ANDERSON DE MATOS GUIMARÃES

GERENCIADOR DE TAREFAS

Brasília – DF
2024

SERVIÇO NACIONAL DE APRENDIZADO COMERCIAL - SENAC

COORDENAÇÃO DE CIÊNCIA DE DADOS

ANDERSON DE MATOS GUIMARÃES

GERENCIADOR DE TAREFAS

Projeto em Linguagem de Programação Python apresentado à disciplina Linguagem Técnica de Programação da Faculdade de Tecnologia e Inovação Senac DF sob orientação do professor Esp. Clenio Emidio da Fonseca para obtenção de nota no segundo bimestre.

Brasília – DF

2024

ANDERSON DE MATOS GUIMARÃES

GERENCIADOR DE TAREFAS

Projeto em Linguagem de Programação Python apresentado à disciplina Linguagem Técnica de Programação da Faculdade de Tecnologia e Inovação Senac DF sob orientação do professor Esp. Clenio Emidio da Fonseca para obtenção de nota no segundo bimestre

Brasília, 22 de novembro de 2024.

BANCA EXAMINADORA

Prof. Esp. Clenio Emídio da Fonseca
Faculdade de Tecnologia e Inovação Senac DF

“Na medida em que a complexidade aumenta, precisamos de ferramentas mais poderosas para ajudar a gerenciar essa complexidade."
(Guido van Rossum, criador da linguagem Python)

1 GERENCIADOR DE TAREFAS

1.1 Visão Geral

O Gerenciador de Tarefas é um aplicativo desenvolvido em Python que tem como objetivo principal auxiliar os usuários a organizarem e gerenciarem suas tarefas de forma eficiente. Utilizando a biblioteca pandas para manipulação e armazenamento de dados e a biblioteca FPDF para geração de relatórios em PDF, o aplicativo oferece uma interface interativa para a realização de operações de CRUD (*Create, Read, Update, Delete*).

1.2 Funcionalidades

1.2.1 Adicionar Tarefa (Create):

Permite ao usuário adicionar novas tarefas ao sistema, incluindo uma descrição e uma prioridade (alta, média, baixa).
As tarefas são armazenadas em um DataFrame do Pandas para fácil manipulação e acesso.

1.2.2 Exibir Tarefas (Read):

Exibe todas as tarefas atualmente armazenadas, listando a descrição e a prioridade de cada uma.
Utiliza a funcionalidade de impressão do Pandas para mostrar os dados em formato tabular, tornando a visualização clara e organizada.

1.2.3 Atualizar Tarefa (Update):

Permite ao usuário atualizar as informações de uma tarefa existente, modificando tanto a descrição quanto a prioridade.
Garante que as alterações sejam refletidas imediatamente no DataFrame.

1.2.4 Remover Tarefa (Delete):

Oferece a funcionalidade de remover uma tarefa específica com base no índice, proporcionando uma maneira simples de manter o sistema organizado e atualizado.

1.2.5 Gerar Relatório em PDF:

Utiliza a biblioteca FPDF para gerar um relatório em PDF das tarefas atuais, incluindo suas descrições e prioridades.
O relatório é formatado de maneira a ser facilmente compreendido e pode ser salvo para futuras referências ou compartilhamento.

1.3 Tecnologias Utilizadas

1.3.1 Python

Linguagem de programação utilizada para o desenvolvimento do aplicativo.

1.4 Pandas

Biblioteca utilizada para manipulação de dados tabulares, facilitando operações de CRUD e armazenamento de tarefas.

1.4.1 FPDF

Biblioteca utilizada para geração de relatórios em PDF, permitindo que os usuários exportem suas tarefas para um formato de documento portátil.

1.5 Estrutura do Código

O código é organizado de forma modular, com funções dedicadas para cada operação (adicionar, exibir, atualizar, remover e gerar relatório). A função principal (main) coordena a interação do usuário com o aplicativo através de um menu interativo, facilitando a navegação e utilização das funcionalidades.

1.6 Exemplo de Uso

1.6.1 Adicionar Tarefa

O usuário escolhe a opção "Adicionar tarefa" no menu.
Insere a descrição e a prioridade da nova tarefa.
A tarefa é adicionada ao DataFrame e confirmada ao usuário.

1.6.2 Exibir Tarefas

O usuário escolhe a opção "Exibir tarefas" no menu.
Todas as tarefas são exibidas no terminal em formato tabular.

1.6.3 Atualizar Tarefa

O usuário escolhe a opção "Atualizar tarefa" no menu.
Insere o índice da tarefa a ser atualizada e as novas informações.
A tarefa é atualizada no DataFrame.

1.6.4 Remover Tarefa

O usuário escolhe a opção "Remover tarefa" no menu.
Insere o índice da tarefa a ser removida.
A tarefa é removida do DataFrame.

1.6.5 Gerar Relatório

O usuário escolhe a opção "Gerar relatório" no menu.
Um relatório em PDF é gerado e salvo no diretório do aplicativo.

2 CÓDIGO

Abaixo segue o código

```
import pandas as pd
from fpdf import FPDF

# Função para adicionar uma tarefa (CREATE)
def adicionar_tarefa(tarefas):
    descricao = input("Digite a descrição da tarefa: ")
    prioridade = input("Digite a prioridade da tarefa (alta, média, baixa): ")
    nova_tarefa = {'Descrição': descricao, 'Prioridade': prioridade}
    tarefas = pd.concat([tarefas, pd.DataFrame([nova_tarefa])],
ignore_index=True)
    return tarefas

# Função para exibir todas as tarefas (READ)
def exibir_tarefas(tarefas):
    print("\nTarefas:")
    print(tarefas)

# Função para atualizar uma tarefa (UPDATE)
def atualizar_tarefa(tarefas):
    indice = int(input("Digite o índice da tarefa a ser atualizada: "))
    if 0 <= indice < len(tarefas):
        nova_descricao = input("Digite a nova descrição da tarefa: ")
        nova_prioridade = input("Digite a nova prioridade da tarefa (alta,
média, baixa): ")
        tarefas.loc[indice, 'Descrição'] = nova_descricao
        tarefas.loc[indice, 'Prioridade'] = nova_prioridade
    else:
        print("Índice inválido.")
    return tarefas

# Função para remover uma tarefa (DELETE)
def remover_tarefa(tarefas):
    indice = int(input("Digite o índice da tarefa a ser removida: "))
    if 0 <= indice < len(tarefas):
        tarefas = tarefas.drop(indice).reset_index(drop=True)
    else:
        print("Índice inválido.")
    return tarefas

# Função para gerar um relatório das tarefas em PDF
def gerar_relatorio(tarefas):
    pdf = FPDF()
    pdf.add_page()
```

```
    pdf.set_font("Arial", size=12)

    pdf.cell(200, 10, txt="Relatório de Tarefas", ln=True, align="C")
    pdf.cell(200, 10, txt="----------------------", ln=True, align="C")

    for index, row in tarefas.iterrows():
        pdf.cell(200, 10, txt=f"Tarefa {index}: {row['Descrição']}
(Prioridade: {row['Prioridade']})", ln=True)

    pdf.output("relatorio_tarefas.pdf")
    print("Relatório gerado em 'relatorio_tarefas.pdf'.")

# Função para exibir o menu
def exibir_menu():
    print("\n" + "="*50)
    print(f"{'Menu de Gerenciamento de Tarefas':^50}")
    print("="*50)
    print(f"{'1. Adicionar tarefa'}")
    print(f"{'2. Exibir tarefas'}")
    print(f"{'3. Atualizar tarefa'}")
    print(f"{'4. Remover tarefa'}")
    print(f"{'5. Gerar relatório'}")
    print(f"{'6. Sair'}")
    print("="*50)

# Função principal
def main():
    tarefas = pd.DataFrame(columns=['Descrição', 'Prioridade'])

    while True:
        exibir_menu()
        escolha = input("Escolha uma opção: ")

        if escolha == '1':
            tarefas = adicionar_tarefa(tarefas)
        elif escolha == '2':
            exibir_tarefas(tarefas)
        elif escolha == '3':
            tarefas = atualizar_tarefa(tarefas)
        elif escolha == '4':
            tarefas = remover_tarefa(tarefas)
        elif escolha == '5':
            gerar_relatorio(tarefas)
        elif escolha == '6':
            print("Saindo...")
            break
        else:
            print("Opção inválida. Tente novamente.")
```

```
if __name__ == "__main__":
    main()
```

3 CONCLUSÃO

O Gerenciador de Tarefas é uma ferramenta prática e eficiente para o gerenciamento de tarefas diárias, oferecendo uma interface amigável e funcionalidades robustas para garantir que as tarefas sejam organizadas, atualizadas e acessíveis de forma simples e rápida. A integração com pandas e FPDF permite uma gestão eficaz dos dados e a possibilidade de geração de relatórios em PDF, tornando o aplicativo uma solução completa para usuários que buscam melhorar sua produtividade e organização.