**实验7：用户身份认证及权限管理实验报告**

姓名：江继军 学号：21373222 班级：213921

实验日期：2024年5月29日 实验地点：一号馆 成绩：

1. **实验目的**
2. 基于Python语言和Flask框架，编写一个用户身份认证和权限管理程序，实现用户登陆登出、管理员及普通用户的身份认证及资源权限分级管理。
3. 通过此实验，进一步理解用户身份认证和系统角色授权、资源权限管理的机制，以此为基础，实现简单的基于用户身份分权限管理的应用程序。
4. 通过此实验，初步学会使用Flask，能够应用Flask框架构建简单的Web展示页面，编写用户登陆登出、系统分角色展示不同信息和资源授权等简单功能。

**二、实验内容**

**1. 环境安装**

下载并安装Python Web框架Flask，搭建基于本机的web服务。

**2. 自行设计和编制一个简单的用户身份认证及权限管理程序**。

1. 基础功能：用户登陆登出、根据用户id识别用户的管理员或普通用户身份，根据用户的身份设计不同系统角色，根据系统角色实现不同内容的页面展示。
2. 复杂功能，例如：a）系统功能设计，可包括除页面展示外的表格查看、文件下载、文件上传、个人信息修改或所有用户信息修改等功能；b) 将所设计实现的部分功能，分别对应至系统中的不同角色，实现角色访问控制；c) 将普通用户或管理员身份，也可加入其他用户身份，对应至系统中的角色，实现不同身份用户登陆后的不同功能操作。（可选，不强制要求；可实现部分要求，但应保证程序的完整性；有加分）

**三、实验环境**

分项列出主要实验环境：

1. 主机/操作系统：Win11
2. Python开发环境：PyCharm Community Edition 2023.3.4
3. Python版本：3.9
4. flask版本：3.0。3
5. flask-login版本：0.6.3
6. flask-sqlalchemy版本：3.1.1
7. flask-wtf版本：1.2.1

**四、实验步骤**

1. **安装第三方库。**

根据命令安装flask、flask-login、flask-sqlalchemy和flask-wtf库，安装命令如下

pip install flask, flask-login, flask-sqlalchemy, flask-wtf

1. **设计用户身份及对应系统角色，为不同角色设计对应的资源权限功能。**

创建用户信息数据库，其中添加role列，表示用户的角色，有普通用户“user”和管理员“admin”，管理员拥有查看全部用户信息、修改用户信息和删除用户的权限，数据库用户构建如下，详细权限设计见3

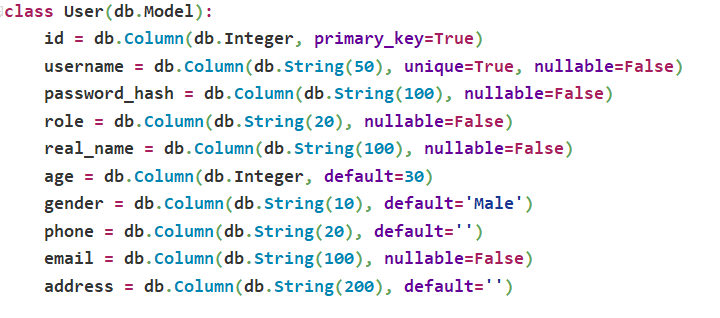


图 1 用户数据库构建

1. **搭建应用于Flask框架的本地web应用。**
2. 创建主页面

主页面包含登录和注册按钮，页面展示如下

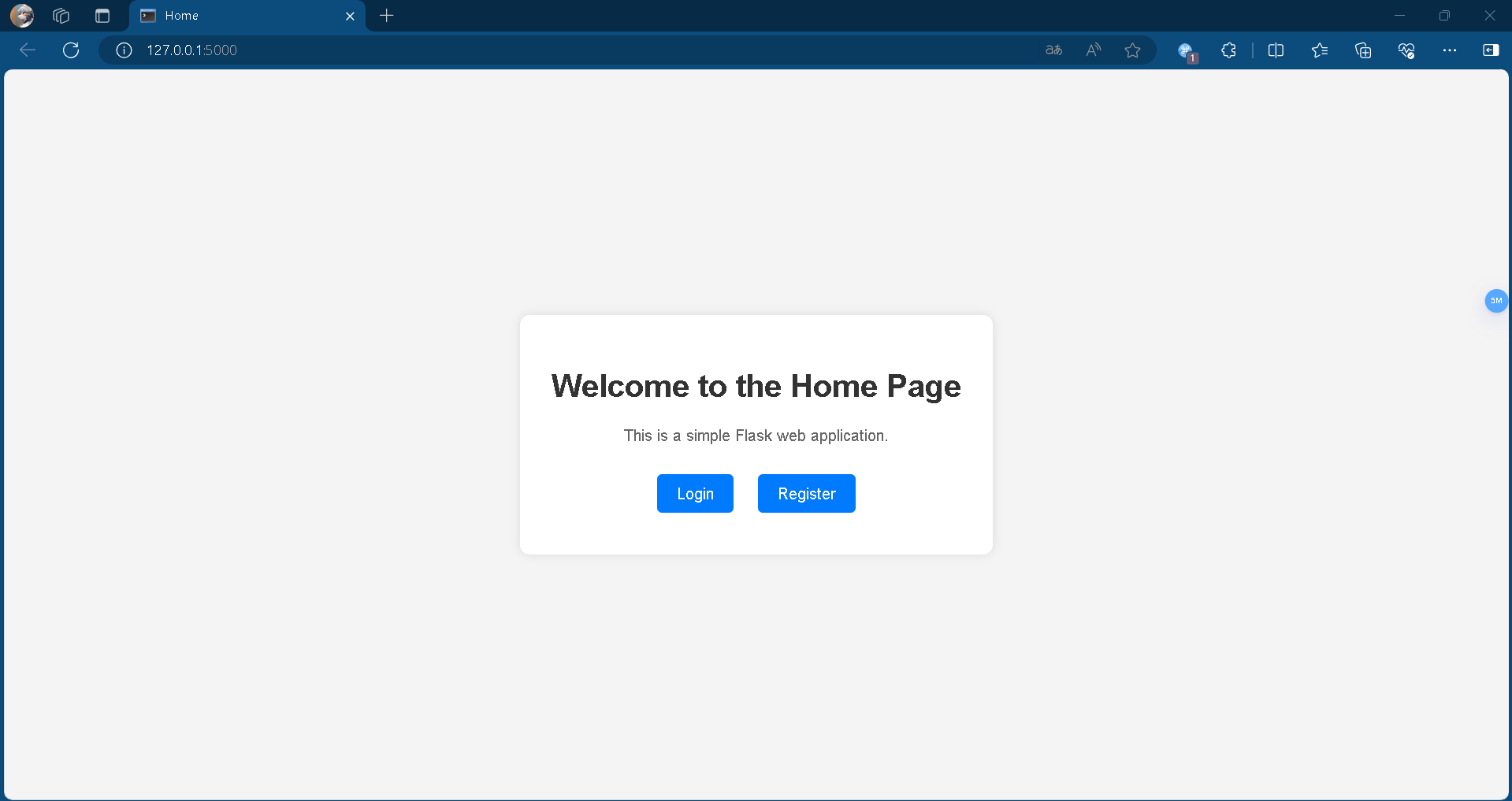


图 2 主页面

1. 注册

点击注册按钮前往注册，注册页面如下，用户填写完用户名、密码、确认密码、真实姓名、邮箱后即可完成注册，如果填写了正确的邀请码，就可以获得管理员身份，注册后跳转到登录页面

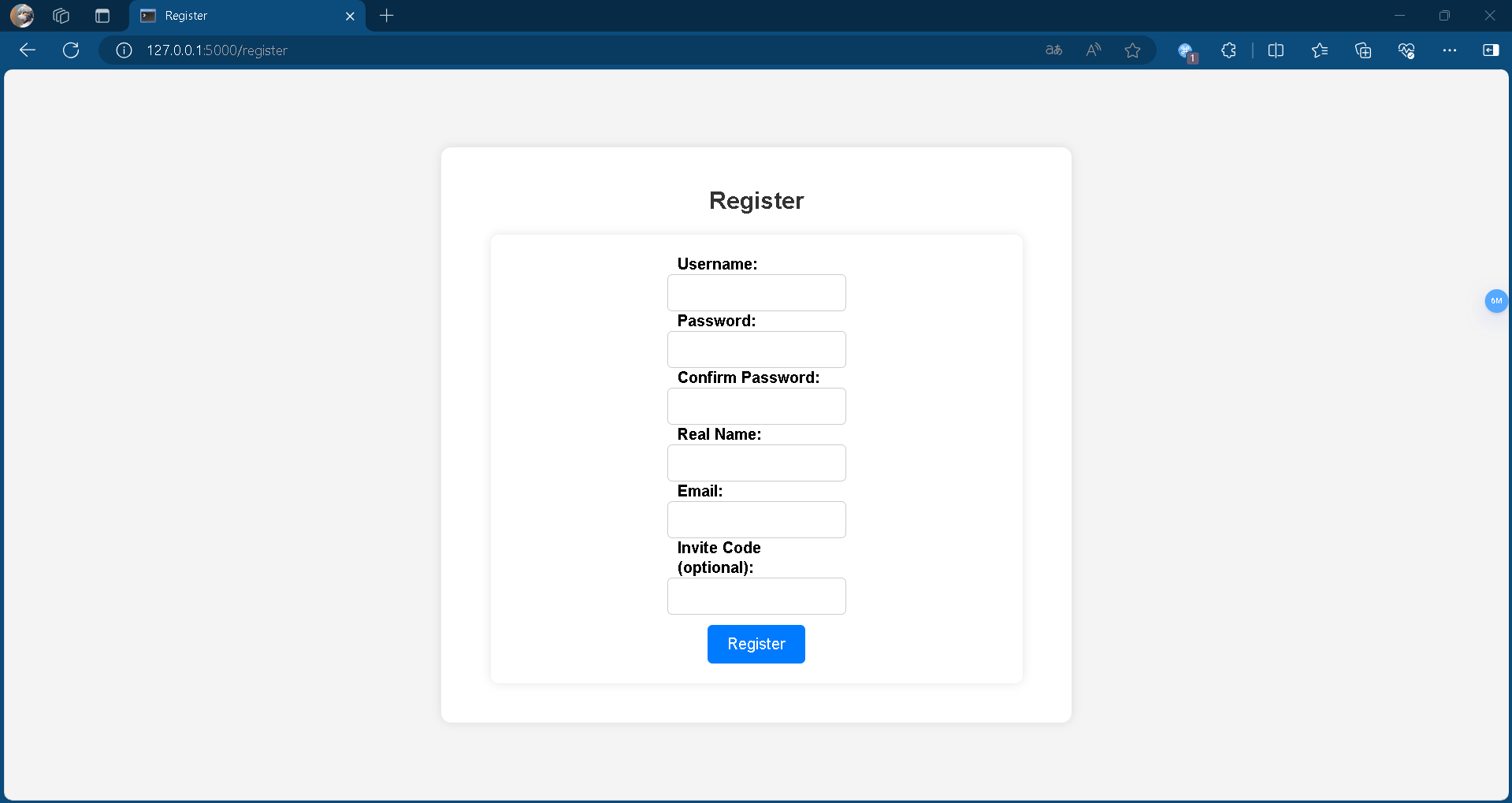


图 3 注册页面

实现代码如下



图 4 注册代码

1. 登录

登录页面如下

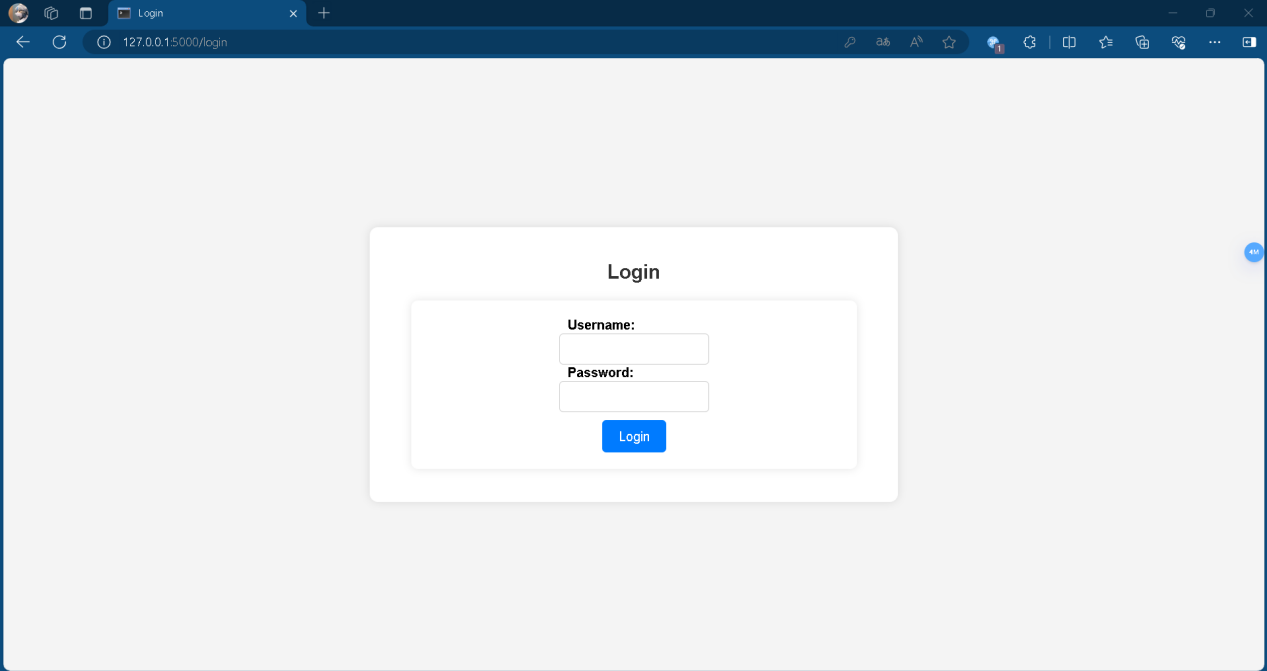


图 5 登录页面

输入正确的用户名和密码即可登录到个人信息页面，否则会提示用户名或密码错误

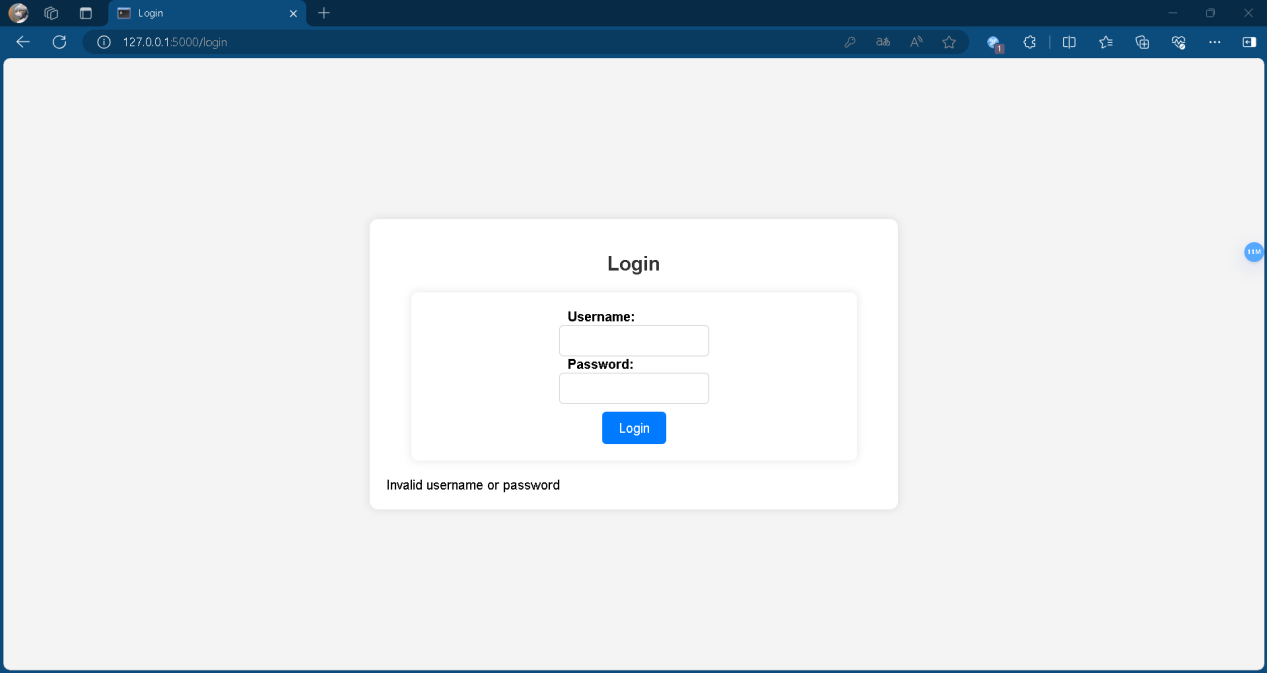


图 6 错误输入提示

实现代码如下



图 7 登录代码

1. 个人信息

登录成功后即可查看个人信息，普通用户个人信息页面如下

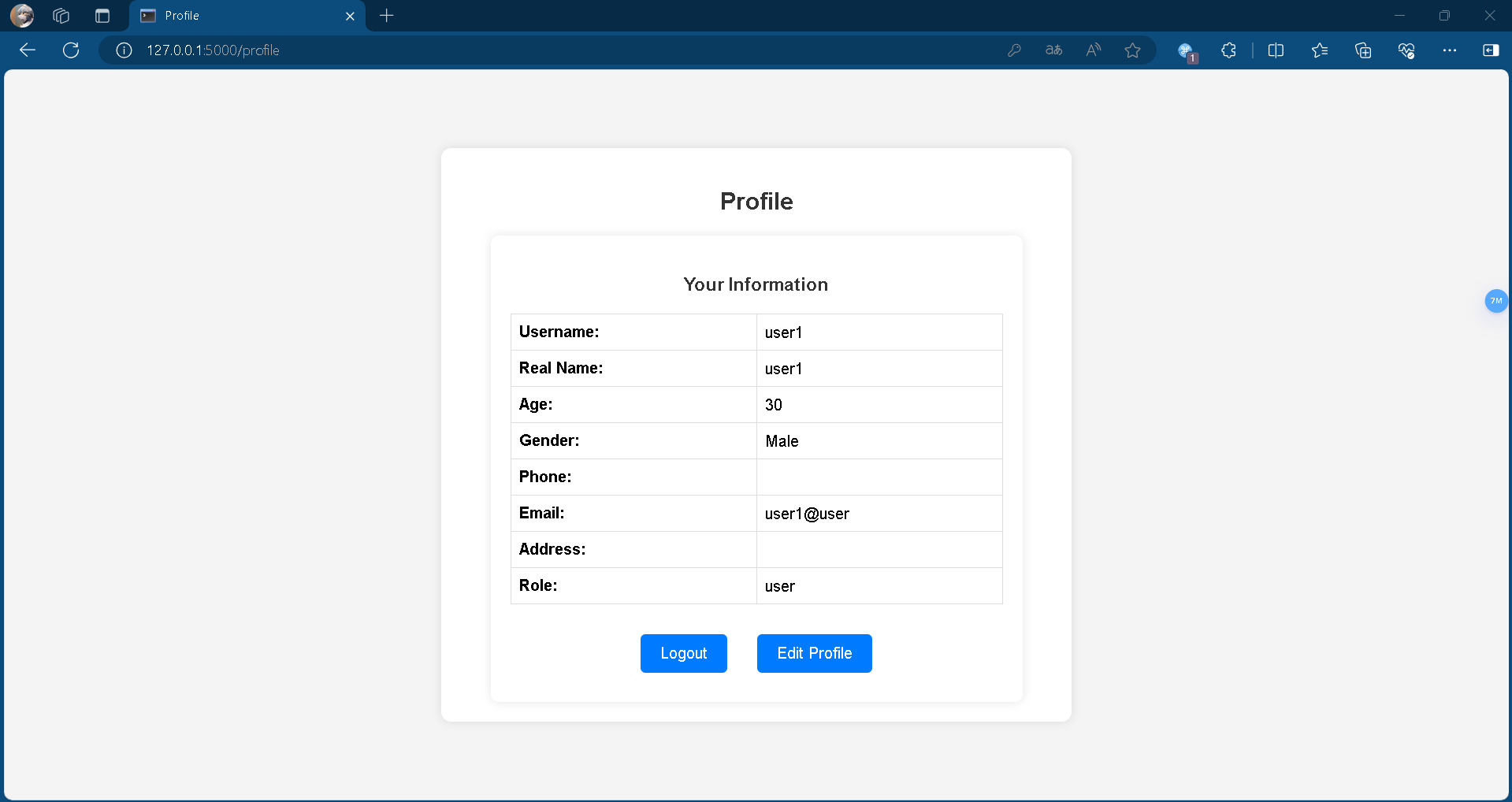


图 8 普通用户个人信息页面

管理员可以查看所有用户信息

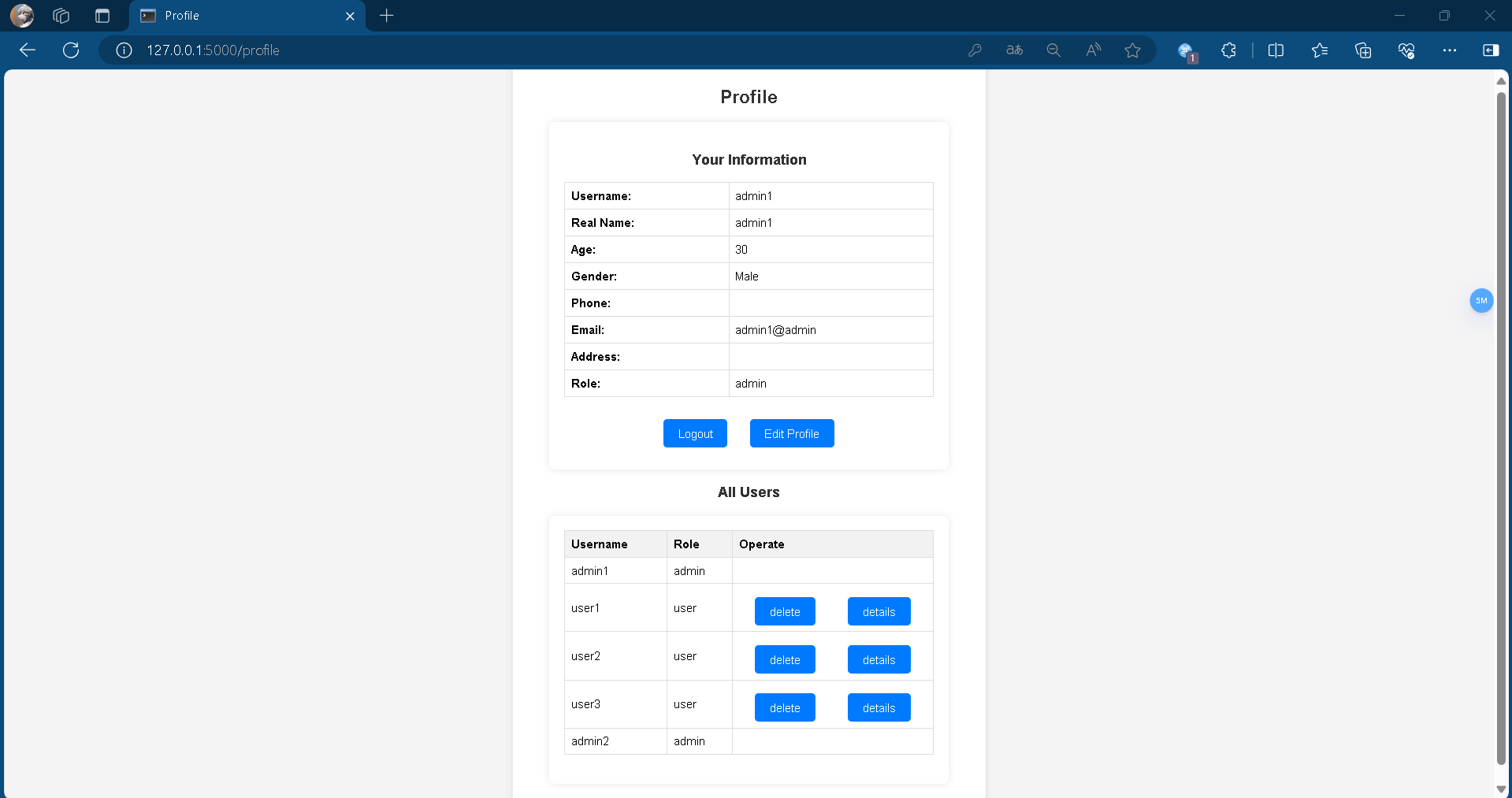


图 9 管理员个人信息页面

个人信息页面代码如下



图 10 个人信息页面代码

1. 登出

在给人信息页面点击登出按钮即可退出登录，返回至主页面，实现代码如下

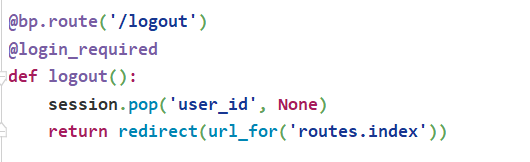


图 11 登出代码

1. 修改个人信息

在给人信息页面点击修改按钮即可跳转去修改个人信息，用户可以按需修改个人信息

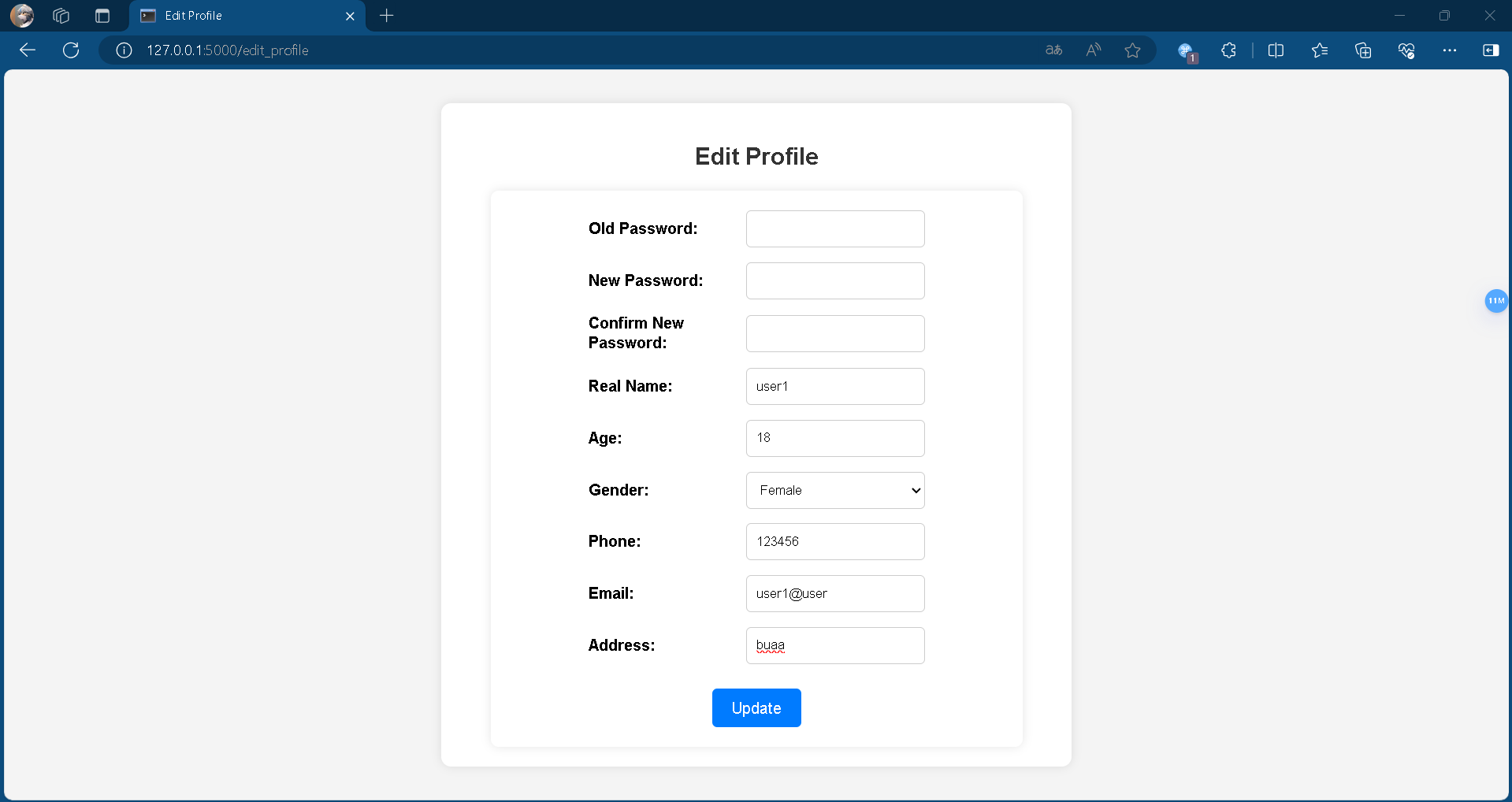


图 12 修改个人信息页面

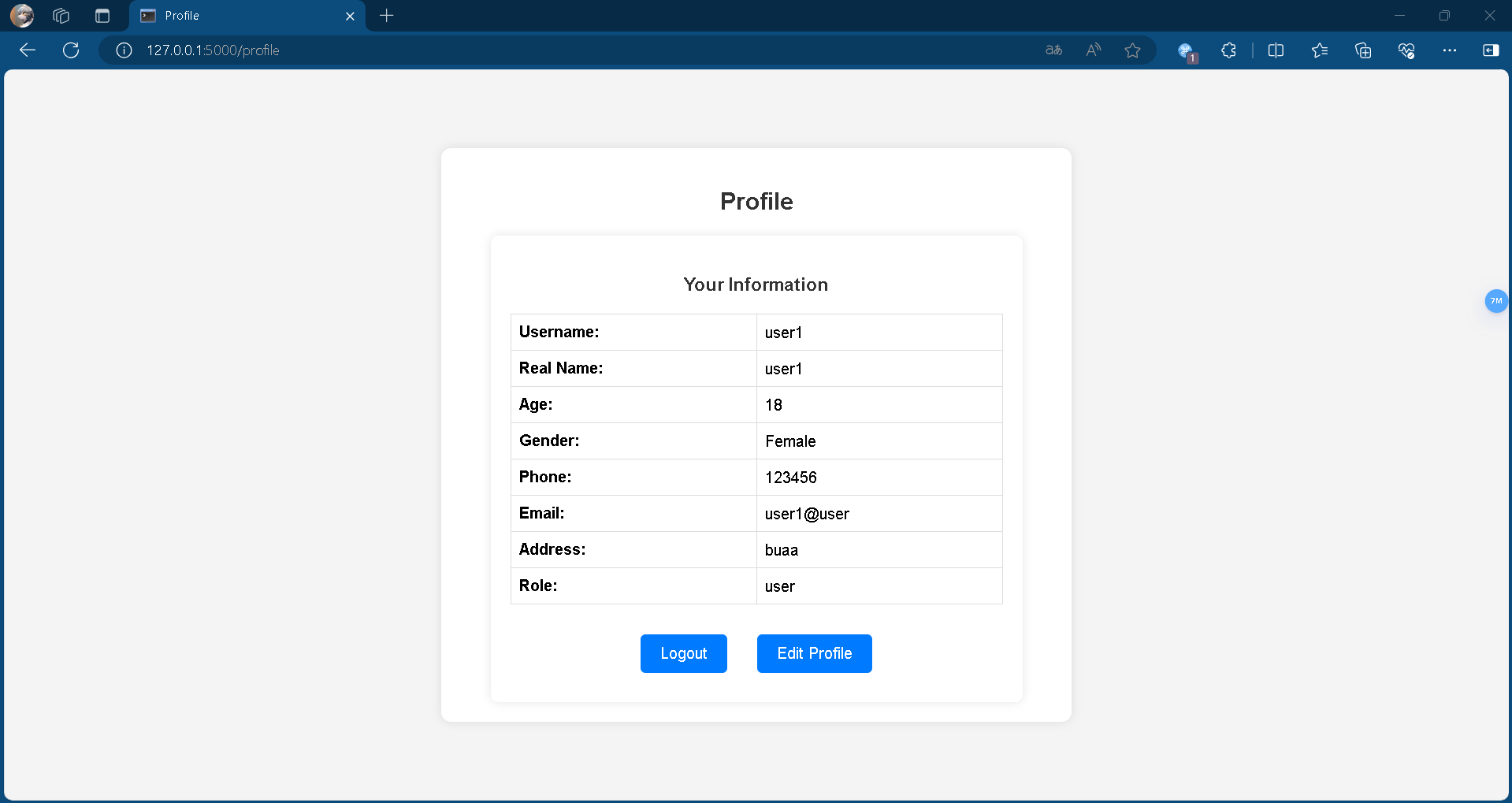


图 13 修改成功截图

修改个人信息代码如下





图 14 更新个人信息代码

1. 管理员权限

在管理员个人信息页面，管理员可以删除、查看和修改普通用户信息

例如删除用户1，点击删除后跳转至/delete\_user/user.id页面，提示删除成功

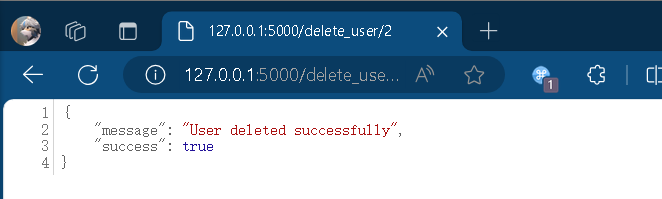


图 15 删除成功页面

再次返回管理员页面查看，发现用户1已被删除

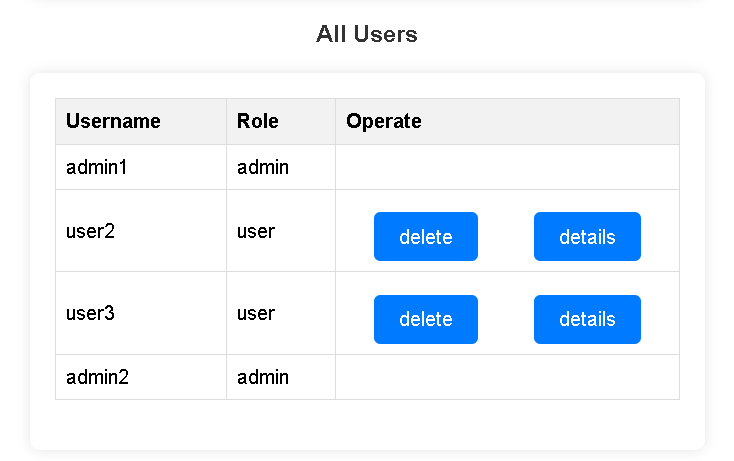


图 16 用户1被删除

删除用户代码如下



图 17 删除用户代码

点击详情按钮可以查看用户详细信息，注意到此时使用了GET请求，所以URL可以看到/view\_user/user.id

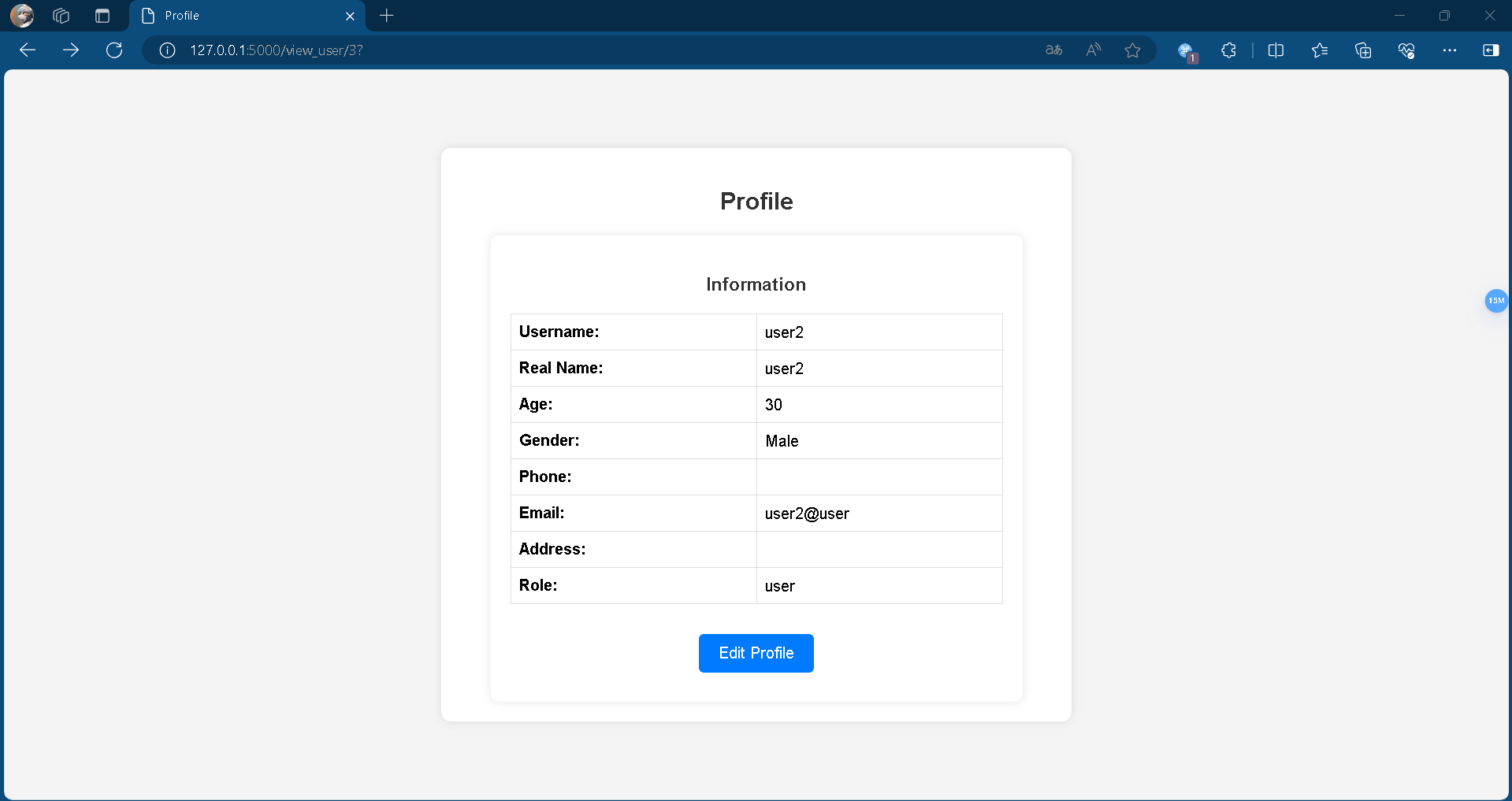


图 18 查看用户2结果

查看用户详情代码如下

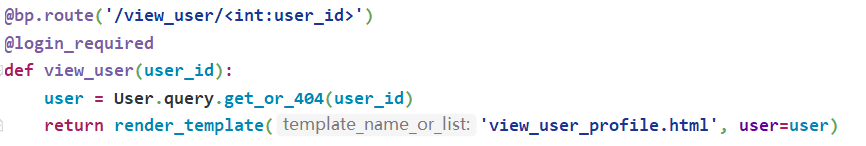


图 19 查看用户详情代码

点击修改信息按钮，即可跳转/edit\_user\_profile/user.id去修改用户2的信息

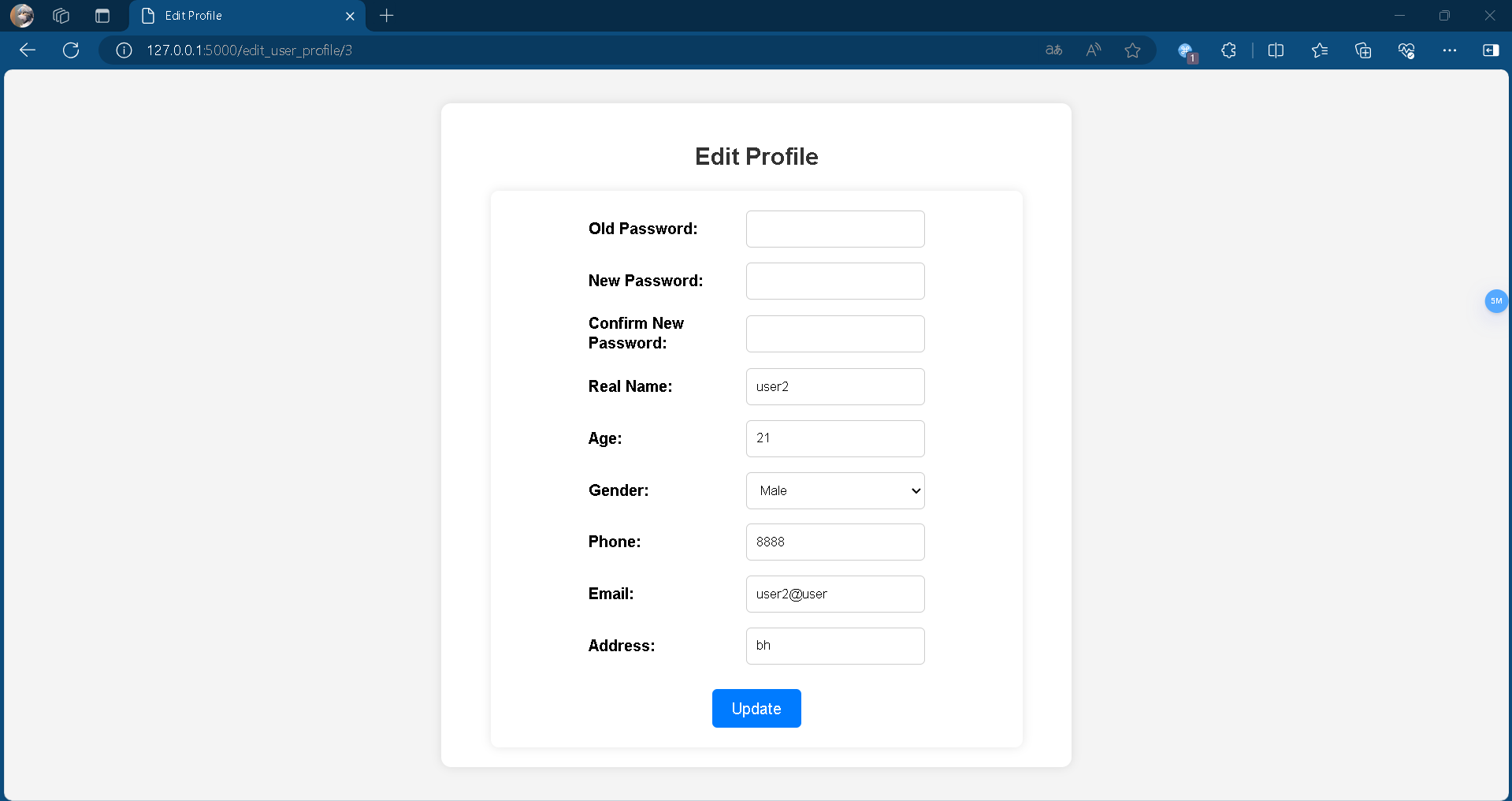


图 20 管理员修改用户信息

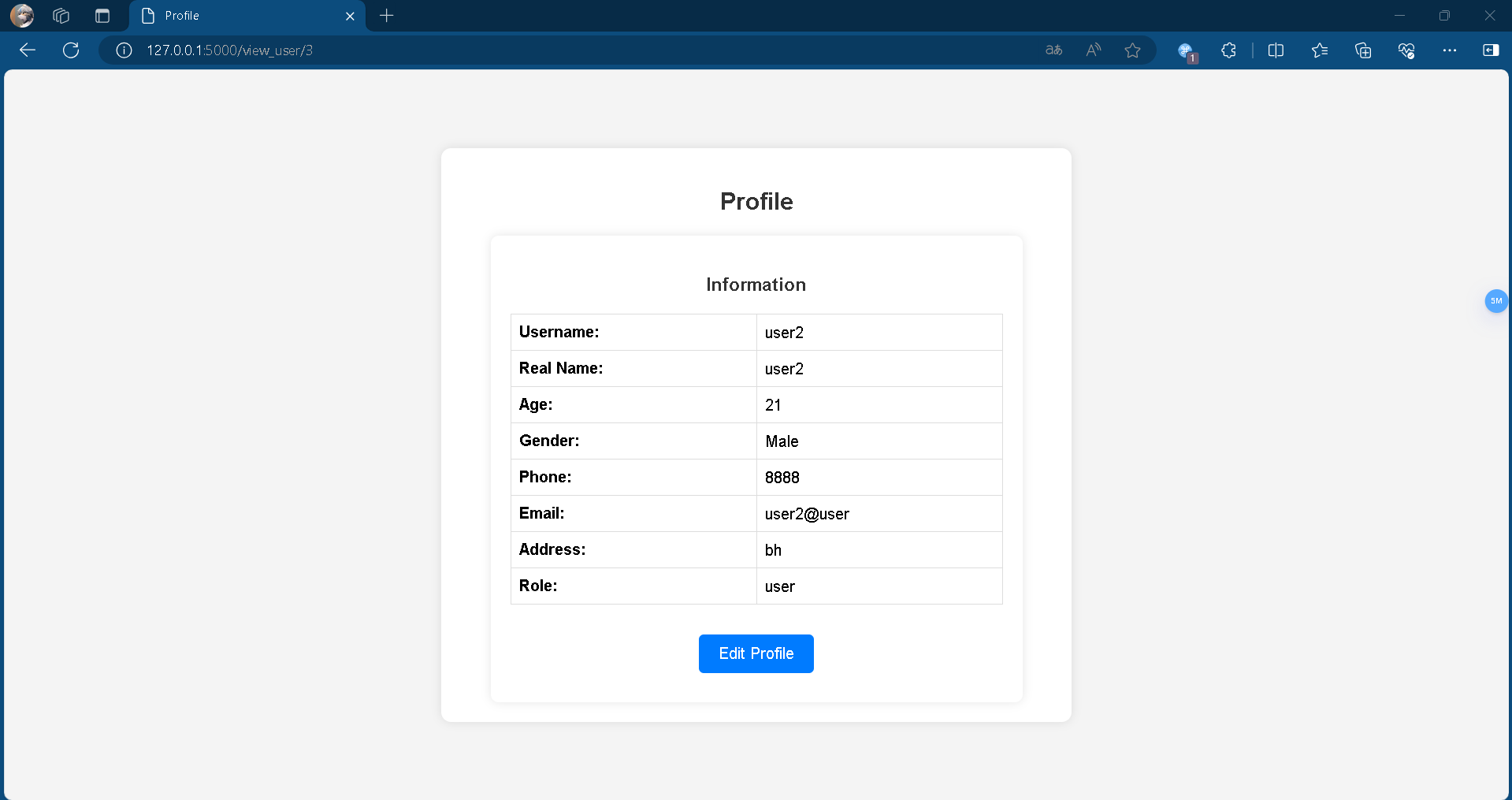


图 21 修改成功页面

管理员修改用户信息代码如下

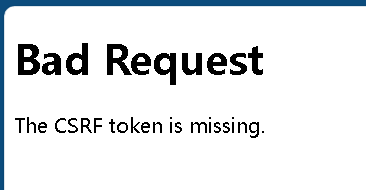




图 22 管理员修改用户信息代码

**五、问题和解决方法**

1. 程序使用了flask\_wtf模块的CSRFProtect来保护表单免受CSRF威胁，但出现了下面的问题



解决方法：

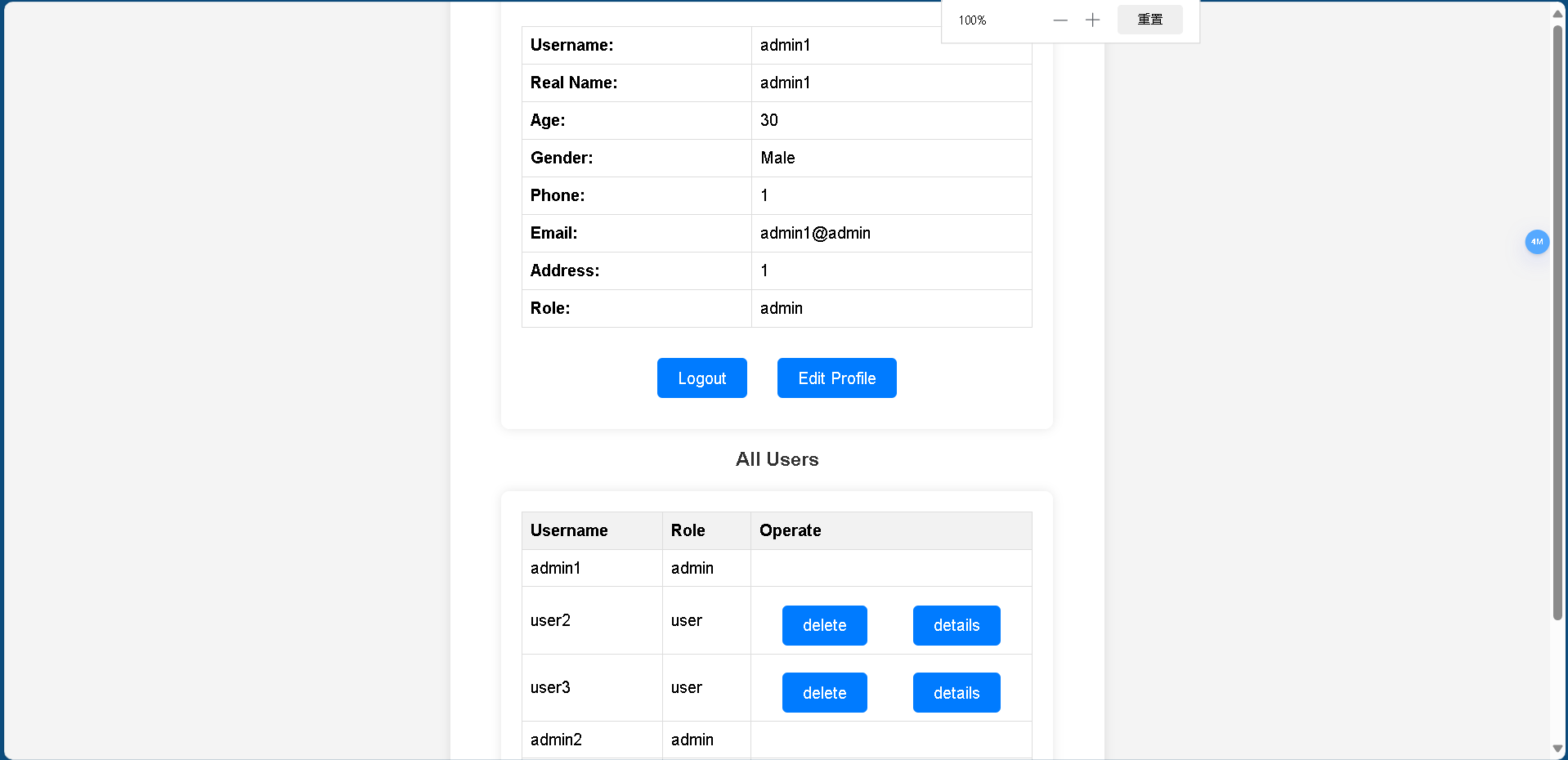
1. 设置默认禁用所有的视图CSRF保护。在config文件里设置

WTF\_CSRF\_CHECK\_DEFAULT = False

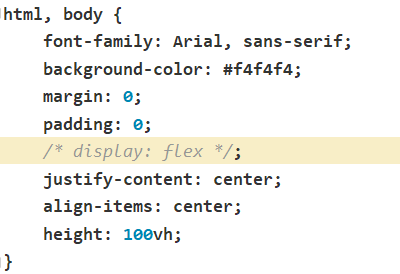
1. 在表单前添加csrf令牌

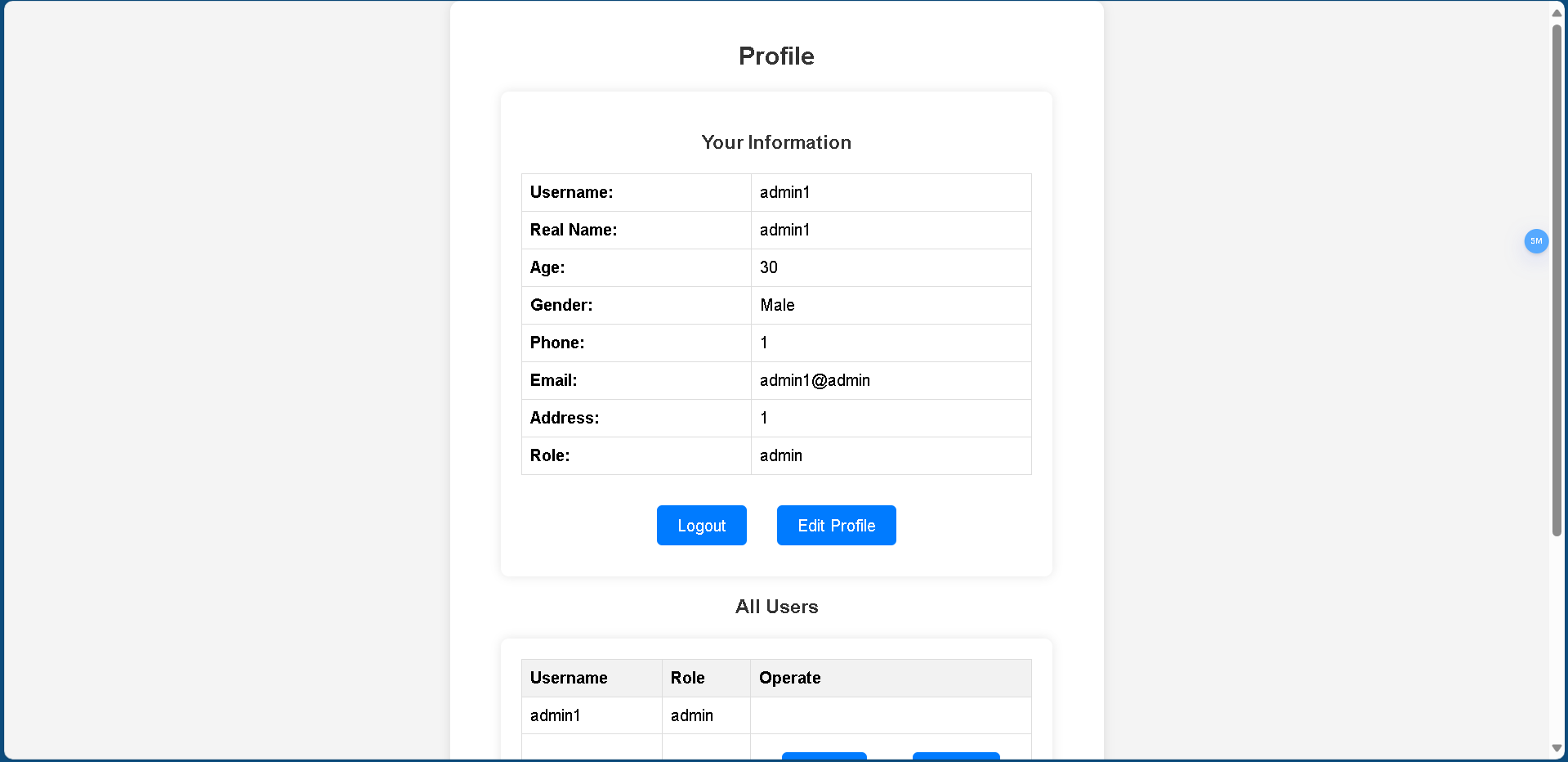
**<input type="hidden" name="csrf\_token" value="{{csrf\_token()}}">**

1. 当页面缩放为100%，即默认缩放时，顶部部分信息被遮挡，无法通过滑动查看



解决方法：检查时发现使用了flex布局，导致此问题，取消此布局后可以显示完全





**六、实验总结**

通过本次实验，我深入了解了Flask框架在Web开发中的应用，包括用户身份认证、权限管理、数据库操作、表单验证以及前端页面设计等方面的知识。特别是通过解决实验中遇到的问题，增强了我对Web开发中常见问题的处理能力和调试技巧。同时，这次实验也让我认识到，良好的前后端协作和布局设计对于提升用户体验的重要性。