# **Exploração Profunda de APIs RESTful com PHP Puro**

**Aluna:** Evellyn de Santana Feliciano  
 **Disciplina:** SW - API  
 **Data:** 23 de Junho de 2025

## **1. Introdução: A Relevância das APIs no Mundo Atual**

Em um cenário digital onde **integrações entre sistemas** são indispensáveis, as **APIs** se destacam como pontes vitais para a comunicação entre aplicações. Elas permitem que softwares distintos troquem dados de forma segura, estruturada e eficiente — algo essencial para e-commerces, aplicativos mobile, plataformas de pagamento, redes sociais, entre outros. O estudo das APIs RESTful em PHP se mostra fundamental para desenvolvedores web que desejam criar soluções modernas, escaláveis e interativas.

## **2. O que é uma API e qual sua função essencial**

### **📌 Definição técnica:**

API (*Application Programming Interface*) é uma interface que define as formas como diferentes softwares podem interagir entre si. Essa interface é composta por um conjunto de regras, rotas e métodos que orientam como os dados são requisitados, enviados, modificados ou removidos.

### **💡 Exemplo real:**

Imagine um aplicativo de delivery. Quando você escolhe um endereço e calcula o frete, o app consulta uma **API de geolocalização** (como a do Google Maps). Ao confirmar o pedido, ele se comunica com a **API do restaurante**. O pagamento? Feito por meio de uma **API de gateway financeiro**, como o Mercado Pago ou Stripe.

APIs **evitam retrabalho**, promovem **modularidade** e permitem o **reaproveitamento de funcionalidades** em larga escala.

## **3. REST x SOAP: Duas filosofias de integração**

REST (Representational State Transfer) e SOAP (Simple Object Access Protocol) são arquiteturas diferentes para criação de APIs. Veja o comparativo técnico:

|  |  |  |
| --- | --- | --- |
| **Característica** | **REST** | **SOAP** |
| Formato de dados | JSON, XML, HTML, texto | Apenas XML |
| Verbos HTTP | Totalmente integrados | Limitado, geralmente POST |
| Curva de aprendizado | Baixa | Alta |
| Leveza | Alto desempenho, menos sobrecarga | Mais pesado por estrutura complexa |
| Suporte a cache | Sim | Geralmente não |
| Segurança | HTTPS + Tokens | Mais robusta via WS-Security |

**REST** é amplamente preferido por sua leveza, simplicidade e compatibilidade com aplicações web modernas. **SOAP** ainda é usado em ambientes corporativos robustos, especialmente onde **transações seguras e contratuais** são críticas.

## **4. Roteamento em APIs RESTful: Como as URLs viram ações**

Em APIs RESTful, cada **recurso** (como usuários, produtos, pedidos) é representado por uma **URL (rota)**. As ações realizadas sobre esses recursos são determinadas pelos **métodos HTTP**.

### **🔍 Exemplo prático de rotas:**

* GET /api/usuarios → Lista todos os usuários
* GET /api/usuarios/8 → Mostra o usuário com ID 8
* POST /api/usuarios → Cria novo usuário
* PUT /api/usuarios/8 → Atualiza dados do usuário 8
* DELETE /api/usuarios/8 → Remove o usuário 8

Essas rotas permitem que sistemas consumidores acessem e manipulem dados de forma padronizada, **seguindo convenções compreensíveis e escaláveis**.

## **5. Métodos HTTP fundamentais: O esqueleto do REST**

### **🔹 GET**

Usado para **consultar recursos**. Exemplo:

http

CopiarEditar

GET /api/produtos

Resposta:

json

CopiarEditar

[  
 {"id": 1, "nome": "Camiseta", "preco": 49.90},  
 {"id": 2, "nome": "Tênis", "preco": 199.99}  
]

### **🔹 POST**

Usado para **criar novos recursos**.

http

CopiarEditar

POST /api/produtos

json

CopiarEditar

{  
 "nome": "Boné",  
 "preco": 59.90  
}

### **🔹 PUT**

Atualiza recursos existentes.

http

CopiarEditar

PUT /api/produtos/1

json

CopiarEditar

{  
 "nome": "Camiseta Premium",  
 "preco": 69.90  
}

### **🔹 DELETE**

Remove o recurso especificado:

http

CopiarEditar

DELETE /api/produtos/1

## **6. Como criar uma API RESTful com PHP puro**

Embora frameworks como Laravel ou Lumen acelerem o desenvolvimento, é possível criar uma API RESTful do zero usando apenas PHP puro, ideal para fins didáticos e projetos simples.

### **📁 Estrutura mínima recomendada:**

bash

CopiarEditar

/api  
 ┣ index.php  
 ┣ /usuarios  
 ┣ listar.php  
 ┣ criar.php  
 ┣ editar.php  
 ┣ deletar.php

### **🧠 Exemplo de roteador básico:**

php

CopiarEditar

<?php  
$request = $\_SERVER['REQUEST\_URI'];  
$method = $\_SERVER['REQUEST\_METHOD'];  
  
if ($request == '/api/usuarios' && $method == 'GET') {  
 include 'usuarios/listar.php';  
} elseif ($request == '/api/usuarios' && $method == 'POST') {  
 include 'usuarios/criar.php';  
} else {  
 http\_response\_code(404);  
 echo json\_encode(["erro" => "Recurso não encontrado"]);  
}  
?>

Esse é o coração da API, onde cada requisição é mapeada para o script correto.

## **7. JSON como alternativa ao banco de dados**

### **✅ Vantagens:**

* Simplicidade extrema (sem necessidade de SGBD)
* Portabilidade e fácil leitura
* Ótimo para projetos pequenos ou protótipos

### **❌ Desvantagens:**

* Baixa escalabilidade (leitura lenta com muitos dados)
* Dificuldade com acessos simultâneos (sem controle de concorrência)
* Risco de corrupção ou perda de integridade

### **📁 Exemplo de arquivo usuarios.json:**

json

CopiarEditar

[  
 {"id": 1, "nome": "Evellyn", "email": "[eve@example.com](mailto:eve@example.com)"},  
 {"id": 2, "nome": "Guilherme", "email": "[gui@example.com](mailto:gui@example.com)"}  
]

### **📜 Código PHP para leitura:**

php

CopiarEditar

<?php  
$dados = file\_get\_contents("usuarios.json");  
$usuarios = json\_decode($dados, true);  
echo json\_encode($usuarios);  
?>

## **8. Conclusão: RESTful em PHP — <3**

Entender APIs RESTful em PHP puro é mais que um exercício técnico: é aprender a **construir pontes entre sistemas**, dominar protocolos da web e desenvolver aplicações interoperáveis com alto potencial de uso no mercado.

A simplicidade do PHP, aliada à organização REST, permite criar APIs limpas, performáticas e facilmente escaláveis. Conhecer essa base te deixa preparada para evoluir para ferramentas mais robustas no futuro — como Laravel, Symfony ou até microserviços em outras linguagens.