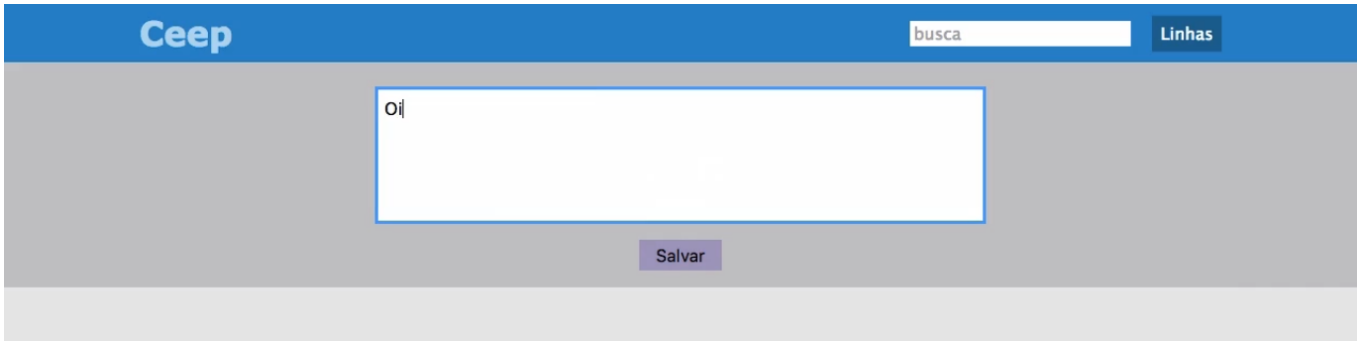
01

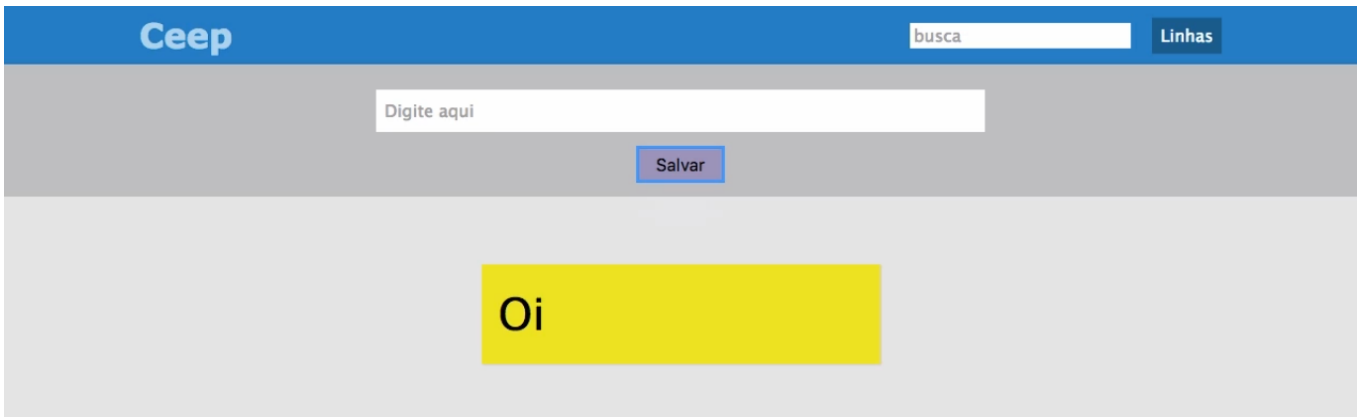
Logando o usuário com localStorage

Transcrição

Vamos continuar o nosso sistema, que chamaremos de Ceep. Nele, conseguimos criar cartões digitando em um campo.



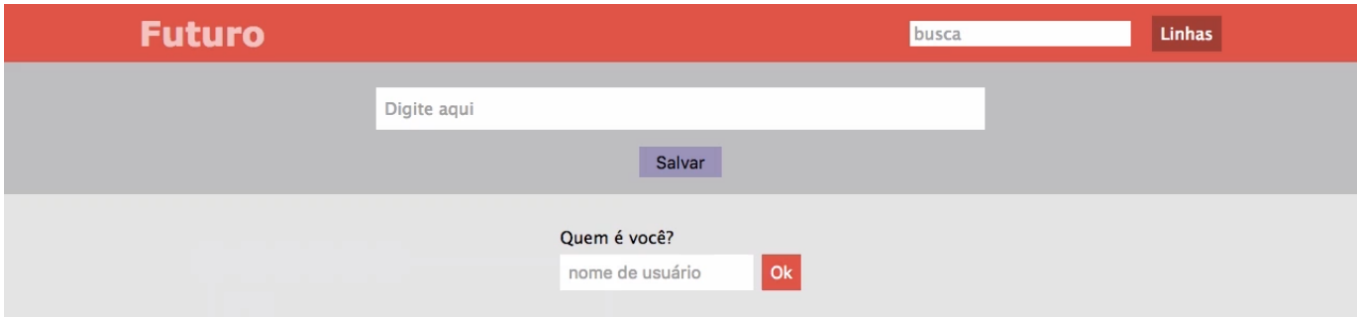
Ao pressionar `Enter`, o cartão fica fixado em um mural.



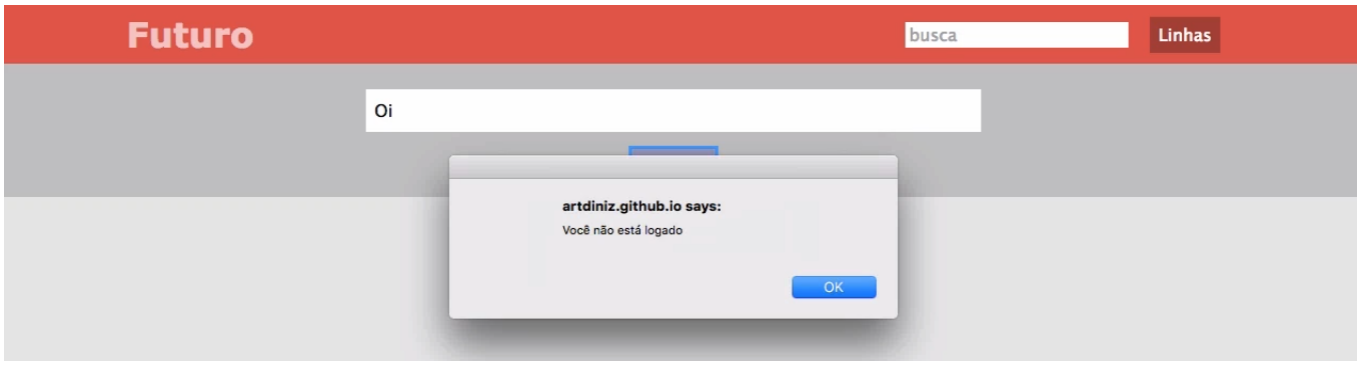
Parece que está funcionando. Mas há um problema: qualquer pessoa que vier nesse computador, verá os meus cartões. E como esse é um computador da Alura, várias pessoas se organizam com ele. Assim, outras pessoas podem criar cartões (aqui representados em cores diferentes).

Por isso, seria interessante que cada uma tivesse o seu próprio mural, para colocar suas ideias, seus lembretes... O que faremos agora será informar ao Ceep que eu sou o Artur e que ele deve colocar os cartões criados no meu mural. Temos um nome para isso: **login**.

Magicamente, nós temos uma visão de como isso funcionará no futuro.

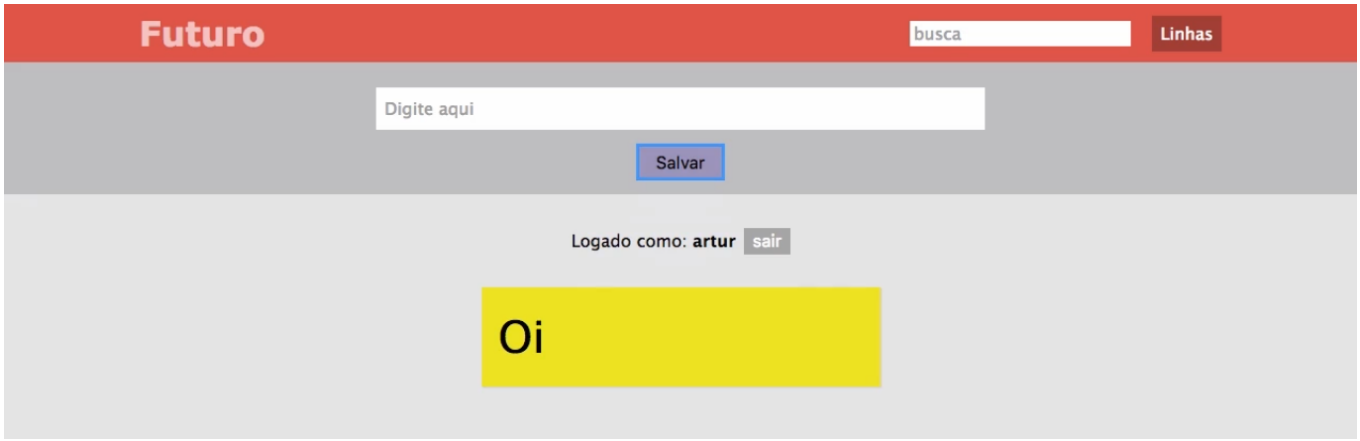


Se tentarmos criar um cartão, aparecerá um aviso que diz `Você não está logado`.

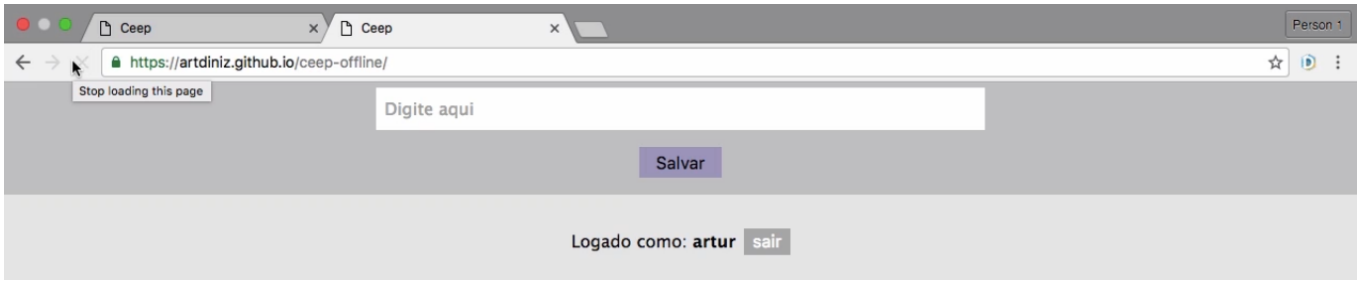


Para logar, temos um campo de formulário. Basta escrever o seu nome.

Agora estamos logados, podemos criar cartões.



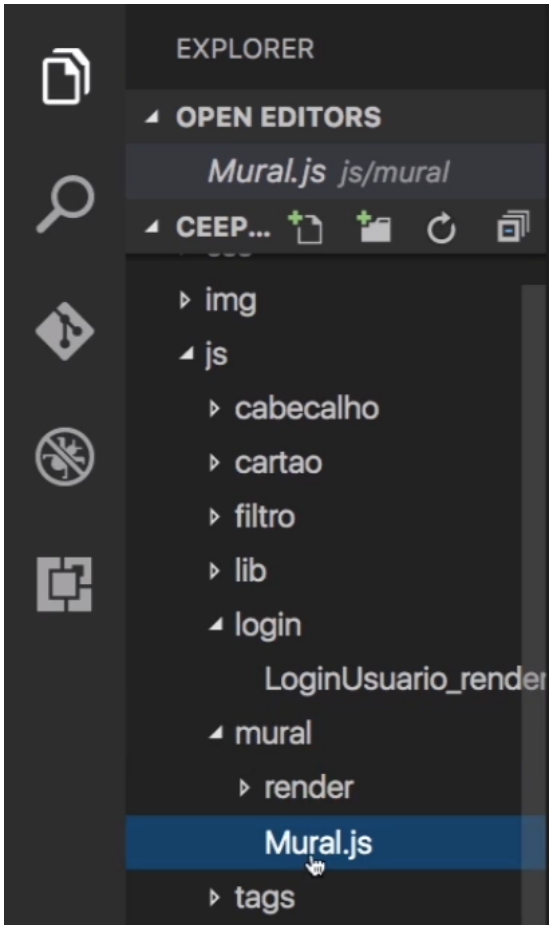
E o que isso tem a ver com o funcionamento offline da aplicação? Ao recarregar a página, temos o seguinte:



Ainda estou logado. E isso acontece porque alguém tem que saber que o último usuário da última página foi o Artur. Se fecharmos a aba e abri-la novamente, também continuaremos logados:

Então, de algum jeito, a informação de quem estava logado foi armazenada. Isso faz parte do funcionamento offline. E é isso que veremos agora. Vamos para o código?

Voltando para a aplicação do presente, temos que fazer mudanças que não me permitam adicionar cartões sem antes logar. Assim, temos que ver onde os cartões são adicionados. Nosso sistema está bem dividido, e cada funcionalidade tem praticamente um arquivo para si. Então, quando o usuário digita um cartão e clica em `Salvar`, ele estará chamando uma função `adiciona`, dentro de um arquivo chamado `Mural.js`, que está dentro da pasta `js > mural`.



Tudo que for adicionar cartões passa por aqui, onde temos a função `adiciona` :

```
const Mural = (function(_render, Filtro) {
  "use strict"
  let cartoes = []
  const render = () => _render({cartoes: cartoes, filtro: Filtro.tagsETexto})
  render()
  Filtro.on("filtrado", render)

  function adiciona(cartao){
    cartoes.push(cartao)
    cartao.on("mudanca.**", render)
    render()
    return true
  }

  return Object.seal({
    adiciona
    })

}) (Mural_render, Filtro)
```

Essa função passa um cartão como parâmetro, há uma lista de cartões, e o mural sabe que precisa renderizar quando o cartão tiver alguma mudança. Ou seja, como o próprio nome diz, a função `adiciona()` adiciona um novo cartão à página. Assim, se eu só deveria adicionar cartões quando logado. Se existe uma condição para que eu adicione cartões, tudo dentro dessa função deve estar dentro de um `if()` .

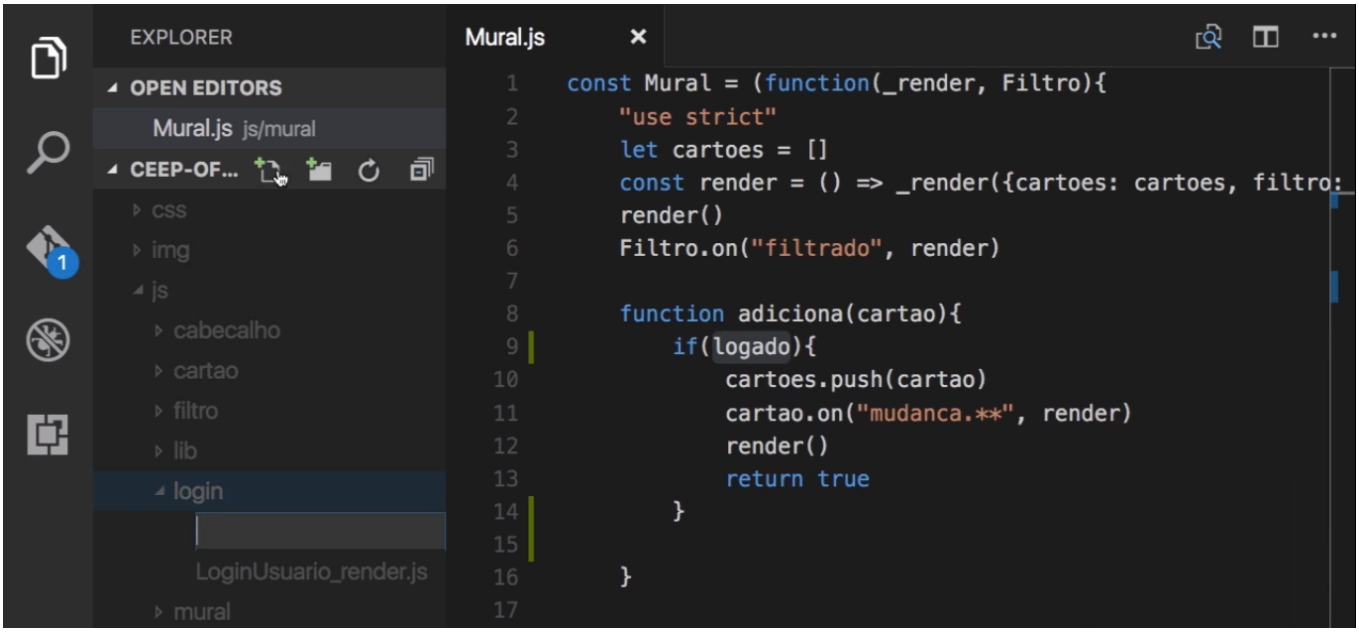
```
function adiciona(cartao){
  if(){
    cartoes.push(cartao)
    cartao.on("mudanca.**", render)
    render()
    return true
  }

}
```

Nesse `if()` precisamos ver se o usuário está logado ou não, portanto um valor que será `true` ou `false` .

```
function adiciona(cartao){
  if(logado){
    cartoes.push(cartao)
    cartao.on("mudanca.**", render)
    render()
    return true
  }

}
```

Poderíamos já criar a variável `logado` ali, mas sua funcionalidade de todo o login ficaria dentro do mural. E teríamos que adicionar muitas linhas a um código que já faz a parte dele. É melhor não misturar as coisas e seguir o padrão do projeto, separando a funcionalidade de login em outro arquivo.

Já existe uma pasta chamada `login` no projeto. Criaremos dentro dela um arquivo com o nome `LoginUsuario.js`.



Dentro desse arquivo, criaremos a nossa variável `logado`. Começaremos determinando que o usuário não está logado, com um `false`, para testar.

```
let logado = false
```

De acordo com a função `adiciona`, se o usuário estiver logado, pode criar um novo cartão. E se não estiver? Precisamos mostrar um alerta para o usuário, que será "Você não está logado".

```
function adiciona(cartao){
  if(logado){
    cartoes.push(cartao)
    cartao.on("mudanca.**", render)
    render()
    return true
  } else {
    alert("Você não está logado")

  }

}
```

Se o código estiver funcionando, já teremos uma variável que mural consegue acessar. Para que isso já funcione na aplicação, precisamos adicionar o novo script na página, no arquivo `index.html`.

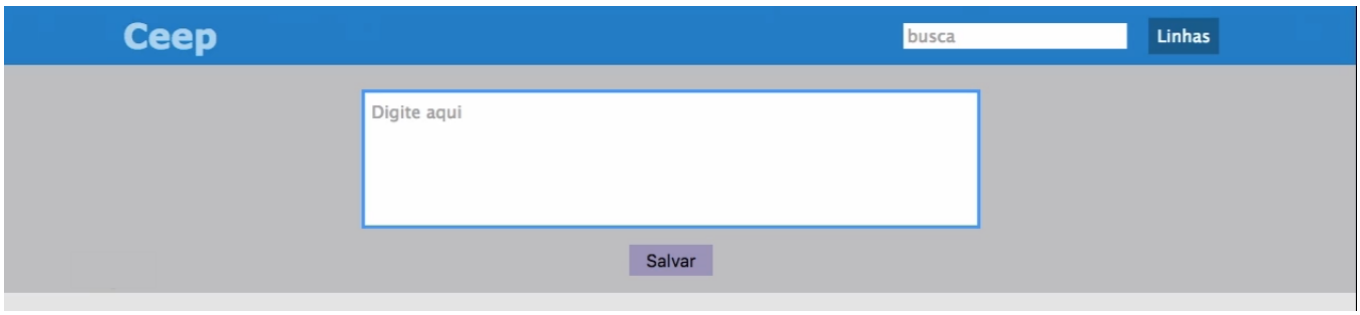
```
...
<script src="js/lib/eventemitter2.js"></script>
<script src="js/lib/KeyNoardNavigation.js"></script>
<script src="js/tags/Tags.js"></script>
<script src="js/cabecalho/mudaLayout.js"></script>
<script src="js/cabecalho/busca.js"></script>
<script src="js/filtro/Filtro.js"></script>
<script src="js/mural/render/Mural_render.js"></script>
```

```
<script src="js/mural/Mural.js"></script>
<script src="js/mural/cartao/render/Cartao_renderHelpers.js"></script>
<script src="js/mural/cartao/render/CartaoOpcoes_render.js"></script>
<script src="js/mural/cartao/render/CartaoConteudo_render.js"></script>
...
```

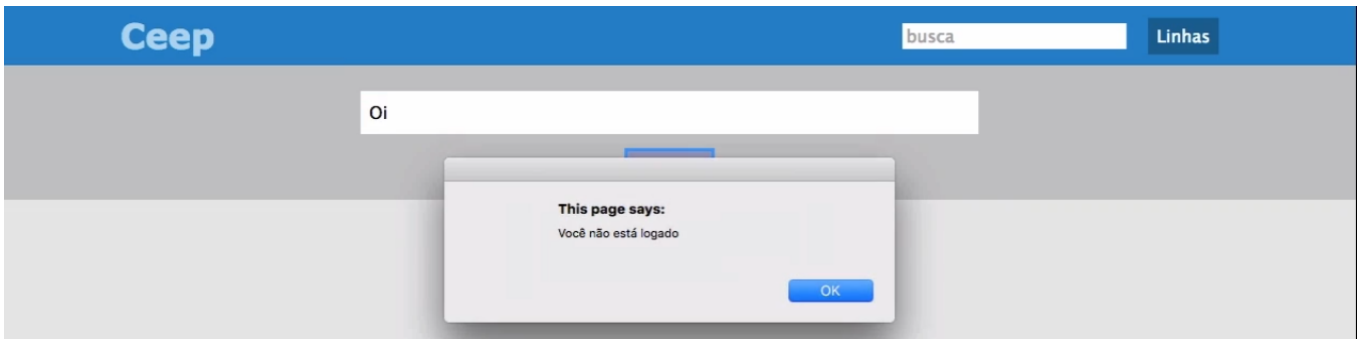
Como é o arquivo do mural que precisa dessa variável, o novo script precisa constar antes dos de mural. Sua source (`src`) deve apontar para `js/login/LoginUsuario.js` .

```
...
<script src="js/lib/eventemitter2.js"></script>
<script src="js/lib/KeyNoardNavigation.js"></script>
<script src="js/tags/Tags.js"></script>
<script src="js/cabecalho/mudaLayout.js"></script>
<script src="js/cabecalho/busca.js"></script>
<script src="js/filtro/Filtro.js"></script>

<script src="js/login/LoginUsuario.js"></script>

<script src="js/mural/render/Mural_render.js"></script>
<script src="js/mural/Mural.js"></script>
<script src="js/mural/cartao/render/Cartao_renderHelpers.js"></script>
<script src="js/mural/cartao/render/CartaoOpcoes_render.js"></script>
<script src="js/mural/cartao/render/CartaoConteudo_render.js"></script>
...
```

Vamos salvar esse arquivo e, na aplicação, recarregar a página.



Tentaremos adicionar um cartão.



Ele nos avisa que não estamos logados. Será que é porque a variável é `false` , ou porque ela nem existe? Podemos testar, voltando ao arquivo `LoginUsuario.js` :

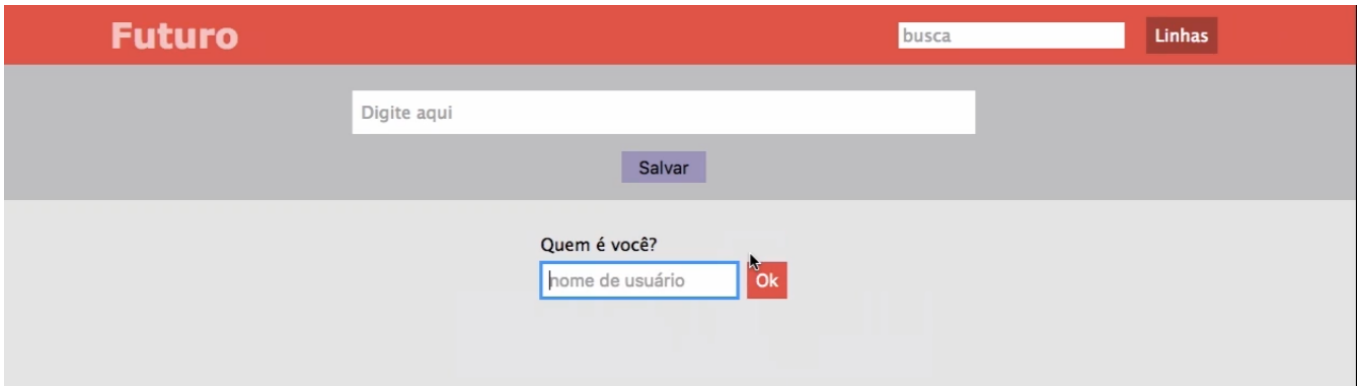
```
let logado = true
```

Voltando à aplicação e tentando criar um cartão:

Está funcionando, portanto a variável está sendo acessada pelo mural. Então manteremos o código do login separado do mural, que já está fazendo o que ele está fazendo. A variável está definida como `true`, o que não muda muita coisa na aplicação.

Mas como sabemos se o usuário está logado? Voltando ao futuro, vemos que quando o usuário está logado, a aplicação mostra um botão de `sair`.

E, quando clicamos em `sair`, ele volta ao formulário de login.

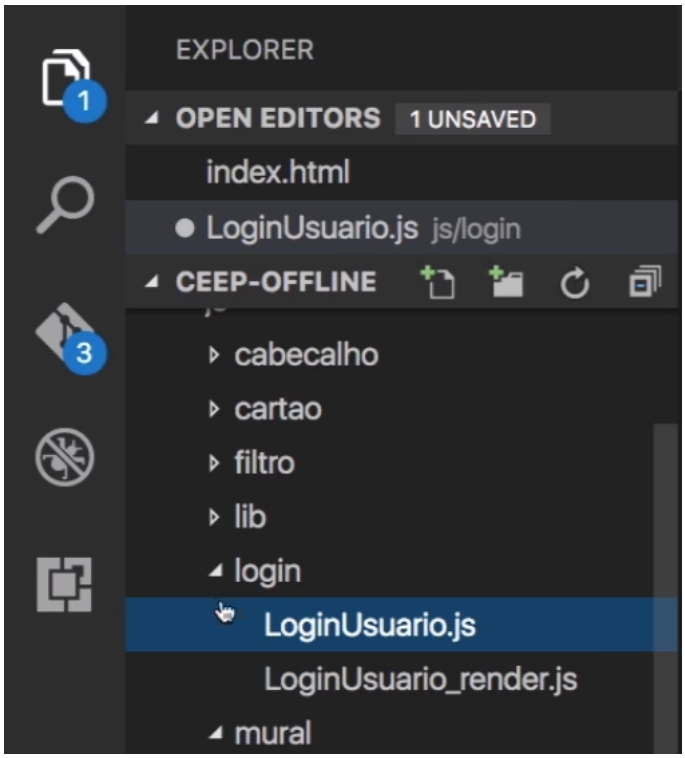


Precisamos estabelecer essa lógica. Mas se tivéssemos que fazer todo o formulário com o campo de texto e o botão vermelho, o código ficaria muito grande para um só curso e nós nos desviaríamos da questão do funcionamento offline. Por isso essa parte visual já está pronta para nós, só precisaremos aprender a usar.

Voltando ao arquivo: se o usuário não estiver logado, precisamos mostrar o usuário para que ele faça o login.

```
let logado = false
```

O código todo está dentro do outro arquivo da pasta `login`, o `LoginUsuario_render.js`.



É ele o responsável por criar o formulário, criar o botão de sair e as demais funções visuais. Para renderizar o formulário, ficaremos no arquivo que criamos, começando com o `LoginUsuario_render`, avisando se há login ou não com um objeto `logado`, começando com a propriedade `false`.

```
let logado = false

LoginUsuario_render({logado: false})
```

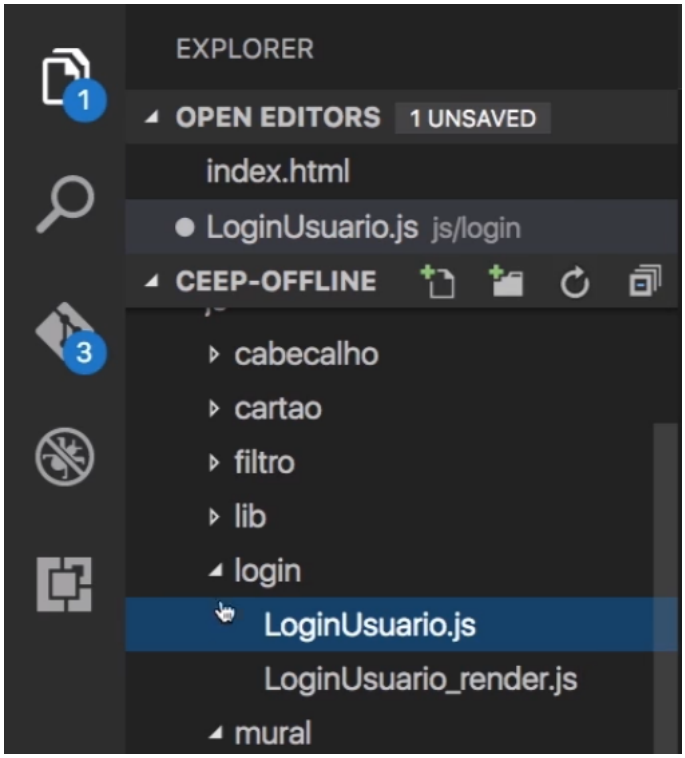
Se o usuário não está logado, precisamos mostrar o formulário. Mas, antes, precisamos adicionar o `LoginUsuario_render` ao `index.html`, junto ao `LoginUsuario`.

```
...
<script src="js/lib/eventemitter2.js"></script>
<script src="js/lib/KeyNoardNavigation.js"></script>
<script src="js/tags/Tags.js"></script>
<script src="js/cabecalho/mudaLayout.js"></script>
<script src="js/cabecalho/busca.js"></script>
<script src="js/filtro/Filtro.js"></script>

<script src="js/login/LoginUsuario_render.js"></script>
<script src="js/login/LoginUsuario.js"></script>

<script src="js/mural/render/Mural_render.js"></script>
<script src="js/mural/Mural.js"></script>
<script src="js/mural/cartao/render/Cartao_renderHelpers.js"></script>
```

```
<script src="js/mural/cartao/render/CartaoOpcoes_render.js"></script>
<script src="js/mural/cartao/render/CartaoConteudo_render.js"></script>
...
```

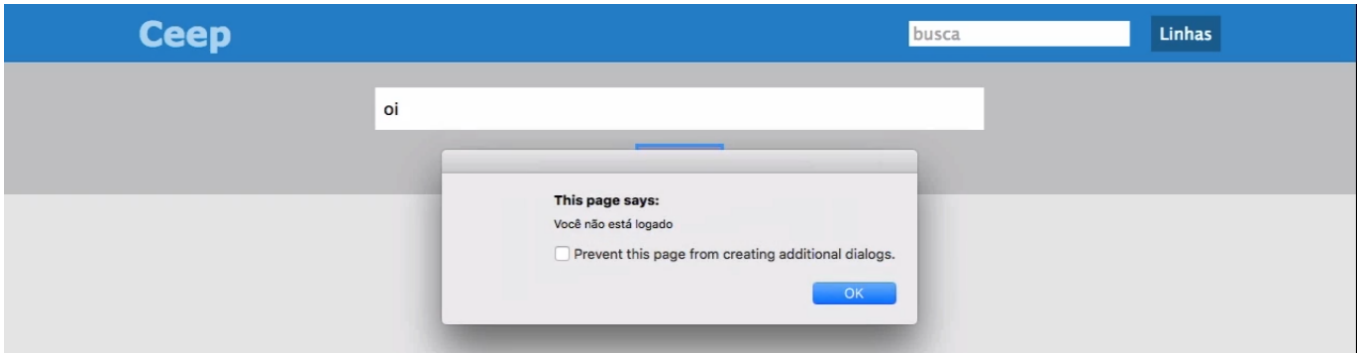
Esse padrão de manter a parte visual separada está aplicado ao sistema inteiro. Se você der uma olhada no sistema todo, confirmará esse padrão observando os arquivos com `_render` no final, responsáveis pelo visual.

Ao salvar esse arquivo, vamos atualizar a aplicação e ver se o formulário aparece.



Se eu digito o meu login:

Ele entende que estou logado como Artur. Então devemos conseguir adicionar um cartão, certo?



Quando tentamos, ele nos informa que não estamos logados. Mas... acabamos de fazer o login!

O que acontece é o seguinte: quando fazemos o render da variável `login` como `false`, não estamos mudando o valor da variável `logado`. Não chegamos a falar isso, e teremos que ensinar essa etapa para o `LoginUsuario_render`.

Uma das coisas que ele conseguirá receber é se o usuário está logado (`onLogin`) ou não (`onLogout`). A outra é o que deve acontece quando o usuário digitar seu nome e clicar no `ok`. Queremos que quando o login acontecer, a variável `logado` fique `true`.

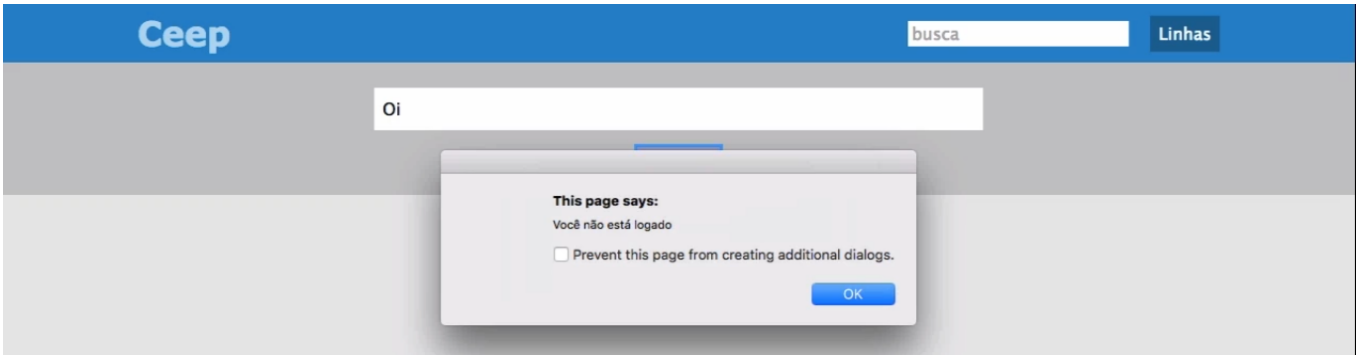
```
let logado = false

LoginUsuario_render({
  logado: false
  ,onLogin: () => logado = true

  })
```

Essa é a famosa função de *callback*. Se você não está acostumado com a função de callback e mexer na página, pode dar uma olhada em outros cursos (https://cursos.alura.com.br/course/javascript-es6-orientacao-a-objetos-parte-2) da Alura que ensinam isso. Também há os cursos de Jquery (https://cursos.alura.com.br/course/jquery-manipulacao-dinamica-de-conteudo), que ensinam a fazer os códigos que estão sendo executados nas nossas funções de render.

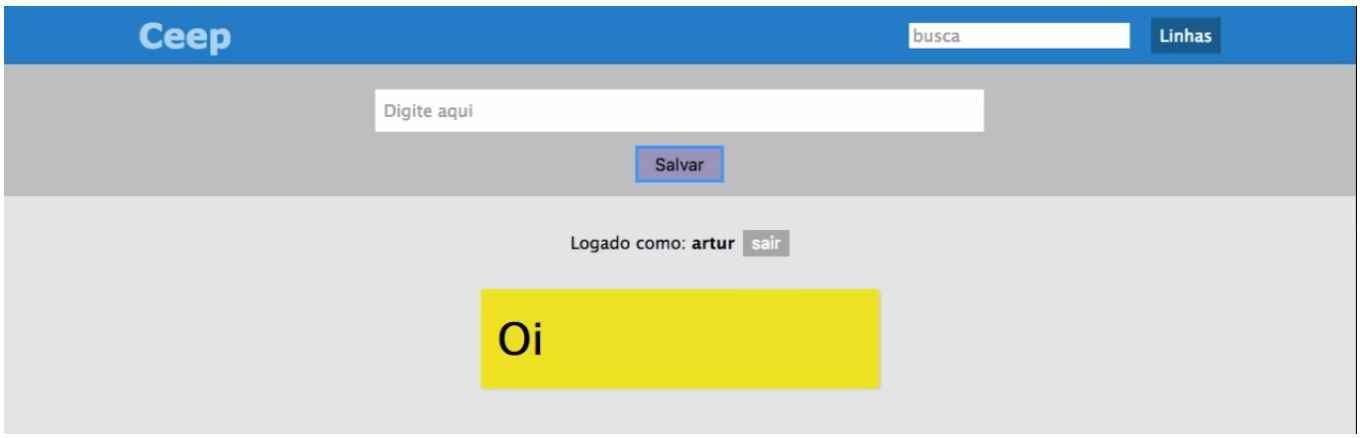
Falta dizermos o que acontece com a variável se o usuário não estiver logado. Sabemos que `logado` será `false`. Assim:

```
let logado = false

LoginUsuario_render({
  logado: false
  ,onLogin: () => logado = true
  ,onLogout: () => logado = false

  })
```

Esse código diz que a variável muda de acordo com o que o usuário mudar na página. Vamos testar, voltando à aplicação e recarregando-a? Se tentarmos criar um cartão sem logar, teremos:



Novamente, a aplicação nos avisa que não estamos logados. Se logarmos como Artur e tentarmos criar novamente:



O nosso sistema de login já está funcionando! Claro, poderíamos acrescentar uma senha para deixá-lo mais seguro, mas para esse sistema o nome já basta.