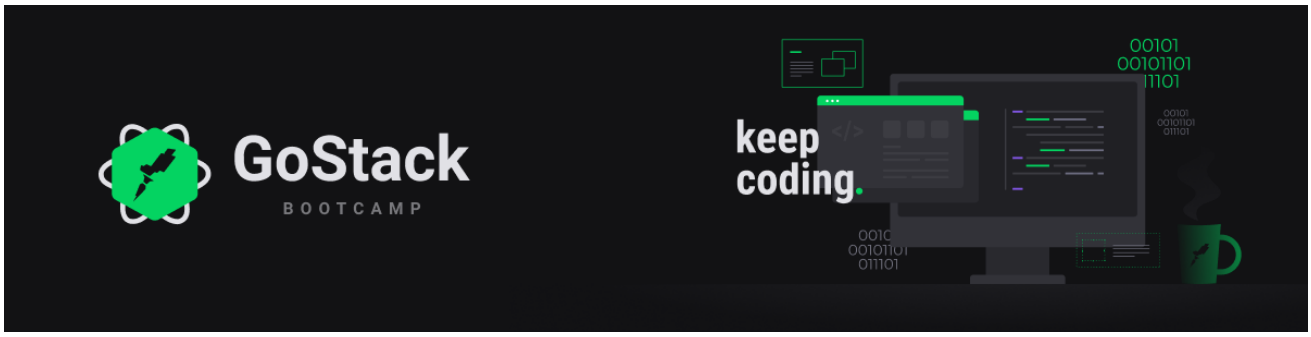
danileao ···		on 20 Aug
..		
assets		last month
README.md		last month

README.md



Desafio 11: GoRestaurant Mobile

> “Nada no mundo supera a persistência.”!

languages 0 | made by Rocketseat | license MIT | Stars 814

Sobre o desafio | Entrega | Licença

🚀 Sobre o desafio

Nesse desafio, você irá desenvolver mais uma aplicação, a GoRestaurant, só que dessa vez a versão mobile para o cliente. Agora você irá praticar o que você aprendeu até agora no React Native junto com TypeScript, para criar um pequeno app para pedidos de comida.

Essa será uma aplicação que irá se conectar a uma Fake API, e exibir e filtrar os pratos de comida da API e permitir a criação de novos pedidos.

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 as seguintes rotas:

Rota `/foods` : Retorna todas as comidas cadastradas na API

Rota `/foods/:id` : Retorna um prato de comida cadastradas na API baseado no `id`

Rota `/categories` : Retorna todas as categorias cadastradas na API

Rota `/orders` : Retorna todas os pedidos que foram cadastrados na API

Rota `/favorites` : Retorna todas as comidas favoritas que foram cadastrados na API

```
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 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 pratos de comida da sua API`** : Sua página `Dashboard` deve ser capaz de exibir uma listagem, com o campo `name` , `value` e `description` de todos os pratos de comida que estão cadastrados na sua API.

- **`Listar as categorias da sua API`** : Sua página `Dashboard` deve ser capaz de exibir uma listagem, com o campo `title` e `image_url` de todas as categorias que estão cadastrados na sua API.

Dica: O campo thumbnail_url será utilizada como imagem da categoria, deixamos três categorias como exemplo no arquivo server.json.

- **`Filtrar pratos de comida por busca ou por categorias`** : Em sua página Dashboard permitir que o input de pesquisa e os botões de categoria façam uma busca na API de acordo com o que estiver selecionado ou escrito no input.

- **`Listar os pedidos da sua API`** : Sua página `Orders` deve ser capaz de exibir uma listagem, com o campo as informações do produto pedido, com `name` e `description` de todos os pedidos que estão cadastrados na sua API.

Dica: Por se tratar de uma Fake API e de não possuir usuários, não será necessário cadastrar o campo `user_id` , considere que deve ser listados todos os pedidos da API como se fossem os seus pedidos.

- **`Listar os pratos favoritos da sua API`** : Sua página `Favorites` deve ser capaz de exibir uma listagem, com o campo as informações do produto favorito, com `name` e `description` de todos os pedidos que estão cadastrados na sua API.

Dica: Por se tratar de uma Fake API e de não possuir usuários, não será necessário cadastrar o campo `user_id` , considere que deve ser listados todos os favoritos da API como se fossem os seus favoritos.

- **`Realizar um pedido`** : Na sua página `Dashboard` , ao clicar em um item, você deve redirecionar o usuário para a página `FoodDetails` , onde será possível realizar um novo pedido, podendo controlar a quantidade desse item pedido, ou adicionar ingredientes extras. Todo o valor deve ser calculado de acordo com a quantidade pedida.

Dica: Você pode usar o método `reduce` para somar o valor de todos os extras pedidos e somá-lo com o valor do prato de comida. Depois disso lembre-se de multiplicar tudo pela quantidade pedida do produto.

Especificação dos testes

Em cada teste, tem uma breve descrição no 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 food plates`** : Para que esse teste passe, sua aplicação deve permitir que sejam listados na sua `Dashboard` , todos os pratos de comidas que são retornados da sua fake API.

- **`should be able to list the food plates filtered by category`** : Para que esse teste passe, sua aplicação deve permitir que sejam listados na sua `Dashboard` , os pratos de comidas filtrados por categoria da sua fake API.

Dica: No json-server você pode usar os query params normalmente para buscar/filtrar de acordo com o valor de um campo do item no arquivo server.json. Por exemplo, para filtrar todos os pratos com a categoria 1, a sua url ficaria como `http://localhost:3000/foods?category_like=1` .

- **`should be able to list the food plates filtered by name search`** : Para que esse teste passe, sua aplicação deve permitir que sejam listados na sua `Dashboard` , os pratos de comidas filtrados por nome da sua fake API.

Dica: No json-server você pode usar os query params normalmente para buscar/filtrar de acordo com o valor de um campo do item no arquivo server.json. Por exemplo, para filtrar todos os pratos com o nome Veggie, a sua url ficaria como `http://localhost:3000/foods?name_like=Veggie` .

- **`should be able to navigate to the food details page`** : Para que esse teste passe, em sua `Dashboard` , você deve permitir que ao clicar em um item, seja navegado para a página `FoodDetails` passando por parâmetro da navegação o id do item clicado.

- **`should be able to list the favorite food plates`** : Para que esse teste passe, sua aplicação deve permitir que sejam listados na sua página `Favorites` , todos os pratos de comidas que estão salvos na rota `favorites` .

- **`should be able to list the orders`** : Para que esse teste passe, sua aplicação deve permitir que sejam listados na sua página `Orders` , todos os pratos de comidas que estão salvos na rota `orders` .

- **`should be able to list the food`** : Para que esse teste passe, sua aplicação deve permitir que seja listado todos os dados de uma comída específica na página `FoodDetails` , baseado no id recuperado pelos parametros da rota.

Dica 1: Use o hook `useRoute` para pegar o atributo `params` que conterá o id enviado por parametro.

Dica 2: Com o id recuperado por parametro da navegação, faça uma busca na sua API com os dados atualizados da food na roda `/foods/:id` .

- **`should be able to increment food quantity`** : Para que esse teste passe, você deve permitir que seja incrementada em 1 a quantidade do item na página `FoodDetails` .
- **`should be able to decrement food quantity`** : Para que esse teste passe, você deve permitir que seja decrementada em 1 a quantidade do item na página `FoodDetails` .

Dica: Não permita que o valor do input seja alterado para menor que 1, para que assim o pedido sempre tenha no mínimo um item.

- **`should not be able to decrement food quantity below than 1`** : Para que esse teste passe, você deve impedir que seja decrementado a quantidade de itens até um número menor que 1, assim o número mínimo de itens no pedido é 1.
- **`should be able to increment an extra item quantity`** : Para que esse teste passe, você deve permitir que seja incrementada em 1 a quantidade de um ingrediente extra na página `FoodDetails` baseado no seu id.
- **`should be able to decrement an extra item quantity`** : Para que esse teste passe, você deve permitir que seja decrementado em 1 a quantidade de um ingrediente extra na página `FoodDetails` baseado no seu id.

🚀 Expandindo os horizontes

Essa é uma aplicação totalmente escalável, isso significa que além das especificações, após finalizado o desafio, é totalmente possível implementar mais funcionalidades a essa aplicação, e essa será uma ótima maneira para fixar os conhecimentos.

Você pode implementar desde funcionalidades simples que não foram específicadas nos testes, como a finalização completa de um pedido, ou uma página que irá mostrar dados do pedido realizado.

Por exemplo, tente implementar uma página contendo o número de itens e os ingredientes extras que foram adicionados à um pedido, essa página será aberta a partir da tela `Orders` ao clicar em algum pedido.

Além disso, use sua criatividade para testar novas coisas, existem muitas possibilidades de aprendizado! 🚀

📆 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!