**NOME:** Lucas Martins Pereira **ETEC MCM 3C**

**Conceitos e funcionamento de APIs RESTful em PHP sem uso de frameworks**

O conteúdo deve abordar os seguintes pontos:

• O que é uma API e para que serve

• Diferença entre API REST e SOAP

• Como funcionam as rotas em uma API RESTful

• Os principais métodos HTTP (GET, POST, PUT, DELETE)

• Estrutura básica de uma API em PHP puro

• Vantagens e desvantagens de usar arquivos JSON em vez de banco de dados

**API**

Uma API consiste em um conjunto de dados armazenados em um sistema remoto. Basicamente, um conjunto de APIs levam informações de um lado para o outro. Em uma de nossas aulas utilizamos uma API que armazena dados de CEP de todo o mundo, também uma que contém materiais de diversos países, com base nisso, foi possível compreender que as informações já armazenadas podem sem integradas a outro tipo de programação.

**API REST e API SOAP**

Existe uma diferença entre uma API REST e uma API SOAP. A REST utiliza HTTP(protocolo web fundamental) e JSON(intercâmbio de dados) e é mais simples, a SOAP usa XML(estrutura os dados entre cliente e o servidor) e é um pouco mais complexa.

**ROTAS em RESTful**

As rotas de API RESTful resumem-se a descrever o endereço de cada função, por exemplo /cep, /cep/país, como aprendemos anteriormente nas nossas aulas de CNW utilizando o prompt de comando do computador para acessar a área desejada de um projeto.

**MÉTODOS em HTTP**

Em relação aos principais métodos HTTP, são eles: GET: pega uma informação EX: mostra algum dado; POST: envia algo novo EX: criar um usuário; PUT: atualizar algo EX: editar algum cadastro e DELETE: apaga algo EX: exclui um produto.

**ESTRUTURA API em PHP**

A Estrutura básica de uma API em PHP envolve requisitar as informações, utilizar métodos padrões do banco de dados e retornar as respostas em JSON

Exemplo de roteamento simplificado:

<?php

header("Content-Type: application/json");

$method = $\_SERVER['REQUEST\_METHOD'];

$path = explode('/', trim($\_SERVER['REQUEST\_URI'], '/'));

if ($path[0] === 'recursos' && $method === 'GET') {

echo json\_encode(["data" => "Listagem de recursos"]);

} else {

http\_response\_code(404);

echo json\_encode(["message" => "Não encontrado"]);

}

?>

**Vantagens e Desvantagens**

O JSON é mais simples em dados de escala menor, porém é menos escalável e se torna um pouco mais complexo quando relacionados a volumes maiores, logo, os bancos se tornam mais adequados na maioria das APIs.