Projeto: Sistema de Gerenciamento de Biblioteca

Descrição:

Desenvolva um sistema simples para gerenciar uma pequena biblioteca, onde será possível:
1. **Registrar novos livros** com título, autor, ano de publicação e número de páginas.
2. **Gerenciar membros** que podem pegar livros emprestados, incluindo nome, data de nascimento e número de telefone.
3. **Registrar e controlar empréstimos de livros**, onde cada empréstimo terá a data de retirada e a data esperada de devolução.
4. **Relatórios e consultas**: permitir que o usuário consulte:
   - Livros disponíveis.
   - Membros ativos com livros em mãos.
   - Histórico de empréstimos.

Requisitos Técnicos:

1. Estrutura do Banco de Dados (PostgreSQL)

- **Tabelas necessárias**:
  - `books`: ID, título, autor, ano de publicação, número de páginas.
  - `members`: ID, nome, data de nascimento, telefone.
  - `loans`: ID, ID do livro, ID do membro, data de retirada, data esperada de devolução, status (empréstimo ativo ou devolvido).

2. Funcionalidades do Sistema (Python + SQL)

- Implementar o backend em Python usando a biblioteca `psycopg2` para realizar as operações com o banco de dados.
- Separar as responsabilidades do código, utilizando o princípio de funções reutilizáveis e organização clara do projeto.

Funcionalidades Específicas:

1. **Conexão com o Banco de Dados**: Criar uma classe ou função para gerenciar a conexão com o PostgreSQL.
2. **CRUD para Livros**: Funções para criar, ler, atualizar e excluir informações de livros.
3. **CRUD para Membros**: Funções para adicionar novos membros, consultar e gerenciar.
4. **Registro de Empréstimos**: Função para registrar e finalizar empréstimos de livros.
5. **Consultas e Relatórios**:

- Consultar todos os livros disponíveis.
- Listar todos os membros com empréstimos ativos.
- Histórico de todos os empréstimos de um determinado membro.

3. Boas Práticas de Código

- **Código Limpo**: Utilizar boas práticas de programação como nomes de variáveis claros, modularização, e uso de docstrings para documentação.
- **Tratamento de Exceções**: Garantir que o código lide adequadamente com possíveis erros, como falha de conexão ao banco de dados ou tentativas de empréstimos duplicados.
- **Bibliotecas**: Além de `psycopg2`, o uso de outras bibliotecas que possam melhorar a legibilidade e manutenção do código é encorajado (ex: `pandas` para relatórios, `argparse` para criar interfaces de linha de comando).

Avaliação

A avaliação será baseada nos seguintes critérios:
1. **Corretude Técnica**: O sistema atende a todos os requisitos funcionais descritos.
2. **Boas Práticas**: O código segue os princípios de código limpo, com modularização e documentação adequada.
3. **Criatividade**: A pessoa pode adicionar funcionalidades extras ou melhorar o design do sistema, como implementar um sistema de notificações por e-mail para lembrar os membros de devolver os livros.
4. **Uso Eficiente do SQL**: Consultas SQL otimizadas e uso adequado do banco de dados relacional.
5. **Tratamento de Exceções**: Como o sistema lida com falhas comuns, como entradas inválidas ou erros de conexão.

Funcionalidades Extras (Opcional)

Para quem quiser adicionar algo extra ao projeto:
- Implementar uma interface de linha de comando ou até mesmo uma interface gráfica simples com `Tkinter`.
- Adicionar um sistema de pesquisa mais avançado, permitindo que os usuários busquem livros por diferentes critérios (autor, data, etc.).
- Criar um sistema de categorias de livros, onde os livros são divididos em diferentes gêneros.

Entrega

O projeto deve ser entregue com:
- O código-fonte completo (organizado em módulos).
- Um arquivo `README.md` explicando como rodar o projeto, suas dependências e uma breve descrição das funcionalidades implementadas.
- Um arquivo SQL com o script de criação das tabelas.