

摸鱼遥感平台

[部署文档]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 日期 | 版本号 | 修订说明 | 修订人 | 审核人 | 批准人 |
| 2022-07-04 | V1.0 | 创建 | 季晓东 | 文皓 |  |
|  |  |  |  |  |  |

目录

[1. 引言 1](#_Toc108038156)

[1.1. 目的 1](#_Toc108038157)

[1.2. 名词解释 1](#_Toc108038158)

[2. 系统配置 1](#_Toc108038159)

[2.1. 服务器运行环境 1](#_Toc108038160)

[2.2. 客户端运行环境 2](#_Toc108038161)

[3. 软件部署流程 2](#_Toc108038162)

[3.1. 服务器软件准备 2](#_Toc108038163)

[3.2. 项目文件准备 3](#_Toc108038164)

# 引言

## 目的

本文档旨在为 **摸鱼遥感平台** 遥感图像在线分析系统的管理员提供系统安装、部署帮助，以便使用者可以独立完成系统的安装和部署，使系统正常运行。

## 名词解释

|  |  |
| --- | --- |
| Filezilla | 一款实现了FTP,SFTP协议的软件，用于从远程主机下载、上传文件。 |
| Ubuntu | Linux操作系统发行版之一。 |
| Docker | Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的镜像中，然后发布到任何流行的 Linux或Windows操作系统的机器上，也可以实现虚拟化。 |
| Dockerfile | 用于指导Docker镜像构建过程的配置文件。 |
| docker-compose | 一款用于管理多个docker容器的软件，由docker项目维护。 |

# 系统配置

## 服务器运行环境

用户本项目的服务端运行在腾讯云提供的云主机上，具体参数如下：

|  |  |
| --- | --- |
| CPU： | Intel(R) Xeon(R) Platinum 8255C CPU @ 2.50GHz，2核 |
| 内存： | 至少4G |
| 磁盘空间： | 40G |
| 操作系统： | Ubuntu 18.04 |

注：本项目要求CPU架构必须是x86\_64。内存至少达到4G，操作系统必须是Linux。请避免使用Arm架构的主机和内存大小过低的机器运行本程序。

## 客户端运行环境

Microsoft Edge浏览器：版本103。

# 软件部署流程

## 服务器软件准备

我们需要在服务器上准备好git、docker、docker-compos、nodejs四个软件。对于使用Ubuntu操作系统的部署者，依次执行如下三条命令即可。

apt-get install git

apt-get install docker.io

apt-get install docker-compose

apt-get install nodejs

对于使用其他Linux发行版的部署者，应使用不同的软件安装指令（如RedHat系Linux需要使用yum）。同时需要注意这三款软件在其他系统的软件源中可能与Ubuntu软件源中的命名不同。

## 项目文件准备

1. **前端文件准备**

在服务器的/home目录下执行:

git clone <https://gitee.com/fliserein/rs-platform-frontend.git>

进入到git clone的对应目录之下, 依次执行如下指令

 npm install

 npm run build

最终将会生成一个dist/文件夹, 其中为我们需要的前端静态资源文件。

1. **后端文件准备**

在服务器的/home 目录下执行：

git clone https://gitee.com/birdchristopher/rs-platform-backend.git

得到项目文件目录如下：

 rs-platform-backend/

├── backend

│   ├── account

│   │   ├── admin.py

│   │   ├── apps.py

│   │   ├── auth\_backend.py

│   │   ├── \_\_init\_\_.py

│   │   ├── models.py

│   │   ├── tests.py

│   │   ├── url.py

│   │   └── views.py

│   ├── backend

│   │   ├── asgi.py

│   │   ├── celery.py

│   │   ├── channelsmiddleware.py

│   │   ├── Config.py

│   │   ├── \_\_init\_\_.py

│   │   ├── settings.py

│   │   ├── urls.py

│   │   ├── util.py

│   │   └── wsgi.py

│   ├── celeryd.conf

│   ├── celeryd.init

│   ├── Dockerfile

│   ├── font

│   │   └── simsun.ttc

│   ├── image\_process

│   │   ├── admin.py

│   │   ├── apps.py

│   │   ├── consumer.py

│   │   ├── \_\_init\_\_.py

│   │   ├── models.py

│   │   ├── routing.py

│   │   ├── serializer.py

│   │   ├── tasks.py

│   │   ├── tests.py

│   │   ├── url.py

│   │   └── views.py

│   ├── Makefile

│   ├── manage.py

│   ├── pip.conf

│   ├── requirements.txt

│   ├── sources.list

│   ├── start.sh

│   └── uwsgi.ini

├── deployment

│   ├── compose

│   │   ├── mysql

│   │   │   ├── conf

│   │   │   │   └── my.cnf

│   │   │   └── init

│   │   │       └── init.sql

│   │   ├── nginx

│   │   │   ├── Dockerfile

│   │   │   ├── log

│   │   │   │   ├── access.log

│   │   │   │   └── error.log

│   │   │   ├── mysite.conf

│   │   │   └── nginx.conf

│   │   ├── redis

│   │   │   ├── Dockerfile

│   │   │   └── redis.conf

│   │   ├── uwsgi

│   │   │   └── backend-master.pid

│   │   └── uwsgi.log

│   └── docker-compose.yml

├── README.en.md

├── README.md

└── requirements.txt

项目文件被下载在/home目录下。

1. **将参数文件上传到指定位置**

Git仓库里不包含后端运行神经网络推理时需要的参数文件。这里我们推荐使用Filezilla软件，将所有的参数上传至rs-platform-backend/backend/params目录下（需要手动新建目录）。上传完毕后params目录内部文件结构如下：

params/

├── deploy\_contrast

│   ├── deploy.yaml

│   ├── model.pdiparams

│   ├── model.pdiparams.info

│   └── model.pdmodel

├── deploy\_detection

│   ├── infer\_cfg.yml

│   ├── model.pdiparams

│   ├── model.pdiparams.info

│   └── model.pdmodel

├── deploy\_road\_r

│   ├── deploy.yaml

│   ├── model.pdiparams

│   ├── model.pdiparams.info

│   └── model.pdmodel

└── deploy\_sort

    ├── deploy.yaml

    ├── model.pdiparams

    ├── model.pdiparams.info

    └── model.pdmodel

1. **将编译完毕的前端文件上传到指定位置**

将包含了所有前端文件的dist目录上传至rs-platform-backend/deployment/compose/nginx下。dist目录的结构如下：

dist/

├── css

├── favicon.ico

├── fonts

├── img

├── index.html

└── js

1. **启动容器**

cd /home/rs-platform-backend/deployment

docker-compose build

docker-compose up

这里rs-platform-backend所在的目录可以由部署者自行定义。只需保证第二、第三条命令执行时所在的目录有docker-compose.yml文件即可。

命令执行无报错则代表项目运行成功。此时直接访问服务端主机所在IP地址即可访问到网站。其界面如下:

