**APIs com API Externa Modificada e API Principal Atualizada**

Vamos modificar a API Externa e atualizar a API Principal para usar uma API externa real:

**1. API Externa (api\_externa.js):**

javascript

Copiar

const express = require('express');

const axios = require('axios');

const app = express();

const port = 3002; // Porta da API Externa

// Habilitando JSON para requisições

app.use(express.json());

// Rota para obter dados de uma API de previsão do tempo

app.get('/api/externa', async (req, res) => {

try {

// API de previsão do tempo OpenWeatherMap

const api\_key = 'your\_api\_key'; // Substitua pela sua chave API

const response = await axios.get(`https://api.openweathermap.org/data/2.5/weather?q=São Paulo&appid=${api\_key}`);

const dadosClima = response.data;

res.json(dadosClima);

} catch (error) {

console.error('Erro ao obter dados do clima:', error);

res.status(500).json({ error: 'Erro ao obter dados do clima' });

}

});

app.listen(port, () => {

console.log(`API Externa rodando em http://localhost:${port}`);

});

**2. API Principal (api\_principal.js):**

javascript

Copiar

const express = require('express');

const axios = require('axios');

const app = express();

const port = 3000;

// Habilitando JSON para requisições

app.use(express.json());

// Rota para obter dados da API Externa

app.get('/api/dados', async (req, res) => {

try {

const response = await axios.get('http://localhost:3002/api/externa'); // Acessando a API Externa

const dadosClima = response.data;

res.json({ mensagem: 'Dados do Clima:', dados: dadosClima });

} catch (error) {

console.error('Erro ao obter dados do clima:', error);

res.status(500).json({ error: 'Erro ao obter dados do clima' });

}

});

app.listen(port, () => {

console.log(`API Principal rodando em http://localhost:${port}`);

});

**Explicação:**

* **API Externa:**
  + Agora a API Externa usa a API OpenWeatherMap para obter dados de previsão do tempo.
  + Você precisa obter uma chave API gratuita da OpenWeatherMap ([https://openweathermap.org/api](https://openweathermap.org/api" \t "_blank)) e inserir na variável api\_key.
  + A API Externa faz uma requisição GET para a API OpenWeatherMap e retorna os dados recebidos.
* **API Principal:**
  + A API Principal continua a fazer uma requisição GET para a API Externa na rota /api/externa.
  + Ela recebe os dados do clima da API Externa e os envia para o cliente.

**Como Executar:**

1. **Instale o Express e o Axios:**

bash

Copiar

npm install express axios

1. **Crie dois arquivos:**

api\_principal.js **e** api\_externa.js.

1. **Cole o código da API Principal em api\_principal.js e da API Externa em api\_externa.js.**
2. **Execute as APIs:**

bash

Copiar

node api\_principal.js

node api\_externa.js

**Teste:**

* Abra seu navegador e acesse http://localhost:3000/api/dados. Você verá a resposta da API Principal, que inclui os dados da API Externa.

**Observações:**

* **Portabilidade:** Este exemplo usa localhost para as URLs. Em um ambiente de produção, você provavelmente precisará usar URLs diferentes.
* **Segurança:** Em uma aplicação real, você precisaria implementar mecanismos de segurança como autenticação e autorização para proteger suas APIs.
* **Complexidade:** Este é um exemplo simples. Em um sistema real, as APIs podem ter várias rotas, lidar com diferentes métodos HTTP (POST, PUT, DELETE), e podem ser muito mais complexas.

Lembre-se que este é apenas um exemplo básico! Você pode adaptar este código para criar APIs mais complexas com diferentes funcionalidades e interações.

**Usando o pm2 para Gerenciar suas APIs**

Vamos usar o pm2 para gerenciar as duas APIs e evitar conflitos de porta.

**1. Instalar o pm2:**

bash

Copiar

npm install -g pm2

**2. Criar o arquivo ecosystem.config.js:**

Crie um arquivo chamado ecosystem.config.js na raiz do seu projeto com o seguinte conteúdo:

javascript

Copiar

module.exports = {

apps: [

{

name: 'api\_principal',

script: 'api\_principal.js',

instances: 1,

autorestart: true,

watch: false,

max\_memory\_restart: '1G',

env: {

NODE\_ENV: 'development'

},

env\_production: {

NODE\_ENV: 'production'

}

},

{

name: 'api\_externa',

script: 'api\_externa.js',

instances: 1,

autorestart: true,

watch: false,

max\_memory\_restart: '1G',

env: {

NODE\_ENV: 'development'

},

env\_production: {

NODE\_ENV: 'production'

}

}

]

};

**3. Iniciar as APIs com pm2:**

bash

Copiar

pm2 start ecosystem.config.js

**Explicação:**

* **ecosystem.config.js:** Define as configurações para cada API.
  + name: Nome da aplicação (usado para identificação).
  + script: Arquivo da API.
  + instances: Número de instâncias da API (1 para cada API).
  + autorestart: Reinicia automaticamente a API em caso de falha.
  + watch: Monitora alterações no arquivo da API.
  + max\_memory\_restart: Reinicia a API se o uso de memória exceder o limite.
  + env: Variáveis de ambiente para o desenvolvimento.
  + env\_production: Variáveis de ambiente para produção.
* **pm2 start ecosystem.config.js:** Inicia as APIs definidas no arquivo ecosystem.config.js.

**Benefícios do pm2:**

* **Gerenciamento de Processos:** O pm2 controla os processos Node.js de forma eficiente, monitorando sua saúde e reiniciando-os automaticamente se necessário.
* **Escalabilidade:** Permite iniciar várias instâncias de uma API para aumentar a capacidade de processamento.
* **Logs:** Facilita a coleta e visualização de logs das APIs.
* **Monitoramento:** Permite monitorar o uso de CPU, memória e outras métricas das APIs.

**Após executar os comandos acima, suas APIs (API Principal e API Externa) devem estar em execução sem erros de porta. Você pode verificar o status das APIs com o comando pm2 list.**

**Outras Funcionalidades do pm2:**

* **pm2 stop <name>:** Para parar uma API.
* **pm2 restart <name>:** Para reiniciar uma API.
* **pm2 delete <name>:** Para remover uma API.
* **pm2 logs <name>:** Para visualizar os logs de uma API.

O pm2 é uma ferramenta poderosa para gerenciar suas aplicações Node.js. Experimente-o e veja os benefícios que ele oferece! 😊

**Opção 3: Usar um gerenciador de processos como o forever**

1. **Instale o forever:**

bash

Copiar

npm install -g forever

1. **Inicie as APIs com o forever:**

bash

Copiar

forever start api\_principal.js

forever start api\_externa.js