IFMT Cuiabá 2020, PSOO: Alberto Sales e Silva, Discentes: Gustavo Lara, Mateus Soares

1. Tecnologias

1.1 Dart

Dart é uma linguagem de script voltada à web desenvolvida pela Google. Ela foi lançada na GOTO Conference 2011, que aconteceu de 10 a 11 de outubro de 2011 em Aarhus, na Dinamarca. O objetivo da linguagem Dart foi inicialmente a de substituir a JavaScript como a linguagem principal embutida nos navegadores. Programas nesta linguagem podem tanto serem executados em uma máquina virtual quanto compilados para JavaScript.

Em novembro de 2013, foi lançada a primeira versão estável, Dart 1.0. Em agosto de 2018 foi lançado o Dart 2.0, um reboot da linguagem, otimizado para o desenvolvimento client-side para Web e dispositivos móveis.

1.2 Flutter

Flutter é um kit de desenvolvimento de interface de usuário (Ui Toolkit), de código aberto, criado pelo Google, que possibilita a criação de aplicativos compilados **nativamente**. Atualmente pode compilar para Android, iOS, Windows, Mac, Linux, Google Fuchsia[1] e Web.

1.3 WhatsApp

WhatsApp é um aplicativo multiplataforma de mensagens instantâneas e chamadas de voz para smartphones. Além de mensagens de texto, os usuários podem enviar imagens, vídeos e documentos em PDF, além de fazer ligações grátis por meio de uma conexão com a internet.

O WhatsApp está em constante evolução, como a maioria dos aplicativos do mercado, e vem sendo cada vez mais usado no mercado para vendas - já sendo considerado obrigatório no processo de vendas de qualquer empresa ou profissional. Não é à toa que a empresa criou também, posteriormente, o **WhatsApp Business**, feito para o mundo comercial, justamente para ajudar os usuários principalmente com vendas e propaganda.

2. Projeto

O projeto consiste em uma Loja Virtual, ela conta com uma parte administrativa, onde cada conta criada no app e usada para logar é uma loja, a loja conta com um número de WhatsApp, e ela pode anunciar seus produtos, com fotos, preço, quantidade disponível;
A parte pública do app, que não necessita de login, disponibiliza esse catalógo de produtos das lojas, e para realizar uma compra, após selecionar o produto e a quantidade, o app inicia uma conversa no WhatsApp da loja com o produto e quantidade desejada, e o restante da negociação como pagamento e entrega é finalizada lá.

3. Arquitetura

3.1 Flutter

Os principais componentes do Flutter incluem:

• Linguagem de programação Dart
• Flutter Engine
• Biblioteca Foundation
• Design-specific Widgets com implementações prontas para Android (Google Material) e iOS (Cupertino)

3.1.1 Flutter Engine

A engine do Flutter, escrito principalmente em C++, fornece suporte de renderização de baixo nível usando a biblioteca de gráficos Skia do Google. Além disso, ele faz interface com SDKs específicos da plataforma, como os fornecidos pelo Android e iOS.

O Flutter Engine é um runtime portátil para hospedar aplicativos em Flutter. Ele implementa as bibliotecas principais do Flutter, incluindo animação e gráficos, I/O de arquivos e rede, suporte à acessibilidade, arquitetura de plugins e um conjunto de ferramentas de tempo de execução e compilação do Dart. A maioria dos desenvolvedores irá interagir com o Flutter por meio do Flutter Framework, que fornece uma estrutura moderna e reativa e um rico conjunto de platform, layout e foundation widgets.

3.1.2 Biblioteca Foundation

A biblioteca Foundation, escrita em Dart, fornece classes e funções básicas que são usadas para construir aplicativos usando o Flutter, como APIs para se comunicar com a engine.

3.1.3 Design-specific Widgets

O framework Flutter contém dois conjuntos de widgets que estão em conformidade com linguagens de design específicas. Os widgets do Material Design implementam a identidade visual do Google e os widgets do Cupertino implementam as diretrizes de interface humana para iOS da Apple.

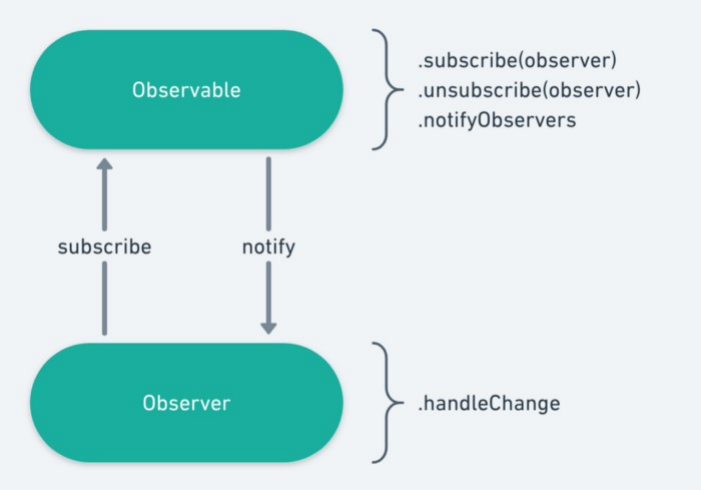
3.2 Projeto

O projeto feito em Flutter Web, vai utilizar o MobX, biblioteca para gerencia de estado, ele cria um Observable ao redor de um Widget que queremos manter sempre atualizado.

Neste Widget então, podemos referenciar uma propriedade marcada como @observable e toda vez que ela for alterada, seu resultado se reflete na tela. Você realiza uma ação (@action) para alterar um observável.

3.2.1 Padrão Observer

Dentro de nosso Widget, teremos o Observer, quando uma ação for realizada que altere o Observable, o Observer será notificado e atualizado.



3.2.2 Arquitetura Store

Repository: é usado para ser chamado dentro da @action no controller. O repositório pode pegar os dados da API e colocar no Store.
Store: mantem os dados da API, aqui o MobX opera, o Store contém as @observable e as @action
Widget: componente visual ou tela contendo o Observer, será atualizado quando houver alterações nos @observable