Terminal Activity

DOCUMENTAÇÃO DO PROJETO

02 // Resumo

03 // Estrutura de classes

04 // Persistência de Dados

07 // Tabelas e diagramas

09 // Encerramento

PRODUCTIVITY APP
// **V1.0**

ALUNO: ARTHUR NOGUERA RAMALHO
MATRICULA:231011103

SEMESTRE:2024.2
DATA: 05/01/25

Documentação do Aplicativo

Gerenciador de Atividades

Resumo do Aplicativo

O Terminal Activity é uma aplicação de gestão de atividades interativa que permite a criação, modificação e exclusão de tarefas e hábitos.

Ele suporta múltiplos usuários, permitindo que cada um gerencie suas próprias atividades de maneira independente.

A interface oferece menus de fácil navegação e suporte à verificação automática da validade de tarefas e hábitos.

Finalidade

A finalidade do aplicativo é ajudar os usuários a organizarem suas atividades diárias, mantendo um controle eficiente de suas tarefas com prazo e hábitos recorrentes. Além disso, o aplicativo:

- Permite a personalização do perfil de cada usuário.
- Garante o salvamento persistente de dados em arquivos JSON.
- Suporta a verificação automática de prazos expirados e a reinicialização de hábitos com base em suas periodicidades.

Estrutura de Classes

1. Classe Usuario

Finalidade: Gerenciar os usuários do sistema.
Principais métodos:

- __init__(self, nome, id, senha, bio, data_criacao): Construtor da classe.
- salvar_usuario(): Salva o usuário em um arquivo JSON.
- carregar_usuarios(): Carrega todos os usuários do arquivo JSON.
- listar_usuarios(): Lista os usuários cadastrados no sistema.
- criar_usuario(): Permite criar um novo usuário.
- remover_usuario(nome): Remove um usuário pelo nome.
- modificar_usuario(usuario_logado): Modifica os dados de um usuário existente.

2. Classe Atividade

Finalidade: Base para gerenciar atividades (tarefas e hábitos).
Principais métodos:

- __init__(self, nome, descricao, data_criacao, id): Construtor da classe.
- salvar_atividade(): Salva uma atividade no arquivo JSON.
- carregar_atividades(): Carrega todas as atividades do arquivo JSON.
- remover_atividade(nome, id): Remove uma atividade com base no nome e no ID do usuário.
- listar_atividades(id): Lista as atividades associadas a um usuário.
- modificar(id): Permite modificar uma atividade existente.

3. Classe Tarefa

Finalidade: Extensão de Atividade, representa atividades com prazo definido.
Construtor específico:

- __init__(self, nome, descricao, data_criacao, data_limite, id)

Método adicional:

- criar_tarefa(id): Cria uma nova tarefa com validação de data limite.

4. Classe Habito

Finalidade: Extensão de Atividade, representa hábitos com periodicidade.
Construtor específico:

- __init__(self, nome, descricao, data_criacao, periodo_repeticao, id)

Método adicional:

- criar_habito(id): Cria um novo hábito com validação de periodicidade.

Persistência de Dados

Os dados são armazenados em dois arquivos JSON:

1. Usuários: arq_usuarios_app_v1.0.json

- Contém informações de cada usuários, como nome, ID, senha, biografia e lista de atividades.

A estrutura básica é a seguinte:

```
[
  {
    "nome": "ExemploUsuario",
    "id": 123,
    "senha": "senha123",
    "bio": "Sou um exemplo.",
    "data_criacao": "01/01/25",
    "atividades": []
  }
]
```

2. Atividades: arq_atividades_app_v1.0.json

- Armazena todas as atividades, incluindo tarefas e hábitos, associadas aos IDs dos usuários.

2. Atividades: arq_atividades_app_v1.0.json

- Armazena todas as atividades, incluindo tarefas e hábitos, associadas aos IDs dos usuários.

A estrutura básica é a seguinte:

```
[
  "ultima_verificacao":
  "01/01/25", "atividades": [
   {
     "nome": "Estudar Python",
     "descricao": "Ler o capítulo sobre persistência de dados.",
     "data_criacao": "01/01/25",
     "concluido": false,
     "data_conclusao": null,
     "id": 123,
     "data_limite": "05/01/25",
     "atrasada": false},
   {
     "nome": "Exercício Diário",
     "descricao": "Praticar 30 minutos de exercícios.",
     "data_criacao": "01/01/25",
     "concluido": false,
     "data_conclusao": null,
     "id": 123,
     "periodo_repeticao": "diário"
   }
  ]
}
```

Fluxo de Verificação de Dados

Inicialização

Sempre que o programa é iniciado, o arquivo de atividades é verificado e inicializado, se necessário:

- Função inicializar_arquivo_tarefas: Cria o arquivo arq_atividades_app_v1.0.json com a estrutura básica, caso ele não exista ou esteja com estrutura inválida.

Salvamento

Sempre que o programa é iniciado, o arquivo de atividades é verificado e inicializado, se necessário:

- Função inicializar_arquivo_tarefas: Cria o arquivo arq_atividades_app_v1.0.json com a estrutura básica, caso ele não exista ou esteja com estrutura inválida.

Atualização Automática

A cada execução do método verificar_validade:

- Tarefas: Verifica a data limite para marcar como atrasada, se aplicável.
- Hábitos: Reinicia o status de conclusão com base no período de repetição (diário, semanal ou mensal).

Vantagens da Abordagem Utilizada

- Persistência Simples: O uso de JSON facilita a implementação e o armazenamento de dados de forma estruturada.
- Modularidade: Métodos específicos para salvar, carregar, remover e verificar dados tornam o sistema mais fácil de manter.
- Flexibilidade: Os dados podem ser facilmente transferidos ou analisados externamente, já que JSON é um formato padrão.

Tabelas e diagramas

Diagrama de classes

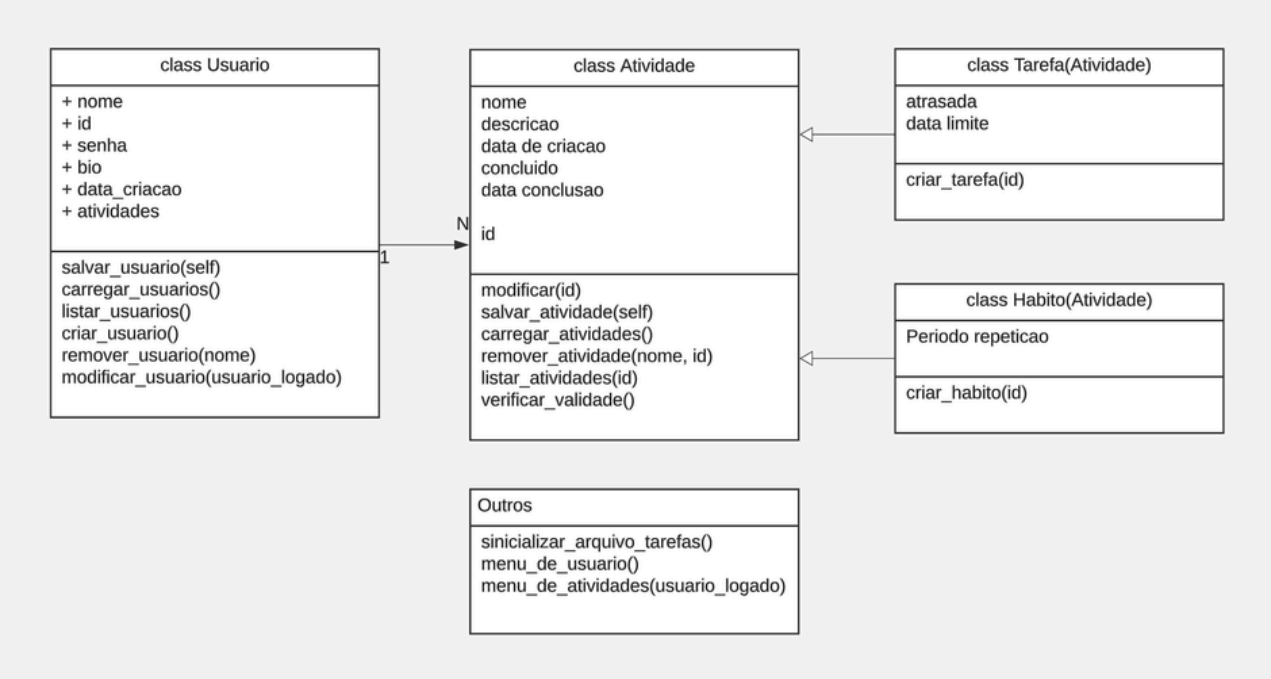
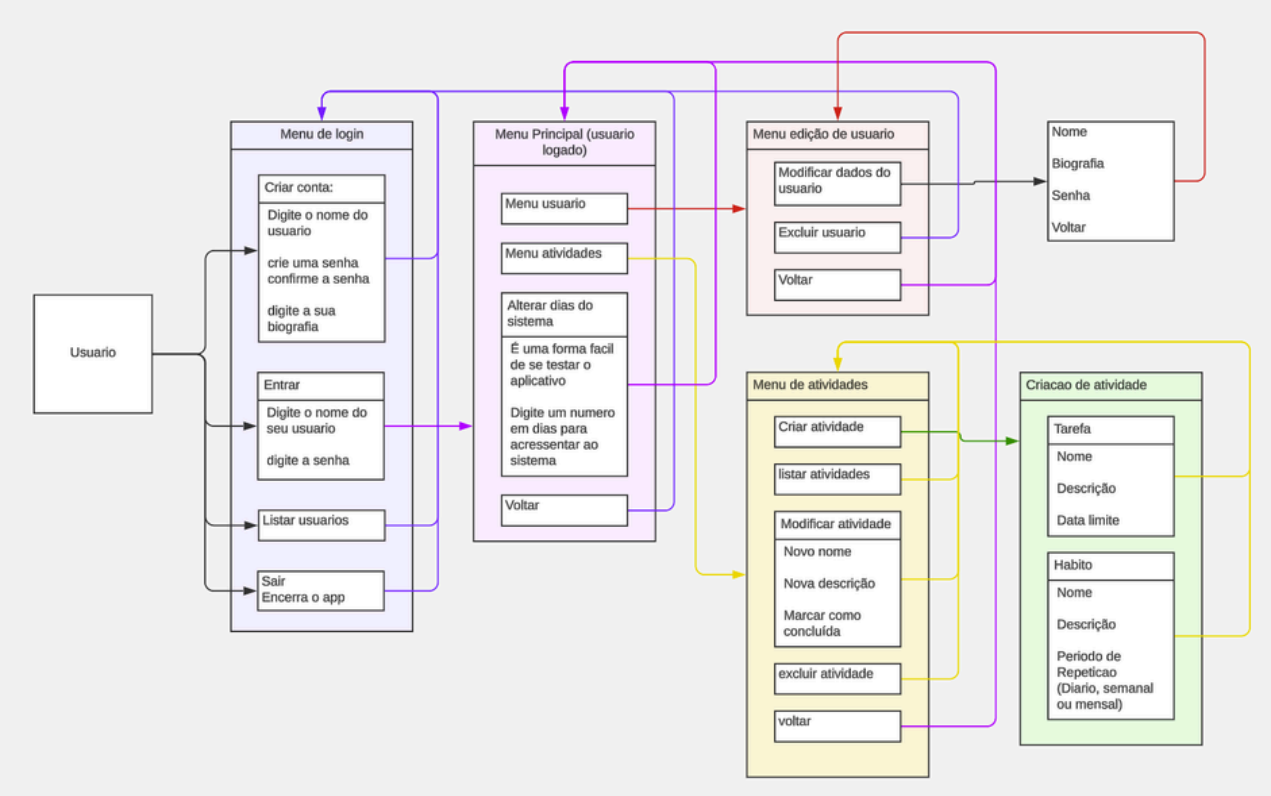


Tabela de construtores

Classe	Construtor	Descrição
Usuario	__init__(self, nome, id, senha, bio, data_criacao)	Inicializa um usuário com suas informações básicas.
Atividade	__init__(self, nome, descricao, data_criacao, id)	Inicializa uma atividade com nome, descrição e data de criação.
Tarefa	__init__(self, nome, descricao, data_criacao, data_limite, id)	Inicializa uma tarefa com prazo.
Habito	__init__(self, nome, descricao, data_criacao, periodo_repeticao, id)	Inicializa um hábito com periodicidade.

Diagrama de caso de uso



Encerramento

O aplicativo foi desenvolvido em ambiente Python, usando o compilador Visual Studios Code e com auxilio do Chat GPT, postado no *"Github"* para a avaliação como "projeto livre" na disciplina *Orientação à Objetos*, ministrada pelo professor Henrique Gomes Moura na *Universidade de Brasília*, *Campus UnB Gama: Faculdade de Ciências e Tecnologia em Engenharia*.

ALUNO: ARTHUR NOGUERA RAMALHO
MATRICULA:231011103
SEMESTRE:2024.2
DATA: 05/01/25