**GK ACESSIBILIDADE APPLICATION SWAGGER**

**Documento Técnico: Especificação da API**

**Visão Geral**

**Este documento técnico detalha a especificação dos endpoints da API para os recursos Empresa, Segurança, Usuário, e Vaga. A API foi projetada para gerenciar operações relacionadas a empresas, autenticação, gerenciamento de usuários e vagas de emprego. Também inclui a definição dos Schemas utilizados para requisições e respostas.**

1. Empresa

1.1. Listar Empresas

* Método: GET
* Endpoint: /api/Empresa/Listar
* Descrição: Retorna uma lista de todas as empresas cadastradas no sistema.
* Autenticação: Sim (se necessário).
* Parâmetros: Nenhum.
* Resposta:
  + Código 200: Retorna a lista de empresas.
  + Código 401: Não autorizado.

1.2. Cadastrar Empresa

* Método: POST
* Endpoint: /api/Empresa/Cadastrar
* Descrição: Cadastra uma nova empresa no sistema.
* Autenticação: Sim (se necessário).
* Parâmetros:
  + Corpo da Requisição: [EmpresaRequest](#schemas) - Contém as informações da empresa a ser cadastrada.
* Resposta:
  + Código 201: Empresa cadastrada com sucesso.
  + Código 400: Requisição inválida.
  + Código 401: Não autorizado.

1.3. Excluir Empresa

* Método: POST
* Endpoint: /api/Empresa/Excluir
* Descrição: Exclui uma empresa do sistema.
* Autenticação: Sim (se necessário).
* Parâmetros:
  + Corpo da Requisição: [Empresa](#schemas) - Contém o identificador da empresa a ser excluída.
* Resposta:
  + Código 200: Empresa excluída com sucesso.
  + Código 400: Requisição inválida.
  + Código 401: Não autorizado.

2. Segurança

2.1. Login

* Método: POST
* Endpoint: /api/Seguranca/Login
* Descrição: Realiza a autenticação do usuário no sistema.
* Autenticação: Não (usado para obter token de autenticação).
* Parâmetros:
  + Corpo da Requisição: [Login](#schemas) - Contém as credenciais de login (nome de usuário e senha).
* Resposta:
  + Código 200: Login bem-sucedido, retorna token de autenticação.
  + Código 400: Requisição inválida.
  + Código 401: Credenciais inválidas.

3. Usuário

3.1. Listar Usuários

* Método: GET
* Endpoint: /api/Usuario/Listar
* Descrição: Retorna uma lista de todos os usuários cadastrados.
* Autenticação: Sim.
* Parâmetros: Nenhum.
* Resposta:
  + Código 200: Retorna a lista de usuários.
  + Código 401: Não autorizado.

3.2. Cadastrar Usuário

* Método: POST
* Endpoint: /api/Usuario/Cadastrar
* Descrição: Cadastra um novo usuário no sistema.
* Autenticação: Sim.
* Parâmetros:
  + Corpo da Requisição: [CadastrarUsuarioRequest](#schemas) - Contém as informações do usuário a ser cadastrado.
* Resposta:
  + Código 201: Usuário cadastrado com sucesso.
  + Código 400: Requisição inválida.
  + Código 401: Não autorizado.

3.3. Atualizar Usuário

* Método: PUT
* Endpoint: /api/Usuario/Atualizar
* Descrição: Atualiza as informações de um usuário existente.
* Autenticação: Sim.
* Parâmetros:
  + Corpo da Requisição: [AlterarUsuarioRequest](#schemas) - Contém as informações atualizadas do usuário.
* Resposta:
  + Código 200: Usuário atualizado com sucesso.
  + Código 400: Requisição inválida.
  + Código 401: Não autorizado.

3.4. Excluir Usuário

* Método: DELETE
* Endpoint: /api/Usuario/Excluir
* Descrição: Exclui um usuário do sistema.
* Autenticação: Sim.
* Parâmetros:
  + Corpo da Requisição: [Usuário](#schemas) - Contém o identificador do usuário a ser excluído.
* Resposta:
  + Código 200: Usuário excluído com sucesso.
  + Código 400: Requisição inválida.
  + Código 401: Não autorizado.

3.5. Dados Complementares de Usuário

* Método: POST
* Endpoint: /api/Usuario/DadosComplementares
* Descrição: Adiciona ou atualiza dados complementares para um usuário.
* Autenticação: Sim.
* Parâmetros:
  + Corpo da Requisição: [DadosComplementaresUsuarioRequest](#schemas) - Contém os dados complementares a serem atualizados.
* Resposta:
  + Código 200: Dados complementares atualizados com sucesso.
  + Código 400: Requisição inválida.
  + Código 401: Não autorizado.

3.6. Listar Candidatos

* Método: GET
* Endpoint: /api/Usuario/ListarCandidatos
* Descrição: Retorna uma lista de todos os candidatos cadastrados.
* Autenticação: Sim.
* Parâmetros: Nenhum.
* Resposta:
  + Código 200: Retorna a lista de candidatos.
  + Código 401: Não autorizado.

4. Vaga

4.1. Listar Vagas

* Método: GET
* Endpoint: /api/Vaga/Listar
* Descrição: Retorna uma lista de todas as vagas cadastradas.
* Autenticação: Sim.
* Parâmetros: Nenhum.
* Resposta:
  + Código 200: Retorna a lista de vagas.
  + Código 401: Não autorizado.

4.2. Cadastrar Vaga

* Método: POST
* Endpoint: /api/Vaga/Cadastrar
* Descrição: Cadastra uma nova vaga no sistema.
* Autenticação: Sim.
* Parâmetros:
  + Corpo da Requisição: [VagaRequest](#schemas) - Contém as informações da vaga a ser cadastrada.
* Resposta:
  + Código 201: Vaga cadastrada com sucesso.
  + Código 400: Requisição inválida.
  + Código 401: Não autorizado.

4.3. Excluir Vaga

* Método: POST
* Endpoint: /api/Vaga/Excluir
* Descrição: Exclui uma vaga do sistema.
* Autenticação: Sim.
* Parâmetros:
  + Corpo da Requisição: [Vaga](#schemas) - Contém o identificador da vaga a ser excluída.
* Resposta:
  + Código 200: Vaga excluída com sucesso.
  + Código 400: Requisição inválida.
  + Código 401: Não autorizado.

5. Schemas

5.1. AlterarUsuarioRequest

* Descrição: Estrutura utilizada para atualizar as informações de um usuário existente.
* Campos:
  + id: Identificador do usuário.
  + nome: Nome completo do usuário.
  + email: Endereço de e-mail do usuário.
  + telefone: Número de telefone do usuário.
  + senha: Nova senha do usuário (opcional).

5.2. CadastrarUsuarioRequest

* Descrição: Estrutura utilizada para cadastrar um novo usuário.
* Campos:
  + nome: Nome completo do usuário.
  + email: Endereço de e-mail do usuário.
  + telefone: Número de telefone do usuário.
  + senha: Senha do usuário.

5.3. DadosComplementaresUsuario

* Descrição: Estrutura que contém dados adicionais de um usuário.
* Campos:
  + endereco: Endereço do usuário.
  + dataNascimento: Data de nascimento do usuário.
  + tipoDeficiencia: Tipo de deficiência, se aplicável.

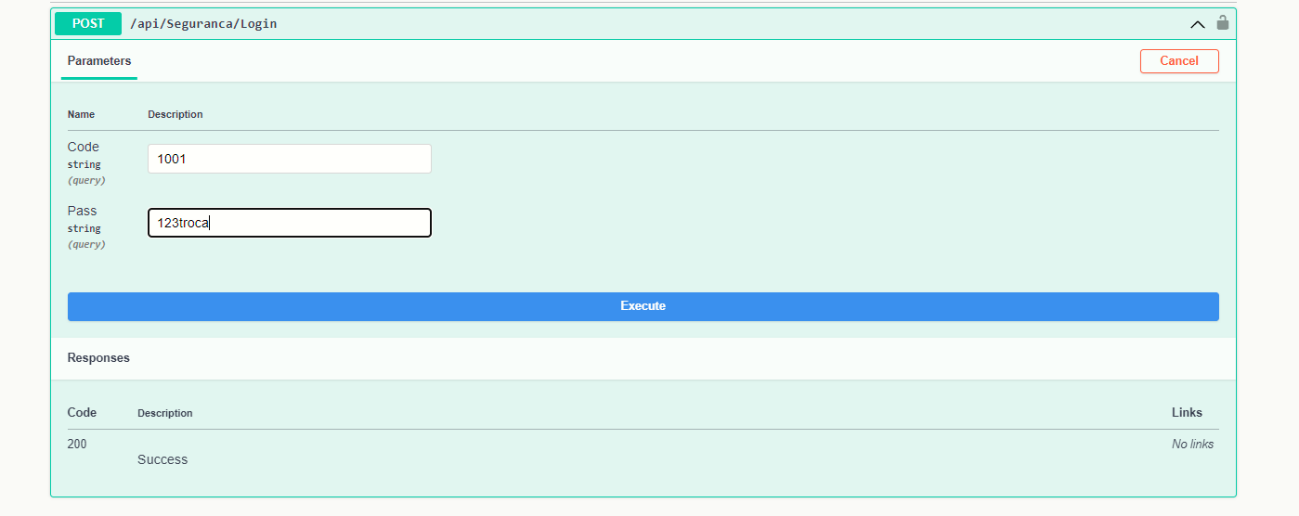
5.4. DadosComplementaresUsuarioRequest

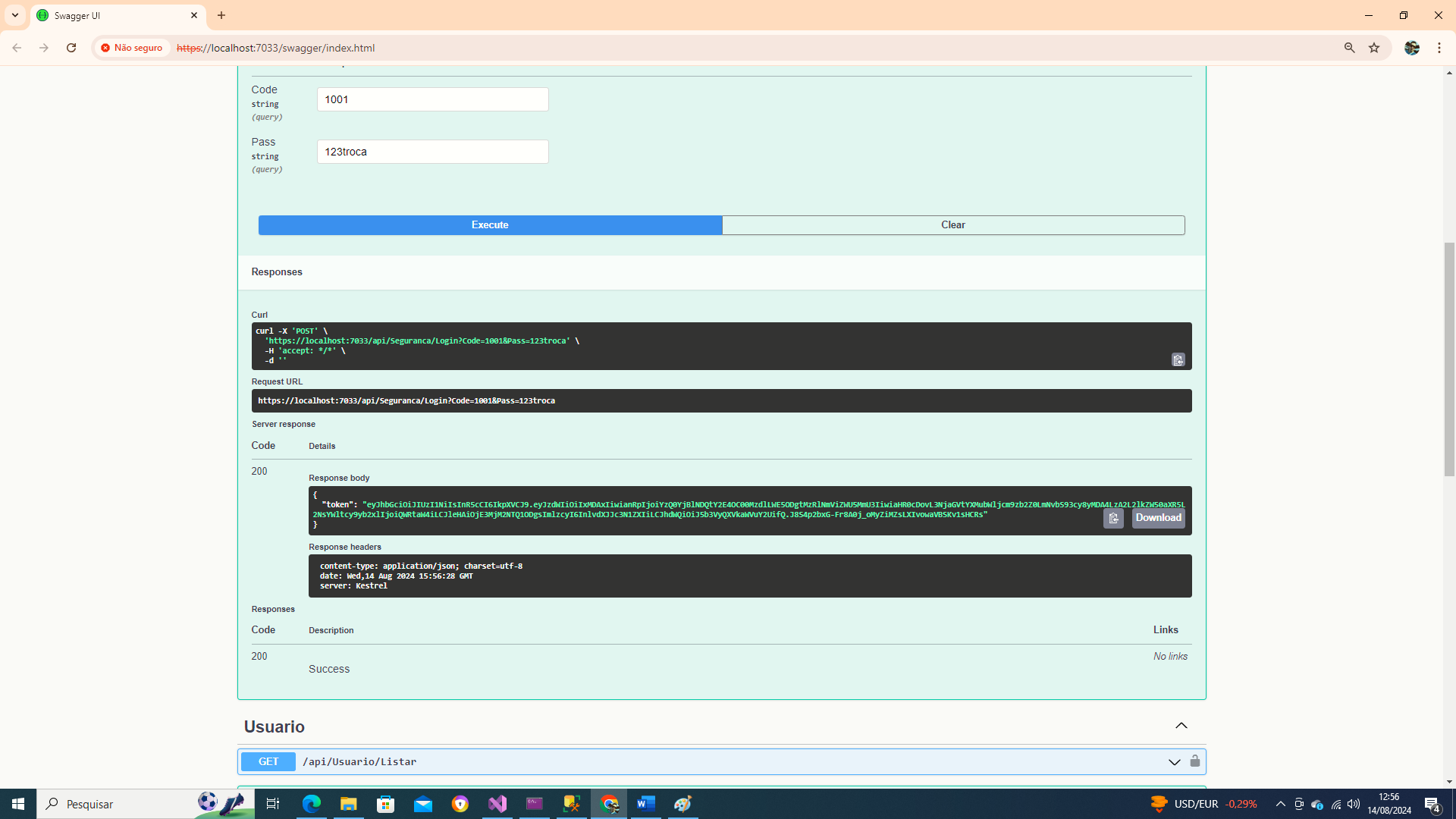
* Descrição: Estrutura utilizada para adicionar ou atualizar os dados complementares de um usuário.
* Campos:
  + idUsuario: Identificador do usuário.
  + dadosComplementares: Objeto [DadosComplementaresUsuario](#schemas).

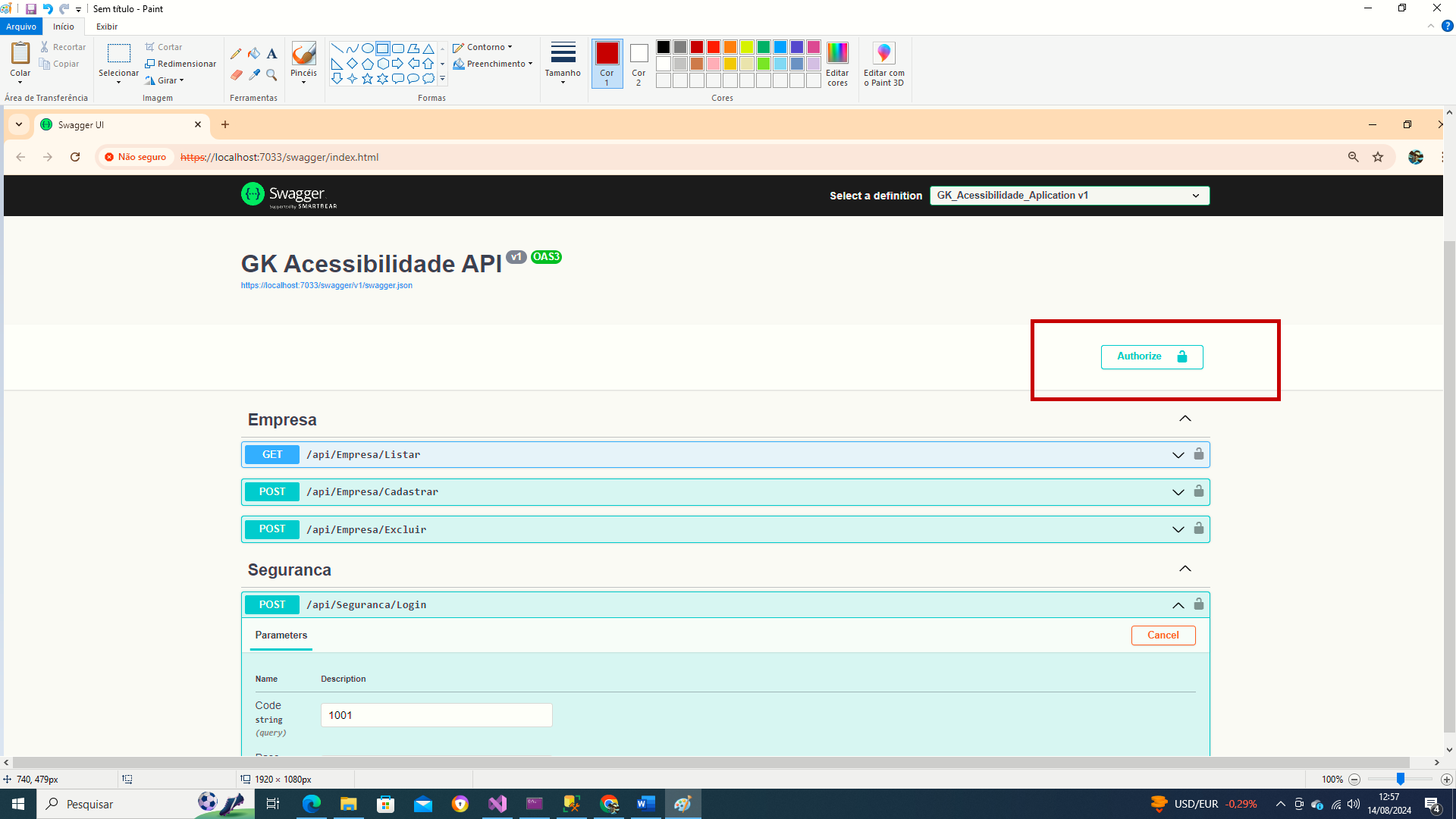
5.5. Empresa

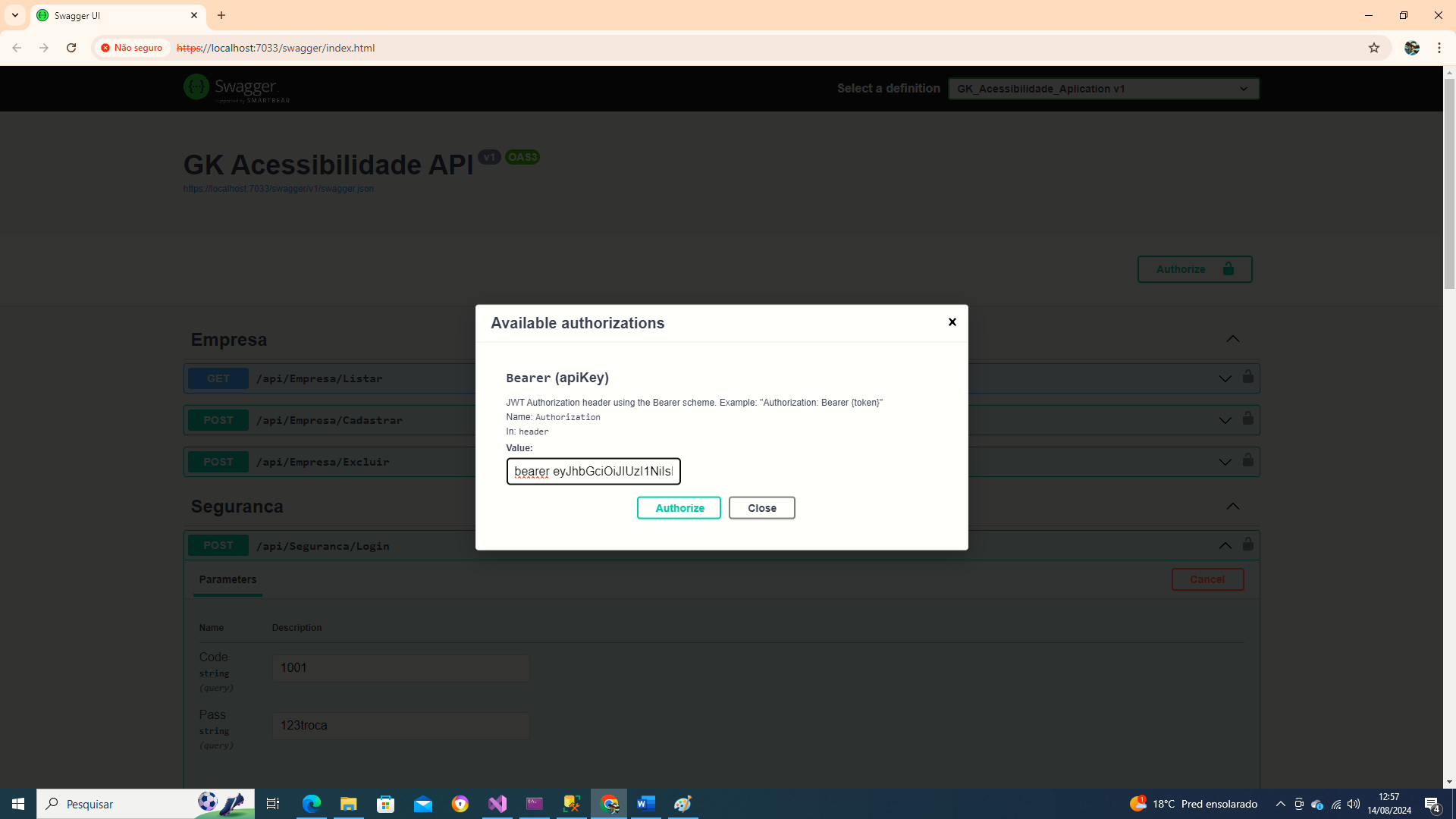
* Descrição: Estrutura que representa uma empresa.
* Campos:
  + id: Identificador da empresa.
  + nome: Nome da empresa.
  + cnpj: CNPJ da empresa.
  + endereco: Endereço

LOGIN Exemplo

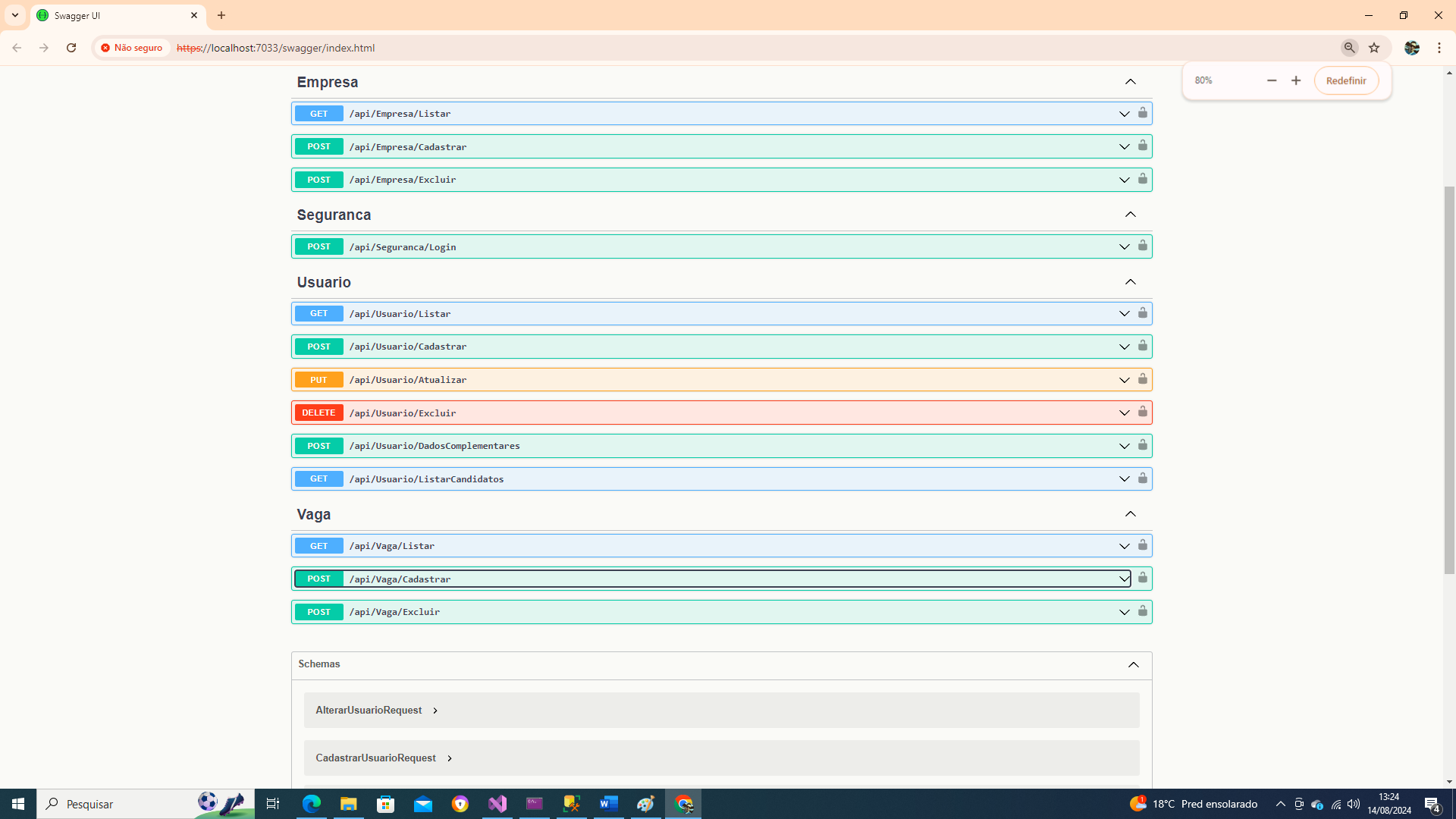


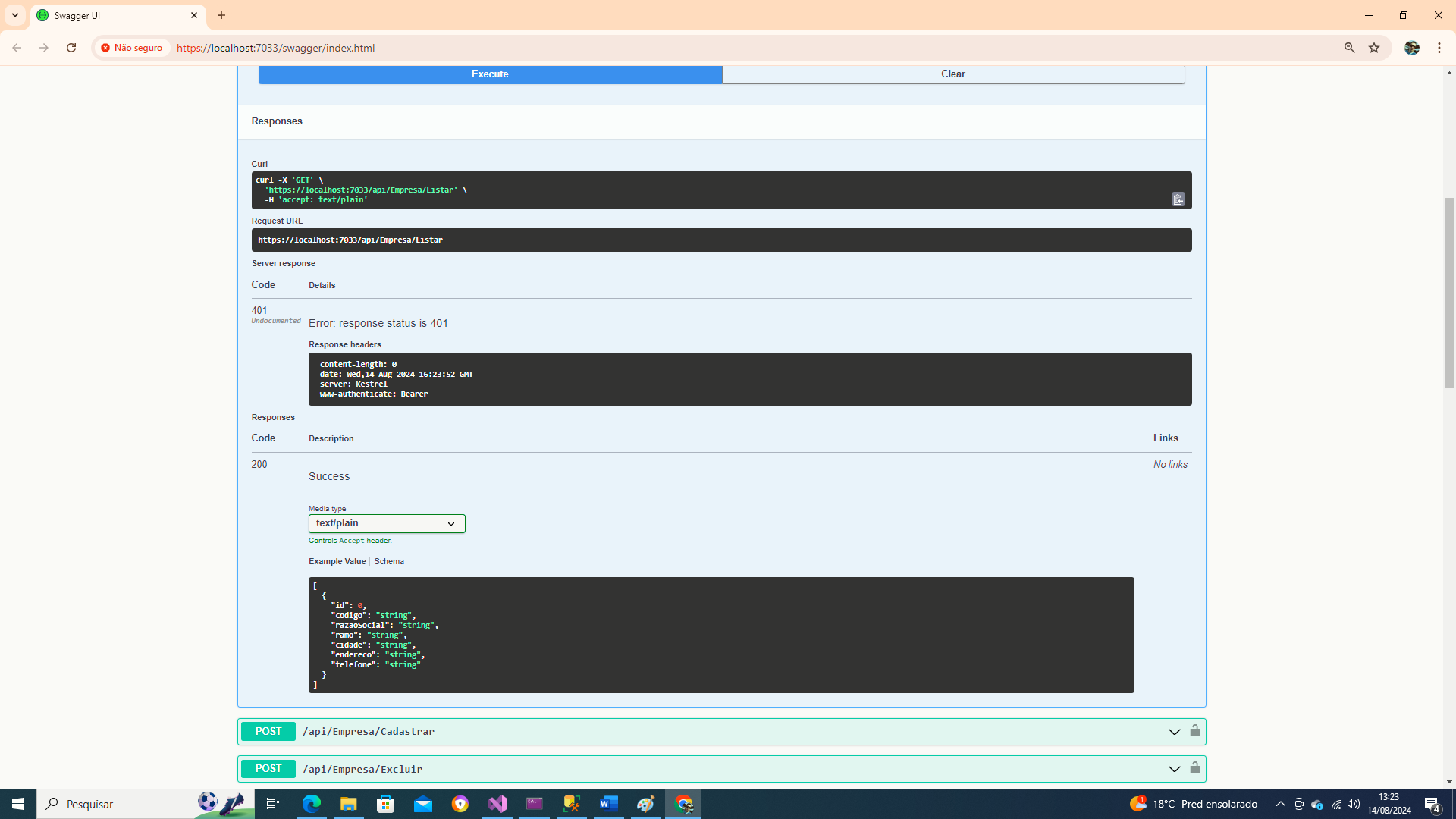
Envia e recebe o Bearer Token  
  


Para iniciar a utilização do Swagger, é necessário o Token e utiliza no authenticate acima:  
  


Copiar e Cola utilizando a palavra bearer antes do token  
adicionar um espaço entre o bearer e o token  
  


2 Endpoints



3 - Caso executem os endpoints sem autenticar o retorno será 401  
  


Para realizar a autenticação será necessário inserir manualmente o primeiro usuário na tabela [SECURITY].[TB\_LOGIN]

**USE [DB\_ACESSECIBILIDADE]**

**GO**

**INSERT INTO [SECURITY].[TB\_LOGIN]**

**([Id\_Type]**

**,[Code]**

**,[Email]**

**,[Pass]**

**,[Date])**

**VALUES**

**(<Id\_Type, int,>**

**,<Code, varchar(100),>**

**,<Email, varchar(100),>**

**,<Pass, varchar(100),>**

**,<Date, datetime,>)**

**GO**