

projeto Restaurant Orders!

► O que foi desenvolvido






Um primeiro time iniciou o desenvolvimento deste projeto e já preparou uma estrutura inicial e fiquei responsável por construir testes para classes já implementadas, implementar uma nova classe para mapear os pratos e suas respectivas receitas (ingredientes e quantidades), também implementar uma classe que gerou os cardápios que devem ser mostrados para as pessoas que frequentam o estabelecimento e outra que fará a gestão de estoque dos ingredientes.

► Habilidades

Neste projeto:

- Praticar o conceito de Hashmaps através das estruturas de dados Dict e Set do Python;
- Praticar os conhecimentos de testes de software;
- Praticar os conhecimentos de orientação a objetos.
- Testando classes já implementadas
- Mapeamento Pratos <> Ingredientes
- Geração dos cardápios
- Estoque de ingredientes

► Orientações

-  `sudo apt update`
-  `sudo apt install python3.10-venv`
-  Crie e ative o ambiente virtual para o projeto: `python3 -m venv .venv && source .venv/bin/activate`
-  Instale as dependências: `python3 -m pip install -r dev-requirements.txt`
-  Linter: `python3 -m flake8`
-  testes: `python3 -m pytest`

PROF

1 - Testando classes já implementadas parte 1

Implementei testes para a classe `Ingredient`, que se encontra no módulo `src/models/ingredient.py`.

Passos Concluídos:

1. Criação dos Testes:

- Testei a instância da classe `Ingredient` para garantir que pode ser criada com os parâmetros corretos.
- Verifiquei se o atributo `restrictions` é corretamente populado.
- Testei o método mágico `__repr__` para assegurar que a representação string do objeto é adequada.
- Testei o método mágico `__eq__` para validar que a comparação de igualdade entre dois ingredientes funciona corretamente.

- Testei o método mágico `__hash__` para confirmar que ingredientes iguais possuem o mesmo hash e ingredientes diferentes possuem hashes distintos.

2. Testes Implementados:

- Validei a criação e inicialização da classe.
- Chequei a funcionalidade dos métodos mágicos.
- Assegurei que os atributos e métodos se comportam como esperado.

►  **Clique aqui para ver o que foi validado pelo avaliador.**

- Hashes de ingredientes iguais e diferentes.
- Comparação de igualdade entre ingredientes.
- Representação string do ingrediente.
- Atributos `name` e `restrictions`.

2 - Testando classes já implementadas parte 2

Implementei testes para a classe `Dish`, que se encontra no módulo `src/models/dish.py`.

Passos Concluídos:

1. Criação dos Testes:

- Testei a instância da classe `Dish` para assegurar a correta criação com os parâmetros esperados.
- Verifiquei a funcionalidade dos métodos `add_ingredient_dependency`, `get_restrictions`, e `get_ingredients`.
- Assegurei que o dicionário de receita do prato devolve a quantidade correta de um ingrediente.
- Validei que erros são levantados ao tentar criar pratos inválidos.

2. Testes Implementados:

- Validei a criação e inicialização da classe.
- Chequei a funcionalidade dos métodos e atributos.
- Assegurei que exceções são corretamente levantadas para dados inválidos.

►  **Clique aqui para ver o que foi validado pelo avaliador.**

- Atributos e métodos da classe `Dish`.
- Hashes e comparações de igualdade.
- Erros ao criar pratos inválidos.
- Quantidade de ingredientes e restrições.

3 - Mapeamento Pratos <> Ingredientes

Implementei a classe `MenuData`, que faz o mapeamento de pratos e ingredientes baseado no arquivo CSV disponibilizado. Ela se encontra no módulo `src/services/menu_data.py`.

Passos Concluídos:

1. Implementação da Classe:

- Criei a classe `MenuData` para ler o arquivo CSV e instanciar os pratos e ingredientes conforme o conteúdo.
- O atributo `dishes` é um `set` que lista todos os pratos corretamente.

2. Leitura do Arquivo CSV:

- Implementei a leitura do arquivo CSV e o mapeamento das receitas com ingredientes e quantidades.

3. Validação dos Dados:

- Assegurei que o `set dishes` contém os pratos e ingredientes corretos.

►  **Clique aqui para ver o que foi validado pelo avaliador.**

- Tamanho e conteúdo do `set dishes`.
- Quantidade de ingredientes e suas respectivas quantidades.

4 - Geração dos cardápios

Implementei o método `get_main_menu` na classe `MenuBuilder` que se encontra no arquivo `src/services/menu_builder.py`.

Passos Concluídos:

1. Implementação do Método:

- Criei o método `get_main_menu` para retornar uma lista de dicionários contendo o cardápio completo.
- O método pode receber uma restrição alimentar opcional e filtra o cardápio conforme necessário.

2. Validação dos Resultados:

- Testei a geração do cardápio sem restrições e com restrições específicas para garantir a correta filtragem dos pratos.

►  **Clique aqui para ver o que foi validado pelo avaliador.**

- Tipo e formato da lista retornada.
- Inclusão e exclusão de pratos com base nas restrições alimentares.

Certifique-se de executar todos os testes usando `pytest` para garantir que todas as funcionalidades estejam corretas e os requisitos atendidos.