API de Autenticação Django Rest ramework

A API de Autenticação do Django é um sistema de autenticação robusto, construído com Django e Django Rest ramework. Ela oferece funcionalidades como registro de usuários, login, redefinição de senha, alteração de e-mail, entre outros recursos relacionados à autenticação.

Índice

Introdução

Pré-requisitos

Certifique-se de ter os seguintes pré-requisitos instalados:

- Python
- Django
- Django Rest ramework

Instalação

PRO

1. **Clone o repositório:**

```
git clone <seu-repositorio>
```

2. **Navegue até o diretório do projeto:**

```
cd base_user
```

3. **Com Docker (Recomendado):**

```
docker-compose up --build
```

A API estará disponível em `http://localhost:8000/api/contas/`

4. **Sem Docker (Desenvolvimento local):**

   Certifique-se de ter o Poetry instalado:

   ```
   curl -sSL https://install.python-poetry.org | python3 -
   ```

   Instale as dependências:

   ```
   poetry install
   ```

   Ative o ambiente virtual:

   ```
   poetry shell
   ```

   Aplique as migrações:

   ```
   python manage.py migrate
   ```

   Execute o servidor de desenvolvimento:

   ```
   python manage.py runserver
   ```

PRO

A API deve estar acessível em `http://localhost:8000/api/contas/`

Uso

Endpoints da API

Todos os endpoints da API estão acessíveis sob o prefixo `/api/contas/`. Os endpoints são:

1. **Informações da Conta do Usuário:** `/api/contas/`
   - Método: `GET`
   - Requer autenticação
2. **Alteração de Conta do Usuário:** `/api/contas/editar-detalhes/`
   - Método: `POST`
   - Requer autenticação

- Atualiza informações do usuário (primeiro nome, sobrenome)

3. **Login de Usuário:** `/api/contas/login/`
    - Método: `POST`
    - Entrada: E-mail e senha
    - Retorna informações do usuário
4. **Cadastro de Usuário:** `/api/contas/cadastro/`
    - Método: `POST`
    - Entrada: Primeiro nome, sobreno
    - Retorna mensagem de sucesso
5. **Ativação de Conta:** `/api/contas/ativar-conta/`
    - Método: `POST`
    - Entrada: Código de ativação
    - Ativa a conta do usuário para que ele possa fazer login
6. **Logout do Usuário:** `/api/contas/logout/`
    - Método: `POST`
    - Requer autenticação
    - az o logout do usuário
7. **Solicitação de Redefinição de Senha:** `/api/contas/resetar-senha/`
    - Método: `POST`
    - Entrada: E-mail
    - Envia um código de verificação para redefinir a senha para o e-mail do usuário
8. **Verificação de Redefinição de Senha:** `/api/contas/resetar-senha/verificar/`
    - Método: `POST`
    - Entrada: Código de verificação e nova senha
    - Redefine a senha do usuário
9. **Solicitação de Alteração de E-mail:** `/api/contas/alterar-email/`
    - Método: `POST`
    - Requer autenticação
    - Envia um código de verificação para alterar o e-mail para o e-mail do usuário
10. **Verificação de Alteração de E-mail:** `/api/contas/alterar-email/verificar/`
    - Método: `POST`
    - Entrada: Código de verificação e novo e-mail
    - Altera o e-mail do usuário
11. **Alteração de Senha:** `/api/contas/alterar-senha/`

- Método: `POST`
- Requer autenticação
- Entrada: Senha antiga e nova senha
- Altera a senha do usuário

Autenticação

- A API utiliza Token Authentication para proteger os endpoints quando necessário.

Contribuição

- aça um fork do repositório.
- Crie um novo branch para sua feature: `git checkout -b feature-name`.
- Commite suas mudanças: `git commit -m 'Adicionar nova feature'`.
- Envie para o branch: `git push origin feature-name`.
- Envie um pull request.

desenvolvimento local

quando criar seu banco local postergree mude para a porta 5433 para que nao haja conflito com a porta do docker

Explicação das alterações:

1. **Endpoints Públicos (AllowAny):**
   - CadastrarUsuarioView
   - LoginView
   - AtivacaoContaView
   - ResetSenhaView
   - VerificarResetSenhaView
   - SolicitarNovoCodigoAtivacaoView

PRO

2. **Endpoints Protegidos (IsAuthenticated):**
   - PerfilView
   - AlterarPerfilView
   - LogoutView
   - AlterarEmailView
   - VerificarAlteracaoEmailView
   - AlterarSenhaView

3. **Benefícios:**
   - Maior segurança
   - Controle de acesso explícito
   - Código mais organizado e claro

4. **Observações:**

- `AllowAny` permite acesso a qualquer usuário (autenticado ou não)
- `IsAuthenticated` requer um token válido ou sessão ativa