CONSULTA DE CEP – MAGAZINE LUIZA

Por: Vitor Araújo Oliveira

GitHub: https://github.com/VitorAraujoOliveira/consulta-de-cep

Requerimentos principais:

· Dado um CEP válido, deve retornar o endereço correspondente

· Dado um CEP válido, que não exista o endereço, deve substituir um dígito da direita para a esquerda por zero até que o endereço seja localizado (Exemplo: Dado 22333999 tentar com 22333990, 22333900 …)

· Dado um CEP inválido, deve retornar uma mensagem reportando o erro: "CEP inválido"O que se espera para as questões 1.

O sistema foi planejado para Windows, modificações podem ser necessárias para aplicação em sistemas IOS e LINUX

# TECNOLOGIAS UTILIZADAS

Para esta construção foram utilizadas as seguintes tecnologias:

**Front End:** Django

**Back End:** Bibliotecas Python /Logica Python (Pandas, pycep\_correios, flask)

**Banco de dados:** Não foi utilizado um banco de dados formal para a aplicação mas sim uma estrutura em arquivos. Existem dois arquivos .csv, são estes, ceps.csv e logs.csv, cada arquivo cumpre sua função específica para coleta e armazenamento de dados.

# ESTRUTURA API

A API possui apenas 4 rotas. Ao iniciar a mesma esta deve fornecer um caminho semelhante a: <http://127.0.0.1:5000/> e, a partir da url inicial é possível acessar as seguintes rotas.

|  |  |  |  |
| --- | --- | --- | --- |
| **Rota** | **Método** | **Content Type** | **Implementação Completa** |
| / | GET | - | Sim |
| /cep/correios/ | POST | Application/JSON | Sim |
| /cep/local/ | POST | Application/JSON | Sim |
| /cep/logs/ | GET | Application/JSON | Sim |

A API pode ser acessada via Postman ou por qualquer outro método que permita a funcionalidade POST e GET.

As funções da API acessam os arquivos CSV com os dados.

A API é dividida em dois arquivos:

**API.py:** (Instancia a API Flask e inicia o servidor)

**Scraper.py:** (Instancia os scripts e arquivos específicos)

# ESTRUTURA DO FRONT END

Para acessar o front end no momento atual do desenvolvimento é necessário ativar o servidor django: Navegue pelo shell do windows ou pelo prompt de comando até a pasta ConsultaCEP e rodar a seguinte linha de comando:

python manage.py runserver

Considerando que o servidor django esteja instalado corretamente a linha deverá ser executada, dando o seguinte resultado

|  |
| --- |
| Watching for file changes with StatReloader |
| Performing system checks... |
|  |
| System check identified no issues (0 silenced). |
| September 23, 2020 - 14:07:14 |
| Django version 3.1.1, using settings 'config.settings' |
| Starting development server at http://127.0.0.1:8000/ |
| Quit the server with CTRL-BREAK. |
| [23/Sep/2020 14:07:41] "GET / HTTP/1.1" 200 3497 |

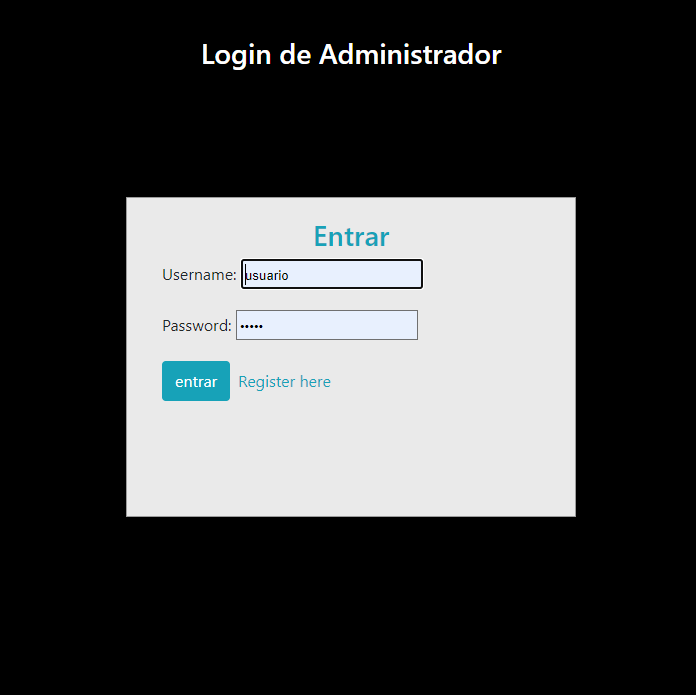
Assim o servidor rodará, basta acessar <http://127.0.0.1:8000/>

Página inicial:

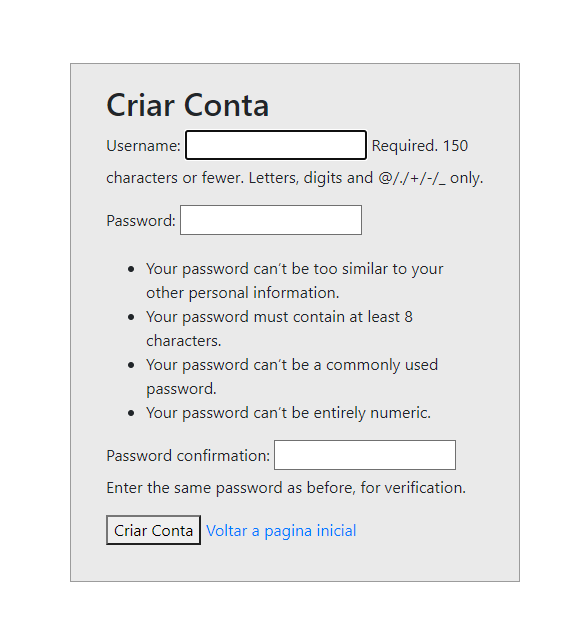


A página inicial possui duas possibilidades. Consultar o CEP ou entrar como Admin. A consulta de CEP é bem direta, aplique o cep e clique em consultar (Consultar pode ser feito pelos correios ou pelo banco local).

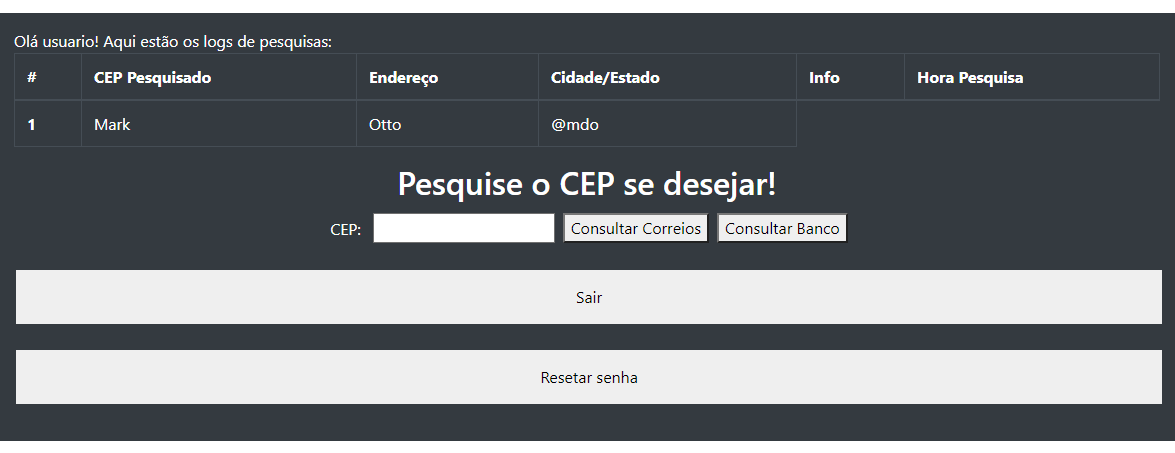
Ao clicar em Admin se seguirá para a próxima tela



O login de administrador. Caso não tenha uma conta basta clicar em register here.



O formulário de criação de conta. Este foi utilizado a partir das tecnologias do django em um exemplo. Apenas editei as informações gerais e o layout CSS. O script veio de um exemplo Django para login e gerenciamento de usuários.



A tela do administrador que consegue verificar os logs de usuário e também verificar o CEP.