



Nome: Wallace Felipe Tavares Moreira

Matrícula: 202109237331

Universidade: UNIVERSIDADE ESTÁCIO DE SÁ

Curso: Desenvolvimento Full Stack

Campus: Jardim América – Itaguaí/RJ

Disciplina: Nível 5 – **RPG0035 - SOFTWARE SEM SEGURANÇA NÃO SERVE!**

Semestre Letivo: Quinto Semestre

Documentação do Aplicativo Node.js

Introdução

Este documento descreve o funcionamento e os endpoints da API RESTful desenvolvida utilizando Node.js, Express, Sequelize, e autenticação JWT para garantir segurança e controle de acesso adequado. O aplicativo gerencia usuários, empresas e contratos através de requisições HTTP.

Tecnologias Utilizadas

- **Node.js:** Ambiente de execução JavaScript.
- **Express:** Framework web para Node.js.
- **Sequelize:** ORM para Node.js, utilizado com SQLite para gerenciamento de banco de dados.
- **JWT (JSON Web Token):** Mecanismo de autenticação seguro e stateless.

- **SQLite:** Banco de dados relacional utilizado para armazenamento de dados.

Pré-requisitos

- Node.js instalado na máquina local.
- NPM (Node Package Manager) para instalação de dependências.
- Conhecimento básico em RESTful APIs e JavaScript.

Instalação e Configuração

- Clonar o Repositório
- Instalar Dependências / `npm install`

Configuração do Banco de Dados

- O arquivo `database.sqlite` será criado automaticamente na raiz do projeto ao iniciar o servidor, conforme definido em `sequelize` na configuração.

Configuração do Ambiente

- Copie `.env.example` para `.env` e configure as variáveis de ambiente necessárias, se aplicável.

Inicialização do Servidor

Para iniciar o servidor, execute o seguinte comando:

- `npm start`

- O servidor será iniciado na porta especificada (padrão 3000) e estará pronto para receber requisições.

Endpoints da API

Autenticação

Login de Usuário

- **POST /api/auth/login**
 - Realiza o login de um usuário com as credenciais fornecidas.
 - Corpo da Requisição

```
{  
  "username": "string",  
  "password": "string"  
}
```

Retorna um token JWT válido por 1 hora se as credenciais forem válidas.

Usuários

Listar Todos os Usuários

- **GET /api/users**
 - Retorna todos os usuários cadastrados.
 - Requer autenticação com token JWT de perfil admin.

Criar Novo Usuário

- **POST /api/users/create**
 - Cria um novo usuário com os dados fornecidos.
 - Corpo da Requisição:

```
{  
    "username": "string",  
    "password": "string",  
    "perfil": "string",  
    "email": "string"  
}
```

Retorna o usuário criado

Empresas

Listar Todas as Empresas

- **GET /api/companies**
 - Retorna todas as empresas cadastradas.

Contratos

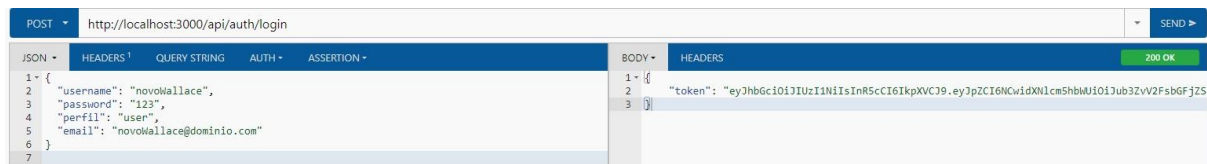
Listar Contratos por Empresa e Data de Início

- **GET /api/contracts/:companyId/:inicio**

- Retorna contratos associados a uma empresa específica e com data de início especificada.
- Parâmetros de Rota:
 - `companyId`: ID da empresa.
 - `inicio`: Data de início no formato YYYY-MM-DD.

Execução:

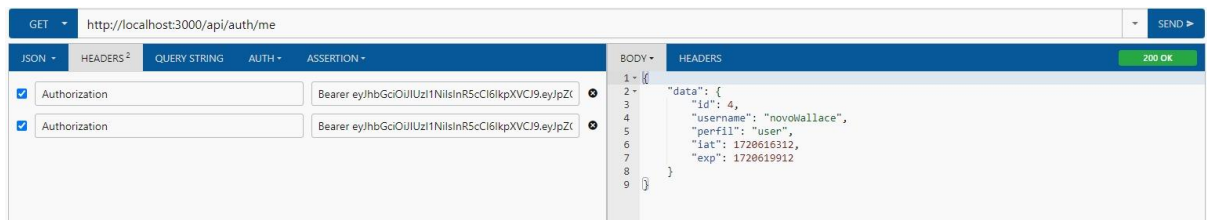
<http://localhost:3000/api/auth/login>



Endpoint para Dados do Usuário Logado:

Método: GET

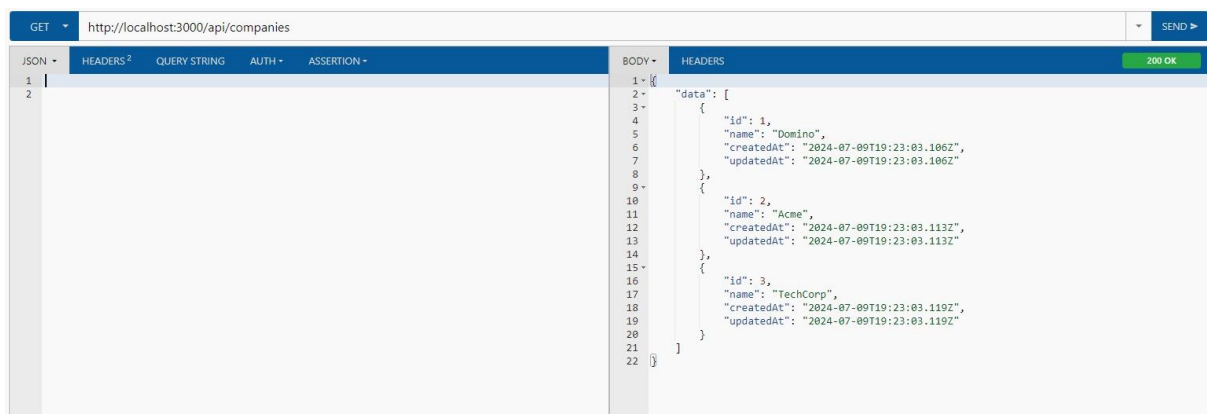
URL: <http://localhost:3000/api/auth/me>



Endpoint para Listar Todas as Empresas Cadastradas:

Método: GET

URL: <http://localhost:3000/api/companies>



Método: GET

The screenshot shows a REST client interface. The top bar displays the method 'GET' and the URL 'http://localhost:3000/api/contracts/2/2024-07-02'. Below this, the 'HEADERS' tab is selected, showing an empty list. The 'BODY' tab is also visible, showing a JSON response. The response is a JSON object with a 'data' array containing one contract object. The contract object has the following fields: 'id' (2), 'data_inicio' ('2024-07-02T00:00:00.000Z'), 'created_at' ('2024-07-09T19:23:03.134Z'), 'updated_at' ('2024-07-09T19:23:03.134Z'), and 'company_id' (2).

```
{
  "data": [
    {
      "id": 2,
      "data_inicio": "2024-07-02T00:00:00.000Z",
      "created_at": "2024-07-09T19:23:03.134Z",
      "updated_at": "2024-07-09T19:23:03.134Z",
      "company_id": 2
    }
  ]
}
```

Método: POST

The screenshot shows a REST client interface with a POST request to `http://localhost:3000/api/users/create`. The response is a JSON object with the following structure:

```
{  "username": "Wallace Tavares",  "password": "123",  "perfil": "user",  "email": "wfelipetm@dominio.com"}
```

The response also includes a status bar indicating "201 Created".

Endpoint para recuperar todos os usuários:

Método: GET

<http://localhost:3000/api/users>

GEThttp://localhost:3000/api/usersSEND

JSONHEADERS³QUERY STRINGAUTH⁺ASSERTION⁺

12345
{
 "username": "admin",
 "password": "123456789"
}

BODYHEADERS200 OK

12345678910111213141516171819202122232425262728293031323334353637383940414243444546474849
{"data": [
 {
 "id": 1,
 "username": "admin",
 "password": "123456789",
 "perfil": "admin",
 "email": "admin@dominio.com",
 "createdAt": "2024-07-09T19:23:03.076Z",
 "updatedAt": "2024-07-09T19:23:03.076Z"
 },
 {
 "id": 2,
 "username": "user",
 "password": "123456",
 "perfil": "user",
 "email": "user@dominio.com",
 "createdAt": "2024-07-09T19:23:03.090Z",
 "updatedAt": "2024-07-09T19:23:03.090Z"
 },
 {
 "id": 3,
 "username": "wallace",
 "password": "123",
 "perfil": "user",
 "email": "colab@dominio.com",
 "createdAt": "2024-07-09T19:23:03.098Z",
 "updatedAt": "2024-07-09T19:23:03.098Z"
 },
 {
 "id": 4,
 "username": "novowallace",
 "password": "123",
 "perfil": "user",
 "email": "novowallace@dominio.com",
 "createdAt": "2024-07-09T19:24:29.652Z",
 "updatedAt": "2024-07-09T19:24:29.652Z"
 },
 {
 "id": 5,
 "username": "Wallace Tavares",
 "password": "123",
 "perfil": "user",
 "email": "wfelipetn@dominio.com",
 "createdAt": "2024-07-10T13:15:27.125Z",
 "updatedAt": "2024-07-10T13:15:27.125Z"
 }
]

Considerações de Segurança

- **Autenticação JWT:** Utilizado para autenticar e autorizar usuários através de tokens seguros.
- **ORM Sequelize:** Evita vulnerabilidades de injeção de SQL ao manipular consultas ao banco de dados.
- **Validação de Entrada:** Assegure-se de sempre validar e sanitizar os dados de entrada para evitar ataques de injeção e outras vulnerabilidades.

Conclusão

Este documento fornece uma visão geral do aplicativo Node.js refatorado, destacando as melhorias de segurança e os endpoints da API RESTful. Certifique-se de manter a aplicação e o banco de dados atualizados e de seguir práticas recomendadas de segurança.

