

服务器配置

阿里云云服务器

- Centos 7
- 1核2G 1M带宽 40G云盘
- ip:47.94.129.13

Jenkins安装

安装

- 在服务器上安装jdk11

```
yum install java-11-openjdk -y
java --version # 判断jdk是否安装成功
```

- 在jenkins官网上安装最新稳定版jenkins

```
# 下载repo
sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
# yum下载
yum install epel-release # repository that provides 'daemonize'
yum install java-11-openjdk-devel
yum install jenkins
# 启动jenkins
systemctl start jenkins
systemctl enable jenkins
```

- `vi /etc/sysconfig/jenkins` 并修改端口为8082

```

# $JENKINS_HOME location. Do not enable this, "true", unless
# you know what you're doing. See JENKINS-23273.
#
#JENKINS_INSTALL_SKIP_CHOWN="false"

## Type: string
## Default: "-Djava.awt.headless=true"
## ServiceRestart: jenkins
#
# Options to pass to java when running Jenkins.
#
JENKINS_JAVA_OPTIONS="-Djava.awt.headless=true"

## Type: integer(0:65535)
## Default: 8080
## ServiceRestart: jenkins
#
# Port Jenkins is listening on.
# Set to -1 to disable
#
JENKINS_PORT="8082"

## Type: string
## Default: ""
## ServiceRestart: jenkins
#
# IP address Jenkins listens on for HTTP requests.
# Default is all interfaces (0.0.0.0).
#
JENKINS_LISTEN_ADDRESS=""

## Type: integer(0:65535)
## Default: ""
## ServiceRestart: jenkins

```

- 访问8082端口,初始化服务

配置

- 初始化账户
- 在jenkins全局配置中配置jdk、maven、git工具（maven、git的安装过程略）
- jenkins插件配置(除默认推荐外的)

blue ocean # pipeline的一个ui
ssh # 远程连接
nodejs # 插件

docker安装

```

sudo yum remove docker \
    docker-client \
    docker-client-latest \
    docker-common \
    docker-latest \
    docker-latest-logrotate \
    docker-logrotate \
    docker-engine

sudo yum install -y yum-utils
sudo yum-config-manager \
    --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo

```

```
sudo yum install docker-ce docker-ce-cli containerd.io
```

```
docker version
```

```
systemctl start docker && systemctl enable docker
```

mysql安装

```
wget https://dev.mysql.com/get/mysql80-community-release-el7-1.noarch.rpm
```

```
#安装yum源
```

```
yum localinstall mysql80-community-release-el7-1.noarch.rpm
```

```
#更新yum源
```

```
yum clean all
```

```
yum makecache
```

```
#开始安装MySQL
```

```
yum install mysql-community-server
```

```
systemctl start mysqld.service
```

```
systemctl enable mysqld.service
```

```
cat /var/log/mysqld.log | grep passwd
```

```
# 登录
```

```
mysql -uroot -p
```

```
# 修改密码
```

```
ALTER USER 'root'@'localhost' IDENTIFIED BY 'your_password';
```

```
#远程访问设置
```

```
mysql> use mysql;
```

```
mysql> update user set host='%' where user='root';
```

nodejs安装

```
# As root
```

```
curl -fsSL https://rpm.nodesource.com/setup_16.x | bash
```

```
# 安装nodejs
```

```
yum install -y nodejs
```

```
# 测试安装是否成功
```

```
node -v
```

```
npm -v
```

nginx安装

```
# 添加nginx源
sudo rpm -Uvh http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-
centos-7-0.el7ngx.noarch.rpm

# 安装nginx
sudo yum install -y nginx

# 设置开机自启动
sudo systemctl start nginx.service
sudo systemctl enable nginx.service
```

后端项目使用Jenkins创建流水线实现CI/CD

- 创建流水线项目 backend-pipeline
- 配置该流水线项目的Jenkinsfile
 - 项目代码与Jenkinsfile文件都保存在github仓库上便于版本管理
 - Jenkinsfile文件内容如下:

```
def VERSION = "v0"
def GITREPO = "git@github.com:zqrzxwkr-group/backend.git"
pipeline{
    agent { label 'master' }
    tools {
        jdk 'jdk11'
        maven 'maven3.6.3'
    }
    stages{
        stage("代码克隆"){
            steps{
                sh "cd /var/lib/jenkins/workspace/backend-pipeline && rm
-rf ./*"
                git branch: 'main', credentialsId: '1', url:
"${GITREPO}"
                echo "代码克隆完成"
            }
        }

        stage("代码构建打包"){
            steps{
                sh "cd /var/lib/jenkins/workspace/backend-
pipeline/backend && mvn clean package"
                echo "代码构建打包完成"
            }
        }

        stage("更新服务"){
            steps{
                sh "cd /var/lib/jenkins/workspace/backend-
pipeline/backend"
                sh "chmod +x restart_jar.sh"
                sh "./restart_jar.sh"
                echo "更新服务"
            }
        }
    }
}
```

```
}
```

- 配置在服务器上面重启java服务的脚本 restart_jar.sh,内容如下:

```
export JENKINS_NODE_COOKIE=dontkillme
#!/bin/bash
APP_NAME=$(pwd)/backend/target/backend-0.0.1-SNAPSHOT.jar

#检查程序是否在运行
is_exist(){
    pid=`ps -ef|grep $APP_NAME|grep -v grep|awk '{print $2}'`
    #如果不存在返回1, 存在返回0
    if [ -z "${pid}" ]; then
        return 1
    else
        return 0
    fi
}

#启动方法
start(){
    is_exist
    if [ $? -eq 0 ]; then
        echo "${APP_NAME} is already running. pid=${pid}"
    else
        echo "启动${APP_NAME}"
        nohup java -jar ${APP_NAME} > log.out 2>&1 &
        echo "启动结束"
    fi
}

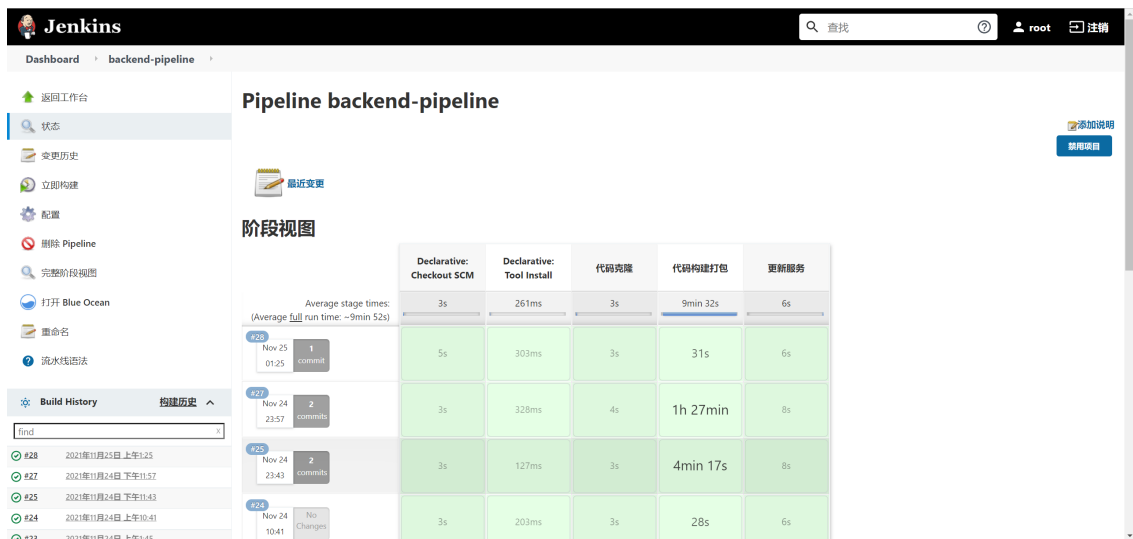
#停止方法
stop(){
    is_exist
    if [ $? -eq "0" ]; then
        kill -9 $pid
    else
        echo "${APP_NAME} is not running"
    fi
}

#输出运行状态
status(){
    is_exist
    if [ $? -eq "0" ]; then
        echo "${APP_NAME} is running. Pid is ${pid}"
    else
        echo "${APP_NAME} is NOT running."
    fi
}

#重启
restart(){
    stop
    sleep 5
    start
}
```

restart

- 配置完成后,每次在本地push代码后,只需在服务器上点击构建就能自动部署了



- 可以访问 <http://47.94.129.13:8081/swagger-ui.html> 来查看接口文档

前端项目部署

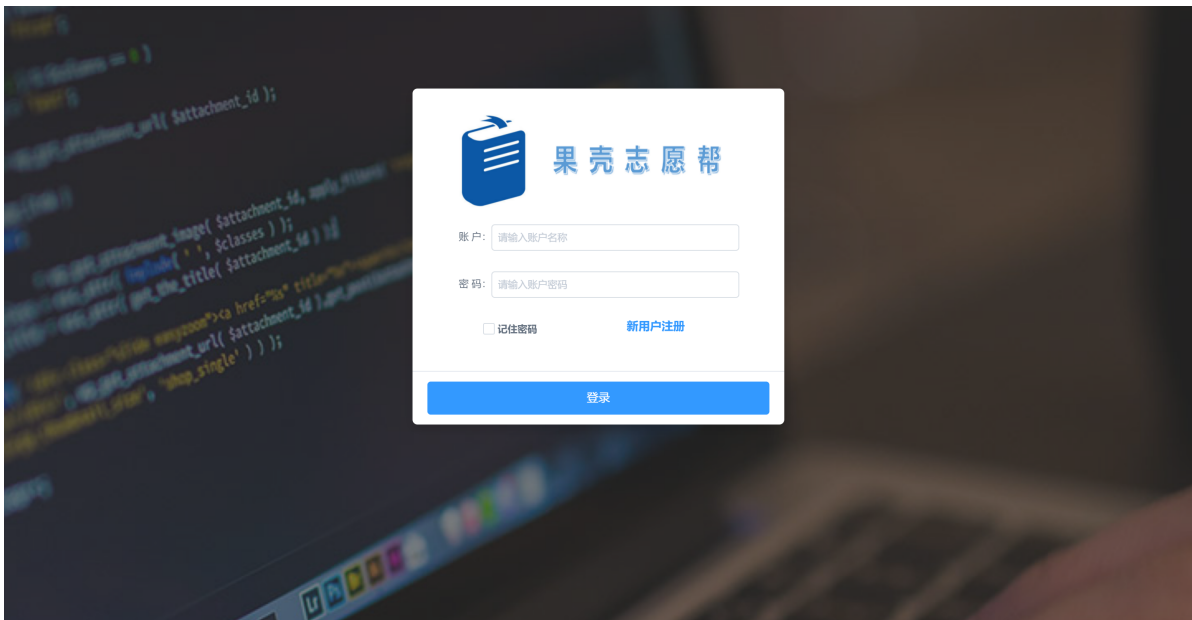
- 在本地执行 `npm run build` 命令打包
- 将打包生成的 `/dist` 文件夹里的 `static`、`index.html` 上传到服务器
- 修改 `nginx.conf` 配置

default [2].conf - 记事本

文件(F) 编辑(E) 格式(O) 视图(V) 帮助(H)

```
server {  
    listen    80;  
    server_name localhost;  
  
    #access_log /var/log/nginx/host.access.log main;  
  
    location / {  
        root /etc/Front/dist;  
        index index.html index.htm;  
    }  
}
```

- 访问服务器的80端口,测试前端界面是否部署成功



前端部署成功