04

Trabalhando com paginação

Transcrição

[00:00] Vamos focar na página *home* agora, que tem uma coisa muito importante se você colocar esse sistema em produção.

[00:07] À medida em que os pedidos forem sendo entregues, os usuários que estiverem chegando na página *home*, abrindo a página principal do site, vão ver a lista de todos os pedidos que foram entregues. E quando falamos em todos, são todos os pedidos mesmo.

[00:26] Vamos no "HomeController.java" e veja que a busca que ele faz é `findByStatus`, mas sem nenhum tipo de limite, ele vai buscar todos os pedidos que estiverem com esse status. Então se tiver um milhão, ele vai buscar um milhão. Alguma coisa vai quebrar rapidamente na tua aplicação e nós vamos resolver isso.

[00:51] Além de termos que diminuir, também tem outra coisa bem interessante, que é o seguinte: eu entrei com o usuário da Maria e cadastrei esse pedido aqui, fui no banco de dados e assim como eu fiz com a Vitrola, eu informei o status dele como "entregue".

[01:08] Então eu alterei o status desse Fitbit Charge para entregue e do Vitrola também coloquei como entregue, e preenchi aqui a data de entrega e preenchi um valor negociado. Se você não entendeu como isso aconteceu, foi assim.

[01:25] O que temos que fazer? Aqui ele está mostrando o pedido do dia 19, que foi entregue, que eu coloquei o da Maria aqui, e aqui o do dia 20. Mas eu não tenho nenhum tipo de ordenação ainda, então ele pode simplesmente me

mostrar o pedido que foi entregue a cinco anos atrás, com o preço de cinco anos atrás. Talvez isso não seja muito atrativo para o nosso site. E o que vamos fazer?

[01:54] A primeira coisa é ensinarmos a nossa aplicação a ordenar os pedidos. Quando eu for fazer uma busca aos pedidos, eu quero buscar os pedidos em ordem de data decrescente. Como fazemos isso? Com o Spring Data JPA.

[02:13] Eu já abri aqui a documentação do Spring, entrei no Spring Data. Está aqui, no menu à esquerda. Quando você faz uma consulta você tem esse "Using Sort".

[02:32] Esse Using Sort é o quê? Para a consulta que você fez, você passa esse `Sort`, eu vou jogar isso no `List<Pedido> findByStatus(StatusPedido status);`. Nós só alteramos isso passando um `Sort sort`. Apertamos o atalho "Ctrl + Shift + O". Tem várias opções, você vai escolher o do Sprint Framework Data. Tem várias outras aqui também, tem que saber importar corretamente, fique atento a isso.

[02:55] E agora temos que passar um tipo de ordenação para ele. Eu vou colocar aqui em cima de `List<Pedido> pedidos` e vou instanciar essa ordenação usando essa classe `Sort`, ela me permite fazer isso. Ele também está me mostrando um exemplo disso aqui, mas é bem fácil fazer. Você clica em `Sort`, coloca `Sort.`. E qual será o parâmetro de `Sort.` ?".

[03:29] Deixe-me abrir o pedido, ver as colunas que ele tem. `nomeProduto`, `valorNegociado`, `dataDaEntrega`. Eu quero ordenar pela data da entrega, `Sort.by("dataDaEntrega")`, que eu não estou mostrando os últimos pedidos entregues, então eu quero ordenar pela data da entrega.

[03:43] E eu quero ordenar de forma descendente, `Sort.by("dataDaEntrega").descending()` Pronto! Então eu tenho o `sort`, vou criar essa variável e vou passar aqui, porque na consulta eu adicionei que só

quero receber um critério aqui de ordenação. E vamos testar isso, vamos ver se isso funciona.

[04:07] Eu volto para a página e tem dois itens: um do 19 e um do 20, a princípio ele tem que permanecer exatamente como está aqui. Mas para você acreditar que isso realmente existe, além de você poder fazer, eu vou colocar na ordem crescente, `.ascending()` para mudar essa ordem e colocar que esse Fitbit, que teve uma entrega mais antiga, venha para primeiro. Funcionou!

[04:31] Só que é aquele problema inicial que eu falei, que é a questão de quantidade de itens que vão aparecer. Nós não resolvemos ainda, nós resolvemos a ordem - e isso é bem legal.

[04:43] E para a questão de ordem, se você der uma olhada nesse "Using Sort" na documentação do Spring Data JPA, ele fala sobre um método chamado de `PageRequest`. O `PageRequest` você pode passar dois parâmetros para ele, que são o seguinte: eu quero a página 1 e eu quero 20 itens por página, o `PageRequest` é só isso.

[05:06] Vou copiar esse `PageRequest.of(1, 20)` e vou colar aqui embaixo do `Sort`. Vou apertar as teclas "Ctrl + Shift + O" e ele não tem mais opções. E eu vou dizer o seguinte: "eu quero que você me dê a página primeira, que é a página 0, e eu quero um item por página, para vermos realmente a coisa acontecendo".

[05:30] E o mais legal é que você pode combiná-lo com o critério de ordenação. Isso é muito importante quando você está paginando, é só adicionar mais esse parâmetro.

[05:39] E eu venho aqui no `of` e peço para ele colocar em uma variável para mim. Vou chamá-la de `paginacao`. `PageRequest paginacao = PageRequest.of(0, 1, sort);`. Em vez de passar só um critério de ordenação, vou passar `paginacao`.

[05:54] Só que ele passa a não funcionar. Por quê? Porque ele está esperando um `Sort` aqui. Mas podemos passar um `Pageable`. Salvou e ele passa a funcionar, ou seja, agora eu estou passando um `PageRequest` e estou recebendo com esse `Pageable`. São tipos diferentes, mas é assim que se faz mesmo.

[06:15] Se você entrar aqui no link do `PageRequest`, entra nesse `AbstractPageRequest`. Você vê que ele implementa essa interface `Pageable`. É isso que eu estou passando para lá e é assim que funciona. Estou colocando o critério de ordenação.

[06:29] O que nós esperamos acontecer agora? Eu espero que ele me dê a primeira página, eu quero que ele me dê um item por página em ordem crescente de data de entrega, ou seja, é para aparecer só um item e esse item tem que ser o Fitbit, porque ele é o mais antigo. Como é a data na ordem crescente, então não é para aparecer mais esses dois. Vamos atualizar e ver se acontece? Aconteceu!

[06:55] Se eu voltar para descendente, `.descending()`, é para aparecer agora a Vitrola.

[07:02] Ótimo! Só que está aparecendo só um item e eu quero dez itens, porque não faz sentido mostrar só isso. Então vou aumentar aqui, `PageRequest.of(0, 1, sort);`, para `PageRequest.of(0, 10, sort);`. Veja que vão aparecer dois, porque só tem dois.

[07:16] É assim que trabalhamos paginação com o Spring Data. É bem simples, isso é bem importante para otimizarmos o nosso site e não darmos informação demais para o usuário. Isso pode tanto travar nossa aplicação quanto se ele estiver acessando de um *browser*, de um smartphone, isso pode travar a própria aplicação dele, que é conteúdo demais. Podemos fazer mais melhorias com relação a isso - e nós vamos fazer.

[07:40] Até o próximo vídeo!