algaworks

Conteúdo do curso online

IGNIÇÃO SPRING REST

Atualizado em 30/05/2023

Fundamentos sólidos de Spring REST

Você vai aprender os fundamentos de REST definitivamente, entender o que é esse estilo arquitetural e as constraints do REST.

Vai entender como funciona o protocolo HTTP e como isso se encaixa com a modelagem de REST APIs, como os principais códigos de status do HTTP, os principais verbos, idempotência, etc.

Vai desenvolver serviços com diversos métodos HTTP, inclusive vai entender e aprender a implementar *Content Negotiation*.

Spring Boot e Spring MVC

Você vai aprender a criar, configurar e desenvolver uma API do zero e passo a passo, usando alguns dos principais projetos do ecossistema Spring, como Spring Boot e Spring MVC.

Vai aprender a criar projetos com Spring Initializr e importar na IDE, além de entender como o Apache Maven funciona e como Spring Boot tira proveito disso para configurar seu projeto "automagicamente".

E ainda, vai aprender o que é, como gerar e executar um FatJAR gerado pelo Spring Boot.

Jakarta Persistence, Flyway e Lombok

Você vai aprender como fazer mapeamento de entidades e relacionamentos com Jakarta Persistence (JPA).

Ainda, vai ver como criar e evoluir o schema do banco de dados com Flyway e usar Lombok nas classes para reduzir código boilerplate.

Spring Data JPA

Você vai conhecer os principais super poderes do Spring Data JPA (SDJ), como criação de repositórios super inteligentes e criação de *query methods* com filtros.

Validações com Jakarta Bean Validation

Você vai aprender a fazer validações de entradas da sua API de forma profissional, adicionando anotações do Jakarta Bean Validation no seu modelo.

Vai aprender também a customizar mensagens de validação, criar grupos de validações e validar associações em cascata.

A melhor IDE do mercado: IntelliJ IDEA

Vamos usar a IDE que é considerada como a melhor do mercado pelos desenvolvedores Java mais experientes: a IntelliJ IDEA.

Sem problemas se você quiser usar outra IDE, mas a IntelliJ IDEA te trará um nível de produtividade muito alto, por isso recomendamos que você experimente.

Técnicas e boas práticas

Muitos desenvolvedores de APIs não se preocupam ou até desconhecem algumas boas práticas de mercado, por isso um dos objetivos deste treinamento é te ensinar a criar a sua primeira REST API já usando as principais boas práticas.

Você vai aprender a trabalhar com ISO 8601 para troca de dados que envolvam data/hora.

Vai aprender também as boas práticas para nomeação de URIs de recursos e como modelar conceitos abstratos e ações não-CRUD na sua API e recursos de sub-coleção.

Discutiremos sobre as diferenças entre usar as entidades como modelo de representação dos recursos ou DTOs, para desacoplar mais os controllers, além de implementar as duas formas.

Usaremos o ModelMapper para fazer Object Mapping e converter DTOs em entidades e vice-versa.

Vamos implementar e organizar as nossas classes usando alguns conceitos e padrões do DDD (Domain-Driven Design).

Você vai aprender diversas dessas boas práticas e com certeza seu código ficará em um nível de qualidade muito similar aos melhores programadores do mercado que já possuem muitos anos de experiência.

As pessoas não vão acreditar que você acabou de aprender quando verem o seu código! 😂

Tratamento e modelagem de erros da API

Tratar exceptions é muito importante, mas tão importante quanto isso, é devolver como resposta o código de status HTTP adequado e uma representação padrão do problema. Infelizmente, pouca gente faz isso direito.

Mas nesse treinamento você vai aprender a tratar as exceptions e devolver uma resposta adequada e consistente (padronizada) para o consumidor da API.

Você vai aprender a usar a anotação @ResponseStatus, tratar exceções em nível do controlador com @ExceptionHandler, tratar exceções globalmente com @ExceptionHandler, @RestControllerAdvice e ResponseEntityExceptionHandler.

Ainda, você vai aprender o que é e como usar o suporte do Spring à RFC 7807 (Problem Details for HTTP APIs).

Conteúdo programático

1. Fundamentos de REST e Spring
1.1. Boas vindas e as oportunidades do mercado
1.2. Quem é você? Quem sou eu?
1.3. Alguns combinados antes de continuar
1.4. O que é uma API?
1.5. O que é REST?
1.6. Conhecendo o protocolo HTTP
1.7. Entendendo os Recursos REST
1.8. Identificando recursos REST
1.9. Por que Spring?
1.10. Conhecendo o ecossistema de projetos Spring
1.11. Estudos de caso

2. Construindo uma REST API
2.1. Preparando o ambiente de desenvolvimento
2.2. Conhecendo o modelo de domínio do projeto
2.3. Alternativas para criar projetos Spring Boot
2.4. Criando o projeto com Spring Initializr
2.5. Entendendo a estrutura do projeto Maven
2.6. Gerando o FatJAR e iniciando a aplicação
2.7. Implementando e testando a requisição de um recurso
2.8. Implementando uma Collection Resource
2.9. Configurando e usando o Lombok
2.10. Métodos e códigos de status HTTP
2.11. Content Negotiation
2.12. Turbinando a produtividade com DevTools

3. Persistindo os dados
3.1. Configurando a conexão com o banco de dados no projeto
3.2. Conhecendo e adicionando o Flyway no projeto
3.3. Criando a primeira migration com Flyway
3.4. Conhecendo o Jakarta Persistence (JPA)
3.5. Mapeando entidades com Jakarta Persistence
3.6. Implementando uma consulta com JPQL
3.7. Conhecendo o Spring Data JPA (SDJ) e criando um repositório
3.8. Injetando e usando o repositório do SDJ
3.9. Implementando Query Methods no repositório
3.10. Implementando endpoint de busca de recurso
3.11. Implementando endpoint de inclusão de recurso
3.12. Implementando endpoint de atualização de recurso
3.13. Implementando endpoint de exclusão de recurso

4. Evoluindo a API
4.1. Conhecendo e adicionando Jakarta Bean Validation no projeto
4.2. Validando entrada de dados com Jakarta Bean Validation

4.3. Implementando Domain Services
4.4. Implementando regra de negócio para restringir e-mails duplicados
4.5. Capturando exceções do controlador com @ExceptionHandler
4.6. Adicionando migration para criação da tabela de veículos
4.7. Criando e mapeando a entidade de veículo
4.8. Implementando os endpoints de consulta de veículos
4.9. Implementando o endpoint de inclusão de veículos
4.10. Implementando regras de negócio no cadastro de veículos
4.11. Protegendo propriedades somente-leitura
4.12. Validando em cascata
4.13. Validando com Validation Groups

5. Aplicando as boas práticas
5.1. Capturando exceções globais com @RestControllerAdvice
5.2. Usando a RFC 7807 (Problem Details for HTTP APIs)
5.3. Customizando as informações do ProblemDetail
5.4. Adicionando campos customizados no ProblemDetail
5.5. Customizando as mensagens de validação
5.6. Tratando exceções customizadas de forma global
5.7. Boas práticas para trabalhar com data/hora
5.8. Isolando o Domain Model do Representation Model
5.9. Criando o Representation Model do recurso de veículo
5.10. Transformando objetos com ModelMapper
5.11. Implementando assembler de Representation Model
5.12. Compondo objetos no Representation Model
5.13. Criando um Representation Model para entrada de dados

6. Modelando sub-recursos e ações não-CRUD
6.1. Criando e mapeando a entidade de autuação
6.2. Implementando serviço de autuação
6.3. Modelando sub-recursos
6.4. Implementando o endpoint de cadastro do recurso de autuação
6.5. Especializando a exceção de entidade não encontrada
6.6. Implementando recurso de coleção de autuações
6.7. Implementando regras de negócio de apreensão de veículo
6.8. Modelando ações não-CRUD
6.9. Implementando endpoints de ações não-CRUD

7. Alcançando o próximo nível
7.1. Traçando um plano