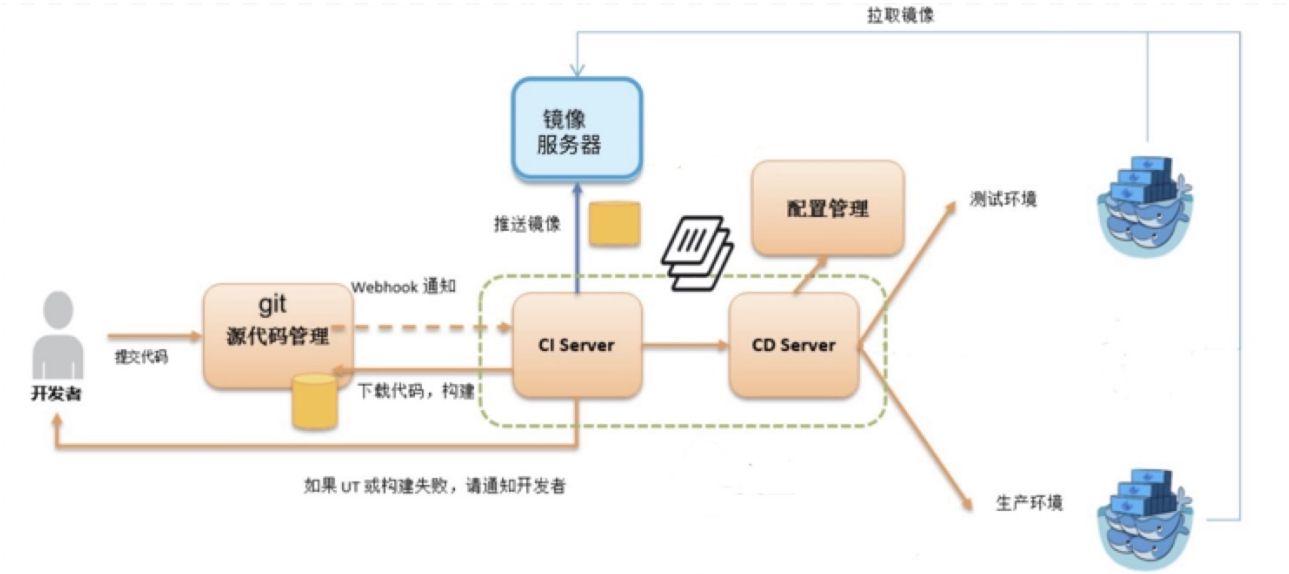
link: null
title: 珠峰架构师成长计划
description: null
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=52 sentences=125, words=630

1.持续集成和布署

- 技术栈 前台Vue,后台Node.js
- 服务器 前台nginx,后台Node.js



2.编写后端服务

2.1 package.json

/usr/projects/vue-back/package.json

```
{
  "name": "vue-back",
  "version": "0.0.0",
  "private": true,
  "scripts": {
+    "start": "node ./server.js "
  }
}
```

2.2 server.js

/usr/projects/vue-back/server.js

```
let http = require('http');
let users = [{id:1,name:'zhufeng1'},{id:2,name:'zhufeng2'},{id:3,name:'zhufeng3'}];
let server = http.createServer(function(req,res){
  console.log(req.method,req.url);
  if(req.url == '/api/users'){
    res.setHeader('Access-Control-Allow-Origin','*');
    res.end(JSON.stringify(users));
  }else{
    res.end('Now Found!');
  }
});
server.listen(3000,()=>{
  console.log('服务正在3000端口上启动!');
});
```

2.3 .gitignore

/usr/projects/vue-back/.gitignore

```
node_modules
lib
package-lock.json
```

3.前端项目

3.1 安装脚手架生成项目

```
cnpm i @vue/cli -g
vue create vue-front
cd vue-front
cnpm i axios -S
```

3.2 App.vue

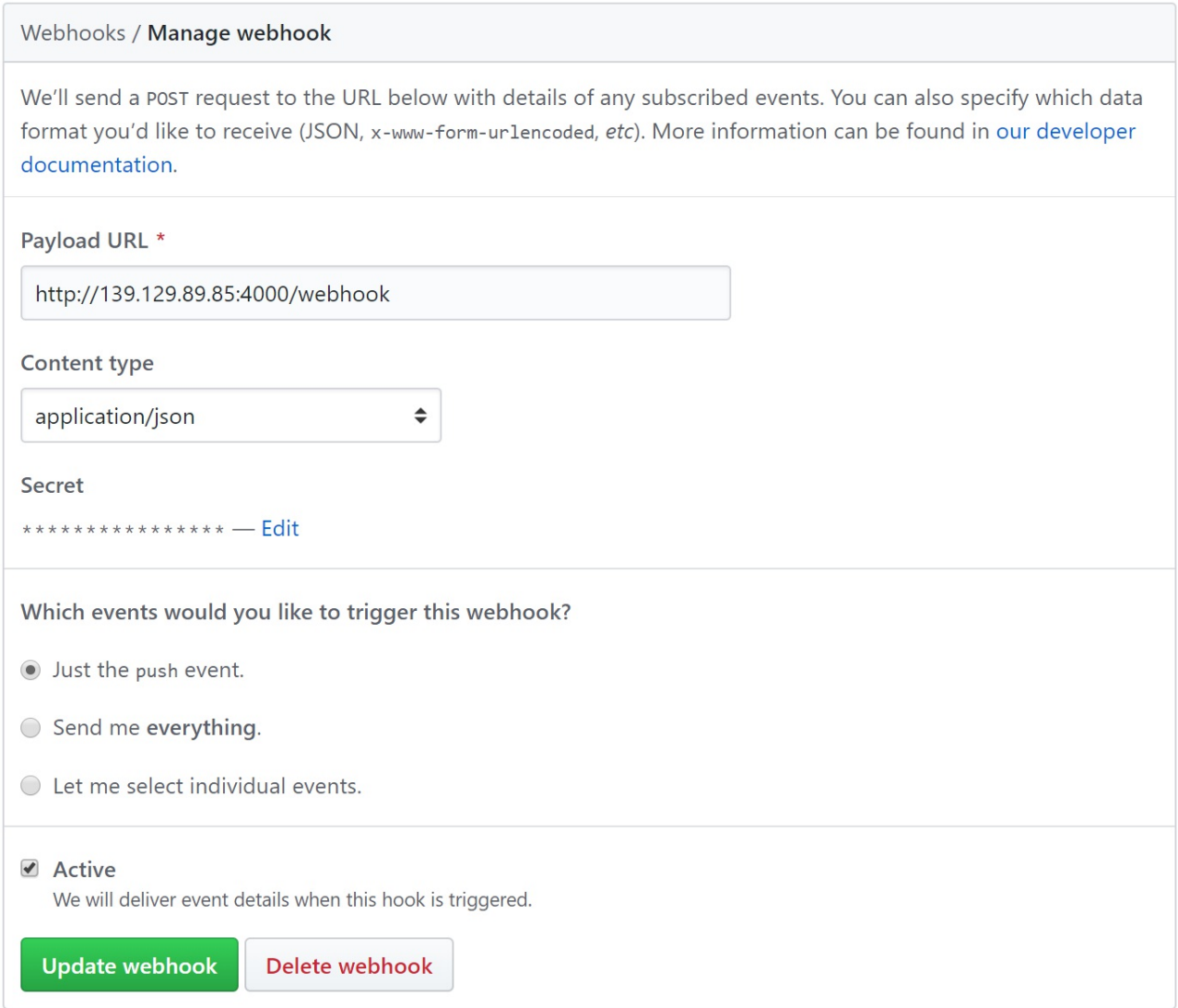
/usr/projects/vue-front/src/App.vue

```
      {{user.id}}:{{user.name}}

import axios from 'axios';
export default {
  name: 'app',
  data(){
    return {
      users:[]
    }
  },
  mounted(){
    axios.get('http://localhost:3000/api/users').then(response=>{
      this.users = response.data;
    });
  }
}
```

4. CICD服务器

- webhooks文档 (https://developer.github.com/webhooks/)
- pushevent (https://developer.github.com/v3/activity/events/types/#pushevent)



4.1 生成项目

```
mkdir vue-webhook
cd vue-webhook
cnpm init -y
cnpm i nodemailer -S
```

4.2 webhook.js

/usr/projects/vue-webhook/webhook.js

```
let http = require('http');
let crypto = require('crypto');
var spawn = require('child_process').spawn;
let sendMail = require('./sendMail');
const SECRET = '123456';
function sign (data) {
  return 'sha1=' + crypto.createHmac('sha1', SECRET).update(data).digest('hex')
}
let server = http.createServer(function(req,res){
  console.log(req.method,req.url);
  if(req.url == '/webhook' && req.method =='POST'){
    let buffers = [];
    req.on('data',function(data){
      buffers.push(data);
    });
    req.on('end',function(){
      let body = Buffer.concat(buffers);
      let sig   = req.headers['x-hub-signature'];
      let event = req.headers['x-github-event'];
      let id    = req.headers['x-github-delivery'];
      if(sig !== sign(body)){
        return res.end('Not Allowed');
      }
      res.setHeader('Content-Type','application/json');
      res.end(JSON.stringify({"ok":true}));

      if(event === 'push'){
        let payload = JSON.parse(body);
        let child = spawn('sh', [`./${payload.repository.name}.sh`]);
        let buffers = [];
        child.stdout.on('data', function (buffer) { buffers.push(buffer)});
        child.stdout.on('end', function () {
          let logs = Buffer.concat(buffers).toString();
          sendMail(`
            部署日期: ${new Date()}
            部署人: ${payload.pusher.name}
            部署邮箱: ${payload.pusher.email}
            提交信息: ${payload.head_commit&&payload.head_commit['message']}
            布署日志: ${logs.replace("\r\n",'')}
        `);
        });
      }
    });
  }else{
    res.end('Now Found!');
  }
});
server.listen(4000,()=>{
  console.log('服务正在4000端口上启动!');
});
```

4.3 sendMail.js

- nodemailer (https://nodemailer.com/smtp/well-known/)

/usr/projects/vue-webhook/sendMail.js

```
const nodemailer = require('nodemailer');
let transporter = nodemailer.createTransport({

    service: 'qq',
    port: 465,
    secureConnection: true,
    auth: {
        user: '836874012@qq.com',

        pass: 'zpdf0teyhjfbpcaff',
    }
});

function sendMail(message){
    let mailOptions = {
        from: '"836874012" ',
        to: '836874012@qq.com',
        subject: '部署通知',
        html:message
    };

    transporter.sendMail(mailOptions, (error, info) => {
        if (error) {
            return console.log(error);
        }
        console.log('Message sent: %s', info.messageId);
    });
}
module.exports = sendMail;
`
```

4.4 package.json

/usr/projects/vue-webhook/package.json

```
{
  "name": "vue-webhooks",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
+    "start": "pm2 start ./webhook.js --watch --name='vue-webhook'",
+    "stop": "pm2 stop vue-webhook"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "nodemailer": "^6.3.0"
  }
}
```

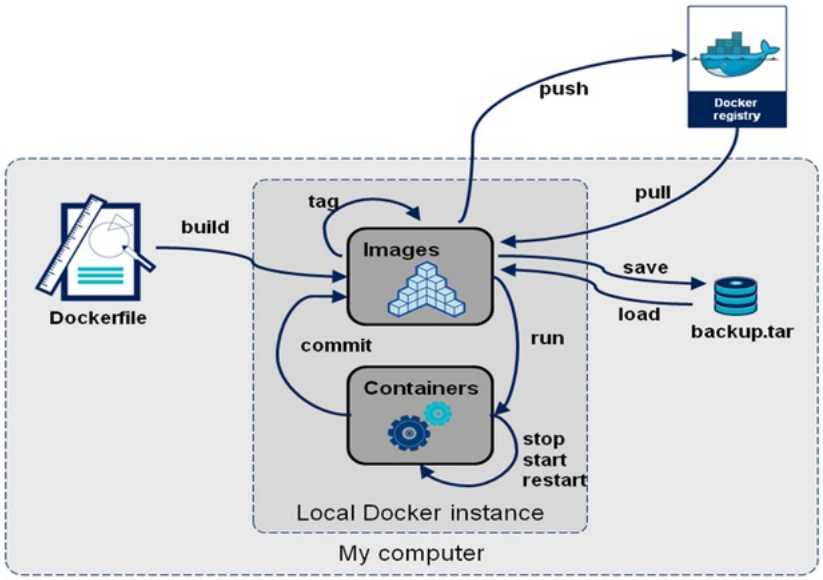
5. 配置服务器

5.1 更新系统

```
#升级所有包同时也升级软件和系统内核
yum update
#只升级所有包，不升级软件和系统内核
yum upgrade
```

5.2 docker是什么？#

- Docker 属于 Linux 容器的一种封装，提供简单易用的容器使用接口。
- Docker 将应用程序与该程序的依赖，打包在一个文件里面。运行这个文件，就会生成一个虚拟容器。程序在这个虚拟容器里运行，就好像在真实的物理机上运行一样



5.3 安装docker

```
yum install -y yum-utils   device-mapper-persistent-data   lvm2
yum-config-manager \
    --add-repo \
    https:
yum install -y docker-ce docker-ce-cli containerd.io
```

5.4 阿里云加速

```
mkdir -p /etc/docker
tee /etc/docker/daemon.json <
```

5.5 生成公钥并添加github

- https://github.com/settings/keys (https://github.com/settings/keys)

```
ssh-keygen -t rsa -b 4096 -C "zhufengnodejs@126.com"
cat /root/.ssh/id_rsa.pub
```

5.6 安装git

```
yum install git -y
git clone git@github.com:zhufengnodejs/vue-front.git
git clone git@github.com:zhufengnodejs/vue-back.git
git clone git@github.com:zhufengnodejs/vue-webhook.git
```

~.gitconfig

```
[alias]
    a = add -A
    c = commit -m"msg"
    p = push origin master
```

5.7 安装node和npm

- nvm (https://github.com/nvm-sh/nvm)

```
wget -qO- https:
source /root/.bashrc
nvm install stable
npm i nrm -g
nrm use taobao
npm i pm2 -g
```

5.8 安装启动服务

5.8.1 vue-back

```
cd /usr/projects/vue-back
npm i
npm run start
curl http:
```

5.8.2 vue-front

```
cd /usr/projects/vue-front
npm i
npm run serve
curl http:
```

5.8.3 vue-webhook

```
cd /usr/projects/vue-webhook
npm i
npm run start
curl http:
curl http:
```

6. 后台布署

6.1 vue-back.sh

/usr/projects/vue-webhook/vue-back.sh

```
WORK_PATH='/usr/projects/vue-back'
cd $WORK_PATH
echo "清理代码"
git reset --hard origin/master
git clean -f
echo "拉取最新代码"
git pull origin master
echo "开始构建镜像"
docker build -t vue-back .
echo "删除旧容器"
docker stop vue-back-container
docker rm vue-back-container
echo "启动新容器"
docker container run -p 3000:3000 -d --name vue-back-container vue-back
```

6.2 Dockerfile

/usr/projects/vue-back/Dockerfile

```
FROM node
LABEL name="vue-back"
LABEL version="1.0"
COPY . /app
WORKDIR /app
RUN npm install
EXPOSE 3000
CMD npm start
```

6.3 .dockerignore

/usr/projects/vue-back/.dockerignore

```
.git
node_modules
package-lock.json
Dockerfile
.dockerignore
```

7. 前台布署

7.1 vue-front.sh

/usr/projects/vue-webhook/vue-front.sh

```
WORK_PATH='/usr/projects/vue-front'
cd $WORK_PATH
echo "清理代码"
git reset --hard origin/master
git clean -f
echo "拉取最新代码"
git pull origin master
echo "打包最新代码"
npm run build
echo "开始构建镜像"
docker build -t vue-front .
echo "删除旧容器"
docker stop vue-front-container
docker rm vue-front-container
echo "启动新容器"
docker container run -p 80:80 -d --name vue-front-container vue-front
```

7.2 Dockerfile

/usr/projects/vue-front/Dockerfile

```
FROM nginx
LABEL name="vue-front"
LABEL version="1.0"
COPY  ./dist/ /usr/share/nginx/html/
COPY ./vue-front.conf /etc/nginx/conf.d/
EXPOSE 80
```

7.3 vue-front.conf

/usr/projects/vue-front/vue-front.conf

```
server {
    listen       80;
    server_name  47.104.15.123;
    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
        try_files $uri $uri/ /index.html;
    }
    location /api {
      proxy_pass http://47.104.15.123:3000;
    }
}
```

7.4 .dockerignore

/usr/projects/vue-front/.dockerignore

```
.git
node_modules
package-lock.json
Dockerfile
.dockerignore
```

8. 集成部署

- Compose 通过一个配置文件来管理多个Docker容器
- 在配置文件中，所有的容器通过services来定义，然后使用docker-compose脚本来启动、停止和重启应用和应用中的服务以及所有依赖服务的容器
- 最后，运行docker-compose up，Compose 将启动并运行整个应用程序 配置文件组成

8.1 docker-compose.yml

/usr/projects/docker-compose.yml

```
version: '2'
services:
 api:
  build:
    context: ./vue-back
    dockerfile: Dockerfile
  ports:
    - "3000:3000"
 web:
  build:
    context: ./vue-front
    dockerfile: Dockerfile
  ports:
    - "80:80"
```

8.2 安装docker-compose

```
curl -L https:
chmod +x /usr/local/bin/docker-compose
```

8.3 启动服务

```
docker-compose up
docker-compouse up -d
```

8.4 cicd.sh

/usr/projects/vue-webhook/cicd.sh

```

WORK_PATH='/usr/projects/vue-back'
cd $WORK_PATH
echo "清理后台代码"
git reset --hard origin/master
git clean -f
echo "拉取后台最新代码"
git pull origin master

WORK_PATH='/usr/projects/vue-front'
cd $WORK_PATH
echo "清理前台代码"
git reset --hard origin/master
git clean -f
echo "拉取前台最新代码"
git pull origin master
echo "打包前台最新代码"
npm run build

cd /usr/projects
echo "删除老资源"
docker-compose down
echo "重启所有服务"
docker-compose up -d
```