Aula 5: Configurando Respostas de Erro na API

September 18, 2025

1 Introdução

Nas aulas anteriores, padronizamos os retornos do `MedicoController` com `ResponseEntity`, usando códigos HTTP apropriados: 201 (Created) para cadastro, 200 (OK) para listagem, atualização e detalhamento, e 204 (No Content) para exclusão. Implementamos também o endpoint de detalhamento. Nesta aula, iniciaremos o tratamento de erros, focando na remoção do stack trace em respostas de erro (código 500) e preparando a personalização para retornar 404 (Not Found) em vez de 500 para IDs inexistentes. As alterações serão feitas no `Visual Studio Code` ou sua IDE preferida, com testes no Insomnia.

2 Revisão do Endpoint de Detalhamento

O endpoint de detalhamento, implementado na Aula 4, é:

```
@GetMapping("/{id}")
public ResponseEntity detalhar(@PathVariable Long id) {
    var medico = repository.getReferenceById(id);
    return ResponseEntity.ok(new DadosDetalhamentoMedico(medico));
}
```

Testando no Insomnia com `GET http://localhost:8080/medicos/6`, obtemos o código 200 (OK) com os dados do médico:

```
{
    "id": 6,
    "nome": "Nome do Médico",
    "email": "medico@voll.med",
    "crm": "233444",
    "telefone": "61999998888",
    "especialidade": "ORTOPEDIA",
    "endereco": {
        "logradouro": "rua 1",
        "bairro": "bairro",
        "cep": "12345678",
        "numero": null,
        "complemento": null,
        "cidade": "Brasil",
        "uf": "DF"
    }
}
```

Porém, ao testar com um ID inexistente (`GET http://localhost:8080/medicos/6999`), o método `repository.getReferenceById(id)` lança uma `EntityNotFoundException`, resultando em um erro 500 (Internal Server Error) com um JSON contendo o stack trace:

```
{
    "timestamp": "2022-10-21T17:59:29.080+00:00",
    "status": 500,
    "error": "Internal Server Error",
```

```
    "trace": "jakarta.persistence.EntityNotFoundException: Unable to find
        med.voll.api.medico.Medico with id 6999\n\tat
        org.hibernate.jpa.boot.internal.EntityManagerFactoryBuilderImpl$JpaEntityNotFoundDelegate.
}
```

Expor o stack trace é uma má prática, pois revela informações sensíveis e desnecessárias, representando uma potencial brecha de segurança. Vamos configurar o Spring para remover o stack trace.

3 Removendo o Stack Trace

Para evitar o envio do stack trace em respostas de erro, adicionaremos uma configuração ao arquivo `application.properties` no diretório `src/main/resources`. O arquivo atual contém:

```
spring.datasource.url=jdbc:mysql://localhost/vollmed_api
spring.datasource.username=root
spring.datasource.password=root
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
```

Adicionaremos a propriedade `server.error.include-stacktrace=never`, conforme a documentação do Spring Boot:

```
spring.datasource.url=jdbc:mysql://localhost/vollmed_api
spring.datasource.username=root
spring.datasource.password=root
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
server.error.include-stacktrace=never
```

3.1 Explicação da Configuração

- `server.error.include-stacktrace=never`: Instrui o Spring a não incluir o stack trace nas respostas de erro, reduzindo a exposição de informações sensíveis.

Após salvar a configuração, o Spring DevTools reinicia o projeto automaticamente. Testando novamente no Insomnia com `GET http://localhost:8080/medicos/6999`, o resultado é o código 500 (Internal Server Error), mas sem o stack trace:

```
{
    "timestamp": "2022-10-21T18:06:01.320+00:00",
    "status": 500,
    "error": "Internal Server Error",
    "message": "Unable to find med.voll.api.medico.Medico with id 6999",
    "path": "/medicos/6999"
}
```

4 Limitations do Código 500

Embora a remoção do stack trace seja uma melhoria, o código 500 não é apropriado para este cenário. Quando o ID do médico não existe, o código HTTP correto é 404 (Not Found), indicando que o recurso não foi encontrado. O erro 500 ocorre porque o Spring Data JPA, ao usar `repository.getReferenceById(id)`, lança uma `EntityNotFoundException` para IDs inexistentes. Na próxima aula, personalizaremos o tratamento de erros para retornar 404 em vez de 500.

5 Próximos Passos

Com a configuração inicial para tratamento de erros implementada, a próxima aula abordará a personalização do tratamento de exceções, mapeando a `EntityNotFoundException` para o código 404 (Not Found). Continue praticando no `Visual Studio Code` ou sua IDE preferida!

6 Dica do Professor

- **Aprofunde-se em configurações do Spring**: Consulte a documentação do Spring Boot sobre propriedades de servidor (`https://docs.spring.io/spring-boot/docs/current/reference/html/application-properties.html#application-properties.server`) para explorar outras opções de configuração de erros.
- **Comunidade no X**: Siga perfis como @SpringExpert e @APIWizard no X para dicas sobre configurações de APIs e boas práticas no Spring Boot.
- **Pratique**: Teste outras propriedades do `application.properties`, como `server.error.include-mess` para personalizar ainda mais as respostas de erro no Insomnia.