Guia Java Spring Expert

Sobre o Java Spring Expert

Java Spring Expert é um treinamento que corresponde à etapa 2 da jornada Ultimate de formação Java Spring da Devsuperior. O treinamento aborda tópicos de nível básico, intermediário e avançado, com foco em tópicos de nível intermediário.

O que faz o Java Spring Expert ser um passo adiante do Java Spring Essential, é principalmente seu foco em testes de software e práticas relacionadas para elevar em muito a qualidade e finalização dos projetos de back end com API Rest.

Neste treinamento vamos trabalhar muito com testes automatizados, desde os fundamentos, até testes de unidade, de integração, de API, caixa branca/preta, TDD, cobertura de testes, e muito mais, além de vários novos desafios onde você consolidará seu aprendizado. Vamos focar em ferramentas tais como JUnit, Mockito, MockMvc, Jacoco e RestAssured. Importante ressaltar que aqui você vai ter desafios com TDD, ou seja, você vai receber um projeto onde os testes já estão escritos previamente, e sua tarefa será implementar as funcionalidades para fazer os testes passarem. Essa prática é muito importante, e muito usada em processos seletivos de empresas.

Conteúdo programático

Módulo 1: Preparação do projeto

- Setup do projeto DSCatalog
- Banco de dados H2, camadas
- Criação de entidades
- Transações e sessão JPA
- Seeding da base de dados
- DTO
- Criando um ambiente de execução no Postman
- Tratamento de exceções
- Operações de CRUD
- Métodos GET, POST, PUT, DELETE
- Dados de auditoria
- Paginação
- Revisão modelo relacional N-N
- Mapeamento JPA N-N

Módulo 2: Testes automatizados

- Tipos de teste
- Benefícios dos testes automatizados
- O que é TDD
- Boas práticas para testes
- JUnit
- Testes Java vanilla
- Padrão de projeto Factory para instanciar objetos
- Exercícios JUnit vanilla
- Testes com Spring, principais annotations
- Testes de repository
- Fixtures no JUnit, BeforeEach
- Exercício com repository
- Mockito vs MockBean
- Testes de unidade da camada Service
- Imports estáticos do Mockito
- Simulando comportamentos diversos com Mockito
- Exercício testes de unidade com Mockito
- Testes na camada web
- Legibilidade e negociação de conteúdo
- Exercício testes na camada web
- Testes de integração
- Teste de integração na camada web
- Implementando o desafio TDD resolvido
- DESAFIO: TDD Event-City

Módulo 3: Validação e segurança

- Implementando entidades User e Role, ORM, seed
- Introdução ao Bean Validation
- Anotações básicas
- Tratando exceção MethodArgumentNotValidException
- Resposta customizada para erro de validação
- Implementando um ConstraintValidator customizado
- Inclusão de segurança ao projeto
- OAuth2, JWT
- Pré-autorizando métodos por perfil de usuário
- DESAFIO: Validação e segurança

Módulo 4: Consultas, finalização do DSCatalog

- Consulta detalhada de produtos
- Problema N+1 consultas
- Correção nos testes automatizados após mudanças
- Configuração de CORS
- DESAFIO: MovieFlix domínio
- DESAFIO: MovieFlix casos de uso

Módulo 5: Cobertura de testes com Jacoco

- Setup do Jacoco no projeto Spring Boot
- Seleção de pacotes para cobertura
- Fluxos de cobertura, caixa branca
- Análise e implementação de testes de unidade
- Mock de dependências com Mockito
- Mocks relacionados a segurança
- Relatório de cobertura
- DESAFIO: Cobertura Jacoco

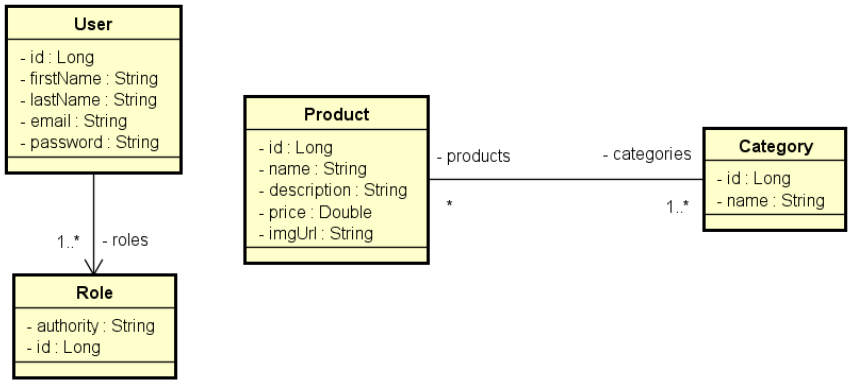
Módulo 6: Testes de API com RestAssured

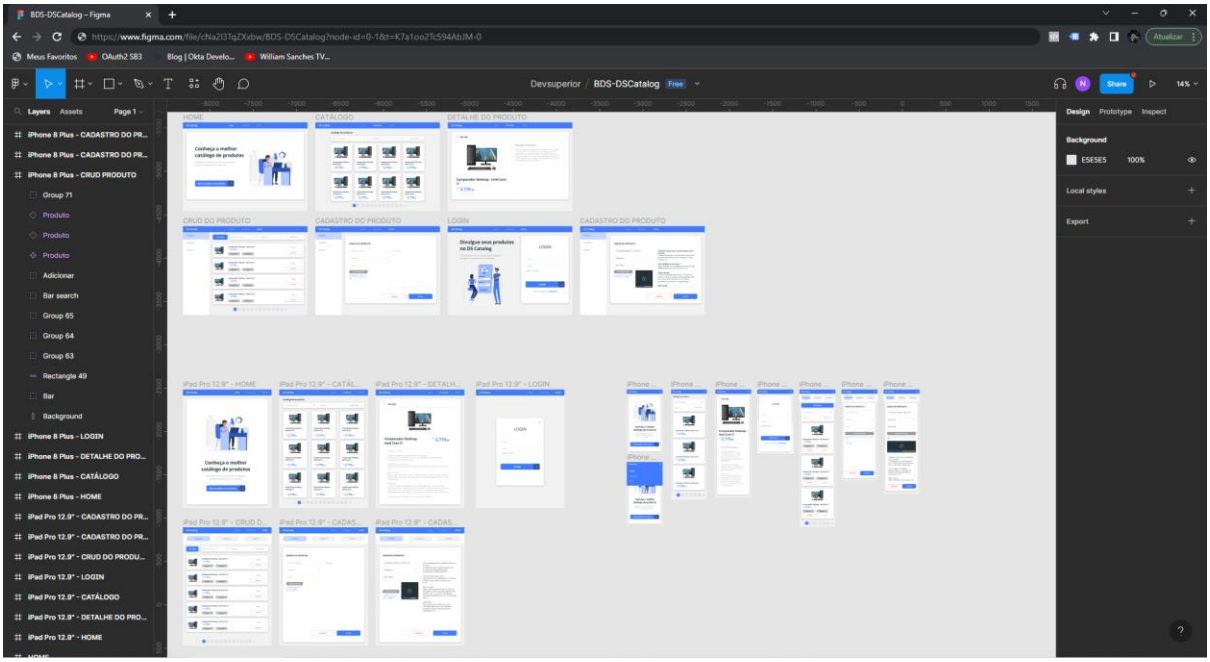
- Setup do Spring Boot com RestAssured
- Domain-specific language (DSL) do RestAssured
- Análise e implementação de testes de API
- Considerações de integração e seed da base de dados
- Paralelo MockMvc e RestAssured
- DESAFIO: Teste de API com RestAssured

Projetos

Projeto DSCatalog

Este é o projeto principal da trilha do Java Spring Expert. O projeto consiste em um sistema de catálogo de produtos, onde o objetivo do projeto é ensinar aos alunos como estruturar com projeto completo com camadas, boas práticas, validação, segurança, tratamento de exceções, consultas ao banco de dados, testes automatizados, dentre outros recursos. Esse projeto também será nosso objeto de estudo para aplicar os tópicos do conteúdo programático da trilha.

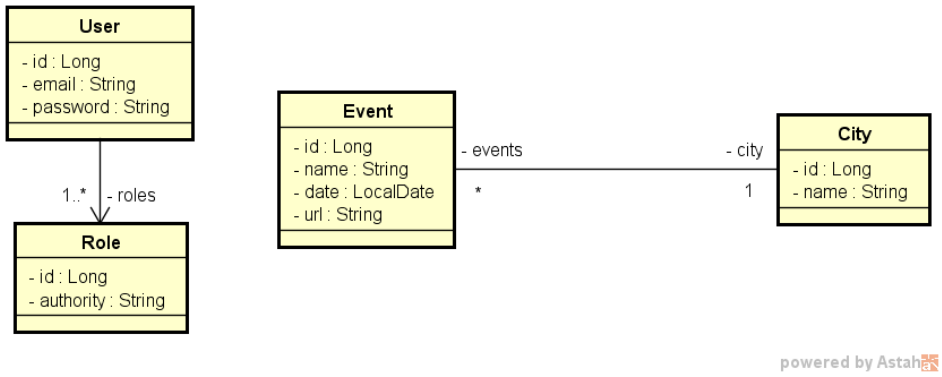




Nota: este é o design Figma do sistema, que é usado para nos guiar durante o desenvolvimento do back end. Porém a construção do front end com as telas gráficas não faz parte desta formação, que é de back end, e não de front end.

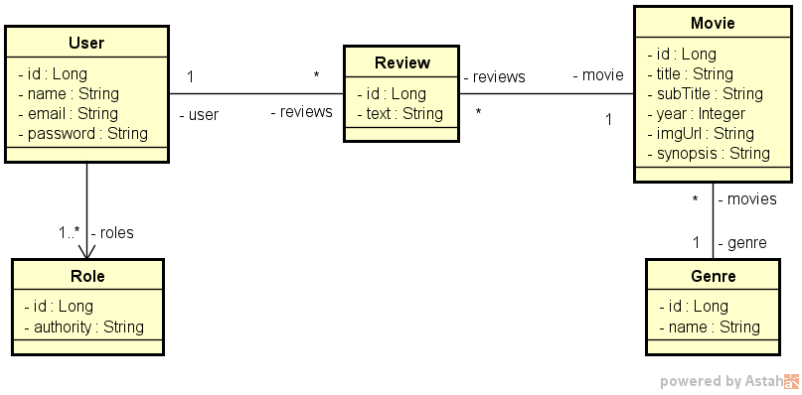
Projeto Event-City

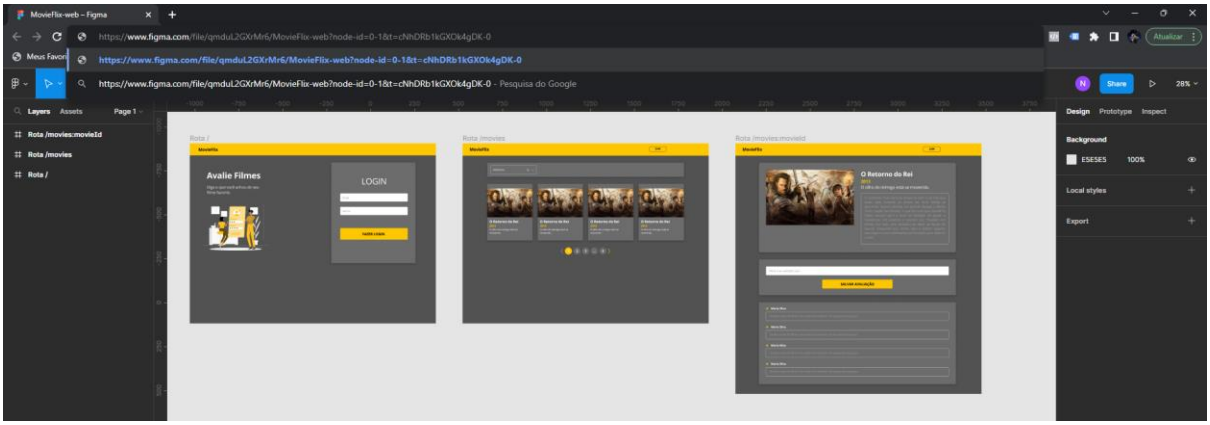
Este projeto é utilizado em dois dos desafios que o aluno terá que realizar durante a trilha de aprendizado. O primeiro deles é para avaliar a competência do aluno no desenvolvimentos de funcionalidades baseado no princípio TDD, onde testes automatizados são escritos previamente como especificação. Esta competência é muito importante, e é muito utilizada em testes de entrevistas de emprego. No segundo desafio, o aluno será avaliado em sua competência para estruturar a segurança de um projeto Spring Boot, com login e controle de acesso por perfis de usuário, além de outros aspectos tais como validação de dados, testes automatizados, dentre outros.



Projeto MovieFlix

O projeto MovieFlix é um sistema no qual os usuários podem navegar em um catálogo de filmes e enviar avaliações sobre os filmes. Este projeto é utilizado em dois dos desafios que o aluno terá que realizar durante a trilha de aprendizado. No primeiro desafio o aluno deverá implementar o modelo de domínio e o seed da base de dados, estruturar o projeto com camadas, validações, segurança e tratamento de exceções, e também recuperar os dados de perfil do usuário. No segundo desafio, o aluno deverá implementar o escopo dos casos de uso do sistema, que consiste no login, navegação dos filmes com busca paginada, filtros, visualização de detalhes, e por fim o registro de avaliações dos filmes.

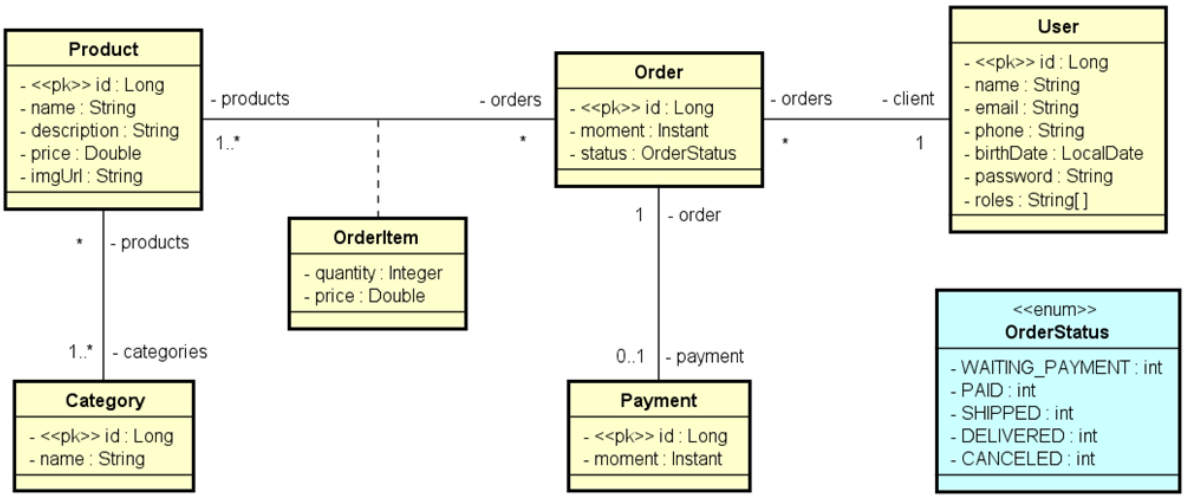


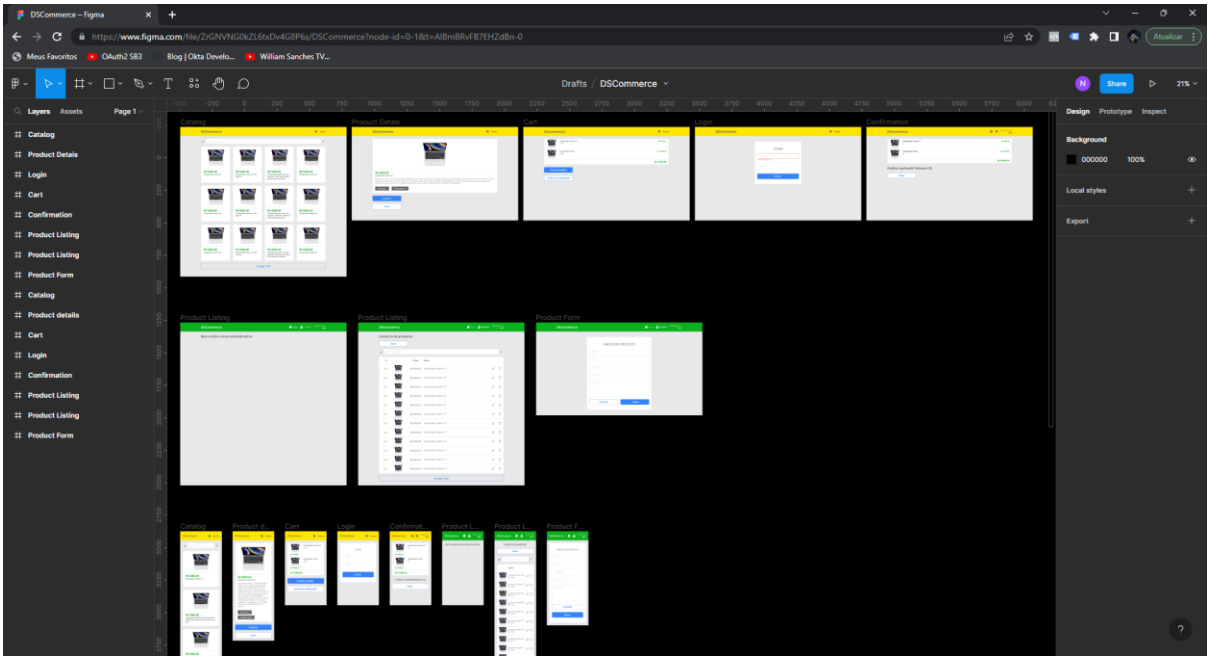


Nota: este é o design Figma do sistema, que é usado para nos guiar durante o desenvolvimento do back end. Porém a construção do front end com as telas gráficas não faz parte desta formação, que é de back end, e não de front end.

Projeto DSCommerce

Este é um projeto de pedidos que nós desenvolvemos passo a passo no Java Spring Essential, e que aqui nós disponibilizamos para que seja utilizado para as aulas sobre cobertura de testes com Jacoco. O projeto consiste em um sistema de comércio eletrônico onde é possível realizar funcionalidades tais como manter um cadastro de produtos, pesquisar produtos para fins de catálogo e carrinho de compras, registrar e recuperar pedidos, consultas ao banco de dados, login, controle de acesso, dentre outras.

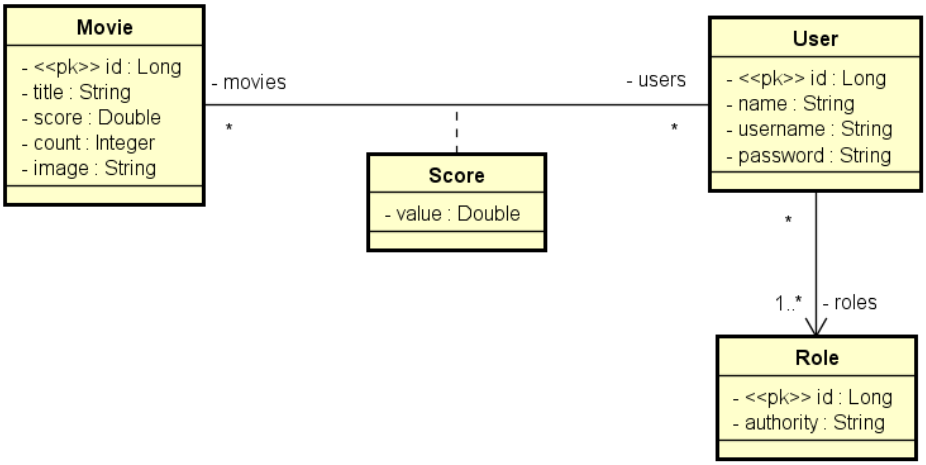


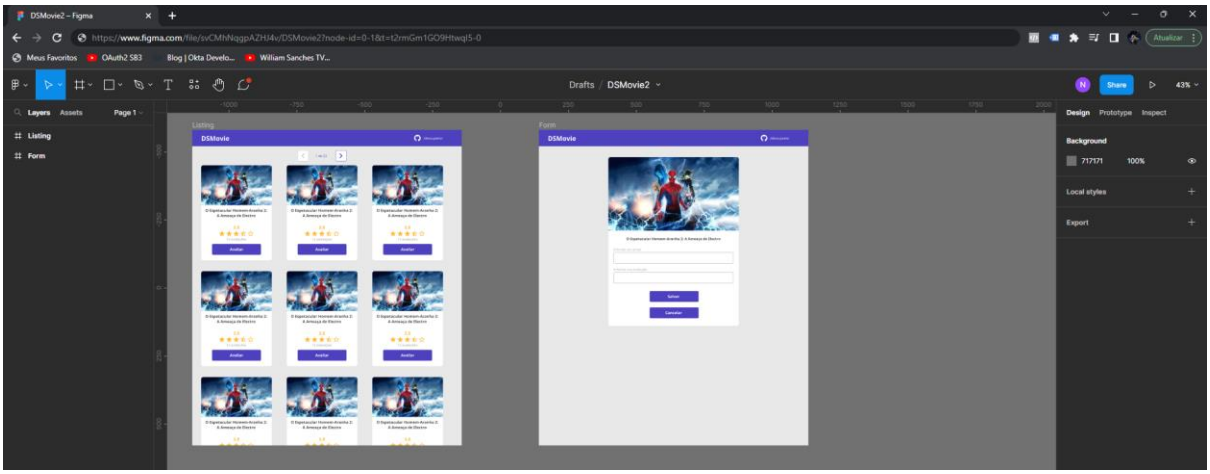


Nota: este é o design Figma do sistema, que é usado para nos guiar durante o desenvolvimento do back end. Porém a construção do front end com as telas gráficas não faz parte desta formação, que é de back end, e não de front end.

Projeto DSMovie

Este projeto é um pequeno sistema de filmes e avaliações dos usuários, onde são explorados os casos de uso de CRUD de filmes, e registro de avaliações, de modo que a cada registro de avaliação, os dados calculados média geral e a contagem de avaliações são armazenados de forma a otimizar o acesso aos dados.





Nota: este é o design Figma do sistema, que é usado para nos guiar durante o desenvolvimento do back end. Porém a construção do front end com as telas gráficas não faz parte desta formação, que é de back end, e não de front end.

Certificação

Ao concluir com sucesso os desafios do Java Spring Expert, você receberá um certificado de conclusão de 200 horas, correspondente ao tempo de aulas e de estudo.

Guia de atualização Spring Boot 3

Versão do Java

O código utilizado nas aulas é compatível com Java **17** e **21**.

Arquivo import.sql

O nome do arquivo SQL para seed da base de dados antes era possível ter o nome **data.sql**, porém agora o padrão é **import.sql**

Pacote jakarta nas especificações do Java Enterprise Edition

Antes especificações como JPA e Servlet ficaram em pacotes iniciados com a palavra javax, enquanto que agora é jakarta. Exemplos:

```
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;

import jakarta.servlet.http.HttpServletRequest;
```

Interface Serializable

Antes era preciso fazer com que algumas classes, como por exemplo as chaves compostas (pk), implementassem a interface Serializable (que permite que os objetos sejam convertidos em bytes para serem salvos em arquivos, trafegarem em rede, etc.), porém agora não é mais necessário.

Método getOne e getReferenceById do JpaRepository

O método getOne do JpaRepository foi marcado para descontinuação (deprecated), e substituído pelo getReferenceById.

Captura de exceção para deleção de entidade de id inexistente

Antes era possível capturar a exceção **EmptyResultDataAccessException** de forma natural quando se tentava deletar uma entidade de id inexistente. Porém agora essa exceção não é mais capturada, então preferimos fazer um teste manual com o método existsById(id) do JpaRepository.

Configurações do H2 no arquivo application-test.properties

Atualizamos a forma como as configurações do banco de dados H2 são feitas no arquivo application-test.properties do projeto.

```
# H2 connection
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.username=sa
spring.datasource.password=
# H2 client
spring.h2.console.enabled=true
spring.h2.console.path=/h2-console
# Show sql
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
```

Mudanças no Spring Security

O Spring Security recentemente marcou vários métodos para descontinuação (deprecated), e alguns detalhes de como se implementam os filtros de configuração de segurança foram alterados.

Mudanças no Spring OAuth2

A partir do Spring Boot 3 o padrão para se implementar OAuth2 nas aplicações mudou drasticamente, pois novas bibliotecas passaram a ser o padrão do framework.