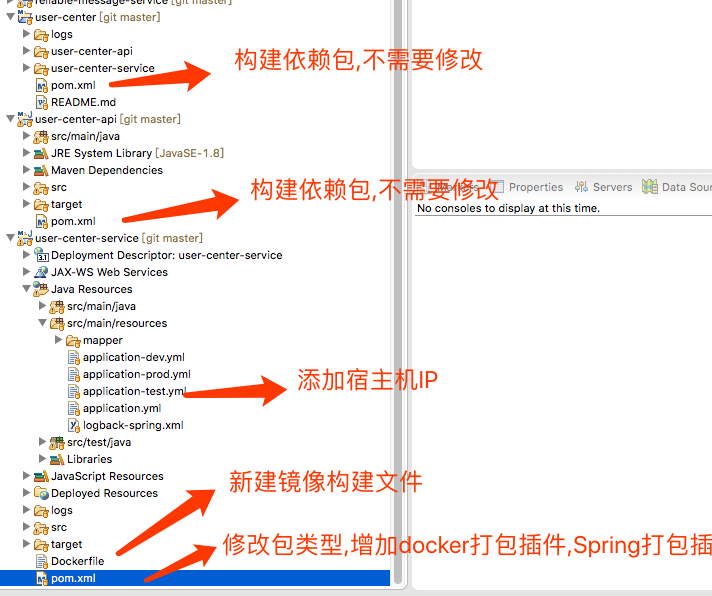
## 一、需要修改的文件

文档以user-center为例,先介绍下需要修改的文件



单模块项目可以把想当作用户中心的user-center-service模块

## 二、pom.xml修改

1、打包方式修改为jar:<packaging>jar</packaging>

2、添加docker构建插件及Spring打包插件:

<plugin>

<groupId>com.spotify</groupId>

<artifactId>docker-maven-plugin</artifactId>

<version>0.4.12</version>

<!--docker镜像相关的配置信息 -->

<configuration>

<forceTags>true</forceTags>

<!--镜像名，这里用工程名 -->

<imageName>120.77.213.135:5000/${project.artifactId}</imageName>

<!--Dockerfile文件所在目录 -->

<dockerDirectory>${project.basedir}</dockerDirectory>

<!--TAG,这里用工程版本号 -->

<imageTags>

<imageTag>${project.version}</imageTag>

</imageTags>

<!--构建镜像的配置信息 -->

<resources>

<resource>

<targetPath>/</targetPath>

<directory>${project.build.directory}</directory>

<include>${project.build.finalName}.jar</include>

</resource>

</resources>

</configuration>

</plugin>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<executions>

<execution>

<goals>

<goal>repackage</goal>

</goals>

</execution>

</executions>

</plugin>

## 三、修改application-test.properties

#宿主机IP,与容器启动参数--net="bridge"配合解决docker在注册中心注册虚拟IP导致不能访问的问题

增加eureka.instance.ip-address: 192.168.0.186

## 四、安装docke(本地可以不安装)

测试环境服务器上已经安装,本地测试构建可以安装一个docker在本地,也可以直接在Jenkins上配置构建任务测试镜像构建,配置方式在后面描述.

## 五、增加Dockerfile

在需要构建镜像的目录下增加文件(user-center放在user-center-service下),文件名为Dockerfile,内容:

FROM openjdk:8-jdk-alpine

VOLUME /tmp

COPY target/user-center-0.0.1-SNAPSHOT.jar  app.jar

ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/app.jar"]

红色部分修改为镜像jar包相对于Dockerfile文件的相对路径

## 六、构建镜像

执行mvn clean package docker:build -Dmaven.test.skip -DpushImage -U 构建并上传镜像到镜像私服120.77.213.135:5000

## 七、登录私服服务器查看镜像是否上传成功

在135服务器执行docker images

## 八、使用构建出的镜像启动容器

docker run -d -p 8401:8401 -v /data/docker\_files/tmp/:/tmp --name=user-center 120.77.213.135:5000/user-center-service --net="bridge" --spring.profiles.active=test

-d 为后台运行

-p为从映射出的端口,配置项目直接访问的端口

-v 挂载宿主机的目录到容器

--name 给运行的容器一个指定的名字

120.77.213.135:5000/user-center-service 镜像

--net="bridge" 网络桥接模式,与eureka.instance.ip-address配合解决docker在注册 中心注册虚拟IP导致不能访问的问题

--spring.profiles.active 启动环境

启动后命令返回一个容器id,或者用docker ps查看当前运行的容器信息,里面包含容器ID.

用docker logs -f -t --tail 10 + 容器id 可以查看日志最后10行

## 九、重新部署步骤

重新部署需要移除旧容器实例,移除前需要先停止.为了避免镜像占用太多空间所以现在构建也先移除旧镜像然后上传新镜像,移除镜像前要先把容器实例移除.

容器停止:docker stop + 容器名称(run命令指定的--name)

容器移除:docker rm + 容器名称

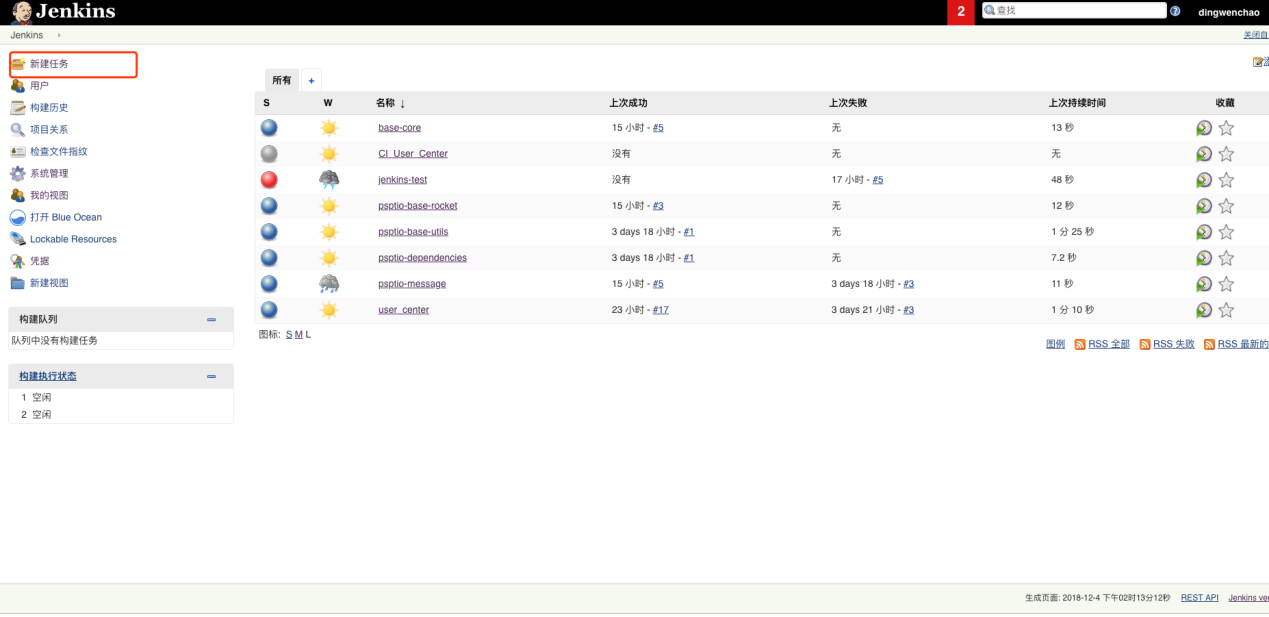
删除旧镜像:docker rmi + 镜像名称:tag

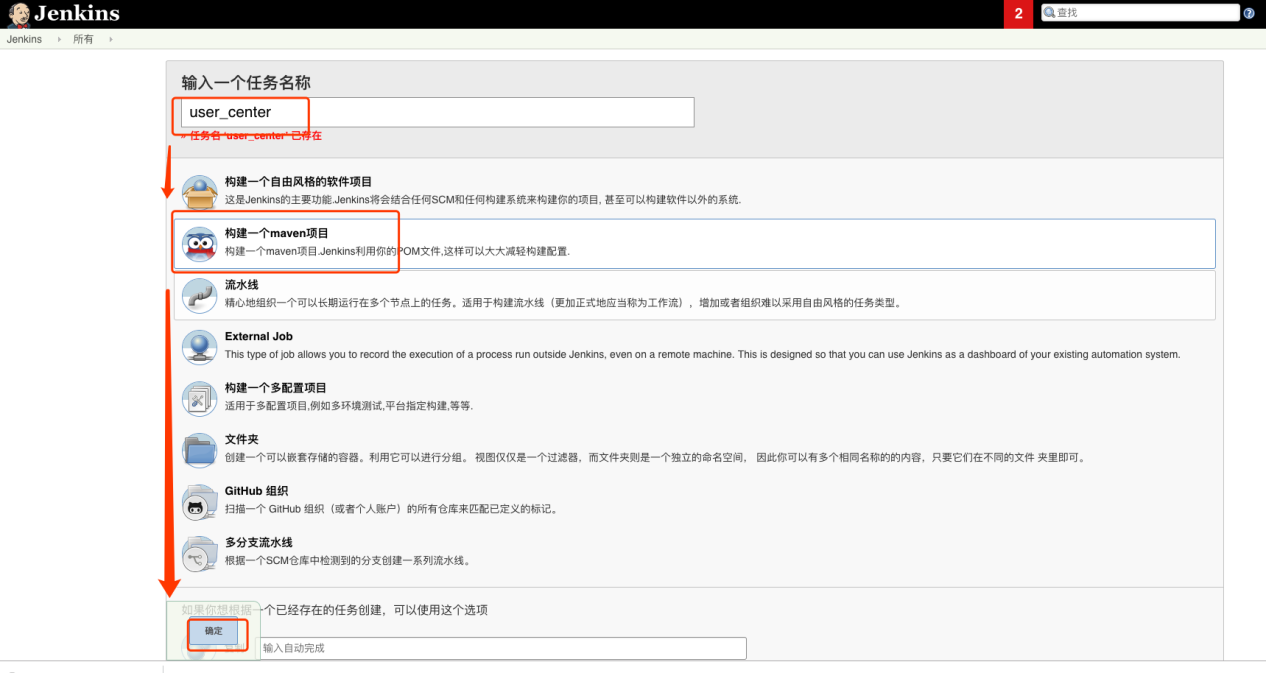
构建/上传新镜像

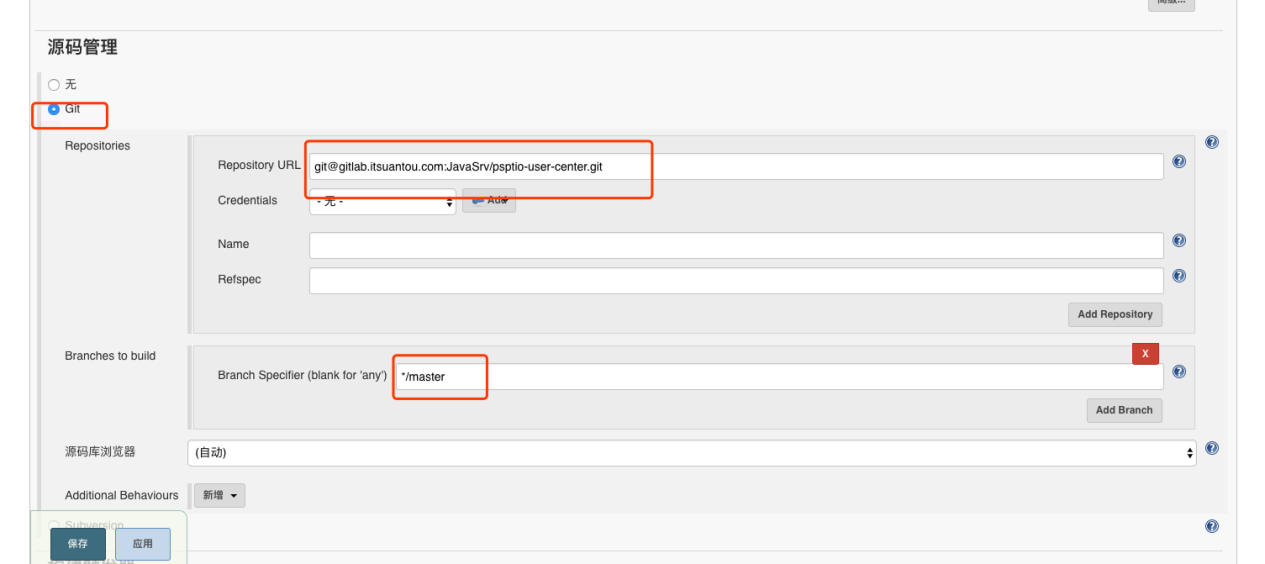
启动容器

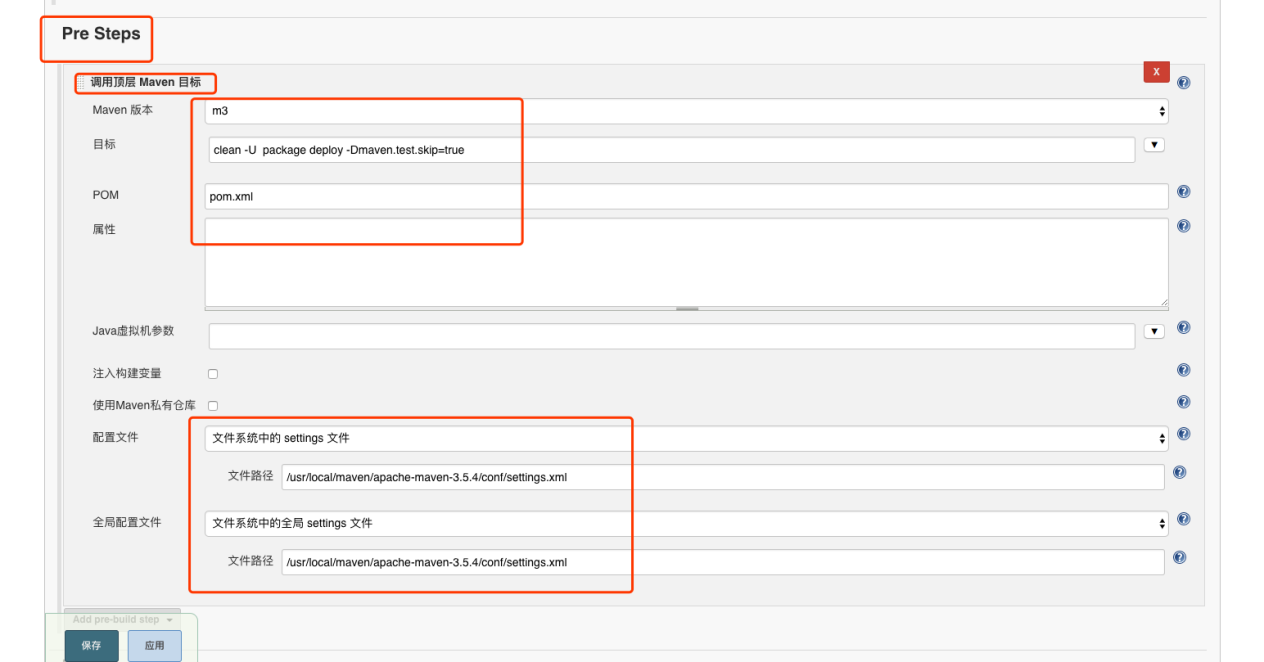
## Jenkins构建部署配置

jenkins地址:http://120.77.213.135:8888/





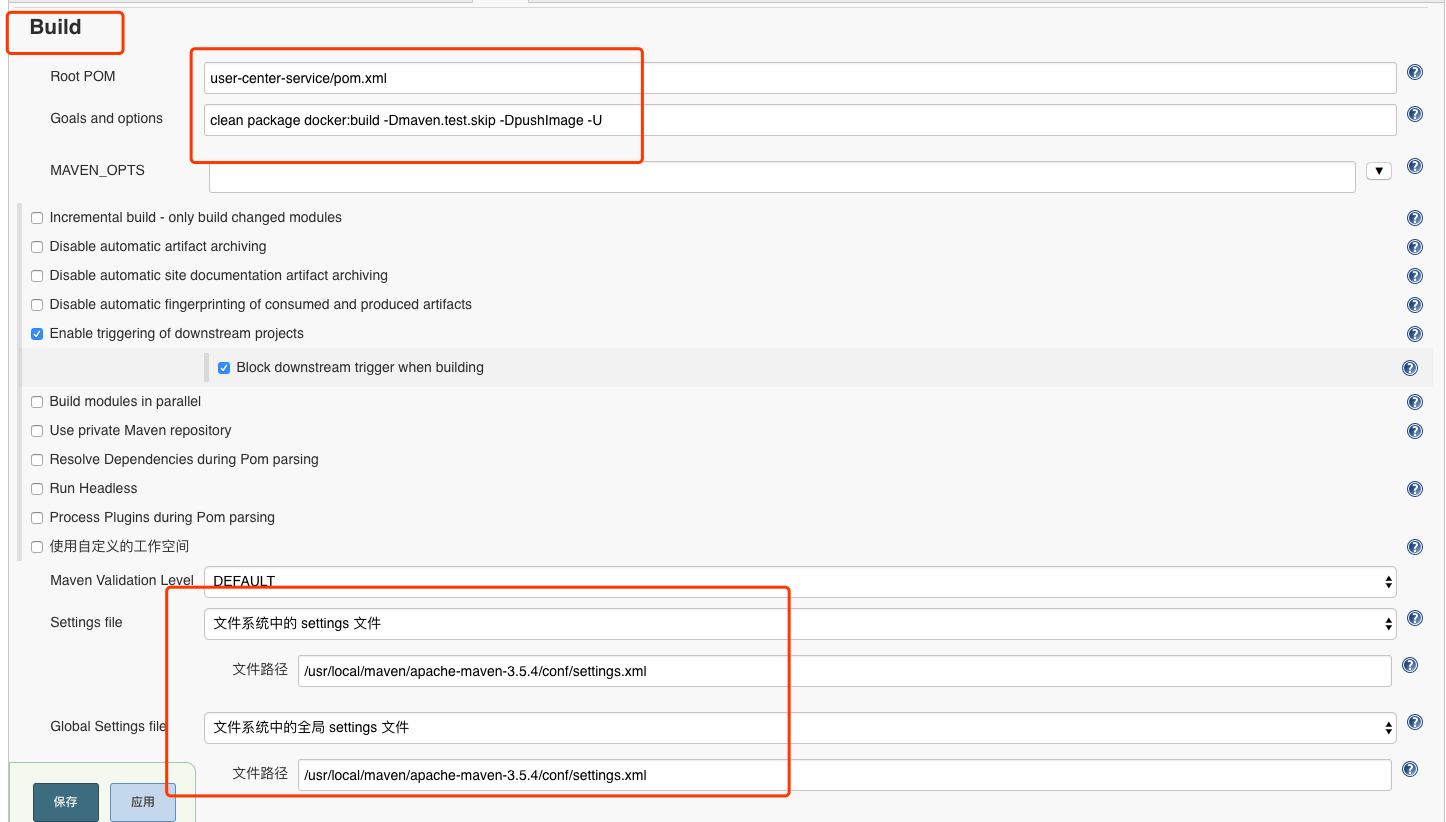




多模块项目需要配置pre steps,用来构建发布所有依赖的API工程,pom.xml是父级pom.

clean package deploy -Dmaven.test.skip=true -U

/usr/local/maven/apache-maven-3.5.4/conf/settings.xml



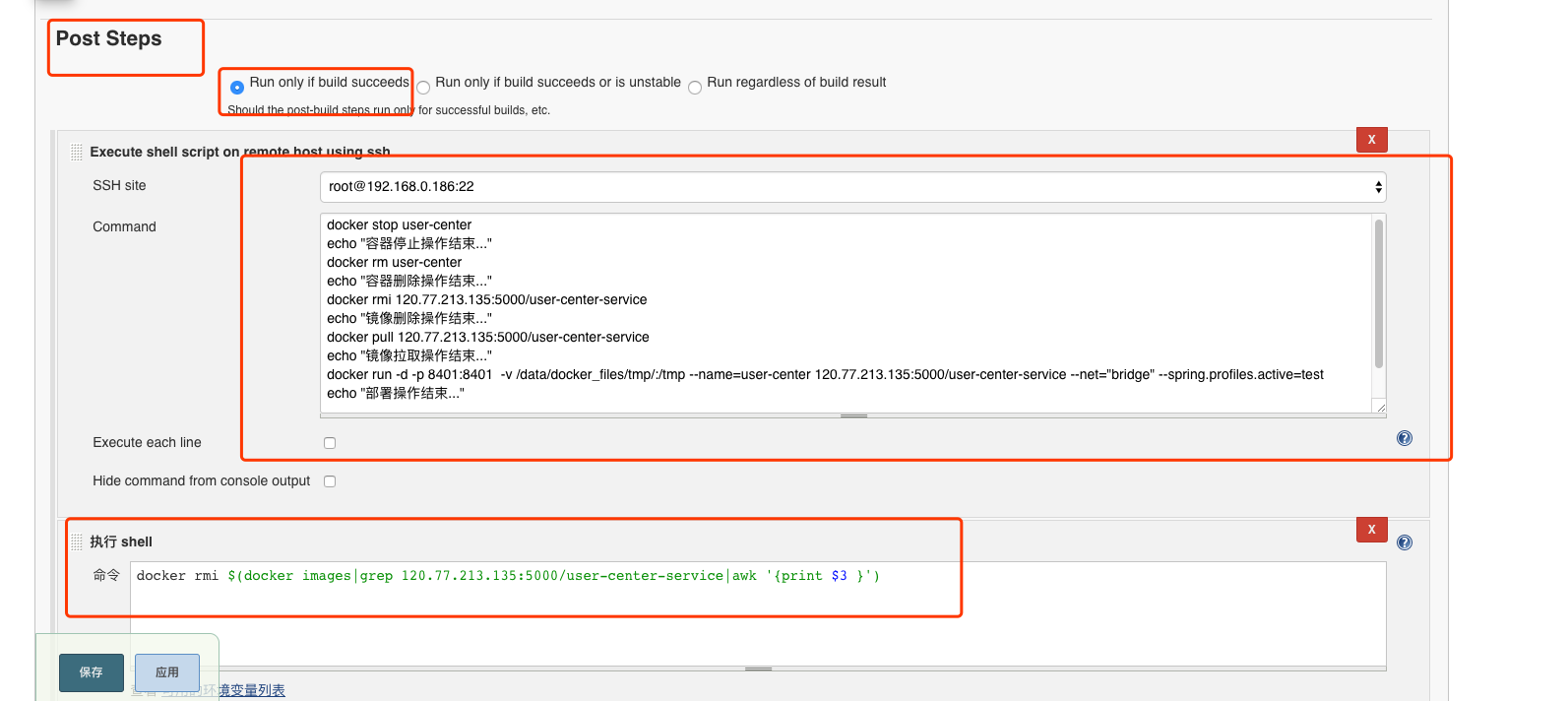
构建上传镜像到私库.

因为user-center的包是在service模块中打的,docker插件也配置在这个pom中,所以配置的user-center-service/pom.xml

clean package docker:build -Dmaven.test.skip -DpushImage -U

其中docker:build是构建镜像命令,-Dmaven.test.skip默认true,也可以写-Dmaven.test.skip=true.

这一步配置完成后即可点击保存.返回视图页面点立即构建,当镜像可以成功上传到135服务器上到私服中时,配置下面到部署流程.



docker stop user-center

echo "容器停止操作结束..."

docker rm user-center

echo "容器删除操作结束..."

docker rmi 120.77.213.135:5000/user-center-service

echo "镜像删除操作结束..."

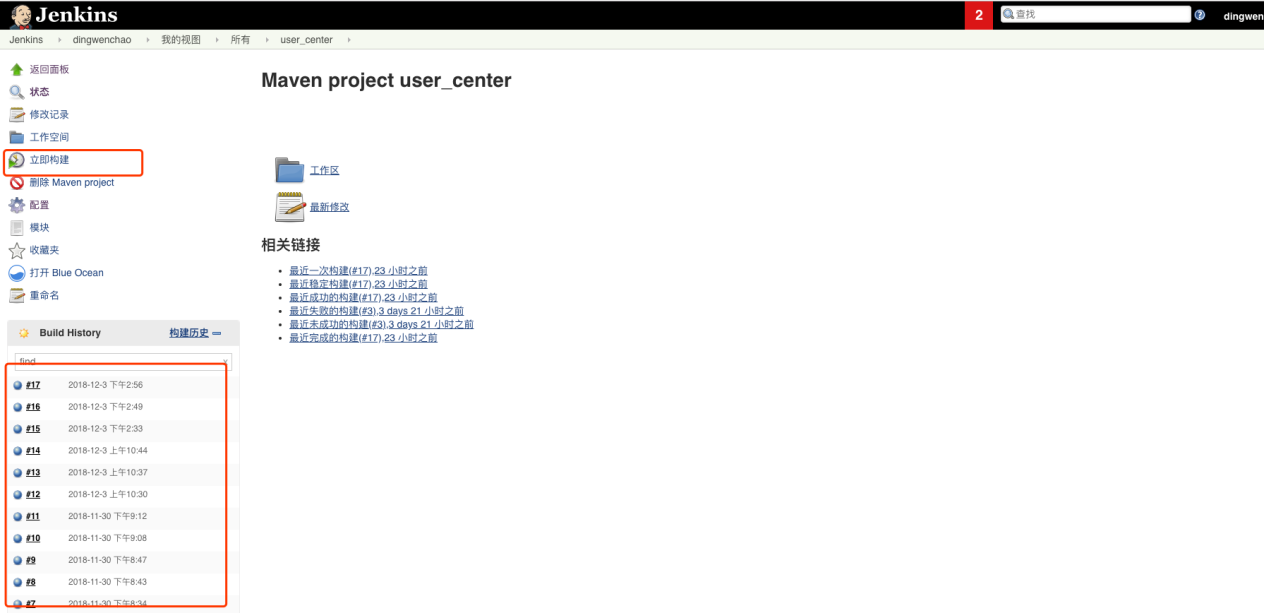
docker pull 120.77.213.135:5000/user-center-service

echo "镜像拉取操作结束..."

docker run -d -p 8401:8401 -v /data/docker\_files/tmp/:/tmp --name=user-center 120.77.213.135:5000/user-center-service --net="bridge" --spring.profiles.active=test

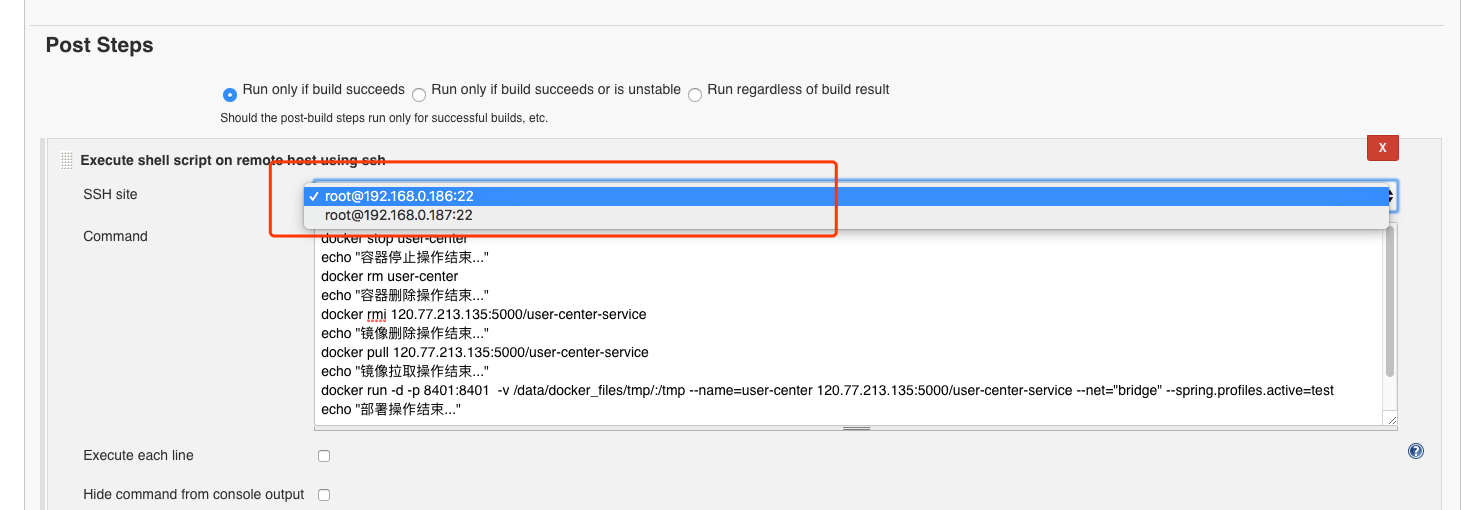
echo "部署操作结束..."

这里的输出只表示上一步操作完成,不代表成功状态.如果构建部署中出现问题需要到控制台日志输出查看,并且可以查看历史构建记录



docker rmi $(docker images|grep 120.77.213.135:5000/user-center-service|awk 'NR==1{print $3 }')

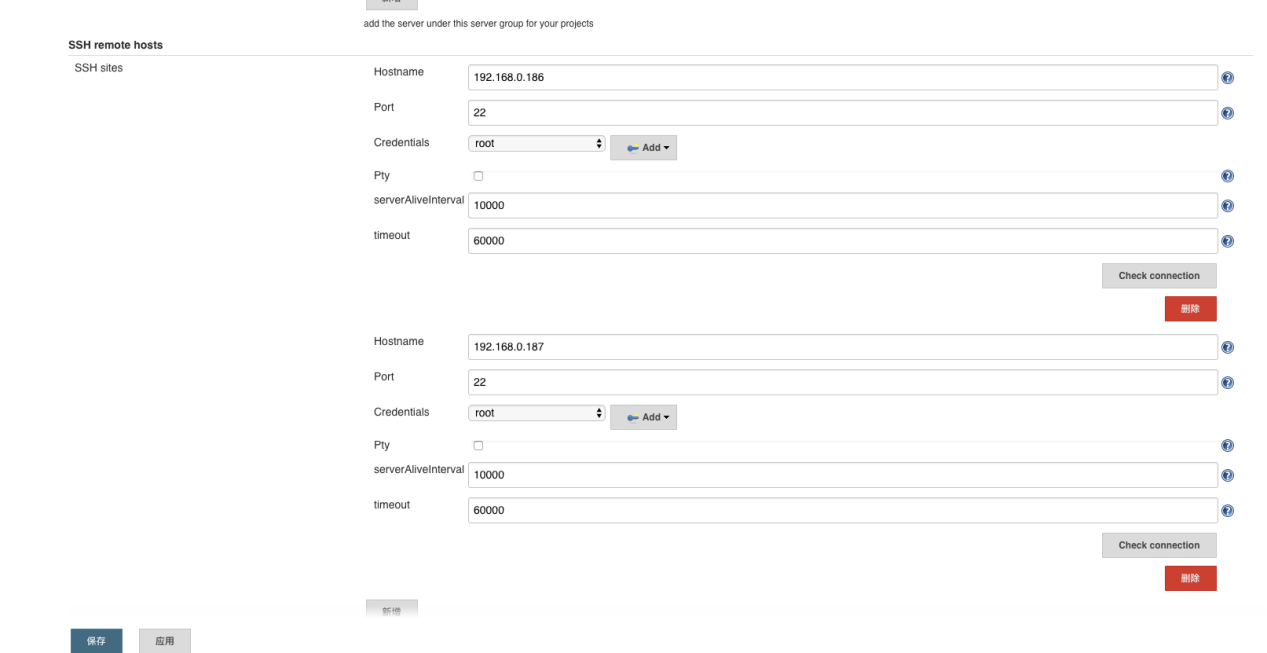
删除旧镜像



调试通过后部署只需要点击立即构建,就可以自动构建依赖-》构建镜像-》上传镜像-》ssh远程pull镜像-》启动容器

ssh连接目前配置了两个,分别是内网186和187,这部分暂时不需要新增.现有配置:

系统管理-》系统设置:



1. **同步docker时间与宿主机时间**

Dockerfile 中增加环境变量: ENV TZ='Asia/Shanghai'

启动命令增加: -v /etc/timezone:/etc/timezone:ro -v /etc/localtime:/etc/localtime:ro

1. **挂载日志目录到宿主机**

修改logbook文件:<property name="logback.logdir" value="/tmp/share/logs"/>

启动命令增加: -v /data/docker\_files:/tmp/share