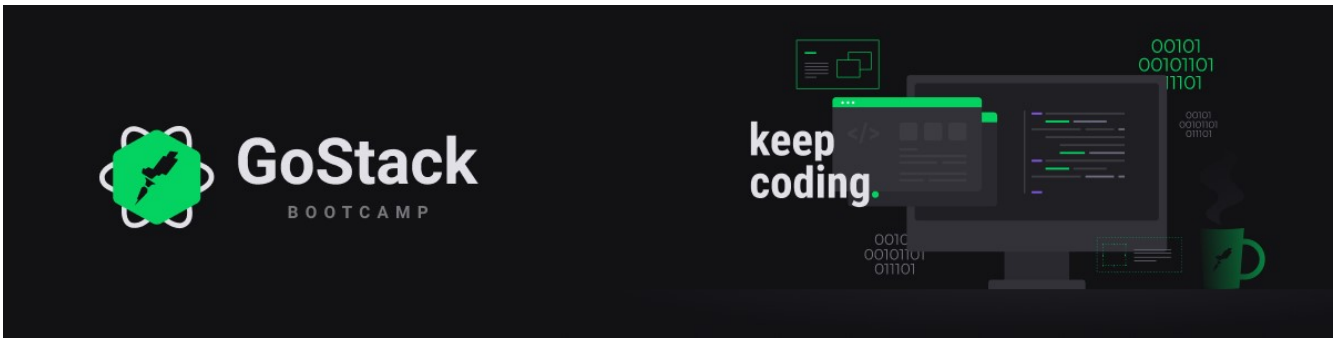
README.md



Desafio 08: Fundamentos do React Native

> "Não existe linha de chegada, a vitória está em se manter correndo"!

languages 0 made by Rocketseat license MIT Stars 1k

Sobre o desafio | Entrega | Licença

🚀 Sobre o desafio

Nesse desafio, você desenvolverá uma nova aplicação, a GoMarketplace. Dessa vez é hora de você praticar o que você aprendeu até agora no React Native junto com o TypeScript, utilizando rotas, Async Storage e a Context API.

Template da aplicação

Para te ajudar nesse desafio, criamos para você um modelo que você deve utilizar como um template do Github.

O template está disponível na seguinte url: **Acessar Template**

Dica: Caso não saiba utilizar repositórios do Github como template, temos um guia em **nosso FAQ**.

Agora navegue até a pasta criada e abra no Visual Studio Code, lembre-se de executar o comando `yarn` no seu terminal para instalar todas as dependências.

Utilizando uma fake API

Antes de tudo, para que você tenha os dados para exibir em tela, criamos um arquivo que você poderá utilizar como fake API para te prover esses dados.

Para isso, deixamos instalado no seu package.json uma dependência chamada `json-server`, e um arquivo chamado `server.json` que contém os dados para uma rota `/products`. Para executar esse servidor você pode executar o seguinte comando:

```
yarn json-server server.json -p 3333
```

Layout da aplicação

Essa aplicação possui um layout que você pode seguir para conseguir visualizar o seu funcionamento.

O layout pode ser acessado através da página do Figma, no seguinte link.

Você precisará de uma conta (gratuita) no Figma pra inspecionar o layout e obter detalhes de cores, tamanhos, etc.

Funcionalidades da aplicação

Agora que você já está com o template clonado e pronto para continuar, você deve verificar os arquivos da pasta `src` e completar onde não possui código, com o código para atingir os objetivos de cada rota.

- **`Listar os produtos da fake API`**: Sua página `Dashboard` deve ser capaz de exibir uma listagem através de uma tabela, com os campos `title`, `image_url` e `price`.

Dica: Você pode utilizar a função Intl para formatar os valores. Dentro da pasta `utils` no template você encontrará um código para te ajudar.

- **`Adicionar itens ao carrinho`**: Em toda sua aplicação, você deve utilizar o Contexto chamado `cart` que deixamos disponível. Você vai precisar completar as funcionalidades dentro de `hooks/cart.tsx` para que você consiga adicionar itens ao carrinho.

Dica: No seu contexto de carrinho, utilize uma propriedade chamada `quantity` para controlar quantos desse item existem no seu carrinho.

Dica 2: Caso um produto que você está adicionando já exista no carrinho, apenas altere a quantidade dele no seu contexto para evitar itens duplicados.

- **`Exibir itens do carrinho`**: Na página `Cart` você deve exibir todos os itens do carrinho, junto com a quantidade, valor único, valor subtotal dos itens e total de todos os items.

-

`Aumentar quantidade de itens do carrinho`: Na página `Cart` você deve permitir que o usuário aumente a quantidade de itens do mesmo produto, para isso você pode utilizar a função `increment` dentro do seu contexto em `/src/hooks/cart.tsx`.

- `Diminuir quantidade de um item do carrinho`: Na página `Cart` você deve permitir que o usuário decremente a quantidade de itens do mesmo produto, para isso você pode utilizar a função `decrement` dentro do seu contexto em `/src/hooks/cart.tsx`.

- `Exibir valor total dos itens no carrinho`: Tanto na página `Dashboard`, quanto na página `Cart` você deve exibir o valor total de todos os itens que estão no seu carrinho.

Especificação dos testes

Em cada teste, tem uma breve descrição do que sua aplicação deve cumprir para que o teste passe.

Caso você tenha dúvidas quanto ao que são os testes, e como interpretá-los, dé uma olhada em **nosso FAQ**.

Para esse desafio, temos os seguintes testes:

- `should be able to list the products`: Para que esse teste passe, sua aplicação deve permitir que sejam listados na sua tela `Dashboard`, todos os produtos que são retornadas do Fake API. Essa listagem deve exibir o `title` e o `price` que deve ser formatado utilizando a função `Intl`.

- `should be able to add a product to the cart`: Para que esse teste passe, você deve permitir que seja possível adicionar produtos da sua `Dashboard` ao carrinho, utilizando o contexto de `cart` disponibilizado.

- `should be able to list the products on the cart`: Para que esse teste passe, você deve permitir que seja possível listar os produtos que estão salvos no contexto do seu carrinho na página `Cart`, nessa página exiba o nome do produto e o subtotal total de cada produto (price * quantity).

- `should be able to calculate the cart total`: Para que esse teste passe, tanto na página `Dashboard`, tanto na página `Cart` você deve exibir o valor total de todos os itens que estão no seu carrinho.

Dica: Para calcular o total de todos os itens, você pode utilizar o reduce para somar todos os valores e retornar o valor total.

- `should be able to show the total quantity of itens in the cart`: Para que esse teste passe, tanto na página `Dashboard`, tanto na página `Cart` você deve exibir o número total de itens que estão no seu carrinho.

Dica: Para calcular o total de todos os itens, você pode utilizar o reduce para somar todos os valores e retornar o valor total.

Dica 2: Utilize o useMemo para armazenar o valor total do carrinho que você calculou.

-

`should be able to increment product quantity on the cart`: Para que esse teste passe, você deve permitir que seja possível incrementar a quantidade de um produto do seu carrinho, utilizando o contexto de `cart` disponibilizado.

- `should be able to decrement product quantity on the cart`: Para que esse teste passe, você deve permitir que seja possível decrementar a quantidade de um produto do seu carrinho, utilizando o contexto de `cart` disponibilizado.

Dica: Ao decrementar a quantidade de um produto, não permita que ele seja decrementado para um valor negativo, sendo a quantidade mínima 1 para estar no carrinho.

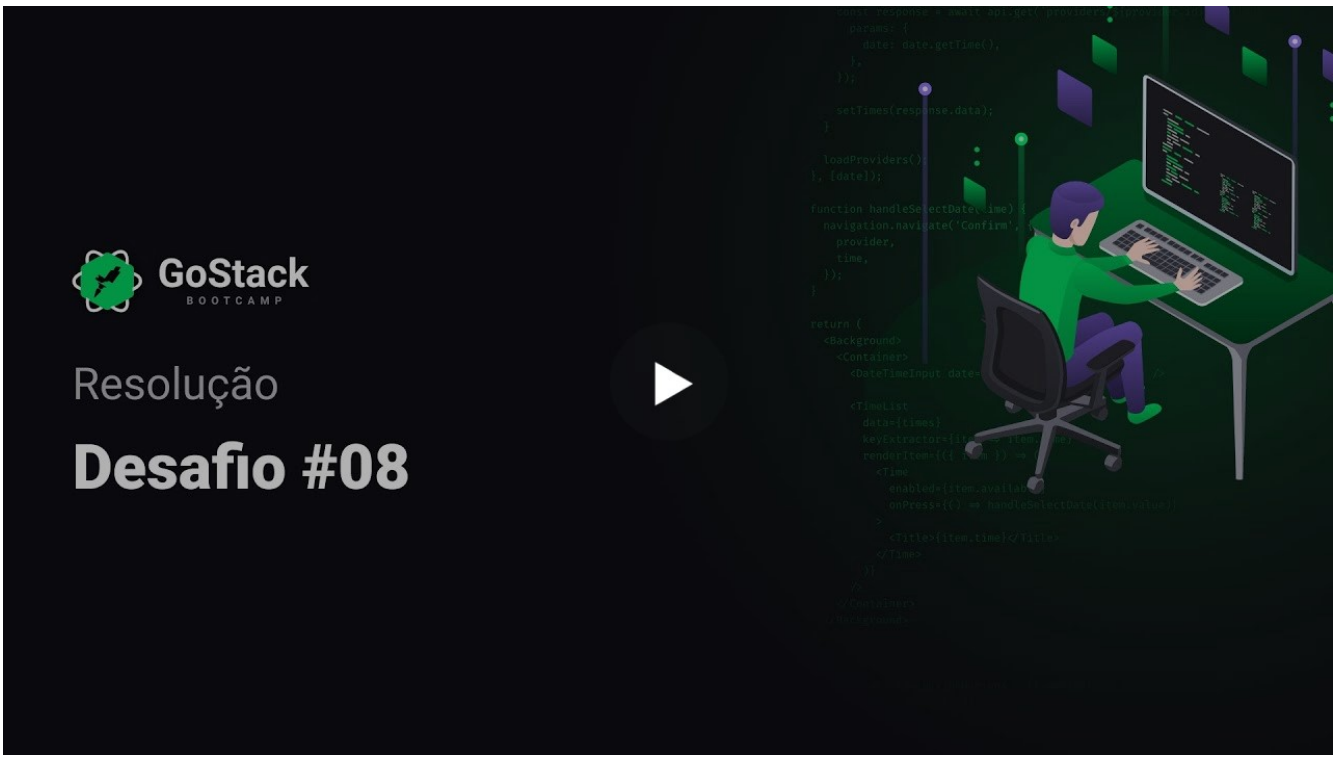
- `should be able to navigate to the cart`: Para que esse teste passe, no seu componente `FloatingCart` na Dashboard, você deve permitir que ao clicar no botão de carrinho com o testID de `navigate-to-cart-button`, o usuário seja redirecionado para a página `Cart`.
- `should be able to add products to the cart`: Para que esse teste passe, no seu arquivo onde contém o contexto do carrinho, você deve permitir que a função `addToCart` adicione um novo item ao carrinho.
- `should be able to increment quantity`: Para que esse teste passe, no seu arquivo onde contém o contexto do carrinho, você deve permitir que a função `increment` incremente em `1` unidade a quantidade de um item que está armazenado no contexto.
- `should be able to decrement quantity`: Para que esse teste passe, no seu arquivo onde contém o contexto do carrinho, você deve permitir que a função `decrement` decremente em `1` unidade a quantidade de um item que está armazenado no contexto.
- `should store products in AsyncStorage while adding, incrementing and decrementing`: Para que esse teste passe, no seu arquivo onde contém o contexto do carrinho você deve permitir que todas as atualizações que você fizer no carrinho, sejam salvas no AsyncStorage. Por exemplo, ao adicionar um item ao carrinho, adicione-o também no AsyncStorage. Lembre de também atualizar o valor do AsyncStorage quando você incrementar ou decrementar a quantidade de um item.
- `should load products from AsyncStorage`: Para que esse teste passe, no seu arquivo onde contém o contexto do carrinho, você deve permitir que todos os produtos que foram adicionados sejam buscados do AsyncStorage.

📅 Entrega

Esse desafio deve ser entregue a partir da plataforma da Rocketseat, envie o link do repositório que você fez suas alterações. Após concluir o desafio, fazer um post no Linkedin e postar o código no Github é uma boa forma de demonstrar seus conhecimentos e esforços para evoluir na sua carreira para oportunidades futuras.

Solução do desafio

Caso você queira ver como resolver o desafio, fizemos um vídeo explicando o passo a passo para cumprir com todos os requisitos da aplicação:



Licença

Esse projeto está sob a licença MIT. Veja o arquivo LICENSE para mais detalhes.

Feito com 💜 by Rocketseat 👋 Entre na nossa comunidade!