03

Guardião global

Transcrição

Nossa abordagem para validar se o usuário está logado funciona, mas é de responsabilidade do desenvolvedor se lembrar de implementar este método sempre que desejar proteger um componente. Se fizéssemos o contrário e protegêssemos todas as rotas que não são públicas? Podemos fazer isso por meio do view router.

Em `index.js`, localizaremos a rota de `NovoUsuario` e `Login`. Definiremos essas rotas como públicas, por meio da própriedade `meta`.

```
path: '/login'
name: 'login'
component: Login,
meta: {
    publica:  true
}
```

COPIAR CÓDIGO

Antes de entrarmos em cada rota, faremos a validação para verificarmos se a rota é pública ou se o usuário está deslogado. Ainda em `index.js` importaremos o provedor

```
import provedor from '@/provedor'
```

COPIAR CÓDIGO

Feito isso, antes de exportar nosso roteador evocaremos nosso método guardião, e antes de cada rota `beforeEach()`, teremos acesso aos parâmetros `routeTo`, `routeFrom` e `next`. Feito isso, construiremos uma pequena validação com a condicional `if`. Em nosso caso, a falta de `token` quer dizer que o usuário está deslogado.

```
router.beforeEach((routeTo, routeFrom, next) => {
    if (!routeTo.meta.publica && !provedor.state.token) {
        return next({ path: '/login' })
    }
    next()
})
```

COPIAR CÓDIGO

Agora nossa rota está bem protegida, e não precisamos implementar o método que valida a cada componente.