Resposta das Questões A e B

a) Explique o conceito de configuração externalizada e centralizada.

I - Configuração Externalizada

Configuração externalizada é o padrão de arquitetura de software que consiste em mover todos os parâmetros de configuração de uma aplicação para fora do seu código-fonte e do seu pacote de implantação (JAR ou WAR). Isso inclui credenciais de banco de dados, chaves de API, URLs de serviços externos, limites de *timeouts* e quaisquer outras propriedades que possam variar entre ambientes ou que precisem ser alteradas sem a necessidade de recompilar e reimplantar o código.

Benefícios:

- **Portabilidade:** O mesmo artefato de *build* (o JAR) pode ser promovido através de todos os ambientes (desenvolvimento, homologação, produção).
- **Segurança:** Credenciais sensíveis não ficam embutidas no código-fonte, facilitando a aplicação de práticas de segurança como criptografia e gerenciamento de acesso.
- **Flexibilidade:** Permite que a configuração seja gerenciada por equipes de Operações (Ops) ou DevOps, separando-a do ciclo de vida de desenvolvimento (Dev).

II - Configuração Centralizada

Configuração centralizada é a prática de armazenar as configurações externalizadas de **todos** os microserviços de um sistema distribuído em um único servidor ou repositório de fácil acesso. O **Spring Cloud Config Server** atua como esse ponto central, servindo as configurações para os clientes (os microserviços) sob demanda.

- **Consistência Garantida:** Todos os microserviços em um ambiente específico (ex: Produção) buscam suas configurações da mesma fonte, eliminando o risco de inconsistências e erros de configuração entre serviços.
- **Gestão Simplificada:** Em vez de gerenciar arquivos de configuração em dezenas ou centenas de microserviços, a gestão é feita em um único repositório central (geralmente Git).
- **Versionamento e Auditoria:** O uso do Git como backend oferece um histórico completo de todas as alterações de configuração, permitindo rastreabilidade, rollbacks rápidos e atendendo a requisitos de compliance e auditoria.

b) Por que é importante ter um Config Server em um sistema bancário com múltiplos ambientes (dev, homologação, produção)?

Em um **banco digital**, a **confiabilidade**, a **segurança** e a **agilidade** são requisitos não negociáveis. A arquitetura de microserviços, comum nesse setor, amplifica a necessidade de um Config Server:

- **Consistência entre ambientes:** Garante que cada ambiente (Dev, QA/Homologação, Produção) use automaticamente as configurações corretas (bancos de dados, APIs, limites, segurança), com versionamento no Git para rastreabilidade e compliance regulatório.
- **Agilidade e zero downtime:** Permite alterar configurações em tempo real (rate limits, timeouts, feature flags) sem reiniciar os microserviços, essencial para operações 24/7 e respostas rápidas a picos, incidentes ou mudanças regulatórias.
- **Segurança de dados sensíveis:** Criptografa senhas, chaves e tokens no repositório Git, descriptografando apenas em runtime, evitando exposição em texto claro e atendendo padrões rigorosos de segurança financeira.
- **Menos erros em deploys:** Centraliza todas as configurações, eliminando o risco de implantar um serviço com propriedades erradas do ambiente e simplificando o pipeline CI/CD.

Em resumo, o Spring Cloud Config é uma peça-chave na arquitetura de um banco digital, pois transforma a gestão de configurações de um processo manual e propenso a erros em um processo **automatizado, seguro, versionado e dinâmico**.