**Documentação - Projeto Sementes - Back End**

**Tecnologias Utilizadas**

* **Node.js:** Ambiente de execução para o código JavaScript.
* **NestJS:** Framework Node.js progressivo para construir aplicações do lado do servidor.
* **TypeScript:** Superconjunto do JavaScript que adiciona tipagem estática.
* **TypeORM:** ORM (Object Relational Mapper) para trabalhar com banco de dados relacionais.
* **PostgreSQL:** Sistema de gerenciamento de banco de dados relacional.
* **JWT (JSON Web Token):** Padrão da indústria para autenticação e autorização.

**Bibliotecas Utilizadas**

* **@nestjs/common:** Módulo que fornece decorators, classes e métodos comuns para aplicações NestJS.
* **@nestjs/passport:** Integração com a biblioteca Passport para autenticação.
* **@nestjs/jwt:** Utilitário para trabalhar com JSON Web Tokens no NestJS.
* **@nestjs/typeorm:** Integração com a biblioteca TypeORM para trabalhar com banco de dados no NestJS.
* **class-validator:** Biblioteca para validação de dados de entrada.
* **typeorm:** ORM para Node.js que suporta diferentes bancos de dados.

**Relações Entre Entidades**

**UsuarioEntity (usuário)**

Representa os usuários do sistema.

* **id:** Identificador único do usuário (UUID).
* **email:** Email do usuário (único).
* **password:** Senha do usuário.
* **name:** Nome do usuário.
* **cargo:** Cargo do usuário.

**AvaliacaoEntity (avaliação)**

Representa as avaliações feitas para os usuários.

* **id:** Identificador único da avaliação (UUID).
* **usuarioAvaliadoId:** Identificador do usuário avaliado (UUID).
* **comunicacao:** Nota de comunicação (0 a 4).
* **proatividade:** Nota de proatividade (0 a 4).
* **inteligenciaEmocional:** Nota de inteligência emocional (0 a 4).
* **flexibilidade:** Nota de flexibilidade (0 a 4).
* **criatividade:** Nota de criatividade (0 a 4).
* **observacao:** Nota de observação (0 a 4).
* **comentario:** Comentário sobre a avaliação.

**Feedback**

Representa os feedbacks relacionados às avaliações dos usuários.

* **id:** Identificador único do feedback (UUID).
* **userId:** Identificador do usuário avaliado (UUID).
* **message:** Mensagem de feedback.

**Resumo das Rotas**

**Avaliações**

* **POST /avaliacoes/criar:** Cria uma nova avaliação.
* **GET /avaliacoes:** Lista todas as avaliações.
* **GET /avaliacoes/:id:** Busca uma avaliação por ID.
* **PUT /avaliacoes/:id:** Atualiza uma avaliação por ID.
* **DELETE /avaliacoes/:id:** Remove uma avaliação por ID.

**Feedbacks**

* **POST /feedbacks:** Cria um novo feedback.
* **GET /feedbacks:** Lista todos os feedbacks.
* **GET /feedbacks/user/:userId:** Busca feedbacks para um usuário específico.

**Rota de Autenticação (Login)**

* **POST /auth/login**
  + **Descrição:** Autentica o usuário com as credenciais fornecidas e retorna um token JWT válido.
  + **Body:**
    - **email**: E-mail do usuário.
    - **password**: Senha do usuário.
  + **Retorno:**
    - **{ token: string }**: Token JWT válido para autenticação subsequente.

**Rotas de Usuários**

* **POST /usuarios/register**
  + **Descrição:** Registra um novo usuário na aplicação.
  + **Body**:
    - **nome**: Nome do usuário.
    - **email**: E-mail do usuário.
    - **password**: Senha do usuário.
    - **cargo**: Cargo do usuário.
  + **Retorno**:
    - **{ message: string, user: UsuarioEntity, token: string }**: Mensagem de sucesso, detalhes do usuário registrado e token JWT para autenticação subsequente.
* **GET /usuarios**
  + **Descrição:** Retorna todos os usuários cadastrados na aplicação.
  + **Retorno**:
    - **UsuarioEntity[]**: Lista de todos os usuários cadastrados.
* **POST /usuarios/login**
  + **Descrição:** Rota de login para autenticação de usuários.
  + **Body**:
    - **email**: E-mail do usuário.
    - **password**: Senha do usuário.
  + **Retorno**:
    - **{ token: string }**: Token JWT válido para autenticação subsequente.

