



**UNIVERSIDADE FEDERAL DE SERGIPE**

DEPARTAMENTO DE COMPUTAÇÃO

TESTE DE SOFTWARE

ISAC KAIK OLIVEIRA SANTOS

Atividade 1

São Cristóvão

2024

## Tutorial

Configuração de ambiente:

Os comandos executados neste documento deverão ser feitos a partir do diretório `validate-docbr`. Para isto, execute o seguinte comando:

- `cd validate-docbr`

Insira o caminho completo até este diretório, se necessário

Algumas Tecnologias e configurações necessárias:

- Python: realizar instalação de acordo com o sistema operacional de sua máquina.
- IDE para execução do projeto. No meu caso, utilizei o VS Code.

É necessário criar um ambiente virtual para manter as dependências do projeto isoladas nesse projeto.

Para criar o ambiente virtual, execute o comando abaixo:

- `python -m venv //` cria um ambiente virtual no diretório atual

O comando acima pode variar dependendo da versão do python instalado na sua máquina.

Execução dos testes já criados do projeto:

Para executar os testes, é necessário ter o `pytest` instalado no ambiente virtual. Se ainda não possui, realize a instalação pelo seguinte comando:

- `pip install pytest`

Agora, para executar os testes deste projeto, execute o seguinte comando:

- `pytest tests/`

O comando acima executará todos os testes da pasta "tests". Nesta versão que baixei do github, possui 32 testes disponíveis em 9 arquivos. Todos passaram com sucesso. Segue imagem da execução dos testes:

```
collected 32 items

tests\test_CNH.py ... [ 9%]
tests\test_CNPJ.py ..... [ 28%]
tests\test_CNS.py ... [ 37%]
tests\test_CPF.py .... [ 50%]
tests\test_Certidao.py ... [ 59%]
tests\test_PIS.py ... [ 68%]
tests\test_RENAVAM.py ... [ 78%]
tests\test_TituloEleitoral.py ..... [ 93%]
tests\test_generic.py .. [100%]

===== 32 passed in 2.76s =====
(validate-docbr) PS C:\Users\Isac - Agape\Documents\Projetos\validate-docbr> pytest tests/
```

Será necessário mais duas bibliotecas serem instaladas: o pytest-cov para validar a cobertura de testes e o mutmut para realizar os testes de mutação. Os comandos para instalação são:

- pip install pytest-cov
- pip install mutmut

Para verificar a cobertura de testes do projeto, execute o comando:

- pytest tests/ --cov

```
----- coverage: platform win32, python 3.12.5-final-0 -----
Name                               Stmts  Miss  Cover   Missing
-----
validate_docbr\BaseDoc.py           35      6  82.86%   11, 20, 32, 42, 55, 67
validate_docbr\CNH.py                45      0 100.00%
validate_docbr\CNPJ.py               37      0 100.00%
validate_docbr\CNS.py                76      3   96.05%  105-107
validate_docbr\CPF.py                45      0 100.00%
validate_docbr\Certidao.py           38      0 100.00%
validate_docbr\PIS.py                33      0 100.00%
validate_docbr\RENAVAM.py            30      0 100.00%
validate_docbr\TituloEleitoral.py    49      0 100.00%
validate_docbr\__init__.py           11      0 100.00%
validate_docbr\generic.py            10      0 100.00%
-----
TOTAL                                409      9   97.80%

Required test coverage of 97.5% reached. Total coverage: 97.80%

===== 32 passed in 8.61s =====
```

Para gerar o relatório de cobertura de teste, execute o comando:

- pytest tests/ --cov=./ --cov-report html

Será criada uma nova pasta no seu projeto chamada htmlcov. Abra o arquivo index.html contido nessa pasta para visualizar o relatório de cobertura.

Com o mutmut já instalado, execute o seguinte comando para gerar os testes de mutação:

- `mutmut run`

O mutmut irá informar quantos mutantes gerou, quantos desses mutantes os testes conseguiu matar, ou seja, apontar erros e quantos não conseguiu. Execute o comando a seguir para gerar um arquivo html contendo a relação dos mudantes gerados por arquivo.

- `python -m mutmut html`

Após a verificação dos mutantes gerados e ao perceber que com eles, o código de teste ainda passou, o ideal é verificar o motivo de não ter informado erro e corrigir os códigos de teste. Executando os mutantes em um arquivo especificamente, o `test_CPF.py`, obtive o seguinte resultado:

```
🕒 Timeout.      Test suite took 10 times as long as the baseline so were killed.
😟 Suspicious.   Tests took a long time, but not long enough to be fatal.
😬 Survived.     This means your tests need to be expanded.
🚫 Skipped.     Skipped.

1. Running tests without mutations
: Running...Done

2. Checking mutants
: 85/85 🚫 0 🕒 0 😟 0 😬 85 🚫 0
(validate-docbr) PS C:\Users\Isac - Agape\Documents\Projetos\validate-docbr>
```

Link do repositório no GitHub:

[https://github.com/isackaik/Teste\\_Software\\_2024\\_santos\\_isac](https://github.com/isackaik/Teste_Software_2024_santos_isac)