Em quais situações se deve optar por um sistema monolítico e quando deve-se optar por um sistema com microsserviços?
Uma implementação de um sistema monolítico pode ser interessante para sistemas em estágio inicial, para testar provas de conceito, para sistemas que precisam ser desenvolvidos rapidamente e para sistemas de pequeno porte.
Microsserviços, por outro lado, são adequados para sistemas de grande porte, com alta quantidade de acessos e portanto necessidade de redundância, tolerância a falhas e escalabilidade.

Temos que construir um novo endpoint em nossa API que participa de um fluxo crítico de pagamentos com o aplicativo MercadoPago via QR nas lojas, informando ao microsserviço responsável pela cobrança do percentual de desconto que deve ser aplicado ao valor total.
Como estamos participando de um fluxo crítico, o aplicativo que nos liga pede que respondamos a eles em menos de 500 ms, incluindo suas novas tentativas, e que mantenhamos uma Taxa de erro <0,5% para não afetar a experiência do comprador na loja.

Para resolver isso, devemos consultar uma api chamada discounts-api que nos informa de todos os descontos que se aplicam ao cliente que está pagando, e entrega o mais conveniente para ele.

A referida api tem o seguinte SLA de resposta

- **Tempo de resposta**
  - **Média: 40 ms**
  - **P95: 150 ms**
  - **P99: 300 ms**
  - **MAX: 400 ms**
- **Taxa de erro: 0,9%**

Elabore uma proposta para os valores de configuração do restclient e explique por que cada valor:

- **Timeout**

  Pode-se definir um timeout de 80 ms, que corresponde ao dobro do tempo médio da resposta. Intuitivamente, esse valor tenderá a ser suficiente para a maioria dos casos, sobretudo considerando a garantia de até 300ms em 99% dos casos. Se esse tempo for excedido, podemos realizar até 6 novas tentativas sem exceder os 500 ms de processamento.
- **Retries**
  - **Quantidade**

    Pelo menos dois, de 80ms, visando diminuir a taxa de erro.
  - **Estratégia**

    Estratégia simples, pois a incidência de erros definida no SLA é baixa. Assim, se acontecer uma falha, podemos tentar novamente em seguida, pois a incidência de erro em muitos clientes será baixa.

Supondo que nosso microsserviço consuma outro que quando entre o código de status que pode retornar é 429, antes do qual tentamos 2 vezes (máximo 3 solicitações, o original mais 2 tentativas), e finalmente retornamos 500, porque sem esse valor não podemos prosseguir.

Cada vez que o serviço que consumimos retorna 429 por mais de um minuto, somos alertados porque o tempo médio de resposta sobe tendendo ao infinito (60 segundos!) E a taxa de erro sobe para 100%. O problema que temos é que quando o serviço que consumimos é restabelecido, o nosso não, mantendo horários elevados até que apaguemos (corte o tráfego de entrada e substitua todas as instâncias).

Inferir por que isso acontece e propor uma alteração na política de repetição e no código de status que nosso microsserviço retorna.

Uma possível causa para o problema é o nosso tempo de retry após ocorrências de Too Many Requests ser muito curto. Se esperarmos um pouco mais antes de uma nova tentativa, podemos obter a resposta esperada em menos tempo do que levaríamos fazendo inúmeros retries sucessivos. Uma possível solução, então, seria aumentar o tempo de retry exponencialmente, caso o erro seja 429. Assim, aumentaremos gradativamente o tempo de espera antes de refazer a requisição, para diminuir a chance de que uma nova tentativa obtenha o mesmo erro novamente.