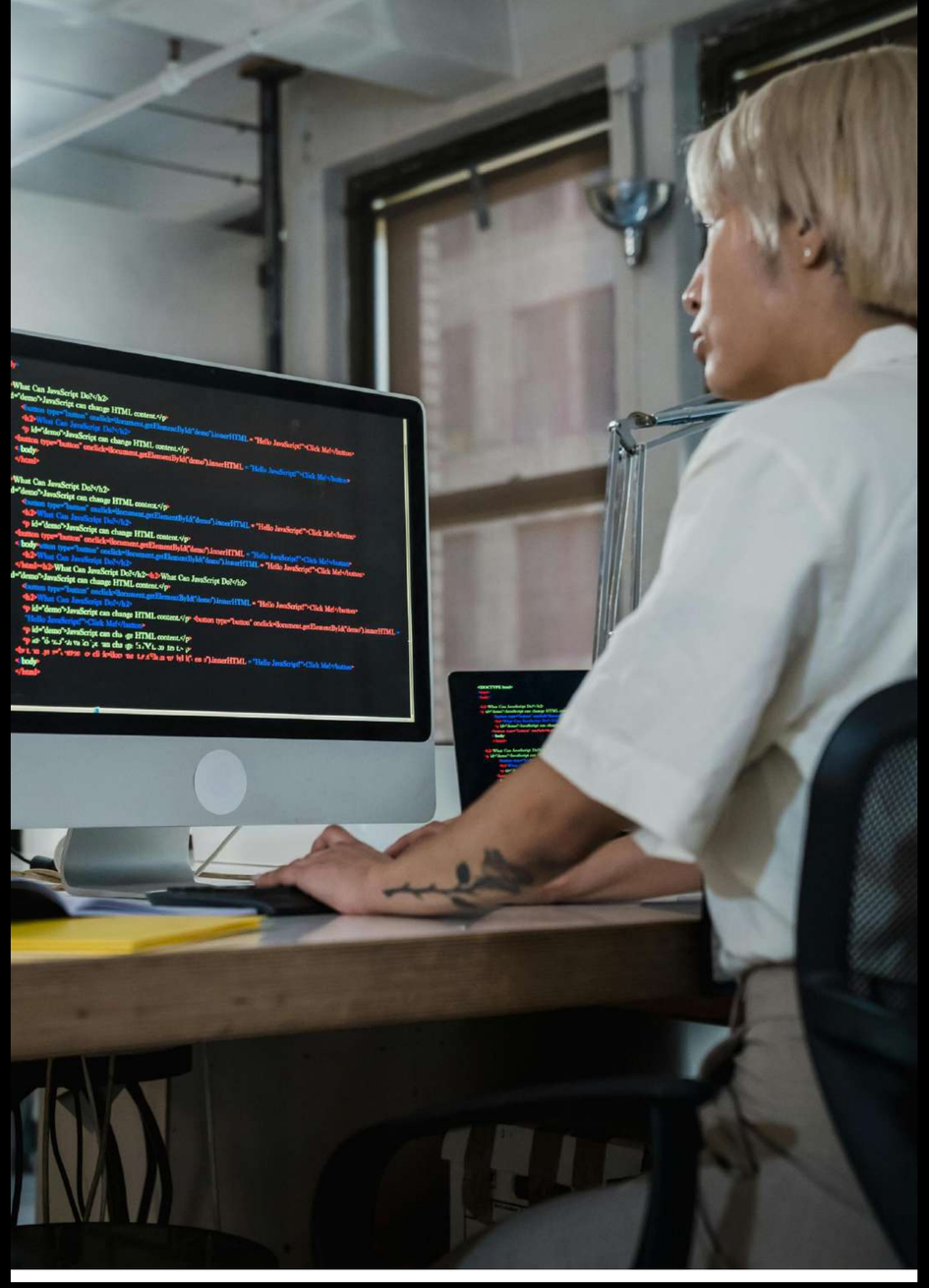


INSTITUTO FEDERAL
Norte de Minas Gerais

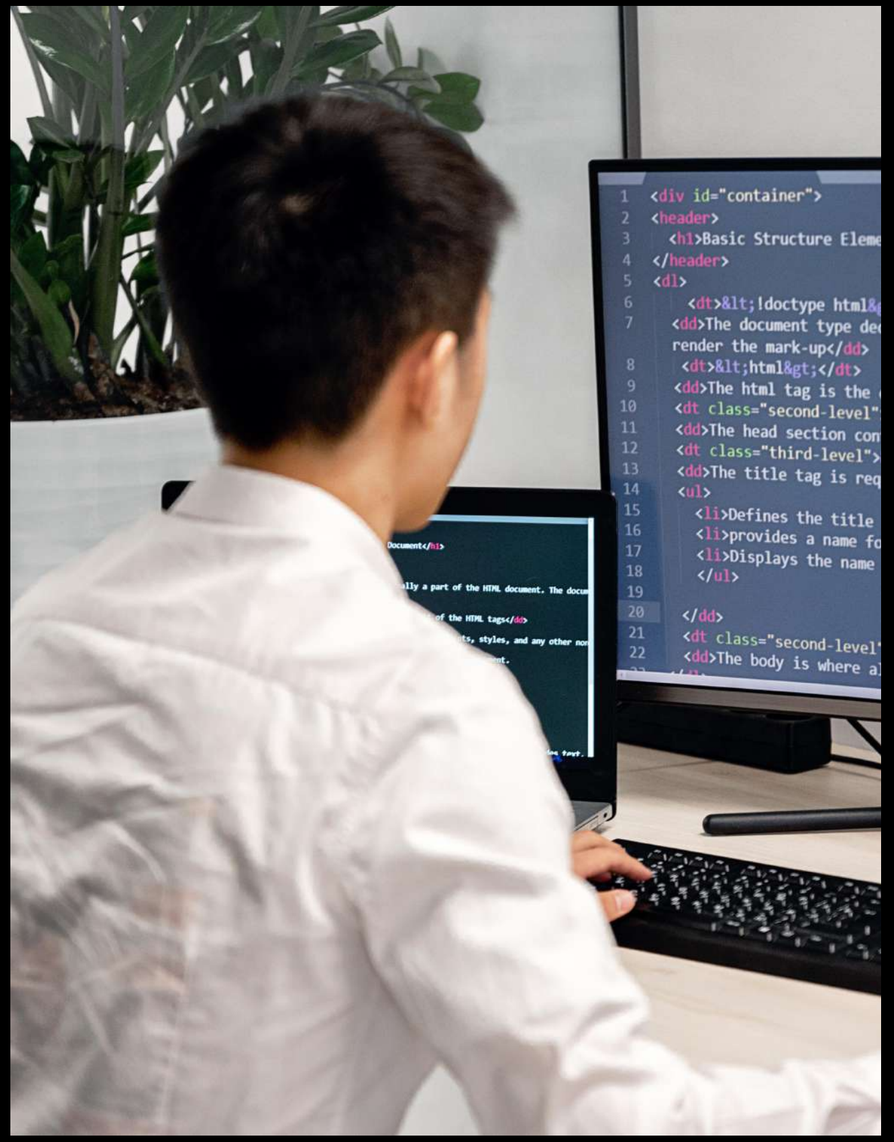
Gerenciamento de Projetos e Ambientes Virtuais no Python (venv)

Thalles Kenedy Oliveira Maia

INSTITUTO FEDERAL
Norte de Minas Gerais

Importância do gerenciamento de dependências

- Reprodutibilidade
- Prevenção de Conflitos
- Manutenção Simplificada

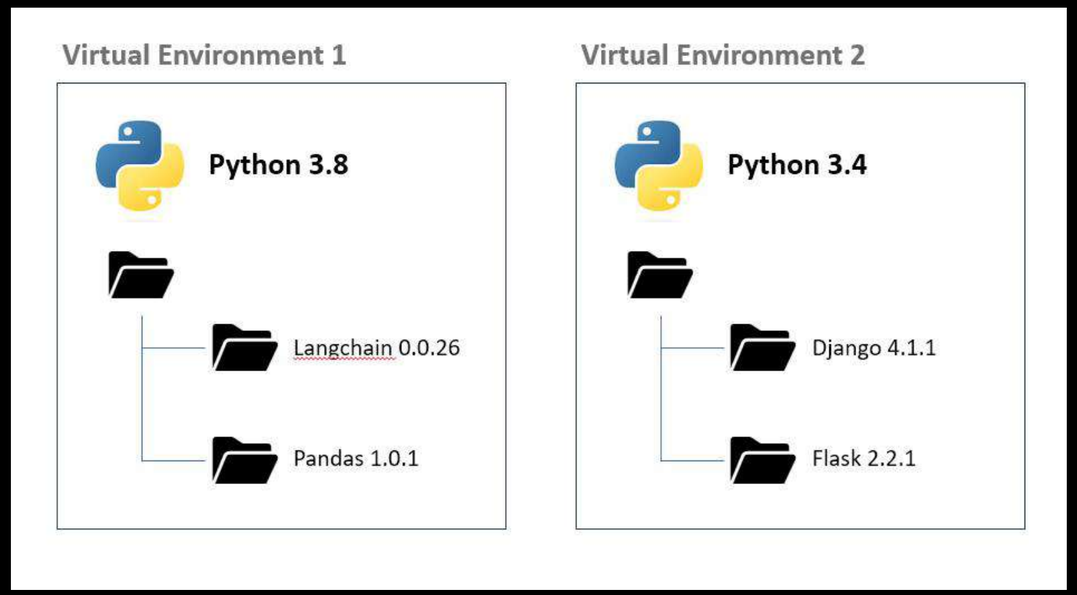


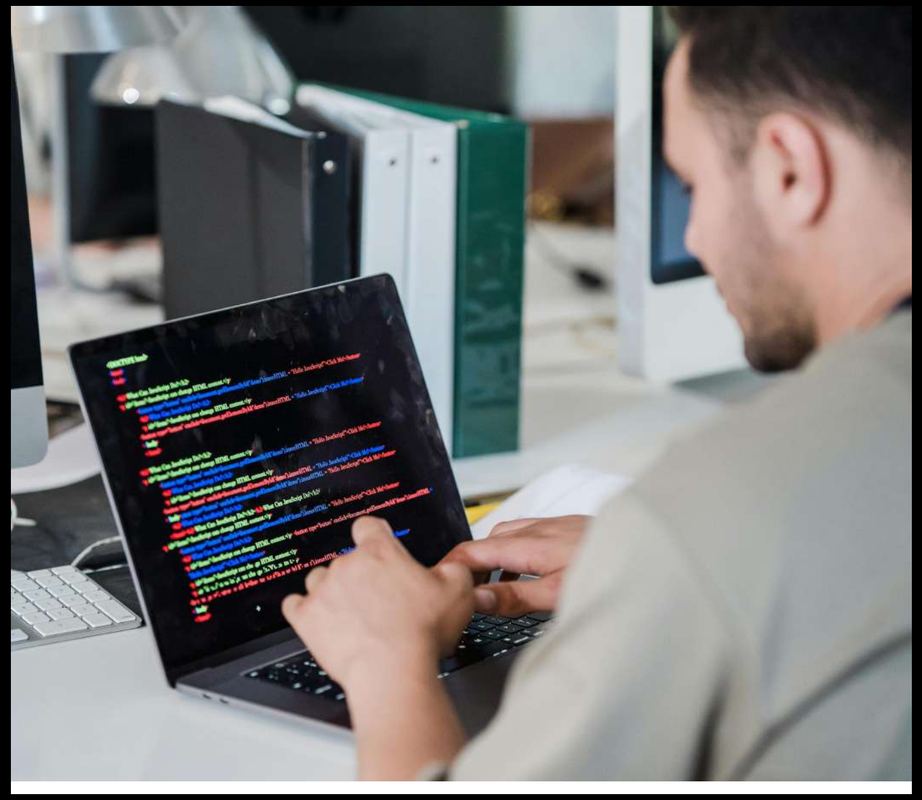
Virtual Environment (venv)

Gerenciamento de múltiplos ambientes de desenvolvimento isolados

- Isolamento
- Organização
- Portabilidade

- Gerenciamento de Versões
- Ativação Explícita





pyenv

Gerencia múltiplas versões do Python
Integração com plugins

Principais funcionalidades do pyenv:

- Instalação de Versões
  (pyenv install 3.10.9)
- Versão Global e Local
- Troca Fácil de Versões
  (pyenv global ou pyenv local)

uv

Gerenciamento de pacotes e ambientes virtuais

Principais funcionalidades do uv:

- Velocidade
- Ferramenta Tudo-em-Um
- Baixo consumo de memória e de recursos

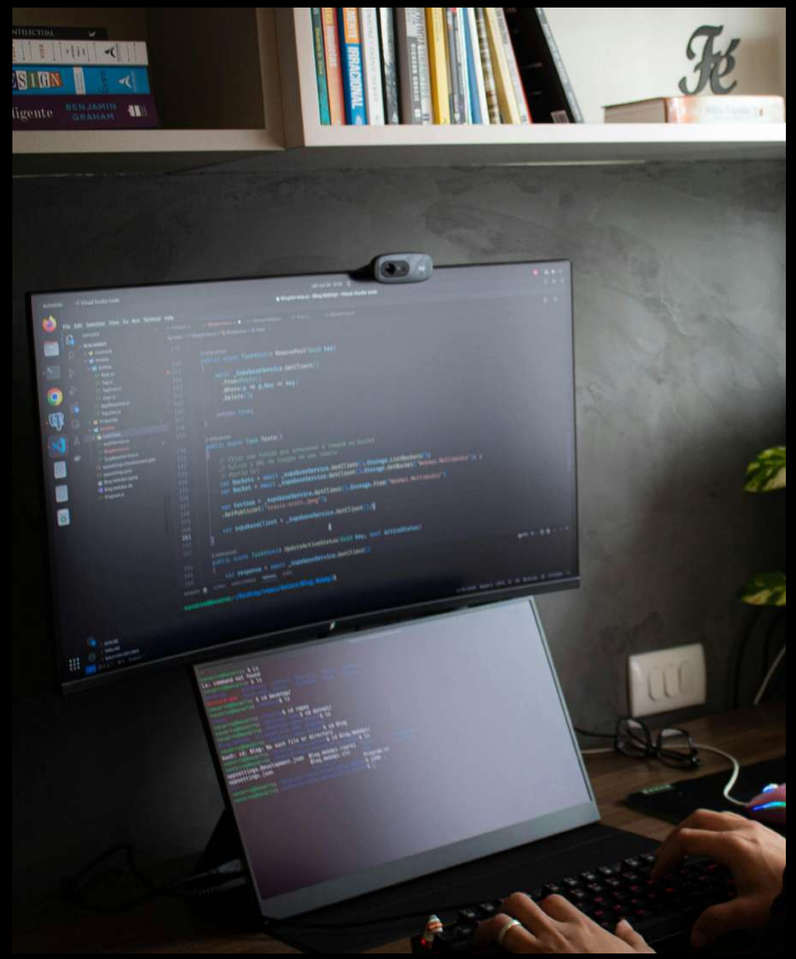
Configuração ambiente uv

Instalar uv:

```
curl -LsSf https://astral.sh/uv/install.sh | sh
```

Conferir a instalação:

```
uv –version
```



Iniciar o uv: (dentro do diretório)

```
uv init
```

Iniciar o uv: (e criar o diretório)

```
uv init nome_do_diretório
```

Criar o venv:

```
uv venv
```

```
ls -a
```

```
main.py
pyproject.toml
.python-version
README.md
.venv
```

Ativar o ambiente virtual:

```
source .venv/bin/activate
```

Instalando dependências:

```
uv add django numpy
```

Remover dependências:

```
uv remove nome_dependencia
```

pyproject.toml

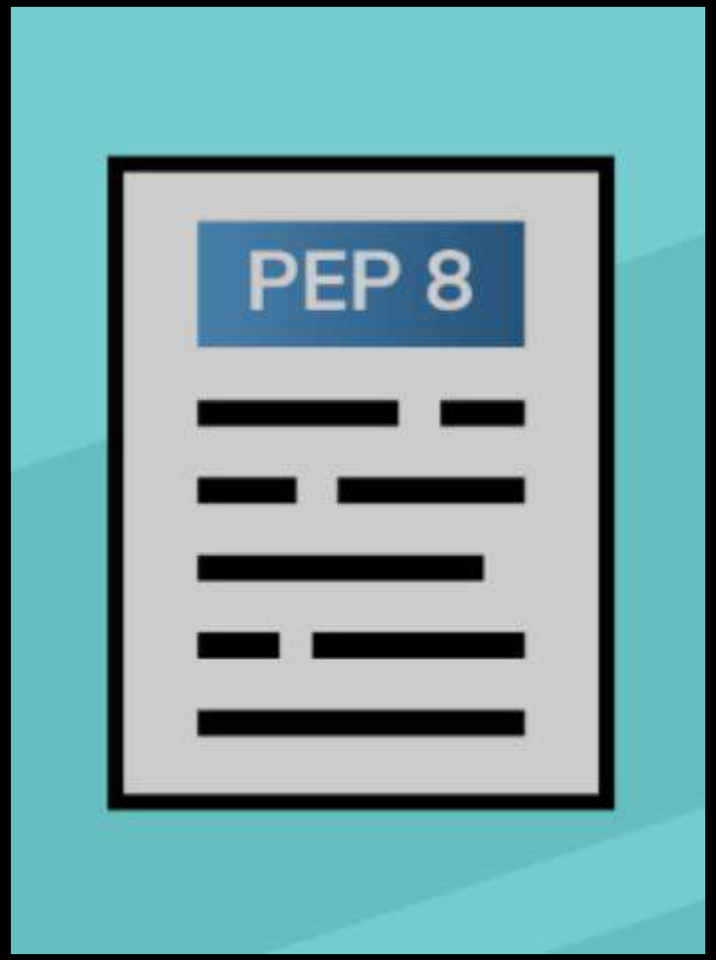
- Metadados do Projeto
- Dependências Principais
- Configuração do Sistema de Build
- Configuração de Ferramentas como formatadores (black) e testadores (pytest).

É a "fonte da verdade" sobre o seu projeto, suas dependências e como ele interage com o ecossistema Python.

cat pyproject.toml

```
[project]
name = "projetotesteuv"
version = "0.1.0"
description = "Add your description here"
readme = "README.md"
requires-python = ">=3.12"
dependencies = [
    "django>=5.2.6",
    "numpy>=2.3.3",
]
```

Boas Práticas em Projetos Python



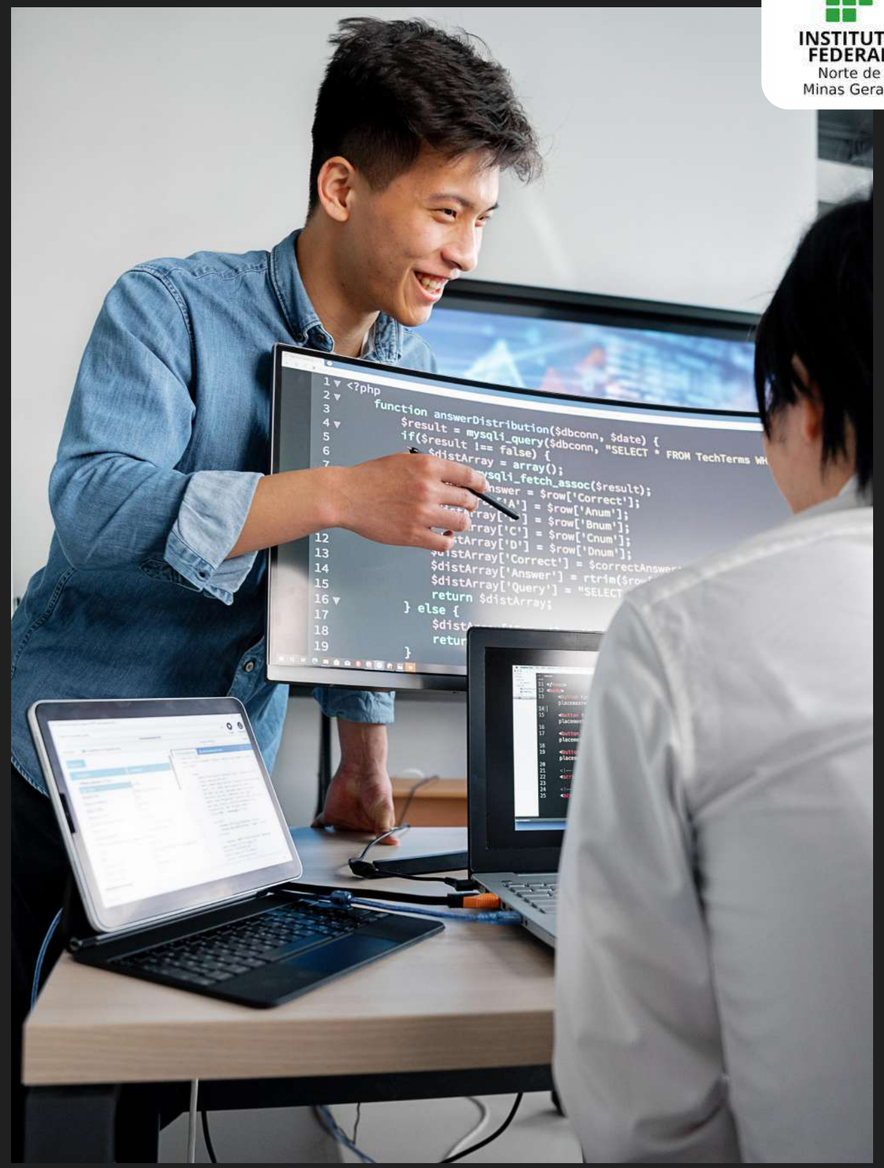
1. Siga um Guia de Estilo
2. Sempre Use um Ambiente Virtual
3. Documente suas Dependências
4. Utilize um Sistema de Controle de Versão
5. Adote uma Estrutura de Projeto Coerente

Obrigado

Links úteis

Guia de Python UV:
https://www.datacamp.com/pt/tutorial/python-uv

Guia de Estilo PEP 8:
https://peps.python.org/pep-0008



INSTITUTO FEDERAL
Norte de Minas Gerais