Sistema de Controle de Carga e Descarga de Mercadorias

Sistema para automatizar o processo de carga e descarga de mercadorias da Fribel Comércio de Alimentos Ltda.

Características:

- Sistema web

- Sistema criado com Python (Django), HTML 5, CSS e JavaScript

- Sistema com gerenciamento de permissão de acesso via autenticação

- Sistema conectado com outros sistemas para importação de informações

Funcionalidades:

- Autenticação

- Listagem de cargas

- CRUD de cargas

- Alerta de conflito de cargas marcadas para descarga no mesmo dia

- Alerta de liberação de carga

- Detalhamento de cargas com informações de nota fiscal e banco de dados

Requisitos:

- python https://www.python.org/

- django https://www.djangoproject.com/

- selenium https://selenium-python.readthedocs.io/

- cx\_oracle <https://oracle.github.io/python-cx_Oracle/>

-Cliente Oracle 64bits

INSTALAÇÃO(CASO O SISTEMA SEJA MUDADO DE MAQUINA):

1-Instalar Python 3 ou mais na maquina

2- Após instalar o Python criar o ambiente virtual

2.2 “ pip install virtualenv ” copiar sem aspas

2.1 usando o prompt de comando vá até a pasta que deseja estalar o projeto

Ex= “ cd C:\Users\marcio\Documents\ProjetoDjango ”

2.3 ainda no prompt de o comando “virtualenv nome\_da\_virtualenv”

2.3.1 será criada uma pasta com o nome que você nomeio acima

2.3.2 no prompt navegue até a pasta você criou

Ex:

“cd C:\Users\marcio\Documents\ProjetoDjango\nome\_da\_virtualenv\Script”

2.3.3 execute “.\activate ”

Será criada um ambiente virtual , é necessário ativa o ambiente virtual todas as vezes que o projeto for iniciado (passo 2.3.2 e 2.3.3 ).

3-instalando o django

3.1 No prompt de comando com o ambiente virtual ativado

“python -m pip install Django”

4-instalando selenium

“ pip install selenium”

5-instalando cx\_oracle

“ python -m pip install cx\_Oracle --upgrade”

6- levantando o servidor django

Com os passos acima o ambiente está pronto para levanta um servidor django para o projeto carga e descarga da fribel

Com um ambiente ativado no prompt de comando vá ate o caminho que está o arquivo manage.py

ex: “cd E:\labes-fribel\_alimentos\carga\_descarga”

execute o comando “python manage.py runserver”obs: irá iniciar em um local host

para inciar com um ip você deve modificar comando para .

“python manage.py runserver ip\_da\_maquina:porta”.

ip\_da\_maquina: para adiciona o comando do ip da maquina você deve também modificar o arquivo que está “labes-fribel\_alimentos\carga\_descarga\carga\_descarga\settings.py”.

o parâmetro ALLOWED\_HOSTS que inicialmente está vazio, com o ip da máquina que vai hospeda o servidor django.

Sobre a porta você deve se atenta as portas que já estão em uso normalmente as portas 9000 ou 8000 estão abertas para uso.

8- ARQUIVOS E PASTAS DO PROJETO

**carga\_descarga**: Raiz do projeto nessa pasta se encontra os arquivo do banco local sqllite, chromedriver e manage.py

**db.sqlite3**: banco de dados local .

**chromedriver**: ferramenta que auxiliar bot de mensagens para whatzaap.

**manage.py**: esse arquivo é responsável por gerencia todo o projeto django para sabe mas sobre a sua funcionalidade no prompt de comando com o ambiente ativado execute o comando “python manage.py help” será listado todos os comando disponível no django

**carga\_descarga- \carga\_descarga\carga\_descarga:** essa pasta está no segundo nível de profundidade apesar dela ter o mesmo nome da pasta raiz seus arquivos são a configurações do projeto:

**settings.py**: configurações

**urls.py**: rotas nesse projeto optamos por não utilizar os url desse arquivo por isso fizemos uma referência para um outro arquivo que contém todas as nossas rotas .

**core-** **\carga\_descarga\core:** essa pasta contem a aplicação está nela os templates htmls, arquivos, estáticos (css,js,imagens).

**migrations**: pasta que contém as migrações feitas no banco local.

**static**: pasta que contém as mídias estáticas como o css, js, imagens.

**templates**: pasta contém os templates html.

**\_\_init\_\_.**py: arquivo Python de inicialização caso você queria iniciar uma função por padrão você deve colocar sua função aqui.

**Admin.py**: esse arquivo Python registra os model para operação na página do admin caso se queira adicionar uma nova model se deve adicionar no admin para que o modelo possa ser visto na página admin do django.

**Apps.py**: esse arquivo Python carrega a referência do app no caso do sistema atual é o core.

**conection\_bd.py**: esse arquivo tem a conexão com o banco, caso seja necessário modificar o usuário com configurações mais ou menos restritivos basta modificar o usuário e senha nesse arquivo. Caso se queria adicionar uma nova consulta no banco o ideal seria criar nesse arquivo, mas não a impedimento para criação em outros arquivos desde que se use a biblioteca cx\_oracle.

**Form.py**: esse é padrão de formulário utilizado no django nesse projeto houve uma tentativa de utilização para mais informações acessa o portal oficial do django.

- django <https://www.djangoproject.com/>

**model.py**: são os modelos de dados a partir dessa página serão geradas as tabelas no banco

**teste\_selenium.py**: esse é o arquivo de teste automatizado com selenium o processo essa funcionalidade está incompleta seria necessário adicionar campos e fazer modificações

**tests.py** : arquivo de teste.

**urls.py** : arquivo com os possíveis caminhos da aplicação caso se queira adicionar um novo caminho.

**views.py:**esse arquivo python contém todas as funções responsável pela funcionalidades da aplicação.

9-CRIANDO USUARIO PARA UTILIZAÇÃO DO SISTEMA

9.1Há duas formas de criar usuário a primeira é via prompt de comando aqui será criado um super usuário não é ideal criar usuários assim, o comando para criação de usuário é:

“ python manage.py createsuperuser” após dar esse comando no terminal irá solicitar suas credenciais nome, e-mail, senha.

9.2 Há segunda forma é criar um usuário pelo admin do django para acessa essa pagina você deve digita o ip ou o domínio e porta no qual o plataforma está hospedado seguido de um “\admin” exemplo: ”192.168.0.31:9000/admin” entre com credenciais superuser no caso para esse projeto o usuário é **usuário** **=** [fribel-admin](http://192.168.0.31:9000/admin/auth/user/5/change/) **Senha = fribel91** clique em usuário adicionar novo usuário é só preencher os campos obs :não marca a opção superusuario para esses usuários

9.3 Adicionando tipo\_user: ainda no admin do django na tela home haverá uma tag chamada tipo\_user clique adicionar novo, haverá uma lista você deve escolher um usuário que ainda não tem um tipo, não pode haver um mesmo usuário com dois tipos diferente, após criar um usuário e atribuir a ele um tipo de usuário, essa nova credencial estará pronta para uso no portal de login de carga e descarga

10-IMPLATAÇÃO

Atualmente o projeto está no servidor biometria seu ip é 192.168.0.31 a porta utilizada é a 9000 sua localização está no disco C nele também há sua env que é o ambiente virtual no caso de reinicialização é necessário ativa a env novamente passo 2.3.2 e 2.3.3. Caso o oracle cliente não esteja instalado o sistema não irá para, porém não irá carrega as informações do banco.

11-CONSIDERAÇÕES FINAIS

O sistema usar como servidor nativo do Django o ideal seria levanta com um servidor com maior robustez e confiabilidade.Em caso de perda do código fonte este projeto pode ser encontrado no github: https://github.com/fabio21777/labes-fribel\_alimentos.git

Em caso de reinicialização do servidor em um terminal ou no vscode execute os comandos

cd C:\carga\_descarga\env\Scripts

.\activate

cd C:\carga\_descarga

python manage.py runserver 192.168.0.31:8000