UNIVERSIDADE ESTÁCIO DE SÁ
CURSO ANÁLISE E DESENVOLVIMENTO DE SISTEMAS
UNIDADE NOVA AMÉRICA

TRABALHO DE GESTÃO DE CLIENTES EM PYTHON (DESENVOLVIMENTO RÁPIDO DE APLICAÇÕES EM PYTHON)

Rio de Janeiro - RJ

Novembro / 2024

202303731213 – Douglas Pereira Galdino

Trabalho de Gestão de Clientes em Python
(Desenvolvimento Rápido de Aplicações em Python)

Trabalho de Gestão de Clientes apresentado a Universidade Estácio de Sá, como exigência para avaliação na disciplina Desenvolvimento Rápido de Aplicações em Python

Orientador:
Prof. Ronaldo Candido dos Santos

SUMÁRIO

1 INTRODUÇÃO

Este documento descreve o desenvolvimento de um sistema de gerenciamento de proponentes, utilizando a metodologia de Desenvolvimento Rápido de Aplicações (RAD), para a associação em que trabalho. A aplicação foi desenvolvida para suprir a necessidade de controle eficiente sobre os dados dos proponentes, possíveis associados que já receberam uma proposta de adesão, seja por um divulgador ou por contato direto com a Associação. O sistema facilita o pré-cadastro e o acompanhamento dos proponentes, centralizando informações essenciais, como dados pessoais e valor da contribuição associativa, e proporcionando uma interface simples e intuitiva.

1.1 DESCRIÇÃO DO PROBLEMA

Atualmente, o processo de cadastro e acompanhamento dos dados dos proponentes é realizado manualmente, o que pode resultar em perda de informações, atrasos no contato com potenciais associados e uma visão limitada sobre o status de cada proponente. A falta de um sistema específico para gerenciar esses dados dificulta o fechamento de novos contratos e a manutenção de um fluxo organizado de prospecção.

1.2 OBJETIVOS

O objetivo principal deste projeto é desenvolver uma solução que atenda à necessidade de organizar e gerenciar os dados de proponentes de forma prática e acessível. O sistema permitirá:

- Cadastro e prospecção: Adicionar e manter registros completos de proponentes, incluindo informações de contato, dados pessoais e valor da contribuição associativa.
- Acompanhamento de status: Acompanhar em qual etapa do processo de prospecção cada proponente está, permitindo visualizar de forma clara os dados necessários para o próximo passo.
- Facilidade de atualização e exclusão: Facilitar o processo de atualização e exclusão de registros, tornando o gerenciamento dos dados mais eficaz.

2 DESENVOLVIMENTO

O sistema de gerenciamento de proponentes foi desenvolvido utilizando Python e a biblioteca Streamlit, visando simplicidade na interface e rapidez no desenvolvimento. A estrutura do projeto foi pensada para garantir que o sistema fosse acessível a usuários com pouca experiência em tecnologia, oferecendo uma experiência intuitiva.

O projeto foi organizado em módulos principais que facilitam o desenvolvimento e a manutenção do código. Entre as funcionalidades implementadas, destacam-se:

- Cadastro de Proponentes: Interface para a inserção de novos proponentes no sistema, armazenando dados essenciais, como CPF, idade, órgão público associado e valor do contrato. Foi desenvolvido um método para formatação do CPF, garantindo que a entrada de dados esteja padronizada e facilitando o processo de validação.
- Consulta e Edição de Proponentes: Página que permite visualizar os proponentes cadastrados, com opções para editar dados existentes. A atualização é realizada de forma a refletir imediatamente no banco de dados e garantir que as informações estejam sempre atualizadas.
- Exclusão de Proponentes: Para facilitar a gestão, o sistema permite a exclusão de proponentes não desejados ou duplicados. A exclusão é confirmada com uma lista de todos os proponentes cadastrados, exibida em formato de tabela após a ação.
- Banco de Dados: O sistema utilizou o SQLite para armazenamento de dados nas fases iniciais do projeto, garantindo persistência das informações e permitindo consultas rápidas. A estrutura foi pensada para que o sistema fosse escalável e possibilitasse futura integração com outros bancos, e isso ocorreu na versão final, sendo utilizado o PostgreSQL.
- Organização do Código e Manutenção: Para manter a organização, as funcionalidades foram divididas em arquivos e módulos separados. Um exemplo é a separação entre o arquivo de controle de proponentes

(ProponenteController) e os modelos de dados, como o arquivo que define a classe Proponente. Com isso, o código está organizado e pronto para receber melhorias no futuro.

2.1 FUNCIONALIDADES DO SISTEMA

Cadastro e Atualização de Proponentes

O sistema permite que cada proponente faça seu pré-cadastro, registrando nome, idade, CPF, órgão público ao qual está associado e o valor proposto para o contrato. As informações podem ser atualizadas a qualquer momento, facilitando o processo de acompanhamento e eventual edição de dados.

Prospecção e Fechamento de Contrato

Proponentes são cadastrados com o objetivo de prospecção e acompanhamento até o fechamento do contrato associativo. O sistema permite visualização e consulta dos dados a qualquer momento, organizando o fluxo para que os responsáveis pela prospecção possam gerenciar as etapas e garantir que nenhum contato seja perdido.

Visualização e Exclusão de Proponentes

Com o intuito de facilitar a exclusão de registros desnecessários, como proponentes que perderam interesse, o sistema exibe uma tabela de dados logo após o botão de exclusão, permitindo uma visão clara e atualizada da base de dados. Essa tabela é atualizada a cada operação de exclusão.

Banco de Dados e Persistência

Utilizando PostgreSQL o sistema consegue manter as informações de forma permanente, o que é ideal para o tipo de aplicação proposta.

3 CONCLUSÃO

Desenvolver este sistema de gerenciamento de proponentes foi uma experiência muito enriquecedora e trouxe uma solução prática para um problema real dentro da Associação. Agora, temos uma ferramenta simples e direta para registrar, acompanhar e gerenciar os dados de pessoas interessadas em se associar. Com isso, o processo de prospecção se torna mais eficiente e organizado, o que facilita tanto o acompanhamento quanto a tomada de decisões.

A escolha de usar o PostgreSQL em vez do SQLite foi motivada pela necessidade de um banco de dados mais robusto, já que isso nos permite gerenciar melhor os dados

e ter mais flexibilidade para escalas maiores no futuro. Isso também abre portas para integrações mais complexas, se precisarmos expandir o sistema.

REFERÊNCIAS

BUG, C. C. Como criar um CRUD WEB em python | Streamlit #1. Disponível em: <https://www.youtube.com/watch?v=9mnNSMCu3dI&list=PLhna1crYw0SMogdpgsahknQUOmIrr74hY&index=3>. Acesso em: 6 nov. 2024.

BUG, C. C. Conectando ao banco de dados com python | Streamlit #2. Disponível em: <https://www.youtube.com/watch?v=lOyCICREgy8&list=PLhna1crYw0SMogdpgsahknQUOmIrr74hY&index=4>. Acesso em: 6 nov. 2024.

BUG, C. C. Consultando em Python os cadastros no banco de dados | Streamlit #3. Disponível em: <https://www.youtube.com/watch?v=J0KxfZEPrDI&list=PLhna1crYw0SMogdpgsahknQUOmIrr74hY&index=5>. Acesso em: 6 nov. 2024.

BUG, C. C. Como separar as páginas em arquivos Python com Streamlit #4. Disponível em: <https://www.youtube.com/watch?v=AqW_6hj1VmI&list=PLhna1crYw0SMogdpgsahknQUOmIrr74hY&index=6>. Acesso em: 6 nov. 2024.

BUG, C. C. Como excluir dados do banco com aplicativo WEB em Python #5. Disponível em: <https://www.youtube.com/watch?v=_lKn0rD0oFw&list=PLhna1crYw0SMogdpgsahknQUOmIrr74hY&index=8>. Acesso em: 6 nov. 2024.

BUG, C. C. Streamlit | Alterando dados no banco com Python #6. Disponível em: <https://www.youtube.com/watch?v=GNN819yeOT4&list=PLhna1crYw0SMogdpgsahknQUOmIrr74hY&index=9>. Acesso em: 6 nov. 2024.

BUG, C. C. Streamlit | Alterando dados no banco com Python #7. Disponível em: <https://www.youtube.com/watch?v=6yJgF3LuHU8&list=PLhna1crYw0SMogdpgsahknQUOmIrr74hY&index=10>. Acesso em: 6 nov. 2024.

BUG, C. C. Streamlit | Alterando dados no banco com Python #8. Disponível em: <https://www.youtube.com/watch?v=zP2mRzjPl-A&list=PLhna1crYw0SMogdpgsahknQUOmIrr74hY&index=11>. Acesso em: 6 nov. 2024.