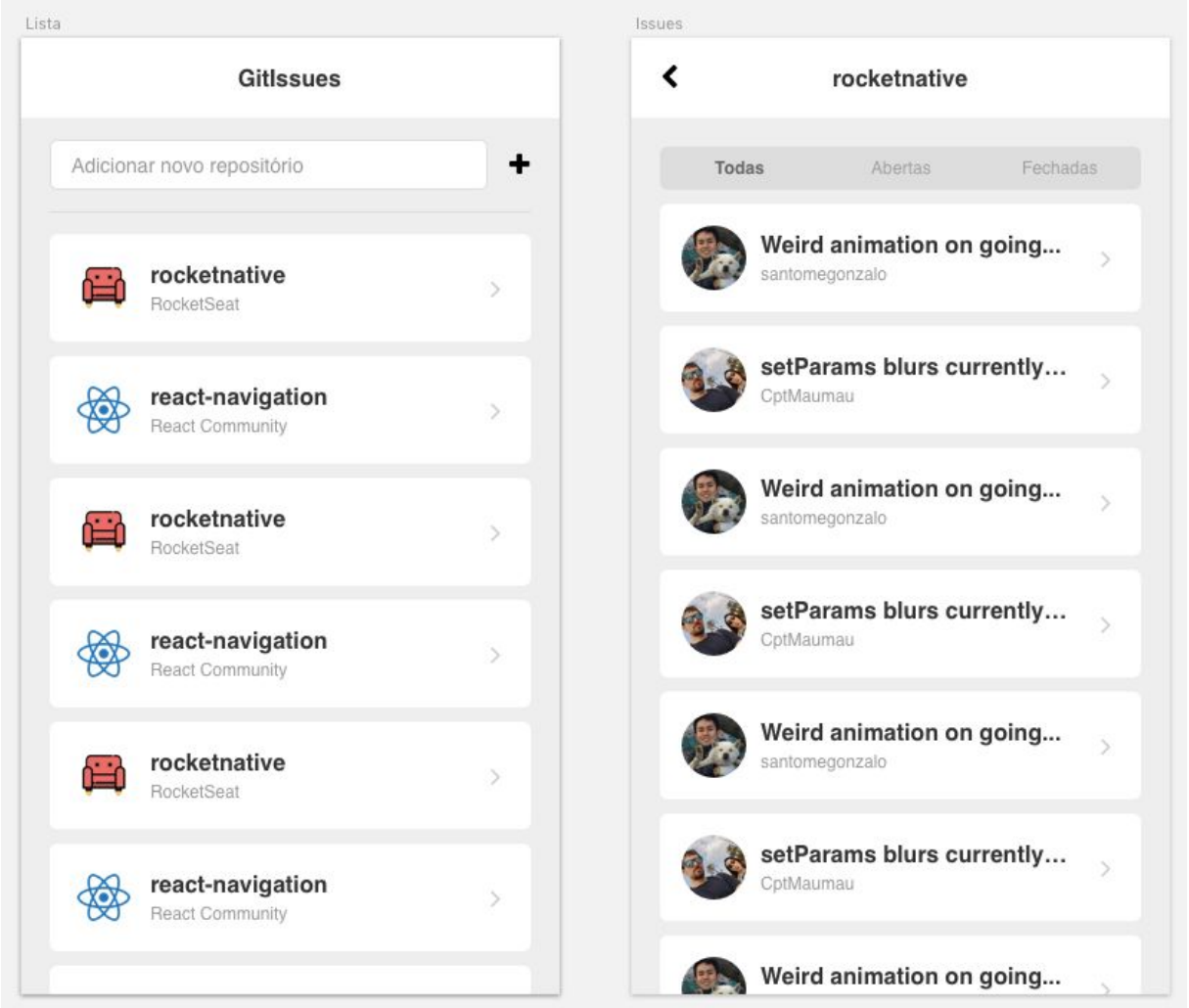
Desafio: Módulo 02

Crie uma aplicação do zero e configura as ferramentas: **ESLint, Reactotron, Babel Module Resolver e EditorConfig**. Nesse desafio você irá construir uma aplicação utilizando a API do Github para listar issues (questões) de um repositório. A interface da aplicação deve ser construída utilizando FlexBox.

O app permitirá ao usuário inserir o nome de um repositório existente que será exibido na lista da primeira tela e clicando sobre os repositórios, poderá listar as issues pertencentes ao mesmo, além disso pode filtrar entre issues Abertas, fechadas ou todas. As duas páginas do app devem ser estilizadas como as seguintes imagens:



Regras:

1. O input de adicionar repositório deve receber a informação de organização/repositório, exemplo: “rocketseat/rocketseat.com.br”;
2. Deve ser possível adicionar repositório de uma organização e usuário também (exemplo: diego3g/rocketnative (usuário) e rocketseat/rocketseat.com.br (organização)).
3. Ao clicar no botão “+” uma request será enviada à API do Github buscando informações do repositório e armazenando os campos ID, nome, organização e avatar no storage (AsyncStorage) do dispositivo;
4. A lista de repositórios adicionada deve ser mantida no AsyncStorage em forma de array e recuperada ao inicializar a aplicação exibindo os dados em tela.
5. O usuário deve poder atualizar a lista de repositórios arrastando a lista pra baixo com a opção refresh do <FlatList />
6. Ao clicar em um repositório, o usuário deverá ser navegado para a tela de issues do repositório e só nesse momento carregar as issues da API **(não armazene as issues no AsyncStorage).**
7. Deve ser possível filtrar entre issues abertas, fechadas e todas (por padrão).
8. A informação do filtro também deve ser guardada no AsyncStorage e recuperada assim que essa tela for aberta mantendo a preferência do usuário. O filtro não precisa ser armazenado por repositório e sim de forma global;
9. A linha do título da issue deve ocupar apenas a linha, não quebrando e mostrando “...” no final do texto para indicar que possui mais conteúdo;
10. O usuário deve poder atualizar a lista de issues arrastando a lista pra baixo com a opção refreshControl do <FlatList />
11. Ao clicar em uma issue, o usuário deve ser redirecionado para a URL da issue pelo navegador, não é preciso abrir a informação na tela do app;

Exemplo URL’s da API:

1. Repositório: https://api.github.com/repos/react-community/react-navigation
2. Issues: https://api.github.com/repos/react-community/react-navigation/issues

Estilização

1. A cor de fundo da aplicação é #EEEEEE
2. A cor de fundo do cabeçalho e das caixas é branca (#FFFFFF)
3. A cor do input no cabeçalho é #EEEEEE
4. A cor dos títulos e botões no cabeçalho é #333333
5. A cor das descrições abaixo dos títulos nas caixas é #999999
6. O tamanho das imagens nas caixas é de 45x45px
7. O tamanho da fonte no input do cabeçalho é 12
8. O tamanho da fonte do título nas caixas é de 16
9. O tamanho da fonte das descrições nas caixas é de 12
10. O tamanho da fonte do ícone no cabeçalho é 16
11. O tamanho da fonte do ícone “>” na caixa é de 20
12. A borda arredondada global é de 5
13. A cor de fundo da caixa de filtro é #DDDDDD
14. A cor do filtro é de #666666
15. O filtro não selecionado recebe opacidade 50% (0.5)
16. O filtro selecionado deve receber negrito

Dicas

1. Utilize o React Navigation para a navegação entre telas
2. As abas de status das issues não precisam utilizar React Navigation

Entrega

Para realizar a entrega e receber o código de correção você deve subir o repositório de sua aplicação para o Github e enviar por dentro do Station. Para isso, pode utilizar esse vídeo e caso queria aprender mais sobre Git e Github, pode ver essa live.

Boa sorte!

“Para quem fica melhor a cada dia, ficar pronto é utopia”!