

---

# Documento de Arquitectura – Célula 1

## Proyecto: Reconocimiento de Personajes de Caricaturas

**Empresa ficticia:** *Acme Inc.*

**Célula:** 1

**Integrantes:** Alberto Nuño, Axel Cerderia

**Fecha:** 2025

**Versión:** 1.0

---

## Índice

1. Introducción
2. Escenario Acme Inc.
3. Objetivo de la Arquitectura
4. Alcance
5. Arquitectura Propuesta
  - 5.1 Componentes Principales
  - 5.2 Flujo de Datos
6. Lineamientos Técnicos Obligatorios
  - 6.1 Infraestructura como Código
  - 6.2 Python y Generación de Datos GenAI
  - 6.3 API Gateway + Cognito
  - 6.4 Seguridad (VPC, KMS, Endpoints, Inspección)
  - 6.5 Pipeline CI/CD
7. Entregables
8. Pruebas con Curl
9. Historias de Usuario

10. Estimación de Costos (Guía)

11. Control de Cambios

---

# 1. Introducción

Este documento describe la arquitectura, requerimientos, entregables y pruebas para el desarrollo asignado a la **Célula 1**, el cual forma parte del proyecto institucional del curso *Cómputo en la Nube – AWS* de la Universidad La Salle.

La solución sigue estrictamente los lineamientos del Documento Maestro, incluyendo el uso de CloudFormation, Python, Cognito, KMS, VPC Endpoints, zona de inspección, y un pipeline CI/CD completo.

---

# 2. Escenario Acme Inc.

La empresa ficticia **Acme Inc.**, dedicada a productos de entretenimiento y animación, desea construir un servicio interno que permita **reconocer personajes de caricaturas** dentro de imágenes cargadas por sus equipos creativos.

Actualmente, el área de animación recibe miles de bocetos, thumbnails y frames pre-renderizados. Necesitan un sistema automático que pueda:

- Identificar a qué personaje pertenece cada imagen
- Acumular estadísticas de uso
- Organizar archivos por personaje
- Permitir a artistas consultar resultados de manera segura

**Acme Inc. quiere un servicio serverless**, fácil de desplegar y escalar, que pueda analizar estas imágenes sin necesidad de mantener servidores.

---

# 3. Objetivo de la Arquitectura

Diseñar y desplegar una solución serverless que:

- Reciba imágenes de caricaturas mediante un endpoint seguro
  - Use Amazon Rekognition para identificar el personaje
  - Almacene resultados y metadatos en DynamoDB
  - Cifre toda la información con KMS
  - Exponga APIs autenticadas con Amazon Cognito
  - Proporcione un pipeline completo CI/CD usando CodePipeline
  - Pueda desplegarse en sandbox, pre-producción y producción
  - Documente pruebas con curl
- 

## 4. Alcance

### Incluye

- Desarrollo en Python
- CloudFormation completo
- Integración con Rekognition
- API Gateway + Lambda + DynamoDB + S3
- Cognito para autenticación
- Cifrado con AWS KMS
- VPC Endpoints y zona de inspección
- Pipeline CI/CD
- Datos generados con GenAI (mínimo 50)
- Diagrama en draw.io

- Pruebas documentadas

## No incluye

- Entrenamiento de modelos ML propios
- Interfaces gráficas avanzadas (más allá de pruebas con curl)
- Integraciones externas no definidas

## 5. Arquitectura Propuesta



### 5.1 Componentes Principales

- **Amazon S3**
  - Bucket para imágenes cargadas
  - Cifrado SSE-KMS obligatorio
- **Amazon Rekognition**
  - Identificación de personajes de caricatura en la imagen

- **AWS Lambda (Python)**
  - Procesamiento principal
  - Integración con Rekognition
  - Escritura en DynamoDB
- **Amazon DynamoDB**
  - Tabla de resultados por:
    - ImageId
    - CharacterName
    - Confidence
    - Timestamp
    - Metadata
- **Amazon API Gateway**
  - Expone endpoints seguros para:
    - Generar URL de carga
    - Consultar resultados
    - Registrar análisis
- **Amazon Cognito**
  - Autenticación y autorización de usuarios
- **AWS KMS**
  - Claves CMK para:
    - S3
    - DynamoDB
    - Logs

- Secrets
  - Variables de entorno
  - **VPC + Endpoints + Zona de inspección**
    - Lambdas en subredes privadas
    - VPC Endpoints para S3, DynamoDB, Logs, Secrets Manager
  - **CodePipeline + CodeBuild + CloudFormation**
    - Pipeline automático sandbox → pre-producción → producción
- 

## 5.2 Flujo de Datos

1. Usuario autenticado con Cognito solicita URL de carga.
  2. Lambda genera una **presigned URL** y la regresa via API Gateway.
  3. El usuario sube la imagen a S3.
  4. Evento de S3 dispara una Lambda.
  5. Lambda analiza la imagen con Rekognition.
  6. Lambda guarda resultado en DynamoDB.
  7. El usuario consulta el resultado mediante API Gateway + token JWT.
  8. DynamoDB regresa información procesada.
- 

## 6. Lineamientos Técnicos Obligatorios

### 6.1 Infraestructura como Código

- Todo via CloudFormation.

- Sin cambios manuales en ambientes objetivo/producción.

## 6.2 Python y Datos GenAI

- Código en Python.
- Datos sintéticos generados (mín. 50), simulando personajes.

## 6.3 API Gateway + Cognito

- Endpoint seguro con JWT.
- Cognito administra usuarios.

## 6.4 Seguridad

- Lambdas en VPC (subredes privadas).
- Endpoints: S3, DynamoDB, Logs, Secrets Manager.
- Cifrado KMS en todos los puntos.
- Logs centralizados.

## 6.5 Pipeline CI/CD

- GitHub → CodePipeline → CodeBuild → CloudFormation
- Validación automática de plantillas
- Despliegue a sandbox → objetivo → producción

---

# 7. Entregables

1. Repositorio GitHub con `src/`, `iac/`, `data/`, `buildspec.yml`.

2. Pipeline funcional.
  3. Despliegue en sandbox, objetivo y producción.
  4. Diagrama draw.io.
  5. Datos generados con IA.
  6. Historias de usuario.
  7. Costos estimados.
  8. Pruebas curl documentadas.
- 

## 8. Pruebas con Curl

### 1. Obtener token JWT (Cognito):

```
curl -X POST https://<cognito-domain>/oauth2/token \  
-H "Content-Type: application/x-www-form-urlencoded" \  
-d "grant_type=password&client_id=<client>&username=user&password=pass"
```

### 2. Solicitar URL de carga:

```
curl -H "Authorization: Bearer <JWT>" \  
https://<api>/prod/get-upload-url
```

### 3. Consultar resultado:

```
curl -H "Authorization: Bearer <JWT>" \  
https://<api>/prod/result?imageId=<id>
```

---

## 9. Historias de Usuario

### HU-01 – Cargar imagen para análisis

**Como** artista de Acme Inc.

**Quiero** subir una imagen a la API

**Para** identificar a qué personaje pertenece.



## HU-02 – Consultar resultados

Como artista

**Quiero** ver qué personaje fue detectado

**Para** organizar mis archivos del proyecto.

## HU-03 – Acceso seguro

Como usuario autenticado

**Quiero** que sólo usuarios autorizados puedan consultar resultados

**Para** proteger contenido confidencial.

---

# 10. Estimación de Costos (Guía)

El análisis debe contemplar:

- S3 (almacenamiento + PUT)
  - Lambda (tiempo de ejecución)
  - Rekognition (Request-based)
  - DynamoDB (lecturas/escrituras)
  - CloudFront si aplica
  - CodePipeline / CodeBuild
  - KMS (uso de claves)
  - VPC Endpoints
- 

# 11. Control de Cambios

Versión	Fecha	Cambios	Autor
1.0	2025	Primera versión del Documento de Célula	Arquitecto del Proyecto
	a	1	

---

