# 第五人格组后端任务整理小结

1. 环境准备与框架搭建

由于前端所给的代码为静态HTML文件，因此首先需要准备相应的环境（这里需要的是python3、pip、Flask），并初步搭建起项目的结构。

在这里，首先就遭遇了环境配置失败的问题，无法用pip直接在系统python环境下安装包，在采取了另外一种方法——使用apt安装Flask等常见包，也遭遇了失败。因此，我不得不采用另外一种方法——创建虚拟环境进行安装。以下是大致操作：

回到你的项目目录：cd ~/idv\_site

创建并激活虚拟环境：python3 -m venv myenv;source myenv/bin/activate

安装依赖：pip install flask flask\_sqlalchemy werkzeug

编辑文件（不用 sudo）：nano login.html

运行项目：python3 app.py

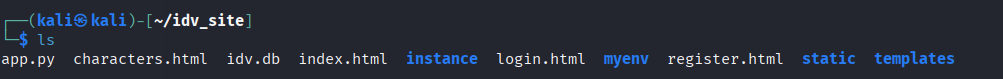
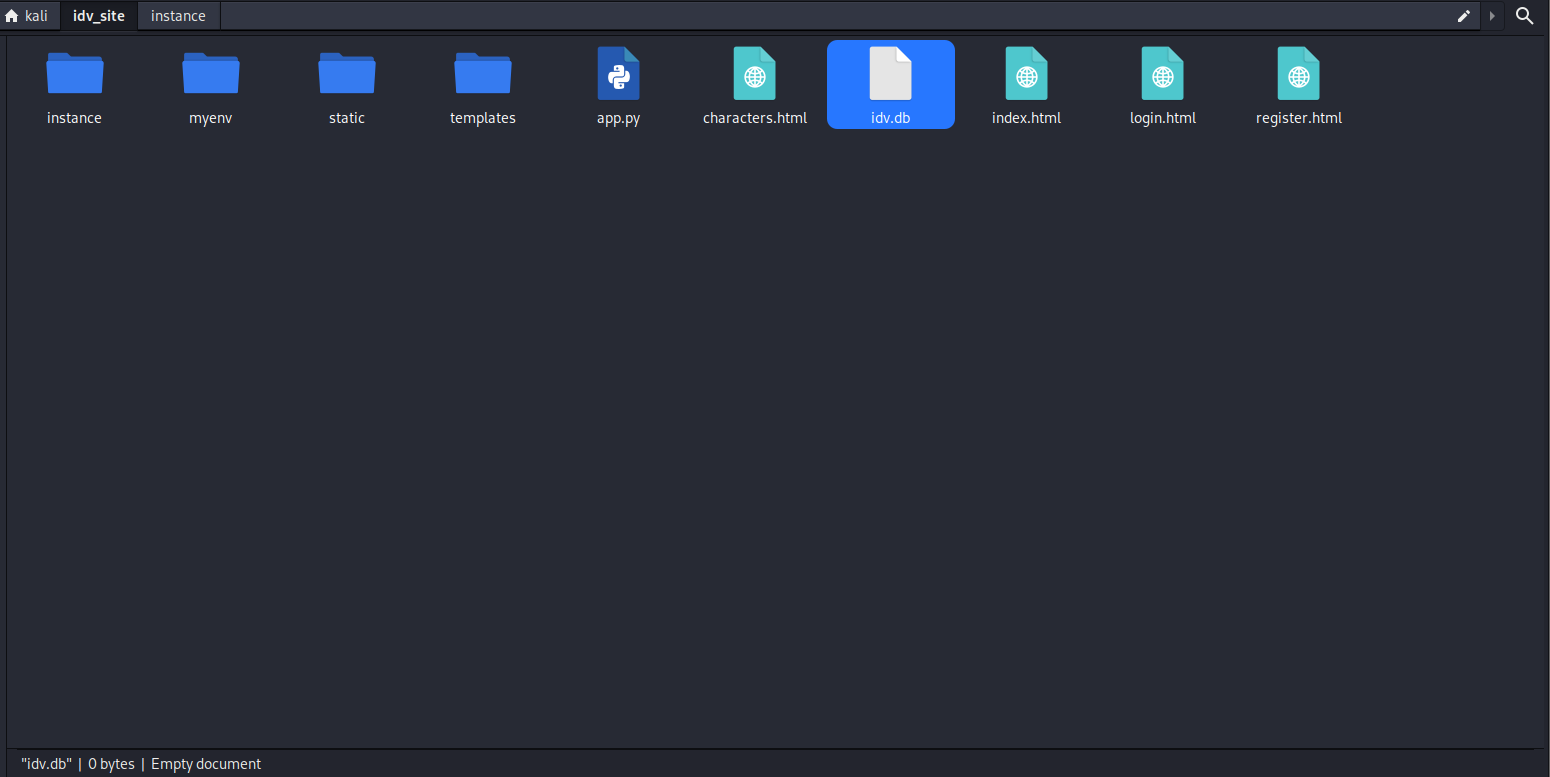


图1-项目基本结构

1. 创建数据库以及后端代码

在前端已有的login.html、register.html、content.html基础上，需要将其嵌入Flask实现的登录和注册功能之中（在这里直接复制到Flask的templates文件夹中即可）。之后，便是构建后端核心代码app.py。在这里出现了许多问题，比如数据库的创建、网站之间的跳转……

Flask的后端处理是一个复杂而艰难的过程，需要考虑到衔接以及进入与跳转的逻辑，这里是GPT所给的一段简短说明：

login.html——JS里的fetch提交给 /login，类型application/x-www-form-urlencoded，Flask后端用request.form获取参数，返回json，前端根据success字段决定跳转或显示错误。

register.html——JS里的fetch提交至 /register，类型application/json，Flask后端用request.get\_json()取参数，处理注册逻辑，返回json，JS弹窗是否注册成功。

进入/跳转逻辑——登录后用window.location.href = '/content';注册成功可以直接跳回登录页window.location.href = '/login';

通过以上操作最终实现了网站的顺畅跳转与运行。

数据库的创建中终端出现了以下报错：RuntimeError: Working outside of application context.这是因为需要将涉及到数据库的操作的代码（db.create\_all()）包裹到Flask应用上下文里。修改成如下形式：

if \_\_name\_\_ == "\_\_main\_\_":

with app.app\_context():

db.create\_all()

app.run(debug=True)

最终得到的数据库如下（账号正常，密码已经经过scrypt算法加密）：

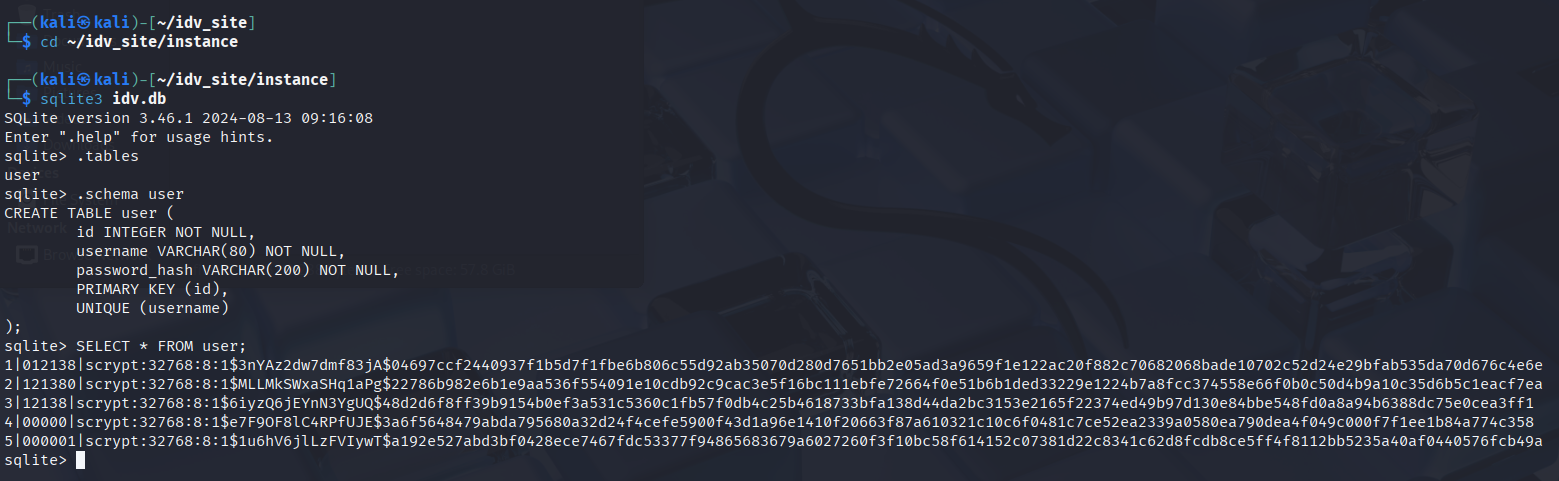


图2-数据库中的内容

1. 模板优化

在前端的HTML代码中，图片和音乐都未能正常展示，经过排查之后发现是由于图片路径拼接变量staticImageUrl未定义或者拼接方式有误。以下是正确格式：

|  |  |
| --- | --- |
| **功能** | **写法举例** |
| 图片 | {{ url\_for('static', filename='images/xxx.png') }} |
| 音乐 | {{ url\_for('static', filename='music/xxx.mp3') }} |
| 字体 | {{ url\_for('static', filename='images/xxx.ttf') }} |
| JS变量 | <script>const staticImageUrl = "{{ url\_for('static', filename='images/') }}";</script> |
| JS图片 | <img src="${staticImageUrl}${character.image}"> |

在此之后，部分图片可以正常展示，但部分图片仍然不能够正常展示。经过排查，在HTML文件中，发现我的原代码为card.innerHTML = ` <img src="${staticImageUrl}${character.image}" ...> `，为解决该问题，需要在在所有 JS 之前加上这句，用Jinja2渲染路径前缀，保证任何部署路径都正确：<script>const staticImageUrl = "{{ url\_for('static', filename='images/') }}";...</script>。在这样修改之后，图片音乐就能够正常展示啦！

1. 网络环境配置

这是最为艰难的一部分，如何配置网络环境使得宿主机能够访问所搭建的网站是一个全新的问题。在这部分我采用了桥接模式，并获取虚拟机的ip地址进行验证，之后配置网站服务器监听所有ip地址，确保防火墙放行指定服务器端口，之后再测试外部访问[http://你的ip地址/](http://10.21.78.179/)即可。

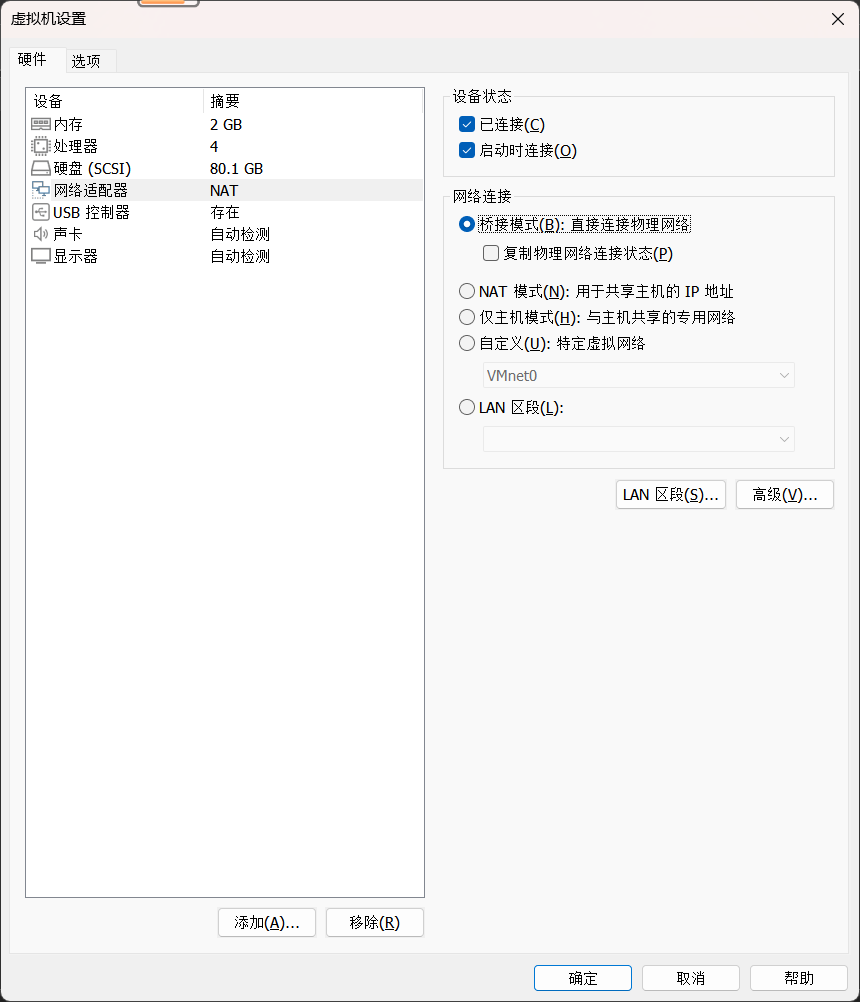


图3-修改虚拟机的网络连接

1. 后续优化问题

在宿主机成功访问网站后却发现跳出的并非是login.html，而是Apache2在Debian系统上的默认欢迎界面，接下来需要按照指示替换网页默认文件，复制网页文件到Apache目录，设置文件权限之后重启Apache服务，进行访问确认即可。

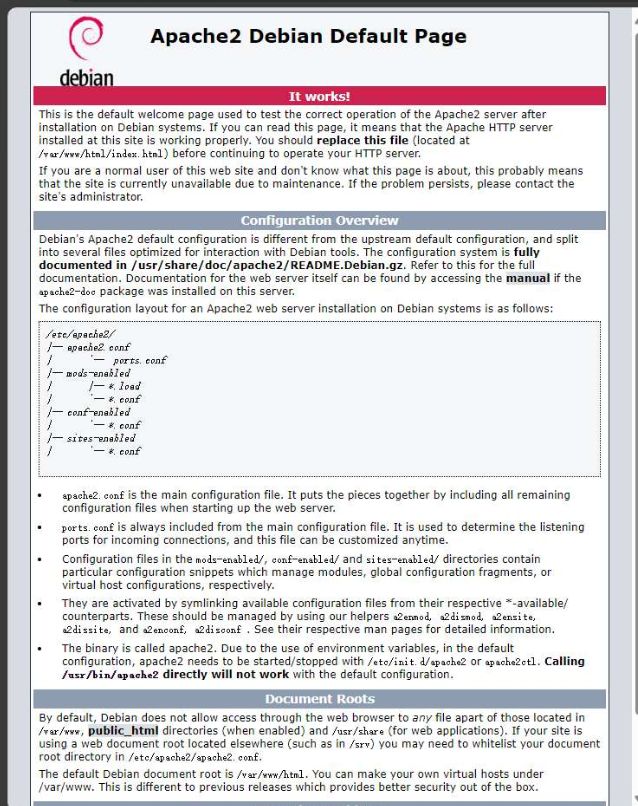


图4-错误界面其一

配置完成之后，出现的仍然不是login.html界面，而是目录列表界面（index.html）。因此我们需要自己创建一个index.html文件，同时调整Apache默认首页顺序，确保能够优先显示login.html，在这之后，重启Apache服务，即可看到正确的login.html页面了。

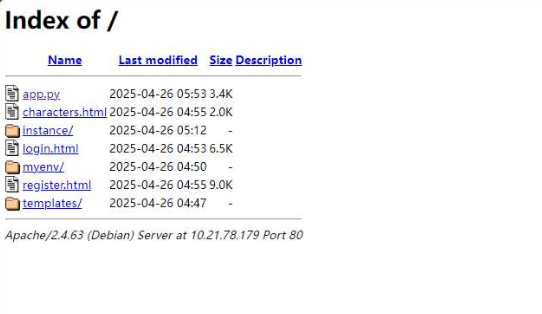


图5-错误界面其二

然而，在这之后，如果我们想要跳转到注册界面是不成功的，因此在这里我们需要配置Apache反向代理到Flask应用端口，之后重新启动并访问你的服务。



图6—错误界面其三