**Django项目部署**

1. **Linux系统**

无界面操作，安全程度高，相对稳定

一般情况，web应用都是部署在Linux系统服务器上

服务器：自己搭建服务器，购买云服务器（方便，成本低） 阿里云、腾讯云、华为云

1. **在Ubuntu系统部署Django项目**

Django自带测试服务器，在Debug模式下可以处理静态资源，测试服务器性能较低，不可以商用

1. **在Ubuntu系统下创建和window相同的python环境**
2. 在Ubuntu下创建虚拟环境
3. 在系统下安装python3 的pip

sudo apt install python3-pip

1. 安装虚拟环境模块

sudo pip3 install virtualenv

sudo pip3 install virtualenvwrapper

1. 配置环境变量

export WORKON\_HOME=$HOME/.virtualenvs

export VIRTUALENVWRAPPER\_PYTHON=/usr/bin/python3

source /usr/local/bin/virtualenvwrapper.sh

重启配置

source .bashrc

1. Git同步项目

Window系统下搜集虚拟环境所需模块 pip freeze > require.txt

Ubuntu系统下安装git

sudo apt install git

创建目录 sudo mkdir py1904

赋予权限 sudo chmod -R 777 py1904

cd py1904

git clone <https://github.com/zzy0371/Py1904.git>

进入虚拟环境 workon py1904env

cd 到require所在目录

pip install -r require.txt

1. **部署在uwsgi服务器**

在虚拟环境下安装 uwsgi

workon py1904env

pip install uwsgi

cd 到项目根目录创建 uwsgi.ini

sudo touch uwsgi.ini

sudo vi uwsgi.ini

[uwsgi]

# socket=外网ip:端口（使用nginx连接时，使用socket）

http=0.0.0.0:11000

chdir=/home/zzy/py1904/Py1904/demo3

wsgi-file=demo3/wsgi.py

processes=4

threads=2

master=True

pidfile=uwsgi.pid

daemonize=uwsgi.log

改setting配置

Debug False

Allow\_host = [‘\*’]

查看uwsgi进程 ps -ef|grep uwsgi

启动

Cd uwsgi.ini 同级目录

确保在虚拟环境下

uwsgi –-ini uwsgi.ini

uwsgi –-stop uwsgi.pid

1. **部署Nginx服务器**

Uwsgi服务器不能处理静态文件

Nginx优点：进行负载均衡，反向解析

sudo apt install nginx

sudo nginx

sudo niginx -s stop

先改uwsgi socket放开 http注释

收集静态资源

在项目根目录新建staticroot

在sttings中配置 STATIC\_ROOT = staticroot文件夹绝对路径

python manage.py collectstatic

在http子级

server {

listen 0.0.0.0:21000;

location / {

include uwsgi\_params;

uwsgi\_pass 0.0.0.0:11000;

}

location /static {

alias /home/zzy/py1904/Py1904/demo3/staticroot;

}

location /media {

alias /home/zzy/py1904/Py1904/demo3/media;

}

}

重启Uwsgi

重启Nginx