**📘 Documentação Técnica – Sistema de Gerenciamento de Clientes**

Seja bem-vindo ao **Sistema de Gerenciamento de Clientes**, uma aplicação simples e funcional desenvolvida em **Python**, com interface gráfica via **Tkinter** e persistência de dados utilizando o banco **SQLite**. Este projeto é ideal para estudantes e iniciantes que desejam aprender a integrar interface gráfica com banco de dados e desenvolver funcionalidades CRUD (Create, Read, Update, Delete).

**⚙️ Funcionalidades Principais**

A aplicação permite:

* ✅ Cadastrar novos clientes.
* 📋 Visualizar a lista completa de clientes.
* 🔍 Pesquisar clientes por nome, sobrenome, email ou CPF.
* ✏️ Editar as informações de um cliente.
* 🗑️ Remover um cliente do banco de dados.

A interface gráfica é intuitiva, com campos de entrada e botões correspondentes a cada ação. Os dados são salvos localmente em um arquivo SQLite chamado clientes.db.

**🗂️ Estrutura do Projeto**

O código-fonte é dividido em três arquivos principais para facilitar o entendimento e a manutenção:

**🔸 Gui.py**

* Responsável por toda a interface gráfica com o usuário.
* Cria os campos de entrada, botões de ação e uma tabela para exibir os clientes.
* Conecta-se com o Backend.py para realizar operações no banco de dados.

**🔸 Backend.py**

* Cuida da comunicação com o banco SQLite.
* Contém funções para criar a tabela, inserir, consultar, atualizar e deletar registros.
* Utiliza comandos SQL parametrizados para evitar falhas de segurança.

**🔸 application.py**

* Arquivo principal que inicializa a aplicação.
* Configura o banco de dados e exibe a janela gráfica com a interface do usuário.

**💡 Requisitos para Execução**

Antes de rodar o sistema, verifique se você possui:

* ✅ Python 3.8 ou superior instalado ([python.org](https://python.org)).
* ✅ Tkinter (vem embutido no Python).
* ✅ SQLite (já incluso no Python via módulo sqlite3).
* ⚙️ **Opcional:** PyInstaller, caso deseje gerar um executável da aplicação.

**🚀 Como Rodar o Projeto**

**Passos básicos:**

1. **Crie uma pasta para o projeto**, por exemplo: gerenciador\_clientes.
2. **Salve os três arquivos** Python (Gui.py, Backend.py, application.py) dentro dessa pasta.
3. **Verifique a instalação do Python** com:

css

CopiarEditar

python --version

1. **Execute o programa** com:

bash

CopiarEditar

cd caminho/para/gerenciador\_clientes

python application.py

Uma janela será aberta com a interface gráfica da aplicação.

**🖱️ Utilizando a Aplicação**

Ao iniciar o sistema, você verá:

* **Campos de entrada** para Nome, Sobrenome, Email e CPF.
* **Botões funcionais:**
  + **Adicionar:** Insere um novo cliente.
  + **Atualizar:** Edita um cliente já existente (selecionado da tabela).
  + **Deletar:** Remove o cliente selecionado.
  + **Buscar:** Filtra os clientes com base nos campos preenchidos.
  + **Limpar:** Limpa os campos de entrada.
* **Tabela (Treeview):** Exibe os registros armazenados no banco.

**🔄 Passo a Passo das Operações**

| **Operação** | **Descrição** |
| --- | --- |
| **Adicionar** | Preencha todos os campos e clique em "Adicionar". O cliente será salvo. |
| **Visualizar** | A tabela mostra automaticamente os dados ao iniciar ou após qualquer ação. |
| **Buscar** | Digite parte ou todo o valor em um dos campos e clique em "Buscar". |
| **Atualizar** | Selecione um cliente na tabela, edite os dados e clique em "Atualizar". |
| **Deletar** | Selecione um cliente e clique em "Deletar" para removê-lo do banco. |

**📦 Gerando um Executável com PyInstaller**

Deseja rodar o programa sem precisar abrir o terminal? Você pode transformar o script em um .exe.

**Instruções:**

1. Instale o PyInstaller:

nginx

CopiarEditar

pip install pyinstaller

1. Gere o executável:

bash

CopiarEditar

cd caminho/para/gerenciador\_clientes

pyinstaller --onefile application.py

1. O executável será criado dentro da pasta dist. Basta clicar duas vezes para abrir.

**Dica:** Compartilhe o .exe e o arquivo clientes.db juntos, caso o banco já tenha dados.

**🗃️ Estrutura da Tabela no Banco de Dados**

A tabela clientes é criada automaticamente na primeira execução, com a seguinte estrutura:

| **Campo** | **Tipo** | **Descrição** |
| --- | --- | --- |
| id | INTEGER | Identificador único (chave primária) |
| nome | TEXT | Nome do cliente |
| sobrenome | TEXT | Sobrenome do cliente |
| email | TEXT | Email do cliente |
| cpf | TEXT | CPF do cliente |

Todos os campos são obrigatórios, exceto o id, que é gerado automaticamente.

**🛠️ Resolvendo Problemas Comuns**

* ❌ **Erro "No module named tkinter"**  
  Tkinter pode estar ausente na sua instalação do Python. Reinstale o Python ou tente:

nginx

CopiarEditar

pip install tk

* ❌ **Erro ao rodar o executável:**  
  Tente rodar pelo terminal para ver mensagens detalhadas:

bash

CopiarEditar

./dist/application.exe

* ❌ **Tabela não carrega os dados:**  
  Verifique se o arquivo clientes.db está no mesmo diretório do executável ou dos scripts Python.

**🎓 Aprendizados que Este Projeto Oferece**

* **Python:** Criação de classes, métodos, modularização e boas práticas.
* **SQLite:** Manipulação de banco de dados com SQL seguro (parametrizado).
* **Tkinter:** Desenvolvimento de interfaces gráficas interativas.
* **Organização de código:** Separação de responsabilidades entre GUI e lógica de dados.
* **Distribuição de software:** Criação de executável com PyInstaller.

**📈 Sugestões de Evolução**

Quer ir além? Experimente implementar:

* ✔️ Validação de CPF.
* ✔️ Exportação de clientes para CSV.
* ✔️ Um botão para recarregar todos os dados após a busca.
* ✔️ Melhorias visuais (cores, ícones, temas).
* ✔️ Relacionamentos com outras tabelas, como "Endereços" ou "Pedidos".