Criar uma API em Node.js de Autenticação

Para criar uma API de autenticação em Node.js, você precisará implementar um sistema que valide as credenciais do usuário e gere tokens de acesso para autorização. Aqui está um guia passo a passo com um exemplo prático:

1. Instalar as bibliotecas necessárias:

```
npm install express jsonwebtoken bcryptjs
```

content_copyUse code with caution.Bash

- express: Framework web para Node.js.
- jsonwebtoken: Biblioteca para gerar e verificar tokens JWT (JSON Web Token).
- bcryptjs: Biblioteca para criptografia de senhas (hashing).

2. Criar um servidor Express básico:

```
const express = require('express');
const app = express();
const port = 3000;

app.listen(port, () => {
  console.log(`Servidor ouvindo na porta ${port}`);
});
```

content_copyUse code with caution.JavaScript

3. Definir um modelo de usuário:

```
const users = [
  {
    id: 1,
    username: 'usuario1',
    password: '$2b$10$x96253M92H8T1Vq4kD2H.u7n6.G4cK0B.m.a7B4uQ02', // Senha
criptografada
  },
  {
    id: 2,
    username: 'usuario2',
    password: '$2b$10$5lC0/W7c4c884g7/37e06u4WjT4iJ5s7O26x/y1Q4S6', // Senha
criptografada
  },
];
```

content_copyUse code with caution.JavaScript

- Utilize bcryptjs.hash(senha, sal) para criptografar senhas antes de armazená-las.
- Defina um salt (sal) forte para aumentar a segurança.

4. Criar o endpoint de registro (POST /register):

```
app.post('/register', async (req, res) => {
  const { username, password } = req.body;

  // Validação de dados (ex: username e senha não podem ser vazios)
```

```
  if (!username || !password) {
    return res.status(400).json({ error: 'Nome de usuário e senha são
obrigatórios' });
  }

  // Verifica se o usuário já existe
  const existingUser = users.find(user => user.username === username);
  if (existingUser) {
    return res.status(400).json({ error: 'Usuário já existe' });
  }
```

content_copyUse code with caution.JavaScript