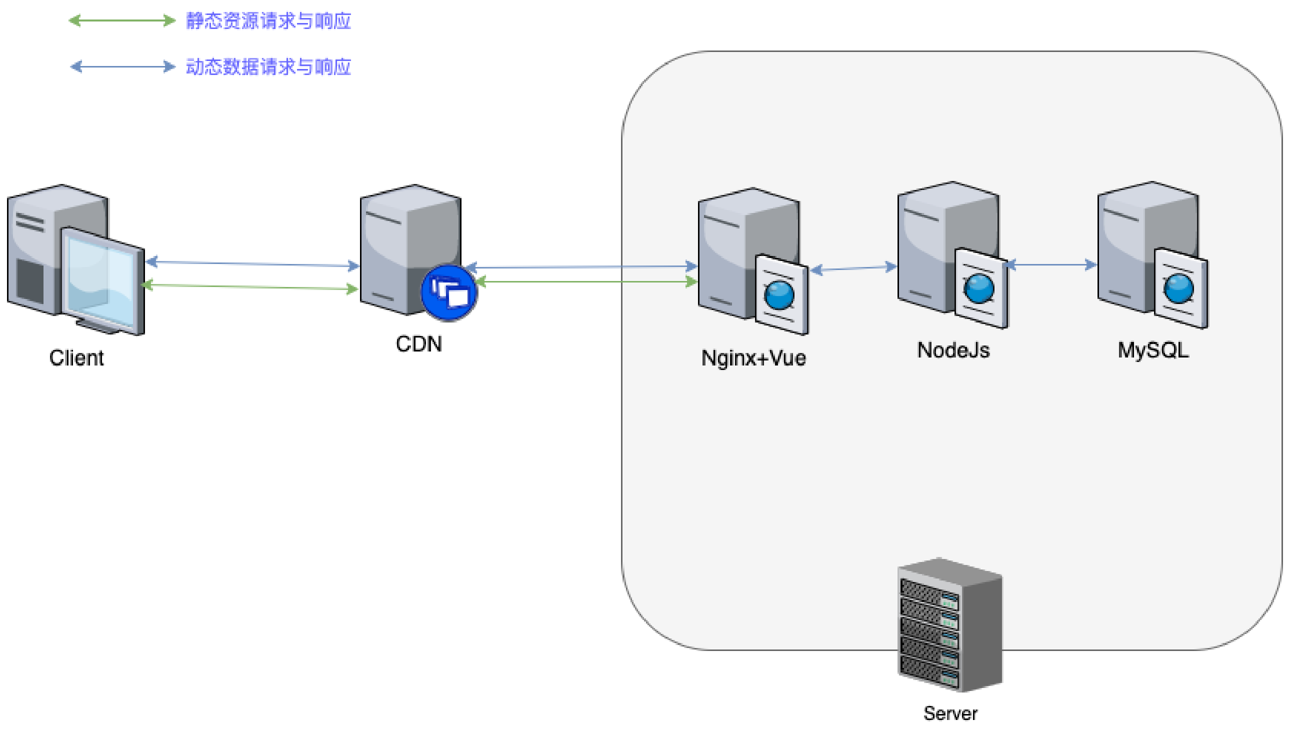
Vue+NodeJS前后端分离项目Docker部署

> 链接：https://m.cuiliangblog.cn/detail/article/10

> 最近在学习Vue开发前端项目，跟着教程做了一个小项目，打算把写好的Vue项目使用docker部署到阿里云主机上去，使用华为CDN加速，记录一下详细的部署过程，供大家参考。

一、项目说明

1. 整体架构图



2. 环境描述

- 使用华为云CDN对静态资源进行分发和加速
- 项目全部部署在阿里云主机，使用Docker运行
- 全站资源使用https加密访问
- 数据库每日定时备份

3. 部署效果

- 访问地址：https://shop.cuiliangblog.cn/
- 账号：admin
- 密码：123456

二、准备工作

1. 源码获取

- Vue前端项目地址

https://gitee.com/cuiliang0302/vue_shop.git

- NodeJS后端项目地址

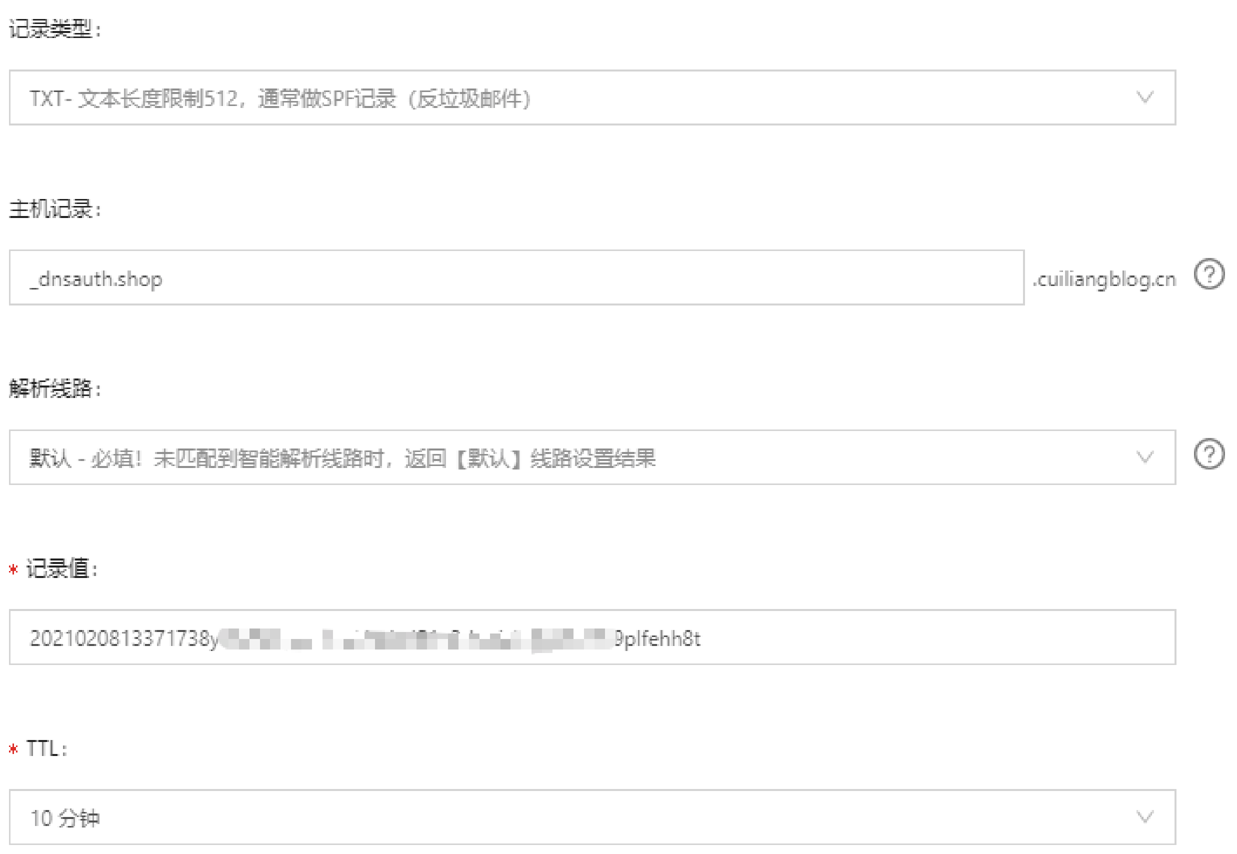
https://gitee.com/cuiliang0302/vue_shop_api.git

2. https证书申请

- 推荐一个免费的SSL证书申请地址

https://freessl.cn/

- 申请完成后根据要求，在云解析DNS中配置相关的TXT记录。



记录类型：

TXT- 文本长度限制512，通常做SPF记录（反垃圾邮件）

主机记录：

_dnsauth.shop .cuiliangblog.cn

解析线路：

默认 - 必填！未匹配到智能解析线路时，返回【默认】线路设置结果

* 记录值：

2021020813371738y[redacted]9plfehh8t

* TTL：

10 分钟

3. Docker部署

- 参考地址

https://www.cuiliangblog.cn/blog/section-97/

4. 创建相关目录与文件

- 将项目代码clone至本地，并上传相关ssl证书，创建文件

```
[root@localhost opt]# tree /opt/docker/
/opt/docker/
├── mysql
├── nginx
|   ├── ssl
|   |   ├── shop.cuiliangblog.cn_chain.crt
|   |   └── shop.cuiliangblog.cn_key.key
|   └── vue_shop
└── nodejs
    └── vue_shop_api
# 将项目克隆到对应目录下
```

5. 创建docker网络用于容器互联

- 容器间需要进行服务发现与调用，可以使用–link将容器之间进行连接，但官方并不推荐这样操作，而是使用docker network方式，具体内容请查看文章：https://www.cuiliangblog.cn/blog/section-127/

```
[root@localhost ~]# docker network create net
f2eb78c2aa8fbdd514456eb758ba1d53c73fa61e4347a5ef446b708abe39327f
[root@localhost ~]# docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
a533f467bc62   bridge    bridge    local
d6522112ce12   host      host      local
f2eb78c2aa8f   net       bridge    local
81f8921cf73b   none      null      local
```

三、数据库部署

1. 创建并启动容器

```
[root@localhost mysql]# cd /opt/docker/mysql/
[root@localhost mysql]# docker pull mysql
[root@localhost mysql]# docker run -p 3306:3306 --name mysql -v
$PWD/conf:/etc/mysql/conf.d -v $PWD/logs:/logs -v $PWD/data:/var/lib/mysql -e
MYSQL_ROOT_PASSWORD=123.com -d --network net --restart=always mysql
dc36586b8f6fadcf945b2cae85a0e6e222f4b6548873a491d7d5522376e71b26
[root@localhost mysql]# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS
                              NAMES
dc36586b8f6f   mysql     "docker-entrypoint.s…"   6 seconds ago   Up 4 seconds
0.0.0.0:3306->3306/tcp, 33060/tcp   mysql
```

- 选项说明

-p 3306:3306：将容器的 3306 端口映射到主机的 3306 端口。(如果不需要远程访问，则无需映射) -v $PWD/conf:/etc/mysql/conf.d：将主机当前目录下的 conf/my.cnf 挂载到容器的/etc/mysql/my.cnf。 -v $PWD/logs:/logs：将主机当前目录下的logs 目录挂载到容器的 /logs。 -v $PWD/data:/var/lib/mysql:将主机当前目录下的data目录挂载到容器的 /var/lib/mysql 。 -e MYSQL_ROOT_PASSWORD=123.com：初始化 root 用户的密码。 -d ：后台运行 –network net：使用自定义的net网络 –restart=always: 不管退出状态码是什么，始终重启容器

2. 进入容器登录数据库

```
[root@localhost mysql]# docker exec -it mysql bash
root@66e5edff50d4:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.23 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

3. 创建数据库

```
mysql> CREATE DATABASE shop CHARACTER SET utf8 COLLATE utf8_general_ci;
Query OK, 1 row affected, 2 warnings (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| shop               |
| sys                |
+--------------------+
5 rows in set (0.01 sec)
```

4. 创建用户并授权

```
mysql> CREATE USER 'admin'@'%' IDENTIFIED BY '1234qwer';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON shop.* TO 'admin'@'%';
Query OK, 0 rows affected (0.01 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
```

5. 导入数据

- NodeJS项目数据库存放地址：vue_shop_api/db/mydb.sql

```
# 将sql文件从主机拷贝到容器/tmp下
[root@localhost mysql]# docker cp /opt/docker/nodejs/vue_shop_api/db/mydb.sql
mysql:/tmp
# 在mysql容器中执行导入数据操作
mysql> use shop;
mysql> source /tmp/mydb.sql;
mysql> show tables;
+--------------------+
| Tables_in_shop     |
+--------------------+
| sp_attribute       |
| sp_category        |
| sp_consignee       |
| sp_express         |
| sp_goods           |
| sp_goods_attr      |
| sp_goods_cats      |
| sp_goods_pics      |
| sp_manager         |
| sp_order           |
| sp_order_goods     |
| sp_permission      |
| sp_permission_api  |
| sp_report_1        |
| sp_report_2        |
| sp_report_3        |
| sp_role            |
| sp_type            |
| sp_user            |
| sp_user_cart       |
+--------------------+
20 rows in set (0.01 sec)
```

6. 测试验证

- 通过–rm，创建一次性容器测试admin用户是否能正常登录，–network net指定与mysql在同一个网络中

```
[root@localhost mysql]# docker run -it --rm --network net mysql bash
root@450468c9b3b0:/# mysql -h mysql -u admin -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.23 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| shop               |
+--------------------+
2 rows in set (0.00 sec)
```

四、NodeJS部署

1. 修改连接数据库配置文件

- 项目数据库配置文件路径：vue_shop_api/config/default.json，根据实际情况修改db_config相关配置

```
{
	"config_name" : "develop",
	"jwt_config" : {
		"secretKey":"itcast",
		"expiresIn":86400
	},
	"upload_config":{
		"baseURL":"http://127.0.0.1:8888",
		"upload_ueditor":"uploads/ueditor",
		"simple_upload_redirect":"http://127.0.0.1/reload"
	},
	"db_config" : {
		"protocol" : "mysql",
		"host" : "mysql",
		"database" : "shop",
		"user" : "admin",
		"password" : "1234qwer",
		"port" : 3306
	}
}
```

2. 编写dockerfile并构建镜像

- 通过查看项目README可知，主要执行两条命令

安装依赖：npm install 启动项目： node app.js

- 编写dockerfile（dockerfile与项目文件夹在同一个目录下），dockerfile的内容分别是：

将项目代码复制到镜像中，并指定工作目录 安装npm依赖库 指定容器运行时监听的网络端口 指定容器运行的时的命令及参数

```
FROM node
COPY vue_shop_api /opt/vue_shop_api
WORKDIR /opt/vue_shop_api/
RUN npm install --registry=https://registry.npm.taobao.org
EXPOSE 8888
CMD ["node","app.js"]
```

- 执行构建镜像命令

```
[root@localhost nodejs]# pwd
/opt/docker/nodejs
[root@localhost nodejs]# ls
dockerfile  vue_shop_api
[root@localhost nodejs]# docker build -t shop_api:v1 .
```

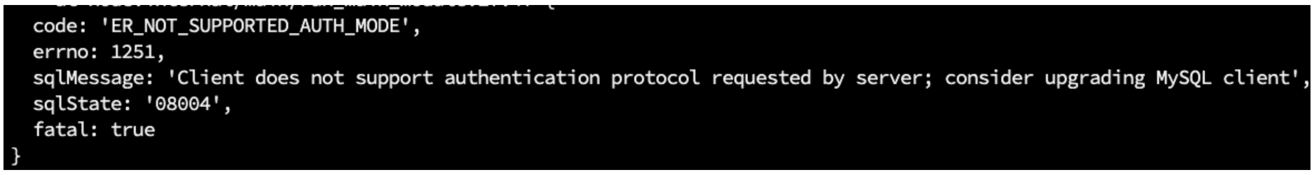
3. 启动容器

```
[root@localhost nodejs]# docker run --name shop_api -d --network net --restart=always
shop_api:v1
[root@localhost nodejs]# docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS
             PORTS                                NAMES
6ba9359556d2   shop_api:v1   "docker-entrypoint.s…"   52 seconds ago   Restarting (1) 7
seconds ago                                       shop_api
5075f372f88f   mysql         "docker-entrypoint.s…"   24 minutes ago   Up 24 minutes
               0.0.0.0:3306->3306/tcp, 33060/tcp   mysql
```

- log和配置文件以及服务端口是否暴露根据需求自行选择

4. 修改mysql用户认证方式

- 使用docker logs shop_api查看日志发现连接数据库报错，提示AUTH_MODE异常



```
  code: 'ER_NOT_SUPPORTED_AUTH_MODE',
  errno: 1251,
  sqlMessage: 'Client does not support authentication protocol requested by server; consider upgrading MySQL client',
  sqlState: '08004',
  fatal: true
}
```

- 查看用户加密认证方式

```
mysql> select user,plugin from mysql.user;
+------------------+-----------------------+
| user             | plugin                |
+------------------+-----------------------+
| admin            | caching_sha2_password |
| root             | caching_sha2_password |
| mysql.infoschema | caching_sha2_password |
| mysql.session    | caching_sha2_password |
| mysql.sys        | caching_sha2_password |
| root             | caching_sha2_password |
+------------------+-----------------------+
6 rows in set (0.00 sec)
```

- mysql8+默认使用了caching_sha2_password 方式认证加密，但是NodeJS并不支持，需要改为mysql_native_password方式认证

```
mysql> ALTER USER 'admin'@'%' IDENTIFIED WITH mysql_native_password BY '1234qwer';
Query OK, 0 rows affected (0.01 sec)

mysql> select user,plugin from mysql.user;
+------------------+-----------------------+
| user             | plugin                |
+------------------+-----------------------+
| admin            | mysql_native_password |
| root             | caching_sha2_password |
| mysql.infoschema | caching_sha2_password |
| mysql.session    | caching_sha2_password |
| mysql.sys        | caching_sha2_password |
| root             | caching_sha2_password |
+------------------+-----------------------+
6 rows in set (0.00 sec)
```

- 重启shop_api容器

```
[root@localhost nodejs]# docker restart shop_api
[root@localhost nodejs]# docker logs shop_api
```

- 此时日志无异常输出

4. 测试验证

- 通过–rm，创建一次性容器测试admin用户是否能正常登录，–network net指定与mysql在同一个网络中，通过centos镜像启动容器使用curl命令模拟发起POST请求，测试能否收到服务器响应

```
[root@localhost ~]# docker run -it --rm --network net centos curl -i -X POST -H
"Content-type:application/json" -d
'{"username":"admin","password":"123456","token":""}' shop_api:8888/api/private/v1/
HTTP/1.1 200 OK
X-Powered-By:  3.2.1
Access-Control-Allow-Origin: *
```

```
Access-Control-Allow-Headers: Content-Type,Content-Length, Authorization, Accept,X-Requested-With
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Methods: PUT,POST,GET,DELETE,OPTIONS
Content-Length: 55
ETag: W/"37-zuuEYGHaMQ3PzXTOnsUYQ09lC70"
Date: Thu, 11 Feb 2021 01:04:32 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{"data":null,"meta":{"msg":"无效token","status":400}}[root@localhost ~]#
```

五、Vue部署

> 从docker_17.05版本以后，新增了Dockerfile多阶段构建，具体使用可参考文章： https://www.cuiliangblog.cn/blog/section-204/ 此次构建的镜像包含两个阶段，分别是使用node镜像编译项目和使用nginx镜像提供服务

1. dockerfile-编译阶段

- 项目配置文件路径：vue_shop/src/main-prod.js，根据实际情况修改axios.defaults.baseURL = "/api/private/v1/"配置即可，此处我们无需修改（如果想让用户直接访问nodejs服务器，此处改为nodejs请求api地址即可）
- 此阶段的dockerfile的内容如下：

将项目代码复制到镜像中，并指定工作目录 安装npm依赖库并编译项目生成打包文件

```
FROM node AS build # AS指定别名，便于在运行阶段通过别名操作，将编译阶段生成的打包文件拷贝到运行镜像中
COPY vue_shop /opt/vue_shop
WORKDIR /opt/vue_shop
RUN npm install --registry=https://registry.npm.taobao.org && npm run build
```

- 编译完后会在镜像的/opt/vue_shop目录下生成一个dist文件的项目打包文件

2. dockerfile-运行阶段

- 此阶段的dockerfile内容如下：

将编译阶段生成的/opt/vue_shop/dist文件添加到镜像中 将ssl证书添加到镜像中 将自定义的配置文件添加到镜像中（配置ssl和location配置），替换默认nginx配置文件

```
FROM nginx
COPY --from=build /opt/vue_shop/dist /opt/vue_shop/dist
COPY nginx.conf /etc/nginx/nginx.conf
COPY ssl /etc/ssl
CMD ["nginx", "-g","daemon off;"]
```

- nginx.conf配置文件如下，主要注意以下几点

ssl证书路径与dockerfile添加的ssl证书文件路径保持一致 location / 配置路径与dockerfile添加的打包文件路径保持一致 location /api/private/v1/ 中的容器名称与创建的nodejs容器名称保持一致 gzip根据情况选择性启用 http跳转到https可以使用return 301或者rewrite，推荐使用return 301

```
# For more information on configuration, see:
#   * Official English Documentation: http://nginx.org/en/docs/
#   * Official Russian Documentation: http://nginx.org/ru/docs/

user root;
worker_processes auto;
error_log /var/log/nginx/error.log;
pid /run/nginx.pid;

# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.
include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile            on;
    tcp_nopush          on;
    tcp_nodelay         on;
    keepalive_timeout   65;
    types_hash_max_size 2048;

    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;
    # http跳转到https
    server {
        listen       80;
        server_name  shop.cuiliangblog.cn;
        return 301 https://$host$request_uri;
    }
    # vue_shop项目
    server {
        listen 443 ssl http2;
        server_name  shop.cuiliangblog.cn;
        charset utf-8;
        ssl_certificate    /etc/ssl/shop.cuiliangblog.cn_chain.crt;#证书路径
        ssl_certificate_key  /etc/ssl/shop.cuiliangblog.cn_key.key;#密钥路径
        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        ssl_ciphers ECDHE-RSA-AES128-GCM-SHA256:HIGH:!aNULL:!MD5:!RC4:!DHE;
        ssl_prefer_server_ciphers on;
        ssl_session_cache shared:SSL:10m;
        ssl_session_timeout 10m;
        gzip on;
        gzip_buffers 32 4K;
        gzip_comp_level 6;
        gzip_min_length 100;
        gzip_types application/javascript text/css text/xml;
        gzip_disable "MSIE [1-6]\."; #配置禁用gzip条件，支持正则。此处表示ie6及以下不启用
gzip（因为ie低版本不支持）
        gzip_vary on;
        location / {
            root  /opt/vue_shop/dist;
        }
        location /api/private/v1/ {
            proxy_pass http://shop_api:8888;
        }
    }
}
```

3. 构建镜像并运行容器

- 整个工作目录结构

```
[root@localhost nginx]# tree /opt/docker/nginx/
/opt/docker/nginx/
├── dockerfile
├── log
├── nginx.conf
├── ssl
│   ├── shop.cuiliangblog.cn_chain.crt
│   └── shop.cuiliangblog.cn_key.key
└── vue_shop
```

- 完整的dockerfile文件

```
FROM node AS build
COPY vue_shop /opt/vue_shop
WORKDIR /opt/vue_shop
RUN npm install --registry=https://registry.npm.taobao.org && npm run build

FROM nginx
COPY --from=build /opt/vue_shop/dist /opt/vue_shop/dist
COPY nginx.conf /etc/nginx/nginx.conf
COPY ssl /etc/ssl
CMD ["nginx", "-g","daemon off;"]
```

- 构建名为vue_shop的镜像

```
[root@localhost nginx]# docker build -t vue_shop:v1 .
```

- 运行容器，将nginx日志挂载至主机log目录下

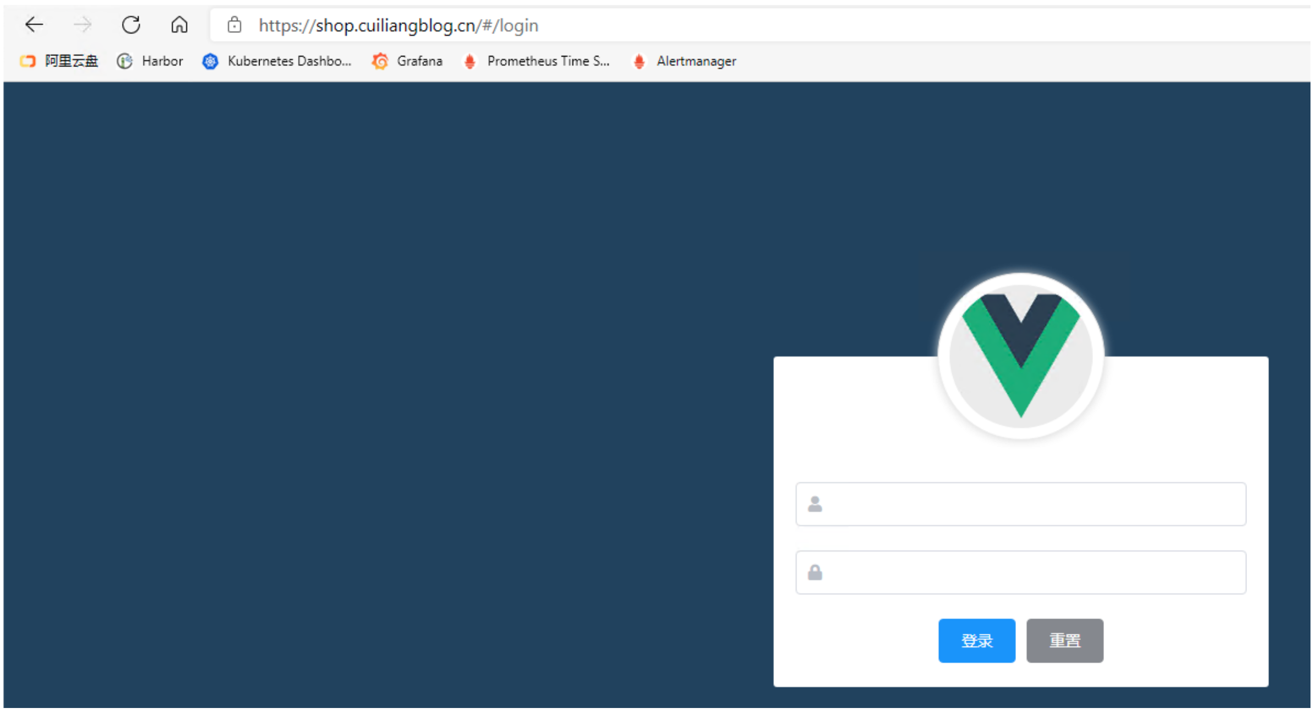
```
[root@localhost nginx]# docker run --name vue_shop -p 443:443 -p 80:80 -d --network net
-v $PWD/log:/var/log/nginx --restart=always vue_shop:v1
f934b7e522fe1652c848d0bf1fa877b05936a2aaafcace1e18b9fd9433d40c54
[root@localhost nginx]# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS
PORTS                                        NAMES
f934b7e522fe   vue_shop:v1    "/docker-entrypoint.…"   7 seconds ago    Up 5 seconds
0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp   vue_shop
1b490fd2e447   shop_api:v1    "docker-entrypoint.s…"   37 minutes ago   Up 37 minutes
8888/tcp                                     shop_api
5f180b407309   mysql          "docker-entrypoint.s…"   10 hours ago     Up 10 hours
0.0.0.0:3306->3306/tcp, 33060/tcp          mysql
```

4. 访问测试

- 本地curl访问测试

```
[root@localhost nginx]# curl -k -s https://127.0.0.1
<!DOCTYPE html><html lang=""><head><meta charset="utf-8"><meta http-equiv="X-UA-
Compatible" content="IE=edge"><meta name="viewport" content="width=device-
width,initial-scale=1"><link rel="icon" href="favicon.ico"><title>vue_shop</title><link
rel="stylesheet" href="https://cdn.staticfile.org/font-awesome/5.15.2/css/all.min.css">
<link rel="stylesheet"
href="https://cdn.staticfile.org/nprogress/0.2.0/nprogress.min.css"><link
rel="stylesheet" href="https://cdn.bootcdn.net/ajax/libs/element-ui/2.15.0/theme-
chalk/index.min.css"><link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/mavon-
editor@2.9.1/dist/css/index.css"><script
src="https://cdn.bootcdn.net/ajax/libs/vue/2.6.11/vue.min.js"></script><script
src="https://cdn.bootcdn.net/ajax/libs/element-ui/2.15.0/index.min.js"></script><script
src="https://cdn.jsdelivr.net/npm/echarts@5.0.2/dist/echarts.min.js"></script><script
src="https://cdn.jsdelivr.net/npm/axios@0.21.1/dist/axios.min.js"></script><script
src="https://cdn.jsdelivr.net/npm/vue-router@3.4.9/dist/vue-router.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/lodash@4.17.20/lodash.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/mavon-editor@2.9.1/dist/mavon-editor.js">
</script><script src="https://cdn.jsdelivr.net/npm/nprogress@0.2.0/nprogress.min.js">
</script><link href="css/goods.232cd085.css" rel="prefetch"><link
href="css/login_home_welcome.d114a304.css" rel="prefetch"><link
href="css/order.173d716f.css" rel="prefetch"><link href="css/power.1f4c73f9.css"
rel="prefetch"><link href="js/goods.185256e6.js" rel="prefetch"><link
href="js/login_home_welcome.677427cc.js" rel="prefetch"><link
href="js/login_home_welcome~users.d32ccdb9.js" rel="prefetch"><link
href="js/order.60de546a.js" rel="prefetch"><link href="js/power.fc19d83f.js"
rel="prefetch"><link href="js/report.372f9ca9.js" rel="prefetch"><link
href="js/users.21d523a1.js" rel="prefetch"><link href="css/app.41819f12.css"
rel="preload" as="style"><link href="js/app.f0802d73.js" rel="preload" as="script">
<link href="js/chunk-vendors.c1384b97.js" rel="preload" as="script"><link
href="css/app.41819f12.css" rel="stylesheet"></head><body><noscript><strong>We're sorry
but vue_shop doesn't work properly without JavaScript enabled. Please enable it to
continue.</strong></noscript><div id="app"></div><script src="js/chunk-
vendors.c1384b97.js"></script><script src="js/app.f0802d73.js"></script></body></html>
```

- 修改本地hosts文件访问测试

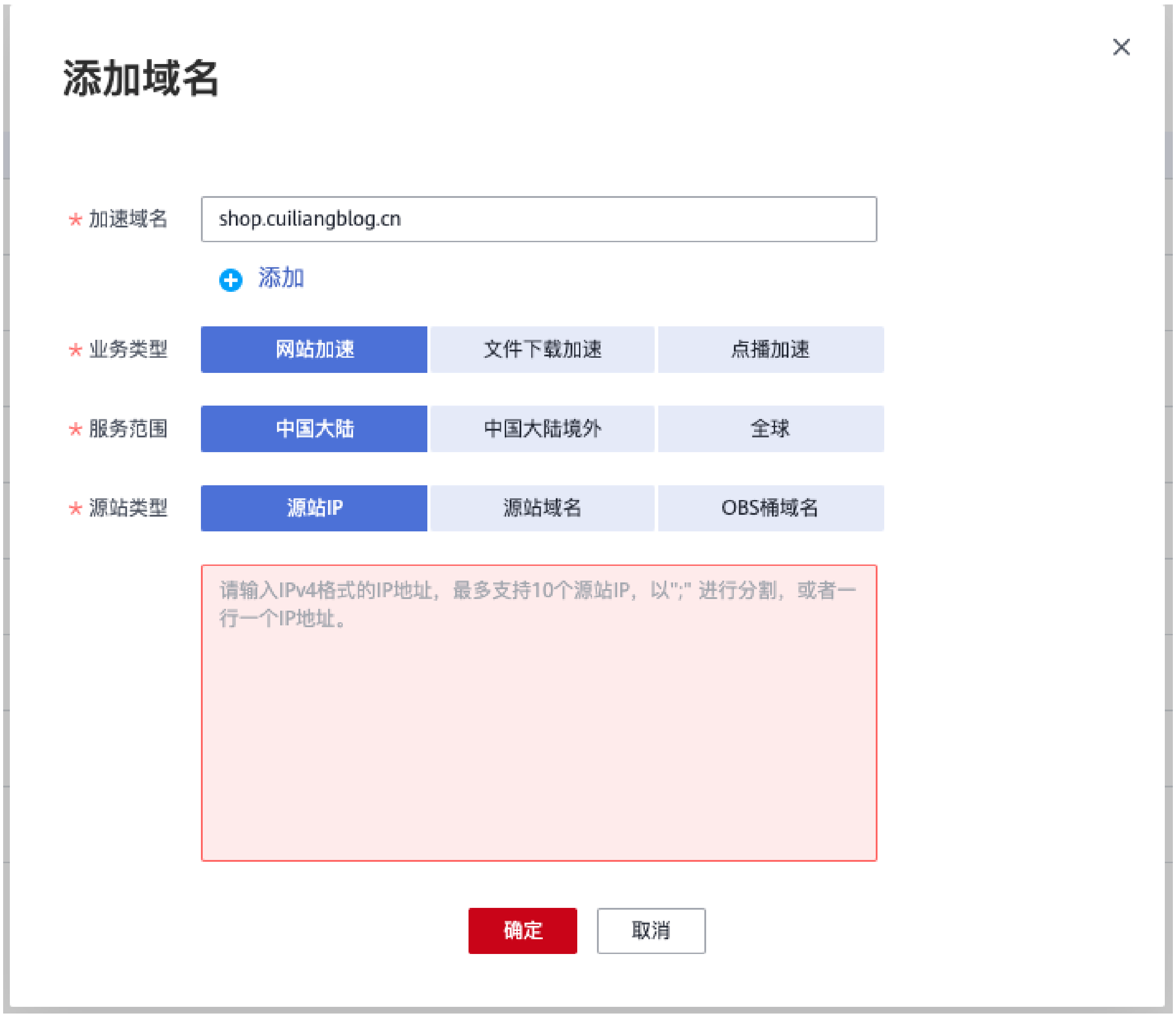


- 至此，项目部署完毕

六、CDN配置

1. 华为云CDN配置

- 登录华为云控制台——CDN——域名管理——添加域名

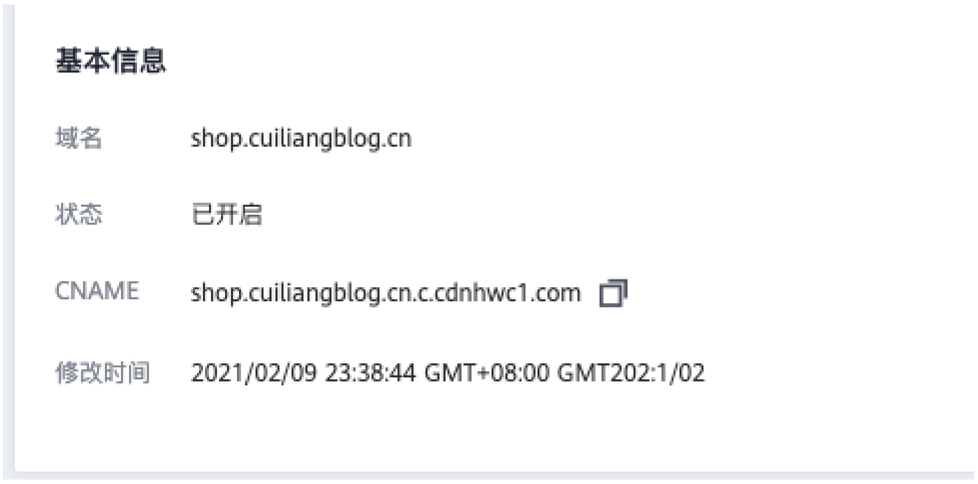


添加域名

* 加速域名	shop.cuiliangblog.cn
* 业务类型	[✓ 网站加速] 文件下载加速 点播加速
* 服务范围	[✓ 中国大陆] 中国大陆境外 全球
* 源站类型	[✓ 源站IP] 源站域名 OBS桶域名

请输入IPv4格式的IP地址，最多支持10个源站IP，以";" 进行分割，或者一行一个IP地址。

- 域名添加完成后会得到一个CNAME解析



基本信息

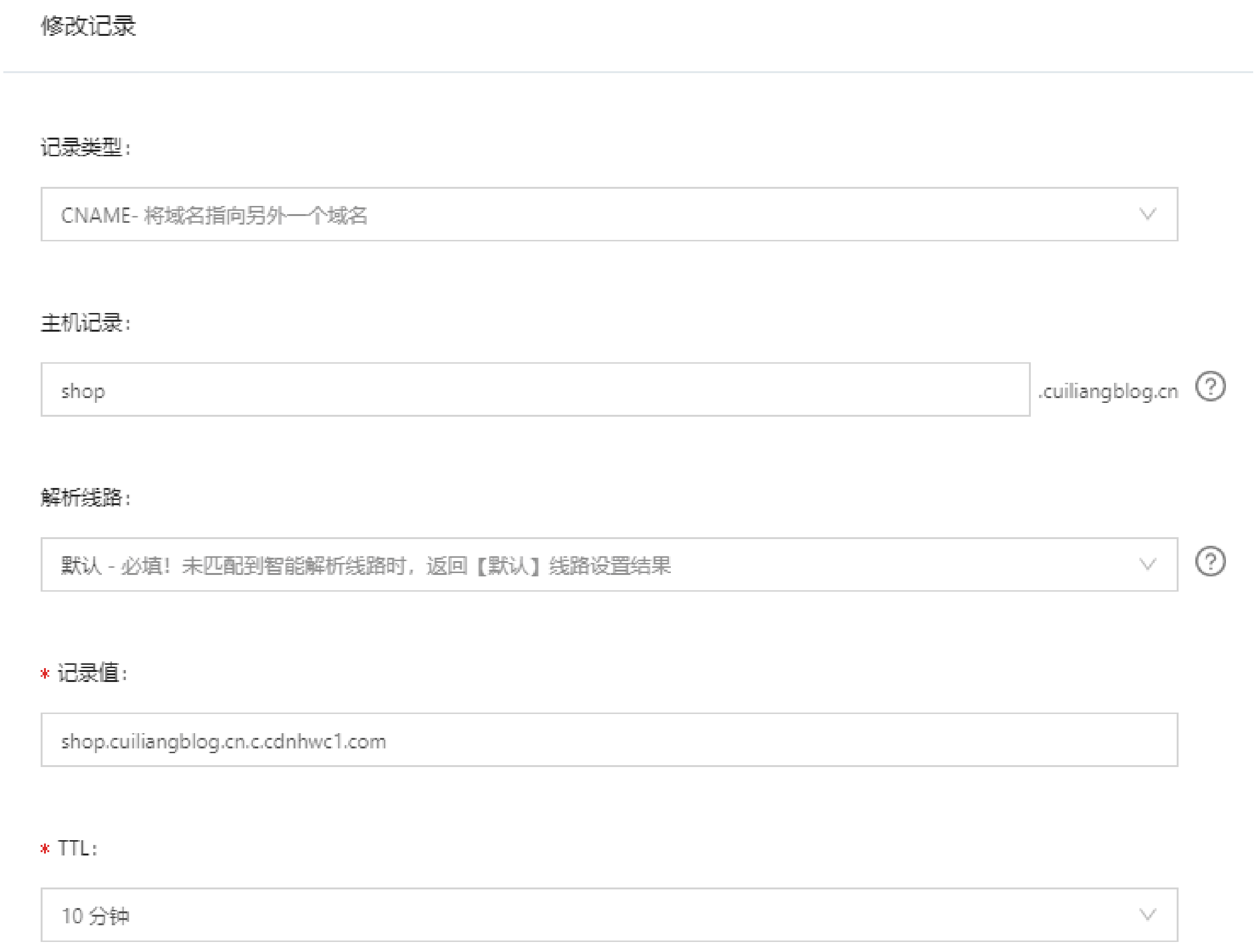
域名	shop.cuiliangblog.cn
状态	已开启
CNAME	shop.cuiliangblog.cn.c.cdnhwc1.com
修改时间	2021/02/09 23:38:44 GMT+08:00 GMT202:1/02

- 配置https安全加速，并开启https回源和强制跳转https

image.png

2. 阿里云DNS云解析配置

- 阿里云控制台——云解析DNS——解析设置，将华为云CDN加速域名填写到记录中



修改记录

记录类型：

CNAME- 将域名指向另外一个域名

主机记录：

shop .cuiliangblog.cn

解析线路：

默认 - 必填！未匹配到智能解析线路时，返回【默认】线路设置结果

* 记录值：

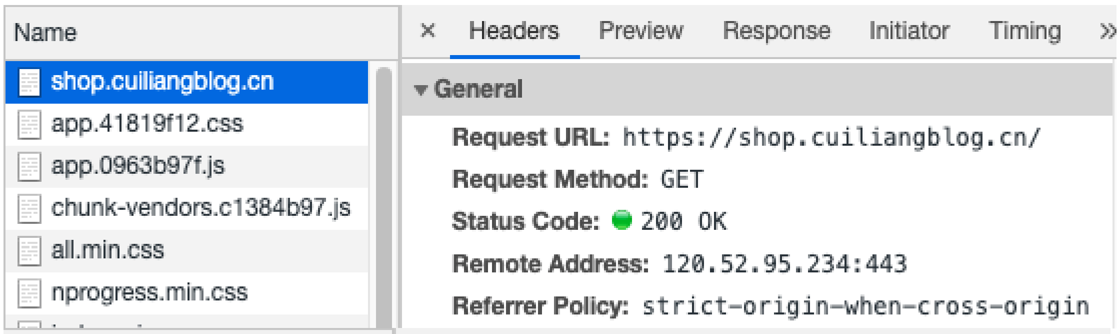
shop.cuiliangblog.cn.c.cdnhwc1.com

* TTL：

10 分钟

3. 访问测试

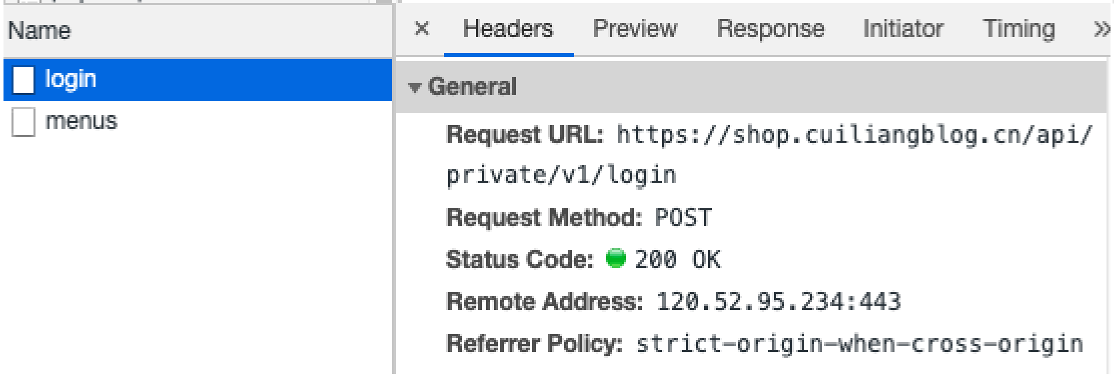
- 访问发现此时请求的主机均为华为CDN服务器



shop.cuiliangblog.cn
app.41819f12.css
app.0963b97f.js
chunk-vendors.c1384b97.js
all.min.css
nprogress.min.css

▼ General

```
Request URL: https://shop.cuiliangblog.cn/
Request Method: GET
Status Code: ● 200 OK
Remote Address: 120.52.95.234:443
Referrer Policy: strict-origin-when-cross-origin
```



login
menus

▼ General

```
Request URL: https://shop.cuiliangblog.cn/api/
private/v1/login
Request Method: POST
Status Code: ● 200 OK
Remote Address: 120.52.95.234:443
Referrer Policy: strict-origin-when-cross-origin
```