Documentação do Microserviço de Cardápios - OpenRu

1. Descrição do Projeto

O microserviço de Cardápios faz parte do projeto OpenRu, um sistema distribuído para gestão de Restaurantes Universitários. Este serviço é responsável por gerenciar os cardápios diários do restaurante, permitindo criar, visualizar, atualizar e excluir cardápios, além de consultar o cardápio do dia.

2. Arquitetura e Estrutura do Projeto

A organização do projeto segue uma arquitetura baseada em Flask, com separação de responsabilidades em módulos:

- app/models: Definição dos modelos do banco de dados (Menu).
- app/schemas: Definição dos schemas de validação e serialização (Marshmallow).
- app/routes: Definição das rotas/endpoints da API.
- app/config.py: Configuração de ambiente e banco de dados.
- migrations/: Controle de versão do banco de dados via Alembic.
- wsgi.py: Ponto de entrada para execução do servidor.

3. Requisitos

Os requisitos estão definidos em requirements.txt. Principais dependências:

- Flask
- Flask-SQLAlchemy
- Flask-Migrate
- Marshmallow
- Alembic
- Psycopg3

4. Configuração do Ambiente

1. Criar ambiente virtual Python.
2. Instalar dependências com: pip install -r requirements.txt
3. Configurar variáveis de ambiente no arquivo .env.
4. Executar as migrações: flask db upgrade
5. Rodar o servidor com: flask run ou via wsgi.py

5. Modelo de Dados

O modelo principal é o Menu, que contém os seguintes atributos:
- id: Identificador único do cardápio.
- date: Data do cardápio.
- main_course: Prato principal.
- side_dish: Guarnição.
- salad: Salada.
- dessert: Sobremesa.
- beverage: Bebida.

6. Endpoints Disponíveis

Rotas principais definidas em app/routes/menu_routes.py:

GET /menus -> Lista todos os cardápios.
GET /menus/<id> -> Retorna um cardápio específico.
POST /menus -> Cria um novo cardápio.
PUT /menus/<id> -> Atualiza um cardápio existente.
DELETE /menus/<id> -> Remove um cardápio.
GET /menus/today -> Consulta o cardápio do dia.

7. Banco de Dados e Migrações

O serviço utiliza SQLAlchemy para ORM e Alembic para migrações. O diretório migrations/ contém as versões e scripts gerados para controle da estrutura do banco.

8. Execução

Para rodar localmente:
1. Configurar .env com parâmetros corretos..
2. Executar migrações.
3. Iniciar o servidor Flask.

Para rodar em produção, utilizar wsgi.py com um servidor compatível (Gunicorn).