

Evaluación del módulo 8

Consigna del proyecto 

Evaluación del módulo

Proyecto: Serverless Inteligente

Situación inicial

Unidad solicitante: **Equipo de Innovación Cloud** de una compañía de comercio electrónico.

La plataforma actual depende de servidores administrados que generan altos costos y tiempos de inactividad cuando la demanda crece repentinamente (ventas flash, campañas de marketing). La dirección tecnológica solicita una **migración a un modelo 100 % serverless** que reduzca la carga operativa, escale de forma automática y pague solo por uso real, usando recursos gratuitos de **AWS Academy** y diagramación en **Cloudcraft**.

Nuestro objetivo

Diseñar e implementar soluciones cloud escalables mediante una arquitectura sin servidor (serverless), aprovechando servicios como **AWS Lambda, API Gateway, DynamoDB y S3**.

Producto esperado

Aplicación funcional basada en arquitectura serverless:

- Backend en **funciones Lambda**.
- APIs seguras vía **API Gateway** (JWT / Cognito).
- Persistencia en **DynamoDB** y objetos en **S3**.
- Entrega de contenido estático (S3 + CloudFront).
- Diagrama completo en **Cloudcraft** con estimación de costos, métricas y estrategias de optimización activas.

Requerimientos

1. **Compute:** funciones Lambda (≥ 4) escritas en Python o Node.js.
2. **API:** API Gateway HTTP/REST con autenticación JWT y rate-limit.
3. **Persistencia:**
 - DynamoDB (modo On-Demand, cifrado KMS).
 - S3 con versionado y OAC (CloudFront).
4. **Mensajería:** integración SNS/SQS para procesos asíncronos.
5. **Observabilidad:** CloudWatch Logs, Metrics y X-Ray habilitados.
6. **Costos:** reporte Cloudcraft y AWS Pricing Calculator.
7. **CI/CD:** despliegues automatizados con AWS SAM o CDK.


Métricas Generales


- CRUD implementado: **mín. 4 / máx. 5 funcionalidades.**
- Tests unitarios: **mín. 8 / máx. 16.**
- Cobertura JaCoCo (o equivalente Istanbul/nyc): **mín. 80 %.**
- Ciclos TDD (RED-GREEN-REFACTOR): **mín. 12.**
- Refactorizaciones: **mín. 3 / máx. 5.**
- Uso de Mockito (o sinon.js): **mín. 1 dependencia mockeada.**

Paso a paso

Este proyecto corresponde al módulo **M8: Arquitecturas Serverless** y consta de **7 lecciones** que avanzarás de manera escalonada con apoyo de los manuales y las clases en vivo. Reservá tiempo asincrónico para completar cada etapa y trae tus dudas a los espacios sincrónicos.

Lección 1 – Introducción a la arquitectura sin servidor

 Objetivo: asimilar los fundamentos de serverless y preparar el entorno de desarrollo.

 Tareas a desarrollar:

- Leer el Manual #1 “Introducción a la arquitectura sin servidor”.
- Configurar VS Code, AWS CLI y SAM CLI con la cuenta de AWS Academy.
- Crear el repositorio Git y carpetas src / test.
- Realizar **1 ciclo TDD** (commit RED) con una función “Hello World” en Lambda.
- Métrica mínima: **1 prueba RED** documentada y ejecutada.


Lección 2 – Sitio estático serverless

 Objetivo: desplegar un sitio web estático globalmente con S3 y CloudFront.

 Tareas a desarrollar:

- Leer el Manual #2.
- Crear bucket S3, habilitar hosting estático y subir la landing.
- Configurar CloudFront + Origin Access Control y TLS (ACM).
- Registrar dominio en Route 53 y apuntar al CDN.
- Capturar métricas de latencia antes/después del CDN.

Lección 3 – Funciones como servicio (FaaS)


 Objetivo: implementar micro-funciones Lambda y buenas prácticas event-driven.

 Tareas a desarrollar:

- Leer el Manual #3.
- Desarrollar funciones Lambda para **usuarios** y **pedidos** (CRUD).
- Configurar triggers: API Gateway (HTTP) y S3 ObjectCreated.
- Escribir pruebas unitarias (mín. 8) y medir cobertura ($\geq 80\%$).
- Documentar cold-starts y optimizar tiempos (layers, tamaño paquete).


Lección 4 – API Gateway

 Objetivo: exponer las funciones mediante una API segura y versionada.

 Tareas a desarrollar:

- Leer el Manual #4.
- Diseñar endpoints REST (/v1/users, /v1/orders).
- Habilitar autenticación JWT con Cognito Authorizer.
- Configurar throttling (100 req/s) y caché selectivo.
- Probar con Postman e incluir capturas de respuestas 200/401.

Lección 5 – Persistencia serverless


 Objetivo: integrar DynamoDB y S3 como almacenes de datos gestionados.


 Tareas a desarrollar:

- Leer el Manual #5.
- Crear tabla DynamoDB **Orders** (PK orderId, SK createdAt).
- Habilitar On-Demand, Point-In-Time Recovery y métricas Contributor Insights.
- Guardar facturas PDF en S3; generar URL firmada con Lambda.

- Medir RCU/WCU y optimizar mediante diseño de clave.

Lección 6 – Representación en Cloudcraft


 Objetivo: diagramar la solución y estimar costos.

 Tareas a desarrollar:

- Leer el Manual #6.
- Representar VPC, subredes, API Gateway, Lambdas, DynamoDB, S3, CloudFront, SNS/SQS.
- Etiquetar flujos de datos (HTTP, eventos, colas).
- Activar la vista **Cost** y exportar CSV + imagen PNG.
- Incluir diagrama en el documento final.

Lección 7 – Crecimiento y optimización

 Objetivo: aplicar estrategias de performance y ahorro de costos.

 Tareas a desarrollar:

- Leer el Manual #7.
- Habilitar Lambda Provisioned Concurrency en horas pico (CloudWatch Schedule).
- Activar S3 Intelligent-Tiering y TTL en API Gateway Cache.
- Configurar alarmas: 5XX API Gateway, Throttles Lambda, RCU DynamoDB.
- Documentar el impacto en costos (antes/después) con Cost Explorer.

¿Qué vamos a validar?

- Funcionamiento integral de la aplicación serverless (API, Lambda, DB, S3).

- Aplicación de patrones: API Gateway, event-driven, mensajería, seguridad JWT.
- Cumplimiento de **Métricas Generales** (CRUD, pruebas, TDD, refactor).
- Documentación clara y completa (Word).
- Diagrama Cloudcraft coherente y costos justificados.
- Estrategias de optimización activas y monitorización operativa.

Referencias

AWS Lambda Developer Guide

<https://docs.aws.amazon.com/lambda/latest/dg/welcome.html>

Amazon API Gateway Developer Guide

<https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html>

Amazon DynamoDB Developer Guide

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Welcome.html>

Amazon S3 Website Hosting

<https://docs.aws.amazon.com/AmazonS3/latest/dev/WebsiteHosting.html>

Cloudcraft – Getting Started

<https://docs.cloudcraft.co/>

AWS Well-Architected Framework – Serverless Lens

<https://docs.aws.amazon.com/wellarchitected/latest/serverless-applications-lens/>

Recursos

- YouTube: “Serverless Land Patterns”, “Deploying APIs with API Gateway & Lambda”, “DynamoDB Single-Table Design”.

- AWS Blog: “Optimizing Lambda Costs”, “API Gateway Caching Strategies”.
- Serverless Land (patrones listos para usar).
- Workshops AWS: “Build a Modern Serverless Web Application”.
- Foros AWS y Stack Overflow para dudas específicas.

Entregables

1. **Documento Word** con:
 - Portada del proyecto.
 - Desarrollo de cada lección (capturas + explicación).
 - Evidencia de pruebas unitarias, ciclos TDD, refactorizaciones.
 - Tabla de métricas alcanzadas.
 - Diagrama Cloudcraft (imagen/link) y análisis de costos.
 - Conclusiones y aprendizajes.
2. **Presentación breve** (5 diapositivas) para exponer resultados.
3. **Repositorio Git** (privado o público) con código fuente y plantilla IaC (SAM/CDK).

Portafolio

Incluye “**Serverless Inteligente**” en tu portafolio destacando:

- Diseño event-driven con Lambda, API Gateway y DynamoDB.
- Seguridad y autenticación JWT.

- CI/CD automatizado y pruebas TDD con cobertura $\geq 80\%$.
- Diagrama profesional en Cloudcraft con optimización de costos.
- Métricas de rendimiento y estrategias de ahorro aplicadas.

¡Éxitos!

Nos vemos más adelante

