Backend com Node.js – Aula 3

Por **Wanderson Guimarães**

O que vamos aprender hoje:

- Revisão da aula anterior
- Exercício de Fixação – Fs , Express, http
- Rodando local server com json-server
- Sobre o MVC
- Início do mini projeto MVC

Revisão da aula anterior

- Callbacks e events
- Axios
- Chamadas Crud (Create, Read, Update e Delete) no Axios
- Express

Exercício de Fixação

- Crie um arquivo index.html com um conteúdo simples.
- Crie o arquivo servidor.js.
- No servidor.js: - Use os módulos fs, http e express
- Use app.use() para capturar qualquer requisição.
- Leia o arquivo HTML com fs.readFile().
- Retorne o conteúdo HTML na resposta.
- O servidor deve rodar na porta 3000.

Rodando local server com json-server

*O json-server cria uma **API REST fake** a partir de um arquivo JSON.*

Ele roda um servidor local com endpoints completos (CRUD).

Json Server - Instalação

npm install axios json-server

Json Server - Criação do arquivo db

```
{
  "usuarios": [
    { "id": 1, "nome": "Maria" },
    { "id": 2, "nome": "João" }
  ]
}
```

Json Server - Rodando o Json Server

```
npx json-server --watch db.json --port 3001
```

Json Server - GET E POST

```
const axios = require('axios');

// Fazendo uma requisição GET

axios.get('http://localhost:3001/usuarios')
  .then(response => {
    console.log('Usuários:', response.data);
  })
  .catch(error => {
    console.error('Erro ao buscar usuários:', error);
  });

// Requisição POST

axios.post('http://localhost:3001/usuarios', {
  nome: 'Carla'
}).then(response => {
  console.log('Usuário adicionado:', response.data);
});
```

Json Server - PUT

```
const axios = require('axios');

axios.put('http://localhost:3001/usuarios/2',
{
  nome: 'João Silva'
})
.then(response => {
  console.log('Usuário atualizado:',
response.data);
})
.catch(error => {
  console.error('Erro ao atualizar usuário:',
error.message);
});
```

Json Server - DELETE

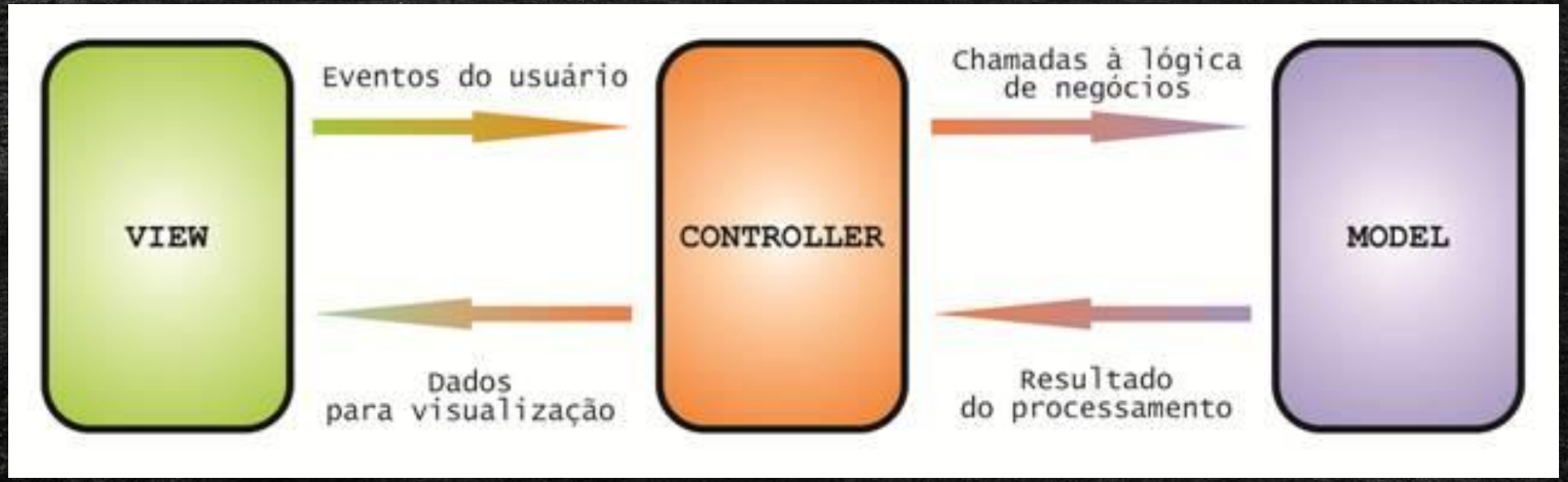
```
const axios = require('axios');

axios.delete('http://localhost:3001/usuarios/3')
  .then(response => {
    console.log('Usuário removido com sucesso!');
  })
  .catch(error => {
    console.error('Erro ao remover usuário:',
error.message);
  });
```

MVC (MODEL – VIEW – CONTROLLER)

- É um padrão de arquitetura de software que organiza uma aplicação em três componentes principais: o modelo, a visão e o controlador.

MVC (MODEL – VIEW – CONTROLLER)



MVC – ESTRUTURA - MODEL

Model

- Responsável pelos dados e regras de negócio
- Comunicação com banco de dados ou APIs externas

MVC – ESTRUTURA - VIEW

- Responsável por apresentar os dados
- Pode ser HTML, JSON, templates, etc.
- Interface com o usuário

MVC – FLUXO

O usuário faz uma requisição (ex: /posts)

O Controller recebe a requisição

O Controller chama o Model para obter/processar dados

O Controller envia os dados para a View

A View gera a resposta visual (HTML, JSON, etc.)

MVC – FLUXO

O usuário faz uma requisição (ex: /posts)

O Controller recebe a requisição

O Controller chama o Model para obter/processar dados

O Controller envia os dados para a View

A View gera a resposta visual (HTML, JSON, etc.)

MINI PROJETO MVC -instalação

```
mkdir meu-app && cd meu-app

npm init  –y  // cria o package.json

npm install express axios
```

MINI PROJETO MVC -instalação

Mãos a obra !!!