API com React Native

O que vamos ver hoje?

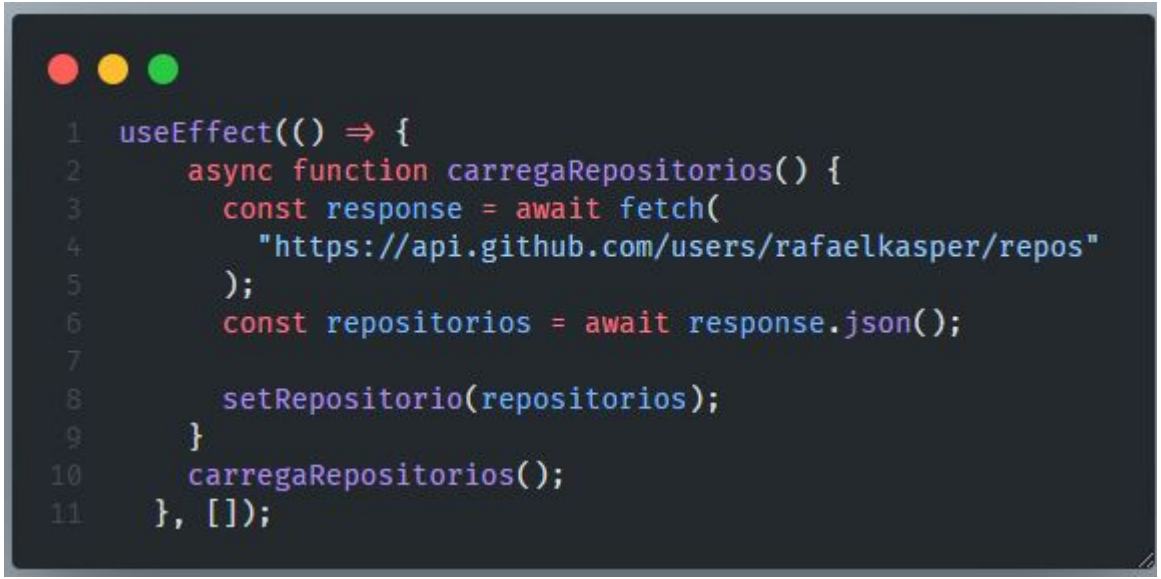
- Fetch
- Axios

API com fetch

Fetch

- É a função nativa do Javascript para o consumo de api’s.
- Faz uma consulta e resulta em uma promessa de que algo será retornado, essa promessa é chamada de Promisse.
- Essa promessa pode tanto ser boa, ter retornado os dados, quanto ter falhado por algum motivo - como no caso da conexão com o servidor cair.

Fetch



```
useEffect(() => {
    async function carregaRepositorios() {
      const response = await fetch(
        "https://api.github.com/users/rafaelkasper/repos"
      );
      const repositorios = await response.json();

      setRepositorio(repositorios);
    }
    carregaRepositorios();
  }, []);
```

API com Axios

Axios

- É um cliente HTTP baseado em promessas totalmente agnóstico, ou seja, que não depende de frameworks e bibliotecas.
- Ele é super maleável, podendo ser utilizado do lado do cliente (React, Vue, etc) e do lado do backend (express, nest, etc).

Axios

- Fornece vários atalhos para cada tipo de requisição:
  - axios.get('url')
  - axios.delete('url')
  - axios.post('url', { ...dados })
  - axios.put('url', { ...dados })

Axios

- Para termos o axios disponível em nossa aplicação, basta instalá-lo com via NPM:
  - npm i axios
- Depois é necessário importá-lo no componente que fará requisições:
  - import axios from 'axios'

Axios

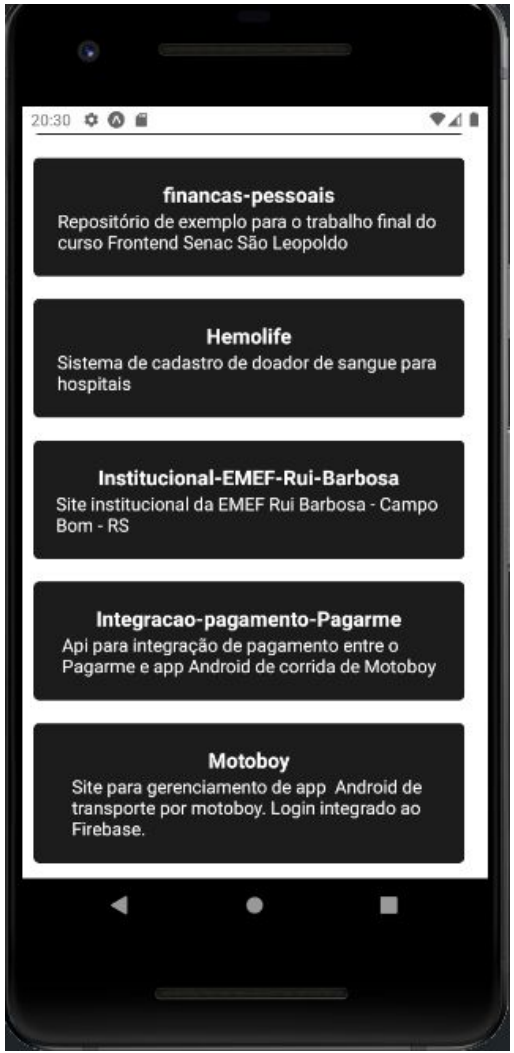
- Como é baseado em promessas, podemos lidar com o sucesso e falha da chamada, usando o bloco then e o catch.

Axios

```tsx
interface Props {
  title: string;
  content: string;
}

const Card: FC<Props> = ({ title, content }) => {
  return (
    <View style={styles.container}>
      <Text style={styles.title}>{title}</Text>
      <Text style={styles.content}>{content}</Text>
    </View>
  );
};
```

```tsx
interface RepositoriesProps {
  id: string;
  name: string;
  description: string;
}

export default function App() {
  const [repositories, setRepositories] = useState<RepositoriesProps[]>([]);

  const getRepositories = async () => {
    await axios
      .get("https://api.github.com/users/rafaelkasper/repos")
      .then(function (response) {
        setRepositories(response.data);
      });
  };

  useEffect(() => {
    getRepositories();
  }, []);

  return (
    <SafeAreaView style={{ flex: 1 }}>
      <View style={styles.container}>
        <FlatList
          data={repositories}
          renderItem={({ item }) => (
            <Card title={item.name} content={item.description} />
          )}
          keyExtractor={(item) => item.id}
        />
      </View>
    </SafeAreaView>
  );
}
```

Axios


20:30
financas-pessoais
Repositório de exemplo para o trabalho final do curso Frontend Senac São Leopoldo
Hemolife
Sistema de cadastro de doador de sangue para hospitais
Institucional-EMEF-Rui-Barbosa
Site institucional da EMEF Rui Barbosa - Campo Bom - RS
Integracao-pagamento-Pagarme
Api para integração de pagamento entre o Pagarme e app Android de corrida de Motoboy
Motoboy
Site para gerenciamento de app Android de transporte por motoboy. Login integrado ao Firebase.

Axios

```
const getRepositories = async () => {
    try {
      const response = await axios.get(
        "https://api.github.com/users/rafaelkasper/repos"
      );
      setRepositories(response.data);
    } catch (error) {
      console.log(error);
    }
  };
```

Dúvidas? 

Tarefa

- Criar um TextInput para receber a entrada do usuário
- Guardar os valores no useState
- Usar Axios para fazer uma requisição para uma api
- Exibir os dados com uma FlatList

Tarefa

- Sugestão de Api’s
  - https://api.github.com/users/{user}/repos
  - https://jsonplaceholder.typicode.com/
  - https://pokeapi.co/docs/v2#pokemon
  - https://rickandmortyapi.com/documentation/#get-a-single-character
  - https://api.adviceslip.com/#endpoint-random
  - https://api.chucknorris.io/

Tarefa

- Se desafia a ir além! Minha sugestão é fazer um app de previsão do tempo bem maneiro com a api: https://openweathermap.org/current#name