02

Efetuando login e armazenando os dados do usuário

Transcrição

Nosso usuário já consegue se registrar na aplicação, agora precisamos desenvolver nossa tela de login.

Na pasta "views", criaremos um novo arquivo chamado `login.vue`. Neste arquivo, precisaremos do `<template>` para escrever o HTML, e como aprendemos nas aulas anteriores, temos que encapsular todas as tags do nosso componente em um único elemento HTML.

Então colocaremos a `<div>` adicionando a classe `"container"`, e depois escreveremos `Login` em `<h1>` da linha seguinte. Começaremos de fato com o formulário de *login*.

Não precisaremos do `action`, portanto o tiraremos. Em `<form>`, teremos outra `<div>` com a classe `"from-group"` também do Bootstrap, como já vimos. O primeiro `<label>` será `for="email"` começando pelo `E-mail`, e o `<input>` do tipo `"email"` e classe `"form-control"`.

Copiaremos e colaremos com "Ctrl + C" e "Ctrl + V" e alteraremos o `<label>` para `for="senha"` e `Senha`, e o `<label>` do tipo `"password"` e classe `"form-control"`.

Além disso, precisaremos do botão de submeter as informações, então o `<button>` será do tipo `"submit"` porque irá enviar nosso formulário. Colocaremos como outra classe do Bootstrap, o `"btn btn-primary brn-block"`, e dentro, escreveremos `Logar`.

```
<template>
    <div class="container">
        <h1>Login</h1>
        <form>
            <div class="form-group">
                <label for="email">E-mail</label>
                <input type="email" class="form-control">
            </div>
            <div class="form-group">
                <label for="senha">Senha</label>
                <input type="password" class="form-control">
            </div>
            <button type="submit" class="btn btn-primary
brn-block">
                Logar
            </button>
        </form>
    </div>
</template>
```

COPIAR CÓDIGO

Atualmente existe a página para o usuário se cadastrar, mas não colocamos nenhum link para ela ainda. Então após `</button>`, faremos através de `<router-link>`, e escreveremos " `Não possui um cadastro, cadastre-se aqui!` ".

Em seguida, deveremos dizer ao `<router-link>` para onde irá levar. Estamos trabalhando com roteamento baseado nos nomes das rotas, então passaremos o `name: 'novo.usuario'` entre as chaves do `:to="{}"` .

```
<template>
    <div class="container">
        <h1>Login</h1>
        <form>
            <div class="form-group">
                <label for="email">E-mail</label>
```

```
                <input type="email" class="form-control">
            </div>
            <div class="form-group">
                <label for="senha">Senha</label>
                <input type="password" class="form-control">
            </div>
            <button type="submit" class="btn btn-primary
brn-block">
                Logar
            </button>
            <router-link :to="{ name: 'novo.usuario' }">
                Não possui um cadastro, cadastre-se aqui!
            </router-link>
        </form>
    </div>
</template>
```

COPIAR CÓDIGO

Desta forma, já teremos nosso formulário de login e agora vamos adicionar nossa rota.

Em `index.js`, copiaremos e colaremos o bloco com a rota, e alteraremos para `path: '/login'`, `name: 'login'` e `component: Login`, sem esquecer de realizar a importação de `Login` a partir de `'../views/Login'` no começo do arquivo.

```
//código anterior omitido

    {
        path: '/login',
        name: 'login',
        component: Login
    }
]

//código posterior omitido
```

COPIAR CÓDIGO

Ao testarmos a aplicação no navegador, nossa tela de login está pronta: temos o formulário, botão de ação e o link que leva o usuário até a tela de cadastro.

Nossa próxima tarefa é implementar as propriedades do usuário por meio o `v-model` . Em `index.js` ainda, iremos até o primeiro `<input>` e chamaremos o `v-model` igual a `"usuario.email"` como fizemos anteriormente.

Depois, adicionaremos `v-model;"usuario.password"` ao `<input>` seguinte.

Em seguida, criaremos uma nova tag `<script>` com `export default` para agrupar as configurações. Criaremos a função `data()` , lembrando que ela retorna um objeto com `return` para que cada componente tenha acesso ao seus dados de forma exclusiva e que não interfiram entre si. Portanto, temos `usuario: {}` e poderemos salvar e executar.

```
// código anterior omitido

<script>
export default {
    data() {
        return {
            usuario: {}
        }
    }
}
</script>
```

COPIAR CÓDIGO

Tudo está funcionando como o esperado no navegador. Nas próximas aulas seguiremos evoluindo com o comportamento de login.