

# Curso de JavaScript 2023



## *Módulo 5 - Aula 10: Node.js*

*- Fundamentos do Node.js: NPM, API REST e banco de dados*



Instrutor: *Prof. Me. Mário Carvalho*

E-mail para contato: [mario.carvalho@ufms.br](mailto:mario.carvalho@ufms.br)

Realização: *UFMS e Semadur*





# 1. Sobre o curso

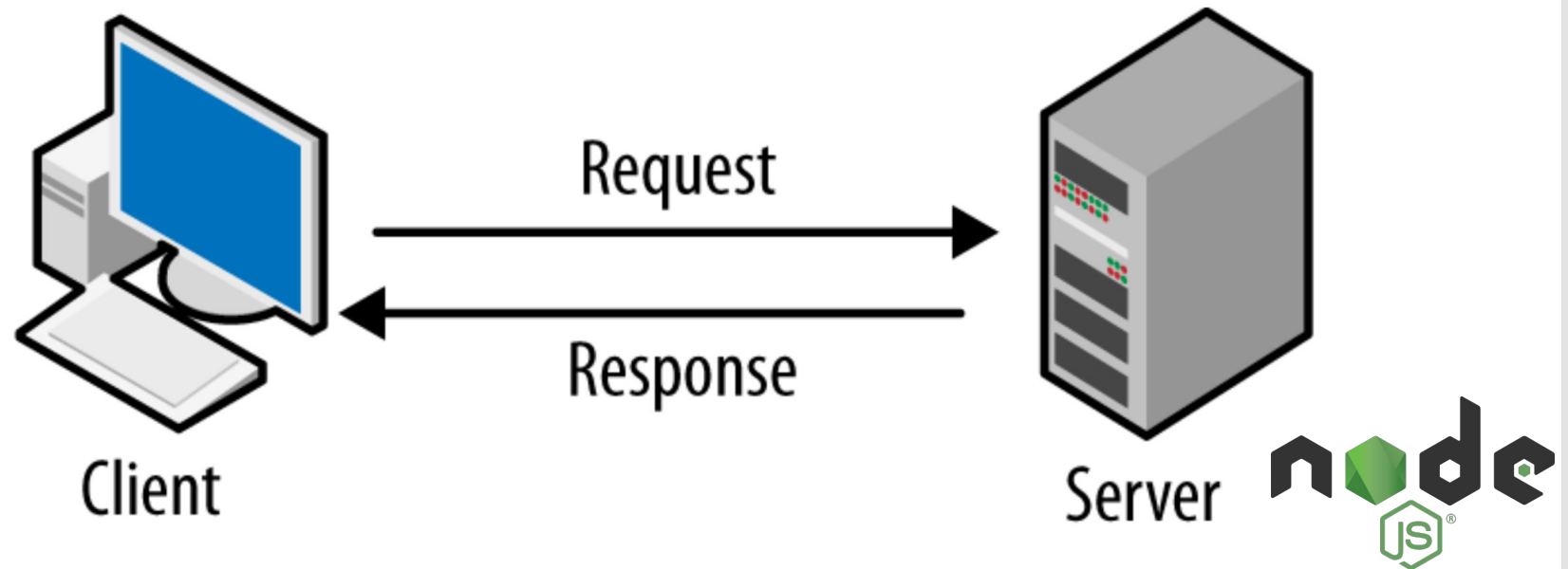
- Instrutor: **Prof. Me. Mário de Araújo Carvalho**
- E-mail para contato: [mario.carvalho@ufms.br](mailto:mario.carvalho@ufms.br)
- Sala Google Meet: <https://meet.google.com/fcq-djzs-dzd>
- Repositório oficial do curso:
- <https://github.com/MarioCarvalhoBr/semadur-curso-javascript-2023>
- **Ava:** <https://www.eadfapec.com.br/course/view.php?id=245&section=4>
- Instituição: **UFMS e Semadur**
- Modalidade: **Online síncrono**
- Duração: 45 horas

# Introdução

- O que é Node.js?

**Definição:** Um ambiente de execução JavaScript no lado do servidor.

**Utilização:** Desenvolvimento de aplicações web escaláveis.





# Introdução

- O que é NPM?

## O que é NPM?

Definição: Node Package Manager, ferramenta para gerenciar pacotes e dependências em projetos Node.js.

Site: <https://www.npmjs.com/>





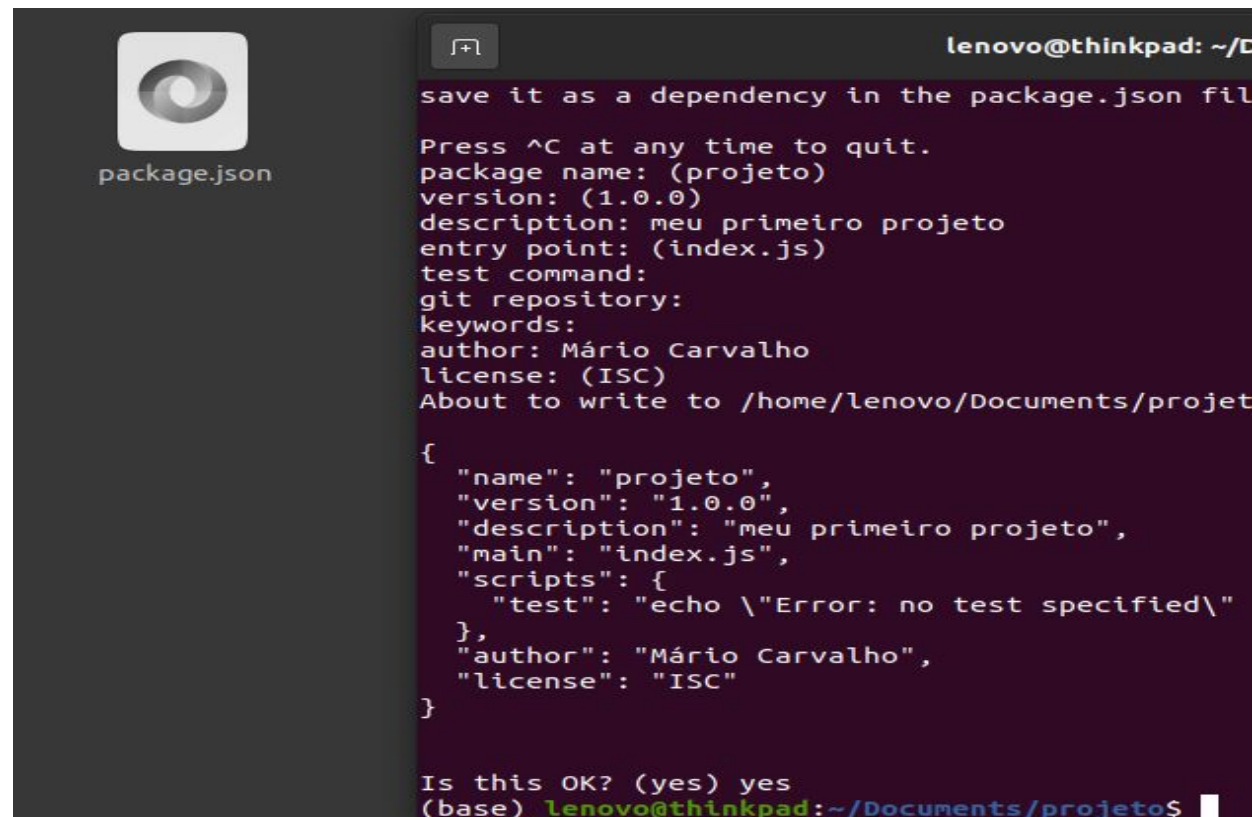
# NPM

- Criando um projeto

Criando um projeto: *mkdir projeto*

Entre na pasta: *cd projeto*

Inicializando o NPM: *npm init*



```
lenovo@thinkpad: ~/D
save it as a dependency in the package.json file

Press ^C at any time to quit.
package name: (projeto)
version: (1.0.0)
description: meu primeiro projeto
entry point: (index.js)
test command:
git repository:
keywords:
author: Mário Carvalho
license: (ISC)
About to write to /home/lenovo/Documents/projet
{
  "name": "projeto",
  "version": "1.0.0",
  "description": "meu primeiro projeto",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\""
  },
  "author": "Mário Carvalho",
  "license": "ISC"
}

Is this OK? (yes) yes
(base) lenovo@thinkpad:~/Documents/projeto$
```

# NPM

- Usando NPM

## Instalando Pacotes com NPM

Exemplo: Instalação do pacote Express.

Comando: *npm install express*

**package.json:** Arquivo que gerencia as dependências do projeto.

Exemplo Prático:

```
{  
  "name": "meu-projeto",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "^4.17.1"  
  }  
}
```





# API REST

- Criando uma API REST Simples com Express

## O que é uma API REST?

**Definição:** Uma interface que permite a interação com recursos web usando métodos HTTP.

**Importância:** Permite comunicação entre cliente e servidor de maneira padronizada.

# API REST

- Criando uma API REST Simples com Express

## Exemplo Prático: Servidor Express Básico

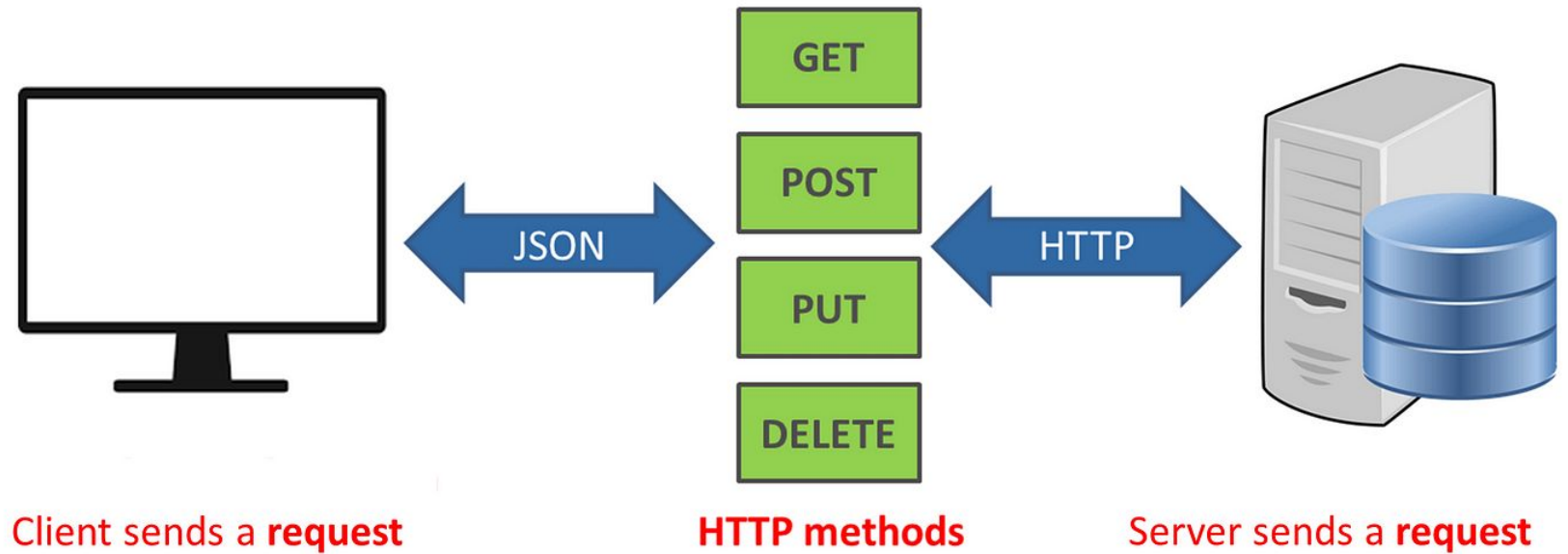
Crie um arquivo `server.js` com o seguinte código:

```
const express = require('express');
const app = express();
app.get('/', (req, res) => {
    res.send('Olá, API REST!');
});
app.listen(3000, () => console.log('Servidor rodando na porta 3000'));
```

**Execução:** No terminal, execute ***node server.js*** e acesse `localhost:3000` no navegador.

# API REST

- Fluxo API REST



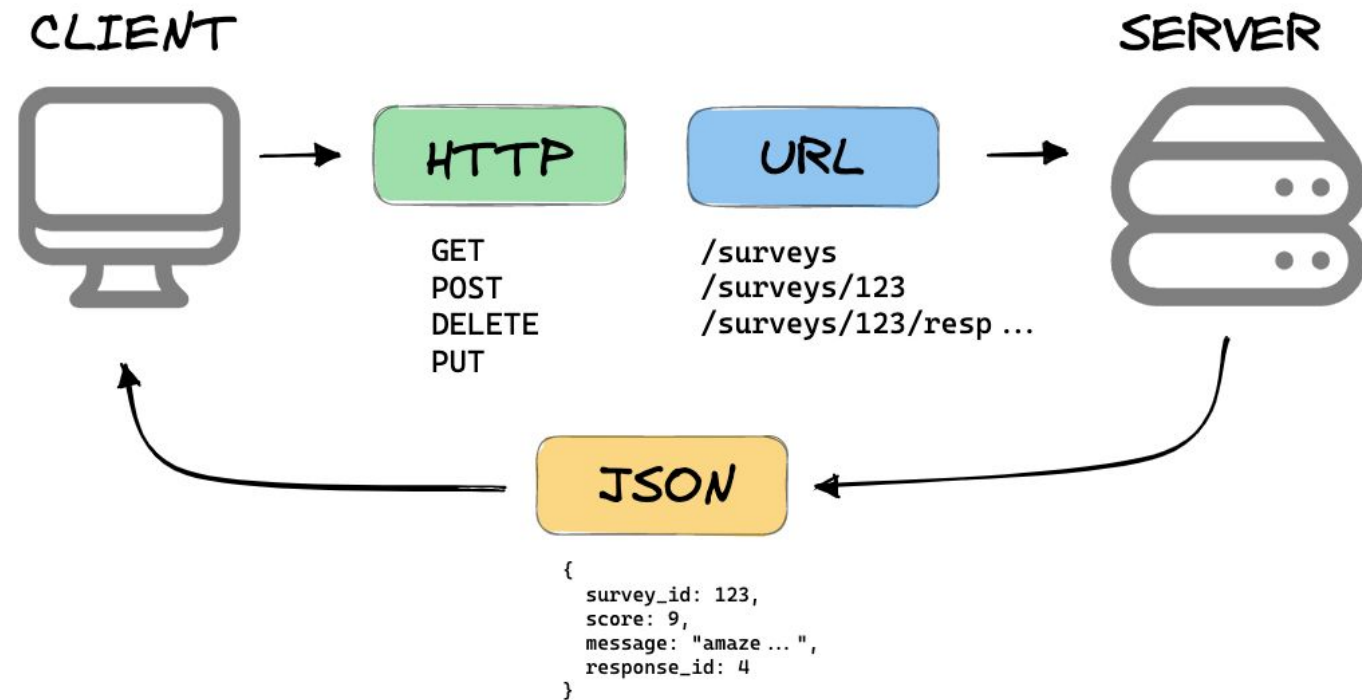
**Fonte:**

<https://medium.com/totvsdevelopers/rest-no-protheus-b%C3%A1sico-do-b%C3%A1sico-a4b4431a4e3e>

# API REST

- Fluxo API REST

## WHAT IS A REST API?





# API REST: Express

- Rotas em Express

## Manipulação de Rotas

**Definição:** Endereços que definem pontos de acesso à aplicação.

**Métodos HTTP:** *GET, POST, PUT, DELETE.*

## Exemplo Prático

```
app.get('/usuarios', (req, res) => {  
    res.send('Lista de usuários');  
});
```





# API REST: Express

- Middleware no Express

## O que são Middlewares?

**Definição:** Funções que têm acesso ao objeto de requisição (req), objeto de resposta (res) e ao próximo middleware.

**Utilização:** Para executar código, fazer modificações nos objetos de requisição e resposta, encerrar o ciclo de requisição-resposta ou chamar o próximo middleware.

**Exemplo Prático: Middleware de Log**

```
app.use((req, res, next) => {  
  console.log(`${req.method} ${req.url}`);  
  next();  
});
```



# Node e MySQL

- Conectando ao MySQL com Node.js

## Instalação e Configuração

**Instalar Pacote MySQL:** `npm install mysql`

**Configuração:** Criar uma conexão com o banco de dados.

## Exemplo Prático: Conexão com MySQL

```
const mysql = require('mysql');  
const conexao = mysql.createConnection({  
  host: 'localhost',  
  user: 'meu_usuario',  
  password: 'minha_senha',  
  database: 'meu_db'  
});
```

# Node e MySQL

- CRUD Básico com MySQL

## Criando Operações CRUD

**CRUD:** Create, Read, Update, Delete.

**Interação com o Banco:** Usando comandos SQL para manipular dados.

## Exemplo Prático: Inserindo dados

```
let sql = `INSERT INTO usuarios (nome, email) VALUES ('Ana', 'ana@email.com')`;
```

```
conexao.query(sql, (erro, resultados) => {  
    if (erro) throw erro;  
    console.log("1 registro inserido");  
});
```

# API REST: Express

- Boas Práticas no Desenvolvimento com Node.js

## Dicas para um Código Eficiente e Seguro

**Segurança:** Use bibliotecas para validar entradas e gerenciar autenticações.

**Manutenção:** Escreva código limpo e organizado.

# Node, API Rest e MySQL

- Exercícios Práticos

## Aplicando o Conhecimento em Node.js

- **API REST Simples:** Construa uma API REST com Express que responda a diferentes rotas.
- **Integração com MySQL:** Crie operações CRUD interagindo com um banco de dados MySQL.
- **Middleware Personalizado:** Implemente um middleware para log ou validação.



# Encerramento

- - Resumo dos tópicos abordados e orientações sobre recursos para continuar aprendendo JavaScript:
  - MDN Web Docs:  
<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>
- - Próximos passos e recursos adicionais para aprendizado autônomo:
  - HTML: <https://www.w3schools.com/html/default.asp>
  - JavaScript: <https://www.w3schools.com/js/default.asp>
  - Node.js: <https://www.w3schools.com/nodejs/default.asp>



Dúvidas?



Obrigado!

Mário Carvalho

[mario.carvalho@ufms.br](mailto:mario.carvalho@ufms.br)