V4 Challenge

🚀 Tecnologias

Esse projeto foi desenvolvido com as seguintes tecnologias:

- Node.js
- PostgreSQL
- Docker
- Docker Compose

Extras:

- Main Libs
  - Express
  - TypeORM
  - JestJS
  - CSV Parser
  - Axios
  - Node Cron

💻 Projeto

O **V4 Challenge** é um projeto que visa fornecer e cadastrar produtos alimentícios.

> This is a challenge by Coodesh

🤔 Como contribuir

- Faça um fork desse repositório;
- Cria uma branch com a sua feature: `git checkout -b minha-feature`;
- Faça commit das suas alterações: `git commit -m 'feat: Minha nova feature'`;
- Faça push para a sua branch: `git push origin minha-feature`.

Depois que o merge da sua pull request for feito, você pode deletar a sua branch.

💾 Testar a aplicação

Para fazer o teste da aplicação, é necessário ter instalado na sua máquina:

- Docker

Variáveis de ambiente

Antes de subir a instância de teste, é necessário configurar em sua máquina as seguintes váriaveis de ambiente para que aplicação possa subir:

Váriavel	Sistema	Descrição
PORT	Aplicação	Endpoint do Pi System
PG_HOST	PostgreSQL (Conexão)	Host do banco de dados
PG_PORT	PostgreSQL (Conexão)	Porta do banco de dados
PG_USERNAME	PostgreSQL (Conexão)	Username do banco de dados
PG_PASSWORD	PostgreSQL (Conexão)	Senha do banco de dados
CRONJOB_URL	Job/CSV	URL do job de importação de CSV
CRONJOB_REGEX	Job/CSV	Regex de período de execução (Ex.: '* 57 22 * * * *')
POSTGRES_USER	PostgreSQL (Banco)	Username PostgreSQL
POSTGRES_PASSWORD	PostgreSQL (Banco)	Password PostgreSQL
POSTGRES_DB	PostgreSQL (Banco)	Database PostgreSQL

Instalação

Para subir a aplicação, basta rodar o comando a seguir no terminal:

```
docker compose up --build -d
```

Caso queria parar a aplicação, rode o seguinte comando no terminal:

```
docker compose down
```

Acessar o swagger

Caso queira ver o swagger da integração reidenizado, basta, após a subida da aplicação, acessar o recurso /api-docs no navegador, que carregará uma página com a documentação da API.

Template da chamada

```
http://{host}:{port}/api-docs
```

Exemplo da chamada

```
http://localhost:3333/api-docs
```

Health check

Caso queira realizar a configuração de health check no ambiente de deploy, basta chamar o recurso /healthz.

Template da chamada

```
http://{host}:{port}/healthz
```

Exemplo da chamada

```
http://localhost:3333/healthz
```

Execução de teste unitários e integração

Caso queira fazer a execução dos testes unitários para implementar na esteira CI/CD para validar antes do deploy, é necessário ter instalado na sua máquina:

- NodeJS
- PostgreSQL

Após configurar as variáveis de ambiente na sua máquina, execute os comandos a seguir:

```
yarn install
yarn test
```

Cron Job

Para fazer a configuração correta do Node Cron, segue como realizar o regex:

```
# ┌────────────── segundo (opcional)
# │ ┌──────────── minuto
# │ │ ┌────────── hora
# │ │ │ ┌──────── dia do mês
# │ │ │ │ ┌────── mês
# │ │ │ │ │ ┌──── dia da semana
# │ │ │ │ │ │
# │ │ │ │ │ │
# * * * * * *
```

Campo	Valor
segundo	0-59
minuto	0-59
hora	0-23
dia do mês	1-31
mês	1-12 (or names)
dia da semana	0-7 (or names, 0 or 7 are sunday)

Mais detalhes da documentação se encontram em Node Cron