
Documento Específico de la Célula 5

Procesamiento de Capturas de Pantalla del Videojuego “Pinky y Cerebro” – Acme Corp

Proyecto: Arquitecturas Serverless en AWS
Célula: 5
Integrantes: David Elizarraz, Diego Sastre, Héctor González, Juan Manuel Palafox
Versión: 1.0
Organización: Universidad La Salle – Facultad de Ingeniería
Arquitectura Asignada: Procesamiento serverless de imágenes (screenshots)
Caso de uso: Análisis y procesamiento de capturas de pantalla del videojuego *Pinky y Cerebro* desarrollado por Acme

---

Índice

1. Introducción
2. Objetivo del Módulo
3. Alcance
4. Arquitectura General
5. Características del Escenario Pinky y Cerebro
6. Componentes AWS Utilizados
7. Lineamientos Técnicos Específicos
8. IaC: Requerimientos CloudFormation
9. Pipeline CI/CD
10. Seguridad, IAM y Cifrado
11. Datos de Prueba (GenAI)

12. Pruebas funcionales (curl)
13. Diagrama de Arquitectura
14. Entregables de la Célula
15. Criterios de Evaluación
16. Control de Cambios

---

1. Introducción

La Célula 5 desarrolla el sistema serverless que procesa las capturas de pantalla generadas por los jugadores del videojuego **Pinky y Cerebro**, propiedad de Acme Corp. Estas imágenes se utilizan para:

- Analizar el desempeño del jugador
- Detectar elementos de la interfaz del juego
- Identificar escenas clave
- Construir un repositorio administrado de evidencia visual
- Facilitar análisis posteriores por áreas de QA, marketing o telemetría

El sistema se implementará usando servicios administrados de AWS y seguirá estrictamente los lineamientos del Documento Maestro.

---

2. Objetivo del Módulo

Construir un **servicio serverless robusto, seguro y escalable** que permita:

- Recibir capturas de pantalla desde el cliente del juego
- Procesarlas automáticamente
- Generar versiones optimizadas

- Clasificarlas con metadatos
- Detectar elementos relevantes (opcional: analítica con Rekognition)
- Almacenarlas en repositorios cifrados
- Exponer APIs seguras para consulta e integración con otros sistemas

---

3. Alcance

Incluye:

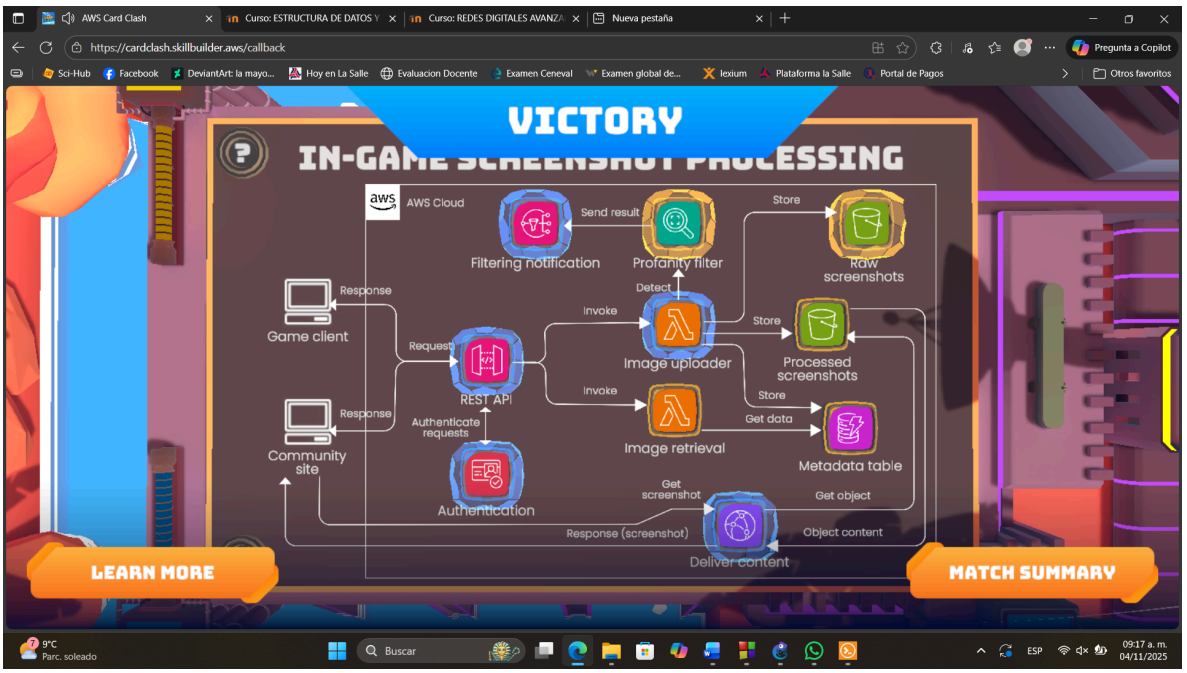
- API autenticada para registrar el jugador y solicitar URLs de subida
- Procesamiento de imágenes mediante Lambda
- Clasificación automática de screenshots
- Generación de thumbnails y optimización
- Registro estructurado de metadatos
- Pipeline CI/CD
- Integración con ambientes sandbox, pre-producción y producción
- Pruebas funcionales mediante curl

No incluye:

- Cliente gráfico del juego
- Algoritmos avanzados de visión artificial fuera de Rekognition
- Dashboard administrativo

---

4. Arquitectura General



El flujo principal contempla:

1. El cliente del videojuego solicita autenticación (Cognito)
2. Obtiene una URL firmada para subir la captura
3. El cliente sube la screenshot a **S3 (bucket raw)**
4. S3 dispara la **Lambda Processor**
5. Lambda realiza:
    - Validación del formato
    - Generación de versión optimizada
    - Generación de thumbnail
    - Análisis opcional con Rekognition (objetos/escenas/texto)
6. Lambda almacena los resultados en **S3 processed**
7. Lambda guarda metadatos en **DynamoDB**
8. API Gateway expone:
    - Obtener capturas de un jugador

    - Consultar metadatos
    - Solicitar nuevas URLs firmadas
9. Todo el flujo está asegurado con Cognito + KMS + VPC Endpoints

---

5. Características del Escenario “Pinky y Cerebro”

Las capturas provienen de un videojuego retro-modernizado desarrollado por Acme. Las imágenes contienen:

- Personajes: Pinky, Cerebro, enemigos y aliados
- HUD del juego (puntos, energía, nivel)
- Elementos de la escena del mundo (laboratorios, trampas, etc.)

El sistema debe:

- Ser capaz de **identificar la escena principal**
- Registrar el **nivel del jugador** (si se detecta en HUD)
- Asociar cada screenshot con:
    - ID del jugador
    - Timestamp del evento
    - Nivel/nivel estimado (opcional con análisis de texto u objetos)
    - Resultado del análisis

---

6. Componentes AWS Utilizados

- **Amazon S3**
    - `acme-pinky-raw`
    - `acme-pinky-processed`
- **AWS Lambda (Python)**
    - Procesamiento de imágenes
    - Extracción de metadatos
- **Amazon Rekognition**
    - Detección opcional de objetos / texto / escenas
- **Amazon DynamoDB**
    - Tabla `PinkyScreenshots`
- **API Gateway**
    - Endpoints REST protegidos
- **Amazon Cognito**
    - User Pool (jugadores)
    - Autenticación JWT
- **AWS KMS**
    - Cifrado de S3, DynamoDB, logs, variables
- **VPC Private Subnets**
- **VPC Endpoints**
- **Zona de inspección**
- **CodePipeline / CodeBuild**

---

7. Lineamientos Técnicos Específicos

- Cada screenshot debe procesarse en < 3 segundos
- Debe generarse al menos:
    - Thumbnail
    - Optimized version
    - Metadatos estructurados
- Las URLs firmadas deben expirar en máximo 5 minutos
- DynamoDB debe soportar consultas por:
    - `playerId`
    - fecha/rango
    - tipo de evento
- El sistema debe ser idempotente ante reenvíos

---

8. IaC: Requerimientos CloudFormation

La plantilla debe crear:

- Buckets S3 con SSE-KMS
- Lambdas en VPC
- Tabla DynamoDB
- API Gateway
- Cognito Authorizer (parametrizado)
- EventBridge (si aplican flujos internos)
- Endpoints VPC (S3, DynamoDB, Logs)

- Outputs obligatorios
- Uso de roles provistos por Ciberseguridad

---

9. Pipeline CI/CD

Debe contemplar:

1. **Source → GitHub**
2. **Build → CodeBuild**
    - Validación de template
    - Empaquetado de Lambdas
3. **Deploy → CloudFormation**
    - Sandbox
    - Pre-producción
    - Producción

---

10. Seguridad, IAM y Cifrado

Obligatorio:

- Cifrado total con KMS
- Roles IAM proporcionados por Ciberseguridad
- Lambdas en subredes privadas
- Acceso a AWS mediante VPC Endpoints
- Validación estricta del input

- JWT obligatorio para todas las APIs
- Sanitización de metadatos extraídos

---

11. Datos de Prueba (GenAI)

La célula debe generar **mínimo 50 screenshots sintéticos** del escenario:

- Pinky vs Cerebro
- Laboratorio destruido
- Pantallas de nivel
- HUD visible

Recomendado: Generar también **10 variaciones de iluminación/filtros** para probar consistencia del procesador.

---

12. Pruebas funcionales (curl)

Autenticación con Cognito

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

Solicitar URL firmada

```
curl -H "Authorization: Bearer <jwt>" \
  -d '{"playerId":"PLAYER42"}' \
  https://<api>/prod/upload-url
```

Consultar screenshots del jugador

```
curl -H "Authorization: Bearer <jwt>" \
  https://<api>/prod/screenshots?playerId=PLAYER42
```

---

13. Diagrama de Arquitectura

Debe incluir:

- S3 raw → trigger Lambda → S3 processed
- DynamoDB
- Rekognition opcional
- API Gateway + Cognito
- VPC + endpoints
- Zona de inspección
- KMS

---

14. Entregables de la Célula

1. Repositorio GitHub completo
2. CloudFormation
3. Código Python
4. Datos sintéticos
5. Pipeline CI/CD
6. Pruebas curl
7. Diagrama draw.io
8. Historias de usuario
9. Costos estimados

---

15. Criterios de Evaluación

- Correctitud del flujo de procesamiento
- Calidad del IaC
- Pipeline automatizado
- Uso correcto de seguridad
- Desacoplamiento de componentes
- Calidad del diagrama
- Pruebas curl funcionando
- Claridad y documentación

---

16. Control de Cambios

Versión	Fecha	Descripción	Autor
1.0	2025	Primera versión del documento de la Célula 5	Equipo de Arquitectura

---