> 本文主要以图文的形式讲解mall在Linux环境下的部署，涉及在Docker容器中安装MySQL、Redis、Nginx、RabbitMQ、MongoDB、Elasticsearch、Logstash、Kibana，以及SpringBoot应用部署，基于CenterOS7.6。

Docker环境安装

- 安装yum-utils：

```
yum install -y yum-utils device-mapper-persistent-data lvm2
```

- 为yum源添加docker仓库位置：

```
yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
```

- 安装docker：

```
yum install docker-ce
```

- 启动docker：

```
systemctl start docker
```

MySQL安装

- 下载MySQL5.7的docker镜像：

```
docker pull mysql:5.7
```

- 使用如下命令启动MySQL服务：

```
docker run -p 3306:3306 --name mysql \
-v /mydata/mysql/log:/var/log/mysql \
-v /mydata/mysql/data:/var/lib/mysql \
-v /mydata/mysql/conf:/etc/mysql \
-e MYSQL_ROOT_PASSWORD=root \
-d mysql:5.7
```

- 参数说明
  - -p 3306:3306：将容器的3306端口映射到主机的3306端口
  - -v /mydata/mysql/conf:/etc/mysql：将配置文件夹挂在到主机
  - -v /mydata/mysql/log:/var/log/mysql：将日志文件夹挂载到主机
  - -v /mydata/mysql/data:/var/lib/mysql/：将数据文件夹挂载到主机
  - -e MYSQL_ROOT_PASSWORD=root：初始化root用户的密码
- 进入运行MySQL的docker容器：

```
docker exec -it mysql /bin/bash
```

- 使用MySQL命令打开客户端：

```
mysql -uroot -proot --default-character-set=utf8
```

- 创建mall数据库：

```
create database mall character set utf8
```

- 安装上传下载插件，并将document/sql/mall.sql上传到Linux服务器上：

```
yum -y install lrzsz
```

- 将mall.sql文件拷贝到mysql容器的/目录下：

```
docker cp /mydata/mall.sql mysql:/
```

- 将sql文件导入到数据库：

```
use mall;
source /mall.sql;
```

- 创建一个reader:123456帐号并修改权限，使得任何ip都能访问：

```
grant all privileges on *.* to 'reader' @'%' identified by '123456';

```

Redis安装

- 下载Redis5.0的docker镜像：

```
docker pull redis:5
```

- 使用如下命令启动Redis服务：

```
docker run -p 6379:6379 --name redis \
-v /mydata/redis/data:/data \
-d redis:5 redis-server --appendonly yes
```

- 进入Redis容器使用redis-cli命令进行连接：

```
docker exec -it redis redis-cli
```

```
[root@local-linux ~]# docker run -p 6379:6379 --name redis \
> -v /mydata/redis/data:/data \
> -d redis:3.2 redis-server --appendonly yes
58d2e04a41fac479596f7bbeda5e8890fa766b05ba1464bac6c44f5ba32347c2
[root@local-linux ~]# docker exec -it redis redis-cli
127.0.0.1:6379> set a 100
OK
127.0.0.1:6379> get a
"100"
```

Nginx安装

- 下载Nginx1.10的docker镜像：

```
docker pull nginx:1.10
```

- 先运行一次容器（为了拷贝配置文件）：

```
docker run -p 80:80 --name nginx \
-v /mydata/nginx/html:/usr/share/nginx/html \
-v /mydata/nginx/logs:/var/log/nginx \
-d nginx:1.10
```

- 将容器内的配置文件拷贝到指定目录：

```
docker container cp nginx:/etc/nginx /mydata/nginx/
```

- 修改文件名称：

```
mv nginx conf
```

- 终止并删除容器：

```
docker stop nginx
docker rm nginx
```

- 使用如下命令启动Nginx服务：

```
docker run -p 80:80 --name nginx \
-v /mydata/nginx/html:/usr/share/nginx/html \
-v /mydata/nginx/logs:/var/log/nginx \
-v /mydata/nginx/conf:/etc/nginx \
-d nginx:1.10
```

RabbitMQ安装

- 下载rabbitmq3.7.15的docker镜像：

```
docker pull rabbitmq:3.7.15
```

- 使用如下命令启动RabbitMQ服务：

```
docker run -p 5672:5672 -p 15672:15672 --name rabbitmq \
-d rabbitmq:3.7.15
```

- 进入容器并开启管理功能：

```
docker exec -it rabbitmq /bin/bash
rabbitmq-plugins enable rabbitmq_management
```

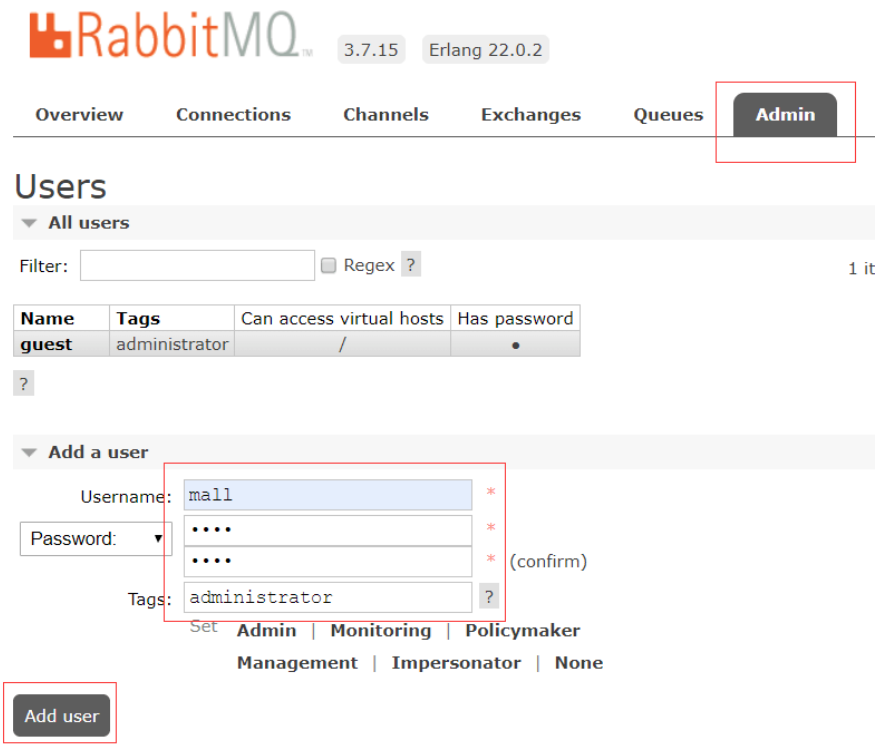
```
[root@local-linux nginx]# docker exec -it rabbitmq /bin/bash
root@dd5fafeb8847:/# rabbitmq-plugins enable rabbitmq_management
Enabling plugins on node rabbit@dd5fafeb8847:
rabbitmq_management
The following plugins have been configured:
  rabbitmq_management
  rabbitmq_management_agent
  rabbitmq_web_dispatch
Applying plugin configuration to rabbit@dd5fafeb8847...
The following plugins have been enabled:
  rabbitmq_management
  rabbitmq_management_agent
  rabbitmq_web_dispatch

started 3 plugins.
```

- 开启防火墙：

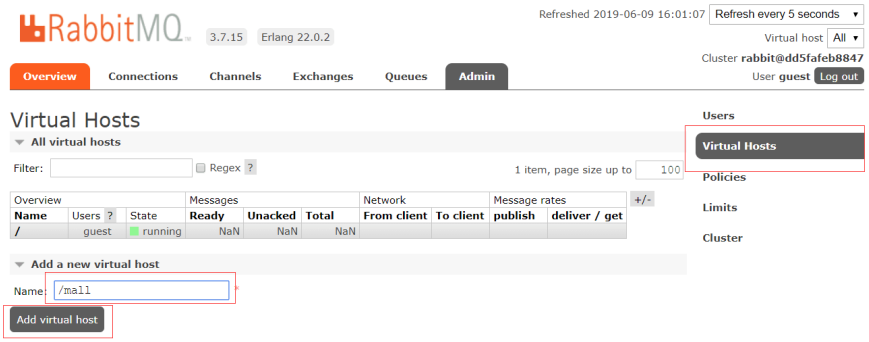
```
firewall-cmd --zone=public --add-port=15672/tcp --permanent
firewall-cmd --reload
```

- 访问地址查看是否安装成功：http://192.168.3.101:15672

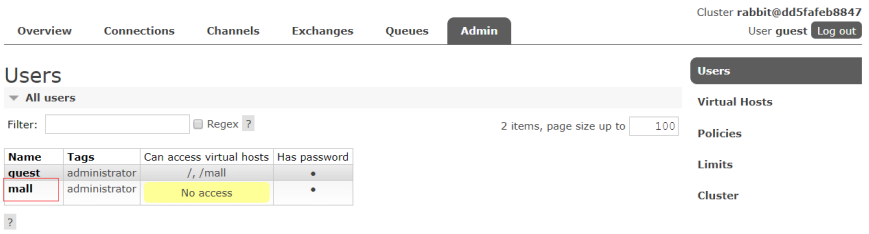
- 输入账号密码并登录：guest guest
- 创建帐号并设置其角色为管理员：mall mall



- 创建一个新的虚拟host为：/mall



- 点击mall用户进入用户配置页面



- 给mall用户配置该虚拟host的权限

Elasticsearch安装

- 下载Elasticsearch7.6.2的docker镜像：

```
docker pull elasticsearch:7.6.2
```

- 修改虚拟内存区域大小，否则会因为过小而无法启动:

```
sysctl -w vm.max_map_count=262144
```

- 使用如下命令启动Elasticsearch服务：

```
docker run -p 9200:9200 -p 9300:9300 --name elasticsearch \
-e "discovery.type=single-node" \
-e "cluster.name=elasticsearch" \
-v /mydata/elasticsearch/plugins:/usr/share/elasticsearch/plugins \
-v /mydata/elasticsearch/data:/usr/share/elasticsearch/data \
-d elasticsearch:7.6.2
```

- 启动时会发现/usr/share/elasticsearch/data目录没有访问权限，只需要修改/mydata/elasticsearch/data目录的权限，再重新启动即可；

```
chmod 777 /mydata/elasticsearch/data/
```

- 安装中文分词器IKAnalyzer，并重新启动：

```
docker exec -it elasticsearch /bin/bash
#此命令需要在容器中运行
elasticsearch-plugin install https://github.com/medcl/elasticsearch-analysis-ik/releases/download/v7.6.2/elasticsearch-analysis-ik-7.6.2.zip
docker restart elasticsearch
```

- 开启防火墙：

```
firewall-cmd --zone=public --add-port=9200/tcp --permanent
firewall-cmd --reload
```

- 访问会返回版本信息：http://192.168.3.101:9200


```
{
  "name": "DESKTOP-5NIMJ19",
  "cluster_name": "elasticsearch",
  "cluster_uuid": "HNvVRirdQW-2Q4xEpXdpsA",
  "version": {
    "number": "7.6.2",
    "build_flavor": "default",
    "build_type": "zip",
    "build_hash": "ef48eb35cf30adf4db14086e8aabd07ef6fb113f",
    "build_date": "2020-03-26T06:34:37.794943Z",
    "build_snapshot": false,
    "lucene_version": "8.4.0",
    "minimum_wire_compatibility_version": "6.8.0",
    "minimum_index_compatibility_version": "6.0.0-beta1"
  },
  "tagline": "You Know, for Search"
}
```


Logstash安装

- 下载Logstash7.6.2的docker镜像：

```
docker pull logstash:7.6.2
```

- 修改Logstash的配置文件logstash.conf中output节点下的Elasticsearch连接地址为es:9200，配置文件地址：

```
input {
tcp {
mode => "server"
host => "0.0.0.0"
port => 4560
codec => json_lines
type => "debug"
}
tcp {
 mode => "server"
 host => "0.0.0.0"
 port => 4561
 codec => json_lines
 type => "error"
 }
 tcp {
 mode => "server"
 host => "0.0.0.0"
 port => 4562
 codec => json_lines
 type => "business"
 }
 tcp {
 mode => "server"
 host => "0.0.0.0"
```

```
  port => 4563
  codec => json_lines
  type => "record"
  }
}
filter{
  if [type] == "record" {
  mutate {
  remove_field => "port"
  remove_field => "host"
  remove_field => "@version"
  }
  json {
  source => "message"
  remove_field => ["message"]
  }
  }
}
output {
  elasticsearch {
  hosts => "192.168.50.66:9200"
  index => "mall-%{type}-%{+YYYY.MM.dd}"
  }
}
```

```
output {
  elasticsearch {
  hosts => "es:9200"
  index => "mall-%{type}-%{+YYYY.MM.dd}"
  }
}
```

- 创建/mydata/logstash目录，并将Logstash的配置文件logstash.conf拷贝到该目录；

```
mkdir /mydata/logstash
```

- 使用如下命令启动Logstash服务；

```
docker run --name logstash -p 4560:4560 -p 4561:4561 -p 4562:4562 -p 4563:4563 \
--link elasticsearch:es \
-v /mydata/logstash/logstash.conf:/usr/share/logstash/pipeline/logstash.conf \
-d logstash:7.6.2
```

- 进入容器内部，安装json_lines插件。

```
logstash-plugin install logstash-codec-json_lines
```

Kibana安装

- 下载Kibana7.6.2的docker镜像：

```
docker pull kibana:7.6.2
```

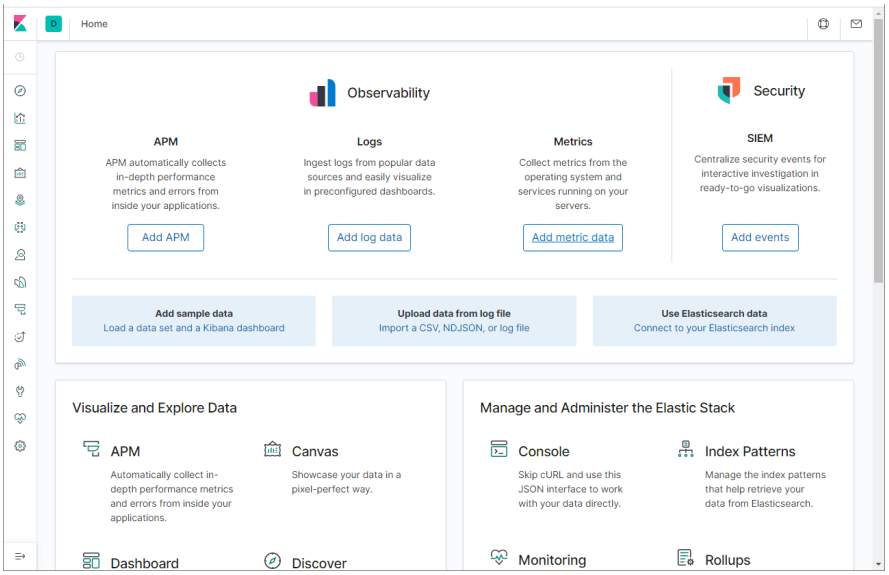
- 使用如下命令启动Kibana服务：

```
docker run --name kibana -p 5601:5601 \
--link elasticsearch:es \
-e "elasticsearch.hosts=http://es:9200" \
-d kibana:7.6.2
```

- 开启防火墙：

```
firewall-cmd --zone=public --add-port=5601/tcp --permanent
firewall-cmd --reload
```

- 访问地址进行测试：http://虚拟机IP:5601



MongoDB安装

- 下载MongoDB4.2.5的docker镜像：

```
docker pull mongo:4.2.5
```

- 使用docker命令启动：

```
docker run -p 27017:27017 --name mongo \
-v /mydata/mongo/db:/data/db \
-d mongo:4.2.5
```

Docker全部环境安装完成

- 所有下载镜像文件：

```
REPOSITORY TAG IMAGE ID CREATED SIZE
redis 5 071538dbbd71 2 weeks ago 98.3MB
mongo 4.2.5 fddee5bccba3 3 months ago 388MB
logstash 7.6.2 fa5b3b1e9757 4 months ago 813MB
kibana 7.6.2 f70986bc5191 4 months ago 1.01GB
elasticsearch 7.6.2 f29a1ee41030 4 months ago 791MB
rabbitmq 3.7.15-management 6ffc11daa8d0 13 months ago 186MB
mysql 5.7 7faa3c53e6d6 15 months ago 373MB
registry 2 f32a97de94e1 17 months ago 25.8MB
nginx 1.10 0346349a1a64 3 years ago 182MB
java 8 d23bdf5b1b1b 3 years ago 643MB
```

- 所有运行在容器里面的应用：

```
[root@local-linux deploy]# docker ps
CONTAINER ID        IMAGE                          COMMAND                  CREATED             STATUS              PORTS                                                                                  NAMES
18fbf8a832a2        logstash:7.6.2                 "/usr/local/bin/dock…"   6 days ago          Up About a minute   5044/tcp, 0.0.0.0:4560-4563->4560-4563/tcp, 9600/tcp                                   logstash
61beaa6f12b9        kibana:7.6.2                   "/usr/local/bin/dumb…"   13 days ago         Up About a minute   0.0.0.0:5601->5601/tcp                                                                 kibana
167aeb634e52        elasticsearch:7.6.2            "/usr/local/bin/dock…"   13 days ago         Up About a minute   0.0.0.0:9200->9200/tcp, 0.0.0.0:9300->9300/tcp                                         elasticsearch
4d7ce609621d        mongo:4.2.5                    "docker-entrypoint.s…"   13 days ago         Up About a minute   0.0.0.0:27017->27017/tcp                                                               mongo
4f9fee5e4b70        redis:5                        "docker-entrypoint.s…"   13 days ago         Up About a minute   0.0.0.0:6379->6379/tcp                                                                 redis
c1fae9ea0016        rabbitmq:3.7.15-management     "docker-entrypoint.s…"   13 months ago       Up About a minute   4369/tcp, 5671/tcp, 0.0.0.0:5672->5672/tcp, 15671/tcp, 25672/tcp, 0.0.0.0:15672->15672/tcp   rabbitmq
116a45d0ea2c        nginx:1.10                     "nginx -g 'daemon of…"   13 months ago       Up About a minute   0.0.0.0:80->80/tcp, 443/tcp                                                            nginx
8457cc4c8988        mysql:5.7                      "docker-entrypoint.s…"   13 months ago       Up 5 minutes        0.0.0.0:3306->3306/tcp, 33060/tcp                                                      mysql
997175ddfa40        registry:2                     "/entrypoint.sh /etc…"   14 months ago       Up 5 minutes        0.0.0.0:5000->5000/tcp                                                                 registry2
```

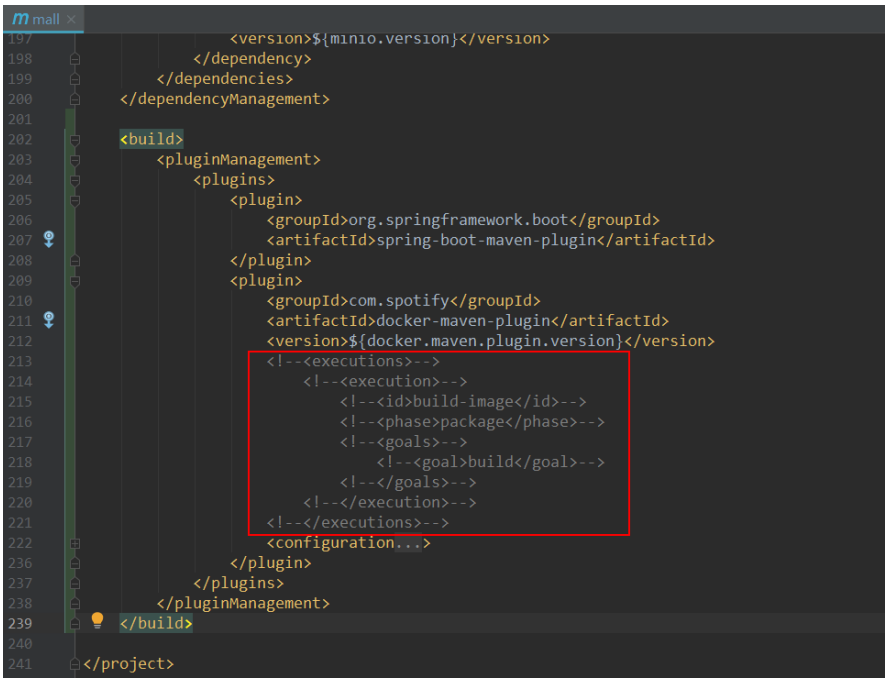
SpringBoot应用部署

构建所有Docker镜像并上传

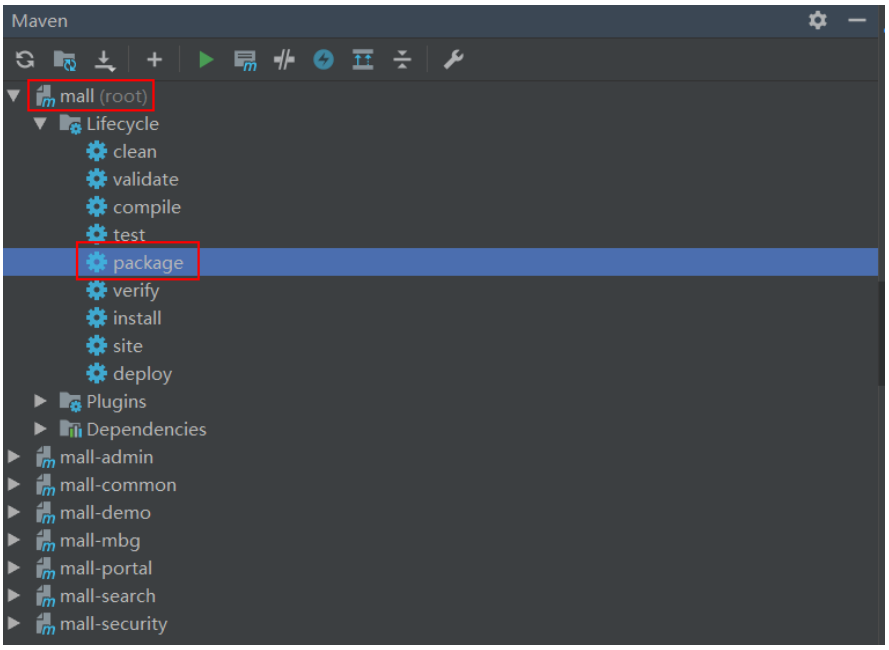
- 修改项目根目录下的pom.xml中的docker.host属性：

```
<properties>
 <docker.host>http://192.168.3.101:2375</docker.host>
</properties>
```

- 如果项目根目录的pom.mxl中docker-maven-plugin的<executions>节点被注释掉了就打开注释，使项目在打包时直接构建Docker镜像；



- 直接双击根项目mall的package命令可以一次性打包所有应用的Docker镜像；




```
REPOSITORY TAG IMAGE ID CREATED SIZE
mall/mall-portal 1.0-SNAPSHOT 70e0f76416a0 21 seconds ago 705MB
mall/mall-search 1.0-SNAPSHOT f3290bd1d0c7 41 seconds ago 725MB
mall/mall-admin 1.0-SNAPSHOT 26557b93a106 About a minute ago 705MB
```

部署mall-admin

```
docker run -p 8080:8080 --name mall-admin \
--link mysql:db \
--link redis:redis \
-v /etc/localtime:/etc/localtime \
-v /mydata/app/admin/logs:/var/logs \
-d mall/mall-admin:1.0-SNAPSHOT
```

注意：如果想使用Logstash收集日志的话，需要将应用容器连接到Logstsh，添加如下配置即可；

```
--link logstash:logstash \
```

部署mall-search

```
docker run -p 8081:8081 --name mall-search \
--link elasticsearch:es \
--link mysql:db \
-v /etc/localtime:/etc/localtime \
-v /mydata/app/search/logs:/var/logs \
-d mall/mall-search:1.0-SNAPSHOT
```

部署mall-port

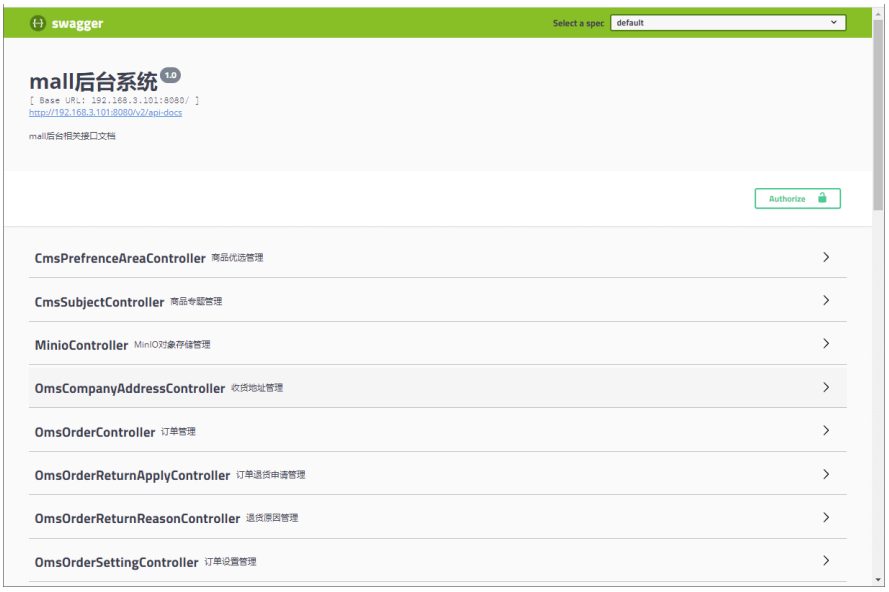
```
docker run -p 8085:8085 --name mall-portal \
--link mysql:db \
--link redis:redis \
--link mongo:mongo \
--link rabbitmq:rabbit \
-v /etc/localtime:/etc/localtime \
-v /mydata/app/portal/logs:/var/logs \
-d mall/mall-portal:1.0-SNAPSHOT
```

开启防火墙

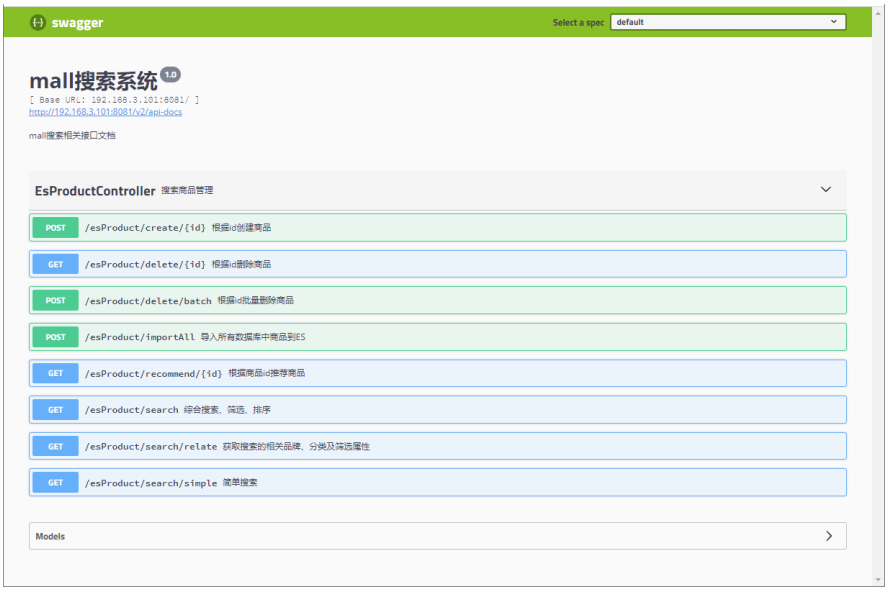
```
firewall-cmd --zone=public --add-port=8080/tcp --permanent
firewall-cmd --zone=public --add-port=8081/tcp --permanent
firewall-cmd --zone=public --add-port=8085/tcp --permanent
firewall-cmd --reload
```

访问接口进行测试

- mall-admin的api接口文档地址：http://虚拟机IP:8080/swagger-ui.html



- mall-search的api接口文档地址：http://虚拟机IP:8081/swagger-ui.html



- mall-portal的api接口文档地址：http://虚拟机IP:8085/swagger-ui.html