04

Autenticação da API no header

Transcrição

[00:00] No ultimo vídeo fizemos o token de usuário e fizemos esse painel aqui bonitinho, para mostrar o token e explicar como ele faz as requisições para nossa API aqui, adicionando no cabeçalho da requisição a chave de acesso que conseguimos ver aqui escondida. Mas ainda não fizemos a autenticação dela, então na verdade tudo isso aqui ainda não estamos usando, se eu vier aqui e fizer a api/produtos, temos acesso, mesmo sem adicionar a chave no cabeçalho. Então vamos fazer isso?

[00:38] Vamos criar um novo método de autenticação, que nem fizemos com a autenticação do usuário, quais são os métodos que vão ser autenticados? Vão ser os métodos a API, então eu posso jogar aqui direto no controller, que o controller inteiro é autenticado, e criamos aqui a classe, vai ser autenticação dos acessos da API, então acesso autenticado, AcessoDaAPIAutenticado, não é um nome lindo, mas é bem claro, então está bom

[01:10] Vou criar essa classe e vou criar junto com usuário autenticado, então class AcessoDaAPIAutenticado e vai estender a classe Authenticator do play.mvc.Security. Acesso da API autenticado eu vou confirmar aqui, podemos importar ela aqui agora que está ok, mas ainda não fizemos nada aqui, vamos subscrever aquele método getUsername e fazer a lógica de autenticação.

[02:00] O que eu tinha dito? Eu tinha dito que o nosso token ia estar no cabeçalho da requisição, então temos que acessar o cabeçalho, precisamos acessar a requisição a partir do context, então context.request e pegamos o header e vamos ver aqui na nossa página qual o header, API -token, eu vou copiar aqui e colar aqui no nosso getHeader, agora temos teoricamente o nosso código.

[02:40] Se ele for um código valido passamos a requisição para frente, se não for não vamos permitir que o usuário acesse a nossa API, então vamos receber aqui injetado o nosso usuarioDAO e vamos pesquisar por um usuário que tenha um token com esse código, então usuarioDAO e fazemos a pesquisa aqui, usarioDAO.comToken(codigo), guarda isso em uma variável possivelUsuario.

[03:18] Caso o usuário exista, retornamos o username dele, se não retornamos nulo, então vamos lá if(possivelUsuario.isPresent()) retornamos o possivelUsuario.get().getnome(), caso não exista retornamos nulo. Então, vamos fazer o teste, vamos ver se conseguimos acessar a nossa API agora, api/produtos, deveríamos tomar um unauthorized na cara. Está funcionando, com a autenticação funcionando, agora precisamos testar se conseguimos realmente fazer a requisição e ter a resposta correta.

[04:10] Como vamos fazer isso? Eu vou usar um comando aqui no terminal, eu vou usar no Mac um curl, mas quando formos fazer a explicação da aula, eu vou mostrar como fazer para Linux, Mac e Windows também. Se eu fizer a requisição agora com a API token nulo, então é “curl –header”, adicionamos os headers, no caso eu estou colocando a API token vazio e a nossa url de requisição, se eu tentar fazer aqui com o token nulo ele vai retornar aquela mesma página html.

[04:48] Que não é legal, depois podemos reformatar e fazer um Json de erros, e se colocarmos efetivamente o nosso token aqui? Eu vou vir aqui, copiar o token, colar e mandar um Enter e recebemos nosso Json bonitinho, olha todos produtos que temos no banco. Então, vamos trocar aquela mensagem de erro de não autorizado, para um Json, como fazemos isso? Vamos vir aqui e vamos escrever o método onUnauthorized.

[05:25] E vamos mandar como resposta um Json com o método não autorizado, eu vou remover isso aqui, colocar aqui um unauthorized e vamos passar aqui um Json.ToJson e o nosso objeto de erro, por enquanto vai ser só uma string,

alteramos aqui depois. Como vamos fazer um objeto de erro? Precisamos definir o que é um erro antes, e eu vou fazer isso com um mapa, então new hashMap<> do java.util, vou guardar isso em uma variável chamada Map do tipo map.

[06:07] Eu vou colocar aqui como string e esse cara aqui vai chamar parametrosDoErro, o que vamos colocar aqui? Eu vou colocar o código e a mensagem do erro de não autorizado, parametrosDoErro.put, a chave vai ser código e o valor vai ser 401, que é o código de não autorizado e aqui em baixo vai ter mensagem “Não autorizado!”.

[06:53] Já temos os nossos parâmetros de erro, agora precisamos de uma raiz escrito erros, então vamos fazer um novo mapa aqui, que é o mapa da resposta de erros, só que aqui vai ser um string object, porque vamos passar um mapa como parâmetro. Então erros aqui new hashMap, e colocamos erros.put, vamos passar aqui a chave erros e o valor vai ser os parâmetros do erro e transformamos isso aqui tudo em um Json, erros vai virar um Json.

[07:30] Agora se fizermos a requisição aqui errada de novo, eu vou dar uma limpada aqui na tela, se alterarmos aqui qualquer coisinha e invalidar a nossa chave, vamos receber uma mensagem de erro. Olha só, erros, código 401, mensagem “Não autorizado!”.

[07:50] Então o que vimos hoje? Vimos como pegar o cabeçalho da requisição, pegando do contexto, a requisição, o header, pegamos o usuário a partir disso e fizemos a autenticação da nossa API, direto aqui no controller, para garantir que todos esses métodos vão ser autenticados, depois transformamos nosso Html de erro de não autorizado em uma mensagem Json toda bonitinha, para o nosso usuário ter bem claro o que ele está recebendo.

[08:22] Na próxima aula vamos pegar e fazer um armazenamento de cada acesso que o usuário fizer na nossa API, para termos um controle e para o usuário também ter um controle de como ele está usando a nossa API.