

## 14 - 部署上线 - 智能协同云图库项目教程 - 编程导航教程

本节重点本节重点内容是项目部署上线，可以独立学习，希望大家能够掌握这种快速上线项目的方法。

### 本节重点

本节重点内容是项目部署上线，可以独立学习，希望大家能够掌握这种快速上线项目的方法。

包括：

aS5CfKr07L5zWtS4lLRbsniG3CEXadUg2Unqe5CHpjo=

1. 服务器初始化
2. 部署规划
3. 安装依赖
4. 后端部署
5. 前端部署
6. 测试验证
7. 扩展知识

本文对应视频教程：

<https://bilibili.com/video/BV1akwGeSERK>

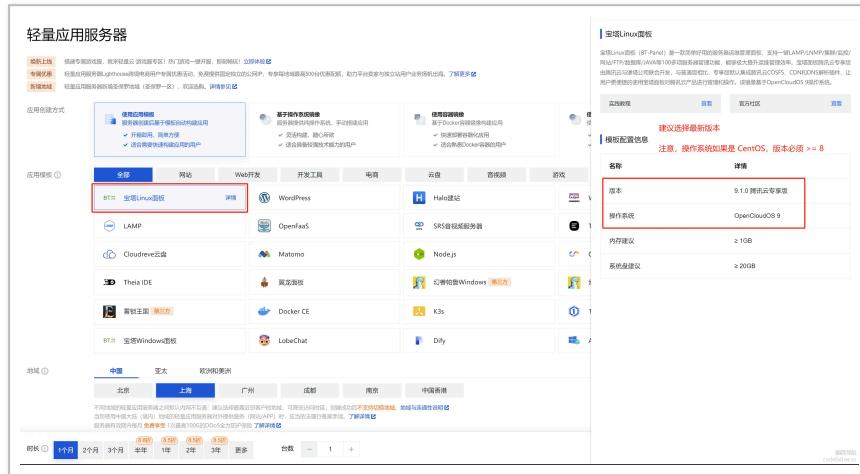
### 一、服务器初始化

首先购买一台服务器，各大云服务商的新用户都比较便宜，建议先看 [云产品](#) 页面。

推荐选择轻量应用服务器，提供了很多开箱即用的模板，帮我们预装了环境和软件，省时省力。

鱼皮这里选择一台预装了宝塔 Linux 应用的轻量应用服务器，配置为 2 核 2 G，部署咱们的项目足够了。应用模板一般选择最新版本就好了，如下图：

HjpOe/rdEcCehdPVL1u7j0N2XeKJoa/vakFU+1wtsMo=

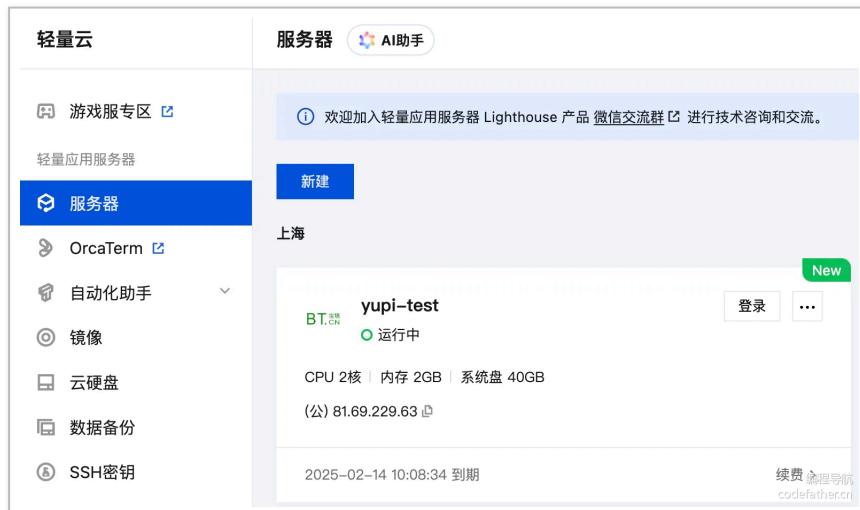


The screenshot shows a web interface for selecting a 'Lightweight Application Server'. At the top, there are several tabs: '轻量应用服务器' (Lightweight Application Server), '云主机' (Cloud Host), '容器' (Container), '函数' (Function), and '游戏' (Game). The '轻量应用服务器' tab is active. Below the tabs, there are two main sections: '轻量应用模板' (Lightweight Application Template) and '轻量应用机房' (Lightweight Application Room). The '轻量应用模板' section is expanded, showing a list of templates with checkboxes. The '宝塔Linux模板' (TongCloud Linux Template) is checked and highlighted with a red box. To the right of this section, there is a table with columns '名称' (Name) and '详细' (Details). The first row in the table is also highlighted with a red box, showing '版本' (Version) as '9.1.0 智云专享版' and '操作系统' (Operating System) as 'OpenCloudOS 9'. Other columns in the table include '内存建议' (Memory Recommendation) and '系统盘建议' (System Disk Recommendation). At the bottom of the page, there are filters for '地域' (Region) and '时间' (Time), and a '搜索' (Search) bar.

宝塔 Linux 是一个可视化 Linux 运维管理工具，提供了很多帮助我们管理服务器的功能，适合中小团队或者个人学习使用。

购买好服务器后，进入控制台，可以看到新增的服务器信息，注意不要主动对外暴露公网 IP！

aS5CfKr07L5zWtS4iLRbsniG3CEXadUg2Unqe5CHpjo=



The screenshot shows the 'Lightweight Cloud' (轻量云) control panel. On the left, there is a sidebar with several options: '游戏服专区' (Game Server专区), '轻量应用服务器' (Lightweight Application Server), '服务器' (Server) (which is highlighted with a red box), 'OrcaTerm' (终端), '自动化助手' (Automation Assistant), '镜像' (Image), '云硬盘' (Cloud Disk), '数据备份' (Data Backup), and 'SSH密钥' (SSH Key). The main content area has a '欢迎加入轻量应用服务器 Lighthouse 产品 微信交流群' (Welcome to the Lightwave Application Server Lighthouse product WeChat group) message. Below this, there is a '新建' (Create) button and a list of servers. One server, 'yupi-test', is shown in detail: it is running in the '上海' (Shanghai) region, has a status of '运行中' (Running), and is associated with a 'BT.CN' (BT.CN) cloud disk. Its specifications are listed as 'CPU 2核 | 内存 2GB | 系统盘 40GB'. The server's IP address is '(公) 81.69.229.63'. At the bottom of the server card, it shows the creation date and time as '2025-02-14 10:08:34 到期' (Created on 2025-02-14 10:08:34, Valid until) and a '续费' (Renewal) button.

点击服务器进入详情页，在防火墙标签页中放通 8888 宝塔面板端口，否则无法在自己的电脑上访问宝塔。

应用类型	来源	协议	端口	策略	备注	操作
自定义	0.0.0.0	TCP	8888	允许	宝塔Linux面板默认端口	<a href="#">编辑</a> <a href="#">删除</a>
HTTP (80)	0.0.0.0	TCP	80	允许	Web服务器HTTP (80), 如 Apache, Nginx	<a href="#">编辑</a> <a href="#">删除</a>
HTTPS (443)	0.0.0.0	TCP	443	允许	Web服务器HTTPS (443), 如 Apache, Nginx	<a href="#">编辑</a> <a href="#">删除</a>
Linux 登录 (22)	0.0.0.0	TCP	22	允许	Linux SSH登录	<a href="#">编辑</a> <a href="#">删除</a>
Windows登录 (3389)	0.0.0.0	TCP	3389	允许	Windows远程桌面登录	<a href="#">编辑</a> <a href="#">删除</a>
Windows登录优化 (3389)	0.0.0.0	UDP	3389	允许	Windows远程桌面登录优化	<a href="#">编辑</a> <a href="#">删除</a>

新版本的轻量应用服务器已经自动为我们放通该端口。否则需要手动新增一条防火墙规则：

应用类型	来源	协议	端口	策略	备注
自定义	全部IPv4地址	TCP	8888	允许	宝塔

进入应用管理标签页，登录宝塔。

8fu8cDJmfdpbl/kjna5xbFRq8brk2wHoCLXGDPF6jqA=

首次登录时，需要先登录服务器，通过输入命令的方式获取宝塔默认账号密码，如图：

应用内软件信息

面板首页地址 <http://81.69.229.63>

面板端口 默认为8888，您可以在登录面板后修改面板端口  
(提示：请前往[防火墙](#)页面放行面板端口)

用户名与密码 请登录实例并执行以下命令来获取管理员用户名和密码

`sudo /etc/init.d/bt default`

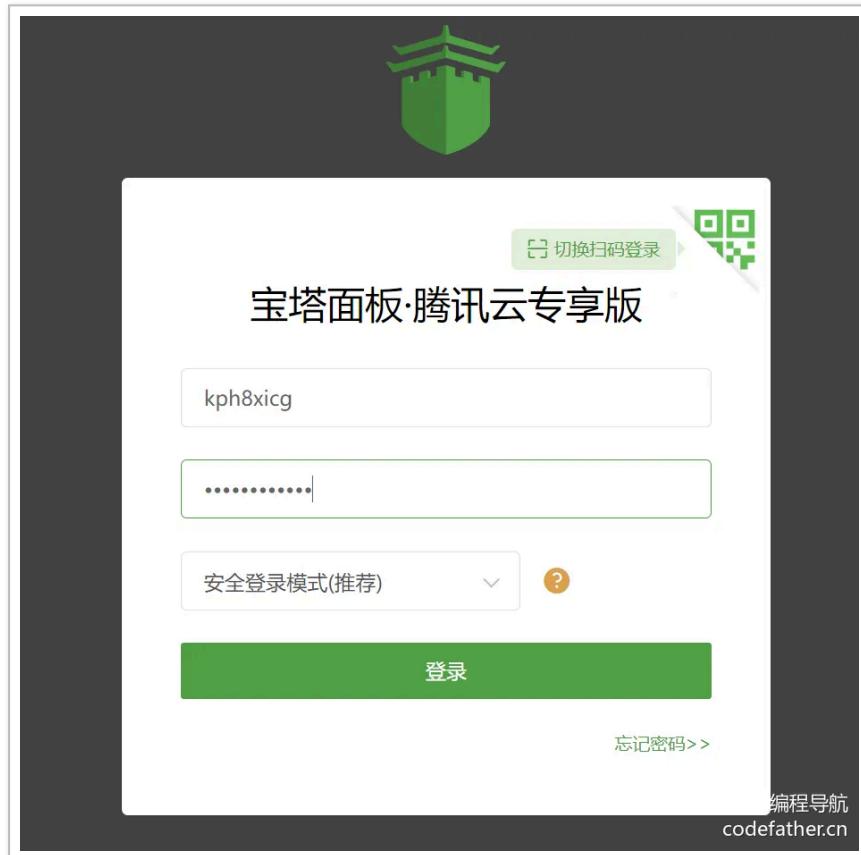
[登录](#)

点击登录后，进入到 web 终端，复制脚本并执行：

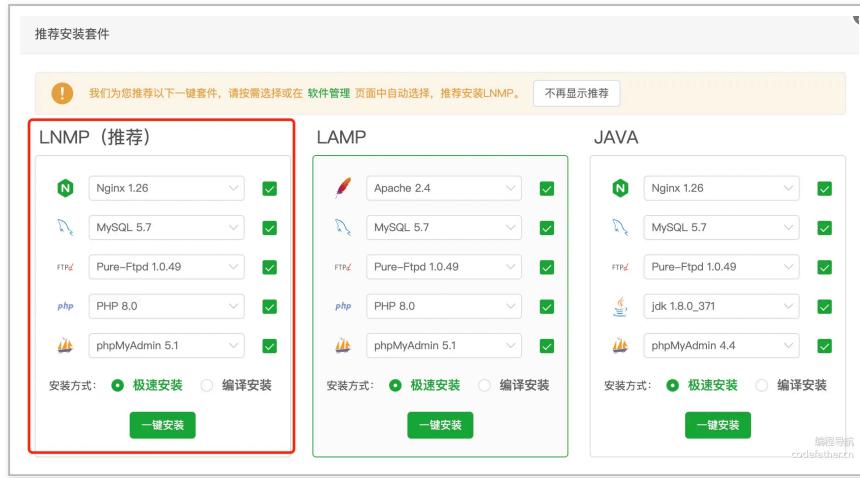
sETFLiz5z9hWsGMqkEZF5CeQ5DuZsw1zk9awkulmoc0=

```
[lighthouse@VM-16-6-opencloudos ~]$ sudo /etc/init.d/bt default
=====
BT-Panel default info!
=====
外网面板地址: http://81.69.229.63:8888/tencentcloud
内网面板地址: http://10.0.16.6:8888/tencentcloud
username: 18105ca1
password: 2793a21d0c5f
=====
Warning:
If you cannot access the panel,
release the following port (8888|888|80|443|20|21) in the security group
注意: 初始密码仅在首次登录面板前能正确获取, 其它时间请通过 bt 5 命令修改密码
=====
[lighthouse@VM-16-6-opencloudos ~]$
```

根据终端输出的信息，访问宝塔面板，输入初始用户名和密码：

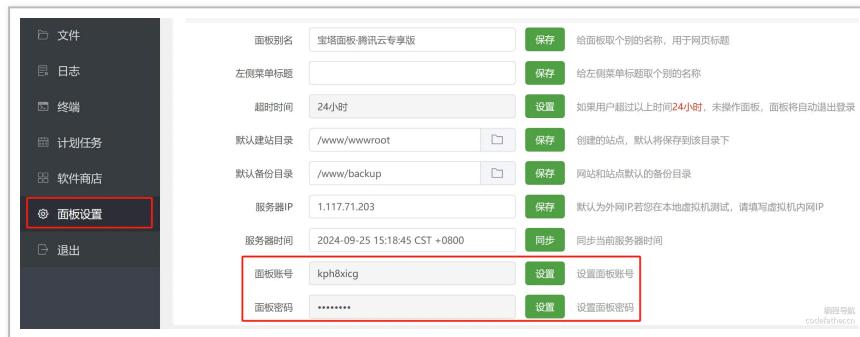


首次进入宝塔时，会提示我们安装环境，这里推荐安装 LNMP (包含 Nginx 服务器)，适合部署前后端分离的项目：



首次进入宝塔面板时，记得修改面板账号密码（每次修改完都要重新登录）：

Ny8IKBIOsoTTCgTymQbjLul9Gmod0P0TJLzDMNBV70g=



## 二、部署规划

在正式操作前后端部署前，我们要先进行一个规划，比如要部署哪些项目和服务、需要哪些依赖、占用哪些端口等。

HjpOe/rdEcCehdPVL1u7j0N2XeKJoa/vakFU+1wtsMo=

### 1、获取源码

本项目代码开源：<https://github.com/liyupi/yu-picture>

建议新手学习和部署 yu-picture-backend 目录，使用传统的分层架构：<https://github.com/liyupi/yu-picture/tree/master/yu-picture-backend>

meP/WqR3MNkD1e0lYI0pAe/471MZ0w9VR3IEl1E7

9H4=

有一定经验的同学可以学习部署 yu-picture-backend-ddd 目录，使用 DDD 领域驱动设计：

<https://github.com/liyupi/yu-picture/tree/master/yu-picture-backend-ddd>

但这两种架构的部署方式是一致的~

## 2、部署方案

为了提高效率，本项目前端和后端均使用宝塔面板进行部署，可以很方便地管理服务器。

涉及到具体的部署方式，前端要遵循 Vue 项目的部署模式，基于 Nginx 运行；后端可以直接利用宝塔的 Java 项目管理器运行 jar 包。

在鱼皮编程导航的 [代码生成器共享平台项目](#) 中，讲解过宝塔 + Nginx + 后端 Java 项目管理器（jar 包）的部署方式。

在鱼皮编程导航的 [AI 答题应用平台项目](#) 中，讲解过 Vercel + Docker + 云托管平台的部署方式，感兴趣的同学可以学习。基本上学会这几种部署方式，能够应对绝大多数部署需求了。

meP/WqR3MNkD1e0lYl0pAe/471MZ0w9VR3lE1E79H4=

## 3、地址规划

前端：通过 Nginx 进行转发，访问地址为 `http://{域名}`。

后端：通过 Nginx 进行转发，访问地址为 `http://{域名}/api`。

实际运行在 8123 端口。**JDK 建议选择 17 版本！**

meP/WqR3MNkD1e0lYl0pAe/471MZ0w9VR3lE1E79H4=

为什么要用 Nginx 转发？

前端和后端域名一致，保证不会出现跨域问题。

Nginx：服务器 80 端口，默认已安装。

sETFLiz5z9hWsGMqkEZf5CeQ5DuZsw1zk9awkulmoc0=

数据库：服务器 3306 端口，默认已安装。

Redis：服务器 6379 端口，需要手动安装。

## 4、注意事项

做好规划后，我们需要在腾讯云控制台的防火墙中开通需要外网访问的服务端口，比如 MySQL 和 Redis：



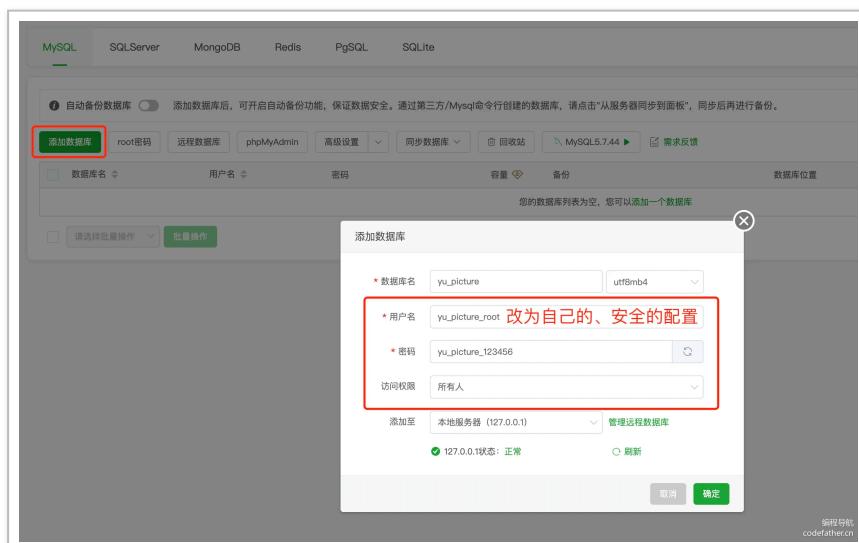
## 三、安装依赖

### 1、数据库

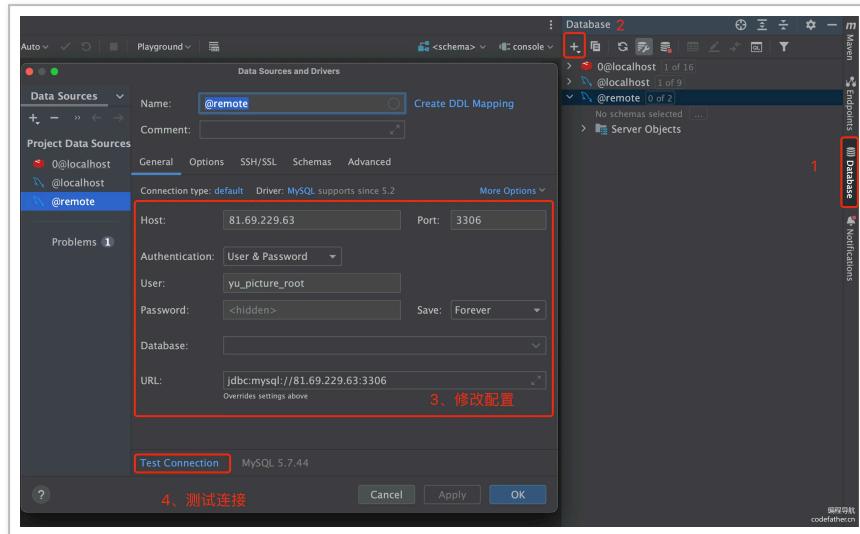
宝塔面板已经自动安装 MySQL 数据库，我们可以直接使用。

先为后端项目添加一个数据库。数据库名称和我们项目需要的数据库名称保持一致（此处为 mianshiya），注意用户名、密码和访问权限：

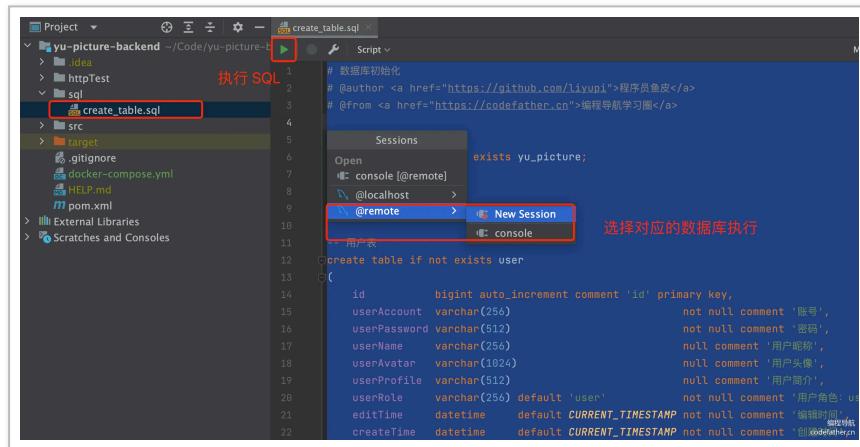
HjpOe/rdEcCehdPVL1u7j0N2XeKJoa/vakFU+1wtsMo=



在 IDEA 中打开后端项目，通过数据库面板在本地检查连接是否正常：

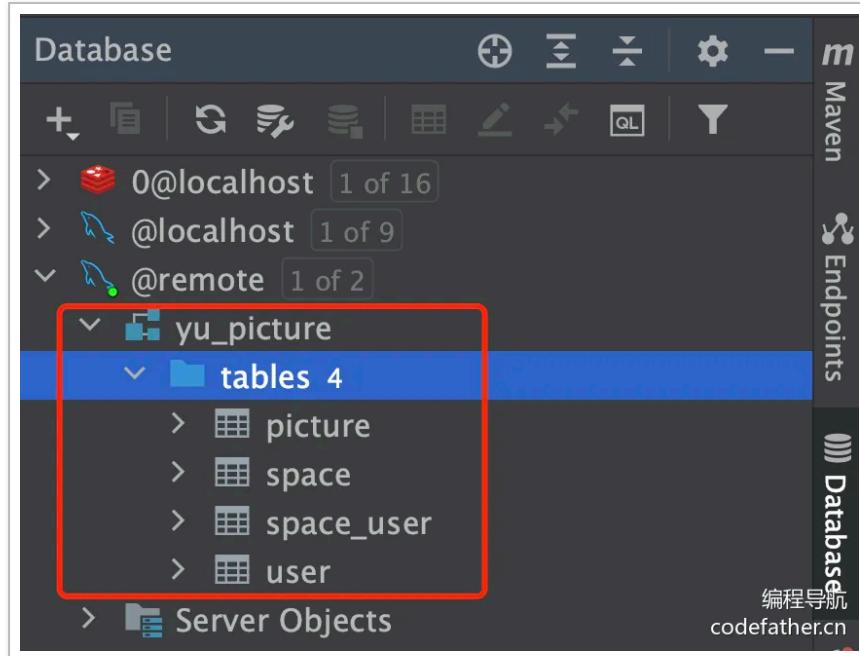


执行脚本，初始化库表：



记得验证数据库表是否创建成功，如下图：

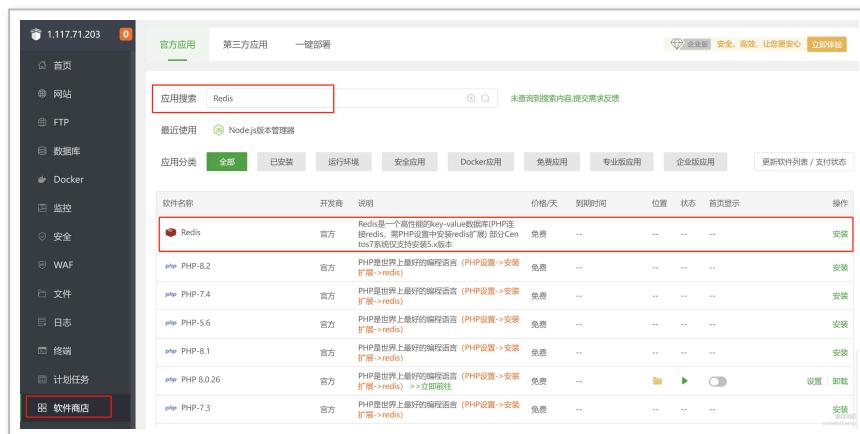
HjpOe/rdEcCehdPVL1u7j0N2XeKJoa/vakFU+1wtsMo=



## 2、Redis

在宝塔面板的软件商店中，搜索并安装 Redis：

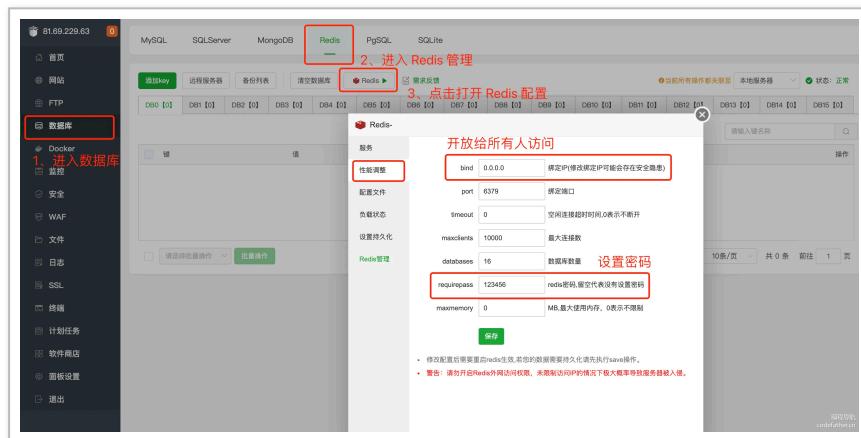
meP/WqR3MNkD1e0IYI0pAe/471MZQw9VR3IE1E79H4=



版本选择默认的即可：



安装完成后，需要配置 Redis，开启远程访问并配置密码，否则我们自己的电脑是无法连接 Redis 的：

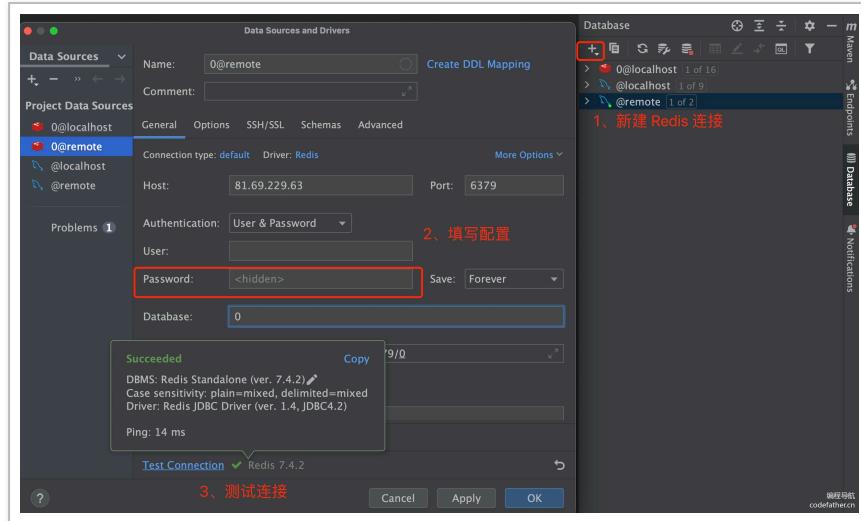


修改配置后，一定要重载配置才能生效：

Ny8IKBIOsoTTCgTymQbjLul9Gmod0P0TJLzDMNBV70g=

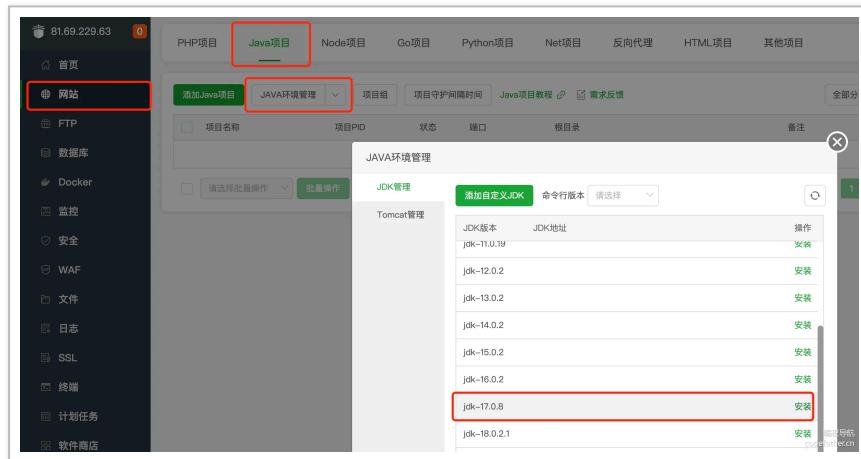


最后，在 IDEA 数据库面板中验证本地能否连接远程 Redis：



### 3、Java 环境

要部署 Java 项目，必须安装 JDK。在宝塔面板中，可以通过下图的方式快速安装指定版本的 JDK。此处我们先安装 JDK 17：



建议多安装几个版本，比如 JDK 8、11、17，需要用哪个版本的时候可以随时切换。

### 4、其他服务

比如 [腾讯云 COS 对象存储](#)、[阿里云百炼 AI](#)，可以去对应的官网开通。

8fu8cDJmfpbl/kjna5xbFRq8brk2wHoCLXGDPF6jqA=

如果不会开通的话，可以通过第 4 章教程开通 COS 对象存储，第 9 章教程开通阿里云百炼 AI。

注意，要给对象存储增加该服务器 IP（或者实际访问前端域名）的跨域配置，否则编辑图片时将无法正确加载图片。



接下来，我们分别进行后端和前端部署。

## 四、后端部署

### 1、修改配置

修改 `application-prod` 生产环境配置，包括数据库、Redis、对象存储、阿里云百炼 AI 的 key 等，替换为上述安装依赖时指定的配置（如用户名、密码）。

注意为了性能，还要关闭 MyBatis Plus 的日志；为了安全，要给 Knife4j 接口文档设置用户名和密码。

hh8kZr5oIIWSNoQ431Z2is4P7tu+SuQnXPZz37lHT1E=

参考配置如下：

```
server:
  port: 8123
  spring:

  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://81.69.229.63:3306/yu_picture
    username: yu_picture_root
    password: yu_picture_123456

  redis:
```

```
database: 0
host: 81.69.229.63
port: 6379
timeout: 5000
password: 123456
mybatis-plus:
  configuration:
    log-impl: ''

knife4j:
  basic:
    enable: true
    username: root
    password: 123456

cos:
  client:
    host: xxx
    secretId: xxx
    secretKey: xxx
    region: xxx
    bucket: xxxx

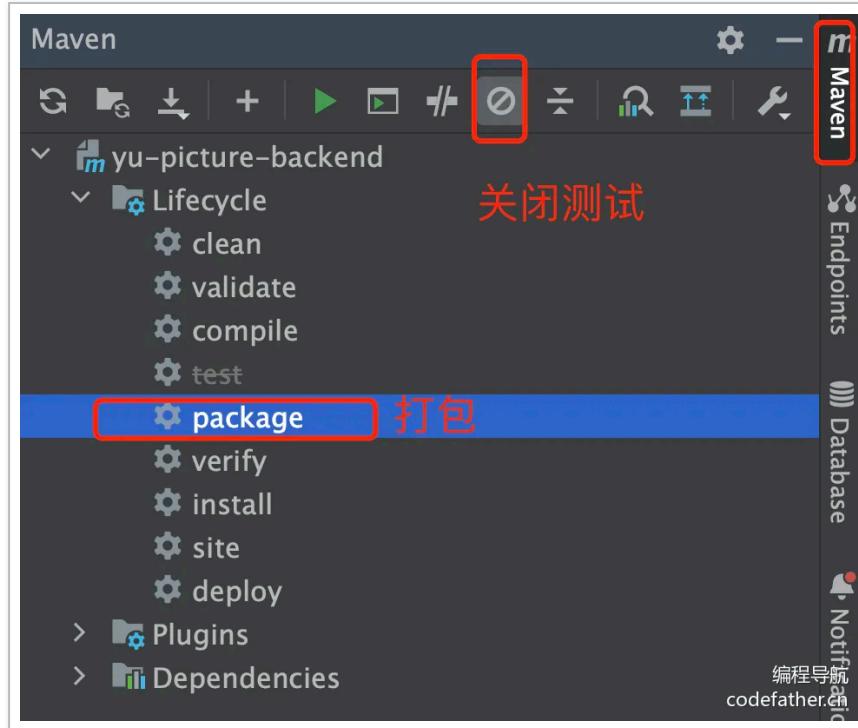
aliYunAi:
  apiKey: xxxx
```

## 2、打包部署

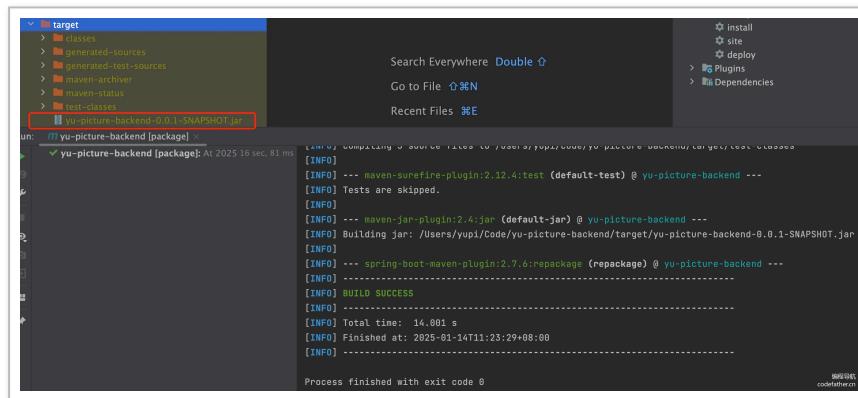
首先更改 pom.xml 文件的打包配置，删除掉主类配置的  
`skip` 配置，才能打包：

```
<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
      <version>${spring-boot.version}</version>
      <configuration>
        <mainClass>com.yupi.yupicturebackend.YuPictureBackend
        <skip>true</skip>
      </configuration>
    </plugin>
  </plugins>
</build>
```

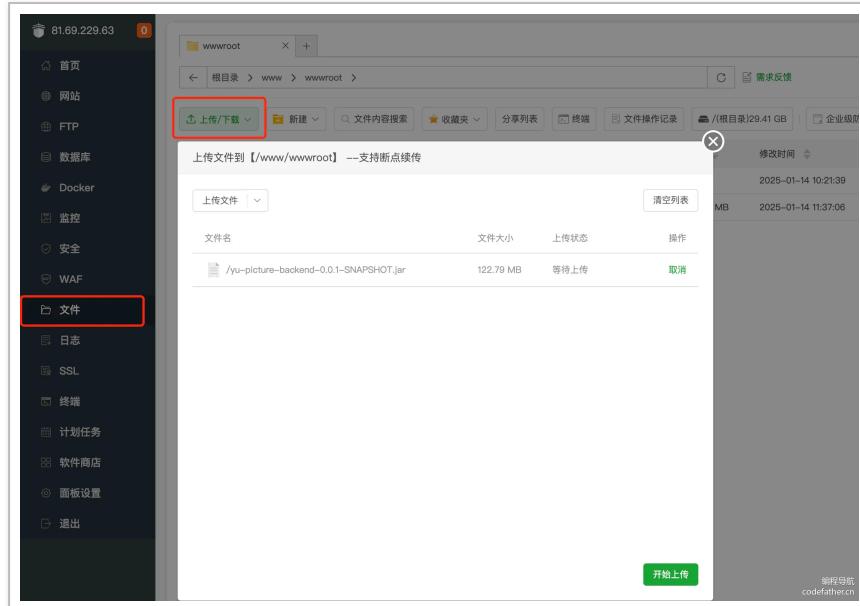
在 IDEA 中打开后端项目，忽略测试并打包：  
HjpOe/rdEcCehdPVL1u7j0N2XeKJoa/vakFU+1wtsMo=



打包成功，得到 jar 包文件：

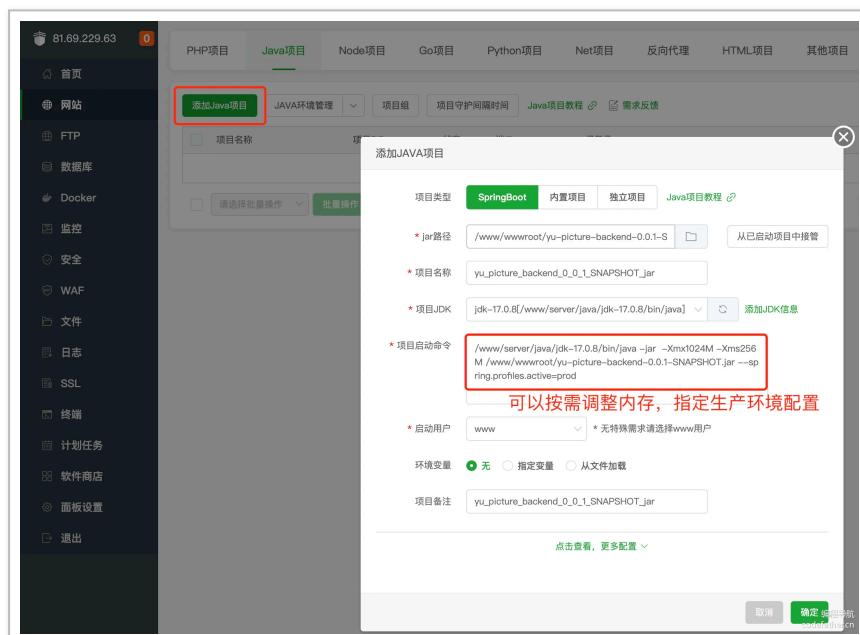


上传 jar 包到服务器，此处为了方便，就放到 web 根目录：



然后添加 Java 项目，\*\* 在项目执行命令中，必须指定生产环境的配置！ \*\* 还可以根据需要调整内存：

Ny8IKBIOsoTTCgTymQbjLul9Gmod0P0TJLzDMNBV70g=



启动成功后，能够看到状态和端口占用如图：



如果发现启动失败，需要先观察日志，下图仅为一个示例：

但是，我们现在无法通过浏览器访问接口文档：

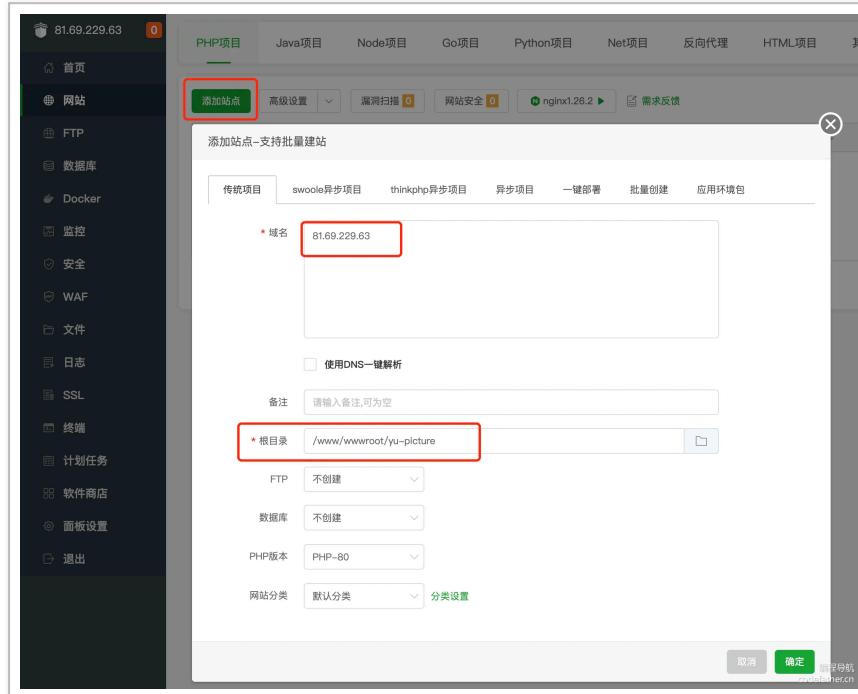
<http://81.69.229.63:8123/api/doc.html> aS5CfKr07L5zWtS4IL  
RbsniG3CEXadUg2Unqe5CHpjo=

A screenshot of a browser error page. At the top left is a small icon of a document with a red 'X'. The main title '无法访问此网站' (Cannot access this website) is in large black font. Below it, the subtext '连接已重置。' (Connection reset) is in smaller black font. A section titled '请试试以下办法:' (Please try the following methods:) lists two items: '检查网络连接' (Check network connection) and '检查代理服务器和防火墙' (Check proxy server and firewall). The error code 'ERR\_CONNECTION\_RESET' is shown at the bottom left. At the bottom right are two buttons: a white '详情' (Details) button with a black outline and a blue '重新加载' (Reload) button with white text and a black outline.

这是因为我们的服务器防火墙没有放开 8123 端口。**这里我们故意不放开**，因为在之前的部署规划中，后端需要通过 Nginx 进行转发，从而解决跨域问题。

### 3、Nginx 转发

新建一个 Nginx 站点，域名填写当前服务器 IP 或者自己的域名，根目录随意填写即可（只要不包含中文）：



如果访问的是后端接口（地址有 `/api` 前缀），则 Nginx 将请求转发到后端服务，对应配置代码如下：

```
8fu8cDJmfdbbl/kjna5xbFRq8brk2wHoCLXGDPF6jqA=
```

```
location /api {  
    proxy_pass http://127.0.0.1:8123;  
    proxy_set_header Host $proxy_host;  
    proxy_set_header X-Real-IP $remote_addr;  
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
    proxy_buffering off;  
    proxy_set_header Connection "";  
}
```

但是，对于本项目，光有 HTTP 转发配置还不够！后端还需要提供 WebSocket 连接，所以也要对 WebSocket 进行转发，再给 Nginx 补充下列配置：

```
location /api/ws {  
    proxy_pass http://127.0.0.1:8123;  
    proxy_http_version 1.1;  
    proxy_set_header Upgrade $http_upgrade;  
    proxy_set_header Connection "upgrade";  
    proxy_set_header Host $host;  
    proxy_set_header X-Real-IP $remote_addr;
```

```

proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_buffering off;
proxy_read_timeout 86400s;
}

```

修改 Nginx 配置如图：



```

#站点修改[81.69.229.63] -- 添加时间[2025-01-14 11:41:32]
#提示: Ctrl+F 搜索关键字, Ctrl+S 保存, Ctrl+H 查找替换

#域名管理
#子目录绑定
#网站目录
#访问限制
#流量限制
#伪静态
#默认文档
#配置文件
#SSL
#PHP
#重定向
#反向代理
#防盗链
#防篡改
#网站安全

57  # {
58  #   expires 12h;
59  #   error_log /dev/null;
60  #   access_log /dev/null;
61  #
62  location /api {
63    proxy_pass http://127.0.0.1:8123;
64    proxy_set_header Host $proxy_host;
65    proxy_set_header X-Real-IP $remote_addr;
66    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
67    proxy_buffering off;
68    proxy_set_header Connection "";
69  }
70  #
71  # 代理 WebSocket 连接 (专门用于 WebSocket 请求)
72  location /api/ws {
73    proxy_pass http://127.0.0.1:8123;
74    proxy_http_version 1.1;
75    proxy_set_header Upgrade $http_upgrade;
76    proxy_set_header Connection "upgrade";
77    proxy_set_header Host $host;
78    proxy_set_header X-Real-IP $remote_addr;
79    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
80    proxy_buffering off;
81  }
82  proxy_read_timeout 86400s;
83
84 }
85
86
87 }

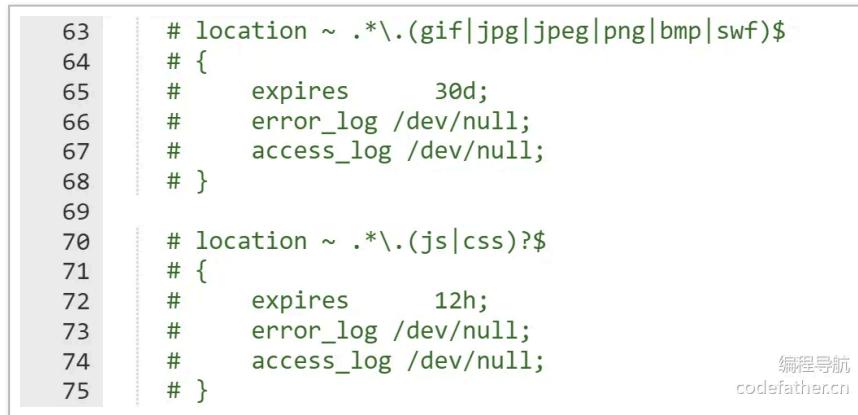
#保存 #历史文件 编程导航 codefather.cn

```

修改完后，就可以通过 80 端口（可以省略）访问到接口了。

Ny8IKBIOsoTTCgTymQbjLul9Gmod0P0TJLzDMNBV70g=

\*\*一定要注释掉下列配置！\*\* 否则访问接口文档时，静态资源的加载可能会出错。因为浏览器会从本地缓存加载资源，而不是动态请求资源。



```

# 63  # location ~ \.(gif|jpg|jpeg|png|bmp|swf)$
# 64  # {
# 65  #   expires 30d;
# 66  #   error_log /dev/null;
# 67  #   access_log /dev/null;
# 68  #
# 69  # 70  # location ~ \.(js|css)?$ {
# 71  #   expires 12h;
# 72  #   error_log /dev/null;
# 73  #   access_log /dev/null;
# 74  #
# 75  #

#保存 #历史文件 编程导航 codefather.cn

```

## 五、前端部署

前端部署可以参考 Vite 官方文档：

<https://cn.vitejs.dev/guide/static-deploy.html>

分为修改配置、打包部署和 Nginx 转发这 3 个步骤。

## 1、修改配置

线上的前端需要请求线上的后端接口，所以需要修改

`request.ts` 文件中的请求地址为线上：

```
const DEV_BASE_URL = "http://localhost:8123";
const PROD_BASE_URL = "http://81.69.229.63";

const myAxios = axios.create({
  baseURL: PROD_BASE_URL,
  timeout: 10000,
  withCredentials: true,
});
```

此外，由于本项目用到了 WebSocket，还要同步修改

`pictureEditWebSocket.ts` 文件中的 WebSocket 的连接地址：

sETFLiz5z9hWsGMqkEZF5CeQ5DuZsw1zk9awkulmoc0=

```
const DEV_BASE_URL = "ws://localhost:8123";
const PROD_BASE_URL = "ws://81.69.229.63";
const url = `${PROD_BASE_URL}/api/ws/picture/edit?pictureId=${this.pi
```



## 2、打包部署

1) 参考 Vite 官网，在 `package.json` 文件中定义 `pure-build`

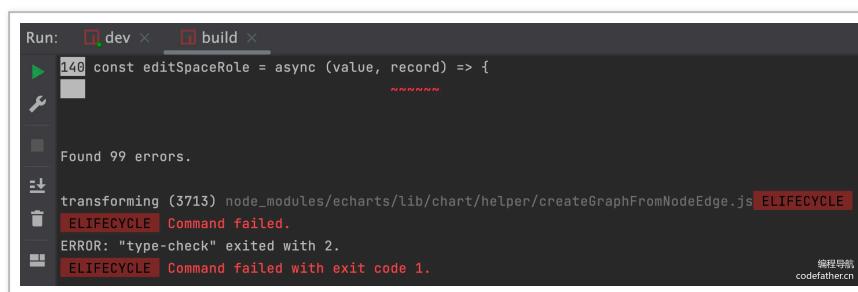
命令：4wYclrFsGRfsAG69iwgDTRz7WD7yXp2uhr9plK6qQJI=

```
{
  "scripts": {
    "dev": "vite",
    "pure-build": "vite build",
    "build": "run-p type-check \"build-only {@}\\\" --",
  }
}
```

为什么明明已经有 `build` 命令了，我们还要自己定义 `pure-build` 命令呢？

因为脚手架内置的 `build` 命令会执行类型检查，如果项目代码中有任何类型不规范的地方，都会导致打包失败！

meP/WqR3MNkD1e0lYI0pAe/471MZ0w9VR3IEl1E79H4=



```
Run: dev x build x
140 const editSpaceRole = async (value, record) => {
  ...
  ...
  Found 99 errors.
  ...
  ...
  transforming (3713) node_modules/echarts/lib/chart/helper/createGraphFromNodeEdge.js ELIFECYCLE
  ELIFECYCLE Command failed.
  ERROR: "type-check" exited with 2.
  ...
  ...
  ELIFECYCLE Command failed with exit code 1.
```

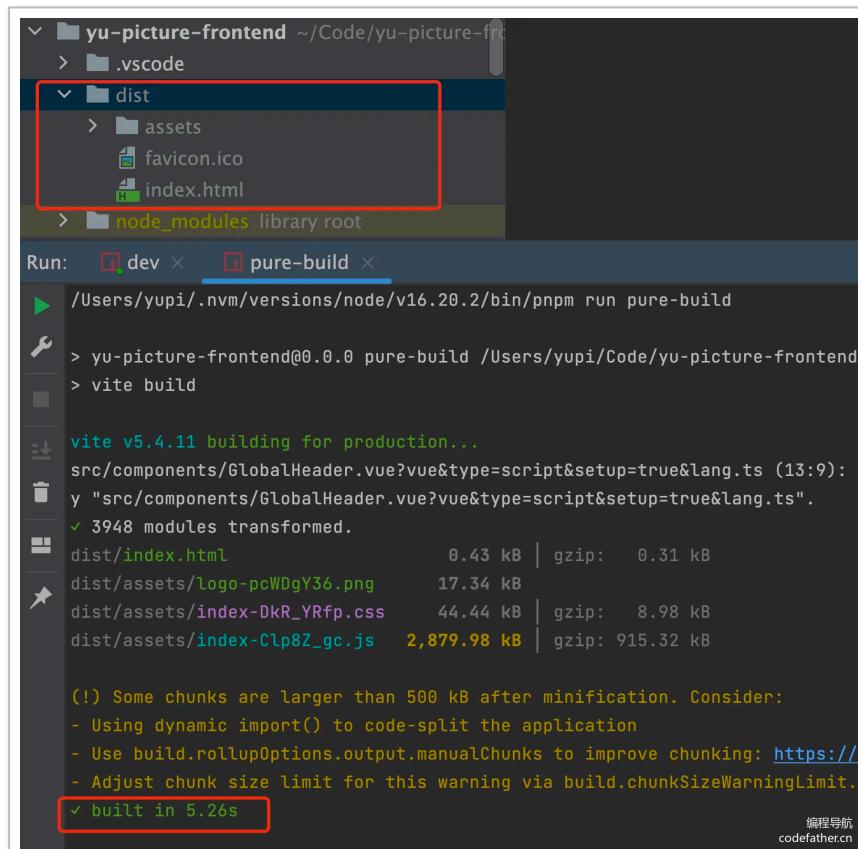
虽然可以自己一个个修复类型，但是太影响效率了，得不偿失，所以引入一个更干净的构建命令。

2) 执行 `pure-build` 命令，执行打包构建。

4wYClrFsGRfsAG69iwgDTRz7WD7yXp2uhr9plK6qQJl=

注意，如果 Node.js 版本较低，会构建失败，这时可以到 [官网](#) 安装更新的版本，比如 v20.17.0 等长期支持版本。

构建成功后，可以得到用于部署的静态文件 `dist` 目录：



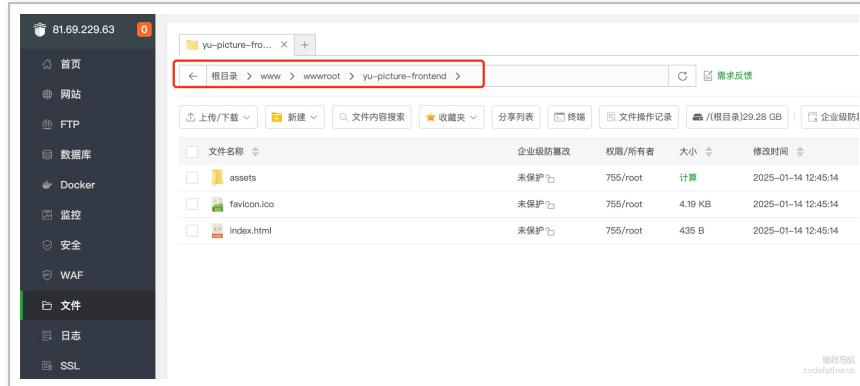
```
Run: dev x pure-build x
/Users/yupi/.nvm/versions/node/v16.20.2/bin/pnpm run pure-build
> yu-picture-frontend@0.0.0 pure-build /Users/yupi/Code/yu-picture-frontend
> vite build

vite v5.4.11 building for production...
src/components/GlobalHeader.vue?vue&type=script&setup=true&lang.ts (13:9):
  "src/components/GlobalHeader.vue?vue&type=script&setup=true&lang.ts".
  ✓ 3948 modules transformed.

dist/index.html          0.43 kB | gzip: 0.31 kB
dist/assets/logo-PCWDgY36.png 17.34 kB
dist/assets/index-DKR_YRfp.css 44.44 kB | gzip: 8.98 kB
dist/assets/index-Clp8Z_gc.js 2,879.98 kB | gzip: 915.32 kB

(!) Some chunks are larger than 500 kB after minification. Consider:
- Using dynamic import() to code-split the application
- Use build.rollupOptions.output.manualChunks to improve chunking: https://
- Adjust chunk size limit for this warning via build.chunkSizeWarningLimit.
✓ built in 5.26s
```

把 dist 目录下的所有文件上传到服务器上（可以新建一个 yu-picture-frontend 目录）。文件较多时，建议先在本地压缩，上传压缩包到服务器后再解压。如图：



### 3、Nginx 转发

一般来说，用户无法直接访问服务器上的文件，需要使用 Nginx 提供静态文件的访问能力。

修改已有站点的网站目录配置，指向前端文件根目录：



然后访问服务器地址（或者自己配置的域名），就能打开前端网站了：



但是经过验证，目前访问除了主页外的其他页面（比如 /add\_picture），如果刷新页面，就会出现 404 错误。  
Ny8IKBIOsoTTCgTymQbjLul9Gmod0P0TJLzDMNBV70g=



这个问题是由于 Vue 是单页面应用（前端路由），打包后的文件只有 `index.html`，服务器上不存在对应的页面文件（比如 /add\_picture.html），所以需要在 Nginx 配置转发。如果找不到某个页面文件，就加载主页 index.html 文件。

修改 Nginx 配置，补充下列代码：

meP/WqR3MNkD1e0lYl0pAe/471MZ0w9VR3lE1E79H4=

```
location / {  
    try_files $uri $uri/index.html /index.html;  
}
```

如图：

站点修改[81.69.229.63] -- 添加时间[2025-01-14 11:41:32]

域名管理	提示: Ctrl+F 搜索关键字, Ctrl+S 保存, Ctrl+H 查找替换
子目录绑定	25    #REWRITE-START URL 重写规则引用, 修改后将导致面板设置的伪静态规则失效 26    include /www/server/panel/vhost/rewrite/81.69.229.63.conf; 27    #REWRITE-END
网站目录	28 29    location / { 30      try_files \$uri \$uri/index.html /index.html; 31    } 32 33    #禁止访问的文件或目录 34    location ~ ^/(user.ini htaccess git env svn project LICENSE){ 35      return 404; 36    } 37 38    #一键申请SSL证书验证目录相关设置 39    location ~ \.well-known{ 40      allow all; 41    } 42 43
访问限制	
流量限制	
伪静态	
默认文档	
配置文件	
SSL	

编程导航  
codefather.cn

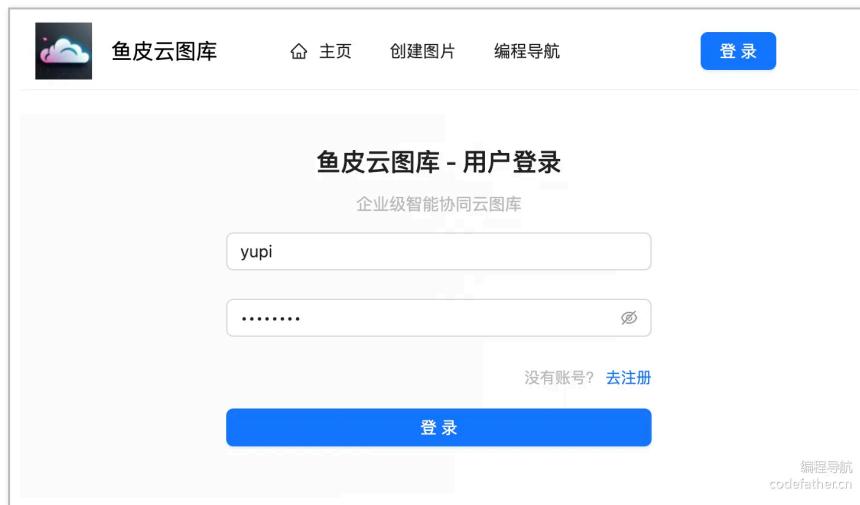
保存配置后, 再次刷新页面, 可以正常访问。

## 六、 测试验证

最后, 我们来对上线效果进行验证。

8fu8cDJmfdpbl/kjna5xbFRq8brk2wHoCLXGDPF6jqA=

### 1) 用户注册登录



然后通过修改数据库的方式, 将该用户的角色设置为管理员, 从而使用更多功能。

+4c8/N1p1ks+s/Pymte9p3eVkNVOHiOfNo0FGyT9zow=

### 2) 进入图片管理 => 批量创建图片页面, 抓取一批图片作为网站的初始数据



公共图库

我的空间

创建团队

批量创建图片

关键词: 二次元超清壁纸

抓取数量: 10

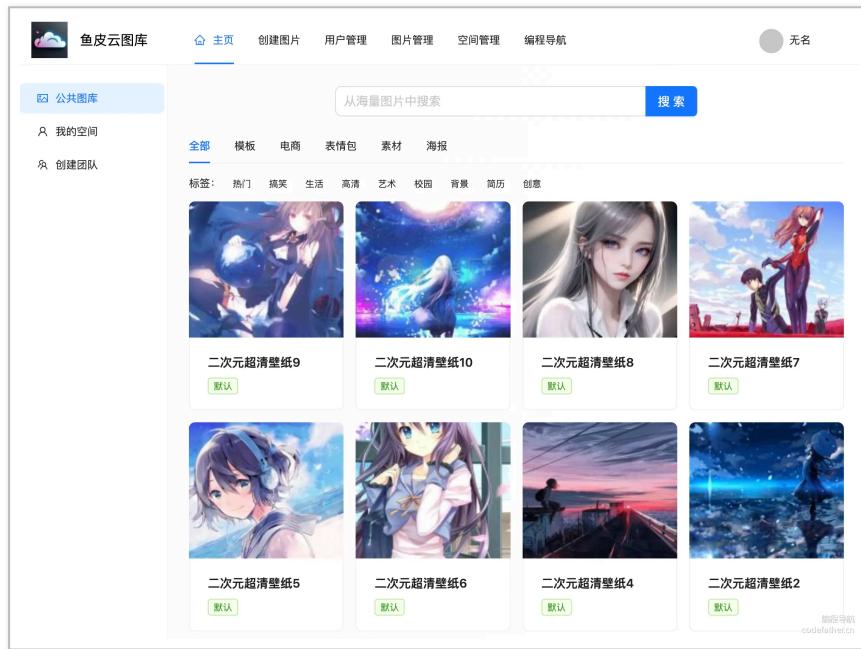
名称前缀: 二次元超清壁纸

执行任务

codeofather.cn

### 3) 进入主页, 查看到了公共图库

8fu8cDJmfdpbl/kjna5xbFRq8brk2wHoCLXGDPF6jqA=



公共图库

我的空间

创建团队

全部

模板 电商 表情包 素材 海报

标签: 热门 搞笑 生活 高清 艺术 校园 背景 简历 创意

二次元超清壁纸9

二次元超清壁纸10

二次元超清壁纸8

二次元超清壁纸7

二次元超清壁纸5

二次元超清壁纸6

二次元超清壁纸4

二次元超清壁纸2

从海量图片中搜索

搜索

codeofather.cn

### 4) 创建一个私有空间



公共图库

我的空间

创建团队

创建私有空间

空间名称: 鱼皮的测试空间

空间级别: 普通版

提交

空间级别介绍

目前仅支持开通普通版, 如需升级空间, 请联系 程序员鱼皮。

普通版: 大小 100.00 MB, 数量 100

专业版: 大小 1000.00 MB, 数量 1000

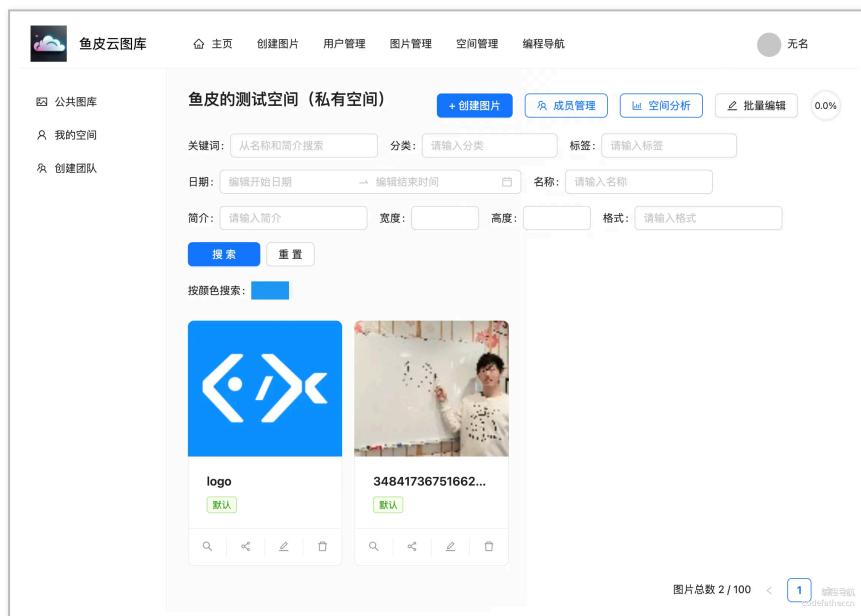
旗舰版: 大小 10000.00 MB, 数量 10000

codeofather.cn

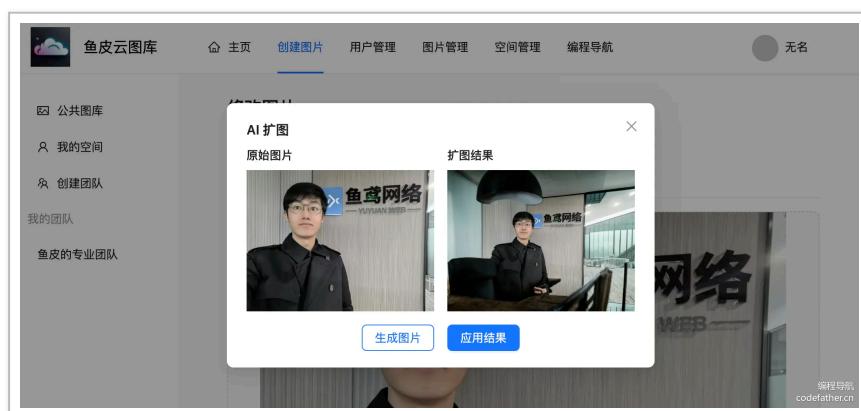
### 5) 通过文件上传和 URL 上传给私有空间上传一些图片:



6) 查看私有空间的图片，尝试各种搜索功能（比如按颜色搜索）：4wYclrFsGRfsAG69iwgDTRz7WD7yXp2uhr9plK6qQJI=



7) 使用 AI 扩图功能来编辑图片（基于 阿里云百炼 AI 实现）

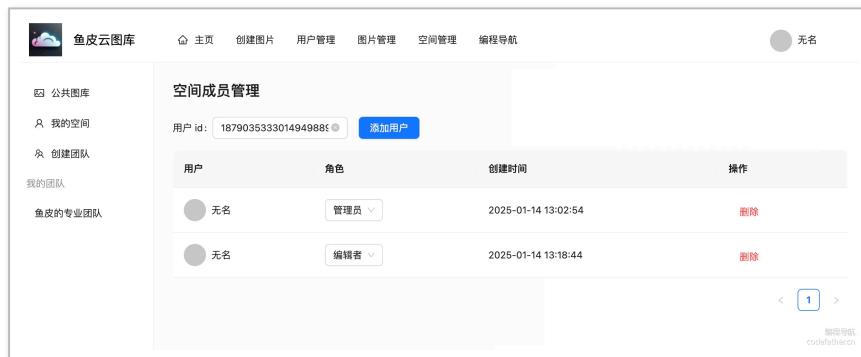


## 8) 创建团队空间



## 9) 给团队添加一位成员，设置角色为“编辑者”

meP/WqR3MNkD1e0lYl0pAe/471MZ0w9VR3lE1E79H4=



## 10) 给团队空间上传一张图片，然后让2名成员同时进入编辑：



如果编辑时，图片无法正常加载，可能是因为对象存储没有配置跨域，补充配置即可。

## 七、扩展知识

再分享一种更快部署后端的方法，可以利用 Docker + Docker Compose 快速部署后端依赖和后端项目本身。

Ny8IKBIOsoTTCgTymQbjLul9Gmod0P0TJLzDMNBV70g=

可以把 Docker 容器技术理解为安装操作系统时的镜像、或者安装 APP 时的安装包，只要定义好 Docker 配置文件，就能快速基于配置启动服务或项目。

而 Docker Compose 可以组合编排多个 Docker 容器，按照顺序快速启动多个服务或项目。

给大家提供一个示例的 Docker Compose 配置文件，定义了 MySQL、Redis 和 Spring Boot 项目的启动，大家可以基于这个文件进行定制修改：

meP/WqR3MNkD1e0lYl0pAe/471MZ0w9VR3lEl1E79H4=

```
version: '3.8'

services:

  mysql:
    image: mysql:8.0
    container_name: mysql_db
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: yu_picture
      MYSQL_USER: root
      MYSQL_PASSWORD: 123456
      TZ: Asia/Shanghai
    ports:
      - "3306:3306"
    volumes:
      - mysql_data:/var/lib/mysql
    command: --default-authentication-plugin=mysql_native_password

  redis:
    image: redis:5.0
    container_name: redis_cache
    restart: always
    ports:
      - "6379:6379"
    volumes:
```

```

- redis_data:/data
environment:
  TZ: Asia/Shanghai

springboot_app:
  image: openjdk:11-jre-slim
  container_name: springboot_app
  working_dir: /app
  volumes:
    - .:/app
  ports:
    - "8123:8123"
  environment:
    TZ: Asia/Shanghai
  command: [ "java", "-jar", "target/yu-picture-backend-0.0.1-SNAPS"]
  depends_on:
    - mysql
    - redis

volumes:
  mysql_data:
  redis_data:

```

有了配置文件后，就可以利用宝塔面板自带的 Docker 能力，去进行项目的部署了，感兴趣的同学可以尝试一下：



在鱼皮编程导航的 [OJ 在线判题项目教程](#) 中，讲解过基于 Docker + Docker Compose 快速部署微服务项目的方法，视频地址：<https://www.bilibili.com/video/BV1Cp4y1F7eA>

## 最后

至此，整个项目已经完成上线，希望大家能通过这个项目掌握企业级项目的开发、优化和上线方法，得到全方面编程技能和程序员素养的提升。

meP/WqR3MNkD1e0IYI0pAe/471MZ0w9VR3IE1E79H4=

# 本期作业

- 1) 完成项目的上线
- 2) 尝试给项目绑定域名、或者申请 HTTPS 证书（通过自行查阅资料实现）  
meP/WqR3MNkD1e0lYl0pAe/471MZ0w9VR3lE1E79H4=
- 3) 完成整个项目，并且自行增加 2 - 3 个扩展点，可以新增功能、也可以是某个优化、还可以将其他项目知识点融合到本项目中。

8fu8cDJmfdpbl/kjna5xbFRq8brk2wHoCLXGDPF6jqA=

---

全文完

---

本文由 简悦 SimpRead 优化，用以提升阅读体验

使用了 全新的简悦词法分析引擎 beta，[点击查看详细说明](#)

