To-Do List App – Teste Prático Dev Jr

Este projeto é uma aplicação web para gerenciamento de tarefas (To-Do List), desenvolvido como parte do processo seletivo para a vaga de Desenvolvedor Python Júnior na Advice Health.

Tecnologias Utilizadas

Back-End

- Python 3.12
- Django 5.x
- Django REST Framework
- SimpleJWT (autenticação com tokens)
- SQLite (em desenvolvimento, fácil de migrar para PostgreSQL)
- Pytest (para testes automatizados)
- Swagger (documentação automática)

Funcionalidades

Funcionalidades Obrigatórias

- [x] Cadastro e login de usuários via API
- [x] CRUD de tarefas com vinculação por usuário
- [x] Marcar tarefas como concluídas/não concluídas
- [x] Filtro de tarefas (por título e descrição)
- [x] Paginação de tarefas
- [x] API desenvolvida em Django REST Framework

Funcionalidades Opcionais

- [x] Cadastro de categorias de tarefas
- [ ] Compartilhamento de tarefas (não implementado)
- [ ] Dockerização (não implementado por limitação de hardware)
- [ ] Testes com Pytest (em desenvolvimento)

Estrutura do Projeto

```
├── core/
│   └── settings.py
│   └── urls.py
├── tasks/
│   ├── models.py
│   ├── views.py
```

```
│   ├── serializers.py
│   ├── urls.py
│   └── tests/
├── manage.py
└── requirements.txt
```

Como Rodar o Projeto Localmente:

1.Clone o repositório

---- git clone https://github.com/Answercode2023/teste-python-back
---- cd teste-python-back

2.Crie o ambiente virtual

--- python -m venv venv
--- venv\Scripts\activate # Windows
--- source venv/bin/activate # Linux/macOS

3. Instale as dependências

--- pip install -r requirements.txt

4. Rode as migrações

---python manage.py makemigrations
---python manage.py migrate

5. (Opcional) Crie um superusuário

--- python manage.py createsuperuser

6. Inicie o servidor

--- python manage.py runserver

🔐 Endpoints Importantes

🔓 Autenticação

--- POST /api/register/ → cria novo usuário

--- POST /api/auth/login/ → retorna token de acesso e refresh

--- POST /api/auth/refresh/ → atualiza token

📌 Tarefas

--- GET /api/tasks/ → lista tarefas

--- POST /api/tasks/ → cria nova tarefa

--- GET /api/tasks/{id}/ → detalhe de tarefa

--- PUT/PATCH /api/tasks/{id}/ → atualiza tarefa

--- DELETE /api/tasks/{id}/ → remove tarefa

--- Filtros disponíveis: ?search=palavra

--- Paginação: ?page=1

🗂 Categorias

--- GET /api/categories/

--- POST /api/categories/

--- PUT/PATCH/DELETE /api/categories/{id}/

🥷 Autor
Lucas Vinicius
Desenvolvedor Python Júnior
LinkedIn | GitHub