Comunicação Cliente x Servidor

Realizando requisições assíncronas utilizando a API Fetch

Prof. Eduardo Lino
eduardo.lino@pucpr.br

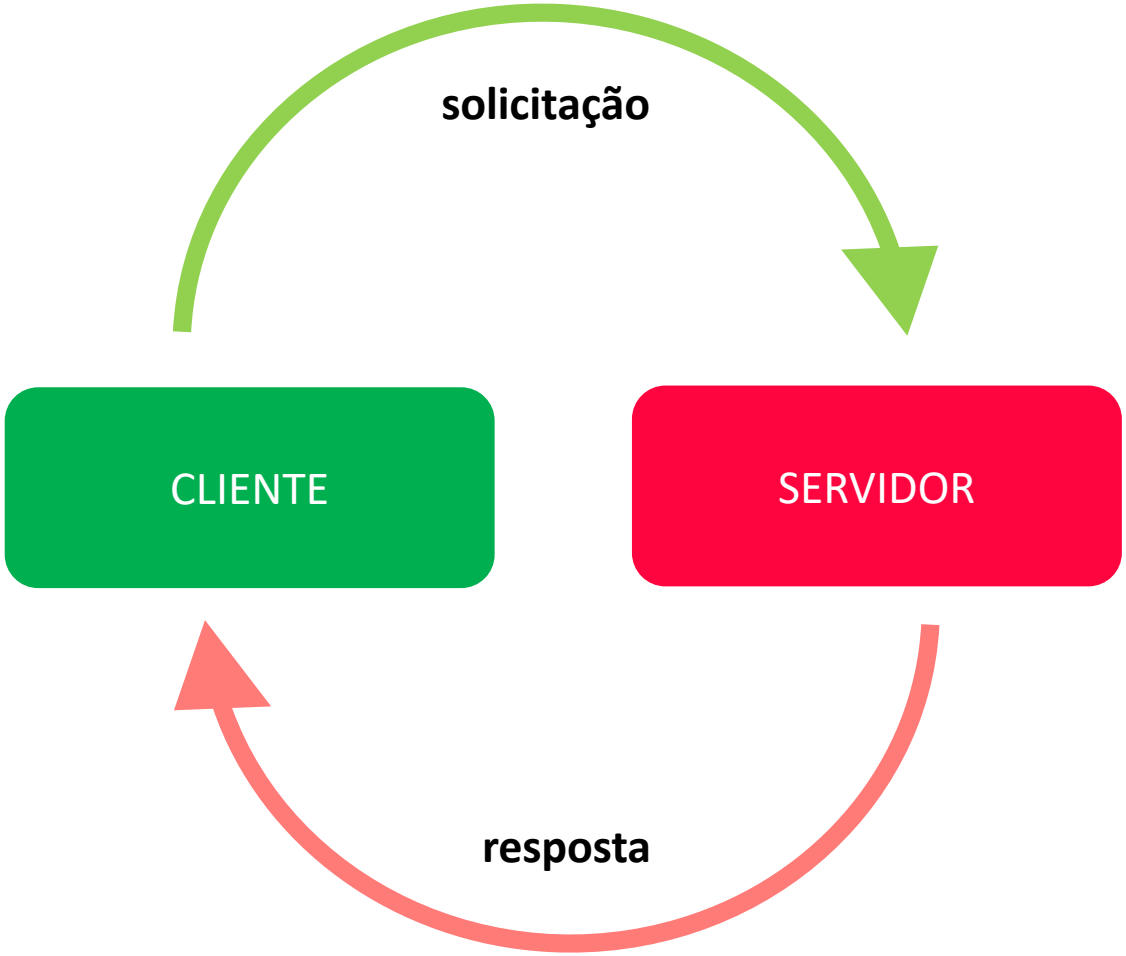
Cliente

- Quem é?
  - Geralmente são microcomputadores ou dispositivos que estão ligados a rede.

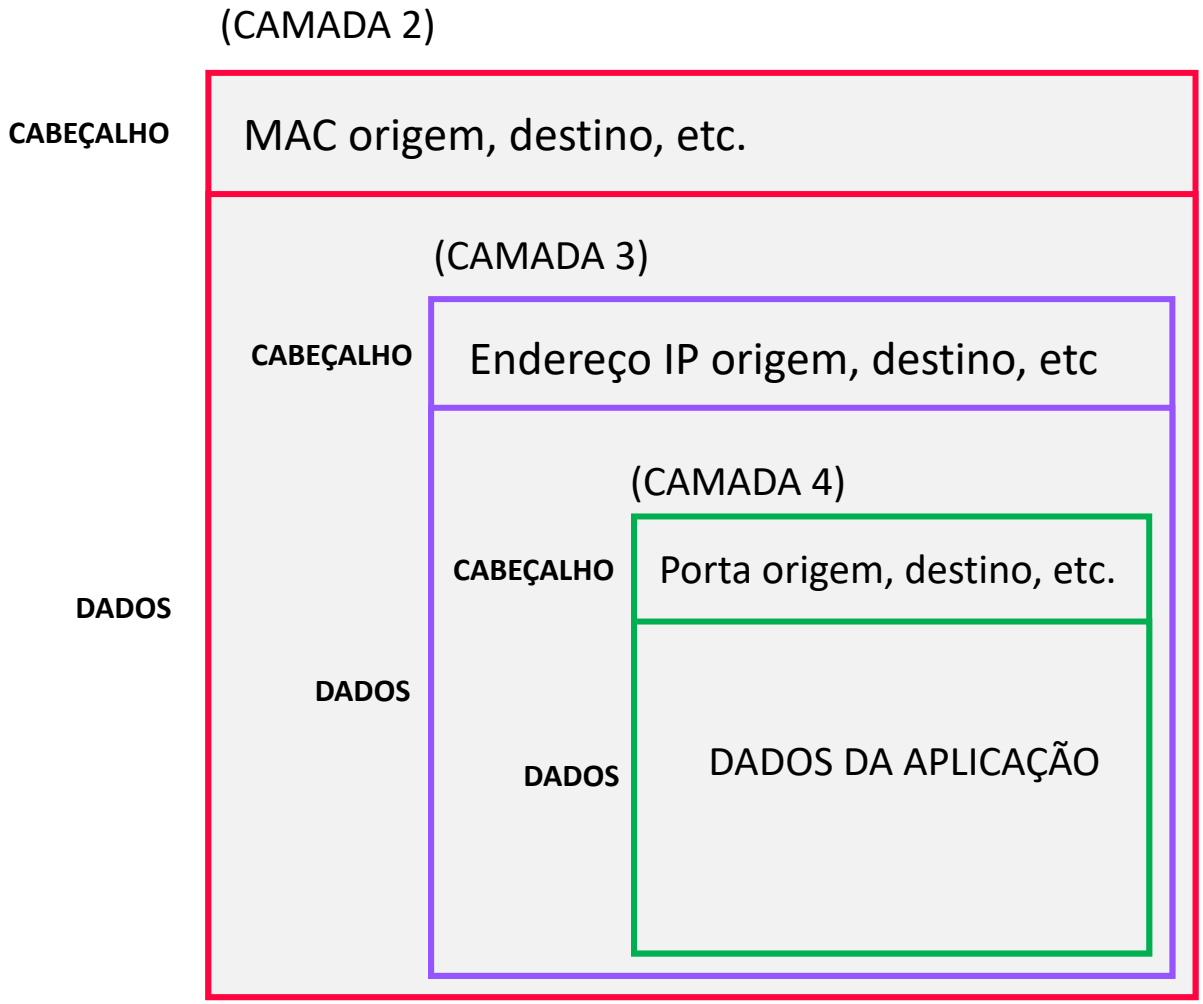
Servidor

- Quem é?
  - Geralmente são equipamentos com maior poder de processamento e armazenamento que fica em espera, aguardando solicitações de **clientes**.

Cliente x Servidor



Encapsulamento



Cliente x Servidor

CLIENTE

SERVIDOR

Cliente x Servidor

CLIENTE

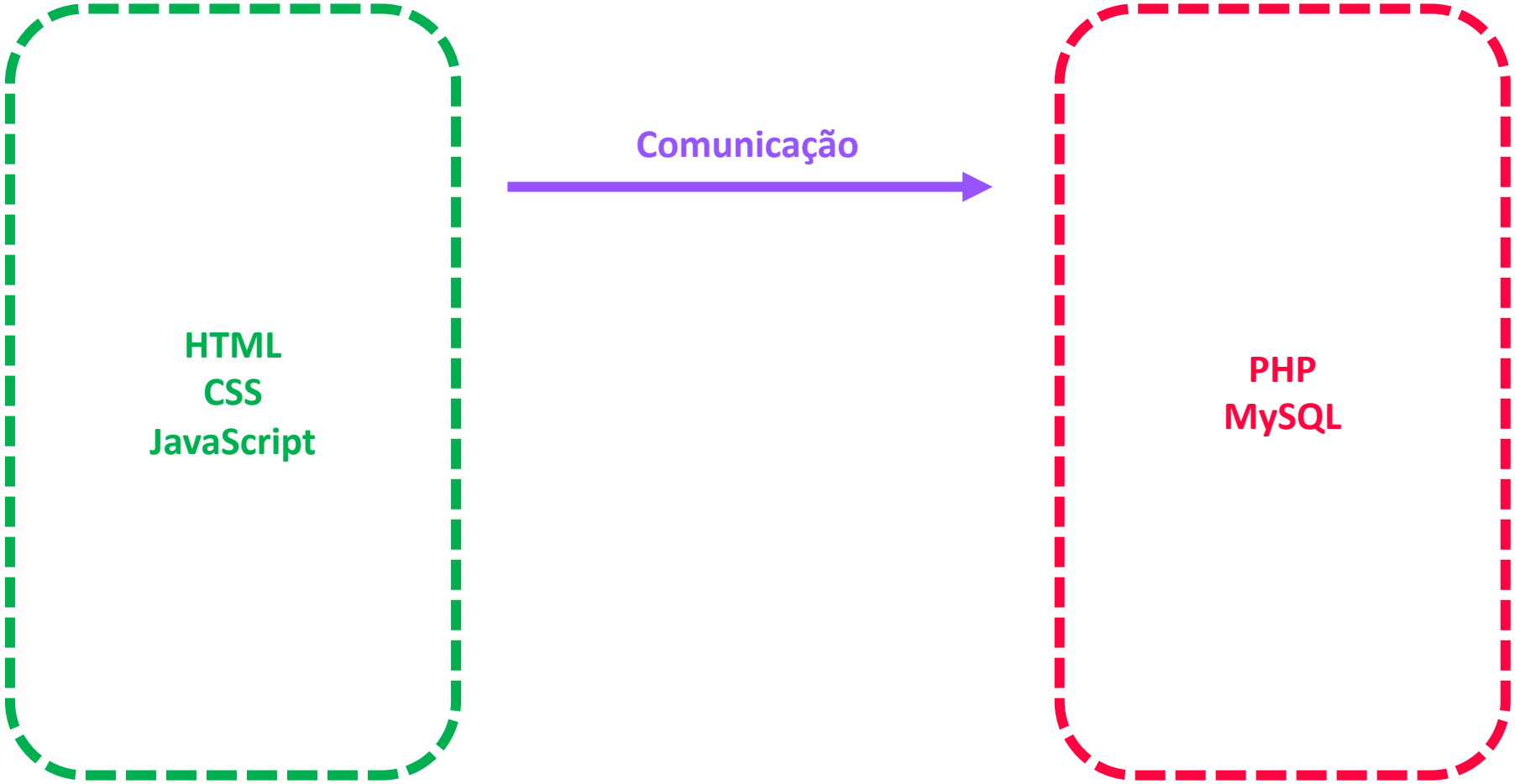
SERVIDOR

Cliente x Servidor

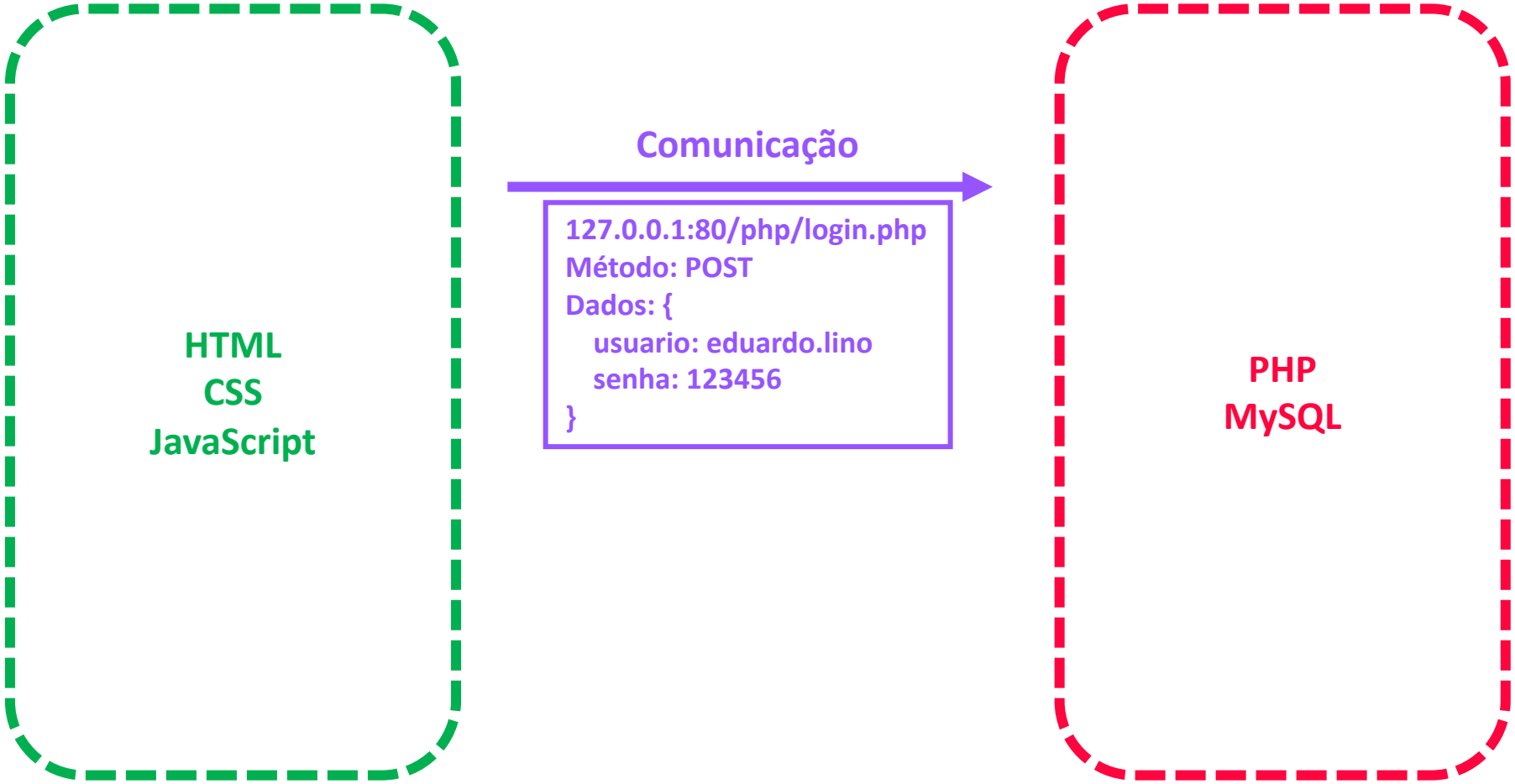
HTML
CSS
JavaScript

PHP
MySQL

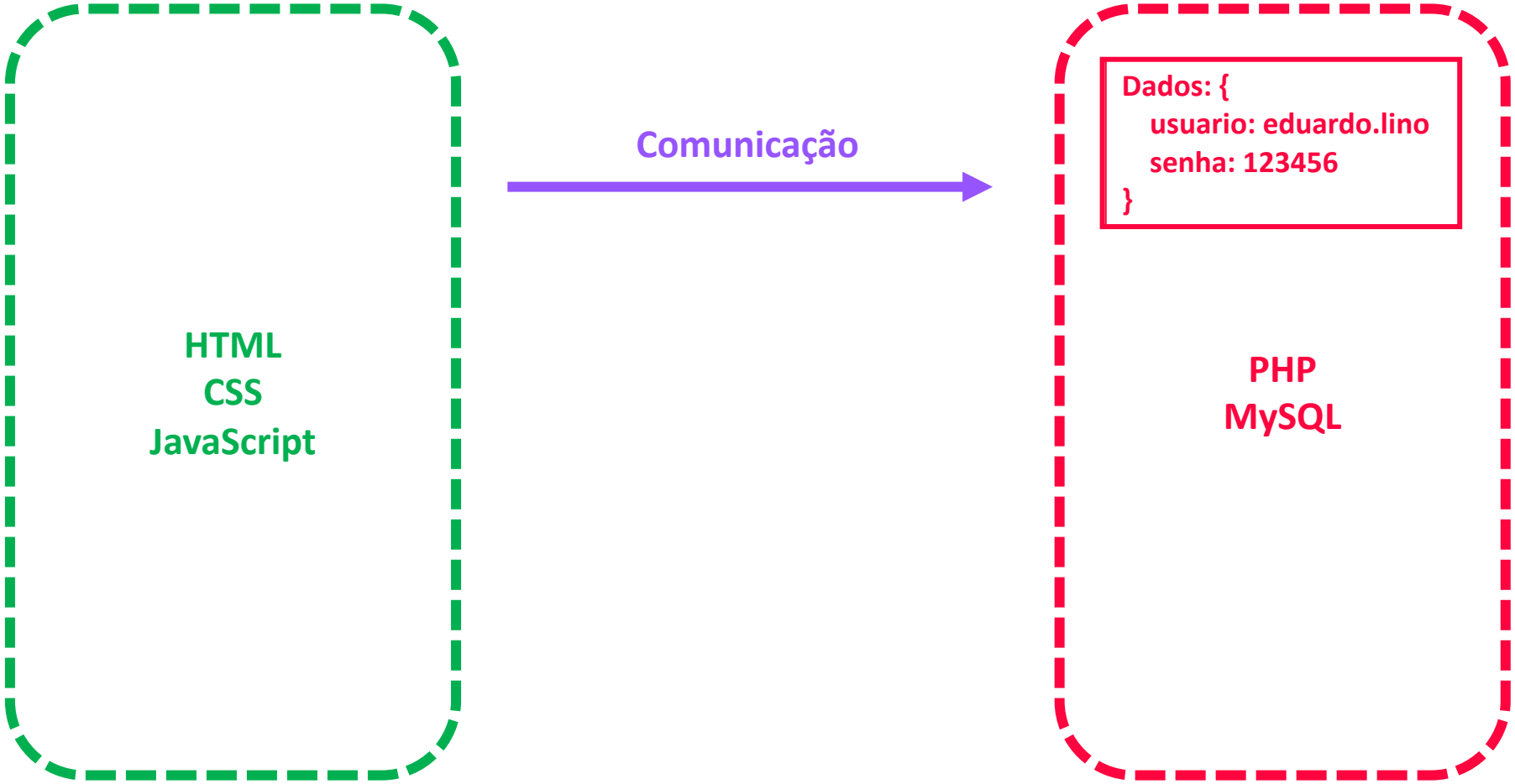
Cliente x Servidor



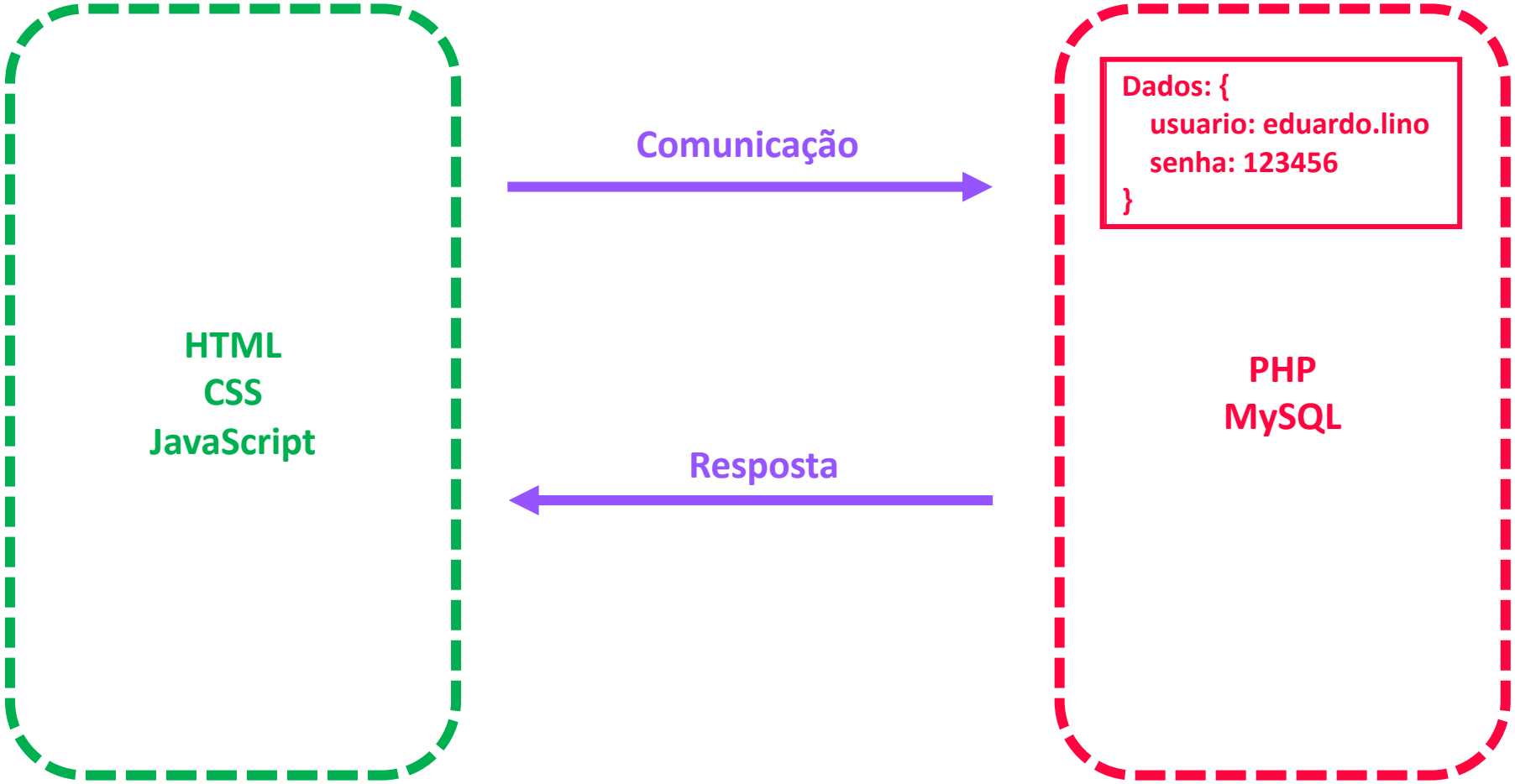
Cliente x Servidor



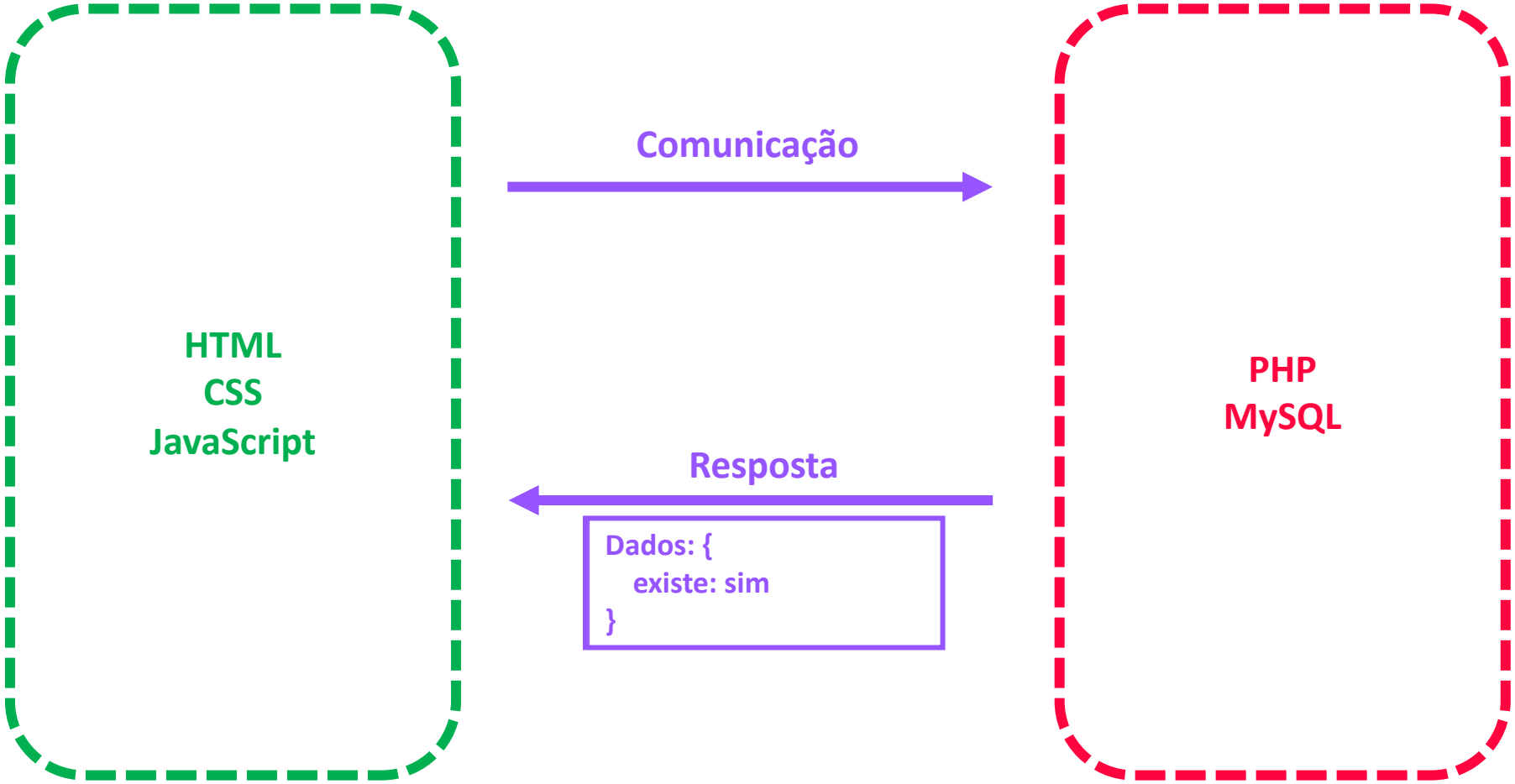
Cliente x Servidor



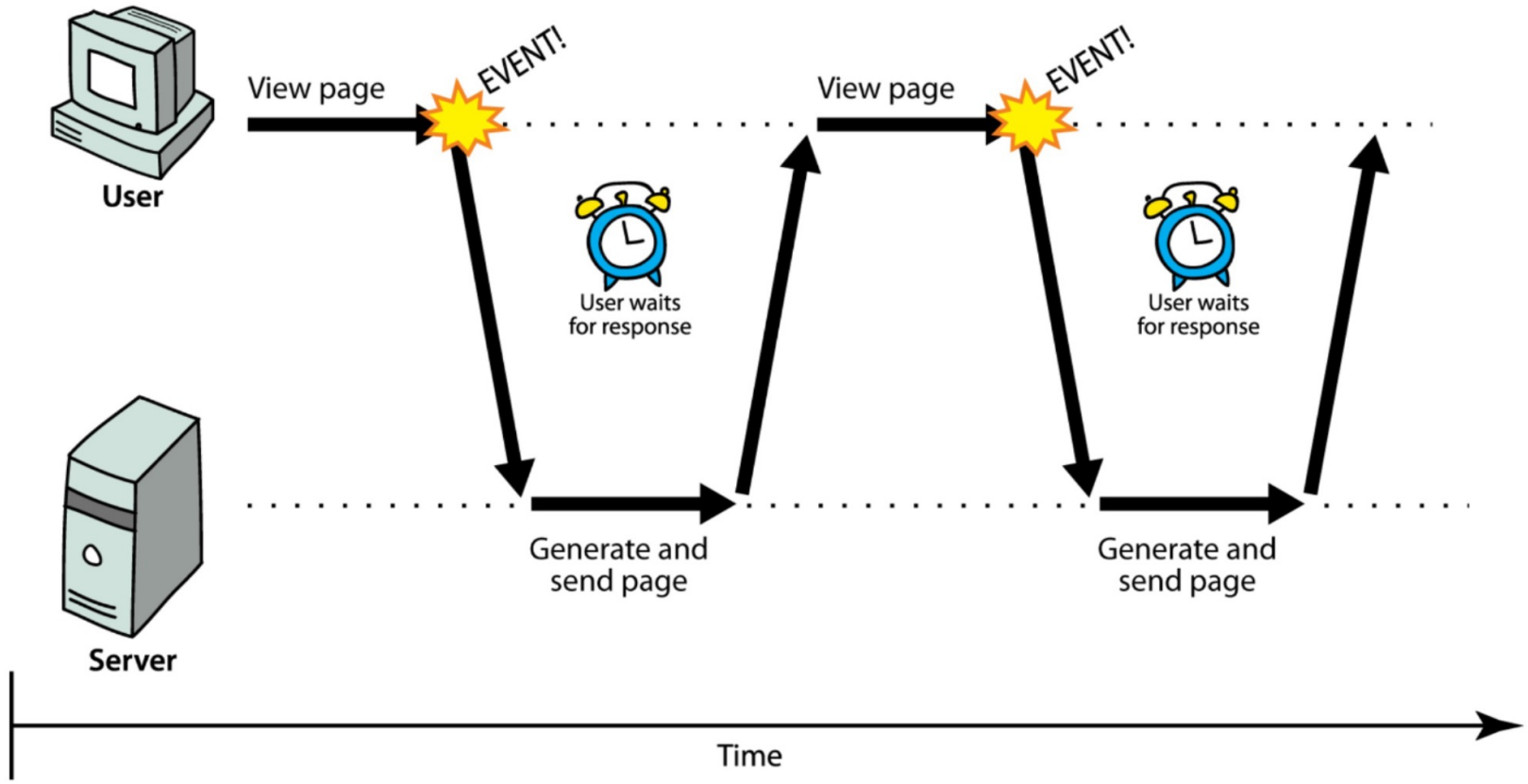
Cliente x Servidor



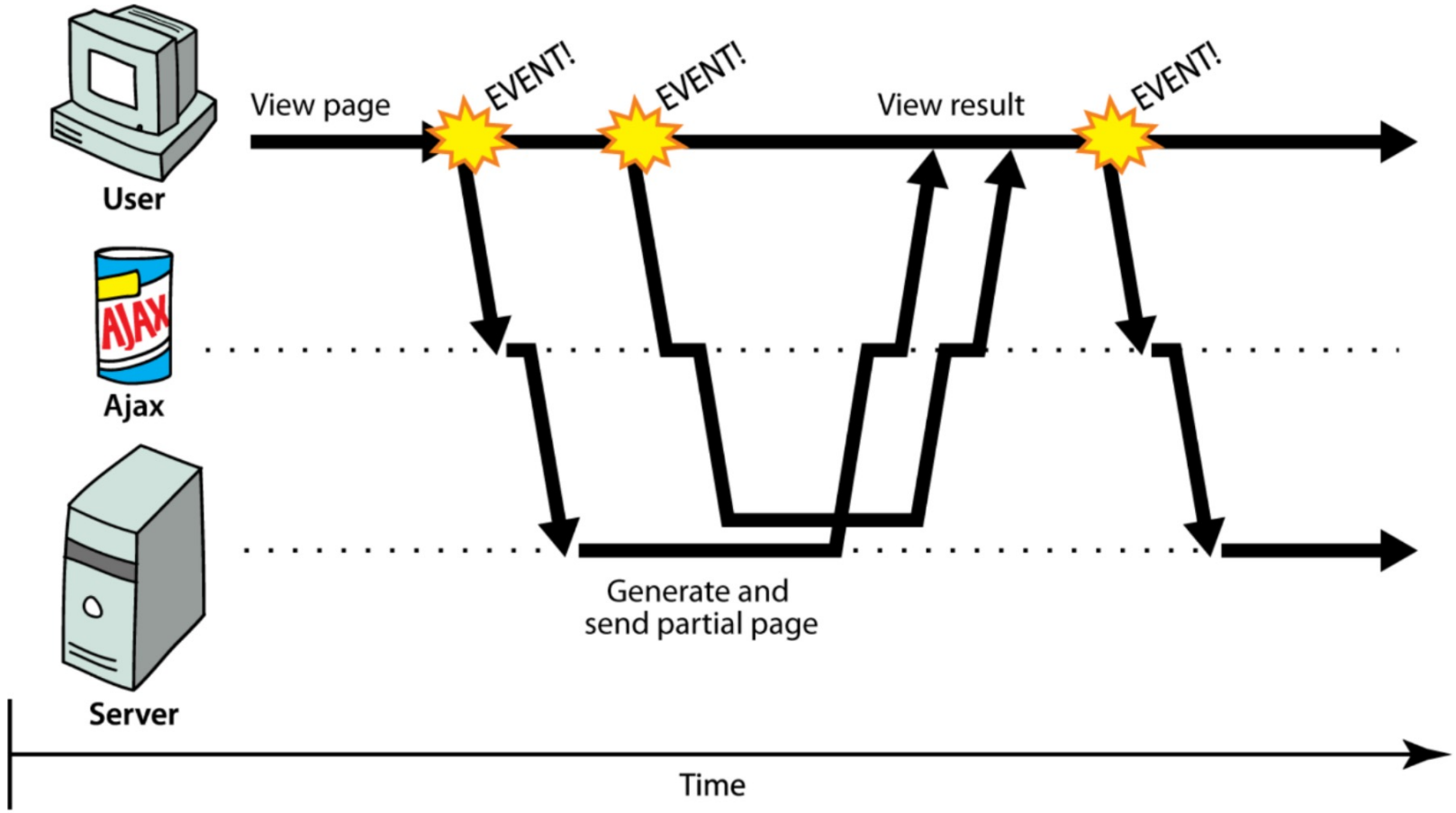
Cliente x Servidor



Requisição Síncrona



Requisição Assíncrona



API Fetch


function onClick(){
fetch(url, {
headers: {
"Content-Type": "application/jso
"Accept": "application/json, text-plain, */*"
JS
{fetch API}
method: 'post',
credentials: "same-origin",
body: JSON.stringify({
name: 'Tushar',
number: '78987'

O que é?

- A API Fetch fornece uma interface para buscar recursos para um servidor.

- O Fetch fornece uma definição genérica de objetos de Request e Response (e outras coisas envolvidas com solicitações de rede).

API Fetch

```
fetch("arquivo.php", {
    method: "POST",
    body: dados
});
```

Promisse

- Promisse é uma promessa de execução assíncrona, onde um objeto é usado para processamento assíncrono.

```
1  // Sintaxe
2
3  new Promise((resolve, reject) => { ... });
4
```

Async e Await

Async () => { Await }

Async Await

- Uma das **desvantagens** de utilizar funções call-back é lidar com o comportamento assíncrono sequencial.

Exercício 1

- Crie uma página de autenticação, onde será feita uma requisição dos dados do login para PHP.

- Deverá analisar os dados sendo enviados para o servidor no DevTool.

Exercício 2

- Crie uma página de autenticação, onde será feita uma requisição dos dados do login para PHP no servidor e verificar no banco de dados se o usuário existe.

- Se existir, deverá retornar uma mensagem em JSON para o JavaScript e mudar de página, caso contrário, avisar o usuário de que os dados de acesso não estão corretos.