Everis Bootcamp Microservicios

Proyecto I

El sistema a desarrollar está planteado en el contexto del negocio bancario que a medida que se va avanzando en los proyectos, se irá ampliando en base a este mismo proyecto.

Bases a Desarrollar

- Desarrollo de microservicios con Java 8.
- Utilizar Spring Boot Webflux como framework base.
- El proyecto debe utilizar Maven como manejadores de dependencias.
- Los microservicios proporcionados deben implementar controladores REST.
- Usar el patrón database per service, por lo que un microservicio no puede tocar ninguna tabla o colección que utilice otro microservicio.
- Utilizar inyección de dependencias.
- Utilizar propiedades de configuración externalizadas con un Config Server.
- Los nombres de las clases, métodos y las URLs deberán estar en inglés.
- La base de datos a utilizar será MongoDB.
- Uso de Lombok para reducir código.
- Manejo de trazas con Logback y utilizar el nivel del log adecuado.

Funcionalidades del sistema

- El sistema debe manejar la información de los clientes de un banco.
- Los clientes del banco son de dos tipos: personal o empresarial.
- El sistema debe manejar la información de los siguientes productos que ofrece el banco:
    - Pasivos (cuentas bancarias)
        - Ahorro: libre de comisión por mantenimiento y con un límite máximo de movimientos mensuales.
        - Cuenta corriente: posee comisión de mantenimiento y sin límite de movimientos mensuales.
        - Plazo fijo: libre de comisión por mantenimiento, solo permite un movimiento de retiro o depósito en un día específico del mes.
    - Activos (créditos)
        - Personal: solo se permite un solo crédito por persona.
        - Empresarial: se permite más de un crédito por empresa.
        - Tarjeta de Crédito personal o empresarial.
- Un cliente personal solo puede tener un máximo de una cuenta de ahorro, una cuenta corriente o cuentas a plazo fijo.
- Un cliente empresarial no puede tener una cuenta de ahorro o de plazo fijo pero sí múltiples cuentas corrientes.
- Las cuentas bancarias empresariales pueden tener uno o más titulares y cero o más firmantes autorizados.
- Un cliente puede tener un producto de crédito sin la obligación de tener una cuenta bancaria en la institución.
- Un cliente puede hacer depósitos y retiros de sus cuentas bancarias.
- Un cliente puede hacer pagos de sus productos de crédito.

- Un cliente puede cargar consumos a sus tarjetas de crédito en base a su límite de crédito.
- El sistema debe permitir consultar los saldos disponibles en sus productos como: cuentas bancarias y tarjetas de crédito.
- El sistema debe permitir consultar todos los movimientos de un producto bancario que tiene un cliente.

Requerimientos no funcionales obligatorios del sistema

- Elaborar y mantener un diagrama en draw.io con el diseño de la solución.
- El repositorio de datos deberá estar en documentos NoSQL.
- Para el manejo de datos se deberá utilizar Spring Data y no se deberá manejar la creación de SQL dinámicos y evitar el uso de la anotación @Query.
- Para todas las entidades de negocio se debe implementar sus operaciones CRUD: Create, FindAll, Update, Delete.
- Crear los endpoints REST para cada una de las operaciones de los repositorios.
- Utilizar los lineamientos REST para las operaciones CRUD.
- El sistema no tendrá implementado ninguna interfaz gráfica, la verificación de las funcionalidades se realizarán utilizando Postman.

Recomendaciones y Consideraciones

- Realicen primero las funcionalidades obligatorias.
- Realicen primero las funcionalidades opcionales más sencillas.
- No deben tener configuraciones en el código.
- Las clases y los métodos deben estar comentados.
- El uso de lambdas y streams de Java 8 es deseable.
- Deben subir su código a un repositorio git en github.
- Cada grupo deberá presentar su propia solución.

Artefactos y entregables.

- Todos los días laborables deberán subir el avance de su desarrollo a sus repositorios en github.
- Cada microservicio deberá tener su propio repositorio.
- La entrega del código de este proyecto tiene como fecha fin de entrega el lunes 21 de junio hasta las 18 horas con el código que esté en ese momento en el repositorio remoto.