Projeto Pyhton

Esta documentação fornece os comandos necessários para acessar e executar o projeto de forma correta. Os passos descritos abaixo garantem uma configuração adequada do ambiente, clonagem do repositório, criação e ativação do ambiente virtual, e execução do servidor. Certifique-se de seguir estas instruções para uma experiência de desenvolvimento suave e bem-sucedida.

Introdução

Bem-vindo ao Projeto! Este é um projeto que foi desenvolvido em Python e que tem como objetivo gerar códigos de barras através da biblioteca barcode. Ele inclui um servidor simples com rotas relacionadas a tags, bem como a implementação de um validador. O projeto utiliza um ambiente virtual para gerenciar suas dependências.

1- Clone o repositório

git clone https://github.com/seu-usuario/Projeto1.git

cd Projeto1

2- Criação e ativação do ambiente virtual

python -m venv .venv

source .venv/bin/activate # No Linux/Mac

.\.venv\Scripts\activate # No Windows

3- Instale as Dependências:

pip install -r requirements.txt

Execução do Projeto

Agora que o ambiente está configurado, você pode executar o projeto com os seguintes comandos:

1- Inicie o Servidor

python run.py – **no terminal**

o servidor está disponível em: `http://localhost:5000`.

Recomendamos a utilização da ferramenta Postman para interagir com as rotas do servidor. O Postman oferece uma interface amigável para testar e validar as requisições HTTP.

2- Testes Automatizados

Para garantir a qualidade do código, são fornecidos testes automatizados**.**

pytest -s -v **-no terminal**

Trabalhando com Alterações Locais com GIT no terminal

- **Verificar o Status do Repositório Local:**

git status

- **Adicionar Alterações ao Stage:**

git add .

Ou, para adicionar arquivos específicos

git add nome-do-arquivo

- **Realizar um Commit:**

git commit -m "Mensagem do commit"

Bibliotecas utilizadas no projeto:

Estas bibliotecas estão listadas no arquivo requirements.txt:

- **astroid==3.0.3:**

Astroid é um módulo utilizado pelo PyLint para analisar código Python. Ele faz parte do conjunto de ferramentas do projeto pylint.

- **blinker==1.7.0:**

Blinker é uma biblioteca para simplificar a implementação de padrões de design de observadores.

- **Cerberus==1.3.5:**

Cerberus é uma biblioteca de validação de dados em Python. Ela pode ser usada para validar dados em estruturas de dicionário.

- **cfgv==3.4.0:**

Cfgv é uma biblioteca para validação de configurações baseada em esquemas.

- **click==8.1.7:**

Click é uma biblioteca para criação de interfaces de linha de comando (CLI) de maneira fácil.

- **colorama==0.4.6:**

Colorama é uma biblioteca que facilita a adição de cores ao texto impresso no terminal.

- **dill==0.3.8:**

Dill é uma biblioteca para serializar objetos do Python, incluindo funções.

- **distlib==0.3.8:**

Distlib é uma biblioteca para lidar com distribuições Python.

- **filelock==3.13.1:**

Filelock é uma biblioteca para criar bloqueios de arquivos, útil para garantir a exclusividade de recursos compartilhados entre processos.

- **Flask==3.0.2:**

Flask é um framework web leve para Python. É usado para criar aplicativos web.

- **identify==2.5.33:**

Identify é uma biblioteca para identificar tipos de arquivos baseados em seus conteúdos.

- **iniconfig==2.0.0:**

Iniconfig é uma biblioteca para leitura e gravação de arquivos INI.

- **isort==5.13.2:**

Isort é uma ferramenta para organizar automaticamente as importações no código Python.

- **itsdangerous==2.1.2:**

Itsdangerous é uma biblioteca para geração de tokens seguros.

- **Jinja2==3.1.3:**

Jinja2 é um motor de template para Python. Ele é usado para criar templates dinâmicos no Flask.

- **MarkupSafe==2.1.5:**

MarkupSafe é uma biblioteca para tornar strings seguras contra injeções de código em templates.

- **mccabe==0.7.0:**

Mccabe é um módulo usado pelo PyLint para medir a complexidade ciclomática do código.

- **nodeenv==1.8.0:**

Nodeenv é uma ferramenta para criar ambientes Python isolados para projetos.

- **packaging==23.2:**

Packaging é um conjunto de ferramentas para trabalhar com distribuições de Python.

- **pillow==10.2.0:**

Pillow é uma biblioteca de processamento de imagens em Python. É usada para trabalhar com imagens, o que pode estar relacionado à geração de códigos de barras.

- **platformdirs==4.2.0:**

Platformdirs é uma biblioteca para determinar diretórios apropriados para armazenamento de dados específicos do sistema operacional.

- **pluggy==1.4.0:**

Pluggy é um mecanismo de plugin para Python.

- **pre-commit==3.6.0:**

Pre-commit é uma ferramenta para execução de ganchos (hooks) pré-commit, como formatação de código e linting.

- **pylint==3.0.3:**

Pylint é uma ferramenta de linting para Python, usada para analisar o código em busca de possíveis problemas e seguir as convenções de estilo.

- **pytest==8.0.0:**

Pytest é uma estrutura de teste para Python. É usada para escrever e executar testes automatizados.

- **python-barcode==0.15.1:**

Python-barcode é uma biblioteca para a geração de códigos de barras em Python.

- **PyYAML==6.0.1:**

PyYAML é uma biblioteca para análise e geração de documentos YAML**.**

- **tomlkit==0.12.3:**

Tomlkit é uma biblioteca para manipulação de arquivos TOML em Python.

- **virtualenv==20.25.0:**

Virtualenv é uma ferramenta para criação de ambientes Python virtuais.

- **Werkzeug==3.0.1:**

Werkzeug é uma biblioteca utilizada pelo Flask. Ela fornece utilitários para construir aplicações web.