**Apostila de Estudo: Sistema de Biblioteca em Python**

**Objetivo**

Este material serve como guia de estudo e referência para o projeto de um sistema de gerenciamento de biblioteca em Python. A proposta envolve uso de módulos, leitura e escrita em arquivos JSON, lógica de controle de fluxo, além da construção de uma interface gráfica com Tkinter.

**1. Estrutura do Projeto**

O sistema foi dividido em quatro módulos principais:

1. app.py - Arquivo principal que exibe o menu e chama os outros módulos.
2. livros.py - Responsável por cadastrar e listar livros.
3. cliente.py - Responsável por cadastrar clientes.
4. emprestimo\_devolucao.py - Responsável por realizar e devolver empréstimos.

Os dados são salvos em arquivos .json dentro da pasta data/:

* livros.json
* cliente.json
* emprestimos.json

**2. Manipulação de Arquivos JSON**

**Bibliotecas usadas**

import json

import os

**Leitura de arquivos JSON**

if os.path.exists(caminho\_arquivo):

with open(caminho\_arquivo, "r", encoding="utf-8") as f:

dados = json.load(f)

**Escrita de arquivos JSON**

with open(caminho\_arquivo, "w", encoding="utf-8") as f:

json.dump(dados, f, indent=4, ensure\_ascii=False)

indent=4: deixa o arquivo mais legível.  
ensure\_ascii=False: permite acentos e caracteres especiais.

**3. Controle de Fluxo com Menu**

Exemplo do menu principal em app.py:

while True:

print("[1] Cadastrar livro")

opcao = input("Escolha: ")

if opcao == "1":

livros.menu\_livros()

elif opcao == "0":

break

Funções específicas para cada opção são chamadas com base na escolha do usuário.

**4. Cadastro de Livros**

**Estrutura do livro**

{

"nome\_livro": "Nome",

"isbn": "123456",

"autor": "Autor",

"categoria": "Categoria",

"disponivel": True

}

**Cadastro**

def cadastrar\_livro(nome, isbn, autor, categoria):

livro = { ... }

lista\_livros.append(livro)

salvar\_livros()

**5. Cadastro de Clientes**

**Estrutura do cliente**

{

"id\_cliente": "001",

"nome\_cliente": "Nome",

"telefone": "12345678",

"email": "email@dominio",

"cpf": "000.000.000-00",

"endereco": "Rua Tal",

"cidade": "Cidade"

}

**Cadastro**

def cadastrar\_cliente(...):

cliente = { ... }

lista\_clientes.append(cliente)

salvar\_usuarios()

**6. Empréstimo e Devolução**

**Empréstimo**

def menu\_emprestimo():

isbn = input(...)

for livro in lista\_livros:

if livro["isbn"] == isbn and livro["disponivel"]:

livro["disponivel"] = False

lista\_emprestimos.append({"isbn": isbn, "cliente\_id": id})

salvar\_emprestimos()

**Devolução**

def menu\_devolucao():

isbn = input(...)

for livro in lista\_livros:

if livro["isbn"] == isbn and not livro["disponivel"]:

livro["disponivel"] = True

salvar\_emprestimos()

**7. Interface Gráfica com Tkinter**

A interface foi construída usando o módulo tkinter:

**Exemplo básico de janela**

import tkinter as tk

root = tk.Tk()

root.title("Sistema Biblioteca")

root.mainloop()

**Campos de entrada**

entry\_nome = tk.Entry(root)

entry\_nome.pack()

**Botões**

tk.Button(root, text="Cadastrar", command=função).pack()

**Listbox para mostrar livros**

listbox = tk.Listbox(root)

listbox.insert(tk.END, "Livro 1")

**8. Dicas e Boas Práticas**

* Sempre salve os dados após cada cadastro ou alteração.
* Use funções pequenas e bem definidas.
* Separe responsabilidades em módulos.
* Comente seu código.
* Sempre verifique se o arquivo existe antes de tentar abri-lo.

**9. Próximos Passos**

* Adicionar validações (CPF, e-mail, etc.)
* Implementar busca por livros ou clientes
* Adicionar datas de empréstimo e devolução
* Exportar relatórios (CSV, PDF)

**10. Conclusão**

Este projeto é uma excelente base para entender como criar sistemas com persistência de dados, interface interativa, e estrutura modular. Estude cada módulo, modifique o projeto, e experimente novas funcionalidades para reforçar seu aprendizado.

**Bom estudo e bons códigos!**