Bootcamp Spring React 3.0 - Cap. 01

CRUD

Competências

- Criar projeto Spring Boot
- Criar monorepo Git
- Organizar o projeto em camadas
    - Controlador REST
    - Serviço
    - Acesso a dados (Repository)
- Criar entidades
- Configurar perfil de teste do projeto
- Seeding da base de dados
- Criar web services REST
    - Parâmetros de rota @PathVariable
    - Parâmetros de requisição @RequestParam
    - Corpo de requisição @RequestBody
    - Resposta da requisição ResponseEntity<T>
- Padrão DTO
- CRUD completo
- Tratamento de exceções
- Postman (coleções, ambientes)
- Dados de auditoria
- Paginação de dados
- Associações entre entidades (N-N)

Código do arquivo pom.xml (Spring Boot 2.4.x)

(favor pegar lá do projeto para baixar no material de apoio)

Vídeos auxiliares

Conceitos sobre desenvolvimento web e REST
https://www.youtube.com/watch?v=b8uLFfzcVQ8

Análise do app da Semana DevSuperior 1.0

https://www.youtube.com/watch?v=PfYifUFmXk8

Introdução a JPA e Hibernate
https://www.youtube.com/watch?v=CAP1IPgeJkw

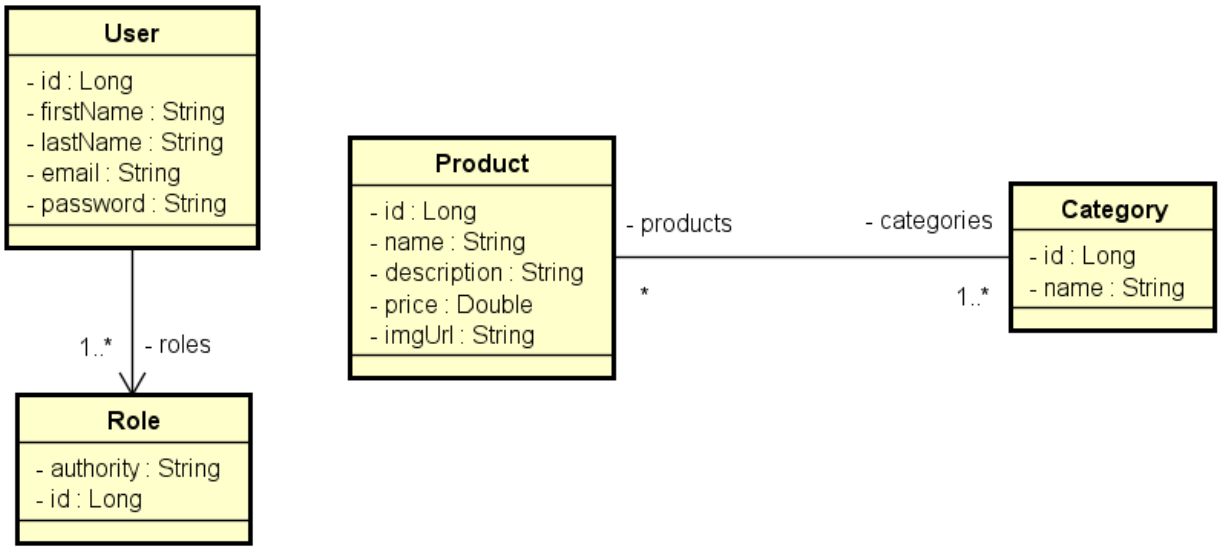
Github do DSCatalog

https://github.com/devsuperior/dscatalog-resources

Figma do DSCatalog

https://www.figma.com/file/1n0aifcfatWv9ozp16XCrq/DSCatalog-Bootcamp

Modelo conceitual do DSCatalog



Desafio para entregar (trabalho final do capítulo)

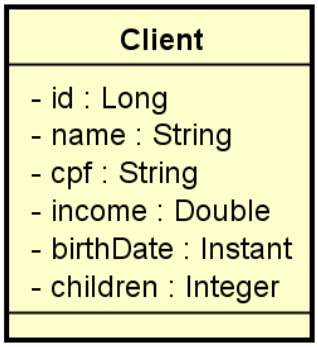
Enunciado

Você deverá entregar um projeto Spring Boot 2.4.x contendo um CRUD completo de web services REST para acessar um recurso de clientes, contendo as cinco operações básicas aprendidas no capítulo:

- Busca paginada de recursos
- Busca de recurso por id
- Inserir novo recurso
- Atualizar recurso
- Deletar recurso

O projeto deverá estar com um ambiente de testes configurado acessando o banco de dados H2, deverá usar Maven como gerenciador de dependência, e Java 11 como linguagem.

Um cliente possui nome, CPF, renda, data de nascimento, e quantidade de filhos. A especificação da entidade Client é mostrada a seguir (você deve seguir à risca os nomes de classe e atributos mostrados no diagrama):



Seu projeto deverá fazer um seed de pelo menos 10 clientes com dados SIGNIFICATIVOS (não é para usar dados sem significado como “Nome 1”, “Nome 2”, etc.).

> ***Atenção:*** *lembre-se de que por padrão a JPA transforma nomes de atributos em camelCase para snake_case, como foi o caso do campo imgUrl do DSCatalog, que no banco de dados tinha o nome img_Url. Assim,* ***o campo birthDate acima será criado no banco de dados como birth_Date, então seu script SQL deverá seguir este padrão****.*

Como o trabalho será corrigido?

1) Importação do projeto

O professor deverá ser capaz de fazer um simples clone do projeto Github, e importar e executar o mesmo no STS sem necessidade de qualquer configuração especial diferente daquelas das aulas.

2) Testes manuais no Postman

O professor já terá preparado em seu computador as requisições Postman abaixo. Todas elas deverão funcionar corretamente:

Busca paginada de clientes

```
GET /clients?page=0&linesPerPage=6&direction=ASC&orderBy=name
```

Busca de cliente por id

```
GET /clients/1
```

Inserção de novo cliente

```
POST /clients
{
  "name": "Maria Silva",
  "cpf": "12345678901",
  "income": 6500.0,
  "birthDate": "1994-07-20T10:30:00Z",
  "children": 2
}
```

Atualização de cliente

```
PUT /clients/1
{
  "name": "Maria Silvaaa",
  "cpf": "12345678901",
  "income": 6500.0,
  "birthDate": "1994-07-20T10:30:00Z",
  "children": 2
}
```

Deleção de cliente

```
DELETE /clients/1
```

Exemplo de uma correção de trabalho:

https://youtu.be/uV2aLDw0Hjw

Collection do Postman

https://www.getpostman.com/collections/8f7f24addf4ecba59fc1