

API REST (Visão Geral)

Uma API REST é um serviço que fornece informações ou funcionalidades para outras aplicações, geralmente por meio do protocolo HTTP.

A comunicação acontece sempre no formato Request -> Response.

Comunicação

Request (Requisição)

- Enviada pelo cliente (frontend, app mobile, Postman, etc.)
- Contém:
 - Método HTTP (GET, POST, PUT, etc.)
 - URL
 - Headers
 - Body (quando necessário)

Response (Resposta)

- Retornada pelo servidor (API REST)
- Contém:
 - Status Code (HTTP)
 - Headers

- Body (dados, mensagem ou erro)

URL Base (Exemplo)

Todos os métodos podem usar a mesma URL base, diferenciando-se pelo verbo HTTP:

<http://localhost:8080/users>

Tipos de Request (Métodos HTTP)

GET - Buscar dados

- Buscar todos os usuários
- Buscar por ID
- Buscar com filtros

POST - Criar dados

- Cria um novo recurso

PUT - Atualizar dados (completo)

- Atualiza todo o recurso
- Necessita do ID

PATCH - Atualização parcial

- Atualiza apenas alguns campos
- Também necessita do ID

DELETE - Excluir dados

- Remove um recurso pelo ID

OBS: Tanto PUT quanto PATCH precisam do ID (identificador), pois a API precisa saber qual recurso será atualizado.

Tipos de Response - Status Code HTTP

2xx - Sucesso

1. 200 - OK
2. 201 - CRIAR
3. 204 - OK(SEM CONTEUDO)

3xx - Redirecionamento

1. 301 - ALTERAÇÃO PERMANENTE
2. 302 - ALTERAÇÃO PROVISÓRIA

4xx - Erro do Cliente

1. 401 - Erro na Senha ou Token.
2. 403 - Sem permissão.
3. 404 - Não encontrado.

5xx - Erro do Servidor

1. 500 - Erro interno no servidor
2. 502 - Erro de comunicação(Bad Request)
3. 503 - Serviço fora do ar.

Spring Boot (Java) - Estrutura da API

Config

- Configurações da aplicação
- Ex: segurança, CORS, beans, propriedades

Controller

- Responsável por receber as requisições HTTP
- Mapeia URLs e métodos

Model / Entity

- Representa a regra de negócio
- Mapeia as tabelas do banco de dados

Service

- Contém a lógica de negócio
- Valida dados
- Manipula o Repository
- Evita regras no Controller

Repository

- Responsável por acessar o banco de dados
- Fornece métodos prontos (CRUD)

Exception

- Tratamento de erros
- Criação de exceções personalizadas

FLUXO PRINCIPAL

Request: Cliente -> Controller -> Service -> Repository -> Banco

Response: Banco -> Repository -> Service -> Controller -> Cliente