

INSTALAÇÃO E CONFIGURAÇÃO DE SERVIDORES
TÉCNICO INTEGRADO EM INFORMÁTICA PARA INTERNET
ROTEIRO DE AULA PRÁTICA

Compatibilizar o Apache com Django
através do módulo mod_wsgi

mod_wsgi - permitirá compatibilizar o servidor apache com aplicações python que suportem a interface Python WSGI, entre elas o framework Django.

Por padrão o python3 vem instalado no debian 11

python3 -V

Parte I

Instalação do Módulo wsgi

```
#inicialmente atualize seu sistema
sudo apt update && sudo apt upgrade -y
sudo apt install libapache2-mod-wsgi-py3
```

```
#setup do python
sudo apt install python-setuptools
```

#o pip é o instalador de pacotes python. No debian está disponível o python3

```
sudo apt install python3-pip
pip3 --version
```

Instalação do ambiente virtual através do virtualenv

```
sudo pip3 install virtualenv
```

Crie um diretório para seus projetos

```
mkdir django_project
```

```
cd django_project
```

O comando abaixo deve ser executado dentro do diretório de projeto criado no passo acima

```
virtualenv env
```

```
. env/bin/activate
```

Perceba que o prompt indicará que está sendo executado em ambiente virtual através do prefixo (env). Exemplo:

```
(env) user@hostname:~/django_project$
```

Instalação do Django

```
pip3 install django
```

Agora recomenda-se criar um subdiretório para armazenar o projeto de teste. Todos os demais projetos deverão ser inseridos em outros subdiretórios de django_project.

OBS.: Todos os comandos a seguir devem ser executados no ambiente virtual do virtualenv

```
django-admin startproject my_django_project .
```

Acesse o novo projeto criado e abra o arquivo settings.py. No fim deste arquivo, adicione a seguinte linha:

```
STATIC_ROOT = "/home/user/django_project/my_django_project/static"
```

OBS.: O caminho descrito acima deve ser editado para corresponder ao caminho correto existente no sistema. Prestem atenção aos nomes de usuário, diretório e subdiretórios dos projetos django.

Finalizando o projeto inicial

```
cd ~/django_project
python3 manage.py makemigrations
python3 manage.py migrate
python3 manage.py createsuperuser
```

A criação do super usuário requisitará informações de email e senha. Esse usuário será utilizado para administração do sistema via interface WEB.

```
python3 manage.py collectstatic
python3 manage.py runserver
```

A partir desse comando o servidor django estará no ar pela porta auxiliar 8000. Faça o acesso via browser e verifique que está funcionando.

O comando ctrl+C irá desativar o serviço.

```
deactivate
```

Este comando irá encerrar o ambiente virtual

Parte II

Até agora realizamos a instalação do framework Django e rodamos seu serviço através do servidor WEB embutido. Agora iremos compatibilizar o site com o nosso servidor Apache. Para isso recomenda-se que, inicialmente, desabilite qualquer site que esteja operando pela porta 80.

Você deverá criar um novo arquivo `.conf` no diretório `/etc/apache2/sites-available`

Neste arquivo realize as seguintes configurações:

```
<VirtualHost *:80>
```

```
ServerAdmin admin@djangoproject.localhost
ServerName djangoproject.localhost
ServerAlias www.djangoproject.localhost
DocumentRoot /home/user/django_project
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
Alias /static /home/user/django_project/static
    <Directory /home/user/django_project/static>
        Require all granted
    </Directory>
Alias /static /home/user/django_project/media
    <Directory /home/user/django_project/media>
        Require all granted
    </Directory>
    <Directory /home/user/django_project/my_django_project>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>

WSGIDaemonProcess  django_project  python-path=/home/user/django_project
python-home=/home/user/django_project/env
WSGIProcessGroup django_project
WSGIScriptAlias / /home/user/django_project/my_django_project/wsgi.py

</VirtualHost>
```

OBSERVAÇÃO: No arquivo acima, todos os caminhos precisam ser editados de forma a corresponderem aos caminhos corretos de cada arquivo do projeto.

A estrutura final do projeto deve ficar assim:

```
django_project
├── env (All ENV File)
├── manage.py
├── my_django_project
│   ├── init.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
```

Habilite o site através do comando `a2ensite`. Após, reinicie o serviço `apache2`.

Para finalizar, precisamos dar permissões para acesso aos arquivos do projeto.

```
sudo chmod 664 /home/user/django_project/db.sqlite3
sudo chown :www-data /home/user/django_project/db.sqlite3
sudo chown :www-data /home/user/django_project
```

OBSERVAÇÃO: Mais uma vez preste muita atenção com a edição dos caminhos. Eles precisam corresponder ao que existe em seu sistema.

```
sudo apache2ctl configtest
Verificação de erros de sintaxe.
```

```
cd django_project/
source env/bin/activate
sudo nano my_django_project/settings.py
```

```
ALLOWED_HOSTS = []
```

Em ALLOWED_HOSTS libere o acesso ao seu servidor. Pode ser feito por IP (inserindo o IP do servidor WEB) ou através de um nome (se existir). Caso opte pelo nome, você deverá editar o arquivo /etc/hosts acrescentando a linha a seguir:

```
127.0.0.1 djangoproject.localhost
```

Substitua 'djangoproject.localhost' pelo nome que escolher.

Agora, por fim, reinicie o apache2 e tente o acesso via browser.

Referências

<https://docs.djangoproject.com/en/4.2/howto/deployment/wsgi/modwsgi/>

<https://www.youtube.com/watch?v=boHX307pyQ4&t=10s>

<https://studygyaan.com/django/how-to-setup-django-applications-with-a-pache-and-mod-wsgi-on-ubuntu>