O que fazer para mitigar possíveis erros e controlar os possíveis erros recebidos da API?

Existem algumas maneiras e boas práticas para mitigar e controlar esses possíveis erros, uma delas é ter logs completos tendo muito claro qual erro aconteceu e onde aconteceu, facilitando o processo de correção, isso vale tanto para falhas, tanto para sucessos nas requisições, este processo combinado com as ferramentas que geram relatórios de erros, assim todos conseguem ser notificados o quanto antes possível dos erros que está acontecendo. Uma outra forma para mitigar possíveis erros é ter a API sendo testada por testes unitários e de integração, por pelo menos 80% do projeto, dando mais segurança a todos. Além de monitorar o tempo de respostas das requisições, sendo o aumento do tempo de resposta um ótimo parâmetro para possíveis erros. Documentar o projeto e configurar as respostas de erros também ajudará a resolver erros posteriores.

Quais boas práticas são aplicadas em banco de dados e no código para garantir performance?

O melhor jeito para se garantir a performance em banco de dados e no código é otimizando suas querys e códigos, no código o simples ato de fazer com que cada método tenha uma só função já faz com que melhore a performance do código, além da duplicidade de códigos, documentação do projeto, variáveis bem definidas, fluxo da lógica de regras de negócios estabelecidos e simples de entender, métodos refatorados, ter testes unitários e de integração no projeto, e rodar o código em um software sonar antes de ser testado, isso e mais outras centenas de boas práticas aumenta o desempenho do código, também uma boa prática é manter o mesmo server para o banco de dados, uma estrutura bem planejada é o começo de tudo, se começar errado, vai terminar errado, manter os dados organizados e adequados a LGPD, otimização de querys, e um profissional especializado em banco de dados(DBA) na equipe.

Como garantir segurança para as APIs do sistema?

Existem diversas maneiras de garantir a segurança de uma API, algumas delas são por meio de tokens de acesso, apenas quem tem o token conseguiria fazer uma requisição, gateway de API, para controlar o tráfego da mesma, o uso de criptografia nos dados fazendo que somente usuários autenticados possam descriptografar esses dados e estabelecer cotas de chamadas, evitando com que a API ultrapasse certo número de requisições por certo intervalo de tempo, protegendo de ataques DNS.

Como trabalhar com simultaneidade se milhares de requisições forem solicitadas simultaneamente?

A simultaneidade é garantida através de uma infraestrutura bem elaborada. Desde o teste de carga na sua aplicação para saber como ela se comporta com um número elevado de solicitações, até um load balance que garante o aumento de processamento do servidor de acordo com o aumento de requisições, sem se esquecer que o banco de dados deve suportar a quantidade elevada de consultas. É essencial que sua aplicação trabalhe com multithread para que tenha a capacidade de processar todas as requisições simultaneamente.

Diagrama:

