Certified Tech Developer
The Ultimate Degree

Especialização em Back End I

Exercício para as mesas de trabalho

- Exercício individual 👨
- Nível de complexidade: intermediário 🔥🔥

Enunciado

Para a primeira parte da atividade, vocês devem:

1. Descarregar o arquivo de exemplos, este aqui tem vários exemplos do assunto.
2. Pegue a solução "click" (ou two_providers).
3. Adicione mais 2 rotas (neste caso, elas podem estar na mesma classe).
4. Um serviço vai ser chamado de **servicoDH** e outro **servicoNoDH**.
5. O servicoDH só pode ser acessado através de um endereço @digitalhouse e o servicoNoDH pode ser acessado por qualquer pessoa.
6. Analisar por que a autenticação é resolvida no gateway e as autorizações nos serviços individuais (dica: o gateway não precisa conhecer todas as permissões para cada serviço. Se esse fosse o caso, com cada mudança de permissão em cada microsserviço eles deveriam modificar o gateway).

A partir do código obtido ao resolver o exercício anterior, vamos fazer nosso próprio esquema de permissões.

1. Criar 10 Serviços (servico1, servico2 ... servico10).
2. Cada serviço deve retornar um texto com seu nome.
3. Você vai criar uma classe (onde estão as entidades ou você pode criar um package chamado " entities ") chamada **DHRoles**.

4. Dentro dele, eles vão colocar um **HashMap<String, String[]>** chamado **permissionsByRole**.
5. A classe deve ter uma construção que alimente esse campo.
6. Este campo será um Map cuja key será o nome do Rol e cujo valor serão as páginas que podem ser acessadas.
7. Em outro campo dentro da mesma classe, eles vão criar outro **HashMap<String**, **String>** campo chamado **roleByUsers**. Neste caso, a chave será o nome do usuário e o valor será o rol.
8. A classe acima pode ser uma classe mono ou uma classe estática. Não é necessário estar no banco de dados.
9. Os roles serão (atribuí-los na construção):
   a. "Pares": servico2, servico4, ..., servico10.
   b."Ímpares": servico1, servico3, ..., servico9.
   c."Fibonacci": Serviços 1, 2, 3, 5, 8.
   d."Multiples3": Serviços 3, 6, 9.
10. Associar um usuário a cada rol (também pode ser na construção).
11. Gerar métodos para facilitar a tarefa na classe criada, tais como **getRoleByUser(String username), getPermissionsByRole(String role).**
12. Chame os métodos no ponto anterior em cada serviço.
    Seria algo parecido com isto:

    if (!getPermissionsByRole(getRoleByUser(username)).contain("serviceName")) {
    throw new UnauthorizedException()}

Mãos à obra!