Laboratorio 2: API de pedidos de cafetería

Objetivo general

El propósito de esta actividad es desarrollar un API REST en Java utilizando Spring Boot que gestione pedidos en una cafetería.
La aplicación deberá exponer métricas mediante Prometheus y visualizar dichas métricas en Grafana.

Objetivos específicos

- Diseñar un API REST simple para registrar y administrar pedidos.
- Exponer métricas de aplicación e infraestructura mediante Micrometer + Prometheus.
- Implementar monitoreo visual con Grafana.

Descripción general

El sistema a desarrollar representará el flujo de pedidos en una cafetería. Cada pedido deberá incluir información básica como el nombre del cliente, tipo de bebida, cantidad y estado.
El API permitirá realizar operaciones CRUD sobre los pedidos y deberá registrar métricas relevantes de negocio y rendimiento.

Requisitos funcionales

1. Creación de pedidos

Endpoint POST /api/orders

Recibe un cuerpo JSON con los campos:

```
{
  "customerName": "Ana",
  "drink": "latte",
  "quantity": 2
}
```

Retorna el pedido creado, incluyendo un identificador único (id) y la fecha de creación (createdAt).

2. Listado de pedidos

Endpoint GET /api/orders

Devuelve un arreglo JSON con todos los pedidos registrados.

3. Consulta individual

Endpoint GET /api/orders/{id}

Retorna el pedido correspondiente al identificador provisto.

4. Actualización de estado

Endpoint PATCH /api/orders/{id}/status/{status}

Permite modificar el estado del pedido (NEW, IN_PROGRESS, READY, DELIVERED, CANCELED).

5. Eliminación de pedido

Endpoint DELETE /api/orders/{id}

Elimina el pedido indicado si existe.

Requisitos no funcionales

1. Tecnológicos

Lenguaje: Java 17 o superior.

Framework: Spring Boot 3.0+.

Construcción: Maven.

Observabilidad: Spring Boot Actuator y Micrometer con Prometheus.

Visualización: Grafana.

Contenedorización: El código debe estar empaquetado en imágenes de Docker y desplegado en un cluster de Kubernetes.

2. Métricas y observabilidad

El API debe exponer un endpoint de métricas compatible con Prometheus en la siguiente ruta:

/actuator/prometheus

Las métricas deberán incluir:

- Métricas por defecto de la JVM y del servidor HTTP.
- Contadores personalizados:
  - coffee_orders_created_total → cantidad total de pedidos creados.
  - coffee_orders_delivered_total → cantidad total de pedidos entregados.

3. Integración con Prometheus y Grafana

Integrar los siguientes servicios:

- prometheus: configurado para hacer scraping del endpoint /actuator/prometheus.
- grafana: configurado para conectarse a Prometheus como fuente de datos.

La configuración de kubernetes deberá permitir scrapear las métricas cada 5 segundos.

4. Dashboards requeridos

En Grafana deberá visualizarse, como mínimo:

- Tasa de requests (RPS).
- Latencia promedio de endpoints.
- Cantidad de pedidos creados y entregados (coffee_orders_created_total, coffee_orders_delivered_total).
- Métricas de memoria JVM (jvm_memory_used_bytes).

Validación

Se considerará correcta la actividad cuando:

- El API responda correctamente a todos los endpoints definidos.
- Las métricas sean accesibles en /actuator/prometheus.
- Prometheus logre detectar y almacenar dichas métricas.
- Grafana muestre dashboards funcionales.

Criterio de evaluación

Bloque	Puntos	Porcentaje
Funcionalidad del API	30	30%
Calidad del código	15	15%
Exposición de métricas	10	10%
Integración con Prometheus	15	15%
Dashboard en Grafana	15	15%
Despliegue en Kubernetes	10	10%
Documentación y entrega	5	5%
Total	**100 puntos**	**100%**