

Evaluación del módulo 7

Consigna del proyecto 

Evaluación del módulo

Proyecto: Microservicios Orquestados

Situación inicial

Unidad solicitante: Departamento de Tecnología de la fintech **MicroPay**.

El sistema monolítico actual impide liberar funcionalidades de forma ágil y presenta cuellos de botella durante picos de uso. Para mejorar la capacidad de respuesta y la resiliencia del negocio, la empresa decidió adoptar una **arquitectura de microservicios** orquestada en la nube con herramientas gratuitas disponibles en **AWS Academy** y representaciones visuales en **Cloudcraft**.

Nuestro objetivo

Diseñar e implementar una arquitectura basada en microservicios utilizando tecnologías de contenedores y orquestación, aplicando patrones arquitectónicos clave y buenas prácticas para lograr alta disponibilidad, escalabilidad y resiliencia en un entorno cloud.

Producto esperado

Implementación funcional de una aplicación compuesta por microservicios:

- Desplegada en **Kubernetes (Amazon EKS)**, gestionada con **eksctl**.
- Integrada mediante un **API Gateway** que aplica autenticación **JWT**.
- Con patrones de **descubrimiento de servicios**, **circuit breaker** y mensajería asíncrona.
- Diagrama arquitectónico completo en **Cloudcraft** con estimación de costos y documentación de diseño.

Requerimientos


- Contenedores Docker alojados en **Amazon ECR**.
- Orquestación y auto-escalado en **EKS** (múltiples zonas de disponibilidad).
- **API Gateway** como punto único de entrada y verificador de tokens JWT.
- **Service Discovery** (AWS Cloud Map) para registro dinámico.
- **Circuit Breaker** con Resilience4j en los microservicios críticos.
- Mensajería con **SNS** y **SQS** para procesos asíncronos.
- Diagrama detallado en Cloudcraft con costos mensuales estimados.
- Evidencia de TDD, pruebas unitarias y refactorizaciones en un documento Word.

Métricas Generales


- CRUD implementado: mín. 4 – máx. 5 funcionalidades.
- Tests unitarios: mín. 8 – máx. 16.
- Cobertura JaCoCo: mín. 80 %.
- Ciclos TDD (RED-GREEN-REFACTOR): mín. 12.
- Refactorizaciones: mín. 3 – máx. 5.
- Uso de Mockito: mín. 1 dependencia mockeada.

Paso a paso

Lección 1 – Fundamentos de Microservicios y TDD

 Objetivo: comprender los principios de microservicios y preparar el entorno de


pruebas.

 Tareas a desarrollar:

- Leer el Manual #1.
- Instalar VS Code, JDK 17, JUnit 5 y JaCoCo.
- Crear estructura src y test para el microservicio **Usuarios**.
- Iniciar repo Git y realizar al menos un ciclo TDD completo (commit RED).
- Documentar la primera prueba RED.

Lección 2 – Patrones Arquitectónicos Clave


 Objetivo: aplicar API Gateway, Service Discovery, Circuit Breaker y Seguridad JWT.

 Tareas a desarrollar:

- Leer el Manual #2.
- Implementar **API Gateway** y autenticación JWT.
- Configurar **AWS Cloud Map** para descubrimiento de servicios.
- Integrar **Circuit Breaker** con Resilience4j en el microservicio Pagos.
- Crear pruebas unitarias con Mockito para la capa de seguridad.

Lección 3 – Orquestación con Kubernetes (EKS)

 Objetivo: empaquetar y desplegar los microservicios en EKS.


 Tareas a desarrollar:

- Leer el Manual #3.
- Crear **Dockerfile** para cada microservicio y subir las imágenes a **ECR**.
- Aprovisionar un clúster **EKS** con eksctl (tres nodos, dos AZ).

- Describir Pods, Services y Deployments en archivos YAML.
- Configurar Horizontal Pod Autoscaler y un Ingress Controller.
- Capturar métricas de CPU y escalado automático.

Lección 4 – Representación Cloud y Costos

 Objetivo: diagramar la arquitectura final y estimar los costos.

 Tareas a desarrollar:

- Leer el Manual #4.
- Dibujar en **Cloudcraft**: VPC, subredes, API Gateway, clúster EKS, bases de datos, SNS/SQS y demás componentes.
- Etiquetar cada microservicio y subnet.
- Utilizar la calculadora integrada para obtener el costo mensual estimado.
- Exportar imagen o link y añadirlo al documento Word.

¿Qué vamos a validar?

- Despliegue exitoso de microservicios en EKS con auto-escalado.
- Aplicación de patrones (API Gateway, Service Discovery, Circuit Breaker, JWT).
- Cumplimiento de las métricas de CRUD, pruebas, TDD y refactor.
- Documentación clara en Word con capturas y explicaciones.
- Diagrama Cloudcraft coherente y costos justificados.
- Buenas prácticas de seguridad, resiliencia y monitoreo.

Referencias 🚧

Librerías y Frameworks:

- JUnit 5: <https://junit.org/junit5/docs/current/user-guide/>
- JaCoCo: <https://www.jacoco.org/jacoco/trunk/doc/>
- Mockito: <https://site.mockito.org/>
- Resilience4j: <https://resilience4j.readme.io/>

Documentación oficial:

- AWS Academy: <https://awsacademy.instructure.com>
- AWS Docs (EKS, ECR, API Gateway, Cloud Map): <https://docs.aws.amazon.com/>
- Kubernetes Docs: <https://kubernetes.io/docs/>
- Cloudcraft: <https://www.cloudcraft.co/>

Guías y cálculo:

- eksctl Documentation: <https://eksctl.io/>
- AWS Pricing Calculator: <https://calculator.aws.amazon.com/>

Recursos 📁

- YouTube: “Kubernetes Crash Course”, “Deploying Microservices on EKS”, “JWT Authentication Explained”.
- AWS Blog: serie “Building Microservices on AWS”.
- Medium / Dev.to: artículos sobre Resilience4j y Cloud Map.
- Plantillas YAML de Kubernetes para Deployments y HPA.
- Foros Stack Overflow y comunidad AWS para resolver dudas.

Entregables ✅

- **Documento Word** con:
 - Portada del proyecto.

- Desarrollo de cada lección con capturas.
 - Evidencia de pruebas unitarias, ciclos TDD y refactorizaciones.
 - Tabla de métricas cumplidas.
 - Diagrama Cloudcraft (imagen/link) y costos.
 - Conclusiones y aprendizajes.
- **Presentación breve** (opcional) para exponer resultados.

Portafolio

Incluye este proyecto como **“Microservicios Orquestados en AWS con EKS”** y destaca:

- Orquestación Kubernetes y auto-escalado.
- Implementación de API Gateway, JWT, Service Discovery y Circuit Breaker.
- Uso de mensajería SNS/SQS para desacoplar procesos.
- Diseño visual y costos estimados en Cloudcraft.
- Buenas prácticas de TDD, pruebas y refactor.

¡Éxitos!

Nos vemos más adelante

