# **Informe de Evaluación Parcial N° 1**

## **Caso de Análisis Semestral: Transformación Digital de NUBEPLAY**

### **1. Análisis de Requerimientos**

|  |  |
| --- | --- |
| **Requisitos Funcionales** | **Requisitos No Funcionales** |
| Gestión de usuarios (crear, actualizar,desactivar y eliminar cuentas). | Escalabilidad del sistema. |
| Configuración de permisos. | Seguridad en la gestión de datos (autenticación, encriptación). |
| Gestión de inventario, ventas, pedidos y logística. | Alta disponibilidad y tolerancia a fallos. |
| acciones de los clientes a través de la web (registro, compras, gestión de perfil,etc.). | Optimización de las rutas de entrega y la gestión de pedidos. |
| Gestión de eventos (inscripción, organización,integración con ticketmaster, etc.). | **—-------------------------------------------------------** |

### **2. Análisis del Sistema Actual**

#### **Sistema Monolítico**

El sistema actual es un único bloque que maneja todas las operaciones (usuarios, inventarios, ventas, pedidos) de manera centralizada.

#### **Problemas**

* Dificultad para escalar, ya que el crecimiento de un componente afecta a todos los demás.
* Riesgo de sobrecarga y fallos por la centralización de todas las funcionalidades.
* Poca flexibilidad en la implementación de nuevas características (como la gestión de eventos).

### 

### **3. Diseño de la Nueva Arquitectura**

#### **Estrategia de Microservicios**

* Dividir el sistema en pequeños servicios autónomos que interactúan entre sí.

#### **Microservicios Propuestos**

* **Servicio de Usuarios:** Gestión de cuentas, roles y permisos.
* **Servicio de Inventario:** Gestión de productos y stock.
* **Servicio de Ventas:** Procesamiento de ventas y generación de facturas.
* **Servicio de Pedidos:** Gestión de pedidos de clientes.
* **Servicio de Logística:** Gestión de envíos y rutas.
* **Servicio de Eventos:** Inscripción y organización de eventos, integración con Ticketmaster.

#### **Diagrama de Despliegue**

* Los microservicios se distribuirán en contenedores Docker, orquestados con Kubernetes, desplegados en un proveedor de la nube como AWS o Azure.

### **4. Planificación de la Migración**

#### **Fase 1: Evaluación y Planificación Inicial**

* Identificación y documentación de componentes y dependencias del sistema.
* Evaluación de riesgos y plan de mitigación.
* Plan de comunicación y asignación de roles.

#### **Fase 2: Diseño de la Nueva Arquitectura**

* Definición de microservicios y sus funcionalidades.
* Elección de herramientas como Docker, Kubernetes, Spring Boot, Kafka.
* Diseño de APIs RESTful y mecanismos de autenticación.

#### **Fase 3: Implementación de Microservicios**

### Microservicios

Para el desarrollo de la solución, se implementaron tres microservicios independientes: **usuarios**, **inventario** y **carrito**, cada uno con su propio proyecto Spring Boot, base de datos, y lógica de negocio. Esta separación permitió aplicar de forma efectiva la arquitectura de microservicios, promoviendo el desacoplamiento y la escalabilidad del sistema.

* **Microservicio de Usuarios**: Gestiona la información de los usuarios del sistema, permitiendo operaciones CRUD sobre datos como nombre, correo, rol, y estado activo. Es el servicio encargado de la autenticación básica y la gestión de roles.
* **Microservicio de Inventario**: Administra el catálogo de videojuegos disponibles, permitiendo crear, consultar, actualizar y eliminar juegos. Cada juego tiene atributos como título, categoría, stock, precio y rating de contenido.
* **Microservicio de Carrito**: Permite a los usuarios agregar juegos a su carrito de compras. Este servicio realiza llamadas a los microservicios de usuarios e inventario para validar la existencia del usuario y la disponibilidad del producto antes de agregarlo al carrito.

Cada servicio expone un único endpoint RESTful, y se comunicaron de forma indirecta mediante la validación y control lógico local, sin compartir base de datos entre sí. Esta separación asegura que cada microservicio pueda escalar o modificarse sin afectar al resto del sistema.

#### **Fase 4: Migración de Datos**

#### **1. Tabla:** usuarios

* **Descripción**: Almacena los datos de los usuarios registrados en la plataforma.
* **Columnas principales**:
  + userid (INT, PK): Identificador único del usuario.
  + nombre (VARCHAR): Nombre del usuario.
  + correo (VARCHAR, único): Correo electrónico.
  + password (VARCHAR): Contraseña cifrada.
  + rol (VARCHAR): Rol asignado (ej. admin, cliente).
  + activo (BOOLEAN): Estado del usuario.
* **Relaciones**:
  + Se relaciona con la tabla carrito, donde un usuario puede tener uno o más carritos.

#### **2. Tabla:** inventario

* **Descripción**: Contiene la información de los juegos disponibles para la venta.
* **Columnas principales**:
  + id\_juego (INT, PK): Identificador del juego.
  + titulo\_juego (VARCHAR): Título del juego.
  + categoria (VARCHAR): Categoría o género del juego.
  + stock (INT): Cantidad disponible.
  + precio (DOUBLE): Precio de venta.
  + rating\_contenido (INT): Clasificación por edad/contenido.
* **Relaciones**:
  + Relación indirecta con carrito\_item, ya que cada ítem del carrito contiene un juego.

#### **3. Tabla:** carrito

* **Descripción**: Representa un carrito de compras asociado a un usuario.
* **Columnas principales**:
  + id\_carrito (INT, PK): Identificador del carrito.
  + id\_usuario (INT, FK): Referencia al userid de la tabla usuarios.
* **Relaciones**:
  + Uno a muchos con carrito\_item: un carrito puede tener múltiples ítems.
  + Muchos a uno con usuarios: un usuario puede tener varios carritos.

#### **4. Tabla:** carrito\_item

* **Descripción**: Representa un ítem específico dentro de un carrito, incluyendo el juego y la cantidad.
* **Columnas principales**:
  + id\_item (INT, PK): Identificador único del ítem.
  + id\_juego (INT, FK): Referencia al juego (inventario.id\_juego).
  + cantidad (INT): Cantidad del juego.
  + id\_carrito (INT, FK): Referencia al carrito (carrito.id\_carrito).
* **Relaciones**:
  + Muchos a uno con carrito.

#### **Fase 5: Migración Gradual**

* Migración por módulos.
* Pruebas de compatibilidad y supervisión en tiempo real.

#### **Fase 6: Pruebas Finales y Despliegue**

* Pruebas de regresión.
* Despliegue en producción con monitoreo post-despliegue.

#### **Fase 7: Optimización y Mantenimiento**

* Mejoras de rendimiento y escalabilidad.
* Plan de mantenimiento y actualizaciones.
* Documentación y capacitación del equipo.

### **5. Riesgos y Plan de Mitigación**

#### **Riesgos**

* **Interrupciones en el servicio:** Impacto en la experiencia del usuario.
* **Incompatibilidad de datos:** Riesgo de datos corruptos o pérdidos.
* **Sobrecarga en la infraestructura:** Demandas inesperadas de recursos.

#### **Plan de Mitigación**

* **Pruebas exhaustivas** para garantizar compatibilidad y estabilidad.
* **Despliegue gradual** para minimizar riesgos.
* **Monitoreo constante** con herramientas como ELK Stack.

### **6. Historias de usuarios**

| **Como Administrador:** | |
| --- | --- |
| **Quiero:** | **Para:** |
| Gestionar usuarios | Poder crear, actualizar, desactivar y eliminar cuentas de usuarios del sistema para mantener un control adecuado sobre los usuarios activos y su acceso al sistema. |
| Configurar permisos | Asignar y modificar permisos de acceso a los módulos y funciones del sistema para garantizar que los usuarios solo puedan acceder a las áreas correspondientes. |
| Monitorización del sistema | Visualizar el estado del sistema, recibir alertas sobre fallos y monitorizar el rendimiento para tomar acciones correctivas rápidamente en caso de problemas. |
| Respaldar y restaurar datos | Realizar copias de seguridad periódicas de los datos y restaurarlos en caso de pérdida para evitar la pérdida de información crítica. |

| Como Gerente de sucursal | |
| --- | --- |
| **Quiero**: | **Para**: |
| Gestionar inventario | Poder agregar, actualizar y eliminar productos del inventario, y ajustar las cantidades de stock, para asegurarme de que el inventario esté actualizado. |
| Generar reportes | Poder generar reportes sobre ventas, inventario y rendimiento de la sucursal para analizar el desempeño y tomar decisiones informadas. |
| Gestionar sucursales | Configurar detalles específicos de la sucursal, como horarios de apertura, personal asignado, y políticas locales, para tener un control adecuado de todas las sucursales y asegurarse de que operen bajo los lineamientos de la empresa. |
| Gestionar pedidos | Supervisar y autorizar pedidos de productos para reabastecer el inventario de la sucursal, para garantizar que cada sucursal tenga el stock adecuado para satisfacer la demanda de los clientes. |

| Como Empleado de ventas | |
| --- | --- |
| **Quiero:** | **Para:** |
| Registrar ventas | Procesar transacciones de venta en el sistema, aplicando descuentos y ofertas cuando sea necesario, para garantizar que las ventas se registren correctamente y los clientes reciban los productos adquiridos sin inconvenientes. |
| atender devoluciones y reclamaciones | Gestionar devoluciones de productos y reclamaciones de clientes para ofrecer un servicio al cliente de calidad. |
| consultar inventario | Consultar la disponibilidad de productos en el inventario en tiempo real, para poder informar a los clientes sobre la disponibilidad de los productos y realizar ventas sin contratiempos. |
| generar facturas | Emitir facturas electrónicas de las ventas realizadas y enviarlas por correo electrónico a los clientes, Para cumplir con las normativas fiscales y asegurar una correcta documentación de las transacciones. |

| Como Encargado de logística | |
| --- | --- |
| **Quiero:** | **para:** |
| Gestionar envíos | Crear, actualizar y hacer seguimiento de los envíos de productos desde la bodega hasta las sucursales y los clientes, para asegurarse de que los productos lleguen a su destino en tiempo y forma, garantizando la satisfacción del cliente. |
| Optimizar rutas de entrega | Planificar y optimizar las rutas de entrega para pedidos locales y regionales, para reducir los tiempos de transporte y los costos asociados, mejorando la eficiencia operativa de la logística. |
| Actualizar estados de productos | Cambiar el estado de los pedidos en el sistema, desde su procesamiento hasta la entrega final, para mantener actualizada la información de los pedidos y proporcionar a los clientes información precisa sobre el estado de sus compras. |
| gestionar proveedores | Mantener y actualizar la información de los proveedores, realizar pedidos de reabastecimiento y gestionar la recepción de mercancías, para asegurarse de que los productos necesarios para satisfacer la demanda estén siempre disponibles. |

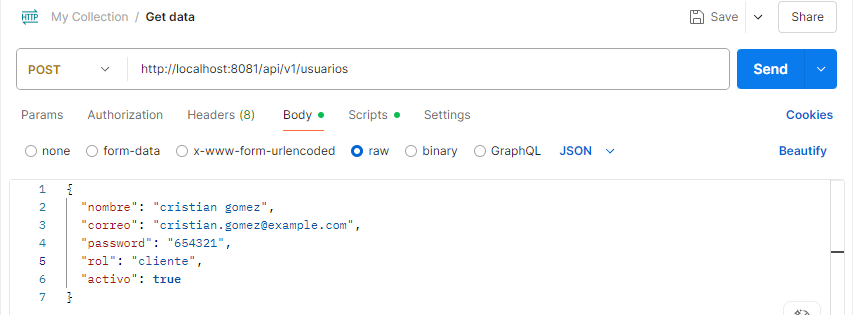
### 

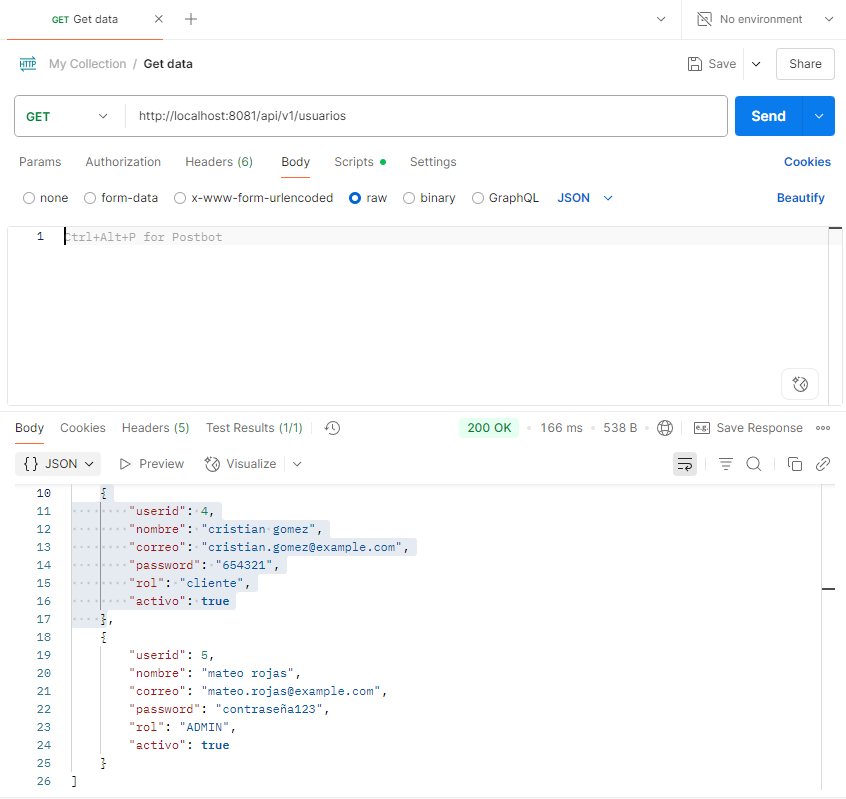
### **7. mockup:**

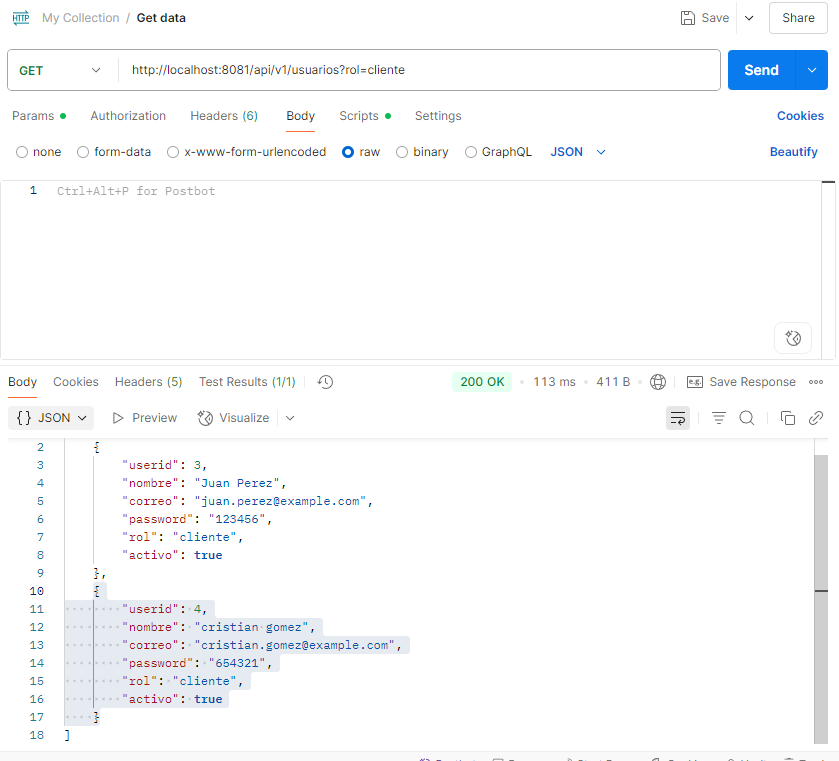
<https://www.renderforest.com/website-maker/1426584/lang/edit?authHash=78b5d8200dad81d2b76d7f4a1c33ec4e>

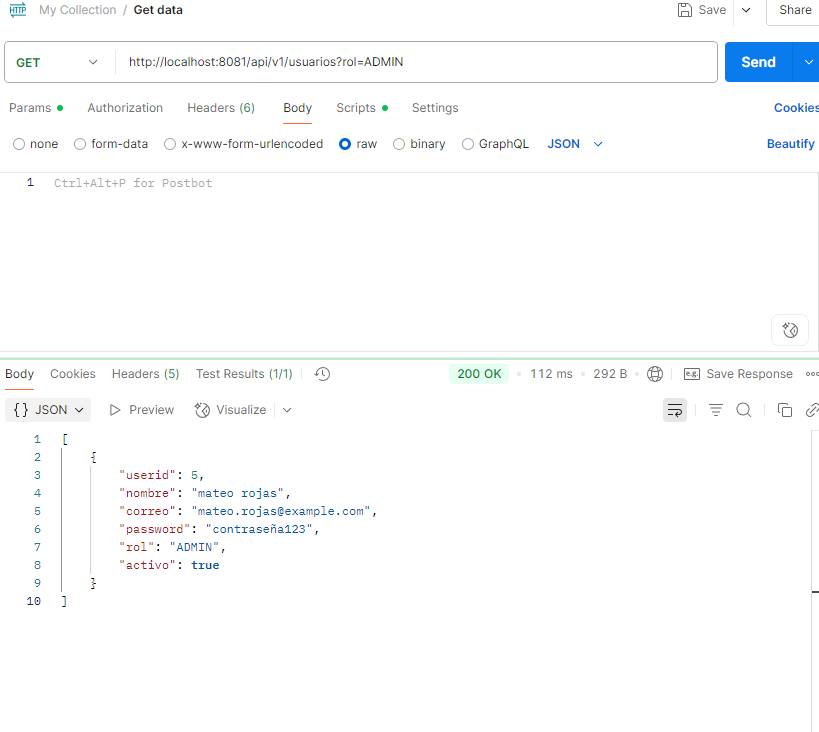
### **Pruebas del Proyecto**

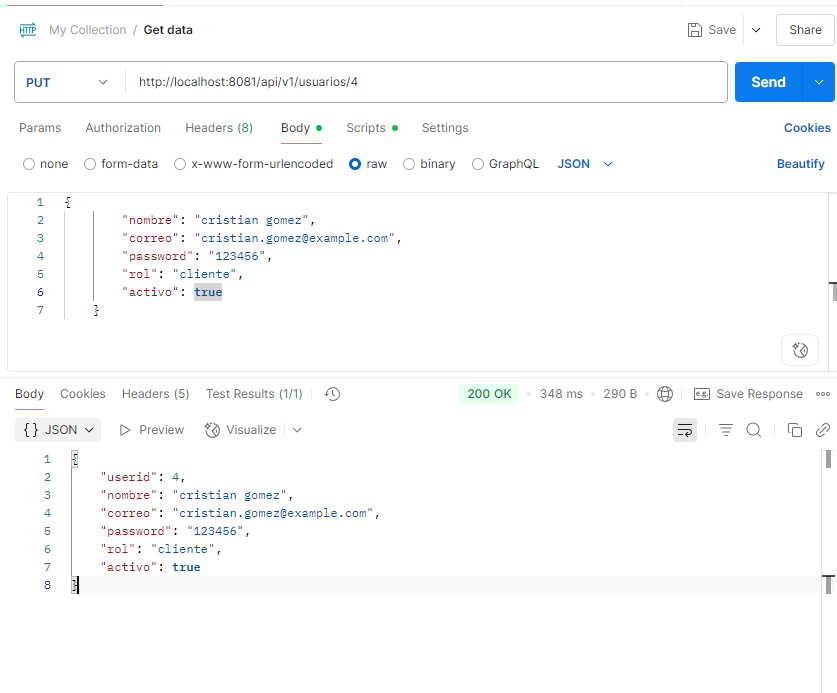
**USUARIO:**

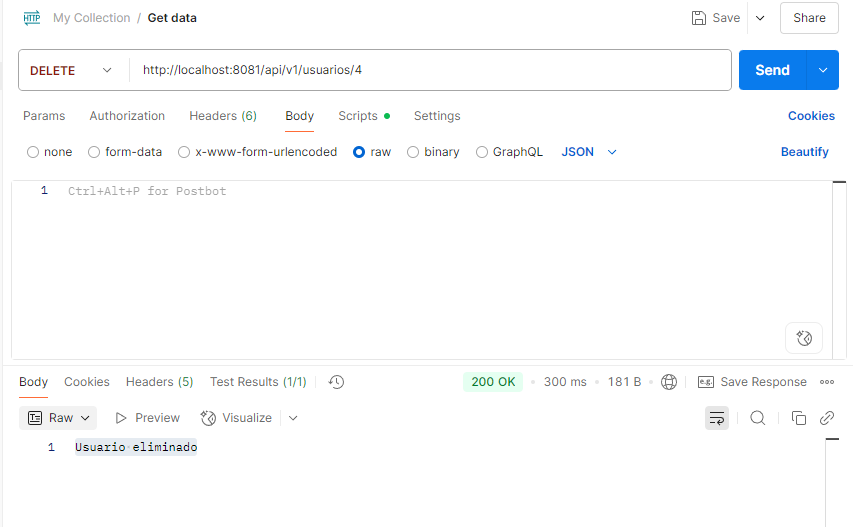


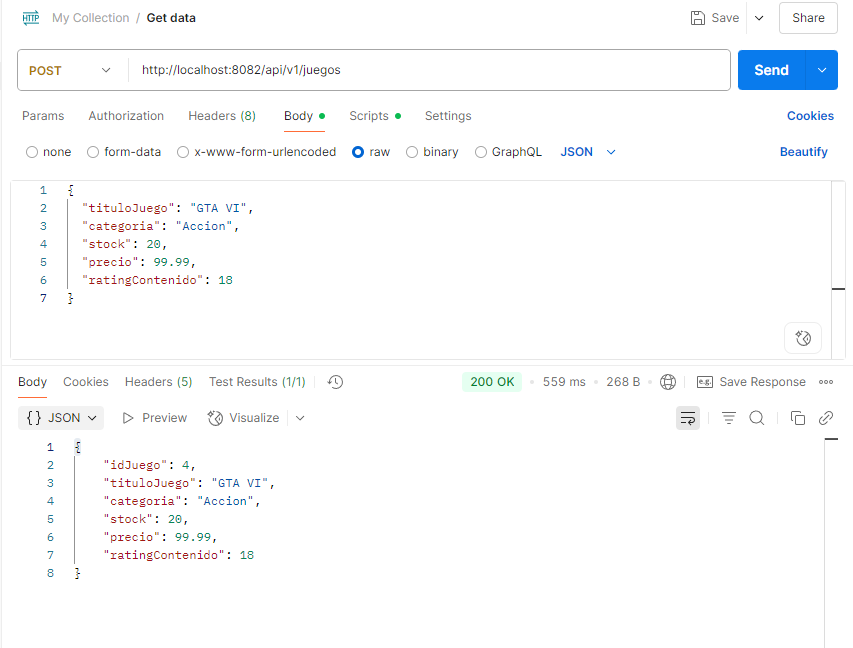


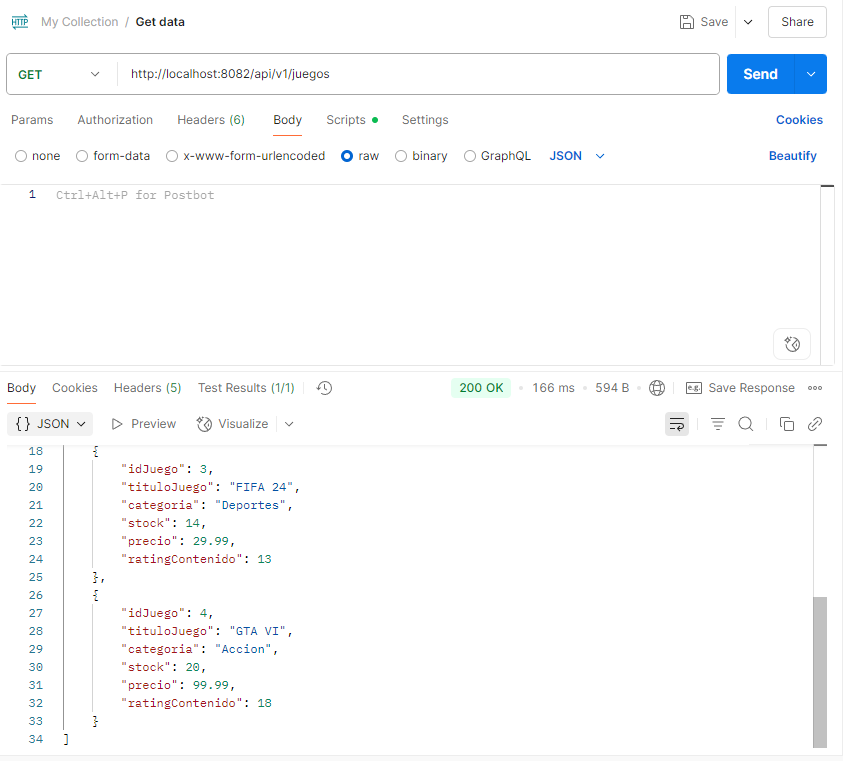


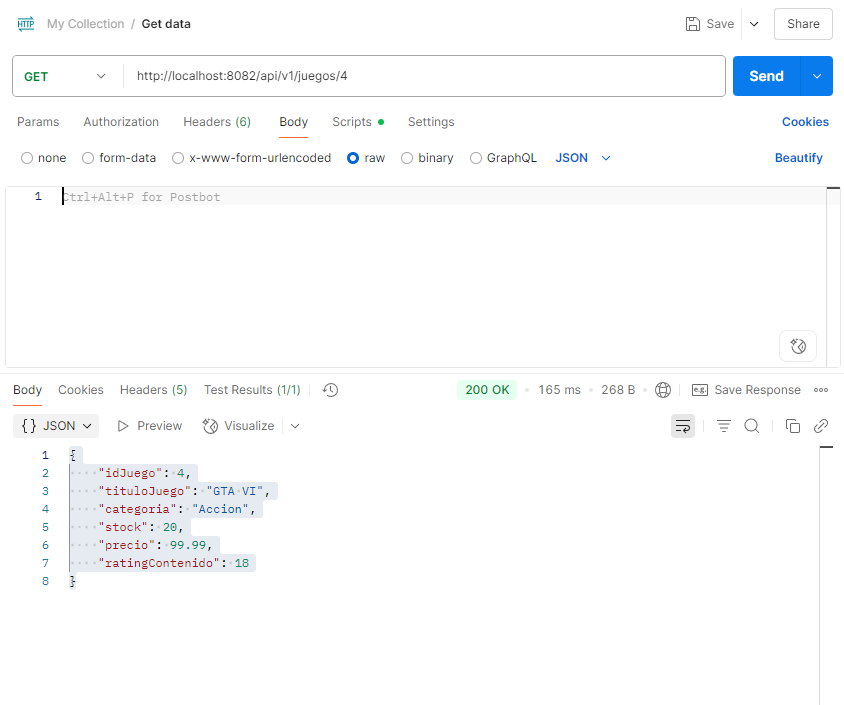


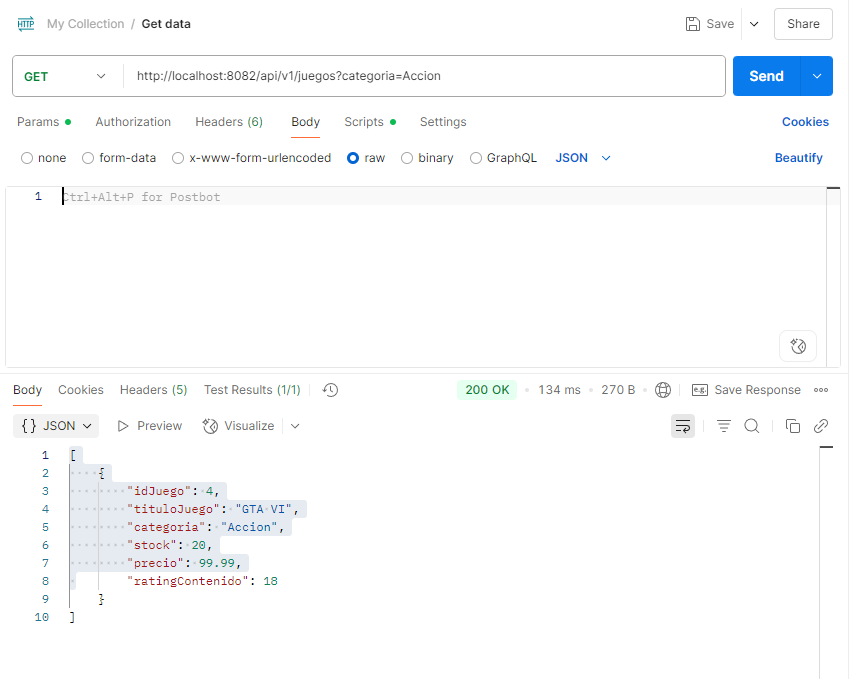


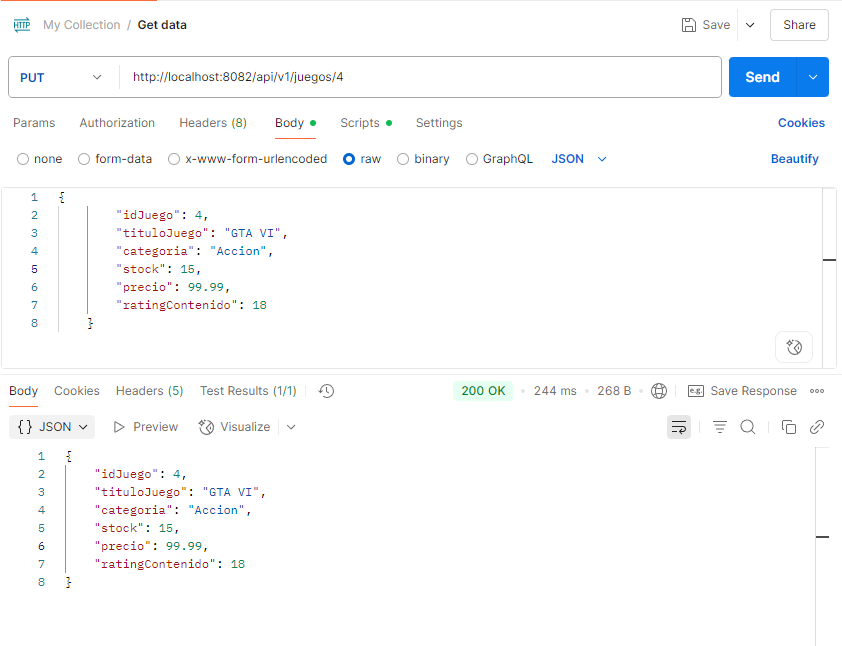


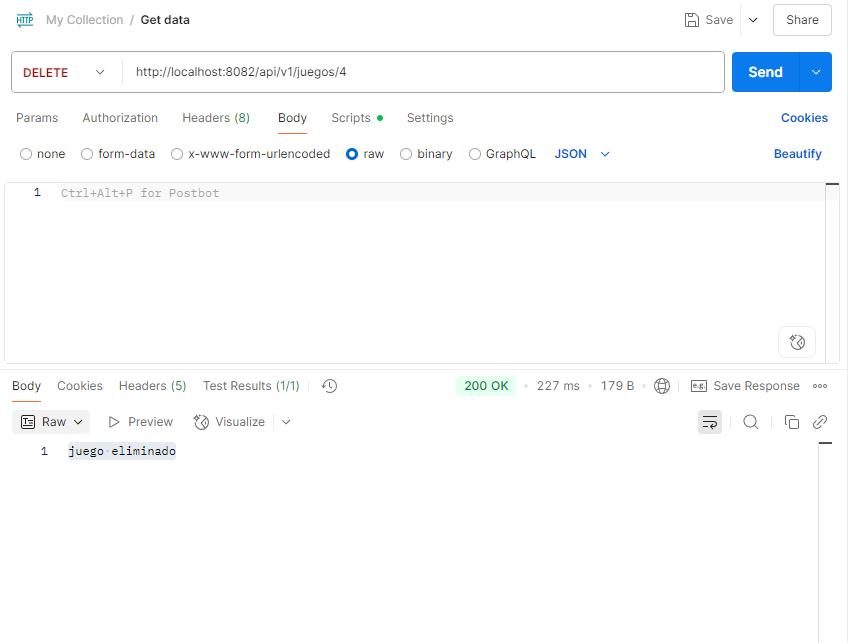
CARRITO  




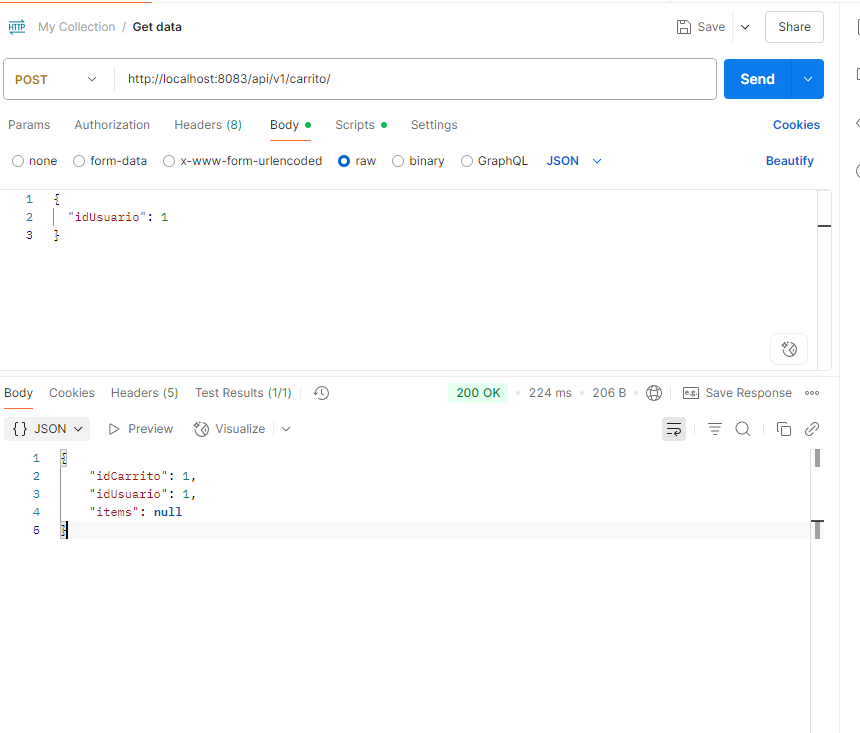


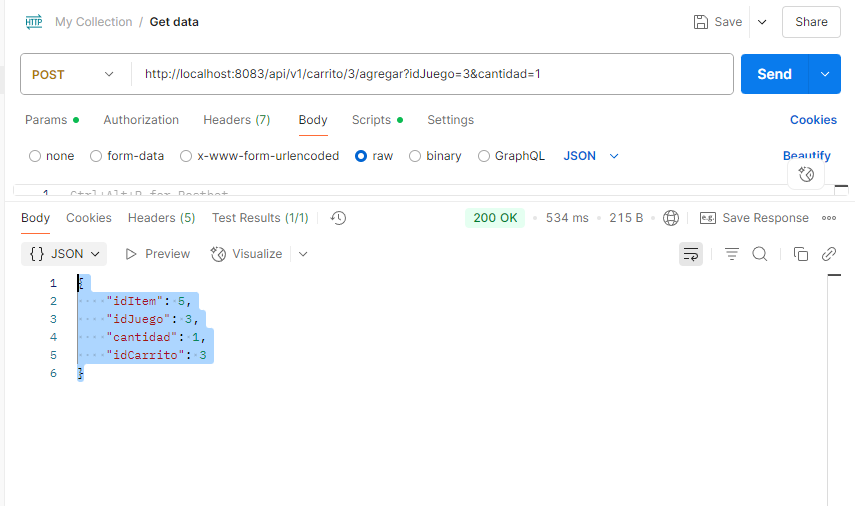






Carrito





### **9. Destino del repositorio de proyecto**

Nombre GITHUB:

[](https://avatars.githubusercontent.com/u/171396992?v=4) **sebasti019**

**Nombre Repositorio: NUBLEPLAY\_ENTREGA**

**Carrito:** NUBEPLAY\_CARRITO/NUBEPLAY\_CARRITO/src/main/java/com/example/NUBEPLAY\_CARRITO

**Usuario:**

NUBEPLAY\_USUARIO/NUBEPLAY\_USUARIO/src/main/java/com/example/NUBEPLAY\_USUARIO

**Inventario:**

NUBEPLAY\_INVENTARIO/NUBEPLAY\_INVENTARIO/src/main/java/com/example/NUBEPLAY\_INVENTARIO

### **10. Conclusión**

**Conclusión**

El desarrollo de los microservicios permitió aplicar los principios de arquitectura distribuida, mejorando la modularidad, escalabilidad y mantenibilidad del sistema. Cada servicio (usuarios, inventario y carrito) fue diseñado de forma independiente, respetando buenas prácticas de diseño backend con Spring Boot y JPA. Además, se logró una integración efectiva mediante pruebas con Postman y una correcta estructuración del repositorio en GitHub, cumpliendo con los requisitos del encargo.