IM宝塔系统部署文档

### 本系统使用必须符合国家法律法规 具体法律规定看文档

### 自己准备好服务器和域名 等资料

### 文档以Ubuntu为例说明【不同系统 可以按照宝塔官方地址安装】

# 安装音视频功能

## 前期准备

IM系统的音视频功能是基于LiveKit实现的，首先需要一台新的服务器，并准备两个域名绑定该服务器IP，并且开放所有tcp、udp端口。

## 安装步骤

### 安装docker

1. 更新系统 sudo yum update -y
2. 安装所需的依赖项 sudo yum install -y yum-utils device-mapper-persistent-data lvm2
3. 安装Docker引擎 sudo yum install -y docker-ce docker-ce-cli containerd.io
4. 启动Docker服务 sudo systemctl start docker
5. 设置Docker服务开机自启 sudo systemctl enable docker
6. 验证Docker安装 docker --version
7. 下载Docker Compose二进制文件并保存到/usr/local/bin/docker-compose

sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

1. 添加执行权限

sudo chmod +x /usr/local/bin/docker-compose

1. 创建软链接

sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

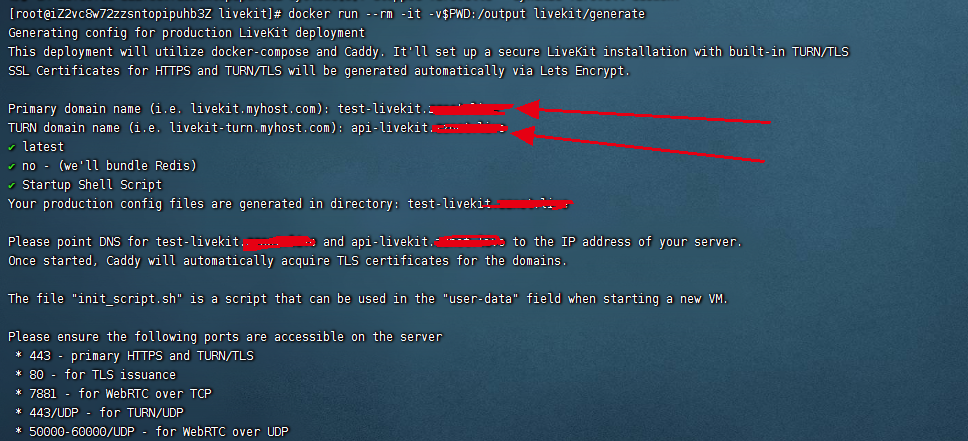
1. 验证安装

docker-compose --version

### 安装LiveKit

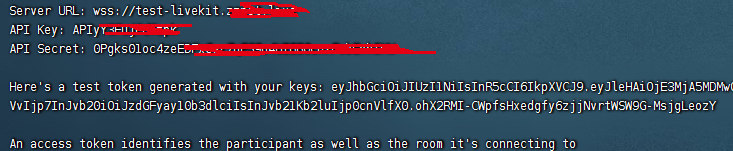
1. 生成配置文件

docker run --rm -it -v$PWD:/output livekit/generate



在执行命令时需要输入提前准备好的两个域名，即箭头所指向的方向 ， 注意，要把所有端口开放

在执行完成后会显示下图内容

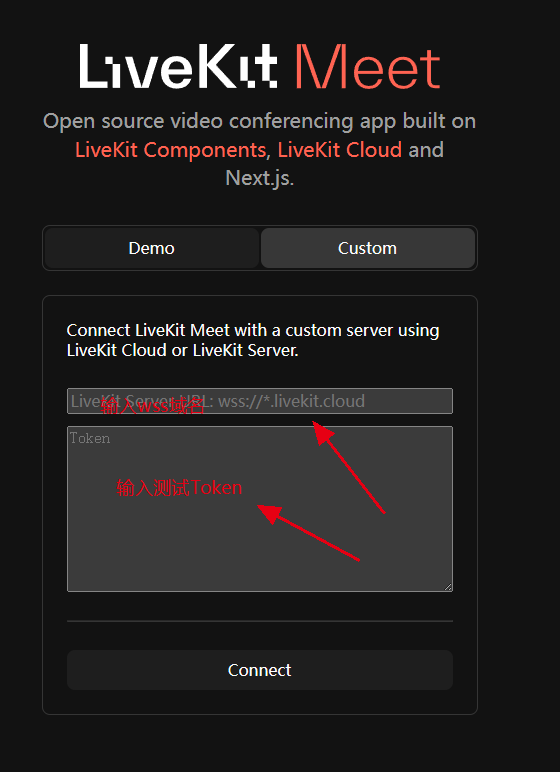


会生成一个应用API key和API Secret以及一个测试Token，并在命令执行的当前文件夹下生成名为你输入的第一个域名的文件夹，

1. 进入域名文件夹内部运行LiveKit docker compose up

如果希望在后台执行输入 docker compose up -d

1. 测试是否成功启动LiveKit

进入https://meet.livekit.io/ ， 输入上图出现的wss域名和token进行测试，如果链接成功，说明项目已经成功启动

**现在已经完成了音视频功能的部署，接下来再重新启动一台服务器，部署后续服务**

# 

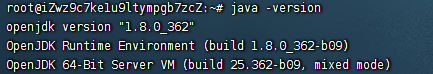
# 安装基础配置

## 安装Java环境

**Java版本要求为1.8**

1. 检查是否自带JDK

Java -version



1. 如果没有出现版本信息则跳至安装步骤，如果不是1.8版本请先卸载再安装Jdk1.8 ， 如果是1.8则跳过安装Java环境这一步骤
2. 安装Jdk1.8

sudo apt update # 升级apt

sudo apt install openjdk-8-jdk # 安装jdk1.8

java -version # 查看Java版本，确定安装成功

## 安装maven

1. 查看是否安装 首先你需要用xshell 或者终端命令

mvn -version

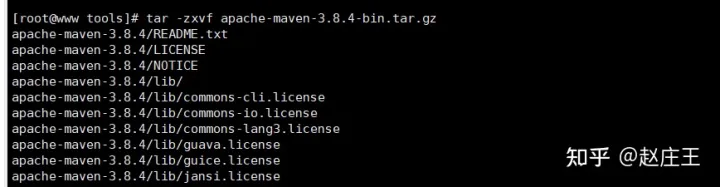
img4.image

1. 下载 Maven，本文使用阿里镜像下载Maven3.6.3，如需其他版本可去Maven官网自行下载：https://maven.apache.org/download.cgi

wget https://mirrors.aliyun.com/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz?spm=a2c6h.25603864.0.0.dba56228ZuFzNX

1. 解压 解压指令 maven压缩包

tar -zxvf apache-maven-3.6.3-bin.tar.gz



配置环境变量 首先你需要知道你的文件所在位置：

img8.image

ls #查看当前路径下的文件及文件夹

pwd #查看当前绝对路径

修改文件

vi /etc/profile 按 a 是获取编辑焦点，

输入 MAVEN\_HOME=你的maven路径

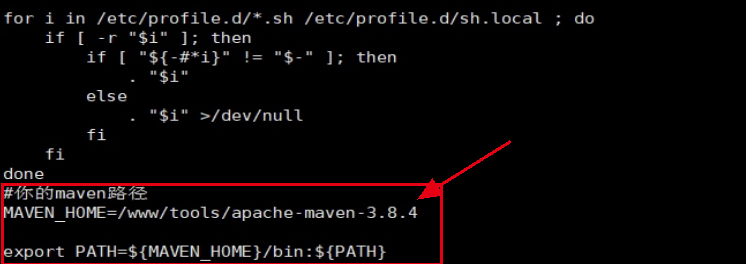
export PATH=MAVEN\_HOME/bin:{PATH}

输入完之后按 ESC 也就是键盘左上角 ，

在文件最后面加入下面配置

Shift+: ，输入x回车退出文档编辑

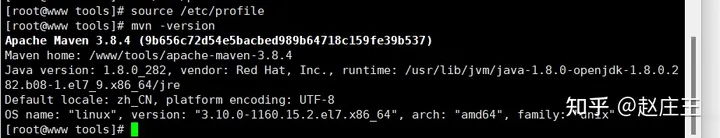
添加配置示例图：



输入指令重新加载环境变量：

source /etc/profile

验证是否安装 mvn -version



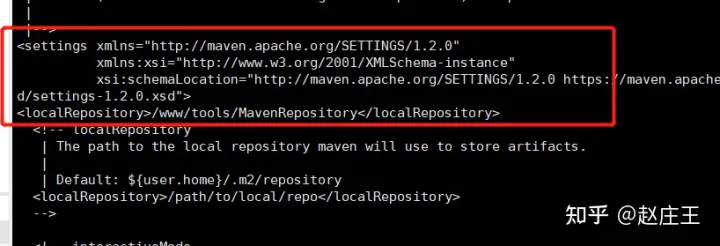
切换国内镜像 如果默认的话，下载会很慢 ，例子为 阿里镜像源 修改settings.xml文件

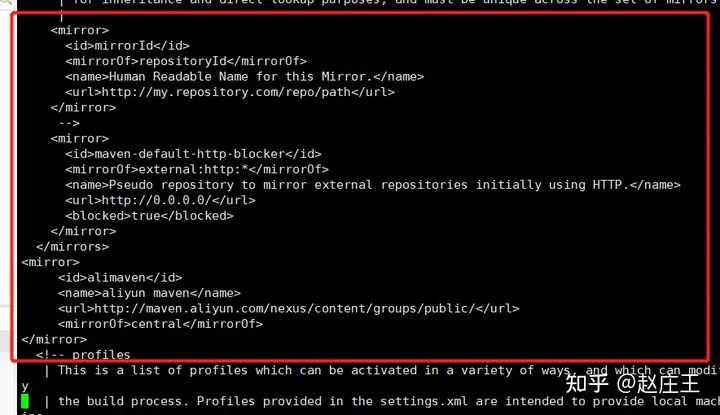
vi 你的maven安装路径/conf/settings.xml

配置资源保存得位置 创建一个文件夹 位置随意，名称随意 ，找到 localRepository (基本在最上边)下面加入代码

代码中的 路径既是你的创建的文件夹名称 MavenRepository

/www/tools/MavenRepository



添加镜像 找到 mirrors 标签(基本快到最 底部了)对，在下方添加一下 代码

<mirrors>

<mirror>

<id>alimaven</id>

<name>aliyun maven</name>

<url>http://maven.aliyun.com/nexus/content/groups/public/</url>

<mirrorOf>central</mirrorOf>

</mirror>

<mirror>

<id>uk</id>

<mirrorOf>central</mirrorOf>

<name>Human Readable Name for this Mirror.</name>

<url>http://uk.maven.org/maven2/</url>

</mirror>

<mirror>

<id>CN</id>

<name>OSChina Central</name>

<url>http://maven.oschina.net/content/groups/public/</url>

<mirrorOf>central</mirrorOf>

</mirror>

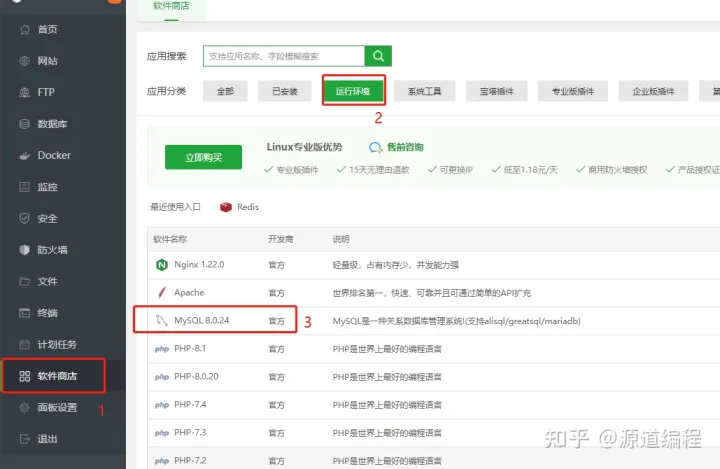
</mirrors>

# 安装mysql redis nginx mongodb docker nacos seaweedfs 项目需要配置环境

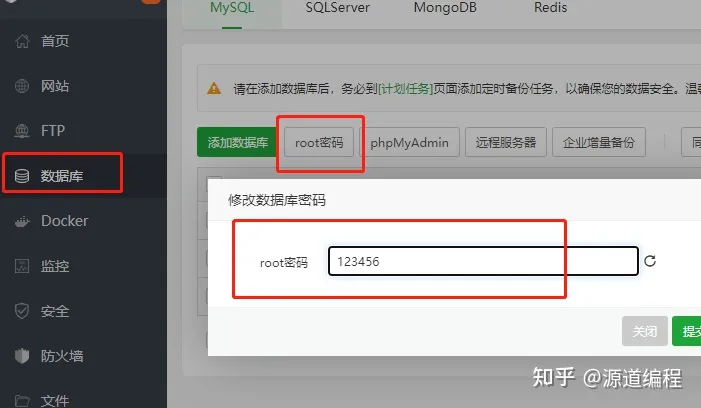
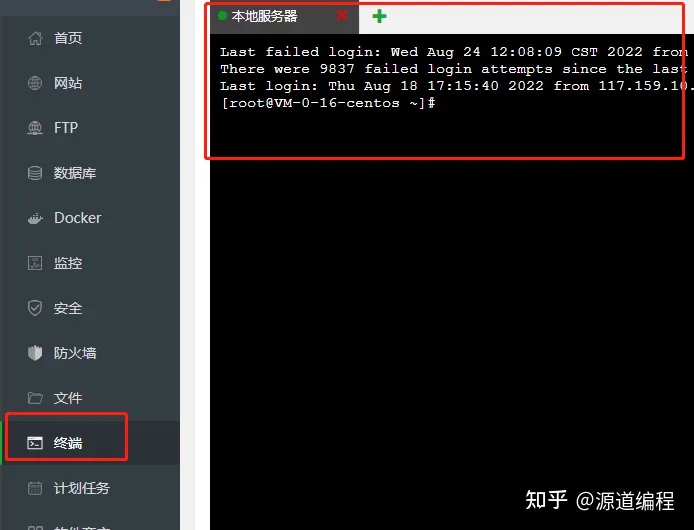
在应用商店安装mysql redis nginx mongodb docker nacos  
注：seaweedfs nacos需要在docker中安装

## 登录到宝塔面板内安装mysql

点击软件商店，选择运行环境，找到mysql，点击后面的安装即可。mysql会自动安装完毕。



配置数据源远程连接：这时候数据库远程还是连不上的，需要命令行进入服务器，设置下才可以。具体步骤如下： 先在宝塔里查看下root用户的密码。复制出来。

 使用终端连接服务器、或者用宝塔里面的终端工具连接也行。 

开启3306端口 1.开启防火墙 systemctl start firewalld

1. 开启3306端口 firewall-cmd –zone=public –add-port=1935/tcp –permanent 命令含义： –zone #作用域 –add-port=1935/tcp #添加端口，

格式为：端口/通讯协议 –permanent #永久生效，没有此参数重启后失效

1. 重启防火墙 systemctl reload firewalld 4.查看端口号 netstat -ntlp //查看当前所有tcp端口· netstat -ntulp |grep 3306 //查看所有3306端口使用情况·

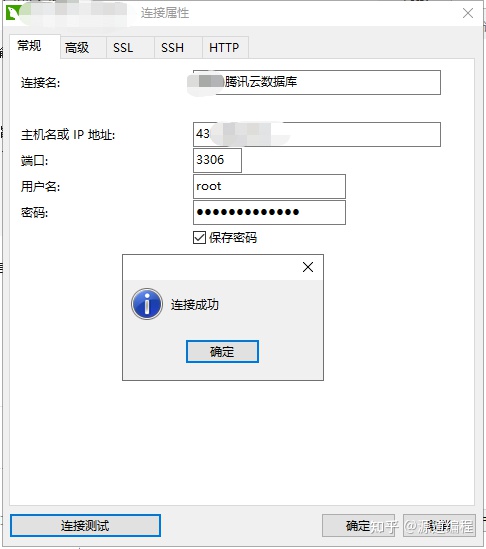
连上服务器后，执行下面步骤即可

【其他方式安装的mysql如果远程连接不上，也可以用下面的方法解决】

#使用root用户进入数据库 输入上面复制的root密码 mysql -u root -p

#使用mysql use mysql

#更新root用户权限，“%”指的是所有地址都可以访问 update user set Host=‘%’ where User=‘root’;

#最后一步，刷新权限即可 flush privileges; 现在即可用Navicat工具来远程连接数据库了。 

注：记得导入后台项目sql文件

## 登录到宝塔面板内安装Nginx

1. 应用商店安装Nginx



在Nginx管理界面，您将会看到有Web服务、配置修改、WAF防火墙、切换版本、负载状态等多个菜单项。切换到Web服务选项卡，在该页面中，您可以停止或重启当前Nginx服务和重新加载Nginx配置文件。

另外，您也可以切换到“配置修改”选项卡，修改当前的Nginx的配置文件，如果需要使当前配置生效，请在Web服务中重载配置文件。

1. Nginx添加站点并配置安全证书

找到宝塔界面的【网站】，点击【添加站点】



输入【域名】后选择提交



在提交后网站中会出现相关操作选项，点击【设置】



点击【ssl】选择【Let‘s Encrypt】，选中当前域名申请免费安全证书，需在服务器安全组和防火墙开放80端口，否则会申请失败。



申请完成后点击【保存】证书，然后设置相关Nginx配置，配置如下：

**对后台项目的Nginx配置如下：**

server { listen 80; server\_name 域名;

listen 443 ssl;  
 #ssl on;  
 ssl\_certificate /etc/letsencrypt/live/域名/fullchain.pem;  
 ssl\_certificate\_key /etc/letsencrypt/live/域名/privkey.pem;  
 #charset koi8-r;  
 #access\_log logs/host.access.log main;  
 location / {  
 root 后台前端打包dist目录地址;  
 index index.html index.htm;  
 error\_log logs/f\_error.log;  
 }  
 location /api {  
 proxy\_set\_header Host $http\_host;  
 proxy\_set\_header X-Real-IP $remote\_addr;  
 proxy\_set\_header REMOTE-HOST $remote\_addr;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 proxy\_pass http://localhost:端口;  
 }  
 location /app {  
 proxy\_set\_header Host $http\_host;  
 proxy\_set\_header X-Real-IP $remote\_addr;  
 proxy\_set\_header REMOTE-HOST $remote\_addr;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 proxy\_pass http://154.204.59.100:8001/api/v1;  
 }  
 # 授权接口  
 location /auth {  
 proxy\_pass http://localhost:端口;  
 proxy\_set\_header X-Forwarded-Proto $scheme;  
 proxy\_set\_header X-Forwarded-Port $server\_port;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 proxy\_set\_header Upgrade $http\_upgrade;  
 proxy\_set\_header Connection "upgrade";  
 }  
 # WebSocket 服务  
 location /webSocket {  
 proxy\_redirect off;  
 proxy\_pass http://localhost:端口/webSocket;  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Upgrade $http\_upgrade;  
 proxy\_set\_header Connection "upgrade";  
 proxy\_set\_header Host $http\_host;  
 proxy\_set\_header X-Real-IP $remote\_addr;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 proxy\_connect\_timeout 60s;  
 proxy\_read\_timeout 86400s;  
 proxy\_send\_timeout 60s;  
 }  
 # 头像  
 location /avatar {  
 proxy\_pass http://localhost:端口;  
 }  
 # 文件  
 location /file {  
 proxy\_pass http://localhost:端口;  
 }  
 error\_page 404 /index.html;  
}

**对网页版前端网站Nginx配置**

server {  
 listen 80;  
 server\_name 域名;  
  
 listen 443 ssl;  
 #ssl on;  
 ssl\_certificate /etc/letsencrypt/live/域名/fullchain.pem;  
 ssl\_certificate\_key /etc/letsencrypt/live/域名/privkey.pem;  
  
 #charset koi8-r;  
  
 #access\_log logs/host.access.log main;  
  
 location / {  
 root pc端打包dist目录;  
 index index.html index.htm;  
 error\_log logs/f\_error.log;  
 }  
  
 location /api/ {  
 proxy\_pass http://localhost:端口/api;  
 proxy\_connect\_timeout 300; #单位秒  
 proxy\_send\_timeout 300; #单位秒  
 proxy\_read\_timeout 300; #单位秒  
  
 proxy\_set\_header Host $host;  
 proxy\_set\_header X-Real-IP $remote\_addr;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 }  
 location /mqtt {  
 proxy\_pass http://localhost:端口;  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Upgrade $http\_upgrade;  
 proxy\_set\_header Connection "upgrade";  
 proxy\_set\_header Host $host;  
 }  
 error\_page 404 /index.html;  
}

##### **移动端网站Nginx配置**

server {  
 listen 80;  
 server\_name 域名;  
  
 listen 443 ssl;  
 #ssl on;  
 ssl\_certificate /etc/letsencrypt/live/域名/fullchain.pem;  
 ssl\_certificate\_key /etc/letsencrypt/live/域名/privkey.pem;  
  
 #charset koi8-r;  
  
 #access\_log logs/host.access.log main;  
  
 location / {  
 root 移动端打包dist目录;  
 index index.html index.htm;  
 error\_log logs/f\_error.log;  
 }  
  
 location /api/ {  
 proxy\_pass http://localhost:端口/api;  
 proxy\_connect\_timeout 300; #单位秒  
 proxy\_send\_timeout 300; #单位秒  
 proxy\_read\_timeout 300; #单位秒  
  
 proxy\_set\_header Host $host;  
 proxy\_set\_header X-Real-IP $remote\_addr;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 }  
  
 location /mqtt {  
 proxy\_pass http://localhost:端口;  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Upgrade $http\_upgrade;  
 proxy\_set\_header Connection "upgrade";  
 proxy\_set\_header Host $host;   
 }  
 error\_page 404 /index.html;  
}

##### 3.登录宝塔面板安装Redis服务：

进入宝塔面板后台管理界面→选择软件管理→运行环境→Redis→点击安装，等待完成。 然后要使用PHP连接Redis的话，还需在PHP设置中安装Redis扩展：

找到/www/server/redis/中的“redis.conf”文件，打开编辑，找到“requirepass foobared”这行，将#注释符去掉 也可不去掉 默认没有密码 。

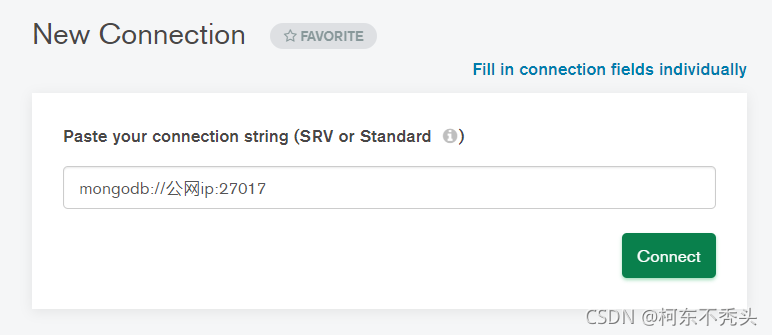
接下来，打开PHP对应的设置，选择“安装扩展” → redis → 安装。

Redis扩展安装好后，重启即可。

如果您想开启Redis服务，则可在宝塔面板中的Redis管理界面，停止或重启当前Redis服务和重新加载Redis配置文件。

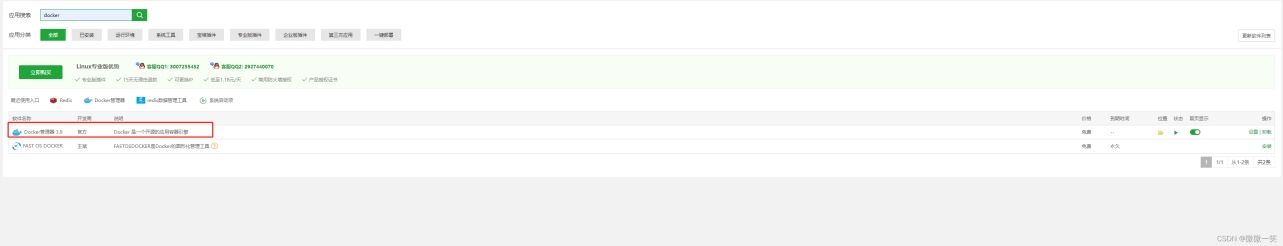
亦可以进SSH输入以下命令来完成重载、重启等操作。

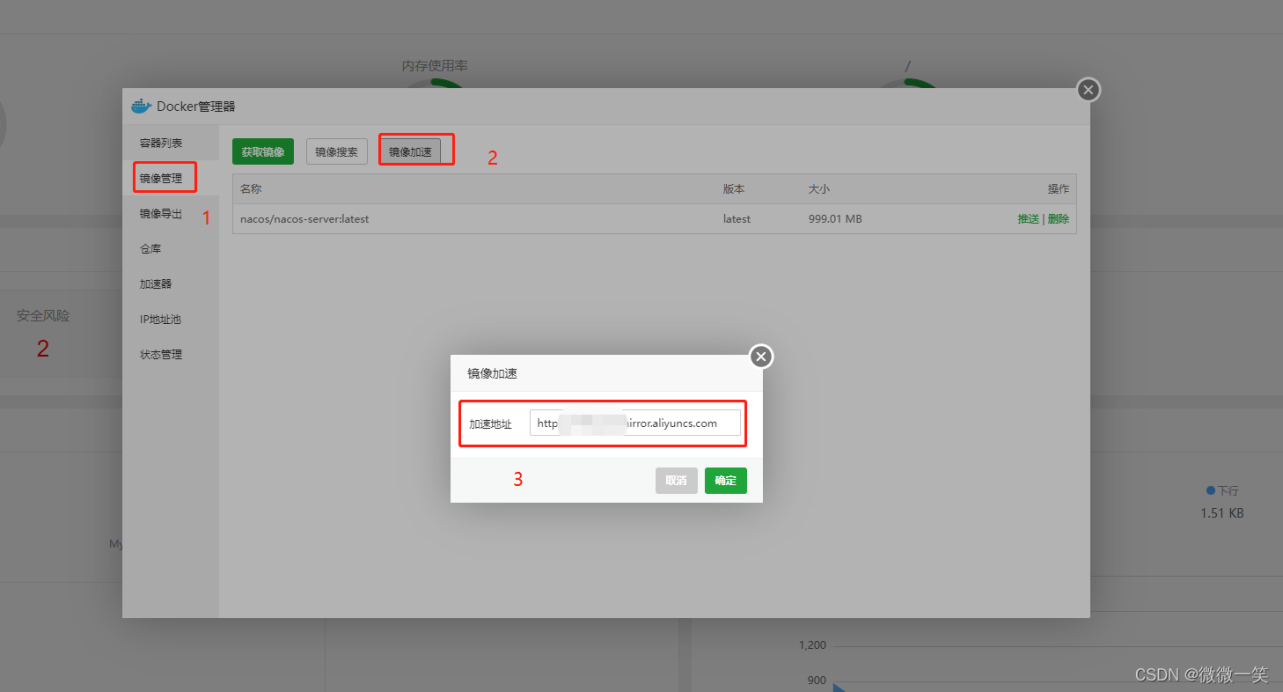
##### 4.登录宝塔安装mongodb

1.点击软件商店，搜索mongodb进行安装 2.找到安装好的mongodb进行设置，将配置文件bindIp改为0.0.0.0 img15.image 阿里云服务器安全组入方向开放端口27017 img16.image 防火墙打开端口27017 img15.image 浏览器输入http://公网ip:27017 出现“It looks like you are trying to access MongoDB over HTTP on the native driver port.”说明配置成功 二、使用MongoDB Compass可视化工具连接数据库 无密码连接 

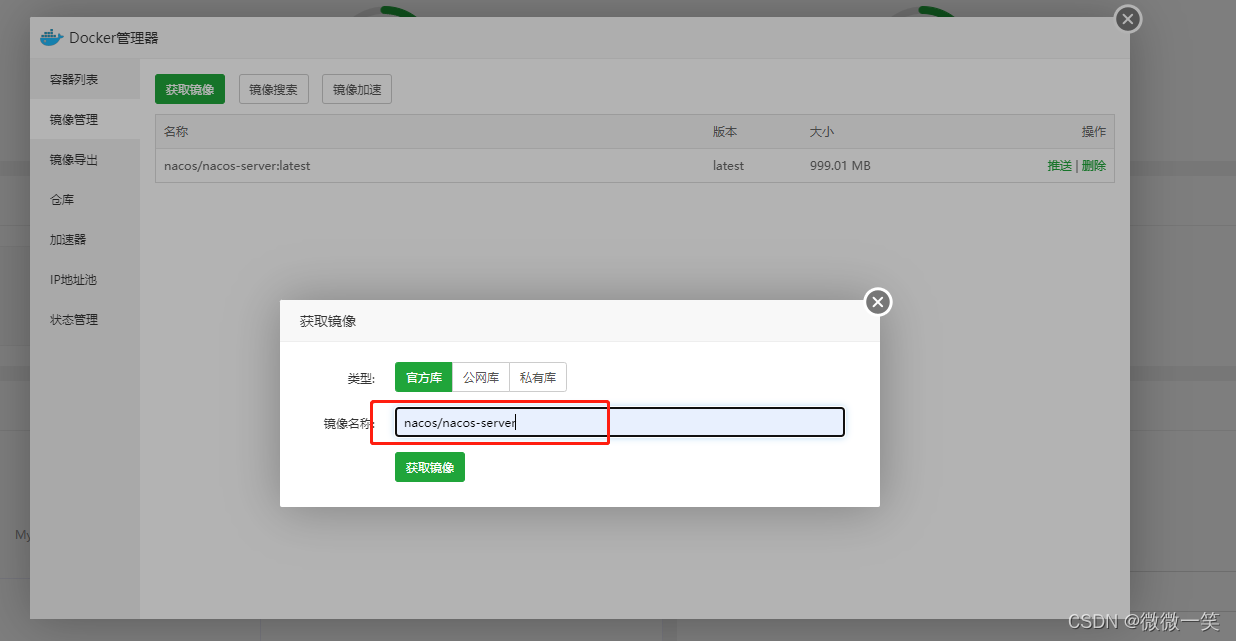
注：记得导入移动端后台数据库文件到mongodb

##### 5.登录宝塔安装docker

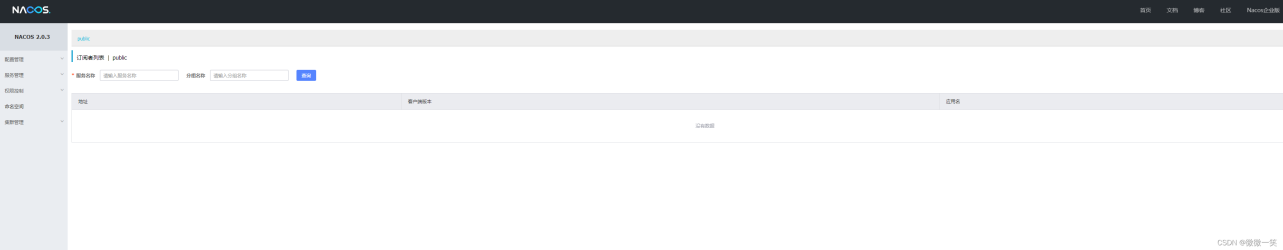
在商店搜索docker 

打开docker管理器 使用阿里云的镜像加速服务: https://cr.console.aliyun.com/cn-hangzhou/instances/mirrors 

##### 6.获取nacos镜像

  容器下载完毕, 不知道怎么安装没关系, 看下边 ↓ 启动nacos镜像 打开宝塔面板上的终端执行以下命令: 命令中已设置了开机自启动

docker run  
–name nacos -d  
-p 8848:8848  
–privileged=true  
–restart=always  
-e JVM\_XMS=256m  
-e JVM\_XMX=256m  
-e MODE=standalone  
-e PREFER\_HOST\_MODE=hostname  
nacos/nacos-server:latest

安装成功 在浏览器上输入访问地址：http://IP地址:8848/nacos，测试是否安装成攻 账号和密码默认是: nacos/nacos 

注：记得导入nacos配置文件和修改项目nacos配置文件修改ip和端口

Nacos配置如下：

######## common.yaml ##########

spring:

data:

mongodb:

uri: mongodb://账号:密码@ip:端口号/chat-server?authSource=admin

user:

jwt: jwt密钥

exp: 用户经验增长值

tokenExpiry: 86400000

app:

uploadUrl: seaweedFS服务的ip:端口号/submit

inviteUrl: ""

publishUrl: chat-node-server服务的ip:端口号/api/v1/publish

smsbao:

username:短信宝用户名

password: 短信宝密码

url: http://api.smsbao.com/sms

content: 【IM】您的验证码是 %s。如非本人操作，请忽略本短信

######## gateway-routes.yaml ##########

spring:

application:

name: gateway-service

cloud:

gateway:

routes:

- id: chat-server

uri: lb://chat-server

predicates:

- Path=/api/\*\*

auth:

ignore:

paths :

- /api/v1/user/register

- /api/v1/user/login

- /api/v1/user/registerMobile

- /api/v1/user/loginMobile

- /api/v1/user/getUserByIds

- /api/v1/user/getCode

- /api/v1/user/checkCode/\*\*

- /api/v1/user/checktoken

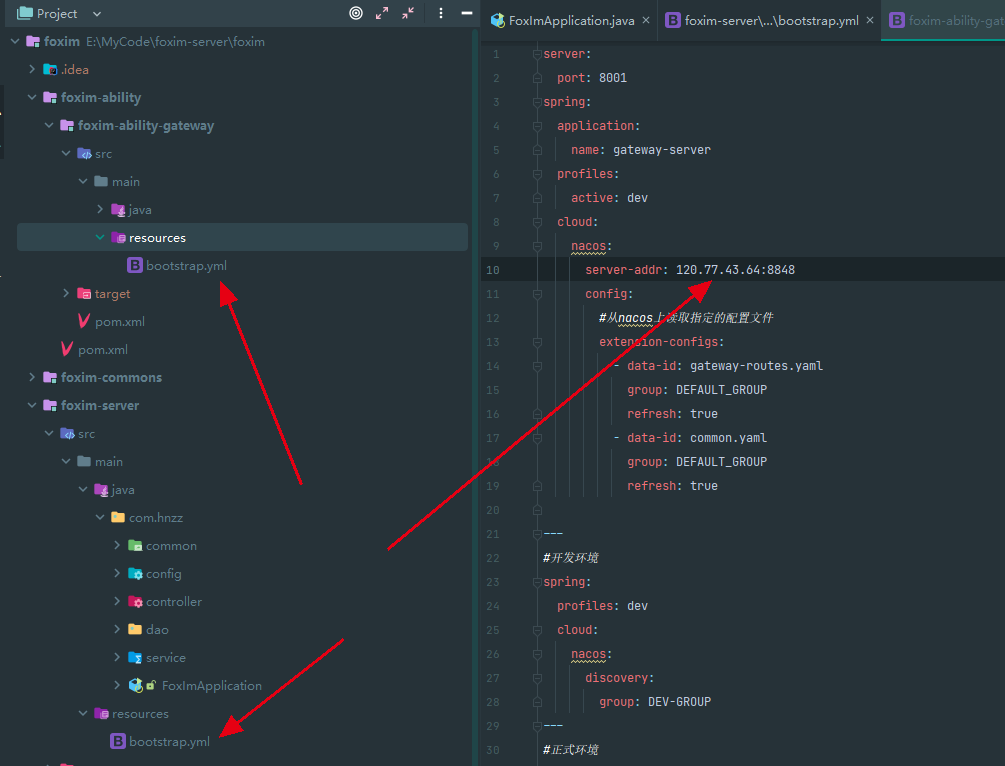
- /api/vl/activities/save

- /api/v1/ids/get

- /api/v1/sms/\*\*

admins:

- /api/v1/admin/\*\*



##### 7.安装seaweedf：

方式一、docker部署seaweedf 总的来说需要两个文件，如下： img30.image 1、compose.yml

需要注意：network\_mode 模式、ip为宿主机ip

version: ‘3.3’

services: master: network\_mode: “host” image: chrislusf/seaweedfs # use a remote image command: “master -ip=172.0.16.143 -ip.bind=172.0.16.143 -metricsPort=9324” volume: network\_mode: “host” image: chrislusf/seaweedfs # use a remote image command: ‘volume -mserver=“172.0.16.143:9333” -ip.bind=172.0.16.143 -port=8080 -metricsPort=9325’ depends\_on: - master filer: network\_mode: “host” image: chrislusf/seaweedfs # use a remote image command: ‘filer -master=“172.0.16.143:9333” -ip.bind=172.0.16.143 -metricsPort=9326’ tty: true stdin\_open: true depends\_on: - master - volume s3: network\_mode: “host” image: chrislusf/seaweedfs # use a remote image command: ‘s3 -filer=“172.0.16.143:8888” -ip.bind=172.0.16.143 -metricsPort=9327’ depends\_on: - master - volume - filer webdav: network\_mode: “host” image: chrislusf/seaweedfs # use a remote image command: ‘webdav -filer=“172.0.16.143:8888”’ depends\_on: - master - volume - filer

prometheus:

image: prom/prometheus:v2.21.0

ports:

​ 9000:9090

volumes:

​ ./prometheus:/etc/prometheus

command: –web.enable-lifecycle –config.file=/etc/prometheus/prometheus.yml

depends\_on:

​ s3

2、install.sh docker-compose -f seaweedfs-compose.yml -p seaweedfs up

方式二、手动安装 SeaWeedfs安装与启动 Linux安装SeaWeedFS mkdir -p /myapp/seaweedfs cd /myapp/seaweedfs 访问Releases · seaweedfs/seaweedfs · GitHub下载linux\_amd64.tar.gz文件,使用XShell工具下载到/myapp/seaweedfs路径。

解压tar.gz 文件：

tar -xzvf linux\_amd64.tar.gz 生成一个weed可执行文件，用于操作master和volume等。 创建数据卷根目录以及各个数据卷目录、master数据目录和filter目录

mkdir -p /myapp/seaweedfs/data/master mkdir -p /myapp/seaweedfs/data/vola mkdir -p /myapp/seaweedfs/data/volb mkdir -p /myapp/seaweedfs/data/volc mkdir -p /myapp/seaweedfs/data/vold mkdir -p /data/weedfs\_data/filer SeaWeedfs启动 启动SeaWeedfs可使用nohup命令进行后台运行并且将日志输出到文件中。

启动mster，-defaultReplication=“001”表示相同机架存储一份副本，而且如果一个卷服务器没在相同的rack开启一个vol就会报错:

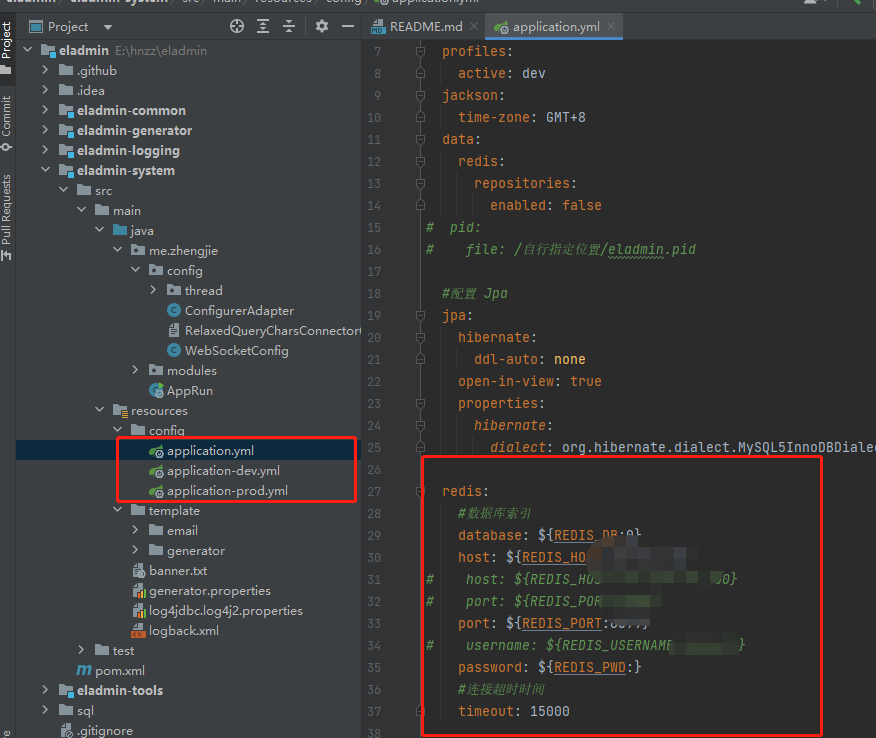
nohup ./weed -v=3 master -mdir=/myapp/seaweedfs/data/master -defaultReplication=“001” >> /myapp/seaweedfs/data/master/wdfsmaster.log & 启动volume，两个卷服务器放在同一个机架:

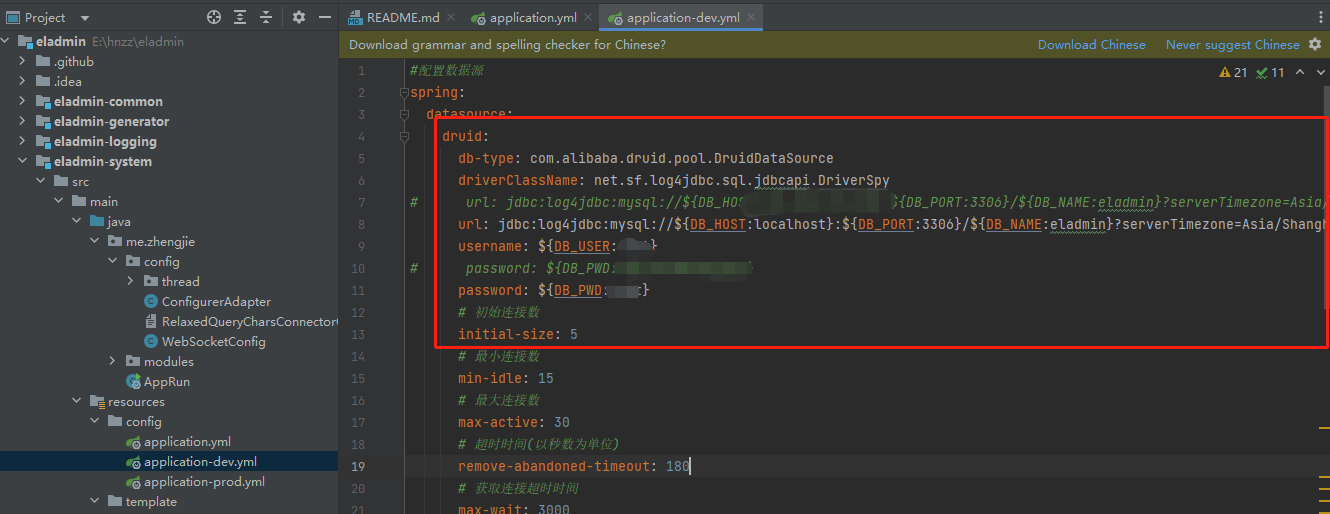
nohup ./weed -v=3 volume -port=8081 -dir=/myapp/seaweedfs/data/vola -mserver=localhost:9333 -rack=rack1 >> /myapp/seaweedfs/data/vola/wefsvola.log & ​ nohup ./weed -v=3 volume -port=8082 -dir=/myapp/seaweedfs/data/volb -mserver=localhost:9333 -rack=rack2 >> /myapp/seaweedfs/data/volb/wefsvolb.log & ​ nohup ./weed -v=3 volume -port=8083 -dir=/myapp/seaweedfs/data/volc -mserver=localhost:9333 -rack=rack3 >> /myapp/seaweedfs/data/volc/wefsvolc.log & ​ nohup ./weed -v=3 volume -port=8084 -dir=/myapp/seaweedfs/data/vold -mserver=localhost:9333 -rack=rack1 >> /myapp/seaweedfs/data/vold/wefsvold.log & 开启filer服务器:

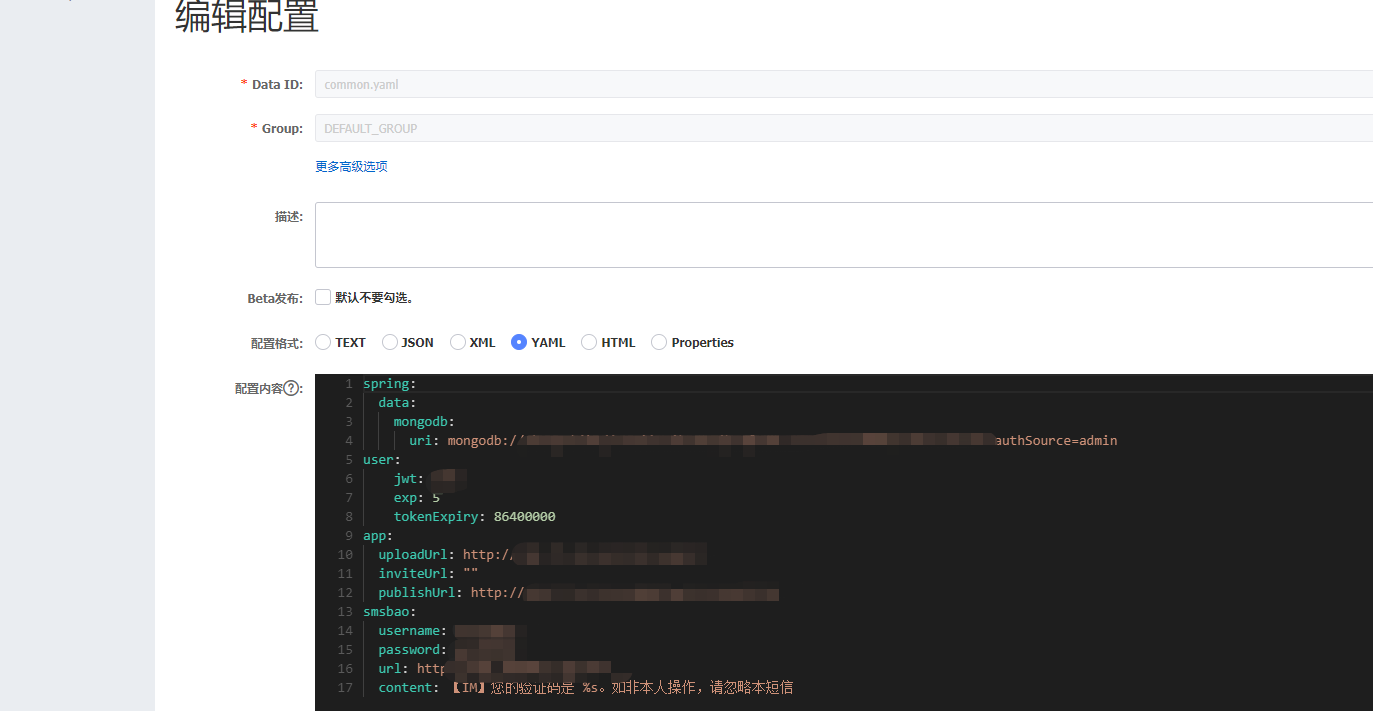
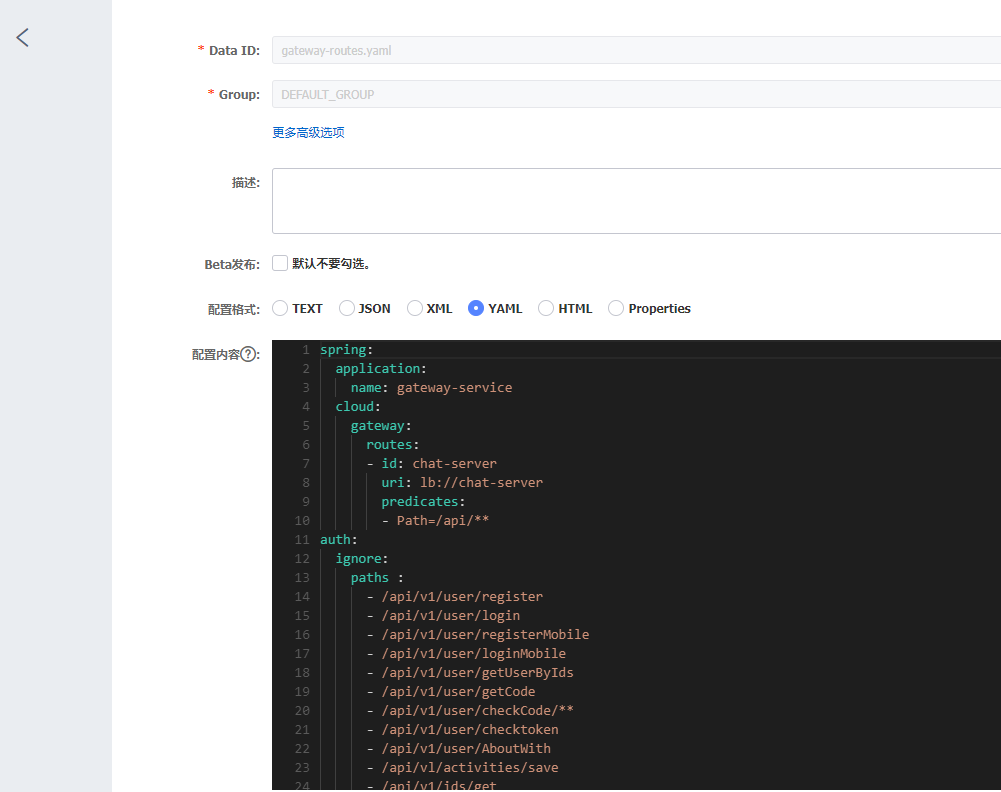
nohup ./weed -v=3 filer -port=8888 -master=localhost:9333 >> /myapp/seaweedfs/data/filer/filer.log & 开启挂载服务器：

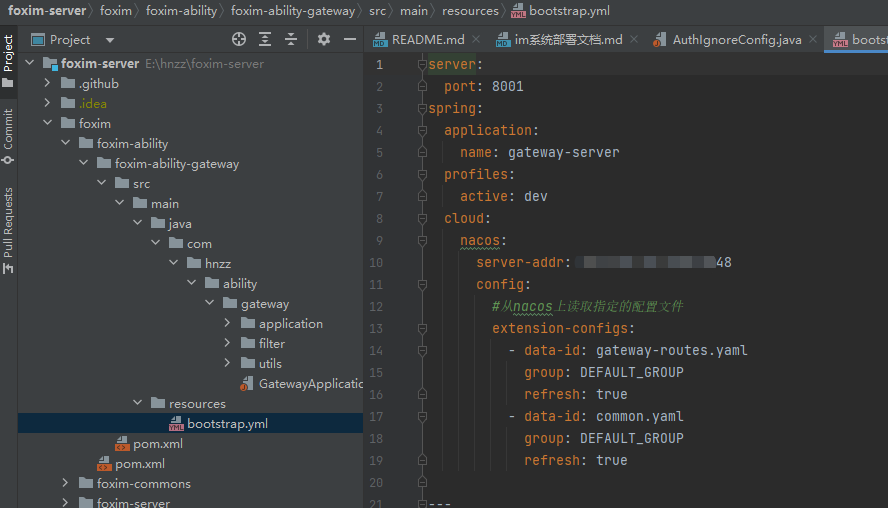
nohup ./weed mount -filer=localhost:8888 -dir=/myapp/seaweedfs/data/filermount/data -filer.path=/ >> /myapp/seaweedfs/data/filermount/wefsmount.log &

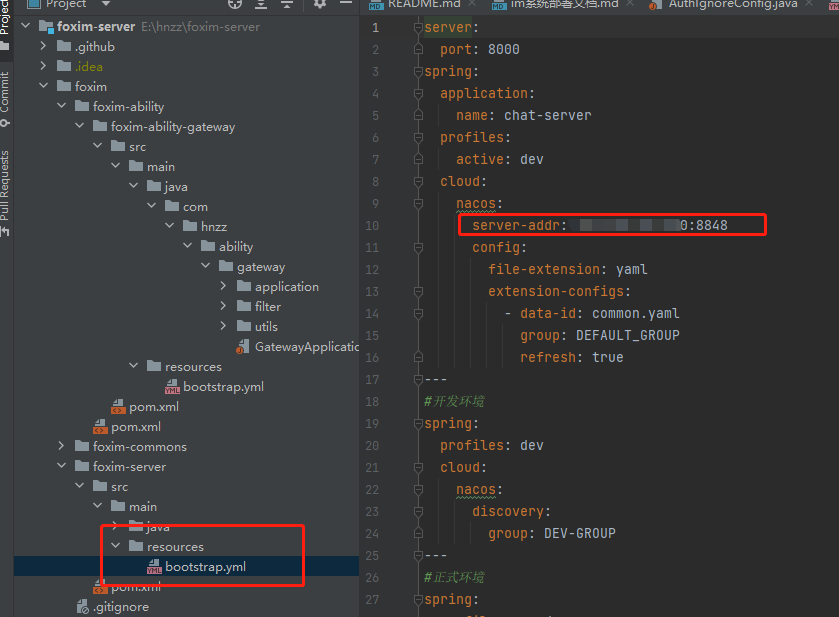
# 配置域名证书

后端admin修改图示： 



移动端nacos修改图示： common.yaml  gateway-routes.yaml 

移动端后台源码修改图示： 



移动端client修改图示： 

后端ui 移动端client分别进行npm run build打包

java源码打包使用maven clean install -DskipTests

上传dist前端打包文件到宝塔域名对应的目录下

1. 上传foxim-server生成的两个jar包到移动端client目录下
2. 上传foxim-admin生成的jar包到后台域名目录下

4.启动命令如下：

nohup java -Xms256m -Xmx256m -Xmn128m -jar /www/jar/foxim-ability-gateway.jar > /www/jar/foxim-ability-gateway.jar\_log.txt 2>&1

nohup java -Xms256m -Xmx256m -Xmn128m -jar /www/jar/foxim-server.jar > /www/jar/foxim-server.jar\_log.txt 2>&1 &

nohup java -Xms256m -Xmx256m -Xmn128m -jar /www/jar/foxim-admin.jar > /www/jar/foxim-admin.jar\_log.txt 2>&1 &