### Exercício 15.4 – Gestão de Inventário com Filtros, Relatórios e Categorias

Este exercício guia a criação de um sistema de inventário com filtros avançados, relatórios de baixo stock, e suporte a categorias. A aplicação utiliza **Spring Boot** com **Vaadin Flow** para a interface gráfica e uma base de dados H2 com persistência em disco.

### Parte 1: Configuração do Projeto no Spring Initializr

1. **Acessar o Spring Initializr:**
   * Vá ao site [Spring Initializr](https://start.spring.io/).
2. **Configurar o projeto:**
   * **Project:** Maven.
   * **Language:** Java.
   * **Spring Boot:** Última versão estável.
   * **Group:** com.example.
   * **Artifact:** inventory.
   * **Name:** inventory.
   * **Package Name:** com.example.inventory.
   * **Packaging:** Jar.
   * **Java Version:** 21.
3. **Adicionar dependências:**
   * **Spring Web:** Para APIs REST.
   * **Spring Data JPA:** Para integração com a base de dados.
   * **H2 Database:** Para persistência em disco.
   * **Vaadin Spring Boot Starter:** Para a interface gráfica.
4. **Gerar o projeto:**
   * Clique em **Generate** para descarregar o projeto como um ficheiro ZIP.
5. **Abrir o projeto no VS Code:**
   * Extraia o ficheiro ZIP.
   * No VS Code, clique em **File > Open Folder** e selecione a pasta do projeto.

### Parte 2: Estrutura do Projeto

1. **Criar as pastas necessárias:**
   * Em src/main/java/com/example/inventory, crie:
     + entity para as classes Item e Category.
     + repository para ItemRepository e CategoryRepository.
     + service para ItemService e CategoryService.
     + views para InventoryView.
2. A estrutura deverá ser:

src/  
├── main/  
│ ├── java/com/example/inventory/  
│ │ ├── InventoryApplication.java  
│ │ ├── entity/  
│ │ │ ├── Category.java  
│ │ │ └── Item.java  
│ │ ├── repository/  
│ │ │ ├── CategoryRepository.java  
│ │ │ └── ItemRepository.java  
│ │ ├── service/  
│ │ │ ├── CategoryService.java  
│ │ │ └── ItemService.java  
│ │ └── views/  
│ │ └── InventoryView.java  
├── resources/  
│ ├── application.properties  
│ └── data.sql

### Parte 3: Implementação do Sistema

O código detalhado das classes já foi fornecido no documento do exercício. Cada seção inclui:

1. **Classes principais e auxiliares:**
   * InventoryApplication: Classe principal para iniciar a aplicação Spring Boot.
   * Item e Category: Classes de entidade com mapeamento JPA.
2. **Repositórios:**
   * ItemRepository: Métodos para pesquisa avançada de itens.
   * CategoryRepository: Gerenciamento de categorias.
3. **Serviços:**
   * ItemService: Contém lógica para CRUD de itens, filtragem e geração de relatórios.
   * CategoryService: Gerencia operações relacionadas a categorias.
4. **Interface Vaadin:**
   * InventoryView: Integra a lógica de front-end com os serviços e repositórios, permitindo:
     + Filtros por nome e categoria.
     + Relatórios de itens com baixo stock.
     + Adição, edição e eliminação de itens.

### Parte 4: Configuração no application.properties

Adicione as configurações para a base de dados H2:

spring.datasource.url=jdbc:h2:file:./data/inventorydb  
spring.datasource.driver-class-name=org.h2.Driver  
spring.datasource.username=sa  
spring.datasource.password=  
spring.jpa.hibernate.ddl-auto=update  
spring.h2.console.enabled=true  
spring.h2.console.path=/h2-console

### Parte 5: Dados Iniciais no data.sql

INSERT INTO category (id, name) VALUES (1, 'Material de Escritório');  
INSERT INTO category (id, name) VALUES (2, 'Eletrónicos');  
  
INSERT INTO item (id, name, quantity, category\_id) VALUES (1, 'Caneta', 100, 1);  
INSERT INTO item (id, name, quantity, category\_id) VALUES (2, 'Lápis', 200, 1);  
INSERT INTO item (id, name, quantity, category\_id) VALUES (3, 'Monitor', 50, 2);

### Parte 6: Testar a Aplicação

1. **Executar o projeto:**
   * Use o comando mvn spring-boot:run.
2. **Acessar o sistema:**
   * Abra <http://localhost:8080> no navegador.
   * Utilize as funcionalidades para filtrar, adicionar, editar e eliminar itens.
   * Acesse o console H2 em <http://localhost:8080/h2-console>.

Se precisar da pasta src compactada, avise! 😊