Universidade Federal de Goiás

Instituto de Informática

Goiânia, 10 de outubro de 2021

Avaliação Arquitetura

Projeto

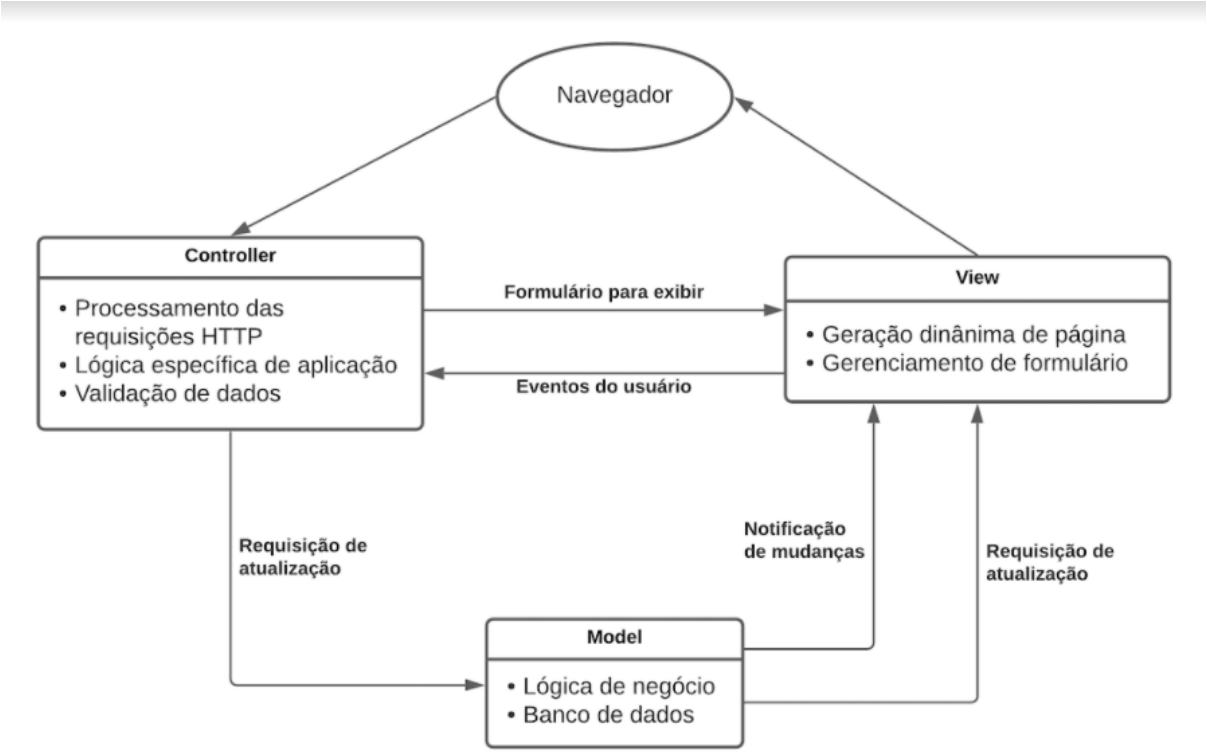
O projeto consiste em um sistema de cadastro de usuários e restaurantes de gerenciamento da relação de fidelidade entre os usuários e os restaurantes. Essa relação se dá através do que é chamado de cartão fidelidade, onde um cliente após consumir algumas vezes em um local tem direito a brindes ou descontos e para isso é necessário receber um carimbo toda vez que consome no local para atingir a contagem necessária.

Esse sistema é constituído por uma API que gerencia os dados e um Client via aplicativo e via navegador WEB. Sendo assim, era necessária uma arquitetura capaz de abstrair essa organização entre os sistemas e que possibilitasse uma versatilidade, usabilidade, manutenibilidade e portabilidade.

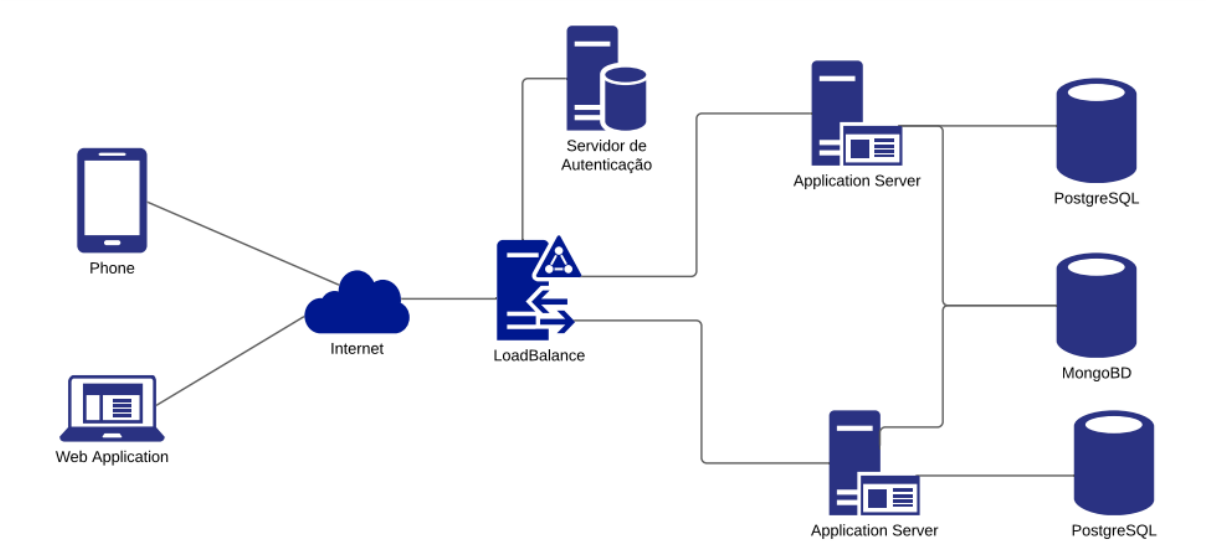
A necessidade de uma arquitetura que conseguisse prover um ecossistema capaz de ser escalável e de fácil manutenção vai de encontro com as necessidades das regras de negócio do produto. Isso acontece pelo fato de após a validação de MVP e algumas versões iniciais do sistema talvez seja necessário adaptação a alguma regra específica de mercado ou até mesmo ao gosto dos clientes. Sendo assim, uma arquitetura de Model-View-Controller atende bem essa situação.

Arquitetura

O padrão de arquitetura escolhido para o sistema é model-view-controller e a abstração geral desse processo é como o diagrama abaixo:



A abstração da aplicação do padrão de projeto MVC no sistema em questão com mais detalhamento segue como mostrado no diagrama abaixo:



As possibilidades levantadas por esse estilo de padrão de arquitetura respeitam as vontades e necessidades do Plano de Negócios do Produto. Isso porque o produto precisa estar apto a se adaptar de forma rápida e/ou sofrer algum tipo de alteração em seu funcionamento ou relacionamentos internos para atender alguma nova demanda.

Com isso, algumas relações entre as partes do padrão e necessidades do sistema podem ser vistas da seguinte forma:

Model	Camada responsável pela gestão e manutenção de dados. O fato de ser uma camada separada garante não só níveis de segurança como também possibilita facilidade de manutenção, uma vez que uma otimização ou refatoração necessária apenas na camada de dados pode ser feita de forma mais independente que em outras arquiteturas.
View	A View é toda a camada considerada que comunica diretamente com o usuário, isso também é um facilitador para isolar tanto problemas quanto melhorias. Além disso, o padrão MVC possibilidade a adoção de mais de uma View, ou seja, posso ter um aplicativo mobile e um navegador consumindo ao mesmo tempo dos serviços da Controller + Model
Controller	Na controller é conectada a comunicação entre Model e View, ou seja, a controller é responsável por pegar as ações do usuário na View e validar com as regras de negócio implementadas na Model fornecendo respostas de sucesso ou erro. Isso também atua como uma parte possível de melhoria e refatoração sem afetar o comportamento dos outros componentes e camadas do projeto.

Conclusão

Observando as necessidades do produto, da regra de negócio e os benefícios fornecidos pelo padrão arquitetural escolhido é visto que a escolha de utilizar o MVC é interessante para o sistema em questão já que consegue fornecer esses resultados a um custo de implementação, tempo, esforço e manutenção mais baixo que outras arquiteturas.

Essas métricas são válidas e satisfatórias pois para o produto em questão é necessário que a arquitetura possibilidade essas versatilidade sem despender de muito custo de tempo ou manutenção, pois dada a natureza de mudança do produto, um arquitetura um pouco mais engessada em padrões mais rígidos poderia ser quebrada numa tentativa de atender uma nova demanda imposta sobre a equipe de produto.

Arquiteturas mais robustas e fortemente ligadas entre as etapas internas, tendem a ofuscar problemas ou prolongar erros de origem simples caso não sejam respeitadas conforme suas documentações em processos específicos. Usar uma arquitetura um pouco mais flexível e de simples entendimento consegue remover isso da visão de qualquer membro do time de produto.