

Semana 2

Desarrollo Backend I (PBY2201)

Formato de respuesta

| **Nombre estudiante:** | Carlos Barría Valdevenito |
| --- | --- |
| **Asignatura: Desarrollo Backend\_1** | **Carrera: Desarrollo de Aplicaciones** |
| **Profesor: Carlos Ignacio Valverde Olivares** | **Fecha: 24-03-2025** |

# Descripción de la actividad

En esta segunda semana realizarás una actividad sumativa grupal con encargo llamada "Proponiendo arquitectura". Aquí, tendrás que brindar una solución con arquitectura de microservicios planteando los beneficios de usar este modelo a través de un caso planteado.

## Instrucciones específicas

Para realizar la actividad de esta semana, tendrás que desarrollar tu proyecto a partir del siguiente caso:

### Contexto del caso

Una tienda de ropa está planeando la implementación de un nuevo sistema de gestión de microservicios para mejorar sus operaciones. El sistema debe gestionar productos, ventas, clientes y sucursales, permitiendo una comunicación eficiente entre los servicios a través de una API. Como desarrollador/A backend, tu tarea es proponer una solución de arquitectura de microservicios, la cual realizarás en el **framework Spring Boot,** en donde diseñarás un sistema para la tienda de ropa que cumpla con los siguientes **requerimientos técnicos**:

* **Autenticación de usuarios:** El sistema debe permitir a los usuarios registrarse, iniciar sesión y recuperar contraseñas de manera segura. Solo los usuarios autorizados deben poder acceder a sus cuentas y gestionar sus datos personales.
* **Gestión de inventario:** El sistema debe gestionar el inventario en tiempo real, permitiendo agregar, actualizar y eliminar productos. El control preciso del inventario es esencial para asegurar que los productos disponibles en la tienda en línea coincidan con el inventario real.
* **Procesamiento de pagos:** El sistema debe procesar pagos de manera segura, integrando varios métodos de pago (tarjetas de crédito, PayPal, etc.). La seguridad en las transacciones es crucial para proteger la información financiera de los usuarios.
* **Gestión de pedidos:** El sistema debe permitir a los usuarios realizar pedidos, verificar el estado de sus pedidos y recibir notificaciones sobre el envío. Esto incluye desde la confirmación del pedido hasta la organización del envío.
* **Análisis de Ventas:** El sistema debe proporcionar informes y análisis de ventas que permitan a la administración tomar decisiones informadas. Los informes deben incluir tendencias de ventas, productos más vendidos y comportamiento del cliente.
* **Comunicación entre Microservicios:** Los diferentes componentes del sistema deben comunicarse de manera eficiente y segura para garantizar que la información relevante se comparta correctamente entre los microservicios.
* **Base de datos y almacenamiento:** El sistema debe utilizar bases de datos eficientes para almacenar la información de usuarios, pedidos, inventario y transacciones, garantizando que las operaciones CRUD (crear, leer, actualizar, eliminar) sean rápidas y seguras.
* **Consistencia de datos:** Los datos compartidos entre los microservicios deben ser consistentes y estar actualizados para mantener la integridad y confiabilidad de la información.
* **Seguridad y cumplimiento:** El sistema debe cumplir con las regulaciones necesarias para proteger los datos de los usuarios, implementando medidas de seguridad como el cifrado de datos y controles de acceso.
* **Escalabilidad:** El sistema debe estar diseñado para escalar y soportar un aumento en el número de usuarios y transacciones sin afectar el rendimiento.

Ahora que cuentas con los requerimientos finales del caso, a continuación, te detallaremos el paso a paso que debes seguir:

### Paso 1: Definir los microservicios

Identifica al menos cuatro microservicios (por ejemplo: gestión de productos, ventas, clientes y pedidos) que cubran las funcionalidades mencionadas en los requerimientos del sistema. Asegúrate de que cada microservicio tenga una responsabilidad clara y específica.

**Paso 2: Diseñar la lógica de los microservicios:**

Desarrolla la funcionalidad interna de cada microservicio, explicando cómo gestionarás las operaciones CRUD para cada entidad (usuarios, productos, ventas, pedidos). Describe cómo estos microservicios implementarán la autenticación, el procesamiento de pagos, la gestión de inventario, entre otros aspectos técnicos.

**Paso 3: Definir las APIs y la comunicación entre microservicios:**

Define los endpoints que permitirán la comunicación entre microservicios, especificando si se utilizará una comunicación síncrona o asíncrona, y por qué. Justifica tu elección con base en la necesidad de consistencia de datos y eficiencia en la comunicación.

**Paso 4: Crear un diagrama de arquitectura:**

Utiliza una herramienta de diagramación para representar gráficamente la arquitectura de microservicios propuesta. Asegúrate de que el diagrama muestre claramente las interacciones entre los servicios, los endpoints, las bases de datos y cómo cada uno cumple con los requerimientos del sistema.

**Preguntas orientadoras**

A continuación, te presentamos algunas preguntas que pueden ayudarte en la realización de tu diagrama y a reflexionar sobre lo que requiere el proyecto. **No es necesario responder estas preguntas,** pero pueden servirte como guía para asegurarte de que tu propuesta cubra todos los aspectos clave:

* ¿Por qué es importante que el sistema permita a los usuarios registrarse, iniciar sesión y recuperar contraseñas de manera segura?
* ¿Cómo ayuda la gestión en tiempo real del inventario a mejorar la precisión y eficiencia del sistema de ventas en línea?
* ¿Qué medidas de seguridad deben implementarse en el sistema para garantizar que los pagos en línea se realicen de manera segura?
* ¿Qué información debe poder verificar el usuario sobre sus pedidos y por qué es importante mantener al cliente informado durante el proceso de compra?
* ¿Qué tipo de datos deben incluir los informes de ventas para ayudar a la administración a tomar decisiones informadas?

**Paso 5: Justificar la escalabilidad y la seguridad:**

Explica cómo la arquitectura propuesta garantizará la escalabilidad para soportar un crecimiento en usuarios y transacciones. Además, detalla las medidas de seguridad que implementarás para proteger los datos y cumplir con las normativas de privacidad.

## Entregables

En este apartado del documento, deberás adjuntar lo siguiente:

* **Propuesta de arquitectura con justificación** de no más de 5 líneas, destacando las razones por las que decidiste utilizar este modelo y cómo satisface los requerimientos del sistema (autenticación, inventario, pagos, etc.).
* **Diseño de la lógica del servicio (CRUD) para cada microservicio,** explicando cómo gestionarás las operaciones de Crear, Leer, Actualizar y Eliminar.
* **Definición de las APIs y la comunicación entre microservicios,** incluyendo las rutas y métodos HTTP, y si la comunicación es síncrona o asíncrona.
* **Diagrama de la arquitectura de microservicios** utilizando un software de diagramado (como Lucidchart, Draw.io o similar). Puedes colocar la imagen o compartir el enlace de tu diagrama.
* **Justificación de la escalabilidad y seguridad** del sistema, explicando cómo el sistema soportará el crecimiento y cómo protegerás los datos.



Importante

Una vez termines de adjuntar tus evidencias, no olvides subir este archivo al AVA.

|  |  |
| --- | --- |
| **Nombre**  **Microservicios** | ms-usuarios |
| **Descripción** | Gestiona la autenticación y administración de usuarios. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Verbo** | **uri** | **Descripción** | **Parámetros entrada** |
| GET | /usuarios | Listar usuarios |  |
| POST | /usuarios | Registrar usuario id(int) |  |
| GET | /usuarios/{id} | Listar usuario único según id seleccionado | id |
| PUT | /usuarios/{id} | Actualizar datos del usuario | id |
| DELETE | /usuarios/{id} | Eliminar usuario | id |

|  |  |
| --- | --- |
| **Nombre Campo** | **Tipo de dato** |
| Id | int |
| Nombre | String |
| Email | String |
| Rol | String (ADMIN/CLIENTE) |

|  |  |
| --- | --- |
| **Nombre**  **Microservicios** | ms-inventario |
| **Descripción** | Administra los productos en stock, permitiendo su consulta y modificación en tiempo real. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Verbo** | **uri** | **Descripción** | **Parámetros entrada** |
| GET | /productos | Obtener lista de productos |  |
| POST | /productos | Agregar un producto |  |
| GET | /productos/{id} | Obtener producto por ID |  |
| PUT | /productos/{id} | Actualizar producto |  |
| DELETE | /productos/{id} | Eliminar producto |  |

|  |  |
| --- | --- |
| **Nombre Campo** | **Tipo de dato** |
| id | int |
| nombre | String |
| descripcion | String |
| precio | Decimal |
| stock | Int |

|  |  |
| --- | --- |
| **Nombre**  **Microservicios** | ms-pagos |
| **Descripción** | Maneja las transacciones de pago de forma segura |

|  |  |  |  |
| --- | --- | --- | --- |
| **Verbo** | **uri** | **Descripción** | **Parámetros entrada** |
| POST | /pagos | Procesar pago |  |
| GET | /pagos/{id} | Consultar estado de pago | id |
|  |  |  |  |

|  |  |
| --- | --- |
| **Nombre Campo** | **Tipo de dato** |
| id | int |
| pedido\_id | int |
| monto | decimal |
| estado | string |
| metodo\_pago | string(tarjeta/PayPal) |

|  |  |
| --- | --- |
| **Nombre**  **Microservicios** | ms-pedidos |
| **Descripción** | Permite la creación, actualización y seguimiento de los pedidos realizados por los clientes. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Verbo** | **uri** | **Descripción** | **Parámetros entrada** |
| POST | /pedidos | Crear un pedido |  |
| GET | /pedidos | Listar pedidos |  |
| GET | /pedidos/{id} | Obtener detalle de un pedido | id |
| PUT | /pedidos/{id} | Actualizar estado del pedido | id |
| DELETE | /pedidos/{id} | Cancelar pedido | id |

|  |  |
| --- | --- |
| **Nombre Campo** | **Tipo de dato** |
| id | int |
| usuario\_id | int |
| productos | int |
| total | decimal |
| estado | string(pendiente/entregado/cancelado) |

|  |  |
| --- | --- |
| **Nombre**  **Microservicios** | ms-ventas |
| **Descripción** | Genera reportes de ventas y comportamiento de clientes |

|  |  |  |  |
| --- | --- | --- | --- |
| **Verbo** | **uri** | **Descripción** | **Parámetros entrada** |
| GET | /ventas | Obtener informe de ventas |  |
| GET | /ventas | Obtener informe de comportamiento de clientes |  |

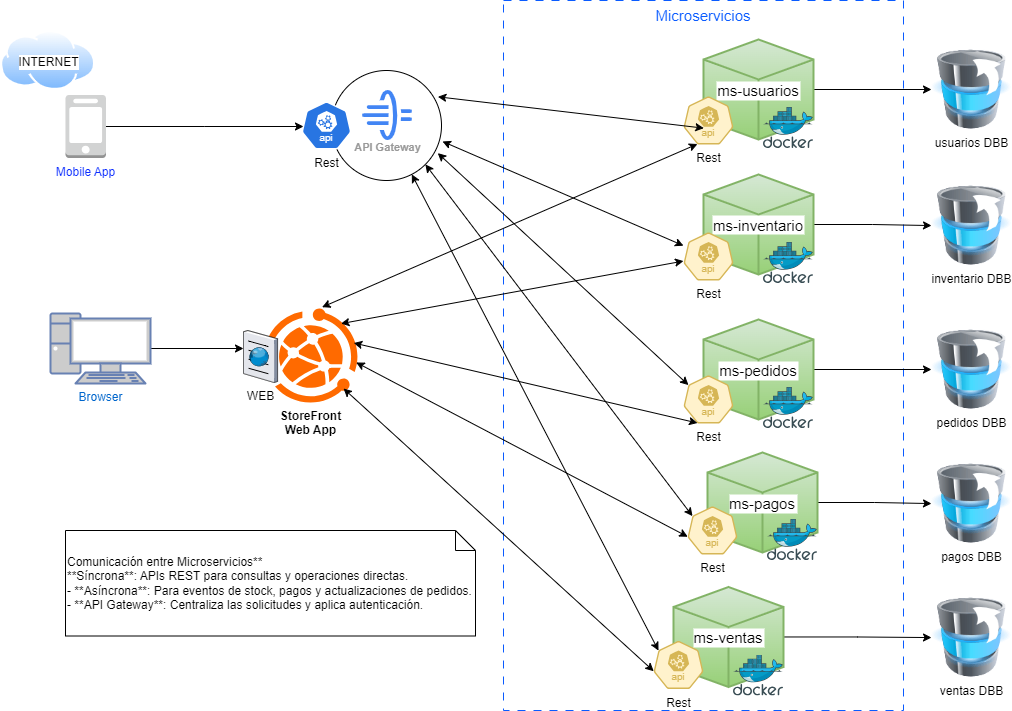
|  |  |
| --- | --- |
| **Nombre Campo** | **Tipo de dato** |
| id | int |
| fecha | date |
| total\_ventas | Decimal |
| productos\_mas\_vendidos | Lista de productos |
| clientes\_frecuentes | Lista de clientes |

**Justificación de Escalabilidad y Seguridad**

Escalabilidad: Kubernetes y Docker permiten el despliegue dinámico según la demanda.

Seguridad: JWT, cifrado de datos y auditoría de accesos protegen la información.

**Diagrama de Arquitectura de microservicios**





Reservados todos los derechos Fundación Instituto Profesional Duoc UC. No se permite copiar, reproducir, reeditar, descargar, publicar, emitir, difundir, de forma total o parcial la presente obra, ni su incorporación a un sistema informático, ni su transmisión en cualquier forma o por cualquier medio (electrónico, mecánico, fotocopia, grabación u otros) sin autorización previa y por escrito de Fundación Instituto Profesional Duoc UC La infracción de dichos derechos puede constituir un delito contra la propiedad intelectual.