

INSTITUTO FEDERAL

Ceará

Campus Tianguá



Nome: Pedro Henrique Sousa da Silva

Matrícula: 2024116020114

Curso: Técnico em Informática para Internet

Disciplina: Desenvolvimento WEB III

Professor: Dr. Nécio de Lima Veras

1. Link do Repositório

O código fonte completo do projeto está disponível no GitHub: **Link:**

<https://github.com/P4dro-Dev/TAREFA-V-DEV-WEB-III-IFCE>

2. Desenvolvimento e Implementação

2.1. Controle de Versão (Git)

Para iniciar a atividade, foi criada uma branch específica chamada banco-de-dados para isolar o desenvolvimento das funcionalidades de persistência de dados.

- Comando utilizado: `git checkout -b banco-de-dados`

2.2. Modelo de Dados (Model)

Foi criada a classe Autor no arquivo `models.py`, contendo os campos `nome`, `email`, `mini_curriculo` e `criado_em`. Este modelo representa a tabela no banco de dados SQLite.

Python

```
class Autor(models.Model):
```

```
    nome = models.CharField('Nome', max_length=200)
```

```
    email = models.EmailField('E-mail', blank=True, null=True)
```

```
    mini_curriculo = models.TextField('Mini currículo', blank=True)
```

```
    criado_em = models.DateTimeField('Criado em', auto_now_add=True)
```

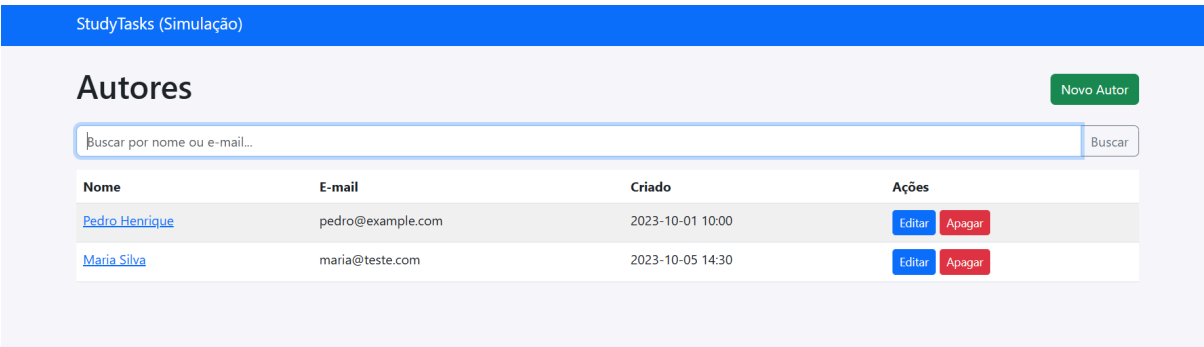
3. Demonstração das Funcionalidades (CRUD)

Abaixo estão os prints comprovando o funcionamento das operações de Criar, Ler, Atualizar, Deletar e Buscar.

A) Listagem de Registros (Read)

A página inicial (/) carrega todos os registros do banco de dados, ordenados pela data de criação.

Imagem 1 - Tela Inicial com Listagem de Autores:

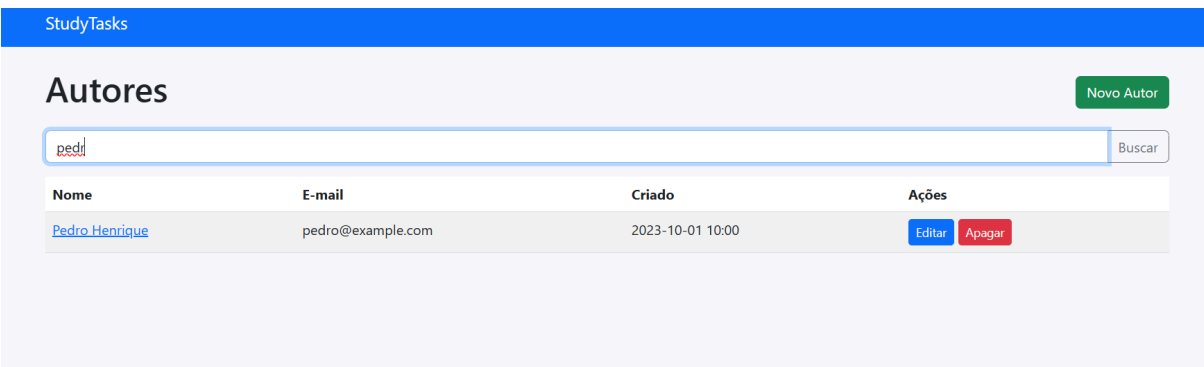


Descrição: A imagem acima mostra a listagem inicial com os autores "Pedro Henrique" e "Maria Silva" já cadastrados.

B) Inserção de Novo Registro (Create)

Ao clicar em "Novo Autor", o sistema exibe um formulário (ModelForm) para cadastro.

Imagem 2 - Formulário de Cadastro:



Preenchimento do formulário para cadastrar o autor "Nécio Veras" com seu e-mail e mini currículo.

Após salvar, o novo autor aparece na listagem principal.

Imagem 3 - Listagem Atualizada após Inserção:

StudyTasks

Criar Autor

Nome:

E-mail:

Mini currículo:

SalvarVoltar

StudyTasks

Autores

Novo Autor

Buscar

Nome	E-mail	Criado	Ações
Nécio Veras	necio123@ifce.edu.br	2025-12-07 19:12	<button>Editar</button> <button>Apagar</button>
Pedro Henrique	pedro@example.com	2023-10-01 10:00	<button>Editar</button> <button>Apagar</button>
Maria Silva	maria@teste.com	2023-10-05 14:30	<button>Editar</button> <button>Apagar</button>

O autor "Nécio Veras" aparece no topo da lista, confirmando a persistência no banco de dados.

C) Busca de Dados (Search)

Foi implementada uma busca que filtra registros tanto pelo **Nome** quanto pelo **E-mail**, utilizando a query `Q(nome__icontains=q) | Q(email__icontains=q)`.

Imagem 4 - Realizando uma busca:

StudyTasks

Autores

Novo Autor

Buscar

Nome	E-mail	Criado	Ações
Pedro Henrique	pedro@example.com	2023-10-01 10:00	<button>Editar</button> <button>Apagar</button>

Ao digitar "pedr" no campo de busca, o sistema filtra dinamicamente a lista para exibir apenas o registro correspondente ("Pedro Henrique").

D) Alteração de Dados (Update)

Ao clicar no botão "Editar", os dados atuais do autor são carregados no formulário para modificação.

Imagem 5 - Edição de Registro:

A imagem mostra a interface de usuário para a edição de um registro de autor. No topo, há uma barra azul com o texto "StudyTasks". Abaixo, o título "Alterar Autor" é exibido. O formulário contém três campos de entrada: "Nome:" com o valor "Pedro Henrique", "E-mail:" com o valor "pedro@example.com" e "Mini currículo:" com o valor "Dev Fullstack". Abaixo dos campos, há dois botões: "Salvar" (azul) e "Voltar" (cinza).

Tela de edição carregando os dados do autor "Pedro Henrique", permitindo alterar o campo "Mini currículo" para "Dev Fullstack".

E) Exclusão de Registro (Delete)

O sistema possui um botão "Apagar" (vermelho) na listagem. Ao clicar, o usuário é direcionado para uma tela de confirmação antes que o registro seja removido permanentemente do banco de dados.

Funcionalidade visível nos botões "Apagar" presentes nas Imagens 1, 3 e 4.

4. Integração com Admin

O modelo Autor foi registrado no `admin.py`, permitindo o gerenciamento também pela interface administrativa do Django (`/admin/`).

```
Python
@admin.register(Autor)
class AutorAdmin(admin.ModelAdmin):
```

```
list_display = ('nome', 'email', 'criado_em')  
search_fields = ('nome', 'email')
```

5. Finalização (Git Merge)

Após a implementação e testes de todas as funcionalidades, as alterações foram "commitadas" e a branch banco-de-dados foi mesclada com a branch principal (main).

Comandos utilizados:

1. `git add .`
2. `git commit -m "Implementa modelo Autor e CRUD completo"`
3. `git push origin banco-de-dados`
4. `git checkout main`
5. `git merge banco-de-dados`
6. `git push origin main`