组长：XXX XXXXXXXX

组员：XXX XXXXXXXX

XXX XXXXXXXX

XXX XXXXXXXX

**云发布部署文档**

病虫害识别系统

2021

目录

[1 整体说明 2](#_Toc91171934)

[1.1 全部业务说明 2](#_Toc91171935)

[1.2 docker环境安装 2](#_Toc91171936)

[2 服务器1部署 3](#_Toc91171937)

[2.1 Mysql安装 3](#_Toc91171938)

[3 服务器2部署 4](#_Toc91171939)

[3.1 Nacos安装 4](#_Toc91171940)

[3.2 病虫识别模型部署 5](#_Toc91171941)

[4 服务器3部署 7](#_Toc91171942)

[4.1 使用docker-compose部署 7](#_Toc91171943)

[5 服务器4部署 10](#_Toc91171944)

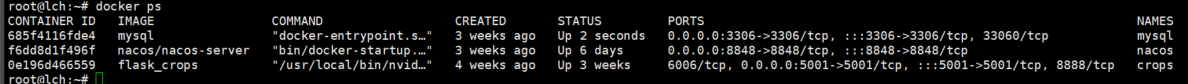
[5.1 使用docker-compose部署微服务 10](#_Toc91171945)

[5.2 前端部署 12](#_Toc91171946)

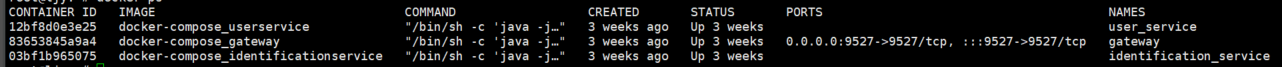
# 整体说明

## 全部业务说明

本项目在部署中共应用了四台阿里云服务器，服务器1（名称lch，ip： 39.99.158.5）用于部署mysql数据库，如图1-1所示；服务器2（名称wg，ip：8.142.15.16）用于nacos以及病虫害识别模型的部署；服务器3（名称ljy，ip：8.142.116.170）用于网关服务、用户服务、以及病虫识别服务的部署，如图1-2所示；服务器4（名称dxd，ip：8.142.25.155）用于病虫信息管理服务、报警信息管理服务和前端的部署，如图1-3所示；



**图1-1 ：服务器1业务**



**图1-2 ：服务器2业务**



**图1-3 ：服务器3业务**

## docker环境安装

三台服务器操作系统均为Ununtu20.04，docker安装采用脚本安装的方式。执行shell指令：

|  |
| --- |
| curl -fsSL https://get.docker.com -o get-docker.sh  sudo sh get-docker.sh |

docker安装完成后，配置镜像加速：

|  |
| --- |
| sudo mkdir -p /etc/docker  sudo tee /etc/docker/daemon.json <<-'EOF'  {    "registry-mirrors": ["https://95rosw2w.mirror.aliyuncs.com"]  }  EOF  sudo systemctl daemon-reload  sudo systemctl restart docker |

安装docker-compose，执行指令

|  |
| --- |
| pip install docker-compose |

至此服务器基础环境配置成功。

# 服务器1部署

## Mysql安装

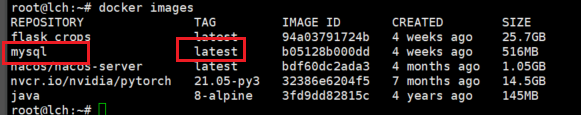
1. pull最新版mysql镜像，执行指令：

|  |
| --- |
| docker pull mysql |

1. 查看docker容器中的镜像

|  |
| --- |
| docker images |

如图：



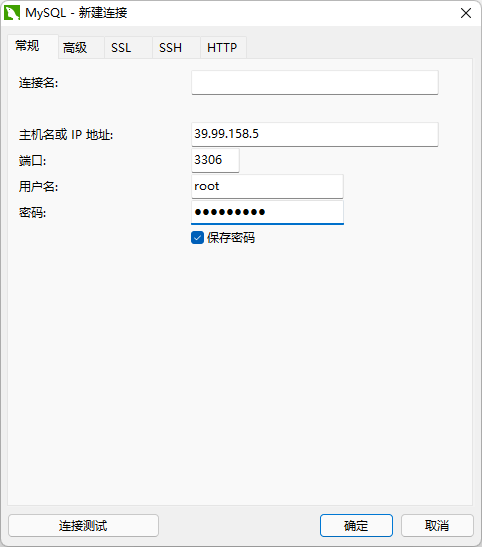
1. 启动镜像（设置密码为lichuanhe）

|  |
| --- |
| docker run -p 3306:3306 --name mysql -e MYSQL\_ROOT\_PASSWORD=lichuanhe -d mysql |

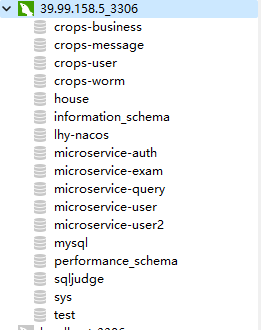
1. 设置安全组，暴露3306端口，如图



1. Navicat连接数据库



1. 连接成功，如图：



# 服务器2部署

## Nacos安装

1. 拉取nacos镜像

|  |
| --- |
| docker pull nacos/nacos-server |

1. 启动nacos容器

|  |
| --- |
| docker run --env MODE=standalone --name nacos -d -p 8848:8848 nacos/nacos-server |

1. nacos的默认端口是8848，默认账号密码是nacos/nacos
2. 配置安全组暴露8848端口，如图



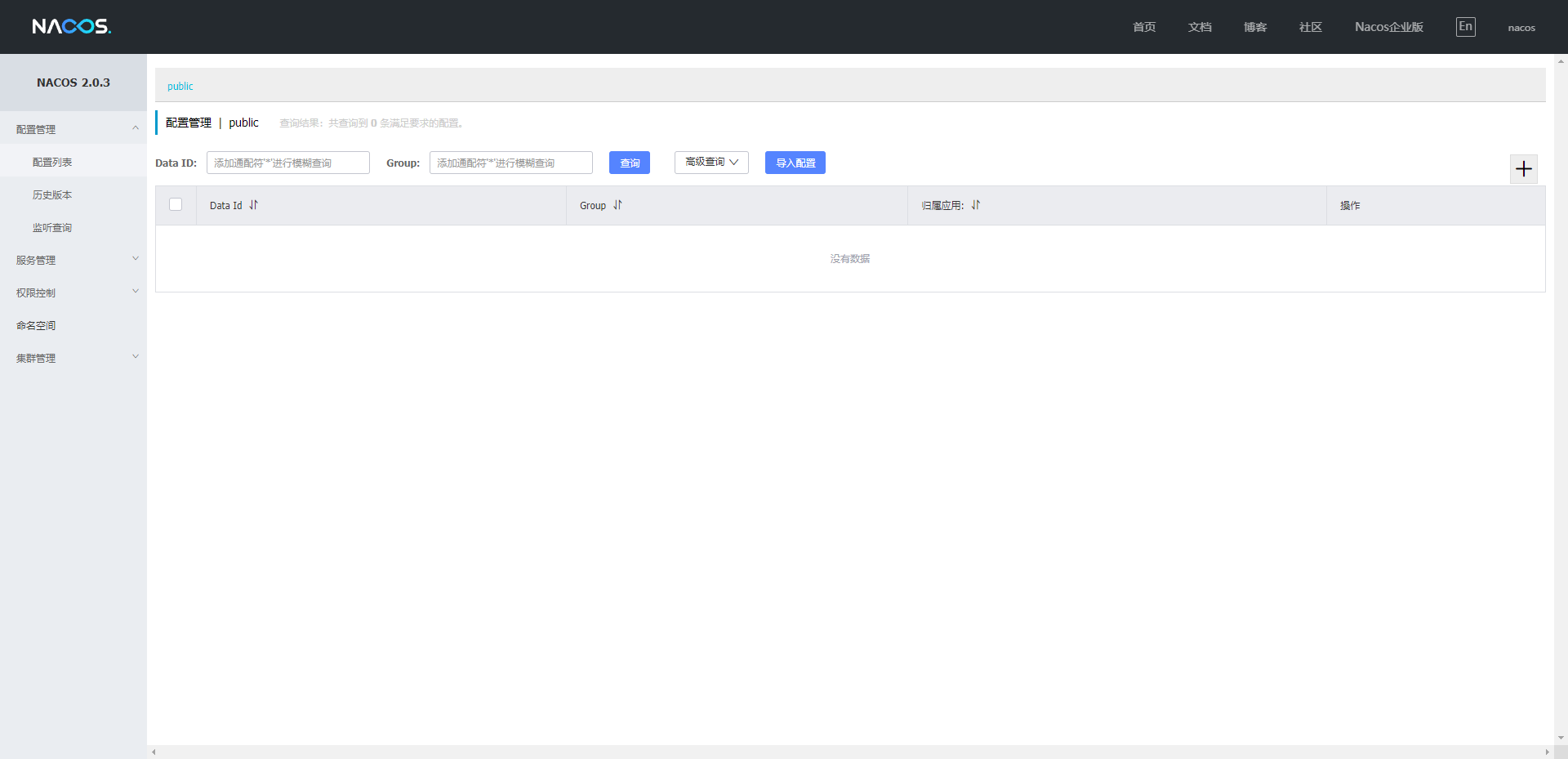
1. 查看nacos容器日志，执行（b5ccf56d36a7为nacos容器id）

|  |
| --- |
| docker logs -f b5ccf56d36a7 |

1. 获得nacos图形化界面地址



1. 至此nacos已配置成功，访问http:// 8.142.15.16:8848/nacos/index.html，如图

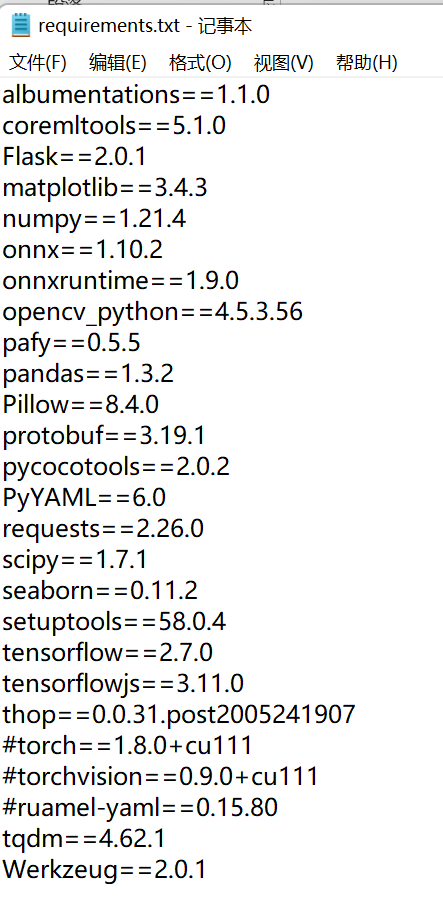


## 病虫识别模型部署

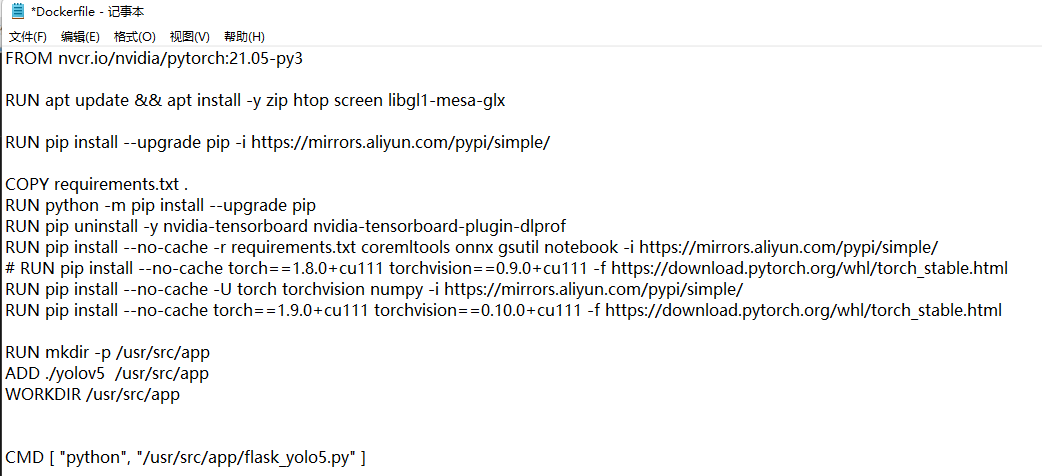
1. 将yolov5模型代码、Dockerfile以及requirenmrnts.txt文件导入服务器/home/docker目录下，如图



1. Requirement.txt文件内记录模型需要的环境，如图：



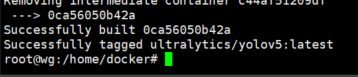
1. Dockerfile文件如图：



该dockerfile，首先拉取pytorch镜像作为依赖环境，然后升级apt及pip，并安装requirements.txt记载的python包，以及其他需要依赖。

1. 切换到/home/docker目录，运行dockerfile，执行：

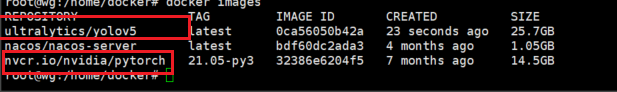
|  |
| --- |
| t=ultralytics/yolov5:latest && sudo docker build -t $t . |



1. 查看docker容器中的镜像

|  |
| --- |
| docker images |

如图：

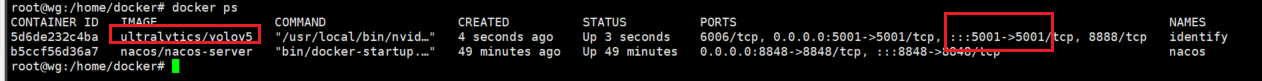


模型镜像搭建成功。

1. 启动flask\_crops容器，并暴露5001端口，执行

|  |
| --- |
| sudo docker run --name identify -p 5001:5001 -d ultralytics/yolov5 |

1. 查看已运行容器，执行docker ps，如图



可见容器已运行成功。

1. 配置安全组，如图。



1. 至此，模型已部署完成

# 服务器3部署

## 使用docker-compose部署

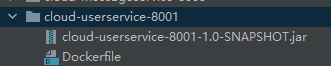
1. 将docker-compose文件夹导入服务器/tmp目录下，如图



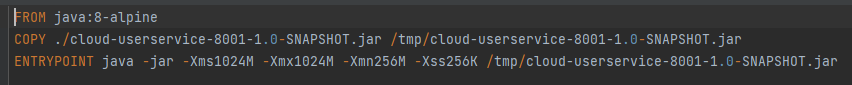
目录结构如下图：



1. 用户服务目录结构



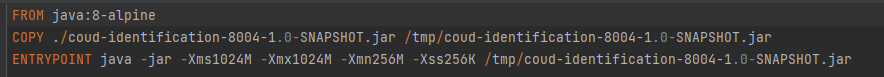
dockerfile如下



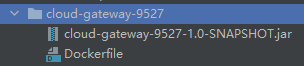
1. 识别服务目录结构如下



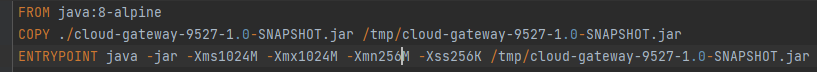
Dockerfile如下：



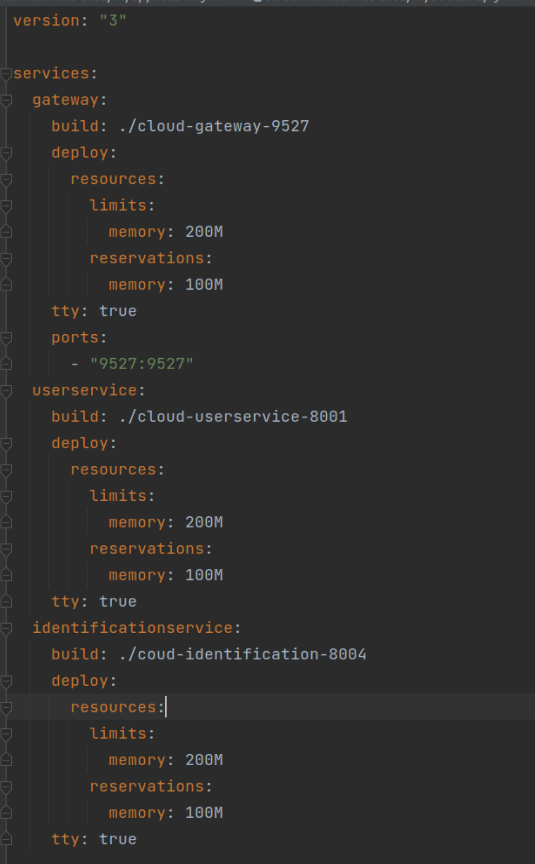
1. 网关服务目录结构如下



Dockerfile如下：



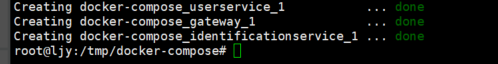
1. docker-compose.yml如下



1. 切换到/tmp/docker-compose目录下，开始打包进行，执行指令

|  |
| --- |
| docker-compose up -d |

如图，说明镜像构建成功



1. 查看容器是否启动成功执行

|  |
| --- |
| docker ps |

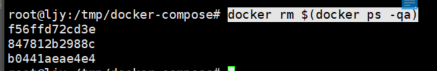


发现容器启动失败

1. 删除所有失败容器

|  |
| --- |
| docker rm $(docker ps -qa) |

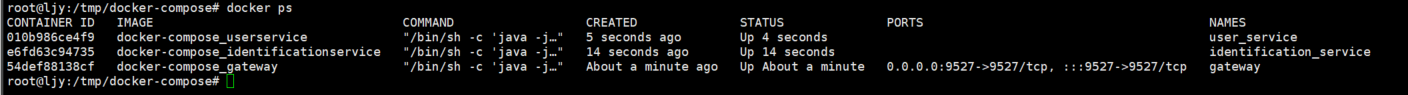
如图，删除成功



1. 手动启动容器，分别执行指令

|  |
| --- |
| docker run --name gateway -p 9527:9527 -d docker-compose\_gateway |
| docker run --name identification\_service -d docker-compose\_identificationservice |
| docker run --name user\_service -d docker-compose\_userservice |

1. 再次查看容器，发现均已启动成功



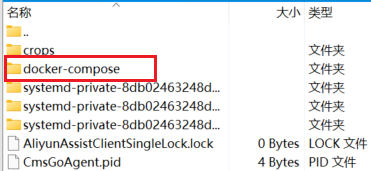
1. 配置安全组，暴露9527端口，如图：



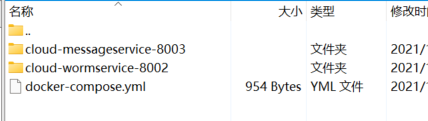
# 服务器4部署

## 使用docker-compose部署微服务

1. 将docker-compose文件夹导入服务器/tmp目录下，如图



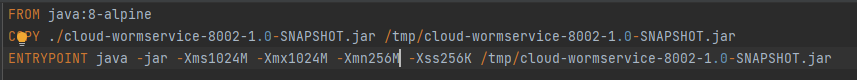
目录结构如下图：



1. 病虫信息管理服务目录结构



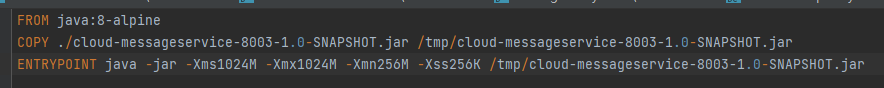
dockerfile如下



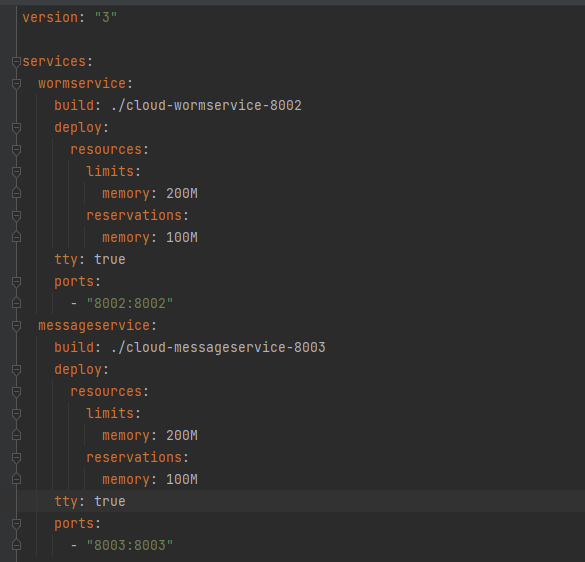
1. 报警信息管理服务服务目录结构如下



Dockerfile如下：

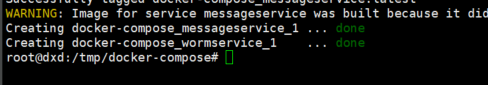


1. docker-compose.yml如下



1. 切换到/tmp/docker-compose目录下，开始打包进行，执行指令

|  |
| --- |
| docker-compose up -d |

1. 查看容器是否启动成功执行

|  |
| --- |
| docker ps |

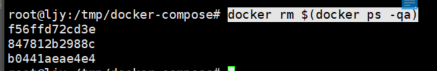


发现容器启动失败

1. 删除所有失败容器

|  |
| --- |
| docker rm $(docker ps -qa) |

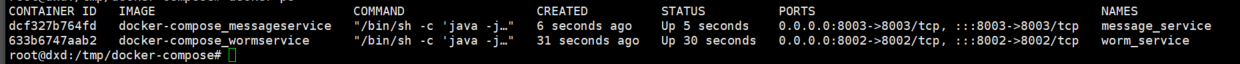
如图，删除成功



1. 手动启动容器，分别执行指令

|  |
| --- |
| docker run --name worm\_service  -p 8002:8002 -d docker-compose\_wormservice |
| docker run --name message\_service -p 8003:8003 -d -v /home/worm\_img:/home/worm\_img -t docker-compose\_messageservice |

1. 再次查看容器，发现均已启动成功



1. 配置安全组，暴露8002及8003端口，如图：



## 前端部署

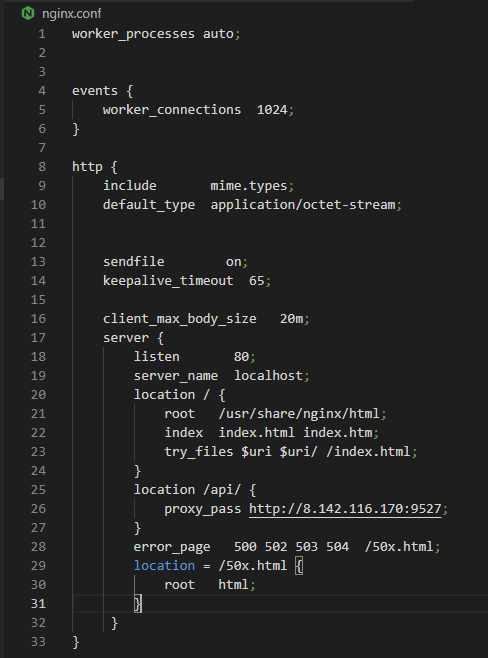
1. 将前端打包文件dist、dockerfile文件、nginx.conf配置文件上传至服务器/tmp/crops/cms目录下，如图



1. Dockerfile如下



1. nginx.conf如下



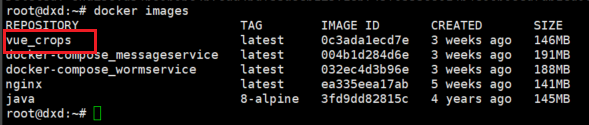
1. 构建镜像，执行指令

|  |
| --- |
| docker build -t vue\_crops . |

1. 查询镜像，输入指令

|  |
| --- |
| docker images |

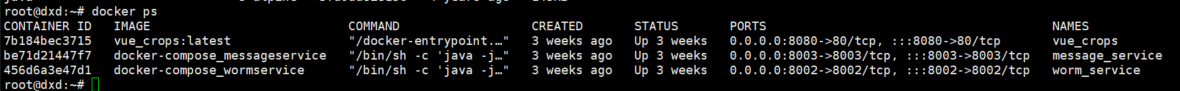
如图：镜像搭建成功



1. 启动容器，执行指令：

|  |
| --- |
| docker run -di -p 8080:80 --name vue\_crops vue\_crops:latest |

1. 输入docker ps查看容器，如图



1. 配置安全组，暴露8080端口

