK: `gadgetId`
    - SK: `imageId`
- **API Gateway**
    - Endpoints REST
    - Integrado con Cognito Authorizer
- **Amazon Cognito**
    - User Pool
    - Roles asociados para acceso a APIs
- **AWS KMS**
    - Cifrado de buckets
    - Cifrado de DynamoDB
    - Cifrado de variables de entorno
- **VPC + Endpoints + Zona de inspección**
    - Lambdas solo en subredes privadas
- **CodePipeline / CodeBuild**
    - Pipeline CI/CD

---

6. Lineamientos Técnicos Específicos

Requisitos mínimos

- Todas las imágenes deben procesarse automáticamente

- Solo usuarios autenticados pueden interactuar con la API
- Todos los metadatos deben estar estandarizados
- Cada imagen debe mantener un identificador único global
- El procesamiento debe ser idempotente
- Las URLs de acceso deben ser firmadas y temporales

Estándares sugeridos

- Dimensiones sugeridas:
    - Thumbnail: 256 px
    - Preview: 1024 px
- Formato recomendado: `.jpg` o `.webp`

---

7. IaC: Requerimientos CloudFormation

La plantilla deberá incluir:

- Buckets S3 con SSE-KMS
- Funciones Lambda con:
    - VPC config
    - Roles definidos por Ciberseguridad
    - Variables de entorno cifradas
- DynamoDB con KMS
- API Gateway con Cognito Authorizer
- Outputs obligatorios:
    - URLs del API

    - Nombre de tablas/buckets
    - ARN de Lambdas
- Parámetros para:
    - ID de User Pool
    - ID de VPC
    - Subnets privadas
    - KMS key ARN

---

8. Pipeline CI/CD

El pipeline debe incluir:

1. **Source** (GitHub)
2. **Build** (CodeBuild)
    - Validar template
    - Procesar dependencias Python
    - Empaquetar Lambdas
3. **Deploy** (CloudFormation)
    - Sandbox
    - Pre-producción
    - Producción

---

9. Seguridad, IAM y Cifrado

Obligatorio:

- Cifrado con **KMS** en todo (buckets, tablas, logs)
- Acceso sólo mediante **VPC Endpoints**
- Lambdas en subredes privadas
- APIs protegidas por Cognito
- Roles IAM proporcionados por Ciberseguridad
- Validación estricta de formato de imagen

---

10. Datos de Prueba (GenAI)

La célula debe generar **al menos 50 imágenes sintéticas** de gadgets de Acme.
Se recomienda utilizar **Kiro** o herramientas equivalentes.

Cada imagen debe incluir metadatos generados:

- gadgetId
- nombre
- categoría
- resolución original
- formato

---

11. Pruebas funcionales (curl)

Al menos los siguientes comandos deben funcionar:

Obtener token JWT

```
curl -X POST https://<cognito-domain>/oauth2/token \
  -d "grant_type=password" \
  -d "username=<usuario>" \
  -d "password=<password>" \
  -d "client_id=<client-id>"
```

Obtener lista de imágenes

```
curl -H "Authorization: Bearer <jwt>" \
  https://<api-id>.execute-api.amazonaws.com/prod/images
```

Obtener URL firmada para subir imagen

```
curl -H "Authorization: Bearer <jwt>" \
  https://<api-id>.execute-api.amazonaws.com/prod/upload-url \
  -d '{"gadgetId": "A123"}'
```

---

12. Diagrama de Arquitectura

Debe ser elaborado en **draw.io** y contener:

- S3 (raw y processed)
- Lambda Processor
- DynamoDB GadgetImages
- API Gateway + Cognito
- VPC, endpoints, zona de inspección
- KMS keys

---

13. Entregables de la Célula

1. Repositorio GitHub completo
2. Pipeline CI/CD funcional
3. Plantillas CloudFormation
4. Código Python

5. Datos generados por GenAI
6. Pruebas curl
7. Diagrama draw.io
8. Backlog / historias de usuario
9. Estimación de costos

---

14. Criterios de Evaluación

- Calidad del IaC
- Pipeline automatizado
- Seguridad aplicada correctamente
- Funcionamiento completo del flujo de imágenes
- Calidad del diagrama
- Pruebas funcionales
- Gestión del backlog
- Orden, claridad y documentación

---

15. Control de Cambios

Versión	Fecha	Descripción	Autor
1.0	2025	Primera versión del documento de la Célula 3	Equipo de Arquitectura

---