# Gerenciador de Clientes: Seu Assistente Pessoal de Dados

Bem-vindo ao **Gerenciador de Clientes**, sua nova ferramenta em Python para manter as informações dos seus clientes organizadas! Sabe aqueles dados importantes como nome, sobrenome, e-mail e CPF? Com este aplicativo, você consegue gerenciar tudo isso de um jeito simples e visual, usando um **banco de dados SQLite**.

Este projeto é um exemplo prático de como a mágica acontece quando combinamos **Python**, **SQL** e **SQLite**, tudo isso com uma **interface gráfica intuitiva feita em Tkinter**. É uma ótima forma de entender o caminho completo: desde criar uma telinha para o usuário até armazenar dados de forma persistente.

## O Que Este Aplicativo Pode Fazer Por Você?

Este gerenciador foi desenhado para ser direto e eficiente, permitindo que você:

* **Adicione** novos clientes à sua base de dados, registrando todas as informações essenciais.
* **Visualize** todos os seus clientes em uma tabela organizada, facilitando a consulta rápida.
* **Busque** clientes específicos. Basta digitar um nome, sobrenome, e-mail ou CPF, e ele encontra quem você procura.
* **Atualize** os dados de um cliente existente, caso haja alguma mudança nas informações.
* **Delete** clientes do banco de dados quando eles não forem mais necessários.

A interface é super fácil de usar, com campos claros para digitar e botões grandes para todas as ações. E para guardar tudo? Usamos um arquivo simples chamado clientes.db, que fica ali na sua pasta.

## Por Dentro do Projeto: A Estrutura

Para manter tudo organizado e fácil de entender, o projeto é dividido em três arquivos Python, cada um com uma função específica:

### Gui.py

* + Este arquivo é a "cara" da sua aplicação. Ele contém a classe Gui, que constrói toda a **interface gráfica** que você vê, usando o **Tkinter**.
  + Aqui você encontra os campos onde digita os dados do cliente (Nome, Sobrenome, Email, CPF), os botões de ação (Adicionar, Atualizar, Deletar, Buscar, Limpar) e a tabela que exibe a lista de clientes.
  + Ele funciona como a ponte entre o que o usuário faz e as operações no banco de dados, "conversando" com a classe Backend.

### Backend.py

* + Este é o "cérebro" da operação. A classe Backend é a responsável por **gerenciar tudo que acontece no banco de dados SQLite**.
  + Ele tem os métodos para se conectar ao banco, criar a tabela de clientes, e realizar todas as operações de inserir, visualizar, buscar, atualizar e deletar.
  + Fique tranquilo: ele usa consultas SQL seguras para proteger seus dados!

### application.py

* + Este é o arquivo que **coloca tudo para funcionar**. É o ponto de partida do seu programa.
  + Sua única tarefa é inicializar o banco de dados (garantindo que tudo esteja pronto) e, em seguida, abrir a janela principal da aplicação gráfica para você começar a usar.

## O Que Você Precisa Para Rodar (Pré-requisitos)

Para colocar seu Gerenciador de Clientes para trabalhar, você vai precisar de algumas coisas simples:

* **Python 3.x:** Recomendamos o Python 3.8 ou uma versão mais recente. Você pode baixá-lo no site oficial: [python.org](https://www.python.org/).
* **Tkinter:** Ótimas notícias! O Tkinter já vem junto com a maioria das instalações do Python, então você provavelmente não precisa instalar nada extra.
* **SQLite:** Mais uma boa notícia! O módulo sqlite3 também já está incluído no Python, sem a necessidade de instalações adicionais.
* **PyInstaller (opcional):** Se você quiser transformar seu aplicativo em um arquivo executável (tipo um .exe no Windows), o PyInstaller será útil. Mas ele é opcional, só se for para "empacotar" o app.

## Mãos à Obra: Como Configurar e Executar

Siga estes passos simples para ter seu Gerenciador de Clientes funcionando:

### Pegue os Arquivos:

* + Crie uma pasta vazia para o seu projeto (por exemplo, gerenciador\_clientes).
  + Salve os três arquivos principais (Gui.py, Backend.py, application.py) dentro dessa pasta.

### Confira o Python:

* + Abra o seu terminal (o "Prompt de Comando" no Windows ou o "Terminal" no Linux/Mac).
  + Digite python --version e pressione Enter.
  + Se você vir a versão do Python (ex: Python 3.9.2), está tudo certo! Se não, é hora de baixar e instalar o Python.

### Dê o Comando de Início:

* + No terminal, navegue até a pasta do seu projeto. Por exemplo, se sua pasta estiver na área de trabalho: cd C:\Users\SeuUsuario\Desktop\gerenciador\_clientes (no Windows) ou cd ~/Desktop/gerenciador\_clientes (no Linux/Mac).
  + Agora, digite o comando mágico para iniciar a aplicação: python application.py
  + Pronto! Uma nova janela gráfica se abrirá, e você verá o Gerenciador de Clientes pronto para ser usado.

## Usando a Aplicação: Um Guia Rápido

Ao abrir o application.py, você verá uma janela dividida em algumas seções:

* **Campos de Entrada:** São os espaços para Nome, Sobrenome, Email e CPF.
* **Botões de Ação:**
  + **Adicionar:** Para incluir um novo cliente na sua lista.
  + **Atualizar:** Para modificar os dados de um cliente que já existe (primeiro selecione-o na tabela!).
  + **Deletar:** Para remover um cliente selecionado.
  + **Buscar:** Para filtrar a tabela e encontrar clientes com base no que você digita nos campos.
  + **Limpar:** Para deixar os campos de entrada vazios novamente.
* **Tabela (Treeview):** É a sua lista dinâmica de clientes. Ela mostra todos os clientes que estão no banco de dados.

### Passo a Passo de Uso:

1. **Para Adicionar um Cliente:**
   * Preencha todos os campos (Nome, Sobrenome, Email, CPF).
   * Clique no botão "**Adicionar**".
   * Uma mensagem de sucesso aparecerá, e o novo cliente será adicionado à tabela.
2. **Para Visualizar Clientes:**
   * A tabela já exibe automaticamente todos os clientes assim que você abre o aplicativo.
3. **Para Buscar Clientes:**
   * Digite o que você procura (um nome, um CPF, parte de um e-mail, etc.) em qualquer um dos campos de entrada.
   * Clique no botão "**Buscar**".
   * A tabela será atualizada para mostrar apenas os clientes que correspondem aos seus critérios.
4. **Para Atualizar um Cliente:**
   * **Primeiro, clique no cliente** que você quer editar na tabela. Os dados dele aparecerão nos campos de entrada.
   * Faça as alterações que quiser nos campos.
   * Clique no botão "**Atualizar**".
   * A tabela se ajustará com as novas informações.
5. **Para Deletar um Cliente:**
   * **Clique no cliente** que você deseja remover na tabela.
   * Clique no botão "**Deletar**".
   * O cliente será removido, e a tabela será atualizada.

## Tornando-o Executável (Com PyInstaller)

Quer compartilhar seu aplicativo sem que a pessoa precise instalar Python? Use o PyInstaller para criar um arquivo executável!

### Como Fazer:

1. **Instale o PyInstaller:**
   * No seu terminal, digite: pip install pyinstaller
2. **Crie o Executável:**
   * Navegue até a pasta do seu projeto no terminal.
   * Execute o comando mágico: pyinstaller --onefile application.py
     + O --onefile é legal porque cria um único arquivo executável, facilitando muito o compartilhamento!
   * Isso vai criar uma pasta chamada dist dentro do seu projeto. Lá dentro, você encontrará o arquivo executável (ex: application.exe no Windows).
3. **Use o Executável:**
   * Vá até a pasta dist (por exemplo, gerenciador\_clientes/dist).
   * Clique duas vezes no application.exe (ou execute-o pelo terminal).
   * O aplicativo abrirá como de costume, mas agora **não precisa mais do Python instalado** no computador de quem for usar!

### Dica Importante:

O arquivo executável pode ser um pouco "pesadinho" (50-100 MB), porque ele inclui o Python e todas as dependências necessárias. Para compartilhar, basta copiar esse .exe (e o arquivo clientes.db, se já tiver dados) para outros computadores.

## A Estrutura do Seu Banco de Dados

Seus dados são armazenados em um arquivo chamado **clientes.db**, que fica na mesma pasta do projeto. Dentro dele, temos uma tabela chamada **clientes**, com a seguinte organização:

* **id**: Um número único para cada cliente, que o sistema cria automaticamente.
* **nome**: O nome do cliente (texto).
* **sobrenome**: O sobrenome do cliente (texto).
* **email**: O e-mail do cliente (texto).
* **cpf**: O CPF do cliente (texto).

A boa notícia é que o método Backend.initDB() cuida de criar essa tabela para você automaticamente na primeira vez que o programa é executado.

## Solução de Problemas Comuns (Dicas Rápidas!)

Se algo não funcionar como esperado, tente estas dicas:

* **"No module named tkinter"**: Isso geralmente significa que o Python não foi instalado corretamente com o Tkinter. Tente reinstalar o Python, ou em alguns casos, pip install tk pode resolver.
* **Problemas ao Executar:** Verifique se os três arquivos (Gui.py, Backend.py, application.py) estão **todos na mesma pasta**. E confirme se você está executando o comando python application.py dentro dessa pasta correta no terminal.
* **Executável Não Abre:** Certifique-se de que o PyInstaller terminou de criar o arquivo com sucesso. No Windows, tente ir na pasta dist e executar o .exe diretamente pelo terminal (./application.exe) para ver se aparece alguma mensagem de erro.
* **Tabela Não Atualiza:** Garanta que o arquivo clientes.db (onde seus dados ficam) esteja na mesma pasta do executável ou dos arquivos Python.

## Para Alunos: O Que Você Vai Aprender Com Este Projeto?

Este Gerenciador de Clientes é um laboratório de aprendizado completo! Com ele, você vai entender na prática:

* **Python:** Como organizar seu código em classes e módulos, e criar funções que fazem o trabalho pesado.
* **SQL e SQLite:** A arte de criar tabelas, e como inserir, consultar, atualizar e deletar dados de forma segura.
* **Tkinter:** Como construir uma interface gráfica real, com campos de texto, botões e tabelas interativas.
* **Boas Práticas de Programação:** A importância de separar a interface da lógica (GUI vs. Backend), usar consultas seguras (parametrizadas) e escrever comentários claros no código.
* **PyInstaller:** A habilidade de transformar seu programa Python em um arquivo executável que qualquer um pode usar, mesmo sem ter Python instalado.

## Próximos Passos (Ideias para Melhorar!)

Depois de dominar o básico, que tal dar um "upgrade" no seu Gerenciador de Clientes?

* **Validação de CPF:** Adicione um código para garantir que o CPF digitado tem o formato correto.
* **Exportar Dados:** Que tal um botão para salvar a lista de clientes em um arquivo CSV, que abre no Excel?
* **Botão "Recarregar Tudo":** Depois de uma busca, seria legal ter um botão para mostrar todos os clientes novamente.
* **Melhorias na Interface:** Deixe o aplicativo ainda mais bonito com algumas cores ou ícones!
  + Contém a classe Gui, que cria a interface gráfica usando Tkinter.
  + Inclui campos de texto para inserir dados do cliente, botões para ações (Adicionar, Atualizar, Deletar, Buscar, Limpar) e uma tabela para mostrar os clientes.
  + Conecta-se à classe Backend para realizar operações no banco de dados.

1. **Backend.py**:
   * Contém a classe Backend, que gerencia todas as operações do banco de dados SQLite.
   * Inclui métodos para conectar ao banco, criar a tabela, inserir, visualizar, buscar, atualizar e deletar clientes.
   * Usa consultas SQL seguras para evitar problemas de segurança.
2. **application.py**:
   * O arquivo principal que inicia a aplicação.
   * Inicializa o banco de dados e abre a janela gráfica.

**Pré-requisitos**

Para executar este projeto, você precisa de:

* **Python 3.x** instalado (recomendado: Python 3.8 ou superior). Você pode baixar em [python.org](https://www.python.org/downloads/).
* **Tkinter**: Já vem incluído com o Python, então você não precisa instalar nada extra.
* **SQLite**: Também incluído no Python (módulo sqlite3), sem necessidade de instalação adicional.
* **PyInstaller** (opcional): Necessário apenas se você quiser criar um arquivo executável. Veja a seção "Criando um Executável" abaixo.

**Como Configurar e Executar**

Siga estas etapas para rodar a aplicação:

1. **Baixe ou crie os arquivos**:
   * Crie uma pasta para o projeto (exemplo: gerenciador\_clientes).
   * Salve os três arquivos (Gui.py, Backend.py, application.py) nessa pasta. Você pode copiar os códigos fornecidos pelo instrutor.
2. **Verifique se o Python está instalado**:
   * Abra um terminal (Prompt de Comando no Windows, Terminal no Linux/Mac) e digite:
   * python --version
   * Se o Python estiver instalado, você verá a versão (exemplo: Python 3.9.2). Caso contrário, baixe e instale o Python.
3. **Execute a aplicação**:
   * No terminal, navegue até a pasta do projeto:
   * cd caminho/para/gerenciador\_clientes
   * Execute o arquivo principal:
   * python application.py
   * Uma janela gráfica será aberta, mostrando a interface do Gerenciador de Clientes.

**Como Usar a Aplicação**

Quando você executa application.py, uma janela aparece com:

* **Campos de entrada**: Para inserir Nome, Sobrenome, Email e CPF.
* **Botões**:
  + **Adicionar**: Insere um novo cliente no banco de dados.
  + **Atualizar**: Atualiza os dados do cliente selecionado na tabela.
  + **Deletar**: Remove o cliente selecionado.
  + **Buscar**: Filtra a tabela com base nos dados inseridos nos campos.
  + **Limpar**: Limpa os campos de entrada.
* **Tabela (Treeview)**: Mostra todos os clientes do banco de dados.

**Passo a passo:**

1. **Adicionar um cliente**:
   * Preencha os campos Nome, Sobrenome, Email e CPF.
   * Clique em "Adicionar".
   * Uma mensagem de sucesso aparecerá, e a tabela será atualizada.
2. **Visualizar clientes**:
   * A tabela mostra automaticamente todos os clientes ao abrir a aplicação.
3. **Buscar clientes**:
   * Insira um valor em qualquer campo (exemplo: Nome ou CPF) e clique em "Buscar".
   * A tabela mostrará apenas os clientes que correspondem aos critérios.
4. **Atualizar um cliente**:
   * Clique em um cliente na tabela (os campos serão preenchidos automaticamente).
   * Edite os campos desejados e clique em "Atualizar".
   * A tabela será atualizada com as novas informações.
5. **Deletar um cliente**:
   * Clique em um cliente na tabela e clique em "Deletar".
   * O cliente será removido, e a tabela será atualizada.

**Criando um Executável com PyInstaller**

Para facilitar o uso da aplicação sem precisar executar o Python no terminal, você pode criar um arquivo executável (.exe no Windows, ou equivalente em outros sistemas) usando o **PyInstaller**.

**Passos para criar o executável:**

1. **Instale o PyInstaller**:
   * No terminal, execute:
   * pip install pyinstaller
2. **Crie o executável**:
   * Navegue até a pasta do projeto no terminal:
   * cd caminho/para/gerenciador\_clientes
   * Execute o comando abaixo para criar um executável a partir de application.py:
   * pyinstaller --onefile application.py
   * Explicação:
     + --onefile: Gera um único arquivo executável (mais fácil de compartilhar).
     + O comando cria uma pasta dist na sua pasta do projeto, contendo o arquivo executável (application.exe no Windows).
3. **Execute o executável**:
   * Vá para a pasta dist (exemplo: gerenciador\_clientes/dist).
   * Clique duas vezes no arquivo application.exe (ou execute pelo terminal).
   * A aplicação abrirá como antes, mas sem precisar do Python instalado no computador.
4. **Dica**:
   * O executável pode ser grande (cerca de 50-100 MB) porque inclui o Python e todas as dependências.
   * Para compartilhar, copie o arquivo .exe (e o arquivo clientes.db, se já tiver dados) para outros computadores.

**Estrutura do Banco de Dados**

O banco de dados SQLite é salvo em um arquivo chamado clientes.db na mesma pasta do projeto. A tabela clientes tem a seguinte estrutura:

* **id**: Um número único para cada cliente (gerado automaticamente).
* **nome**: O nome do cliente (texto).
* **sobrenome**: O sobrenome do cliente (texto).
* **email**: O email do cliente (texto).
* **cpf**: O CPF do cliente (texto).

O método Backend.initDB() cria essa tabela automaticamente na primeira execução.

**Dicas para Solução de Problemas**

* **Erro: "No module named tkinter"**:
  + Certifique-se de que o Python está instalado corretamente. Tkinter vem com o Python, mas pode estar faltando em algumas instalações. Reinstale o Python ou instale Tkinter:
  + pip install tk
* **Erro ao executar o programa**:
  + Verifique se todos os arquivos (Gui.py, Backend.py, application.py) estão na mesma pasta.
  + Confirme que você está executando python application.py na pasta correta.
* **O executável não abre**:
  + Certifique-se de que o PyInstaller foi executado com sucesso.
  + No Windows, tente executar o .exe pelo terminal para ver mensagens de erro:
  + ./dist/application.exe
* **A tabela não atualiza**:
  + Certifique-se de que o arquivo clientes.db está na mesma pasta do executável ou dos arquivos Python.

**Para Alunos: O que você pode aprender com este projeto?**

* **Python**: Como criar classes, métodos estáticos, e organizar código em módulos.
* **SQL e SQLite**: Como criar tabelas, inserir, consultar, atualizar e deletar dados usando consultas SQL seguras.
* **Tkinter**: Como criar uma interface gráfica com campos de texto, botões e tabelas.
* **Boas práticas**: Uso de consultas parametrizadas para segurança, separação de responsabilidades (GUI vs. Backend), e comentários claros.
* **PyInstaller**: Como transformar um programa Python em um executável para facilitar a distribuição.

**Próximos Passos**

Experimente adicionar novas funcionalidades, como:

* Validação de CPF para garantir que o formato está correto.
* Exportar a lista de clientes para um arquivo CSV.
* Adicionar um botão para recarregar todos os clientes após uma busca.
* Melhorar a interface com cores ou ícones.