

11/8/2025

TP1

Desenvolvimento de Serviços com SpringBoot

Professor(a): Flávio da Silva Neves

1. ESCOLHA DA FERRAMENTA DE CONSTRUÇÃO:

Para projetos acadêmicos e pequenos, Maven é mais simples e direto

2. INICIAÇÃO DO PROJETO:

Eu escolhi o Spring Initializr pois ele permite criar rapidamente um projeto Spring Boot configurado com as dependências desejadas, pois eu prefiro algo mais visual e sequencial ao invés da linha de comando.

3. GERENCIAMENTO DE DEPENDÊNCIAS:

Utilizei o arquivo pom.xml do Maven para declarar todas as dependências necessárias.

Os principais benefícios da minha escolha:

- Controle claro das bibliotecas usadas.
- Facilidade para adicionar/remover dependências.

4. UTILIZAÇÃO DE AUTOCONFIGURAÇÃO:

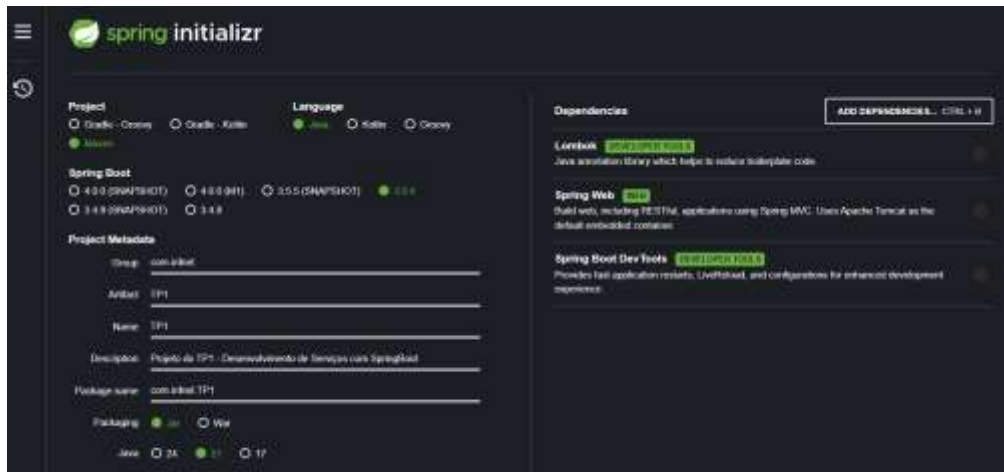
Spring Boot configura automaticamente o contexto da aplicação para o Spring Web, via dependência spring-boot-starter-web, evitando necessidade de configuração manual de servidores, roteamento, e beans.

Utilizei em conjunto o @SpringBootApplication que inclui @EnableAutoConfiguration, no início onde da aplicação, resultando em um código mais limpo, menos repetição de código e foco no negócio.

5. CONFIGURAÇÃO DA IDE:

Utilizei IntelliJ IDEA. Com as seguintes configurações:

- Importação do projeto Maven.
- Ativação do plugin Lombok (instalado no IntelliJ).
- Configuração do JDK 21 para o projeto.
- Configuração do Spring Boot run/debug configurations para executar o projeto com facilidade.



6. DESENVOLVIMENTO DE SERVIÇOS REST

Organização das Rotas

Adição

- GET /adicao?number1=5&number2=3 → Recebe parâmetros via URL.
- POST /adicao → Recebe um JSON no corpo da requisição no formato:

```
json
{
  "number1": 5,
  "number2": 3
}
```

Subtração

- GET /subtracao?number1=5&number2=3
- POST /subtracao → JSON no corpo.

Multiplicação

- GET /multiplicacao?number1=5&number2=3
- POST /multiplicacao → JSON no corpo.

Divisão

- GET /divisao?number1=5&number2=3
- Retorna erro 400 caso number2 seja 0.
- POST /divisao → JSON no corpo, também com validação para divisão por zero.

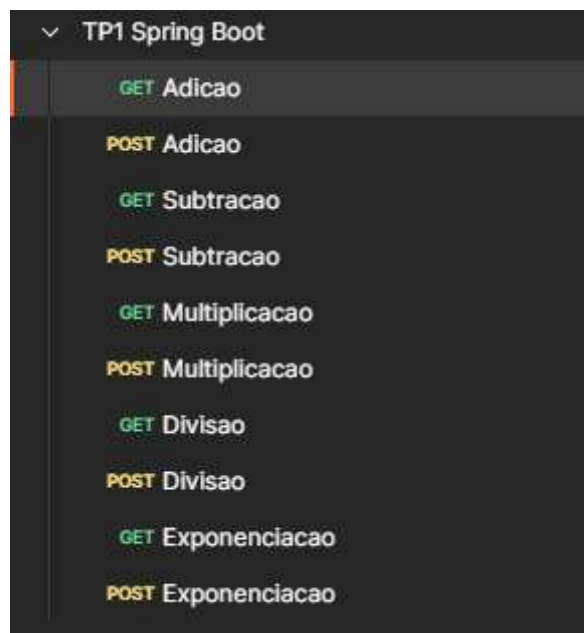
Exponenciação

- GET /exponenciacao?number1=2&number2=3
- POST /exponenciacao → JSON no corpo.

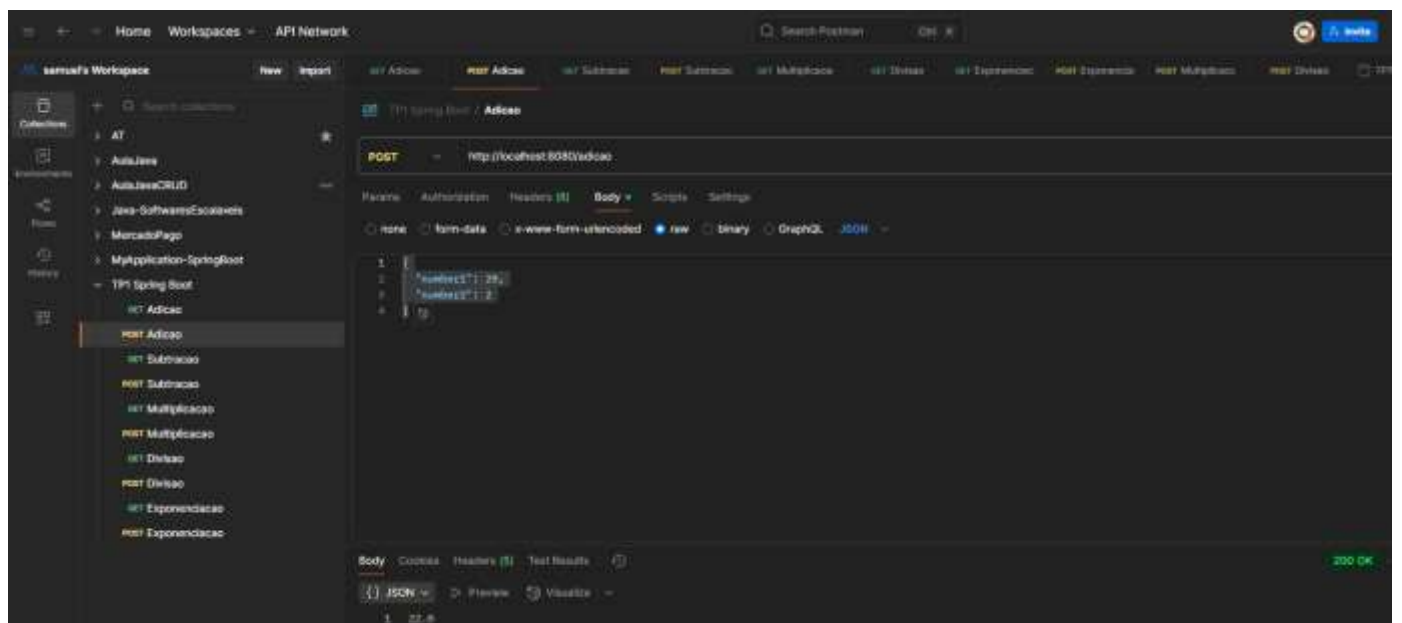
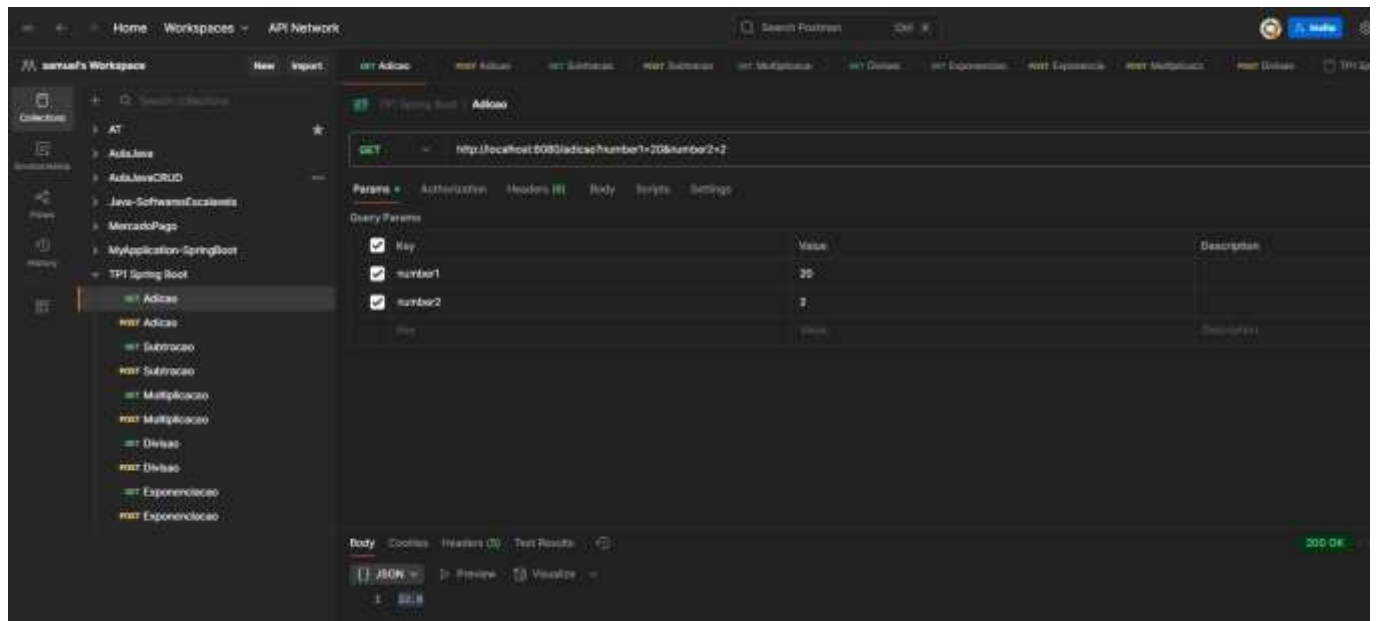
Padrão Adotado

- GET: usado para testes rápidos via query string, ideal para chamadas diretas no navegador ou no Postman sem corpo na requisição.
- POST: usado para enviar os números no corpo da requisição em formato JSON, permitindo melhor integração com sistemas que enviam dados mais complexos.
- Validação: na operação de divisão, foi usado ResponseEntity para retornar mensagens de erro adequadas.
- Modelo de Dados: a classe CalculationModel centraliza os atributos number1 e number2, facilitando a manutenção e o reuso.

Testes dos Endpoints com POSTMAN



Adição – GET / POST



Subtração – GET / POST

The screenshot shows the Postman interface with a workspace named "sarah's Workspace". The left sidebar lists collections, including "TP1 Spring Boot", which contains several endpoints. The "Subtracao" endpoint is selected, and the "GET" method is chosen. The URL is `http://localhost:8080/subtracao/number1=20&number2=2`. The "Query Params" section shows two parameters: "number1" with a value of "20" and "number2" with a value of "2". The "Body" tab is active, showing a status of "200 OK".

GET `http://localhost:8080/subtracao/number1=20&number2=2`

Params Authentication Headers (0) Body Scripts Settings

Query Params

Key	Value	Description
number1	20	
number2	2	

Body Cookies Headers (0) Test Results 200 OK

JSON Preview Visualize

1 - 33.8

The screenshot shows the Postman interface with the same workspace and collection. The "Subtracao" endpoint is selected, and the "POST" method is chosen. The URL is `http://localhost:8080/subtracao`. The "Body" tab is active, showing a status of "200 OK". The body is formatted as JSON and contains the following data:

```
{
  "number1": 20,
  "number2": 2
}
```

POST `http://localhost:8080/subtracao`

Params Authentication Headers (0) Body Scripts Settings

none form-data x-www-form-urlencoded raw JSON Binary GraphQL

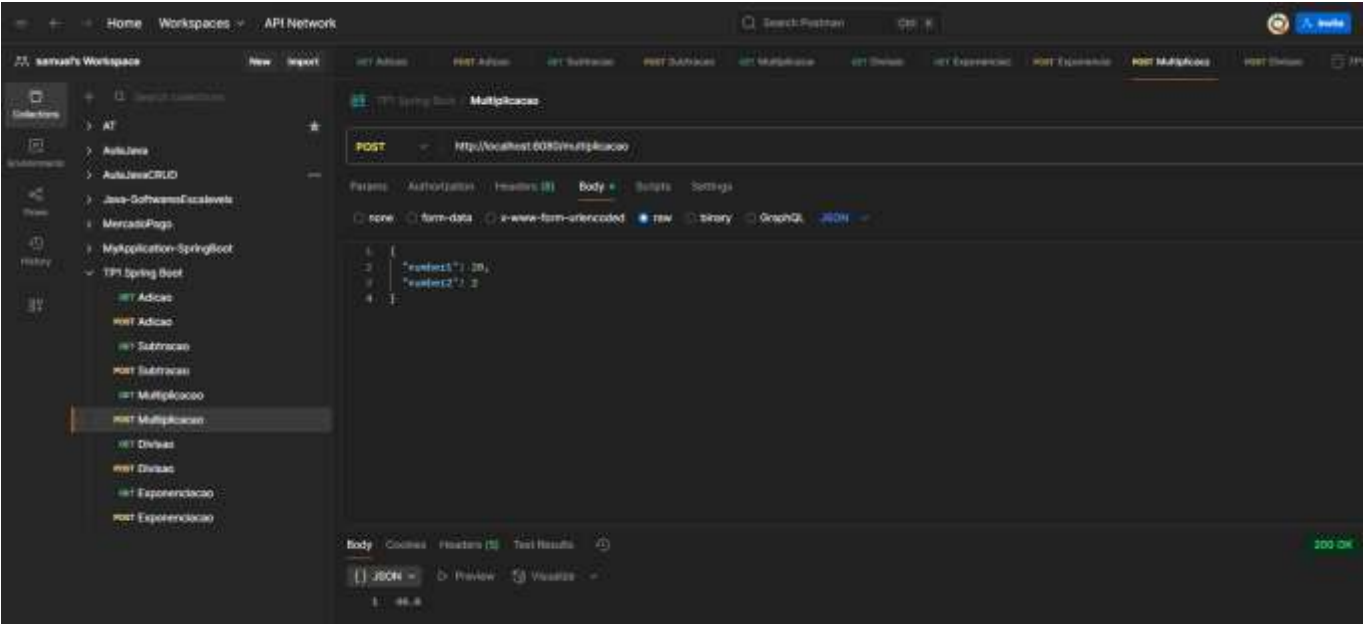
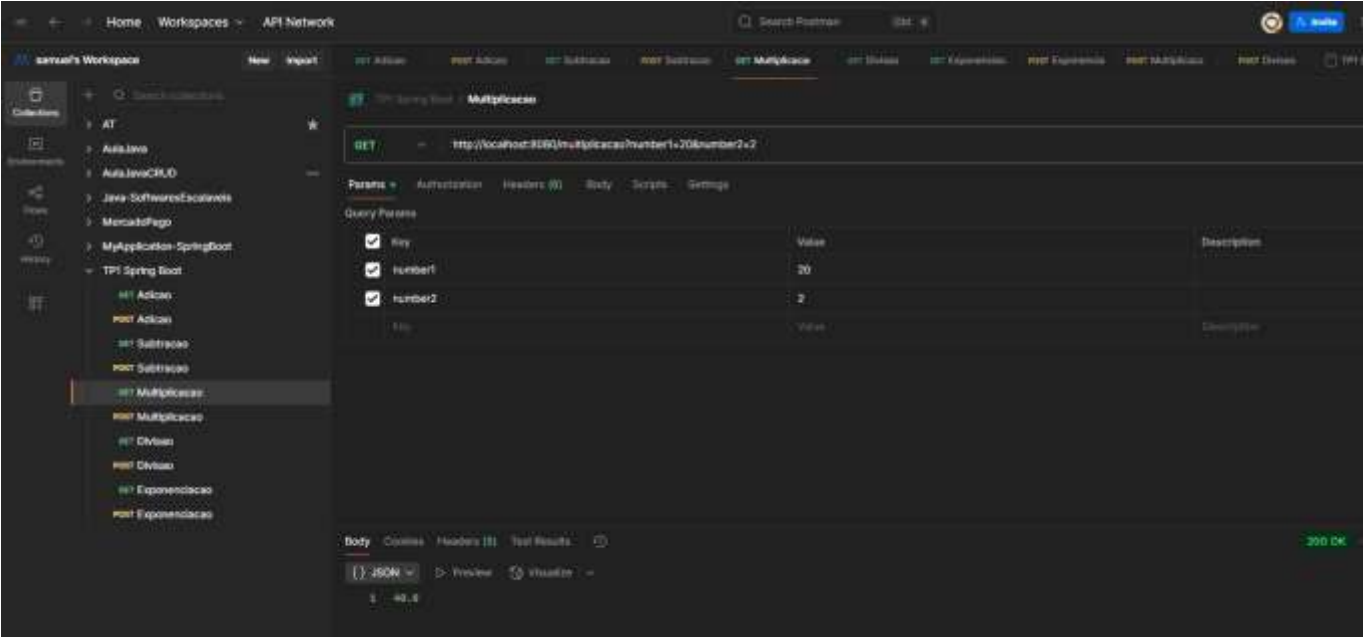
```
1 {
2   "number1": 20,
3   "number2": 2
4 }
```

Body Cookies Headers (0) Test Results 200 OK

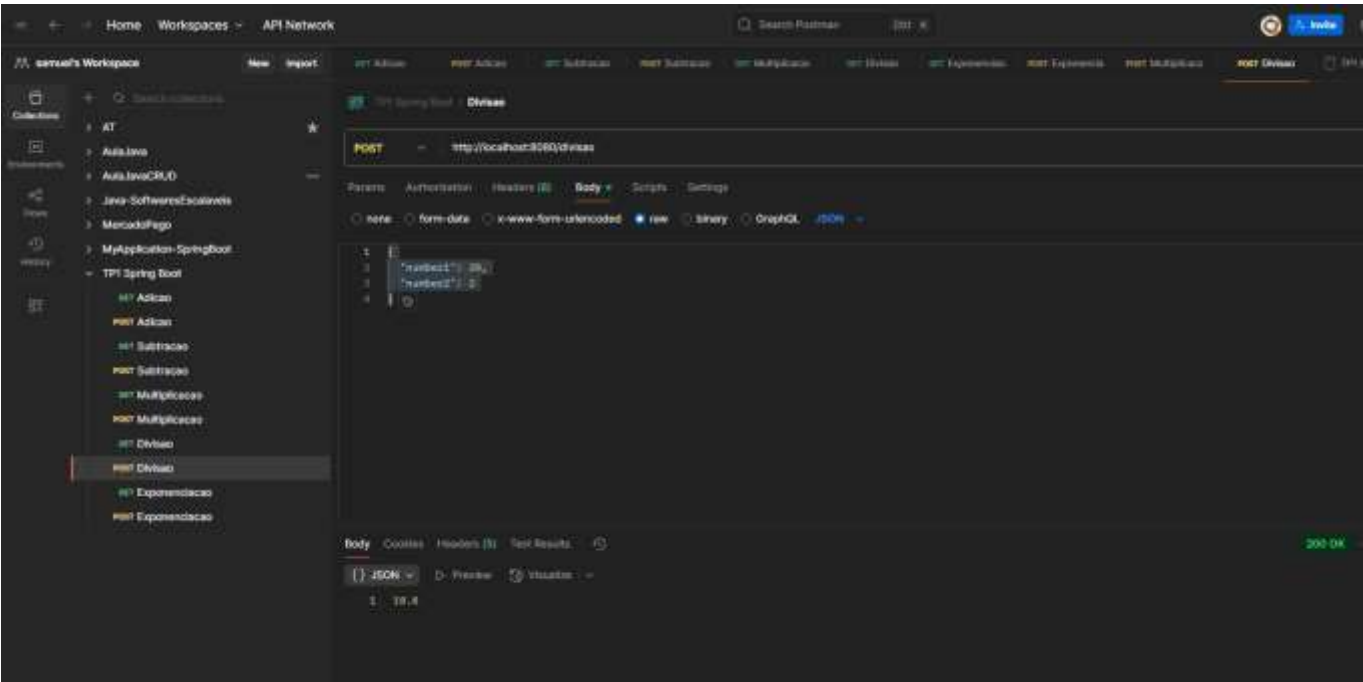
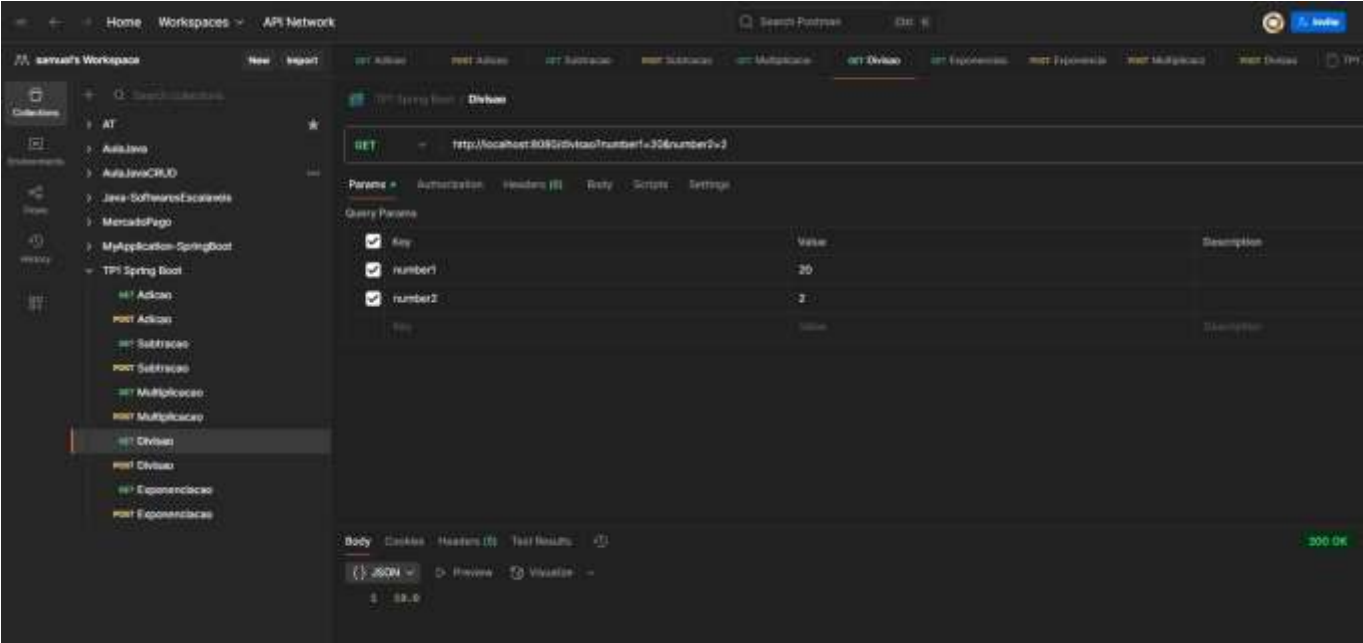
JSON Preview Visualize

1 - 33.8

Multiplicação – GET / POST



Divisão – GET / POST



Exponenciação – GET / POST

