**注意：**

谨慎运行sudo rm -rf /usr/lib/python3.6/site-packages/之类的命令，因为yum和dnf依赖于它，导致系统性问题

**1、安装Nginx**

可以参考CentOS环境中安装Nginx详细过程笔记博客文章

**2、安装Python3.6**

[root@localhost home]# wget   https://www.Python.org/ftp/python/3.6.1/Python-3.6.1.tar.xz

[root@localhost home]# tar  xJf  Python-3.6.1.tar.xz

[root@localhost Python-3.6.1]# cd  Python-3.6.1

[root@localhost Python-3.6.1]# ./configure  --prefix=/usr/local/python3 && make && make install

#创建软连接

[root@localhost Python-3.6.1]#  ln -s /usr/local/python3/bin/python3 /usr/bin/python3

[root@localhost Python-3.6.1]#  ln -s /usr/local/python3/bin/pip3 /usr/bin/pip3

如果需要替换python版本，只需要把软连接修改一下即可：

[root@localhost Python-3.7.3]#  ln -s /usr/local/python3/bin/python3.7 /usr/bin/python3

[root@localhost Python-3.7.3]#  ln -s /usr/local/python3/bin/pip3.7 /usr/bin/pip3

**3、安装virtualenv并通过该工具搭建Python应用环境**

virtualenv是一个创建独立Python开发环境的工具，

它可以为应用提供独立的Python运行环境，解决不同应用间多版本的冲突问题

**①安装virtualenv**

[root@localhost ~]# pip3 install virtualenv

安装成功后在Python3.6的安装目录中的bin目录下会有一个virtualenv可执行文件

**②创建应用目录demo**

[root@localhost ~]# mkdir  demo

**③在demo目录下创建一个python3.6虚拟环境**

可以使用-p PYTHON\_EXE选项在创建虚拟环境的时候指定python版本(只能指定已安装好的python版本)

[root@localhost ~]# cd  demo

[root@localhost demo]#  /usr/local/python3/bin/virtualenv  -p /usr/bin/python3 venv

创建成功后在demo目录下新建了一个venv目录，该目录其实类似Python的安装目录

**④激活虚拟环境**

[root@localhost demo]#  source venv/bin/activate

激活后就可以在该环境下安装Python各种模块了

如果需要关闭虚拟环境的密令如下：

（venv）[root@localhost demo]#  deactivate

**⑤安装flask框架**

（venv）[root@localhost demo]#  pip  install flask

安装成功后在demo目录下创建一个基于flask框架的代码示例来测试的python文件index.py,

**⑥安装gunicorn和部署项目**

（venv）[root@localhost demo]#  pip  install gunicorn(古尼科恩)

安装成功后，使用gunicorn(古尼科恩)运行基于flask的index.py文件

（venv）[root@localhost demo]#  gunicorn --workers=3 index:app -b 127.0.0.1:8080

其中的--workes指定运行的进程数，后面的表示地址绑定

index:app中  index代表当前运行的module名，也就是文件名，后面的app是创建的Flask对象

**部署过的项目命令：**

gunicorn --workers=3 helloworld:app -b 127.0.0.1:8080

gunicorn --workers=3 catdog\_server:app -b 127.0.0.1:8080

具体配置的查看官网：​http://docs.gunicorn.org/en/latest/settings.html#config

运行成功后，可以查看一下index进程

[root@localhost demo]#  ps -ef | grep index

关闭Gunicorn（古尼科恩）进程

1. 显示正在运行的进程：pstree -ap|grep gunicorn
2. 关闭正在运行的进程：kill -9 进程id

⑦修改Nginx配置文件(/etc/nginx/nginx.conf)，具体如下：

#user nobody;

worker\_processes 1;

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

#pid logs/nginx.pid;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

#log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

# '$status $body\_bytes\_sent "$http\_referer" '

# '"$http\_user\_agent" "$http\_x\_forwarded\_for"';

#access\_log logs/access.log main;

sendfile on;

#tcp\_nopush on;

#keepalive\_timeout 0;

keepalive\_timeout 65;

#gzip on;

server {

listen 80;

server\_name pw.com;

location / {

#设置代理地址

proxy\_pass http://127.0.0.1:8080;

proxy\_redirect off;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

location /static/ {

if (-f $request\_filename) {

rewrite ^/static/(.\*)$ /static/$1 break;

}

}

}

}

gunicorn启动python文件后，启动Nginx，直接访问本机的ip地址即可

(venv)[root@localhost HelloWorld]# nginx

Nginx日志存放位置：/var/log/nginx/access.log