🎯 EXERCÍCIO: API com Middleware de Autenticação

📋 Objetivo

Criar uma API usando Express que implemente middlewares globais e específicos, com sistema de autenticação para rotas privadas.

⚙️ Requisitos

1. Configuração Base

- Crie uma API usando Express
- Configure o servidor para rodar na porta 3000
- Implemente parsing de JSON

2. Middleware Global

- Crie um middleware global que registre no console a mensagem: **"Chamando API"**
- Este middleware deve executar em **todas as rotas**

3. Rota Pública

- **Rota:** `GET /usuarios`
- **Funcionalidade:** Retornar lista de usuários
- **Acesso:** Público (sem autenticação)

4. Rota Privada

- **Rota:** `POST /usuarios`
- **Funcionalidade:** Criar novo usuário
- **Acesso:** Apenas usuários autorizados

5. Sistema de Autenticação

5.1 Arquivo separado

- Crie um arquivo chamado `auth.js`
- Implemente um middleware de verificação de autorização

5.2 Regras de autorização

- A informação do tipo de usuário deve vir no campo `tipoUsuario` no **body da requisição**
- **Usuários autorizados:** Apenas quando `tipoUsuario === "ADM"`

5.3 Respostas do middleware

- **Usuário não autorizado:**
  - Status: `401`
  - Mensagem: `"Funcao nao permitida para esse usuario"`
- **Usuário autorizado:**
  - Permitir acesso à rota
  - Rota deve retornar:
    - Status: `201`
    - Mensagem: `"Rota permitida"`

6. Aplicação do Middleware

- ⚠️ **IMPORTANTE:** O middleware de autenticação deve ser aplicado **APENAS** na rota `POST /usuarios`
- **NÃO** deve ser um middleware global

🗂️ Estrutura de Arquivos Esperada

```
projeto/
├── index.js          # Arquivo principal com servidor e rotas
├── auth.js           # Middleware de autenticação
└── package.json      # Dependências do projeto
```

🧪 Casos de Teste

Utilize a collection do Postman fornecida para verificar os seguintes cenários:

Teste 1: Rota pública

```http
GET /usuarios
```

Resultado esperado: Acesso liberado, retorna dados

Teste 2: Usuário autorizado

```http
```

```
POST /usuarios
{
  "tipoUsuario": "ADM"
}
```

Resultado esperado: Status 201 + "Rota permitida"

Teste 3: Usuário não autorizado

```http
POST /usuarios
{
  "tipoUsuario": "COMUM"
}
```

Resultado esperado: Status 401 + "Funcao nao permitida para esse usuario"

Teste 4: Campo ausente

```http
POST /usuarios
{
  "nome": "João"
}
```

Resultado esperado: Status 401 + "Funcao nao permitida para esse usuario"

✅ Critérios de Avaliação

- [ ] API funcionando corretamente na porta 3000
- [ ] Middleware global registrando "Chamando API" no console
- [ ] Rota `GET /usuarios` acessível publicamente
- [ ] Arquivo `auth.js` criado com middleware de autenticação
- [ ] Rota `POST /usuarios` protegida pelo middleware
- [ ] Validação correta do campo `tipoUsuario`
- [ ] Status codes e mensagens conforme especificação
- [ ] Middleware aplicado apenas na rota específica (não globalmente)

💡 Dicas

1. **Ordem importa:** Middlewares devem ser declarados antes das rotas que os utilizam

2. **Separação de responsabilidades:** Mantenha o middleware de autenticação em arquivo separado
3. **Teste gradualmente:** Implemente e teste uma funcionalidade por vez
4. **Use a collection:** Ela contém todos os casos de teste necessários

---

Boa sorte! 🚀