Módulo 03
Resumo do Capítulo 01 - Java e Spring: Iniciando

Palavras Novas Chaves:

Java, SpringWeb, SpringBoot e RestControler

Resumo:

Nesse módulo vamos começar a programar em Java usando o SpringBoot e SpringWeb. Iremos construir uma aplicação simples e de forma gradual para que você absorva os principais conceitos deste capítulo.

Competências:

1. Criação e manipulação de objetos Java.
2. Manipular erro 404 do HTTP.
3. Manipulação de uma Web API e formato JSON (GET).
4. Reconhecer a estrutura de uma WebAPI.

Repositório do projeto:

https://github.com/bootcampfullstack/bootcamp-devjr-modulo7-cadastro-produto.git

Bibliotecas Utilizadas nesse módulo:

1. Spring https://spring.io/
2. Java SDK: https://jdk.java.net/17/
3. Extensões VSCode.
    a. Extension Pack for Java
    b. Spring Boot Extension Pack

Conteúdo Programático Capítulo 01: Iniciando com Java e Spring

1. Apresentação do capítulo e instalação do Java.
2. Configurando as extensões do Java e Spring no VSCode.
3. Criando um projeto Spring pelo site https://start.spring.io.
4. Criando um projeto Spring pelo VSCode. Passos:
    a. Show All Commands: Control+Shift+P
    b. Spring Initializr: Create Maven Project
    c. Spring Boot Version: 2.6.3 (ou a mais recente)
    d. Language: Java
    e. Group Id: com.abutua
    f. Artifact Id: product-backend
    g. Packing: jar
    h. Java Version: 17 (ou a mais recente)
    i. Dependencies: Spring Web e Spring Boot DevTools
    j. Salvar e abrir o projeto.

5. Analisando a estrutura de um projeto SpringBoot.
6. Criando o primeiro RestController: HelloController.
7. Criando o primeiro endpoint.
8. Criando o monorepo para os projetos do backend e frontend.
9. Construindo o controller: ProductController.java
    a. Crie a pasta *resources* dentro da pasta *productbackend*.
    b. Crie o arquivo ProductController.java
    c. @RestController e @GetMapping
    d. Endpoint /product (versão 1)

Projeto do Capítulo:

1. Crie um repositório (**PÚBLICO**), não use um repositório existente.
2. Crie um GitHub Project para o repositório (**PÚBLICO**).
3. Assistir a aula 33 do capítulo.
4. Criar no mínimo 6 tarefas (issue). Criar tarefas bem definidas com objetivos claros.
5. Ao final de cada tarefa fazer um commit. É esperado 6 commits no mínimo.
6. Critérios de aprovação do projeto:
    a. Ter um repositório único para o projeto.
    b. Ter o branch main.
    c. Ter o project (quadro Kanban) público.
    d. Ter pelo menos 6 tarefas.
    e. Ter todas as tarefas fechadas.
    f. Ter pelo menos 6 commits, um para cada tarefa.
    g. Ser parecido com as imagens de referência.
    h. Implementar todas as funcionalidades apresentadas.

Execução do Backend:

1. Criar uma aplicação Java/SpringBoot e salvar no monorepo.
2. Criar quatro classes:
    - com.abutua.model.Student.java
      Atributos: int id, String name, String email, String phone, int idCurso, int period.
    - com.abutua.model.Course.java
      Atributos: int id, String name.

- com.abutua.controller.StudentController.java
- com.abutua.controller.CourseController.java

3. Desenvolver os seguintes endpoints:
    - GET /students
    - GET /students/{id}
    - GET /courses
    - POST /students
4. Ter exatamente 3 alunos cadastrados ao executar a aplicação com os ids 1, 2 e 3. (Cenário necessário para executar os testes)
5. Ter exatamente 3 cursos cadastrados ao executar a aplicação com os ids 1, 2 e 3. (Cenário necessário para executar os testes)
6. Lista de Tarefas sugeridas pela ordem de execução:
    1. Criar o projeto
    2. Criar as classes do projeto
    3. GET students
    4. GET students/{id}
    5. GET courses
    6. POST students

Execução do FrontEnd:

Adaptar o projeto aluno feito por você no módulo 5 para fazer chamadas HTTP no backend desenvolvido nesse projeto.

Lista de Tarefas sugeridas pela ordem de execução:

1. Carregamento dos alunos na Tabela.
2. Carregamento dos cursos no Select.
3. Botão Salvar.