

Logout

Transcrição

[00:00] Na aula anterior vimos um pouco de como funciona o sistema de autenticação do play, baseado em anotações que nem vemos aqui `@Authenticated` e permitimos que nosso usuário conseguisse fazer login no sistema, só que o usuário ainda não consegue fazer log out e ele fica preso para sempre, isso não é legal, vamos resolver isso então, primeiro vamos fazer o painel do usuário, mesmo que vazio só para conseguirmos ver alguma coisa quando fizermos login.

[00:29] Então vamos criar um painel de usuário, eu vou criar aqui um novo arquivo chamado `panel.scala.html`, e o que ele vai ter? Ele só vai ter o `@main("Painel do usuário")`, com o conteúdo bem-vindo e vai ser isso isso,

Bem-vindo

, só para sabermos que conseguimos logar e para termos aquele header bonitinho.

[01:00] E por que eu estou falando isso? Porque é legal que adicionemos um botão de log out no header, então vamos lá, `panel.render`. Vamos atualizar aqui para ver se ele reconhecesse que o painel existe ou então precisamos aqui importar `views.html.*`, então agora está tudo certo, ele reconheceu o painel, se atualizarmos aqui e já estou logado, deve funcionar.

[01:33] Então usuário/panel, da última foi `unauthorized` agora temos a tela de bem-vindo, então vamos fazer uma adição? Vamos adicionar aquele botão de log out, vamos lá no main e vamos vir na lista de botões que temos, mas vamos adicionar o botão de log out, só se já estivermos logado, como sabemos se já estamos logado na nossa view?

[01:56] O request no qual inserimos o nosso username está disponível aqui, se fizermos `@request.username` temos aquele username que colocamos no usuário autenticado, esse `getUsername` vai retornar o conteúdo aqui para nossa tela, então podemos conferir se isso está vazio. `@if(request().username())` e colocamos o nosso botão, então vamos colocar aqui um botão de log out e vamos colocar também um botão para o painel, deixa eu só colocar tudo isso aqui inline porque fica mais fácil de ler.

[02:40] Então, primeiro o botão para o painel do usuário controller, então `routes.UsuarioController.painel` e agora um link para fazer log out, ainda não temos essa rota, mas podemos fazer o método depois a rota, aqui em vez de painel vai ser faz log out e o botão vai estar escrito log out. Agora se estivermos logado, temos o botão de painel e de log out, então vamos abrir o arquivo de rotas e aqui junto com o login eu vou fazer um GET /logout que vai linkar com aquele método, `controllers.UsuarioController.fazLogout`.

[03:33] Que também não recebe nada, vou só dar um enter aqui para separar um pouco, e agora podemos ir lá no usuário controller e fazer a nossa lógica de log out, vai ser uma rota autenticada, porque não faz sentido acessar o log out se você está deslogado, então `@Authenticated(UsuarioAutenticado.class)`, public Result fazLogout, o que vamos fazer aqui? Vamos remover usuário da sessão, como faz isso?

[04:14] Você chama o objeto session e chama o método clear dele, assim você limpa a sessão inteira, então vamos mandar uma mensagem de sucesso para o usuário, success "Logout efetuado com sucesso", e já direcionamos para página de login, que é a única que ele vai ter acesso deslogado, então `Return redirect(routes.UsuarioController.formularioDeLogin())`, a nossa lógica de log out está pronta, vamos ver como que funciona?

[05:01] Eu venho aqui vou atualizar e vai aparecer os nossos novos botões assim que terminar de compilar, então podemos ir lá na nossa tela, atualizar aqui e podemos vir no painel, de redirecionar para o painel e no log out, vamos fazer o log out com sucesso, agora se eu tentar acessar o painel de novo, usuario/painel eu tomo um unauthorized de novo, então agora falta uma coisa, eu não quero ficar dando um unauthorized na cara do usuário, eu quero por exemplo redirecionar ele de volta para página de login, vamos fazer isso?

[05:40] Vamos vir no usuário autenticado e sobre escrevemos outro método, é o onUnauthorized, ele lida quando o username da nossa request está vazio, então se o username estiver vazio ele vai vir e cair nesse método. Então fazemos o que precisarmos aqui, vamos mandar uma mensagem de fracasso para o nosso usuário, então context.flash e aqui a sintaxe é um pouco diferente, não podemos passar os valores aqui, temos que colocar um put como se fosse um mapa mesmo.

[06:20] Vamos falar que danger e vamos indicar que o usuário não está autorizado para essa página, então redirecionamos ele para página de login. Então vamos lá, redirect(routes.UsuarioController.formularioDeLogin()), agora perdemos totalmente o acesso ao nosso painel caso tentemos acessar ele deslogado, recebemos uma mensagem, então o usuário sabe o que está acontecendo.

[07:02] Agora vamos fazer um refactor rápido para substituir todos aqueles TODOS pelas rotas adequadas e já aproveitar e fazer o login do usuário. Então vamos lá TODO fazer login, podemos fazer o login do usuário aqui automaticamente, então inserimos o usuário na session, session(AUTH, usuario.getEmail()).

[07:30] Aqui no painel de usuário vamos dar redirect para routes.UsuarioController.painel, vou copiar isso aqui e tirar esse TODO da rota aqui, agora aqui para o login é a mesma coisa, vamos redirecionar para o formulário de login, tem mais algum? Tem mais um aqui que é para o login de novo, então colamos aqui, formulário de login e removemos o TODO.

[08:05] Então o que aprendemos aqui, aprendemos a remover o usuário da sessão executando um log out, e aprendemos a tratar o caso em que o usuário não logado tenta acessar uma página que requer autenticação que nem o nosso painel, e o que fazemos? Redirecionamos ele para o painel de login com uma mensagem de não autorizado. No próximo vídeo vamos aprofundar um pouco mais na autenticação para aumentar a segurança do nosso sistema, até a próxima.