**Desafío: Desarrollo de un microservicio en Java para un Asistente Virtual**

**Contexto:**

Imagina que eres parte de un equipo de desarrollo que trabaja en la creación de un asistente virtual para una empresa. Este asistente virtual debe ser capaz de recibir preguntas de los usuarios, procesarlas y devolver respuestas adecuadas. El asistente virtual estará compuesto por múltiples microservicios en Java, Node.js y Python, y se integrará con otros sistemas de backend.

En este desafío, se te pide que desarrolles uno o más microservicio en **Java 8** que cumpla con los siguientes requisitos.

**Requerimientos del Desafío:**

1. **Desarrollo del Microservicio:**
   * **Crea un microservicio RESTful en Java utilizando Spring Boot que exponga una API para gestionar solicitudes de usuarios.**
   * **La API debe recibir una solicitud HTTP POST con una pregunta del usuario en formato JSON y devolver una respuesta simulada, como si fuera un asistente virtual.**
   * **La respuesta debe ser un JSON con un campo respuesta y un contexto con la información de la conversacion (id, timestamp y otros datos que sean necesarios).**
2. **Simulación de Respuestas:**
   * **El microservicio debe simular el comportamiento de un asistente virtual, y para ello, debe tener una lógica que devuelva respuestas simples según ciertas palabras clave.**
   * **Por ejemplo, si el usuario pregunta "¿Cómo está el clima?", el microservicio puede devolver una respuesta como "El clima está soleado". Si la pregunta es "¿Qué hora es?", debe devolver la hora actual.**
3. **Manejo de Excepciones:**
   * **El microservicio debe ser capaz de manejar preguntas que no se puedan responder, devolviendo una respuesta con un mensaje de error (por ejemplo, "No entiendo la pregunta").**
   * **La API debe ser robusta y manejar adecuadamente errores como entradas inválidas o problemas internos.**
4. **Integración con otros microservicios (Opcional):**
   * **En el futuro, este microservicio podría integrarse con otros microservicios que se encargan de la consulta de datos en tiempo real (por ejemplo, consultar el clima o la hora desde una API externa).**
   * **Si lo deseas, puedes añadir una integración ficticia con un microservicio externo para mostrar cómo se realizaría la consulta a un servicio remoto.**
5. **Pruebas Unitarias:**
   * **Crea pruebas unitarias utilizando JUnit para asegurar que tu microservicio funciona correctamente en varios casos (respuestas válidas, manejo de errores, etc.).**
6. **Documentación:**
   * **Asegúrate de documentar cómo funciona el microservicio, las rutas que expone, los parámetros que acepta y los posibles resultados que devuelve. Utiliza Swagger.**
   * **Se deberá documentar y justificar la arquitectura de la solución ideada.**
   * **Diseñar un diagrama de componentes y/o integraciones de una posible solución identificando las funciones y responsabilidades de cada uno y cómo sería la comunicación entre sí.**
   * **Elegir un caso de uso (un flujo conversacional del asistente) y explicar con un sencillo diagrama de secuencia cómo sería la interacción entre componentes**

**Opcional (Extras):**

1. **Autenticación y Seguridad:**
   * **Implementa una capa de autenticación básica utilizando JWT o algún otro mecanismo de seguridad para proteger el acceso a la API del microservicio.**

Aclaración: Para los puntos 2 y 3 se pueden usar herramientas que manejen flujos conversacionales (ibm watson es un plus) o bien tenerlos en un archivo de configuración.

**Fecha de entrega:** 17/02/2025