

**《软件综合实践》报告**

**Report on Project of Course “Software Comprehensive Practice”**

组长：叶博文 学号：LX20220407005 手机号：15195959335

**实验课程：**软件综合实践（课程号0601125）

Software Comprehensive Practice（Course Code 0601125）

**主讲教师**：邓劲柏

**学 分 数：** 2 学分

**开设学院：**计算机与软件学院

**开设时间：**2023~2024学年第二学期

成绩Grade：\_\_\_\_\_\_\_\_\_\_\_\_\_

河 海 大 学

**目标**

[1 项目概述 1](#_Toc202214728)

[1.1 项目背景及目的 1](#_Toc202214729)

[1.2 开发环境 1](#_Toc202214730)

[2 需求分析 3](#_Toc202214731)

[2.1 用户认证与账户安全 3](#_Toc202214732)

[2.2 患者需求 4](#_Toc202214733)

[2.3 医生需求 4](#_Toc202214734)

[2.4 系统管理员需求 6](#_Toc202214735)

[2.5 通用界面与系统性能需求 6](#_Toc202214736)

[2.6 数据流图 7](#_Toc202214737)

[2.6.1 顶层数据流图 7](#_Toc202214738)

[2.6.2 一层数据流图 7](#_Toc202214739)

[2.7 数据字典 8](#_Toc202214740)

[2.7.1 users表 8](#_Toc202214741)

[2.7.2 doctor\_patient\_assignments表 8](#_Toc202214742)

[2.7.3 predictions表 9](#_Toc202214743)

[3 系统设计 10](#_Toc202214744)

[3.1 概要设计 10](#_Toc202214745)

[3.1.1 数据库概念设计 10](#_Toc202214746)

[3.1.2 数据库逻辑设计 11](#_Toc202214747)

[3.1.3 UI界面设计 12](#_Toc202214748)

[3.2 详细设计 18](#_Toc202214749)

[3.2.1 时序图 18](#_Toc202214750)

[3.2.2 类图 24](#_Toc202214751)

# 项目概述

## 项目背景及目的

癌症已成为当今世界对人类健康威胁最大的疾病之一，在中国的发病率和死亡率也逐年攀升，对社会和家庭造成了沉重的负担。临床实践反复证明，癌症的早期发现、早期诊断与早期治疗是提高患者生存率、改善预后效果的关键所在。然而，传统的癌症诊断流程复杂、周期长，且对医疗资源和医生经验有较高要求，难以满足大规模普查和早期筛查的需求。

近年来，随着人工智能与大数据技术的迅速发展，机器学习在医疗领域的应用展现出巨大的潜力。通过对大量的临床病例数据进行学习和分析，机器学习模型能够发现人类专家难以察觉的复杂模式，从而在疾病风险预测方面提供有价值的参考。在此背景下，开发一个智能化的癌症风险辅助预测系统，不仅能够为医生的临床决策提供数据支持，提高早期筛查的效率和准确性，也符合国家推动“人工智能+医疗”发展的战略方向，具有重要的现实意义与应用价值。

按照上面的背景，我决定开发一个癌症风险辅助预测系统。该系统采用了机器学习的技术，根据患者的身体情况信息，构建一个能够辅助医生进行早期的癌症风险预测的软件。

本报告详细地阐述一个癌症风险辅助预测系统设计与实现。利用机器学习技术，根据患者的各项生理及生化指标，构建一个能够辅助医生进行早期的癌症风险评估的软件。具体要求如下：

1. 需要实现用户登录系统。
2. 需要在用户端实现历史预测记录信息查询功能。
3. 需要在医生端实现输入患者的检测的身体情况信息并按照该信息进行预测功能。
4. 需要服务端管理员对用户进行管理（增删改查用户），并对医生注册请求进行实名验证。
5. 需要满足界面简洁，功能齐全，响应速度在一秒内。
6. 需要对用户的密码进行哈希转换。
7. 需要采用模块化的架构设计，将数据访问、用户认证和用户界面（UI）等核心功能进行分离，以提高代码的可维护性、可扩展性和可测试性。

## 开发环境

1. 开发语言: Python 3.10.11
2. Web框架：Streamlit 1.46.0

Streamlit 是一款开源的Python库，专为机器学习和数据科学项目创建美观、高性能的Web应用而设计。 它能够让开发者仅用Python脚本就快速构建出交互式的用户界面，无需编写任何前端代码（HTML, CSS, JavaScript）。 本项目选用Streamlit旨在快速开发出一个用户友好的、用于数据输入和结果展示的“单一用户工具”界面，即对于每一个访问应用的用户，Streamlit都会在服务器上为其创建一个独立的运行实例。这种架构允许开发者可以像编写简单的单人脚本工具一样进行开发，无需处理多用户并发的复杂性，从而极大地提高了开发效率，并能快速交付一个用户友好的、用于数据输入和结果展示的交互界面。

1. 数据库：SQLite 3.45.3

SQLite 是一款轻量级的、无服务器的、自包含的嵌入式SQL数据库引擎。 本项目使用SQLite来存储和管理用户、预测及关联数据，其优点在于无需独立安装和配置数据库服务器，极大地简化了开发和部署流程。

1. 关键依赖库：

* numpy 2.2.3 & pandas 2.2.3：用于高效地进行数据读取、处理和数值计算。
* matplotlib 3.10.0 & seaborn 0.13.2：用于生成数据可视化图表。
* scikit-learn 1.6.1：用于加载和运行预先训练好的机器学习模型。
* xgboost 3.0.2：用作机器学习分类器。
* imblearn 0.8.2： 用于处理不平衡数据集（如使用SMOTE）。

# 需求分析

## 用户认证与账户安全

系统的基础是为所有用户提供一个安全可靠的认证机制。

1. 用户注册

* 系统需要提供一个一级注册页面，允许患者和医生进行自服务注册。
* 该注册页面需要用户提供以下信息以创建账户，都不能为空 ：

1. 姓 (First Name) 和名 (Last Name)，将被后台合并为全名 (Full Name)
2. 用户名 (Username)
3. 身份证号 (ID Number)
4. 密码 (Password) 和确认密码 (Confirm Password)
5. 注册角色 (Sign Up As): 用户需选择注册为“患者(Patient)”或“医生(Doctor)”

* 后端将根据用户输入自动处理以下逻辑：

1. 检查输入的信息是否完整、密码和确认密码是否一致：如果不一致，显示“Password do not match.”的提示。
2. 密码哈希处理：用户提交的密码将通过SHA-256算法进行哈希计算，数据库中仅存储哈希值。
3. 状态分配：如果注册角色为“医生”，其初始账户状态将被设为“pending\_approval”，需等待管理员审核。如果角色为“患者”，其账户状态将被直接设为“active”。

* 管理员账户由系统后台直接创建，不通过一级注册页面。

1. 用户登录

* 所有用户（患者、医生、管理员）均通过统一的登录页面，使用其账户名和密码登录系统。

1. 密码安全

* 为确保账户安全，用户密码在存入数据库前，必须使用SHA-256的哈希算法进行单向加密处理。 数据库中严禁存储任何形式的明文密码。
* SHA-256 (Secure Hash Algorithm 256-bit) 是一种广泛使用的密码学哈希函数。它的核心作用是接收任意长度的输入数据（如用户密码），并将其转换为一个固定长度的、256位的唯一摘要（通常表示为64个十六进制字符）。该算法具有以下关键特性，使其成为密码存储的理想选择：

1. 单向性（不可逆）：从生成的哈希值反向推导出原始密码在计算上是不可行的。这意味着即使数据库被泄露，攻击者也无法直接获取用户的明文密码。
2. 确定性：相同的输入将永远产生相同的哈希输出，这使得系统可以通过比较哈希值来验证用户输入的密码是否正确。
3. 雪崩效应：输入的微小变化（如密码中一个字符的更改）会导致输出的哈希值发生巨大且不可预测的变化，这可以防止攻击者通过相似哈希值来猜测密码。
4. 忘记密码

* 系统需提供忘记密码功能，允许用户通过验证身份来重置密码。该流程分为两步：

1. 身份验证：用户首先需要输入其用户名（Username）和身份证号（ID Number）进行身份验证 。
2. 密码重置：如果提交的信息与系统中记录的数据匹配，系统将引导用户进入密码重置页面。在此页面，用户需要输入新密码（New Password）和确认新密码（Confirm New Password）。新密码在提交后将经过同样的SHA-256哈希处理后更新到数据库中。

* 如果第一步中提交的用户名和身份证号不匹配，系统将显示错误提示信息，并阻止用户进入密码重置步骤 。

## 患者需求

登录后，患者进入个人中心，其核心功能是信息查询。

1. 历史记录查询

患者可以查询自身的历史预测记录，包括预测时间、检查医生、预测风险等级及具体概率值。该历史预测记录是按照预测时间降序排序（从最新到最久的）。患者也可以按照检查医生姓名进行搜索记录。

1. 医患关系管理

患者可以搜索系统内的医生，并向指定的医生发送建立医患关系的请求。患者也可以按照医生姓名进行搜索。

## 医生需求

登录后，医生进入个人中心，拥有数据录入、预测和查看并处理患者发送的关联请求。

1. 账户激活

医生账户在注册后，必须经过系统管理员的后台验证或审批，其账户才能被激活并正常登录使用，以保障系统的安全性。

1. 预测功能

医生能够为与其关联的患者输入检测到的身体情况信息，并调用系统的核心预测功能。

输入检测到的身体情况包括：患者年龄，癌症分期（‘I’，‘II’,‘III’，‘IV’），肿瘤大小（以cm为单位），肿瘤类型（'Lung', 'Stomach', 'Cervical', 'Liver', 'Colorectal', 'Breast'），转移（’yes’或者’no’），医疗类型（'Radiation', 'Chemotherapy', 'Surgery', 'Targeted Therapy', 'Immunotherapy'）和合并症（'No Comorbidities', 'Diabetes, Hepatitis B', 'Hepatitis B', 'Hypertension', 'Diabetes, Hypertension', 'Diabetes, Hepatitis B', 'Hypertension, Hepatitis B'）。

输出预测结果为：预测结果类别（'High Risk'，'Medium Risk'，'Low Risk'），和预测结果风险概率。

为确保在临床环境中的流畅使用体验，预测操作的端到端延迟（从医生点击“预测”按钮到界面显示结果）应该小于 2秒。

该预测功能的核心是一个基于 XGBoost (eXtreme Gradient Boosting) 算法训练的机器学习分类模型。该模型的开发流程如下：

1. 数据来源与预处理: 模型基于Kaggle的 "China Cancer Patient Records" 数据集进行训练。在训练前，数据经过了严格的预处理，包括：对分类特征（如TumorType, TreatmentType，Comorbidities）进行独热编码（One-Hot Encoding），对有序特征（CancerStage，Metastasis，SurvivalStatus）进行顺序编码（Ordinal Encoding），对数值特征（Age, TumorSize）进行标准化处理（StandardScaler），以及按照特征相关性进行特征选择。这里，Metastasis和SurvivalStatus认为有序特征是为了保证数据的一致性，也对于数据分析过程不会发生混乱事情。
2. 处理数据不平衡: 为解决原始数据中类别不平衡的问题，采用了SMOTE（Synthetic Minority Over-sampling Technique）算法对训练数据进行过采样，以提高模型对少数类样本的识别能力。在医疗诊断等场景下，负样本（如“低风险”）的数量远超正样本（如“高风险”），这会导致模型在训练时倾向于预测多数类，从而忽略对少数关键病例的识别。SMOTE是一种先进的过采样技术，专门用于解决此问题，其工作原理并非简单地复制少数类样本，而是生成新的、“合成的”样本。具体步骤如下：

* 对于少数类中的每一个样本，算法会找出其在少数类中的k个最近邻居。
* 然后，从这k个邻居中随机选择一个。
* 最后，在原始样本和被选中的邻居之间的连线上，随机选择一点作为新生成的合成样本。

1. 模型训练与优化: 在多种备选模型（包括逻辑回归、支持向量机、随机森林等）的评估中，XGBClassifier 表现出最优的综合性能。XGBoost (eXtreme Gradient Boosting) 是一种基于梯度提升决策树（Gradient Boosting Decision Tree）的高级集成学习算法。其核心思想是迭代地构建一系列弱学习器（通常是决策树），每一棵新树都致力于修正前面所有树的预测残差（即错误）。XGBoost通过引入正则化项（L1和L2）来控制模型复杂度，有效防止了过拟合，并通过并行处理等优化技术大幅提升了训练速度，使其在处理表格型数据时表现尤为出色，非常适合本项目的医疗预测任务。

为了进一步提升模型表现，使用了网格搜索交叉验证（GridSearchCV）对XGBoost模型的超参数进行了优化。网格搜索交叉验证（GridSearchCV）是一种系统性的超参数优化技术。其工作方式是：首先，开发者为模型的关键超参数（如XGBoost的 n\_estimators 树的数量、max\_depth 树的最大深度、learning\_rate 学习率等）定义一个候选值网格。然后，GridSearchCV会遍历这个网格中所有可能的超参数组合。对于每一种组合，它都采用k折交叉验证的方法来训练和评估模型，从而得到一个稳健的性能分数。最终，该技术能够自动找出在交叉验证中表现最佳的超参数组合，以构建出最优的预测模型。

1. 模型持久化: 最终优化后的XGBoost模型，连同所有必需的预处理工具（如编码器和缩放器），都通过Python的pickle库被序列化并保存为 .pkl 文件。Streamlit应用程序在运行时会加载这些文件，以确保对医生输入的新数据应用完全相同的预处理流程，并使用训练好的模型进行实时预测。
2. 患者信息查询

能够查询特定患者的历史预测记录。系统将以列表和可视化图表（例如，以时间为X轴、预测概率为Y轴的折线图）的形式展示这些记录，使医生能够直观地观察患者癌症风险随时间的变化趋势。

1. 患者关联请求查询并处理

医生能够查看并处理患者发送的关联请求，可选择批准或拒绝。医生也可以通过患者姓名进行搜索，以方便医生快速地找到该患者。

## 系统管理员需求

系统管理员拥有系统的最高管理权限，包括：对所有用户进行管理（具有增删改查操作）和对医生的注册请求进行批准或拒绝。

1. 用户管理

管理员拥有对系统内所有账户（包括患者、医生和其他管理员）进行增、删、改、查（CRUD）的最高权限。

1. 医生注册审核

管理员审核新注册的医生账户，验证其身份后，决定批准或拒绝其注册请求。

## 通用界面与系统性能需求

1. 页面加载性能

为保证流畅体验，各主要功能页面的加载时间应在常规网络环境下不超过1秒。

1. 界面设计

所有界面需保持简洁直观，功能齐全，无不必要的装饰元素，确保信息传达的清晰性。

1. 数据输入校验

为保证模型的预测质量和系统的可靠性，当用户在前端界面输入数据时，系统将进行前后端双重验证，以防止无效数据（如超出范围的年龄）影响预测结果。

## 数据流图

### 顶层数据流图



图 1 顶层数据流图

### 一层数据流图

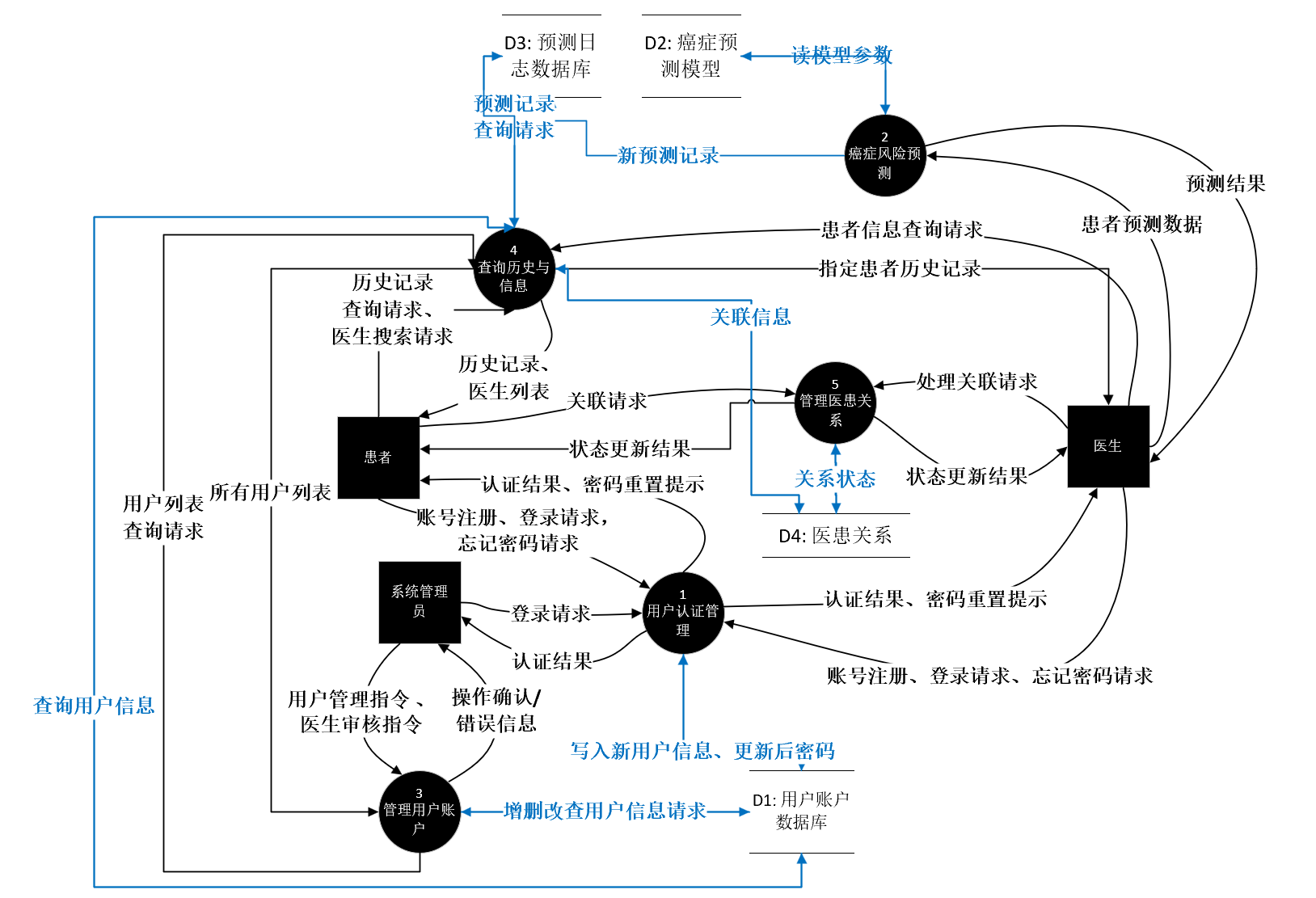


图 2 一层数据流图

## 数据字典

### users表

|  |  |  |  |
| --- | --- | --- | --- |
| **字段标识** | **字段类型** | **约束** | **字段名称** |
| user\_id | INT | PRIMARY KEY, AUTOINCREMENT | 用户唯一标识 |
| username | VARCHAR(20) | NOT NULL, UNIQUE | 用户名 |
| password\_hash | VARCHAR(128) | NOT NULL | 加密后的密码 |
| full\_name | VARCHAR(30) | NOT NULL | 用户真实姓名 |
| role | VARCHAR(10) | NOT NULL | 用户角色 ('patient', 'doctor', 'admin') |
| status | VARCHAR(20) | NOT NULL | 账户状态 ('active', 'pending\_approval') |
| id\_no | VARCHAR(18) | NOT NULL, UNIQUE | 身份证号 |
| dob | DATETIME | NOT NULL | 出生日期 |

表 1 用户账户表 users

### doctor\_patient\_assignments表

|  |  |  |  |
| --- | --- | --- | --- |
| **字段标识** | **字段类型** | **约束** | **字段名称** |
| assignment\_id | INT | PRIMARY KEY, AUTOINCREMENT | 关系唯一ID |
| doctor\_id | INT | FOREIGN KEY，NOT NULL | 医生ID（从users.user\_id） |
| patient\_id | INT | FOREIGN KEY, NOT NULL | 患者ID（从users.user\_id） |
| status | VARCHAR(20) | NOT NULL | 关系状态 ('requested', 'active') |

表 2 医患关系表 doctor\_patient\_assignment

### predictions表

|  |  |  |  |
| --- | --- | --- | --- |
| **字段标识** | **字段类型** | **约束** | **字段名称** |
| prediction\_id | INT | PRIMARY KEY, AUTOINCREMENT | 预测记录唯一标识 |
| doctor\_id | INT | FOREIGN KEY，NOT NULL | 执行预测的医生ID |
| patient\_id | INT | FOREIGN KEY, NOT NULL | 接受预测的患者ID |
| prediction\_timestamp | DATETIME | NOT NULL | 预测执行时间 |
| age | INT | NOT NULL | 患者年龄 |
| cancer\_stage | VARCHAR(5) | NOT NULL | 癌症分期 (‘I’,’II’,  ’III’,’IV’) |
| tumor\_size | REAL | NOT NULL | 肿瘤大小（单位：cm） |
| tumor\_type | VARCHAR(20) | NOT NULL | 肿瘤类型 |
| metastasis | VARCHAR(5) | NOT NULL | 转移 |
| treatment\_type | VARCHAR(20) | NOT NULL | 医疗类型 |
| comorbidities | VARCHAR(15) | NOT NULL | 合并症 |
| prediction\_class | VARCHAR(15) | 无 | 预测结果类别 |
| prediction\_probability | REAL | NOT NULL | 预测结果的概率 |

表 3 预测日志表 predictions

# 系统设计

## 概要设计

### 数据库概念设计

#### ER-图



图 3 E-R图

#### 数据库表详细说明

1. users表

users表包含的属性：用户唯一标识user\_id（主键），用户名username，加密后的密码password\_hash，用户真实姓名full\_name，用户角色role，账户状态status，身份证号码id\_no和出生日期dob。

1. doctor\_patient\_assignments表

doctor\_patient\_assignments表包含的属性：关系唯一标识assignment\_id（主键），医生编号doctor\_id（外键），患者编号patient\_id（外键）和关系状态status。其中，doctor\_id和patient\_id均为外键，分别引用users表中的user\_id，用于建立医患之间的关联。

1. predictions表

predictions表包含的属性：预测编号prediction\_id（主键），医生编号doctor\_id（外键），患者编号patient\_id（外键），预测执行时间prediction\_timestamp，患者年龄age，癌症分期cancer\_stage，肿瘤大小tumor\_size，肿瘤类型tumor\_type，转移metastasis，合并症comorbidities，预测结果的类别prediction\_class，预测结果的概率prediction\_probability（风险概率）。其中，患者年龄是从users表中的dob属性和predictions表中的prediction\_timestamp计算出来的。

其中，users表与doctor\_patient\_assignments表具有多对多关系，即一个患者可以与多个医生关联且一个医生可以与多个患者关联。而，users表与predictions表具有一对多关系，即一个医生可以进行多次预测且一个患者可以具有多个预测结果。

### 数据库逻辑设计

#### 关系模式

1. users表（user\_id, username，password\_hash，full\_name，role，status，id\_no, dob）
2. doctor\_patient\_assignment表（assignment\_id，doctor\_id，patient\_id，status）
3. predictions表（prediction\_id，doctor\_id，patient\_id，prediction\_timestamp, age, cancer\_stage，tumor\_size，tumor\_type，metastasis，comorbidities，prediction\_class，prediction\_probability）

#### 系统功能结构图

我们将系统从功能的角度对系统进行规划。顶层是我们所要实现的癌症风险辅助预测系统。该系统从功能上可以划分成四个模块：用户认证与管理模块、医生功能模块、患者功能模块和管理员功能模块：

1. 用户认证与管理模块可以再次划分成几个功能块：用户注册、用户登录和重置密码管理。
2. 医生功能模块可以再次划分成医患之间关系的管理、患者历史信息记录的查询和患者癌症风险的预测功能。
3. 患者功能模块包括两部分功能块：个人历史记录信息的查询和医患之间关系的管理。、

管理员功能模块也有两部分功能块：对医生注册时候所提供的实名认证的验证功能和对所有用户管理的功能。

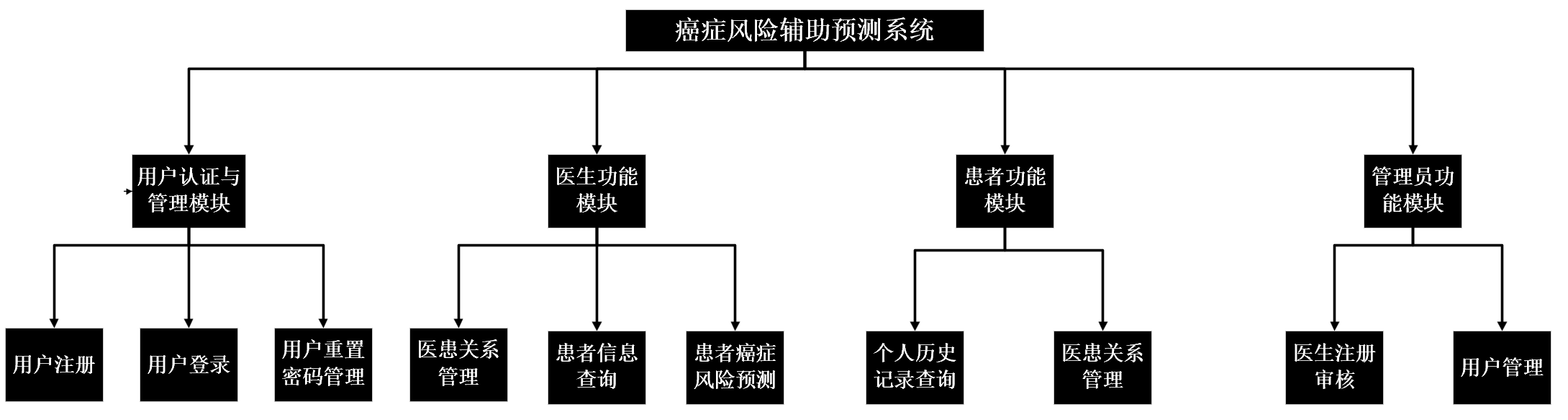


图 4 系统功能结构图

### UI界面设计

#### 登录界面

用户登录操作步骤：

1. 在“Username”输入框中输入您的账号。
2. 在“\*\*\*\*\*\*\*”输入框中输入您的密码。
3. 点击“SIGN IN”按钮以进入系统。如果账号或密码错误，将提示错误信息。
4. 如果您没有账号，可以点击“SIGN UP”按钮进行注册。
5. 如果忘记密码并需要修改密码，可以点击“Forgot password？”按钮进入修改密码页面。

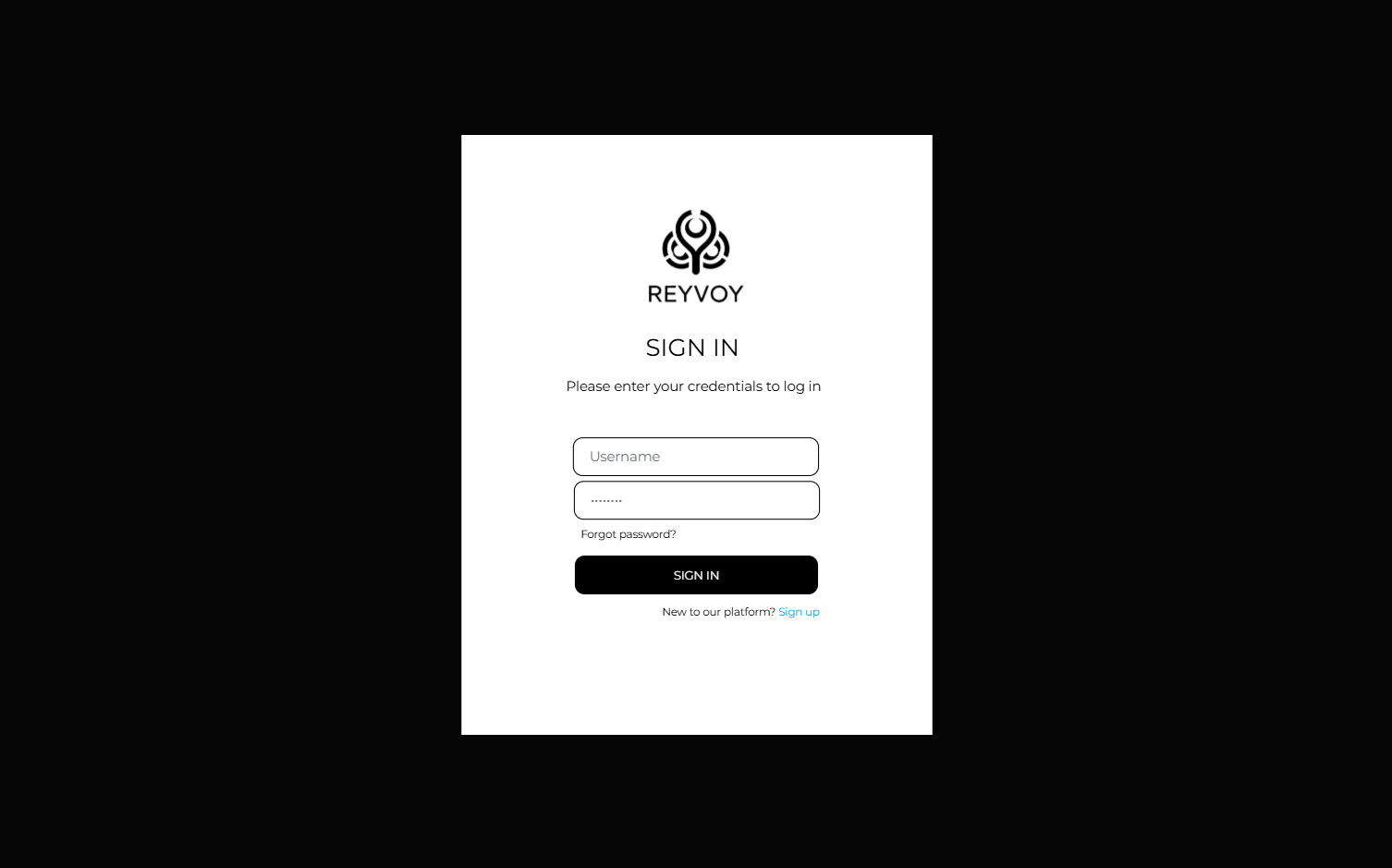


图 5 用户登录界面

#### 注册界面

用户注册操作步骤：

1. 在“First Name”和“Last Name”输入框中分别输入您的姓和名。
2. 在“Username” 输入框中输入您想要的用户名。
3. 在“ID No.” 输入框中输入您的身份证号码。
4. 在“Sign up as” 输入框中选择您注册为“Doctor/Patient”（注册为Doctor的时候需要被管理员验证后，账户才能激活）。
5. 在“Birth Date”输入框中输入您的出生日期。
6. 在“Password”输入框中输入您的密码。
7. 在“Confirm Password”输入框中再次输入您的密码，以确保两次输入的密码一致。
8. 完成输入后，系统将会进行验证，确保账号和密码符合要求并且两次输入的密码一致。
9. 如果想返回登录页面，点击“SIGN IN”按钮。

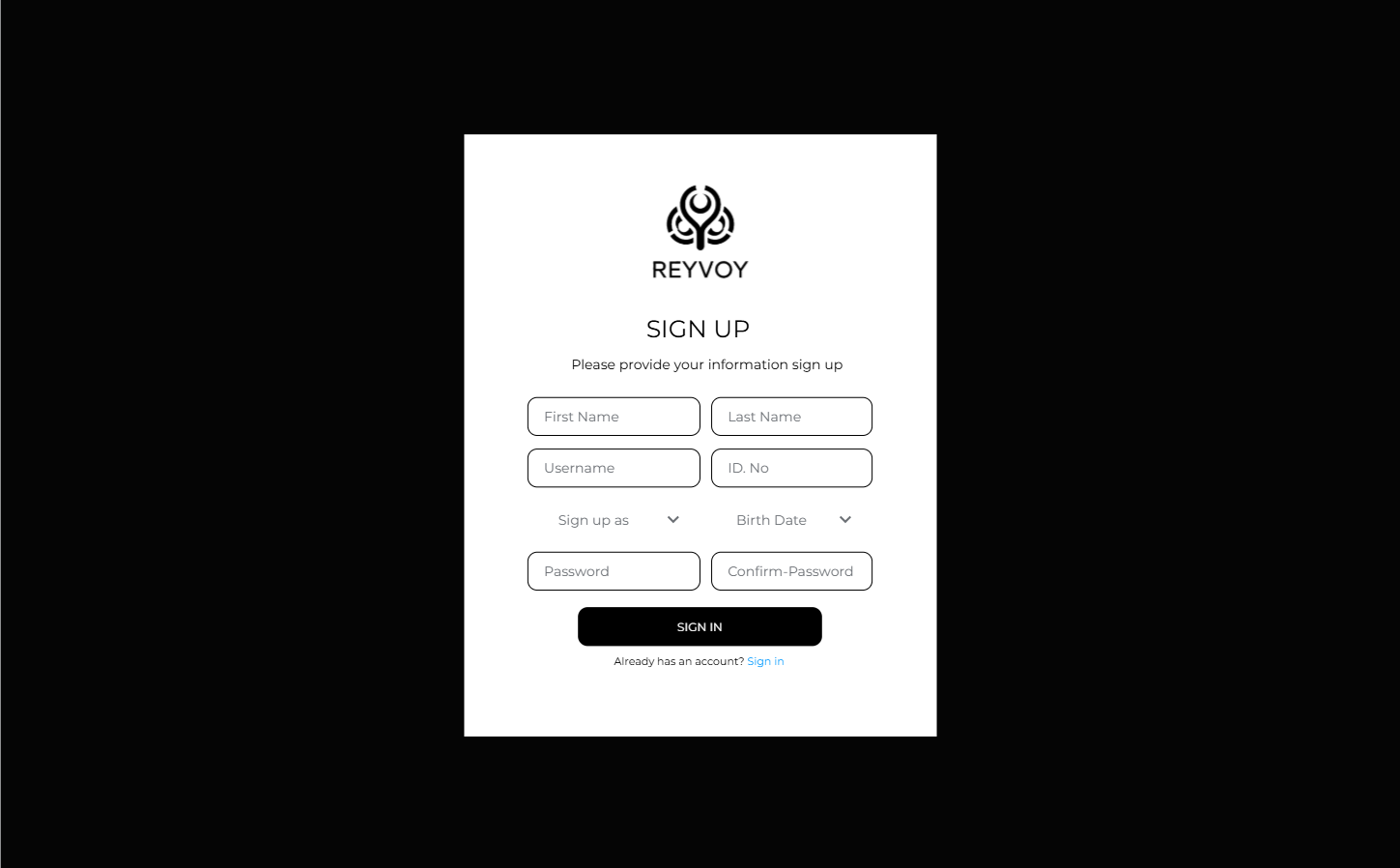


图 6 用户注册界面

#### 用户忘记密码界面

用户忘记密码操作步骤：

1. 在“Username”输入框中输入您的用户名。如您忘记用户名，请联系人工客服协助处理。
2. 在“ID No.” 输入框中输入您的身份证号码。
3. 点击“RESET PASSWORD”按钮让系统进行验证。
4. 验证通过后，进入了修改密码页面。
5. 在“New Password”输入框中输入您的新密码。
6. 在“Confirm Password”输入框中再次输入您的密码。
7. 点击“RESET PASSWORD”按钮 让系统进行验证。

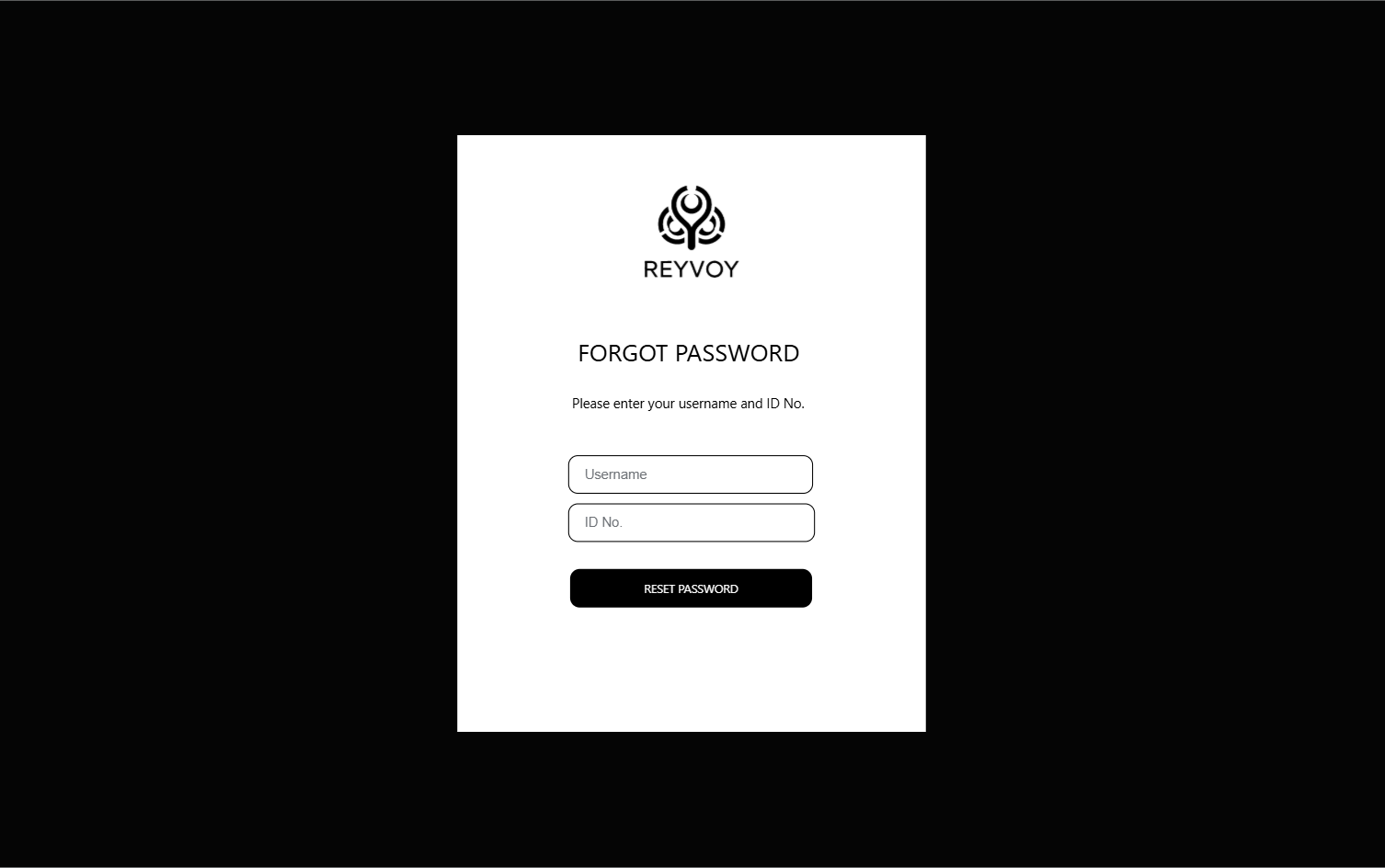


图 7 忘记密码界面1

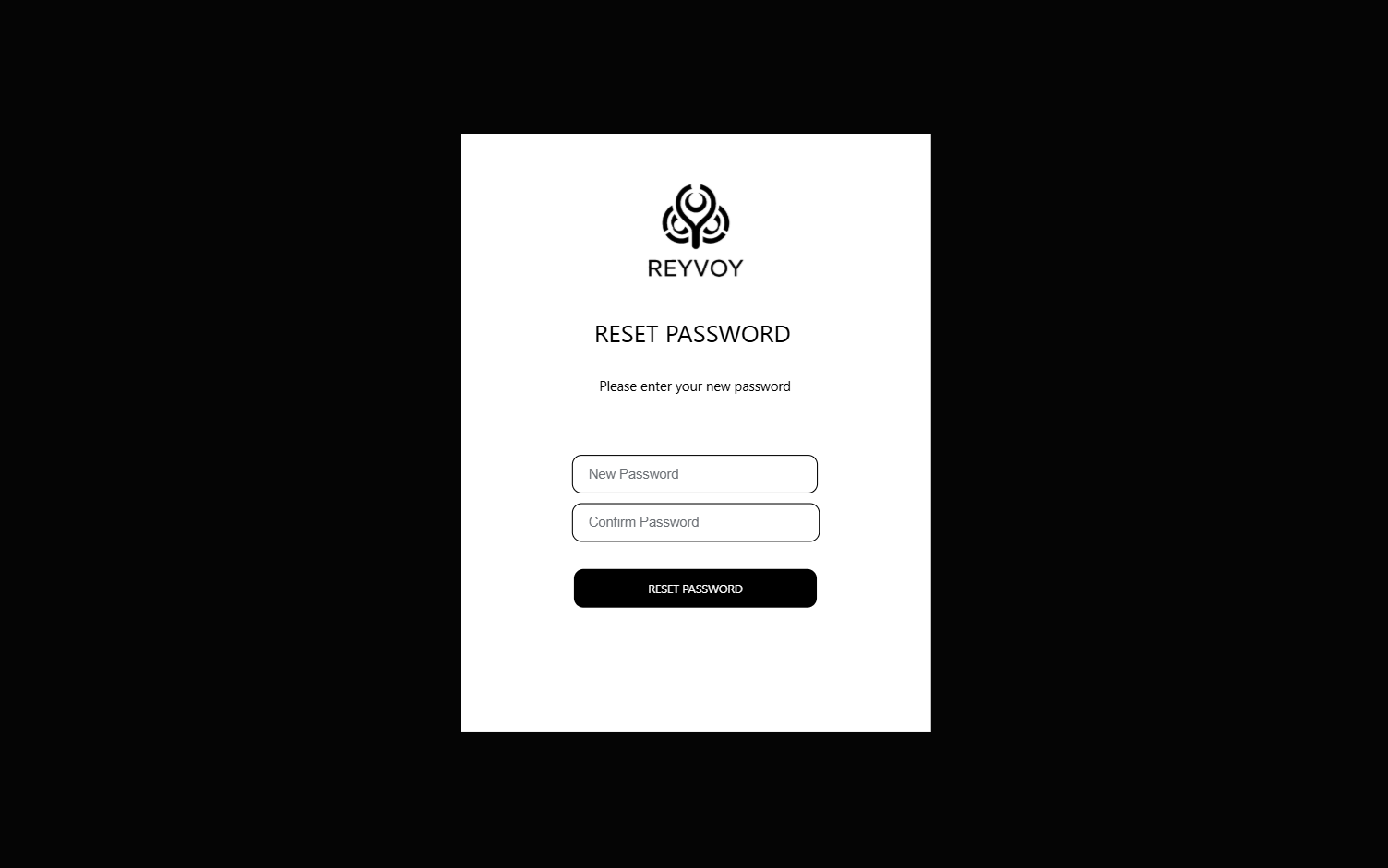


图 8 忘记密码界面2

#### 系统管理员界面

管理员界面操作步骤：

1. 如果想要添加用户，点击“Add User”按钮进行添加用户。
2. 如果想要按照用户名查找用户，通过“Search by Username”中。
3. 如果想要修改用户信息，点击修改图标。
4. 如果想要删除用户信息，点击删除图标。
5. 对于Status为“Pending Approval”的医生用户，管理员可以通过Doctor Approval页面进行验证该用户的身份，然后判断是否批准该医生的注册请求。
6. 最后，如果想要退出，点击左下角中的“Log Out”按钮。

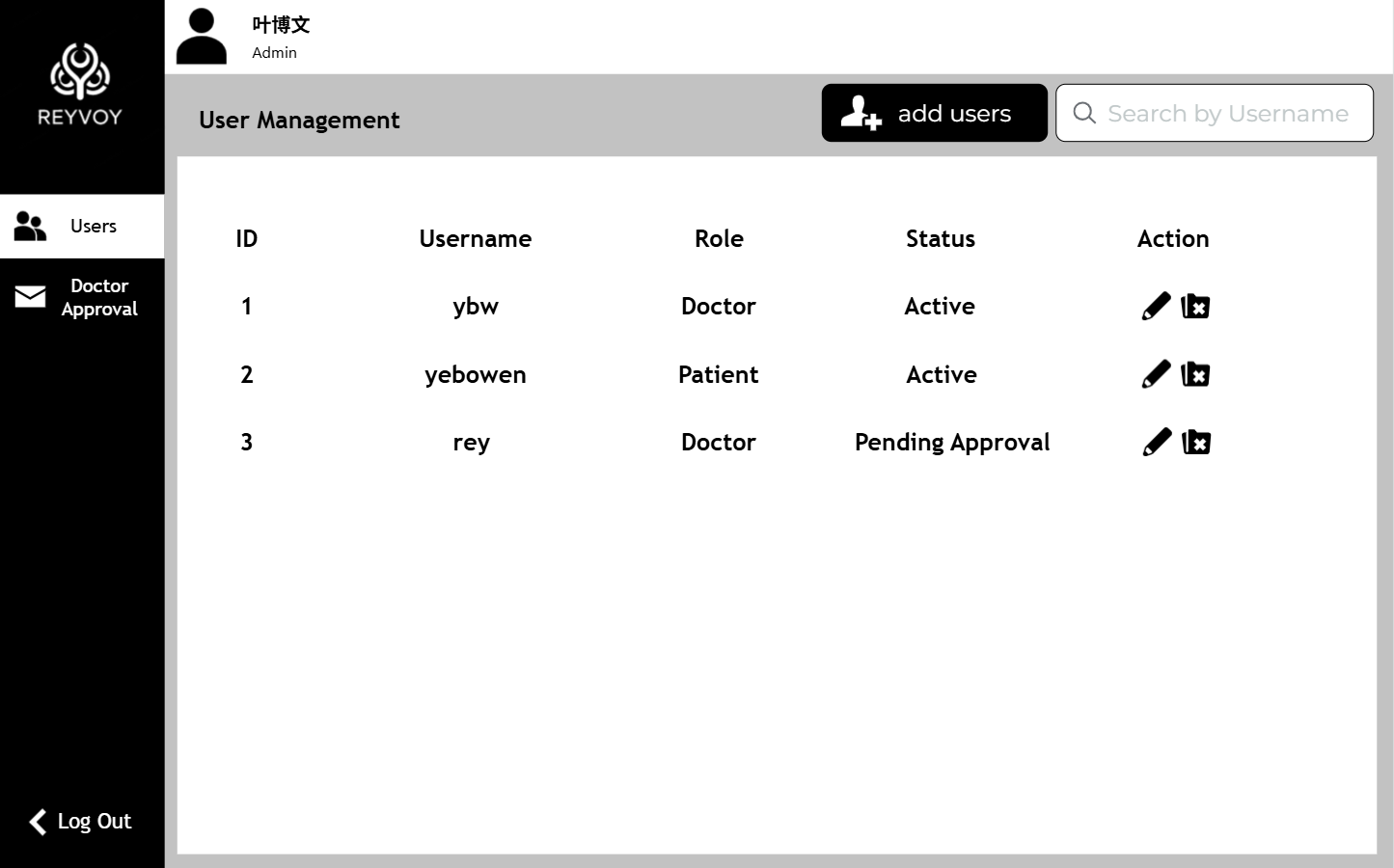


图 9 管理员界面 1

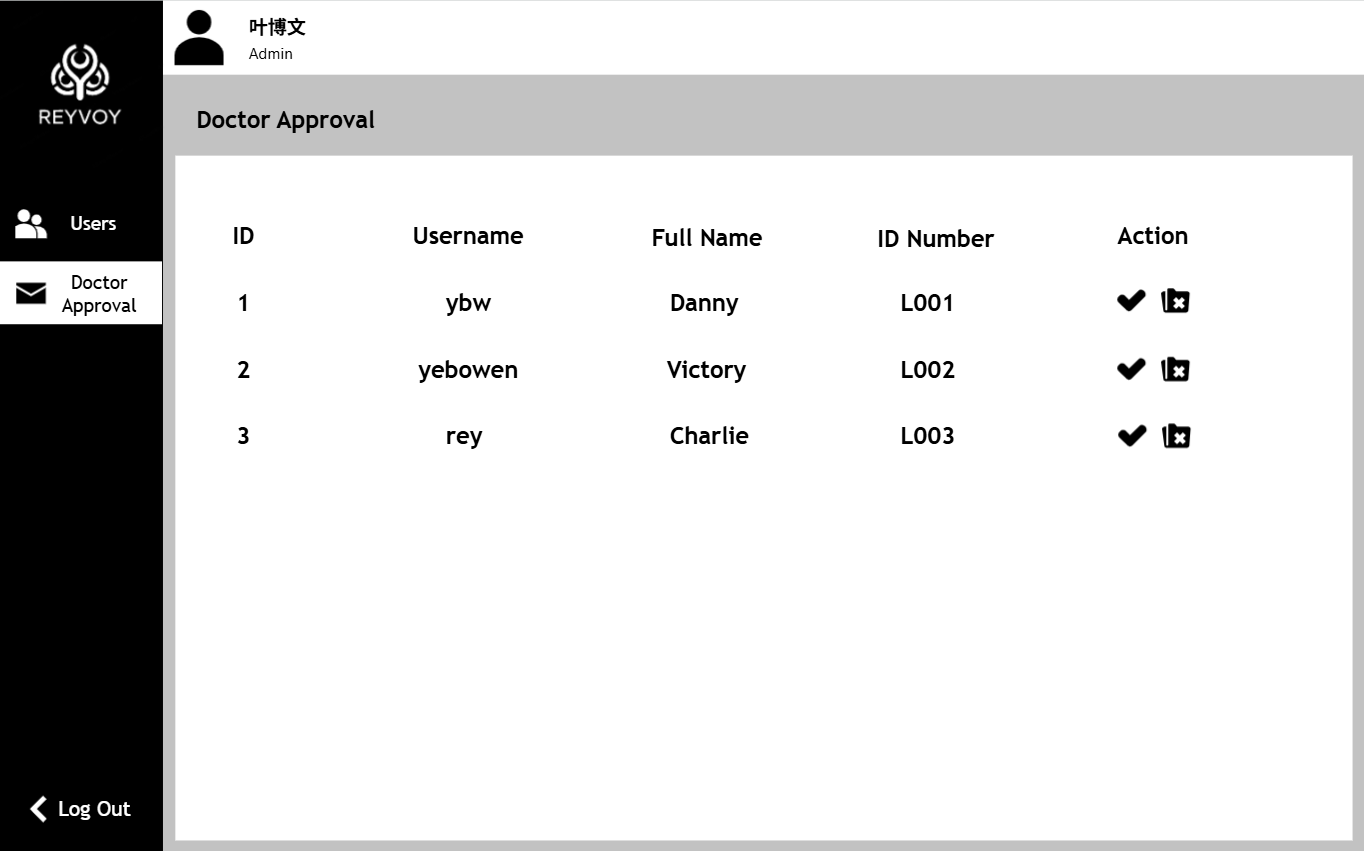


图 10 管理员界面 2

#### 医生界面

医生界面操作步骤：

1. 左边有一个菜单栏，包括三项菜单：Dashboard、Predict和Patient Request。
2. 在Dashboard中，医生可以查询与他所关联的患者的信息，并通过右上角中的Search功能进行按照患者姓名进行查找记录。
3. 在Predict中，医生可以查找要被预测的患者，然后输入该患者的身体情况信息后点击“Predict”按钮进行预测。系统会调用已训练过的模型按照所输入的信息来预测该患者的癌症风险概率。
4. 在Patient Request中，医生可以看到用户对他发送建立医患关系的请求并选择批准或拒绝该请求。
5. 最后，如果想要退出，点击左下角中的“Log Out”按钮。

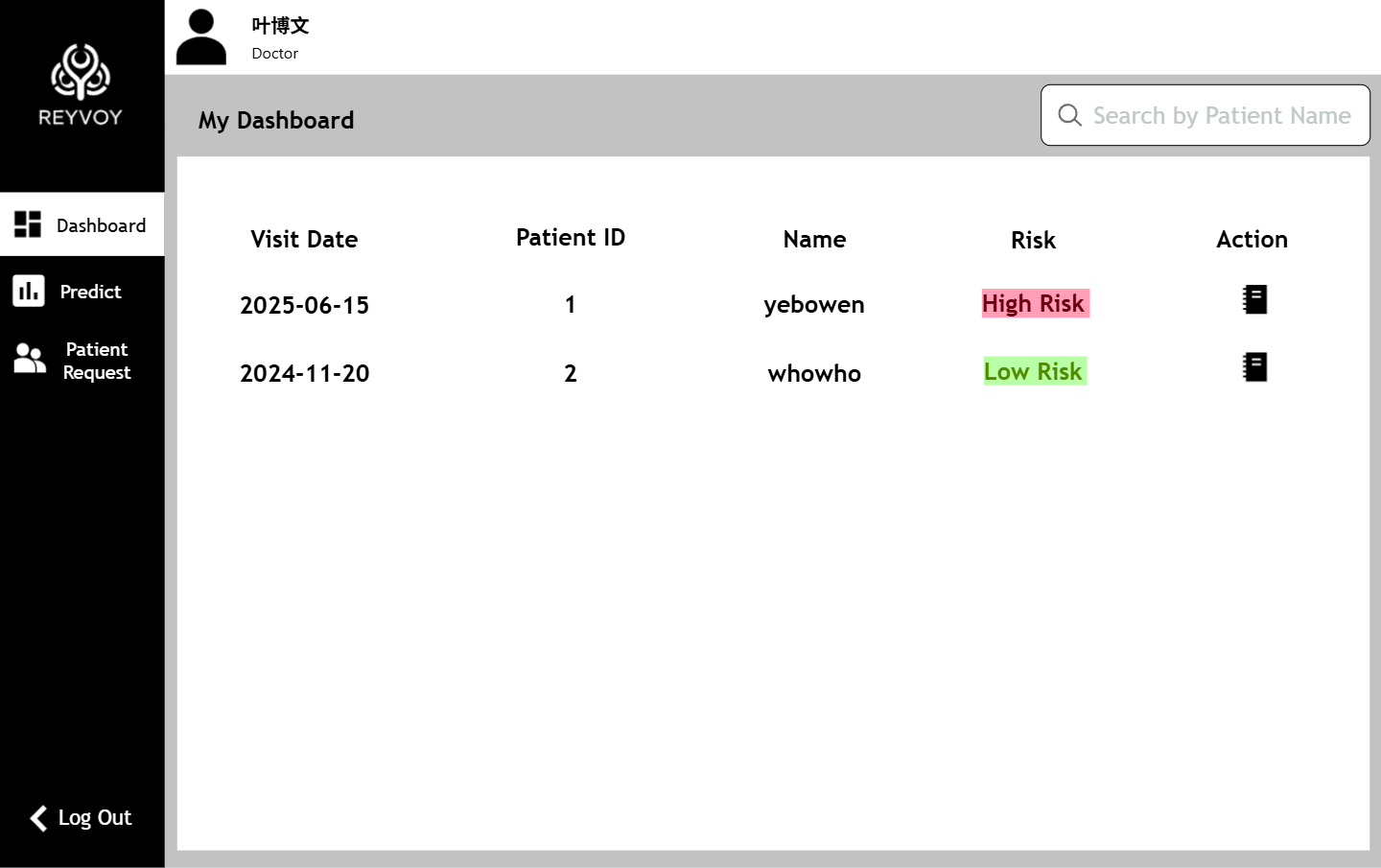


图 11 医生界面 1

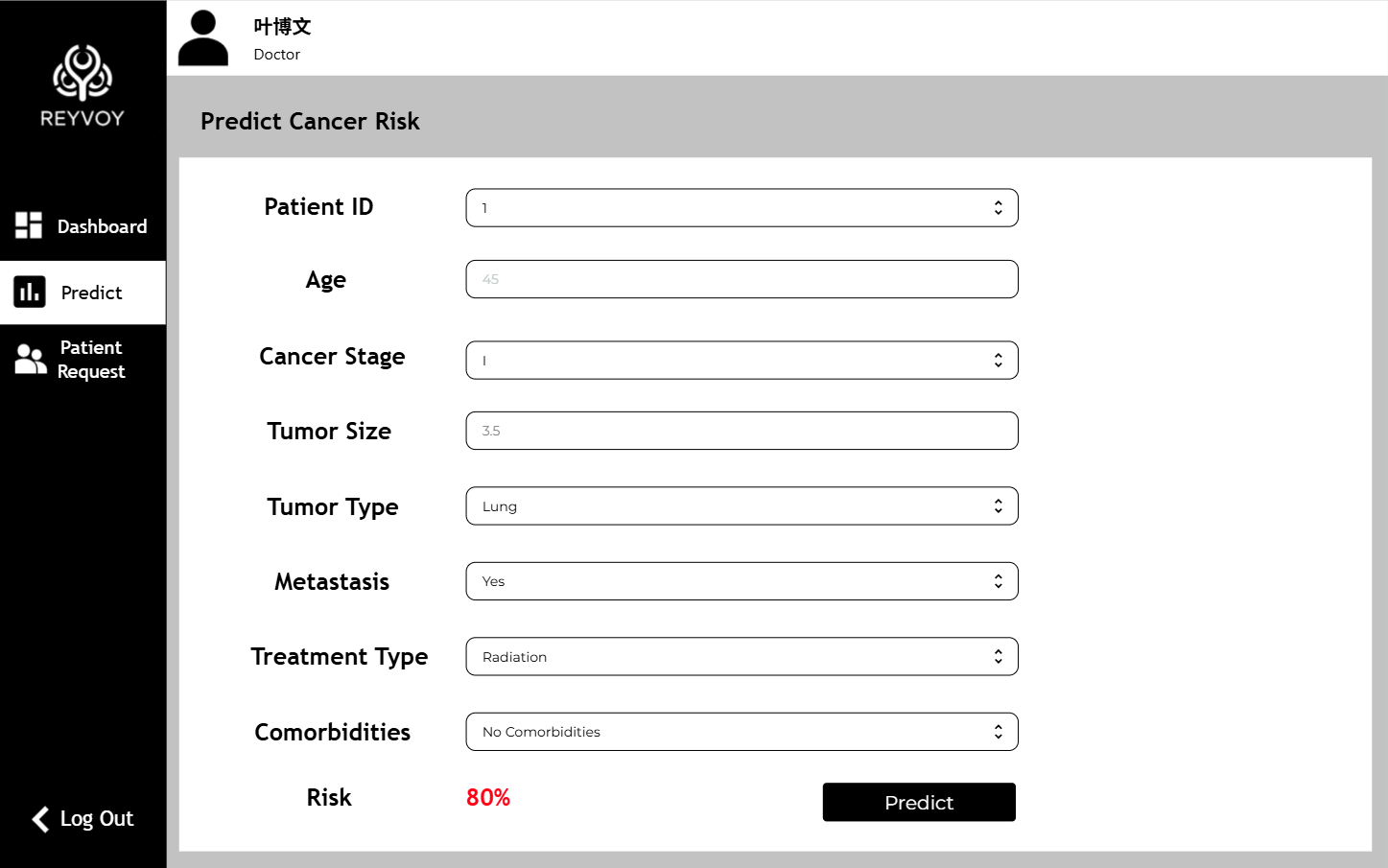


图 12 医生界面 2



图 13 医生界面 3

#### 患者界面

患者界面操作步骤：

1. 左边有一个菜单栏，包括两项菜单：Dashboard和Find Doctor。
2. 在Dashboard中，患者可以查询自己历史预测记录的信息。患者也可以使用右上角的Search功能进行按照医生名进行查找记录。
3. 在Find Doctor中，患者可以向指定的医生发送建立医患关系的请求。一旦请求被医生接受，该医生便获得了为该患者执行新风险预测的权限。

最后，如果想要退出，点击左下角中的“Log Out”按钮。

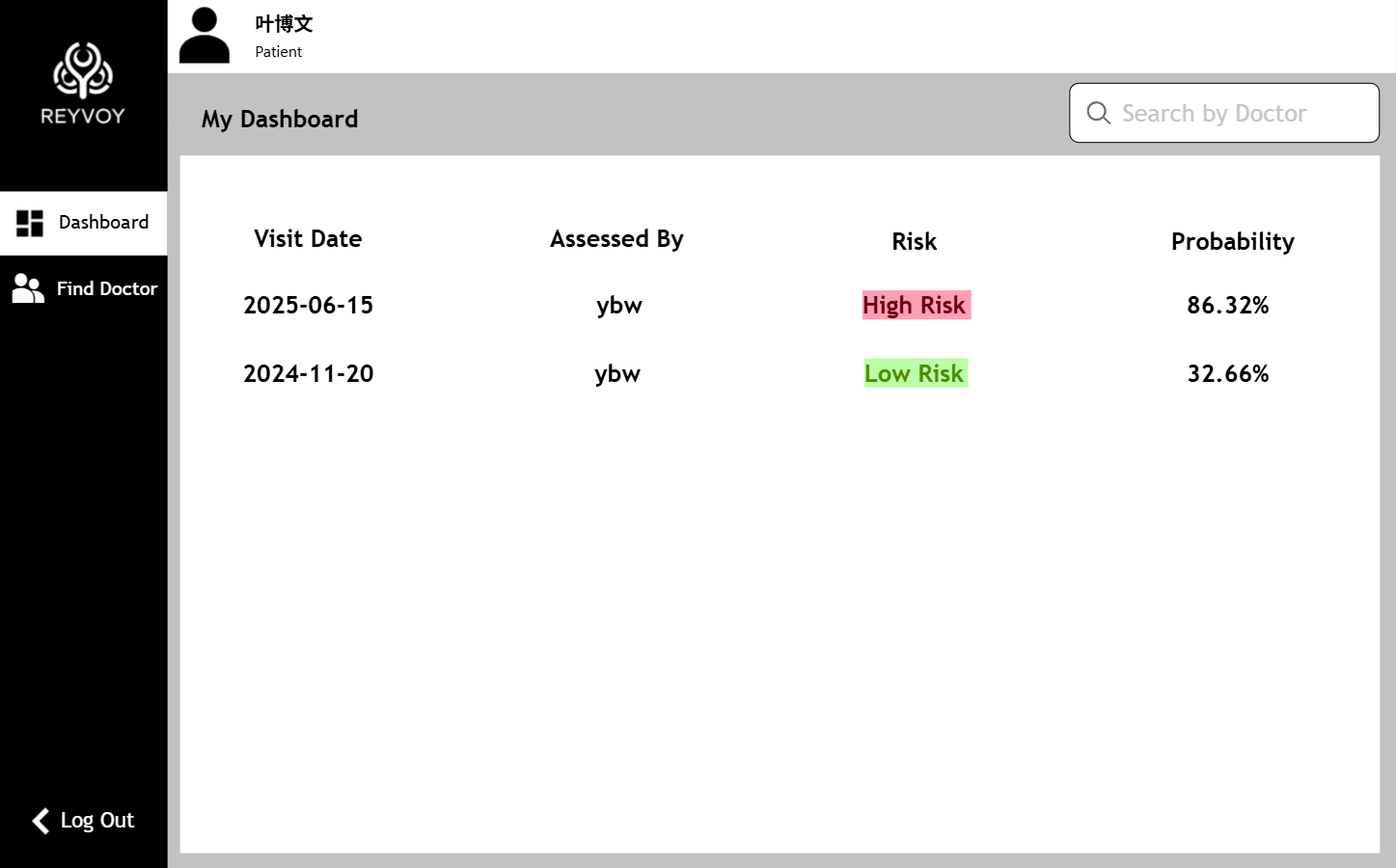


图 14 患者界面 1

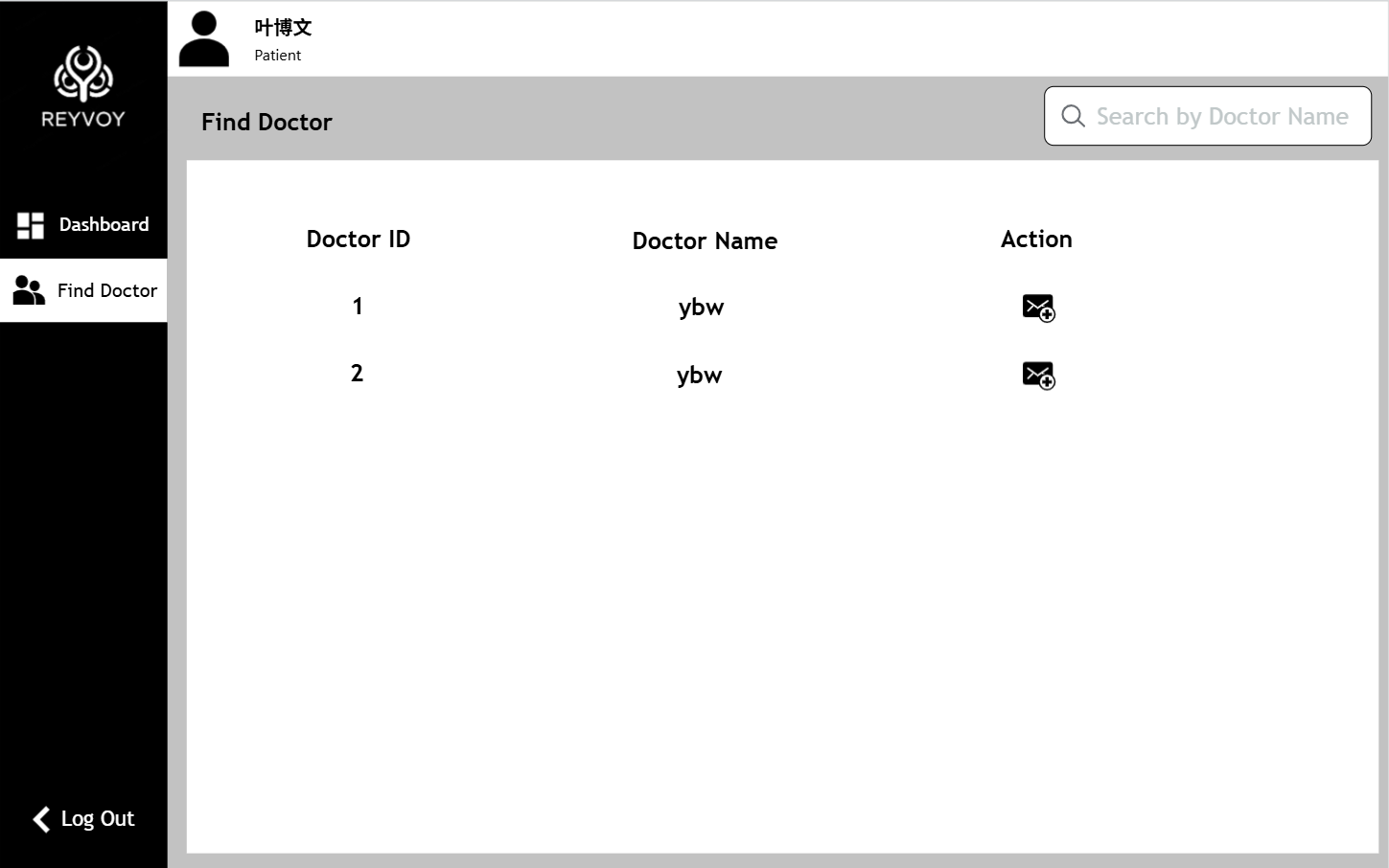


图 15 患者界面 2

## 详细设计

### 时序图

#### 用户登录顺序图

在本系统中，用户登录流程和用户操作流程都是核心功能。用户登录流程图展示了用户从打开登录页面到成功登录系统的整个过程，包含了用户注册、用户登录、用户登录验证和系统反馈等步骤。用户登录时序图展示了用户在登录过程中各个组件之间的交互。

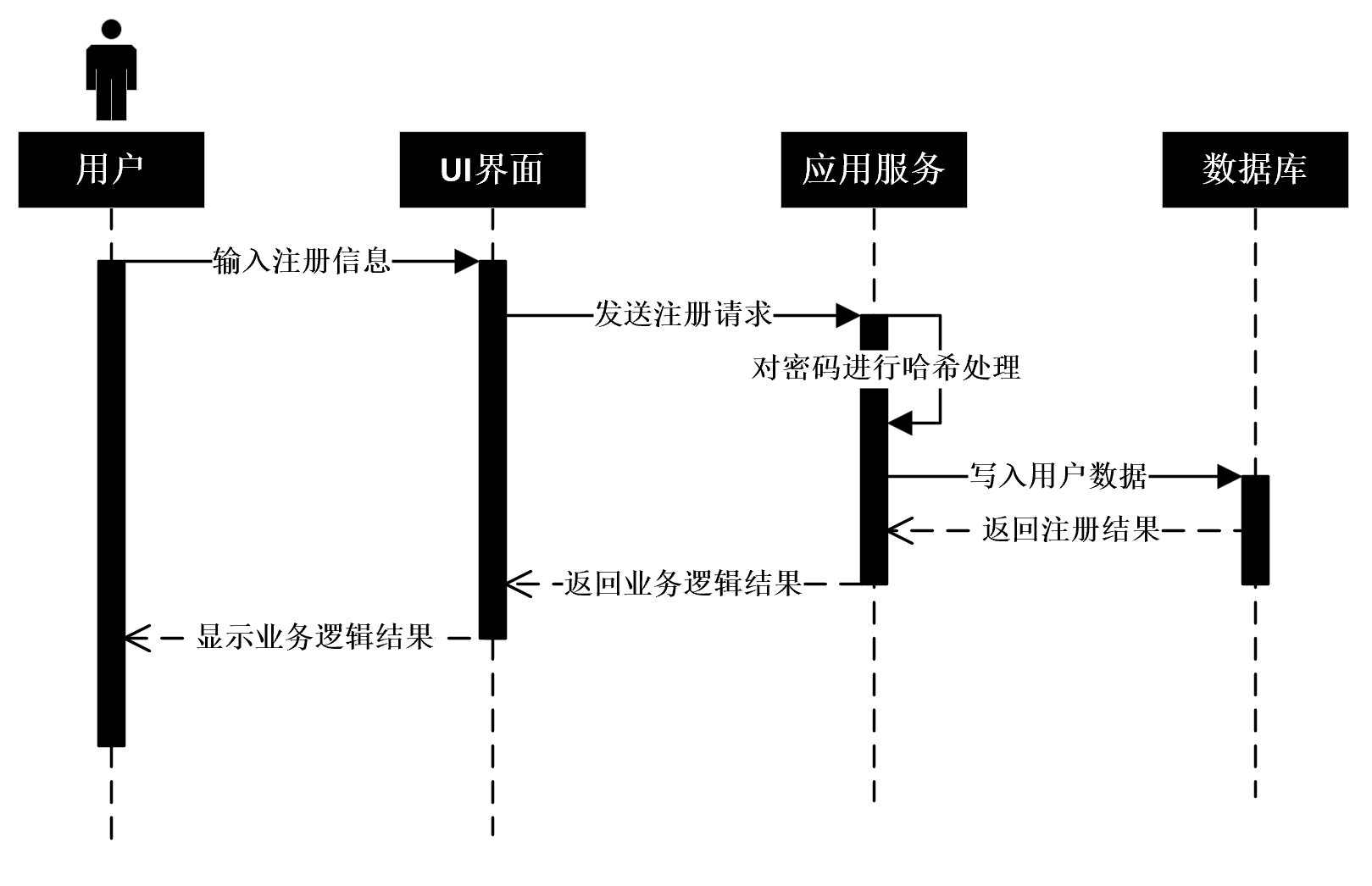


图 16 用户注册顺序图

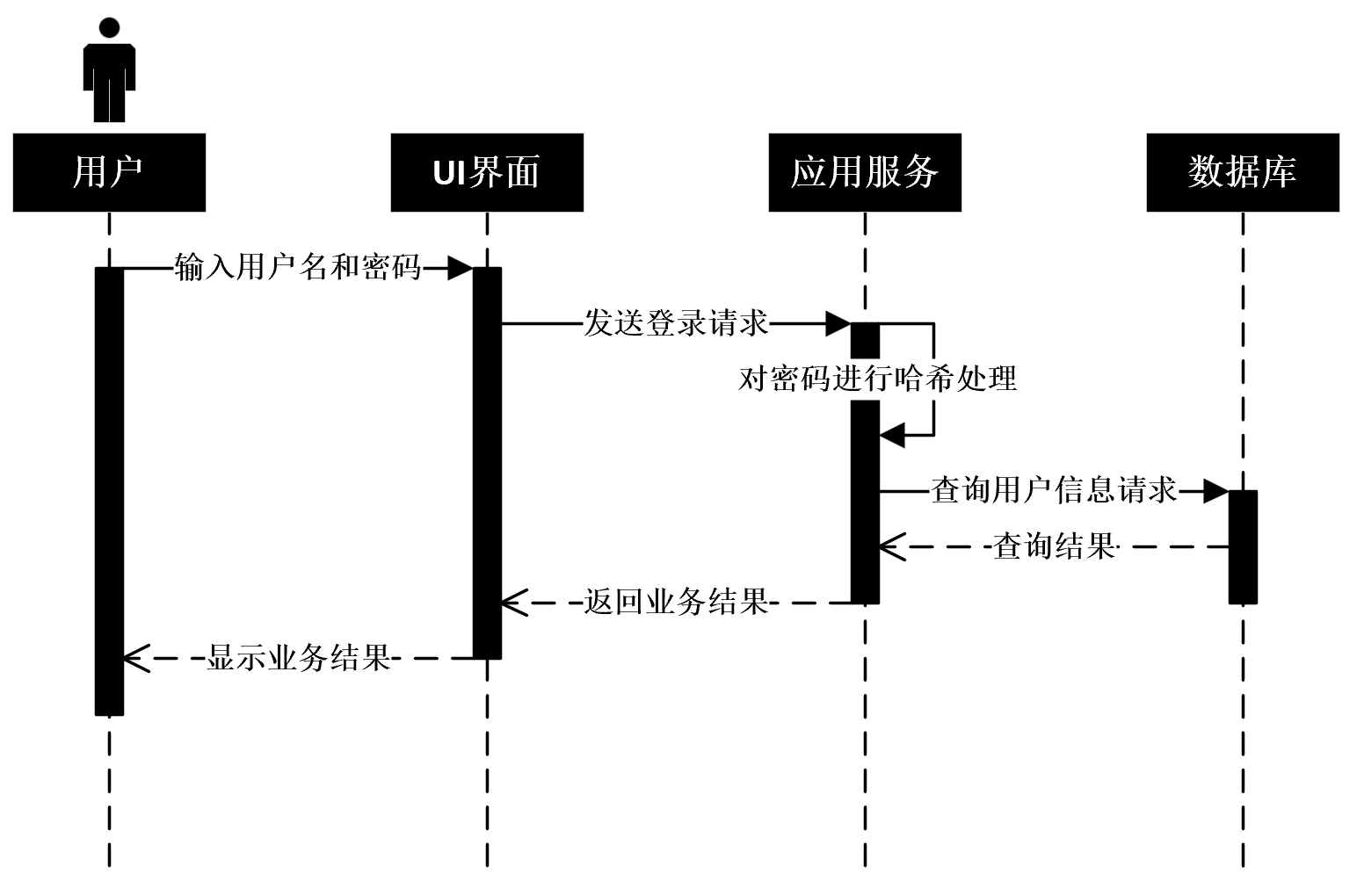


图 17 用户登录过程时序图

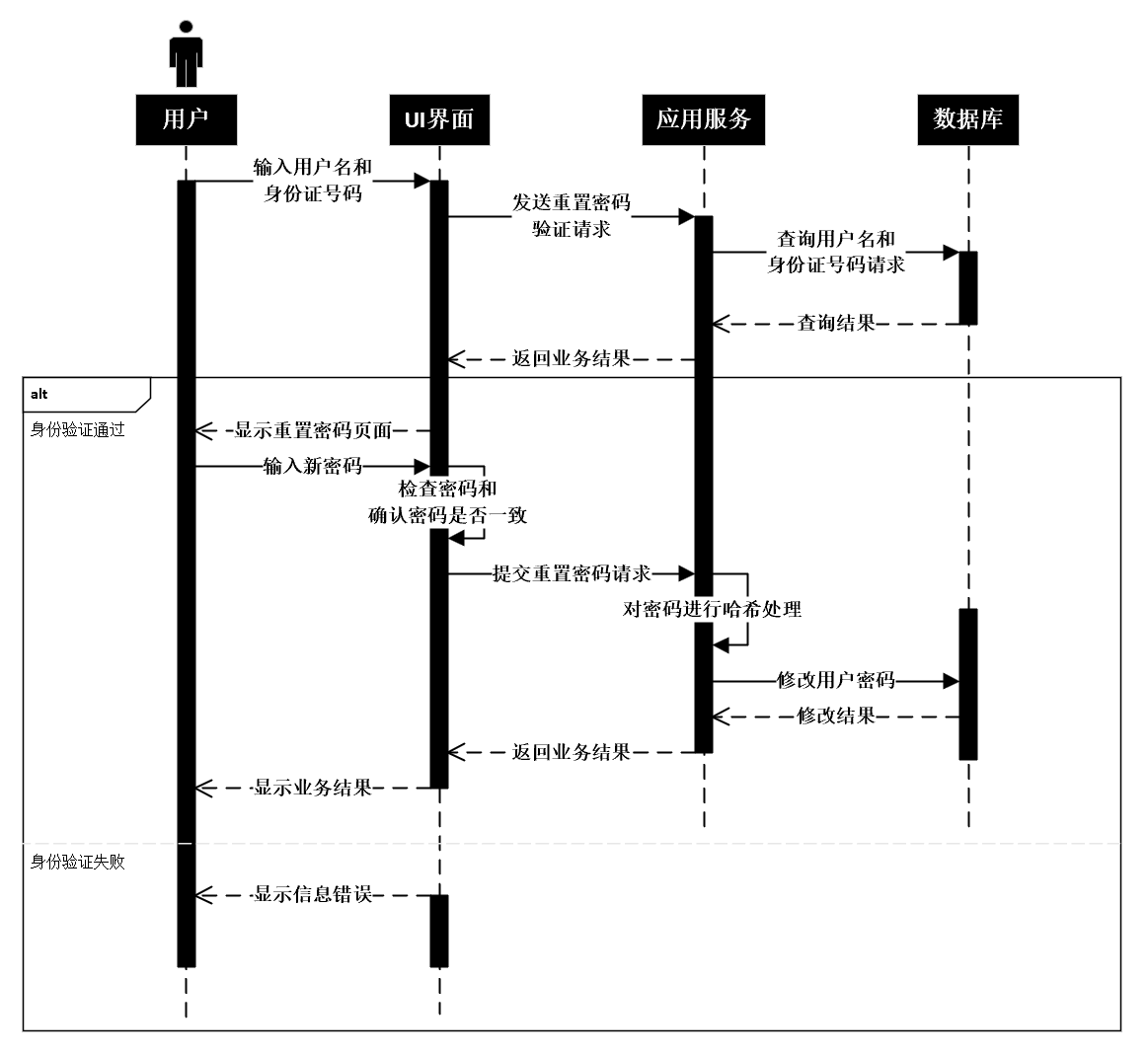


图 18 用户忘记密码顺序图

#### 管理员操作时序图

管理员对用户的增删改查操作的顺序图相似，所以这里只列出增删操作的顺序图和批准医生注册的顺序图。它们的顺序图如下图所示。



图 19 管理员增加用户顺序图

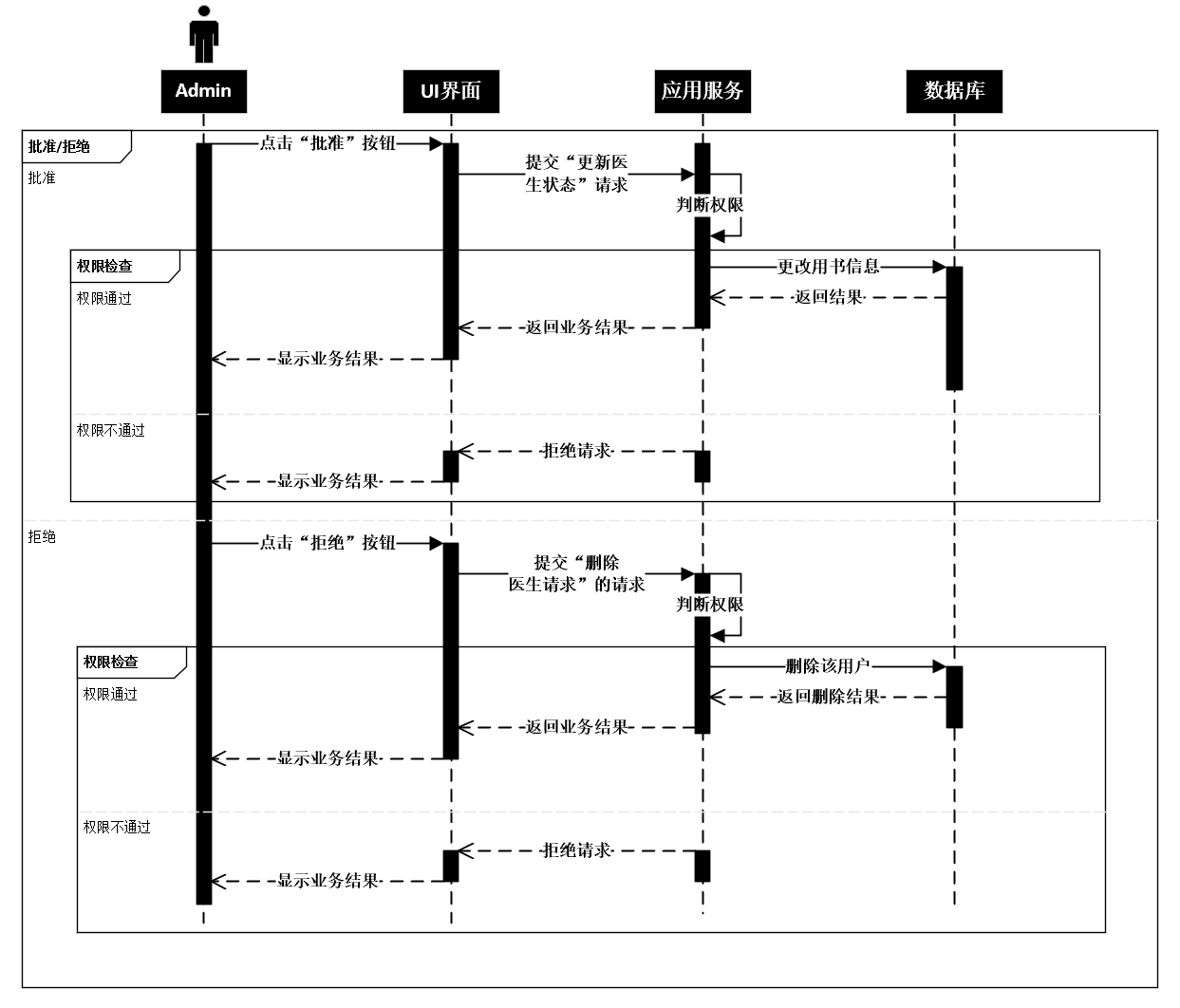


图 20 管理员批准/拒绝医生注册时序图

#### 医生操作顺序图

本系统中，医生主要有三个功能：查询与他关联的患者的历史预测记录信息、查看并判断是否批准患者对他发出的关联请求和按照检测用户的身体情况进行癌症风险预测。

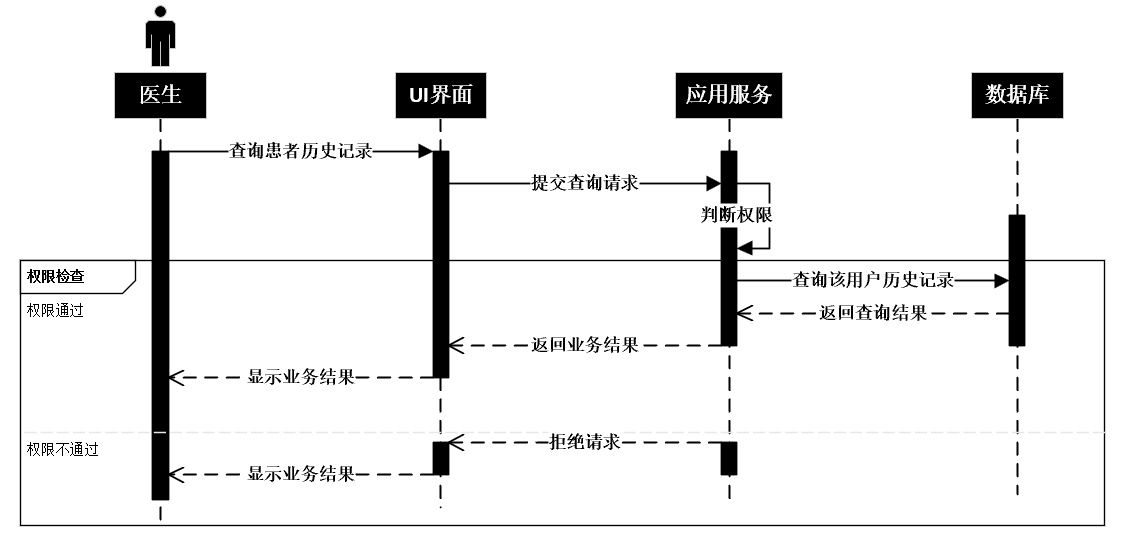


图 21 医生查询特定用户信息时序图



图 22 医生批准关联请求操作时序图

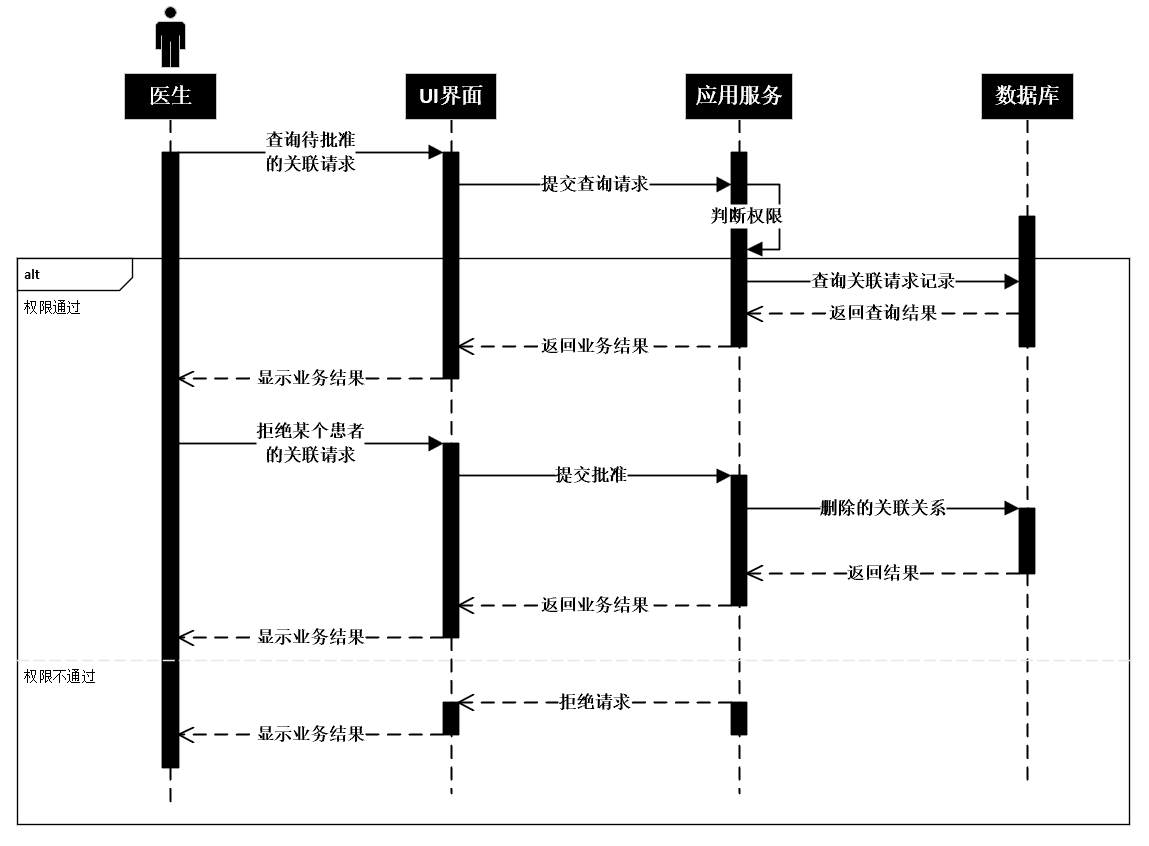


图 23 医生拒绝关联请求操作时序图



图 24 医生癌症风险预测时序图

#### 患者操作时序图

本系统中，患者有两个主要功能：查询历史预测记录和对某个医生发出关联请求。

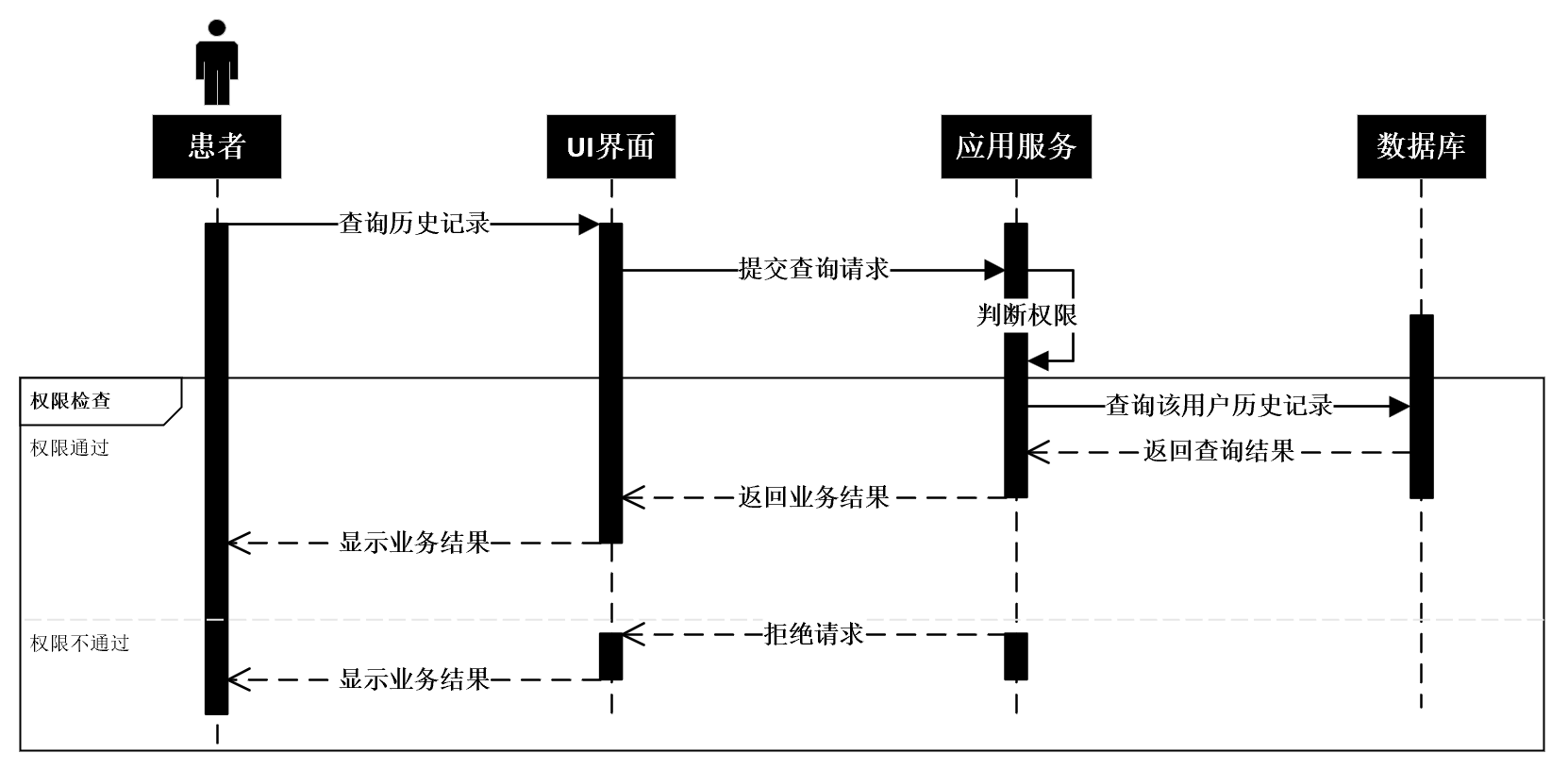


图 25 患者查询历史记录时序图

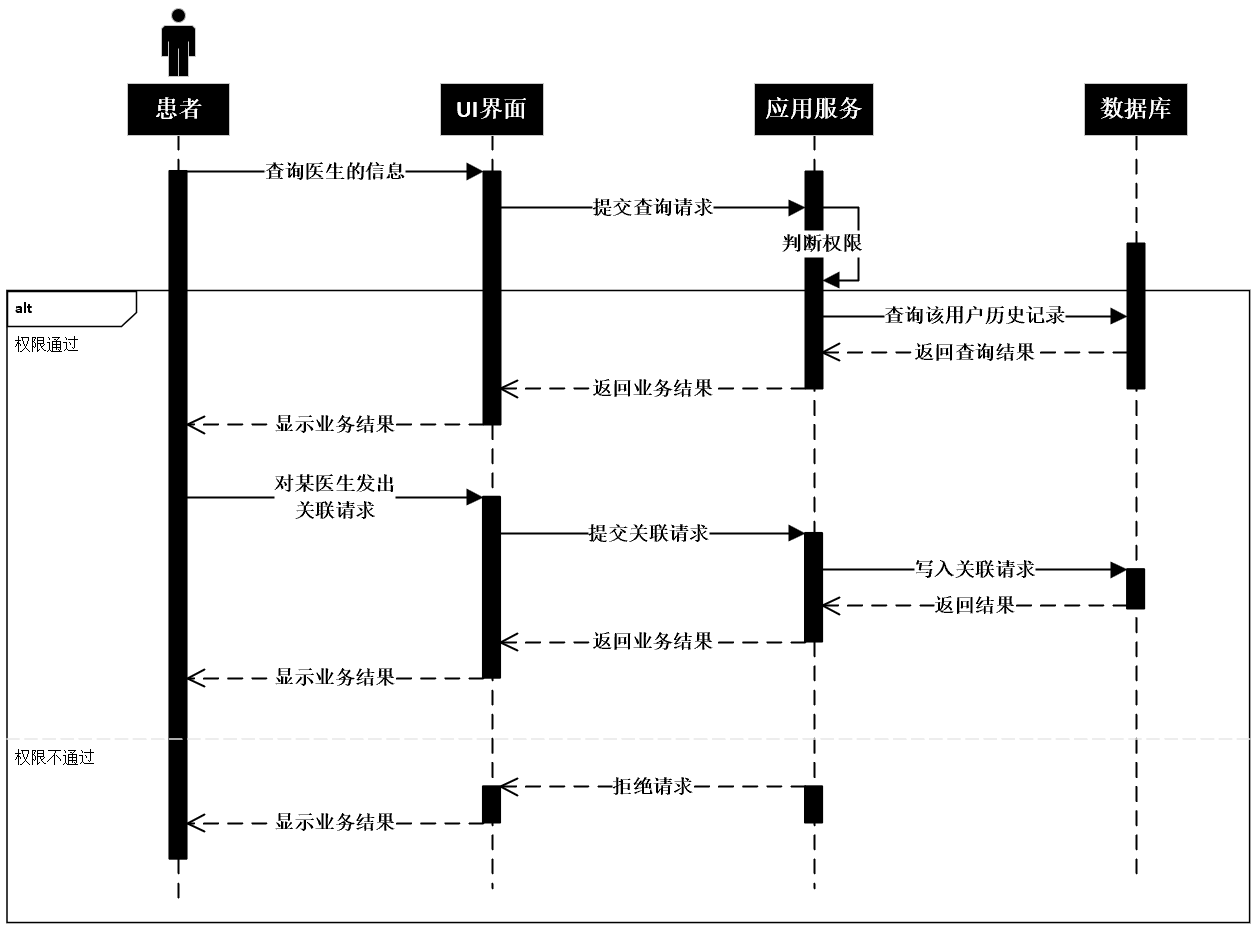


图 26 患者发出关联请求时序图

### 类图

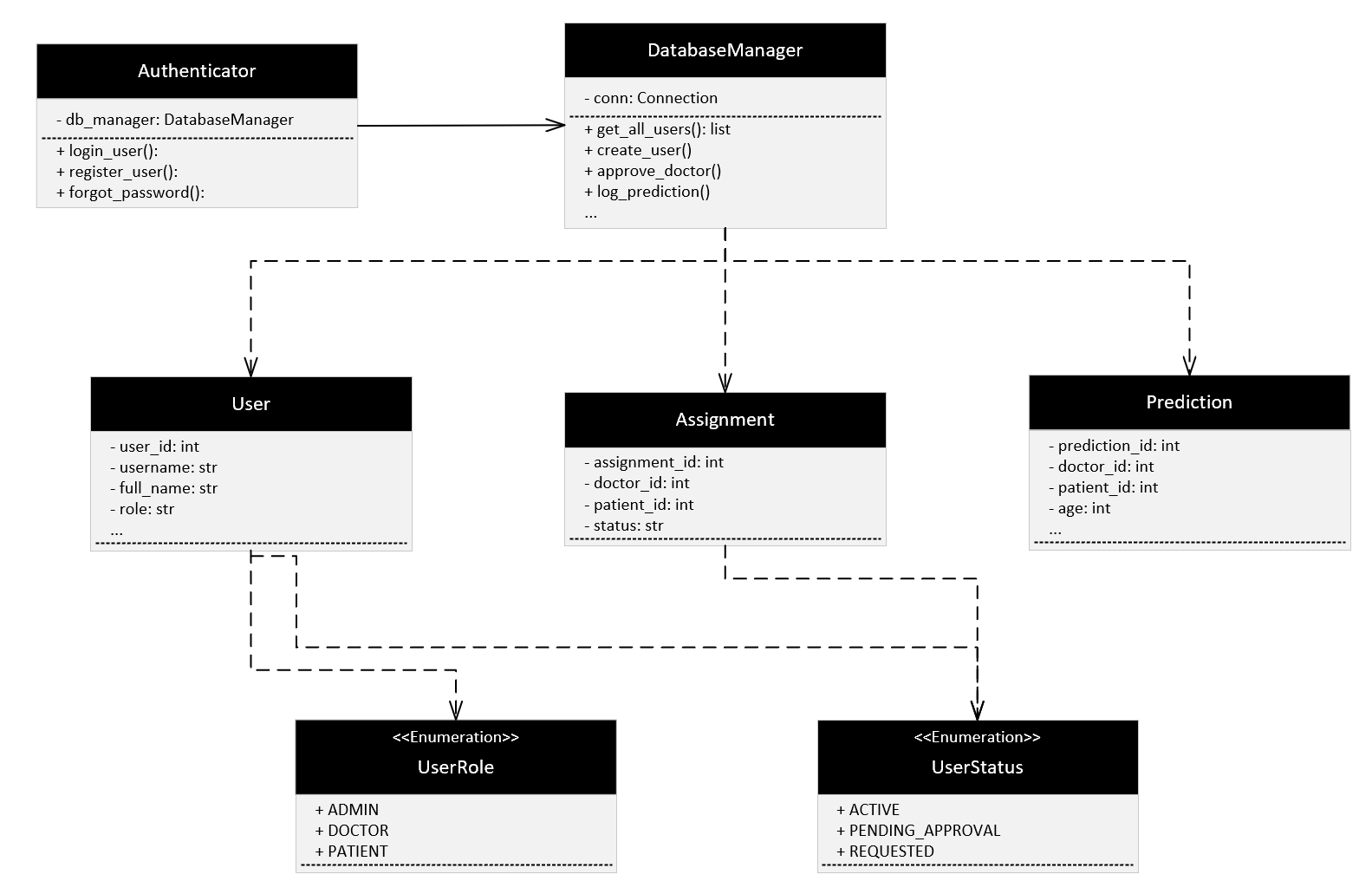


图 27 本系统类图

# 系统实现

## 数据库实现

数据库表创建：

1. ‘users’表

**CREATE** **TABLE** **IF** **NOT** **EXISTS** users **(**

user\_id INTEGER **PRIMARY** **KEY** AUTOINCREMENT**,**

username VARCHAR**(**20**)** **NOT** **NULL** **UNIQUE,**

password\_hash VARCHAR**(**128**)** **NOT** **NULL,**

full\_name VARCHAR**(**30**)** **NOT** **NULL,**

role VARCHAR**(**10**)** **NOT** **NULL** **CHECK(**role **IN** **(**'doctor'**,** 'patient'**,** 'admin'**)),**

status VARCHAR**(**20**)** **NOT** **NULL** **CHECK(**status **IN** **(**'active'**,** 'pending\_approval'**)),**

id\_number VARCHAR**(**18**)** **NOT** **NULL** **UNIQUE**

dobDATE

**);**

代码 1 'users'表创建SQL语句

1. ‘doctor\_patient\_assignments’表

**CREATE** **TABLE** **IF** **NOT** **EXISTS** doctor\_patient\_assignments **(**

assignment\_id INTEGER **PRIMARY** **KEY** AUTOINCREMENT**,**

doctor\_id INTEGER **NOT** **NULL,**

patient\_id INTEGER **NOT** **NULL,**

status VARCHAR**(**20**)** **NOT** **NULL** **CHECK(**status **IN** **(**'requested'**,** 'active'**)),**

**FOREIGN** **KEY(**doctor\_id**)** **REFERENCES** users**(user\_id)** **ON** **DELETE** **CASCADE,**

**FOREIGN** **KEY(**patient\_id**)** **REFERENCES** users**(user\_id)** **ON** **DELETE** **CASCADE**

**);**

代码 2 'doctor\_patient\_assignments' 表创建SQL语句

1. ‘predictions’ 表

**CREATE** **TABLE** **IF** **NOT** **EXISTS** predictions **(**

prediction\_id INTEGER **PRIMARY** **KEY** AUTOINCREMENT**,**

doctor\_id INTEGER **NOT** **NULL,**

patient\_id INTEGER **NOT** **NULL,**

prediction\_timestamp DATETIME **NOT** **NULL** **DEFAULT** **CURRENT\_TIMESTAMP,**

age INT **NOT** **NULL,**

cancer\_stage VARCHAR**(**5**)** **NOT** **NULL** **CHECK(**cancer\_stage **IN** **(**'I'**,** 'II'**,** 'III'**,** 'IV'**)),**

tumor\_size REAL **NOT** **NULL,**

tumor\_type VARCHAR**(**20**)** **NOT** **NULL** **CHECK(**tumor\_type **IN** **(**'Lung'**,** 'Stomach'**,** 'Cervical'**,** 'Liver'**,** 'Colorectal'**,** 'Breast'**)),**

metastasis VARCHAR**(**5**)** **NOT** **NULL** **CHECK(**metastasis **IN** **(**'Yes'**,** 'No'**)),**

treatment\_type VARCHAR**(**20**)** **NOT** **NULL** **CHECK(**treatment\_type **IN** **(**'Radiation'**,** 'Chemotherapy'**,** 'Surgery'**,** 'Targeted Therapy'**,** 'Immunotherapy'**)),**

comorbidities VARCHAR**(**15**)** **NOT** **NULL** **CHECK(**comorbidities **IN** **(**'No Comorbidities'**,** 'Diabetes, Hepatitis B'**,** 'Hepatitis B'**,** 'Hypertension'**,** 'Diabetes, Hypertension'**,** 'Diabetes, Hepatitis B'**,** 'Hypertension, Hepatitis B'**)),**

predicted\_class VARCHAR**(**15**)** **NOT** **NULL,**

prediction\_probability REAL **NOT** **NULL,**

**FOREIGN** **KEY(**doctor\_id**)** **REFERENCES** users**(user\_id)** **ON** **DELETE** **CASCADE,**

**FOREIGN** **KEY(**patient\_id**)** **REFERENCES** users**(user\_id)** **ON** **DELETE** **CASCADE**

**);**

代码 3 'predictions'表创建SQL语句

## 前端页面实现

用户可以通过前端页面实现对数据库中进行操作。可以通过注册账户对‘users’表进行添加信息。

‘patient’可以对医生发出关联请求，对‘doctor\_patient\_assignments’表进行添加信息，也可以查询自己的历史预测记录，即对‘predictions’表进行查询信息。

‘doctor’可以对‘patient’所发来的关联请求进行批准或拒绝：如果批准，则相当于对‘doctor\_patient\_assignments’表进行修改操作；而如果拒绝，则相当于对‘doctor\_patient\_assignments’表进行删除操作。‘doctor’也可以查看与他关联的患者的信息，即对‘predictions’表进行查询操作。

‘admin’可以进行用户的管理，即对用户的信息进行增删改查的操作。又可以对status为‘pending\_approval’的‘doctor’进行批准或拒绝。

具体前端代码如下图所示：

1. 公共使用的UI Components，被Admin Dashboard，Doctor Dashboard和Patient Dashboard通用。

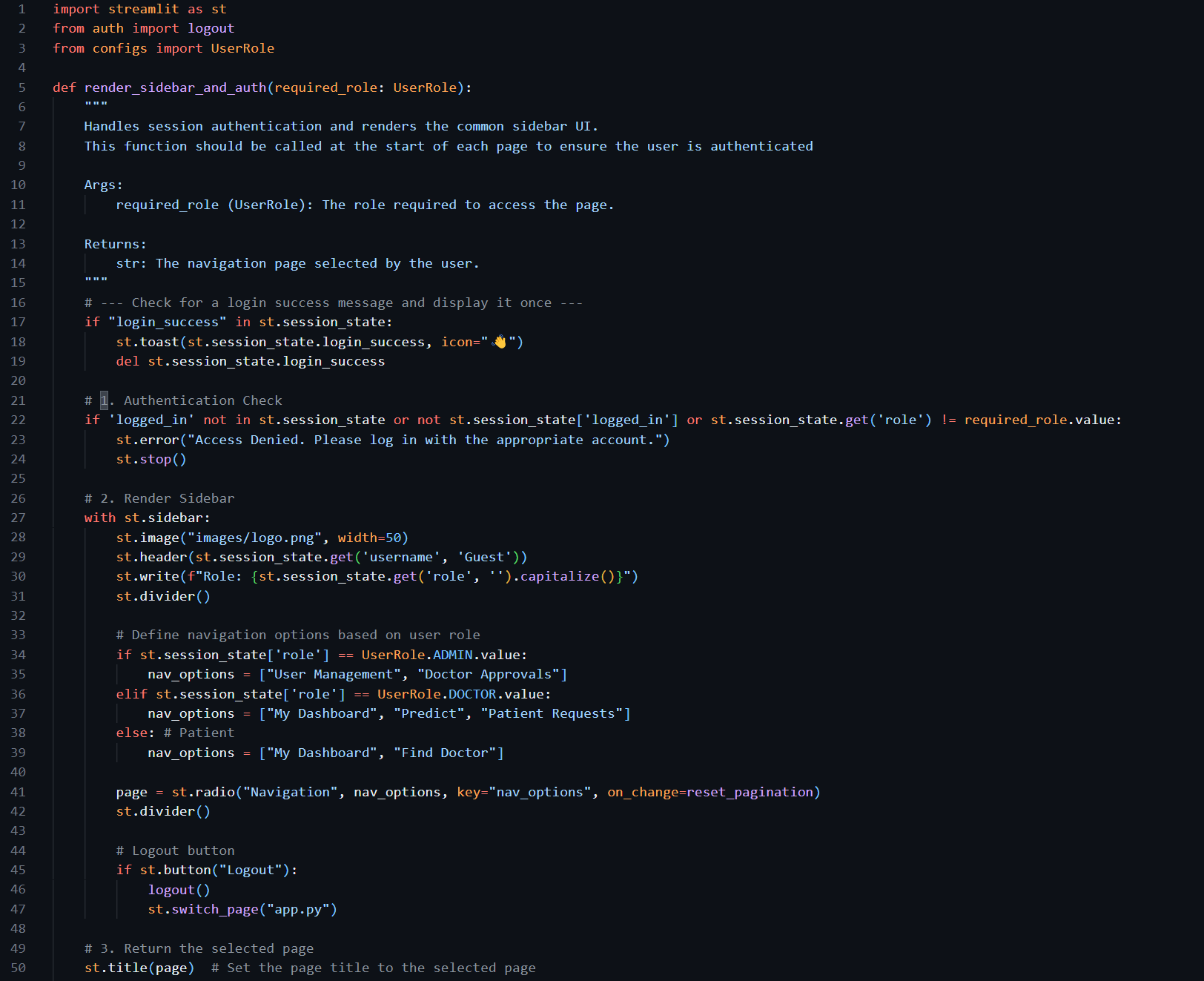


图 28 公用UI Components 代码 1

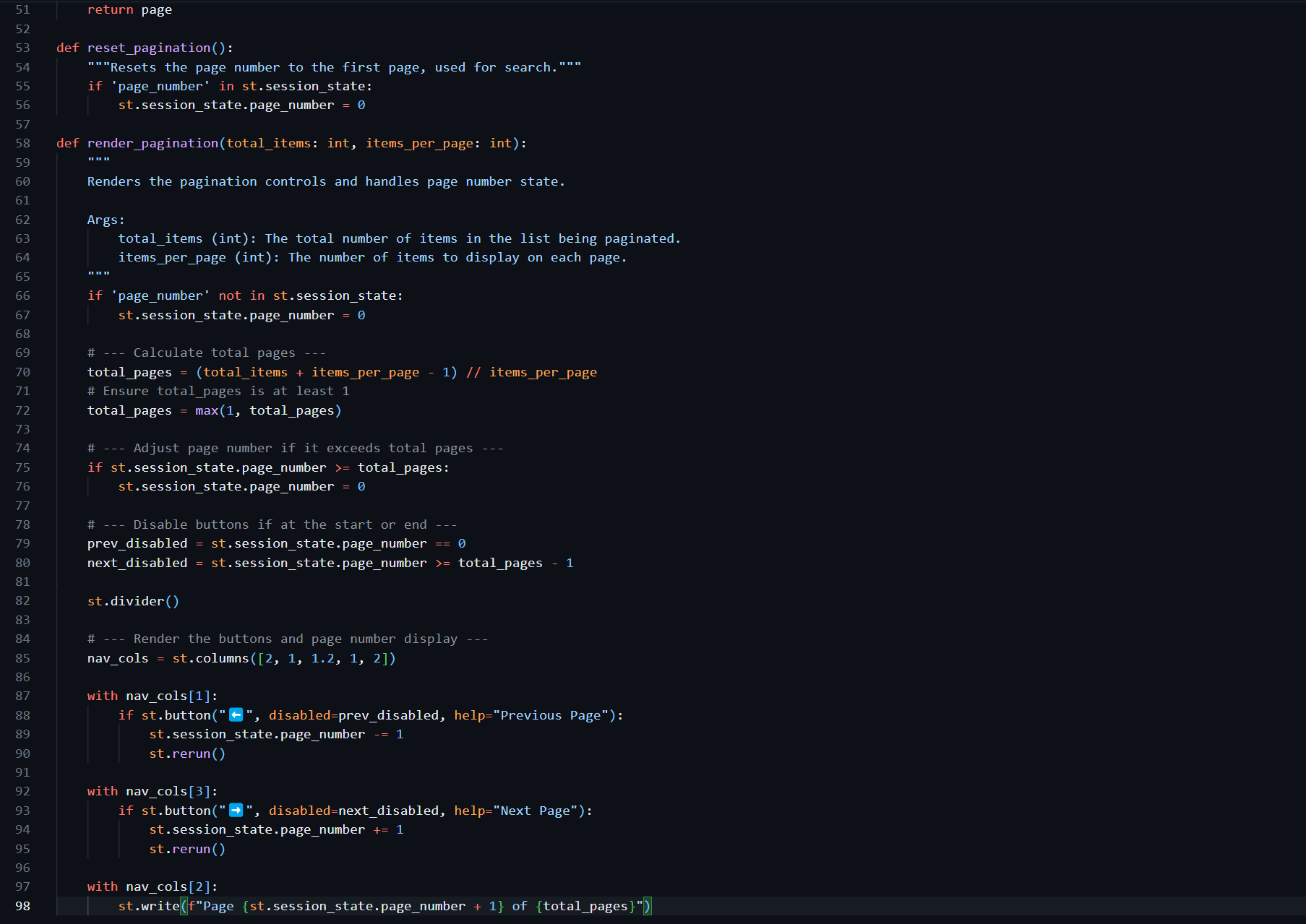


图 29 公用UI Components 代码 2

1. ‘Admin’的‘User Management’页面。在此可以查询所有用户的信息，同时，也可以添加用户、删除用户和修改用户的信息。

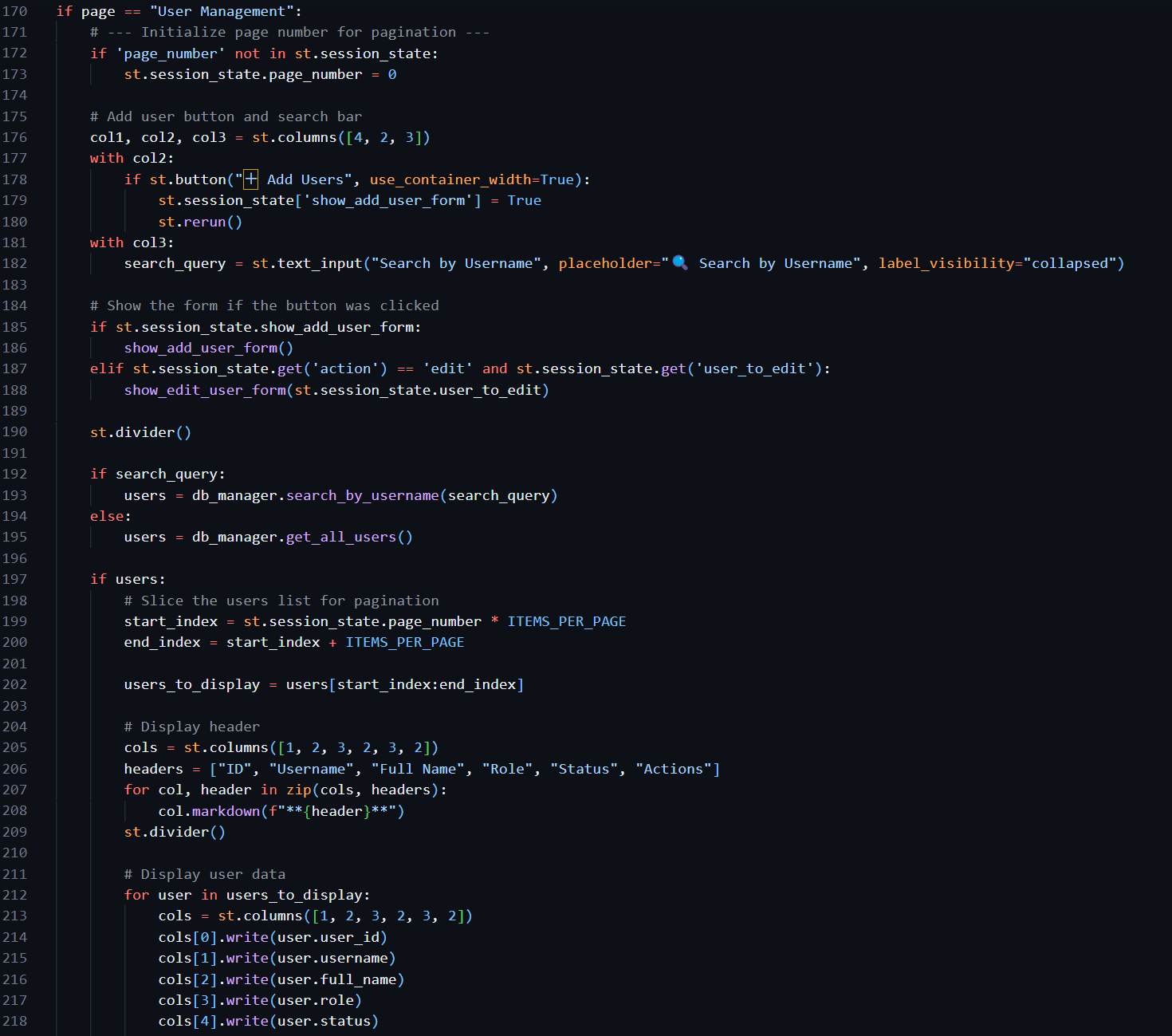


图 30 管理员对用户信息进行查询页面代码 1

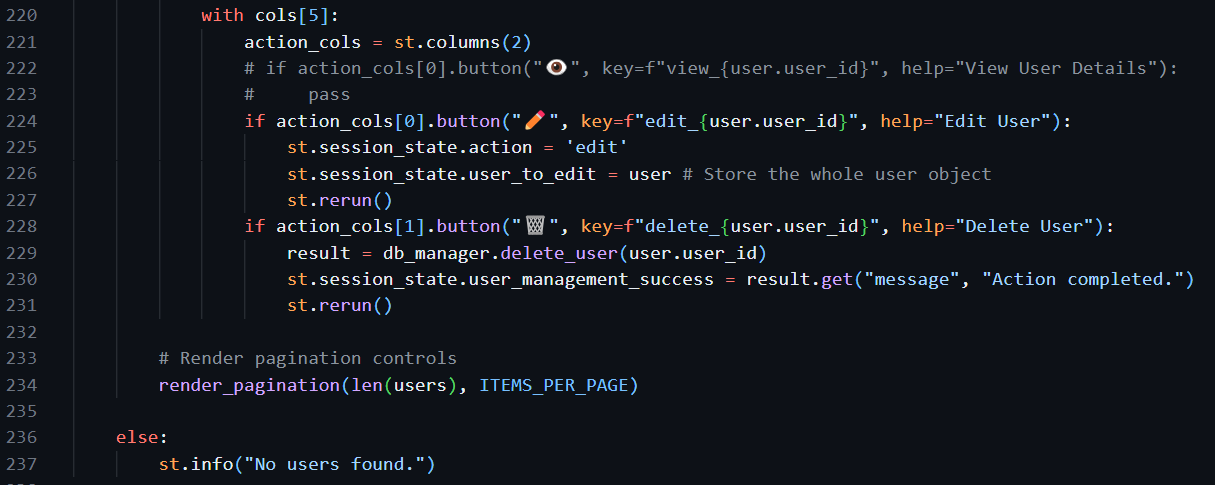


图 31 管理员对用户信息进行查询页面代码 2

1. ‘admin’的‘Doctor Approvals’页面。在此可以对‘doctor’注册请求进行批准或拒绝。

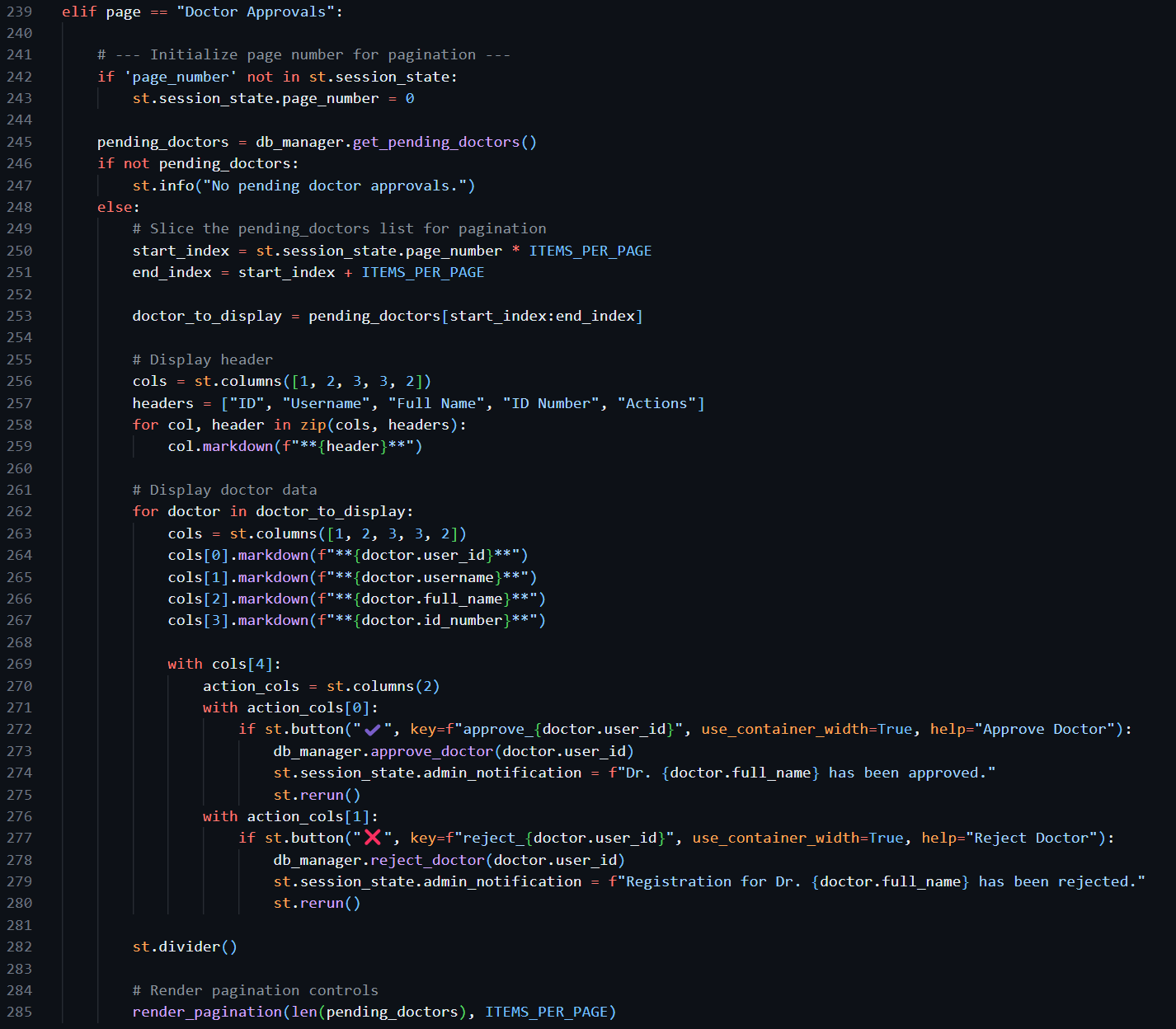


图 32 管理员对医生批准或拒绝页面代码

1. ‘Admin’添加用户的页面。在此，可以输入用户的信息并提交。

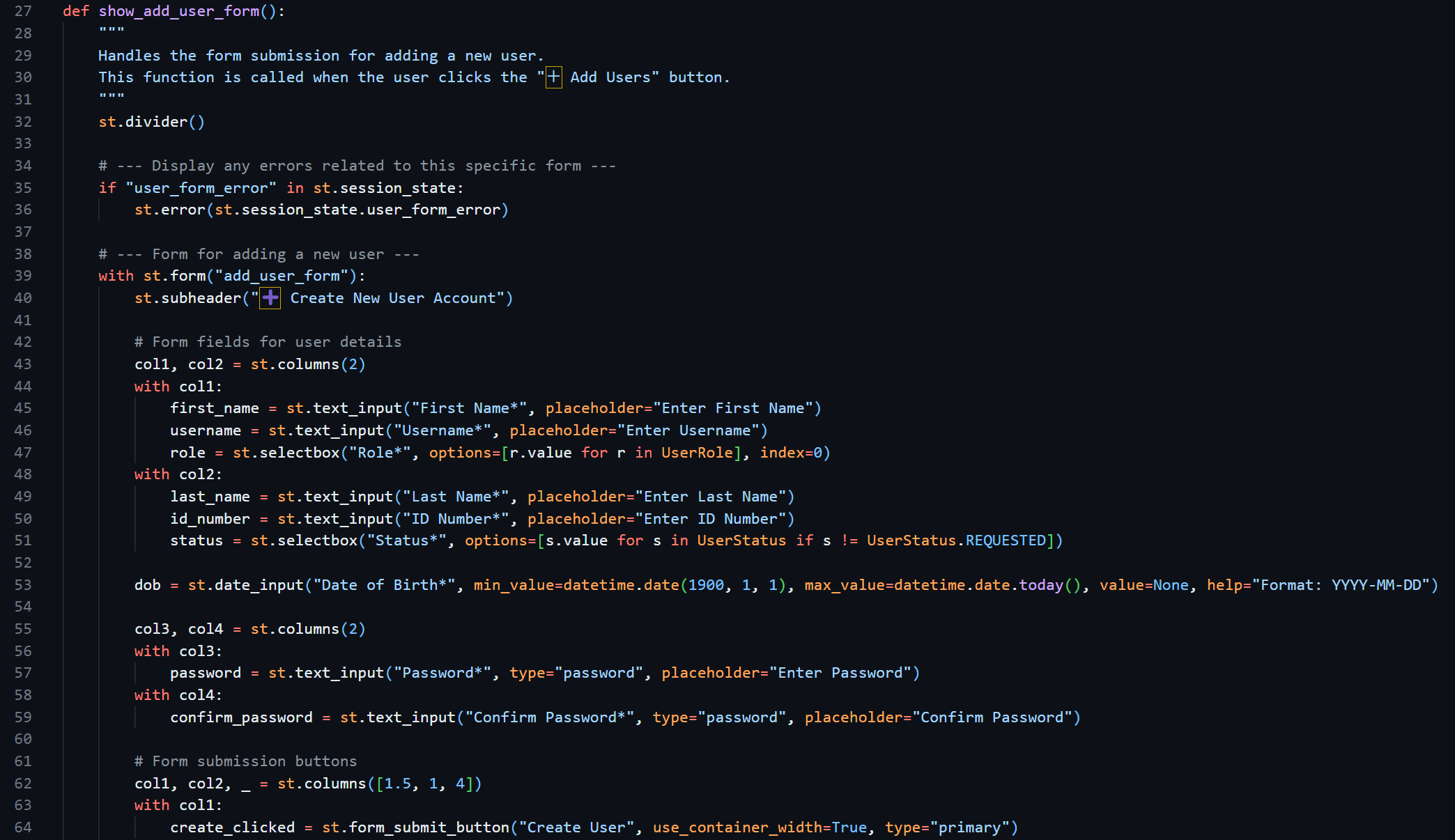


图 33 管理员添加用户的页面代码 1

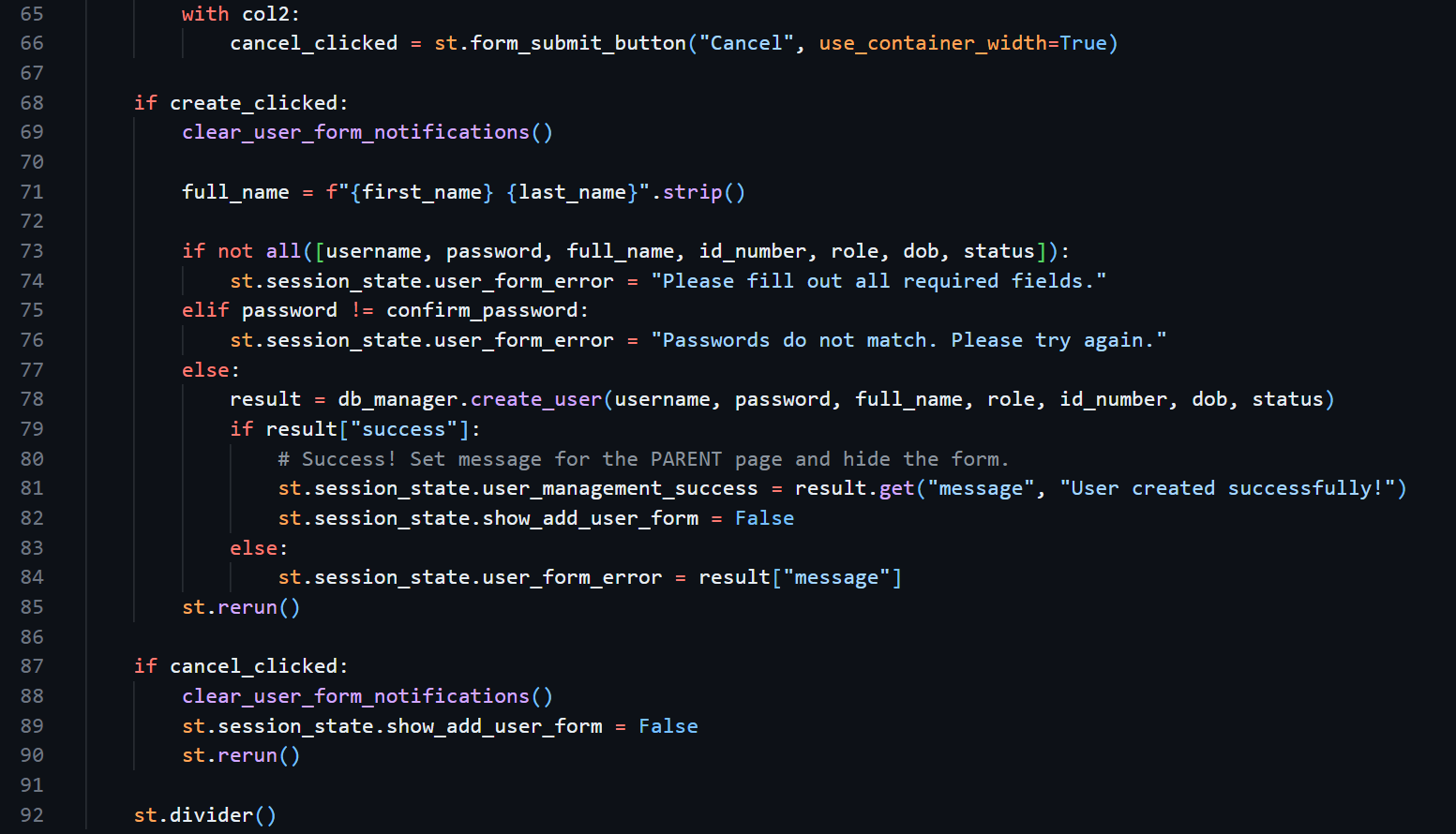


图 34 管理员添加用户的页面代码 2

1. ‘Admin’修改用户的页面。在此，可以输入需要修改的用户信息，默认值为当前的信息。



图 35 管理员对用户信息进行修改操作的代码 1



图 36 管理员对用户信息进行修改操作的代码 2

1. ‘Doctor’的‘My Dashboard’页面。在此，可以查看与他关联的患者的信息，并可以使用查找功能按照患者的姓名进行查找。

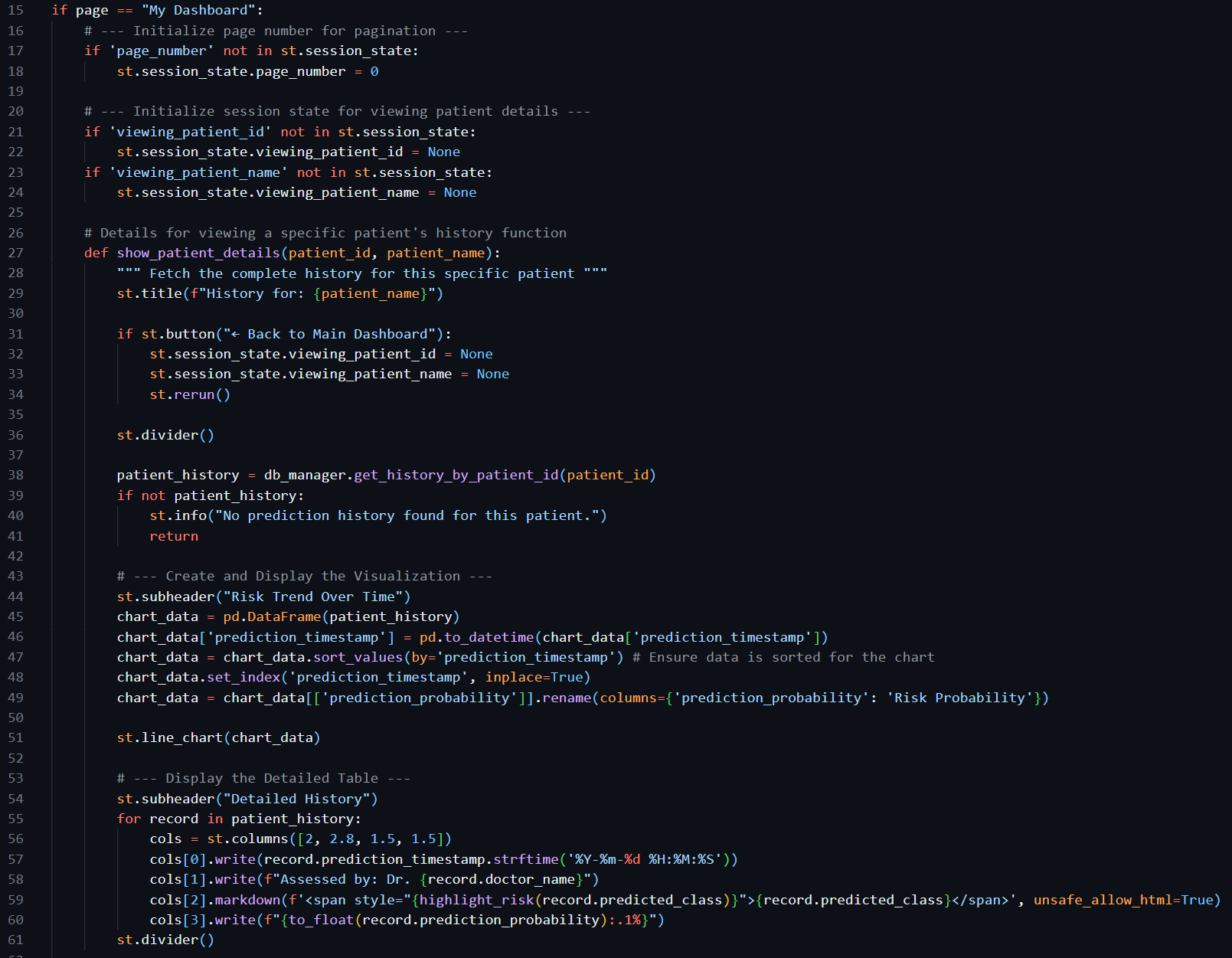


图 37 医生的主页面代码 1

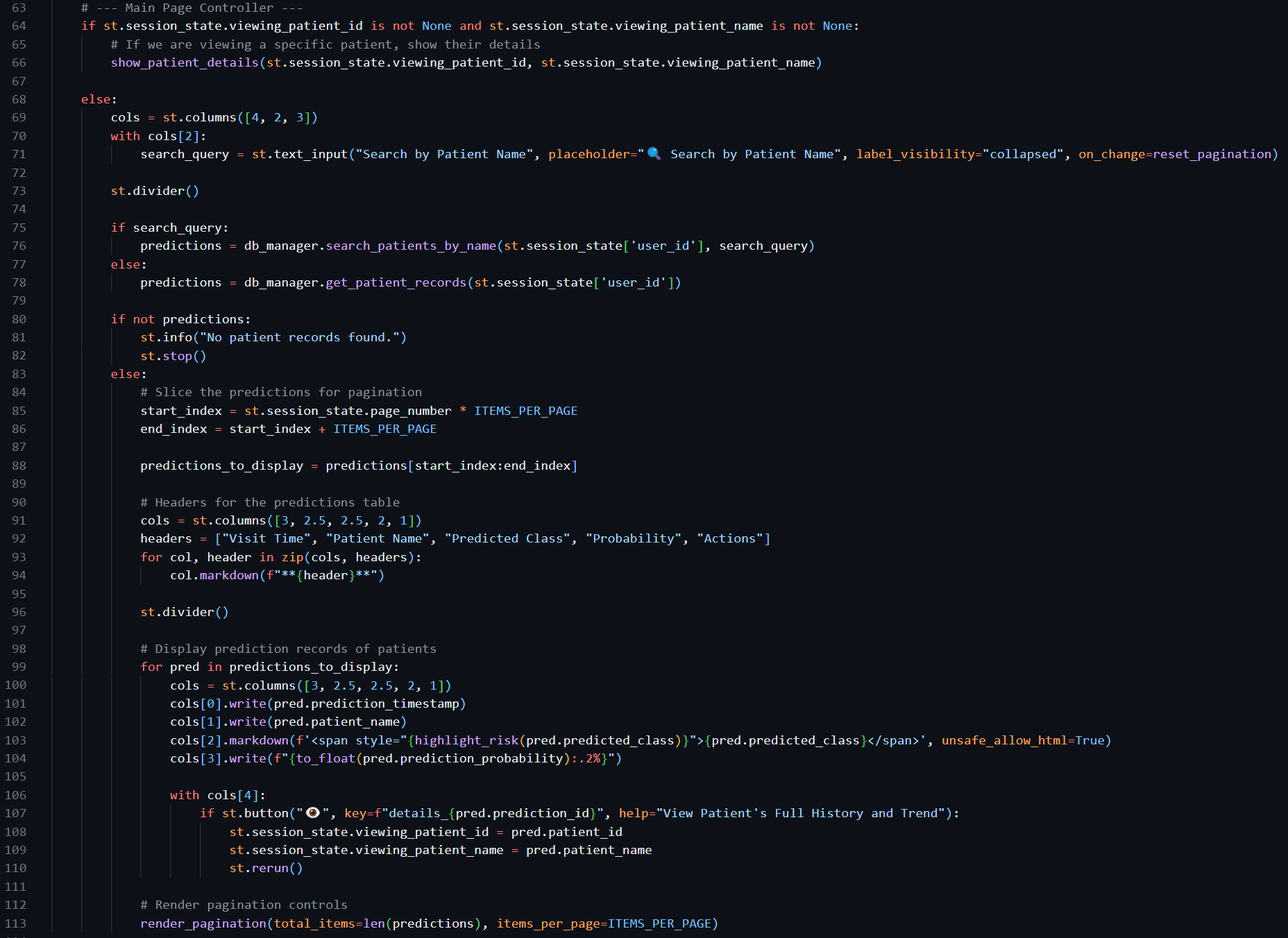


图 38 医生的主页面代码 2

1. ‘Doctor’的预测页面。在此，可以输入患者的身体情况信息并提交。然后，系统通过后台处理后返回患者的癌症风险度预测。

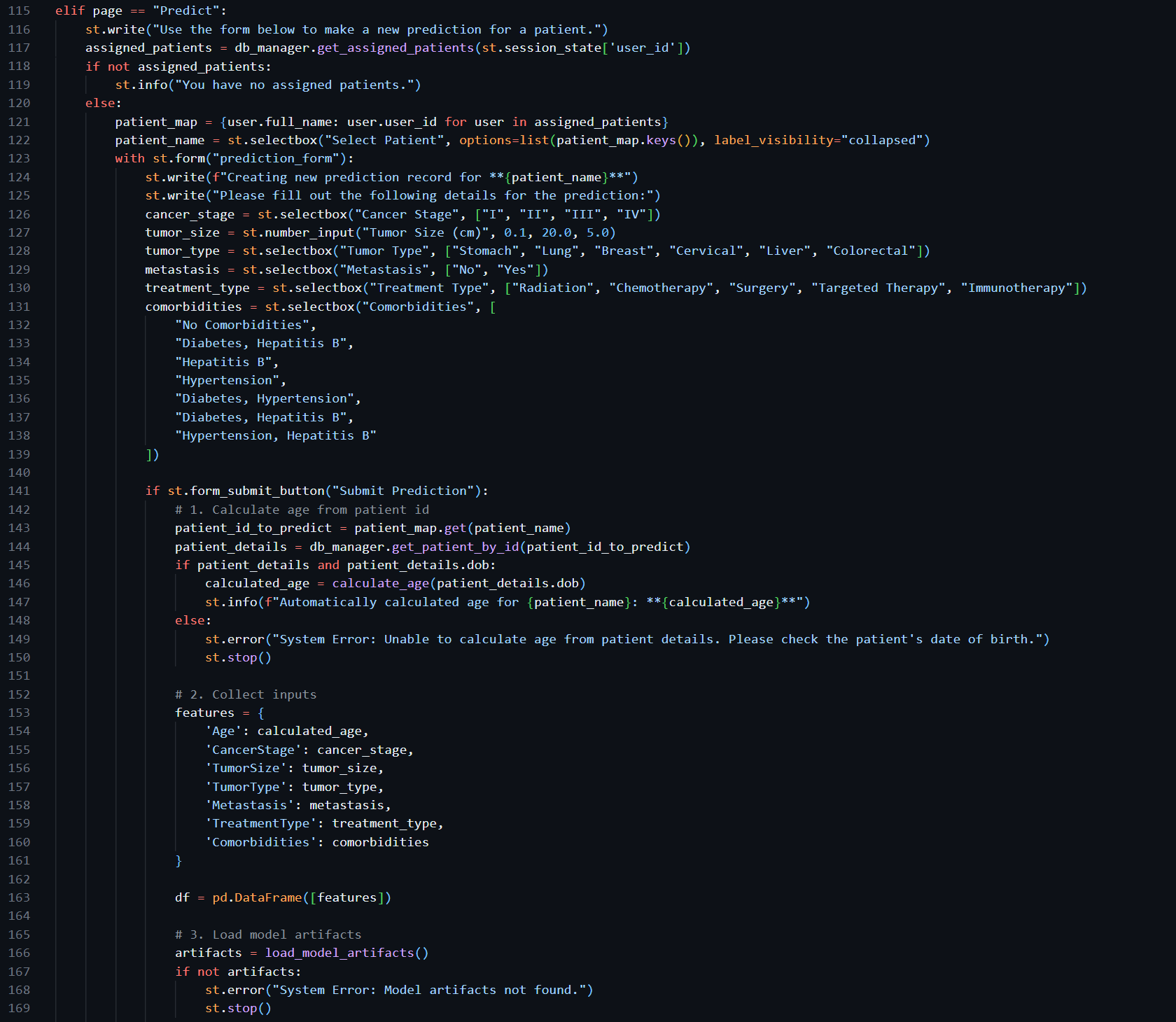


图 39 医生预测页面代码 1

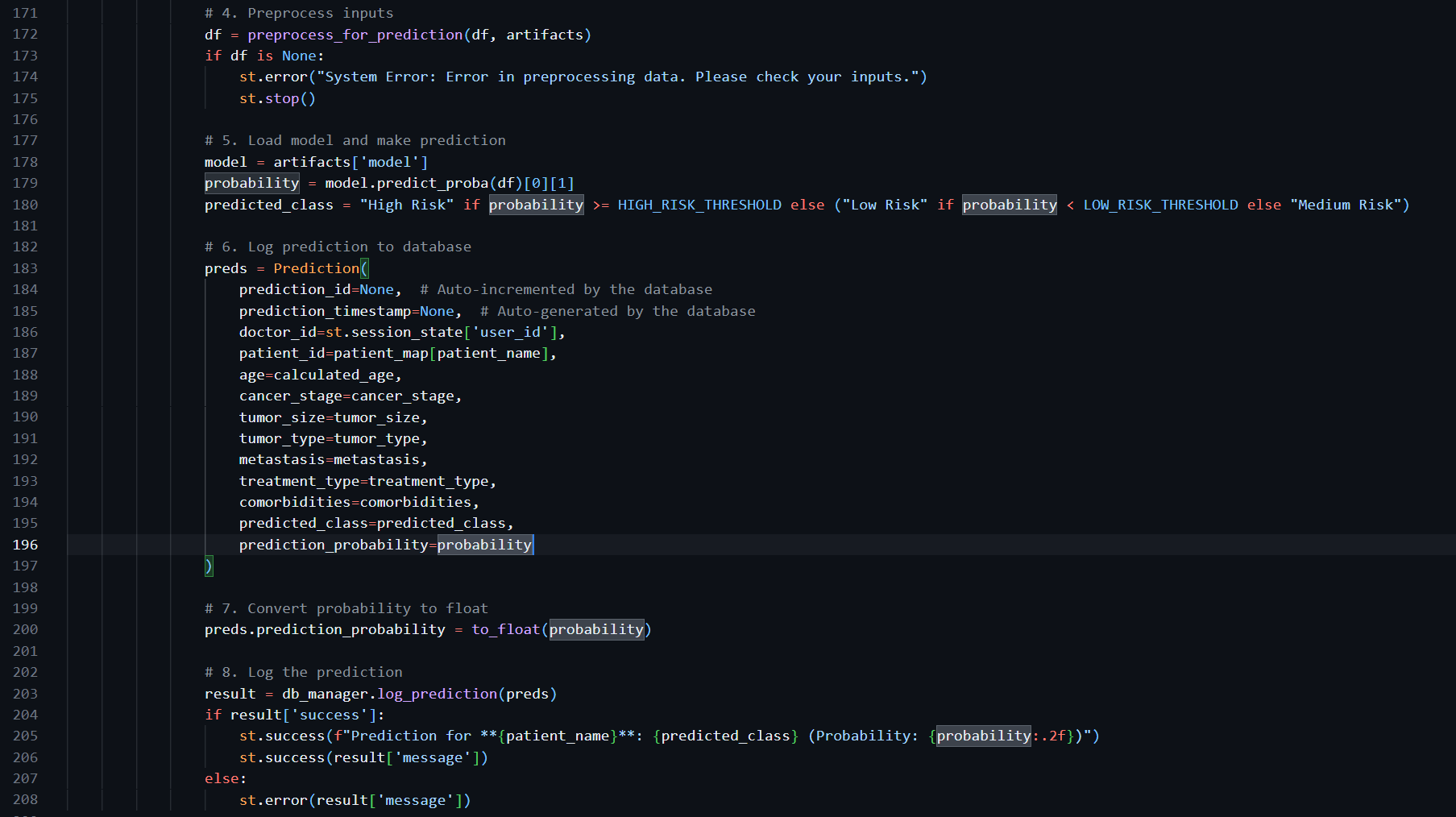


图 40 医生预测页面代码 2

1. ‘Doctor’的‘Patient Request’界面。在此，医生可以处理患者对他发来的关联请求，即可以批准或拒绝该请求。

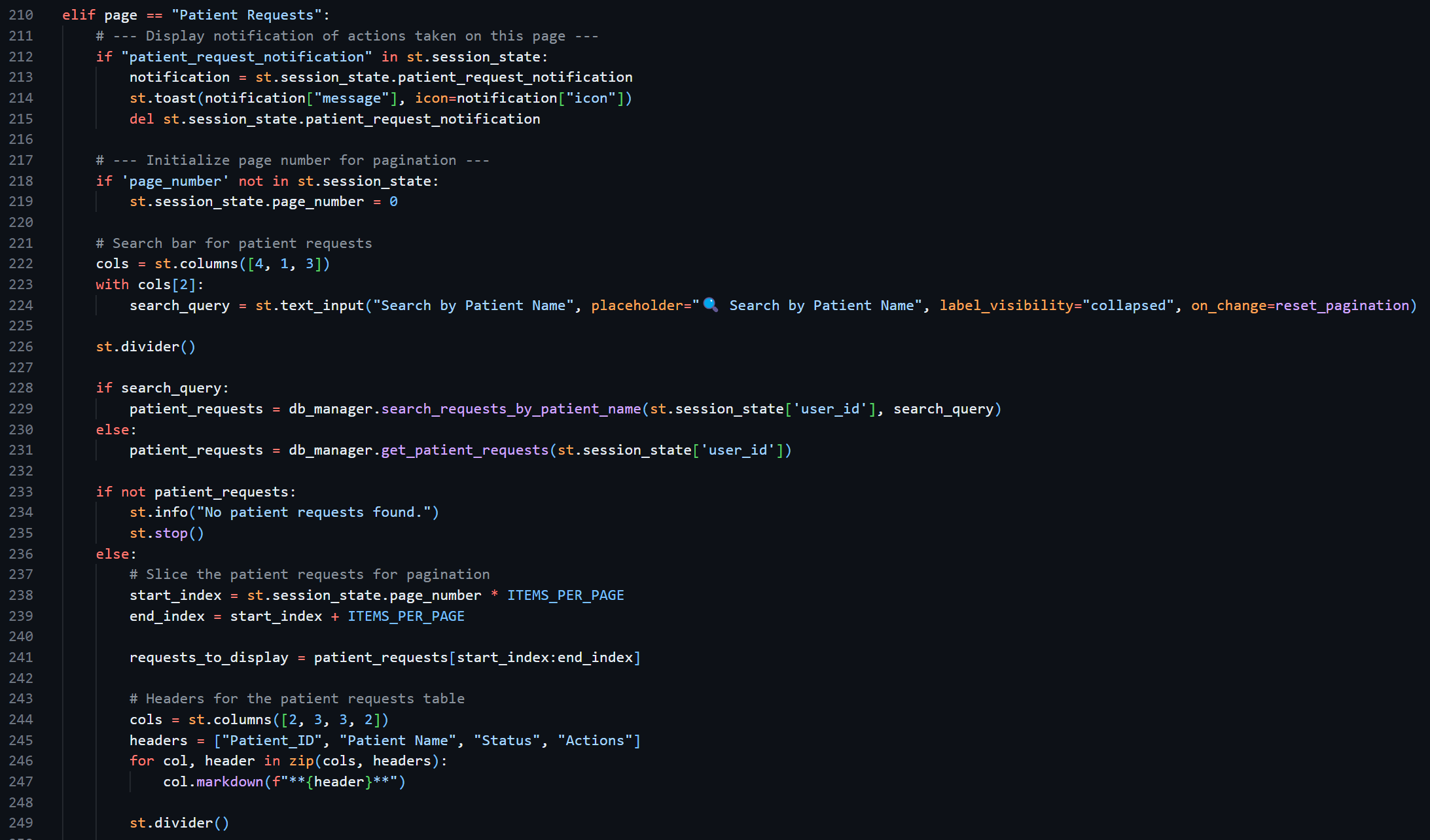


图 41 医生处理患者关联请求的页面代码 1



图 42 医生处理患者关联请求的页面代码 2

1. ‘Patient’查看自己历史预测记录的页面。

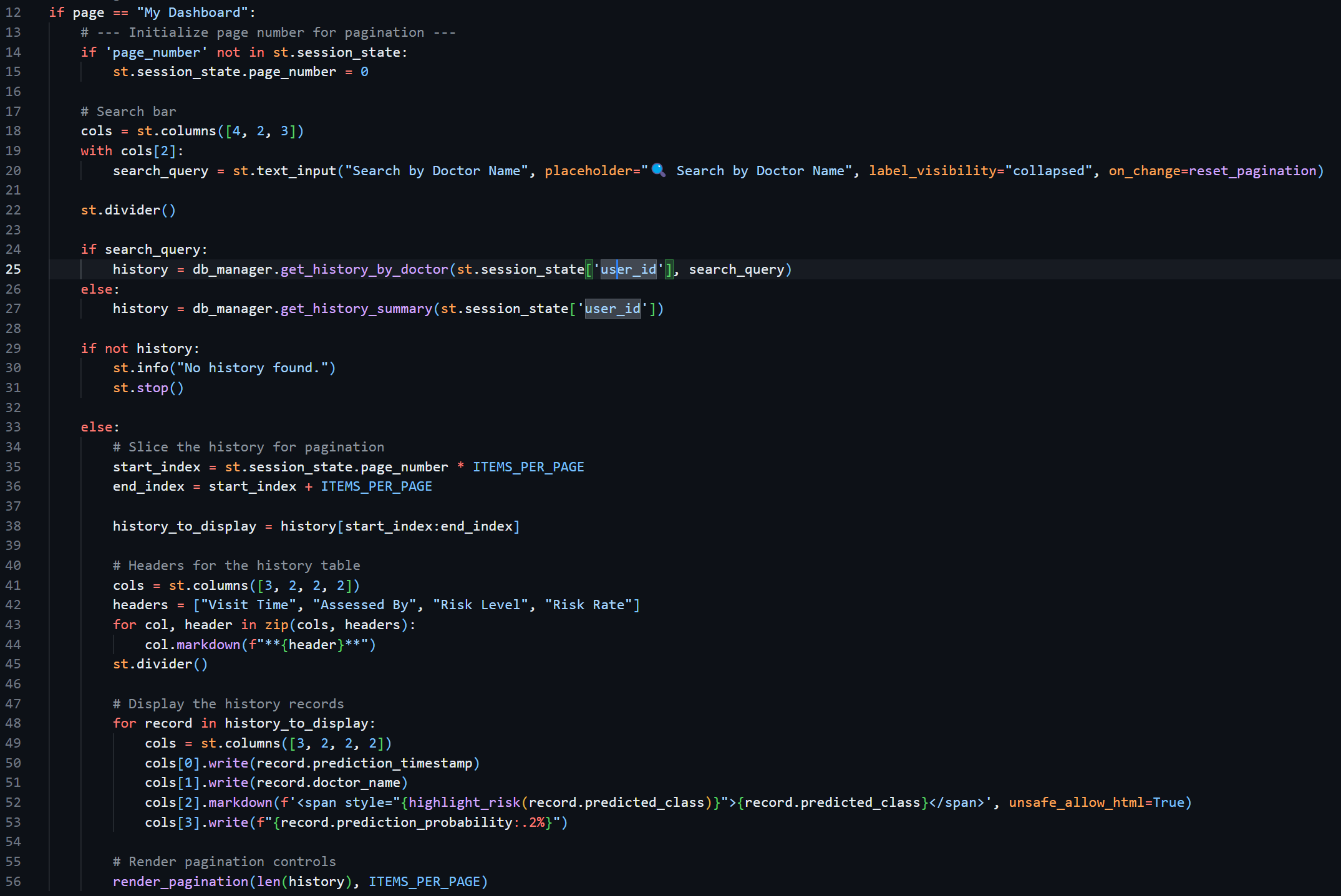


图 43 患者查看自己历史预测记录页面代码

1. ‘Patient’查看当前未关联的医生，同时也可以对他们发出关联请求的页面。

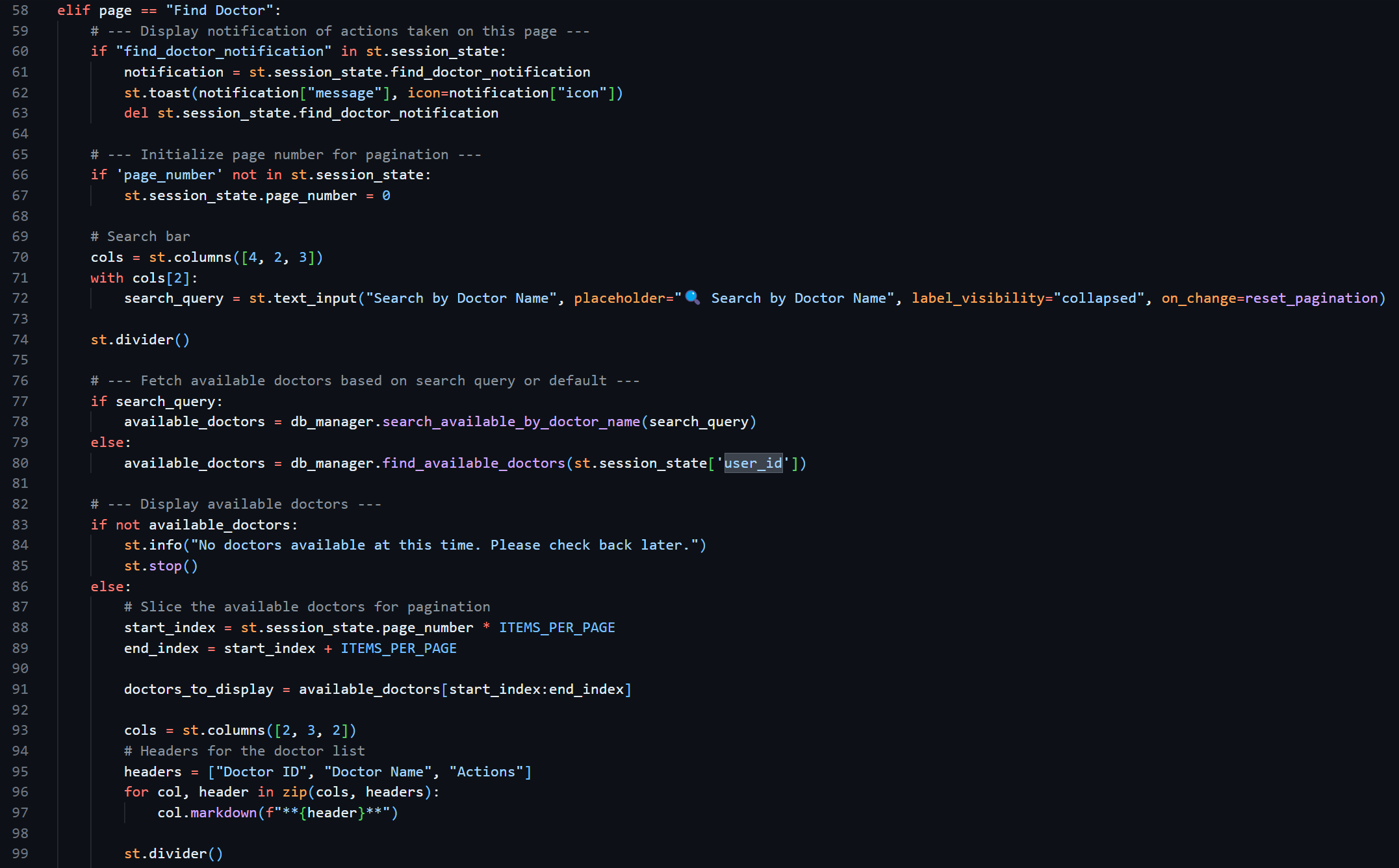


图 44 患者对医生发出关联请求的页面代码 1

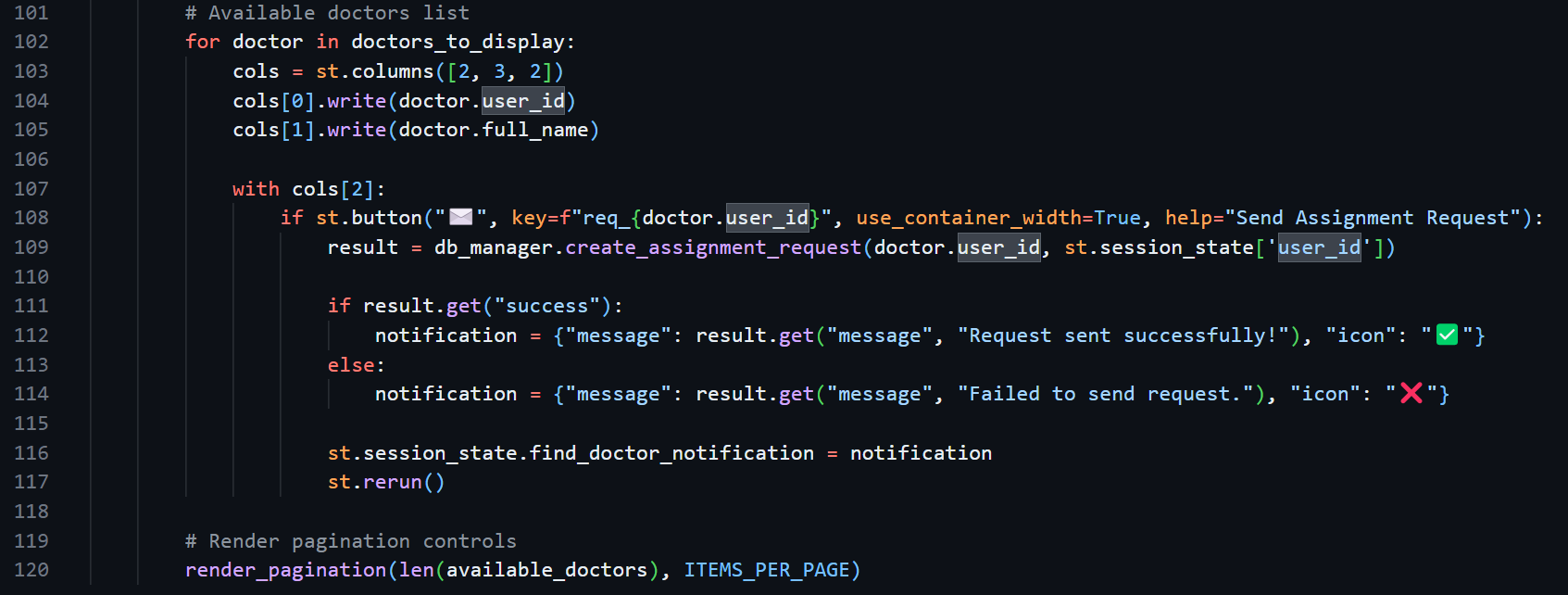


图 45 患者对医生发出关联请求的页面代码 2

1. 用户登录界面。在此用户可以提交自己的‘Username’和‘Password’进行登录，同时也可以点击‘Sign Up’按钮跳转到用户注册页面。



图 46 用户登录页面代码 1

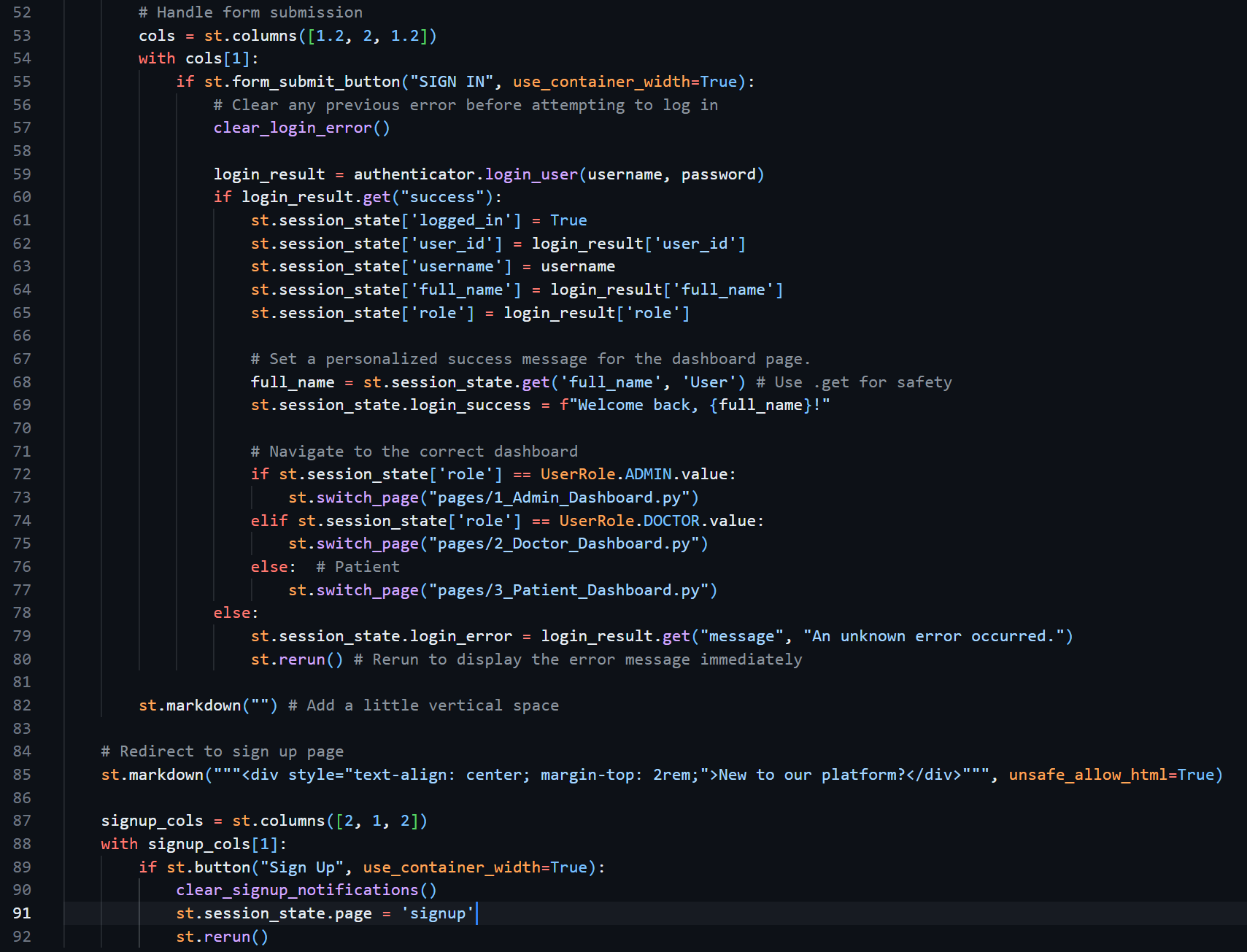


图 47 用户登录页面代码 2

1. 用户注册界面。在此，用户可以提交注册所需要的信息进行注册。如果注册的‘role’为医生，则需要等待被管理员批准。否则，账户直接激活，可以及时登录。用户也可以点击‘Sign In’按钮跳转到用户登录界面。

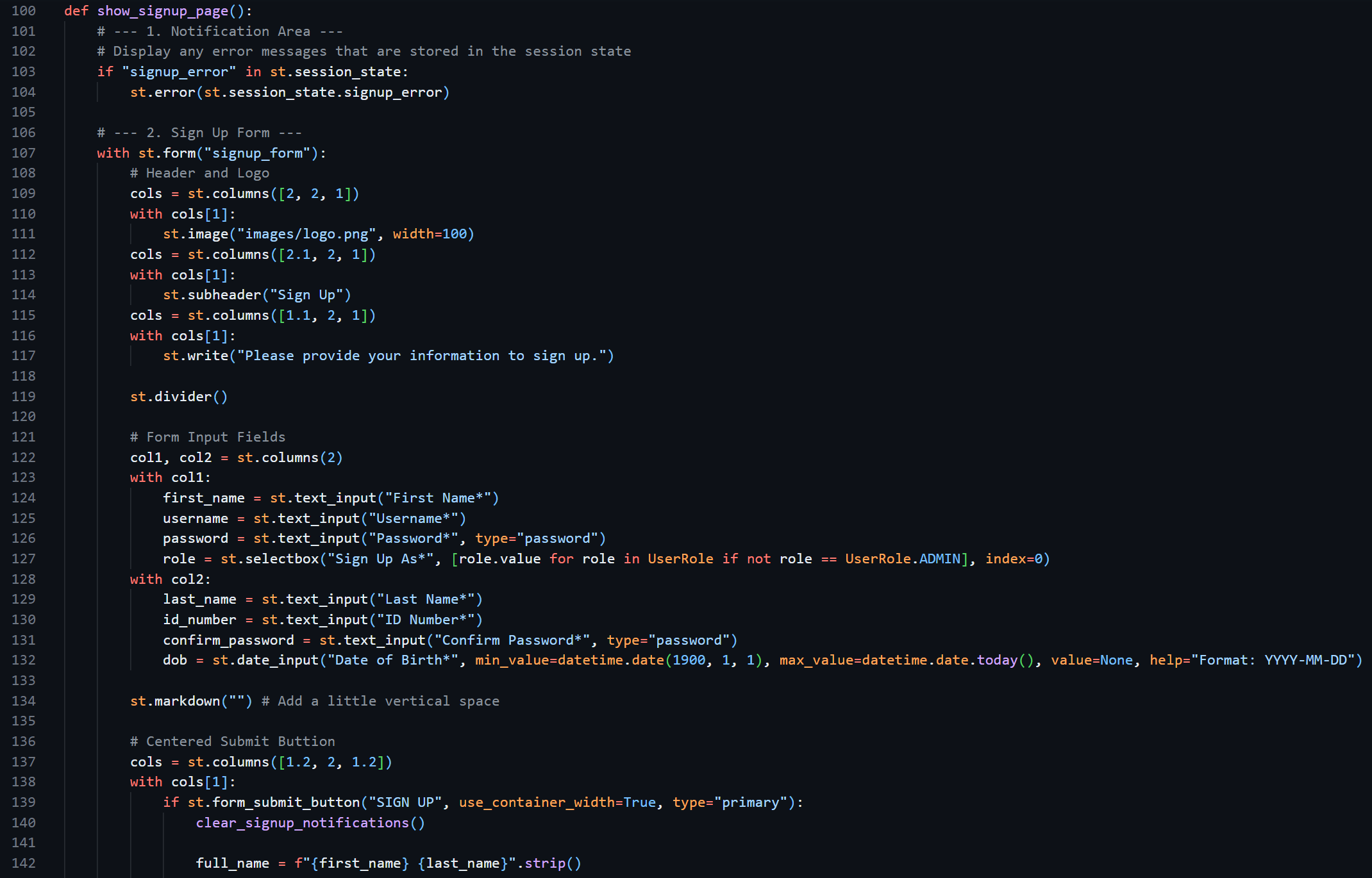


图 48 用户注册界面代码

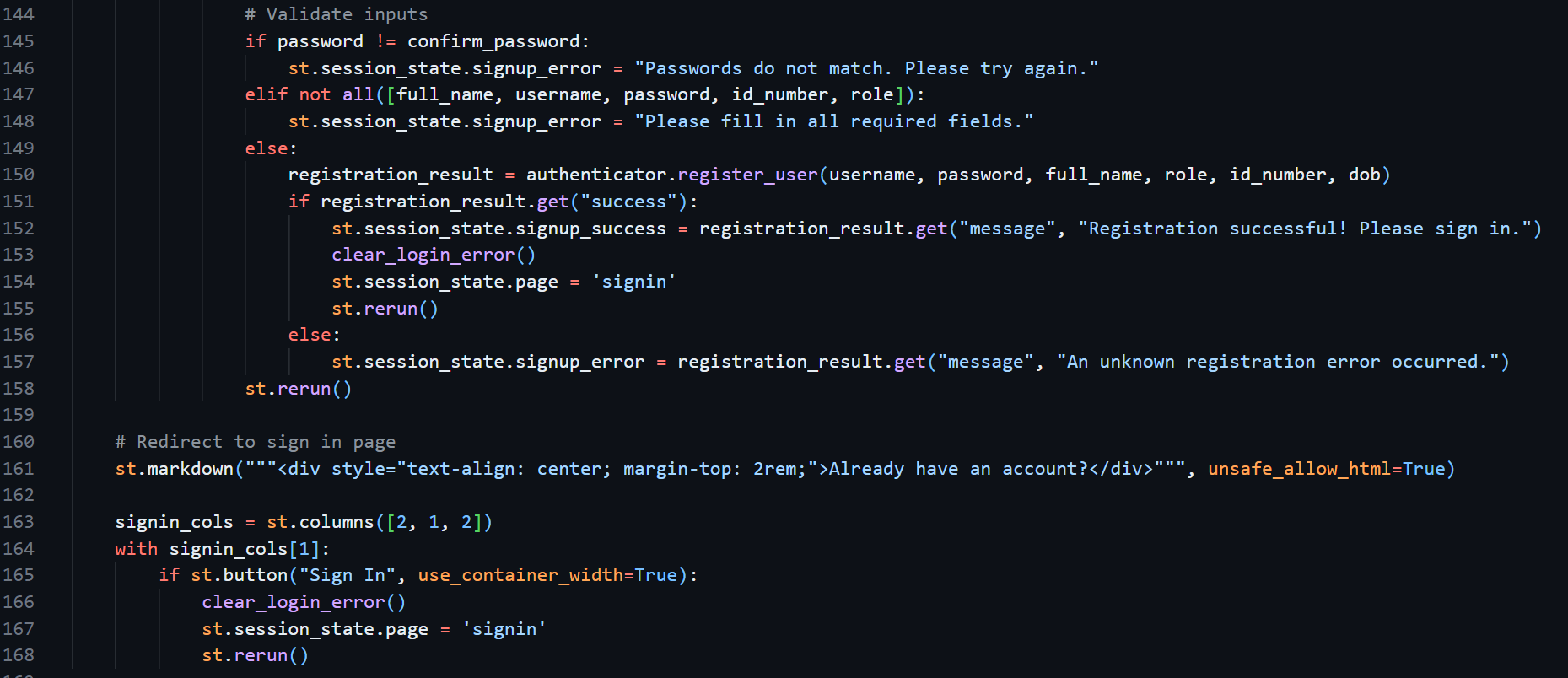


图 49 用户注册界面代码 2

# 系统测试

## 软件功能模块测试

1. 用户注册模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 用户通过输入注册所需要的信息进行账户注册 | |
| 测试员 | Patient Test, Doctor Test | |
| 前驱条件 | 1. Patient Test填写数据库已存在的username。 2. Patient Test 填写的密码和确认密码不一致。 3. Patient Test 填写正确的输入。 4. Doctor Test填写正确的输入。 | |
| 期待结果 | （3）（4）注册成功。（1）（2）显示错误，账户无法注册。 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果 |
| 条件（3）测试结果 | 条件（4）测试结果 |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | 1. Patient Test输入了已存在的username 2. Patient Test输入的密码和确认密码不一致 | |

表 4 用户注册模块测试

1. 用户登录模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 用户拥有自己的账号，通过输入正确的用户名和密码进入系统 | |
| 测试员 | Patient Test, Doctor Test | |
| 前驱条件 | 1. Patient Test输入错误的密码。 2. Patient Test 输入正确的用户名和密码。 3. Doctor Test输入正确的用户名和密码，但是账号注册未被管理员批准。 4. Doctor Test输入正确的用户名和密码，且已被管理员批准。 | |
| 期待结果 | （2）（4）注册成功。（1）（3）显示错误，账户无法登录。 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果 |
| 条件（3）测试结果 | 条件（4）测试结果 |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | （1） Patient Test输入了错误的密码。  （3） Doctor Test注册请求未被管理员批准。 | |

表 5 用户登录模块测试

1. 管理员添加用户功能模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 管理员登录后可以进行添加用户，需要输入用户的完整信息，如图所示。 | |
| 测试员 | Admin | |
| 前驱条件 | 1. Admin输入数据库已存在的username。 2. Admin 输入的密码和确认密码不一致。 3. Admin 填写正确的输入。 | |
| 期待结果 | （3）添加成功。（1）（2）显示错误，无法添加。 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果 |
| 条件（3）测试结果 | |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | 1. Admin输入了已存在的username 2. Admin输入的密码和确认密码不一致 | |

表 6 管理员添加用户功能模块测试

1. 管理员修改用户信息功能模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 管理员登录后可以对用户的信息进行修改，可以修改任何一个信息（不需要修改所有信息）。 | |
| 测试员 | Admin | |
| 前驱条件 | 1. Admin输入数据库已存在的username。 2. Admin 输入的密码和确认密码不一致。 3. Admin 填写正确的输入。 | |
| 期待结果 | （3）添加成功。（1）（2）显示错误，无法添加。 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果 |
| 条件（3）测试结果 （patient\_tester 变为 p\_tester） | |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | 1. Admin输入了已存在的username 2. Admin输入的密码和确认密码不一致 | |

表 7 管理员修改用户信息功能模块测试

1. 管理员删除用户功能模块

|  |  |
| --- | --- |
| 模块功能描述 | 管理员登录后可以对用户进行删除，直接按删除按钮用户就被删除。 |
| 测试员 | Admin |
| 前驱条件 | 1. 无 |
| 期待结果 | （1）成功删除用户。 |
| 实际结果 | 条件（1）测试结果（p\_tester 被删除） |
| 实际结果描述 | 与期待结果一致。 |
| 异常原因 | 无 |

表 8 管理员删除用户功能模块测试

1. 管理员处理医生注册请求功能模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 管理员登录后可以对医生进行批准或拒绝。 | |
| 测试员 | Admin | |
| 前驱条件 | 1. Admin批准d\_five的注册请求 2. Admin拒绝d\_twelve的注册请求 | |
| 期待结果 | * 1. d\_five账户的status变为“active”   2. d\_twelve账户被删除 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果（d\_twelve信息被删除） |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | 1. Admin输入了已存在的username 2. Admin输入的密码和确认密码不一致 | |

表 9 管理员处理医生注册请求功能模块

1. 医生查询用户的预测记录功能模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 医生登录后可以查询与他相联的患者信息，并可以点击“眼睛”按钮进行详细地查看某个用户的预测记录。 | |
| 测试员 | Doctor One | |
| 前驱条件 | 1. 查询所有预测记录（从最新到最久的排序）。 2. 查询某个患者的预测记录。 | |
| 期待结果 | （1）（2）查询成功。 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果 |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | 无 | |

表 10 医生查询用户的预测记录功能模块测试

1. 医生对某个患者进行癌症风险预测功能

|  |  |
| --- | --- |
| 模块功能描述 | 医生登录后可以对与他相联的患者进行癌症风险预测。 |
| 测试员 | Doctor One |
| 前驱条件 | 1. 输入用户的身体情况并点击“预测” |
| 期待结果 | （1）预测成功 |
| 实际结果 | 条件（1）测试结果 |
| 实际结果描述 | 与期待结果一致。 |
| 异常原因 | 无 |

表 11 医生对某个患者进行癌症风险预测功能测试

1. 医生对某个患者关联请求进行处理功能模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 医生登录后可以处理患者发来的关联请求。 | |
| 测试员 | Doctor One，Patient Four，Patient Five | |
| 前驱条件 | 1. 医生批准Patient Four的关联请求。 2. 医生拒绝Patient Five的关联请求。 | |
| 期待结果 | （1）关联成功，医生可以对Patient Four进行查询和预测。  （2）关联失败。 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果 |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | 无 | |

表 12 医生对某个患者关联请求进行处理功能模块测试

1. 患者查询自己的历史预测记录信息功能模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 患者登录后可以查询自己的历史预测记录信息。 | |
| 测试员 | Patient Four, Patient Five | |
| 前驱条件 | 1. Patient Four 查询自己的历史预测记录。 2. Patient Five查询自己的历史预测记录。 | |
| 期待结果 | （1）查询得到。  （2）查询不到。 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果 |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | （2）还没有预测过。 | |

表 13 患者查询自己的历史预测记录信息功能模块测试

1. 患者对某个医生发出关联请求功能模块

|  |  |
| --- | --- |
| 模块功能描述 | 患者登录后可以对某个医生发出关联请求。 |
| 测试员 | Patient Four, Doctor Two |
| 前驱条件 | 1. Patient Four对Doctor Two 发出关联请求。 |
| 期待结果 | （1）发出请求成功。 |
| 实际结果 | 条件（1）测试结果 |
| 实际结果描述 | 与期待结果一致。 |
| 异常原因 | 无 |

表 14 患者对某个医生发出关联请求功能模块

1. 搜索功能模块

|  |  |  |
| --- | --- | --- |
| 模块功能描述 | 用户可以使用搜索功能进行搜索想要查询的信息。 | |
| 测试员 | Admin，Doctor One，Patient Three | |
| 前驱条件 | 1. Admin按照用户名来搜索用户信息。 2. Doctor One按照患者姓名查找患者历史预测记录。 3. Patient Three按照医生姓名查找自己的历史预测记录。 | |
| 期待结果 | 1. 搜索到包含搜索符的用户名的用户信息。 2. 搜索到包含搜索符的患者姓名的患者信息。 3. 搜索到包含搜索符的医生姓名的医生信息。 | |
| 实际结果 | 条件（1）测试结果 | 条件（2）测试结果 |
| 条件（3）测试结果 | |
| 实际结果描述 | 与期待结果一致。 | |
| 异常原因 | 无 | |

表 15 搜索功能模块测试

# 小结

本项目成功构建了一个功能全面、多角色的癌症风险预测系统。该系统利用机器学习模型，为不同用户群体（患者、医生、管理员）提供了安全、独立且高度互动的操作界面。在开发过程中，我们不仅实现了核心业务逻辑，还采用了一系列先进的软件工程实践和Streamlit开发模式，确保了应用的稳定性、可维护性和优良的用户体验。

## 核心技术与系统架构

本项目基于Python和Streamlit框架构建，实现了清晰、可扩展的系统架构：

1. 多页面应用：利用Streamlit的pages/目录结构，为不同用户角色创建了独立的、受保护的仪表盘页面，实现了应用的模块化。
2. 基于角色的访问控制: 设计并实现了一套完整的认证与授权系统。用户登录后，系统会根据其角色（Admin, Doctor, Patient）动态渲染侧边栏导航并严格限制页面访问权限。
3. 组件化UI设计: 将重复的UI元素，如侧边栏 (render\_sidebar\_and\_auth) 和 分页控件 (render\_pagination)，抽象成可重用的函数，并集中在ui\_components.py文件中管理，极大地提高了代码的复用性和可维护性。
4. 前后端交互模式: 确立了一套标准化的后端（DatabaseManager）与前端（Streamlit页面）的交互模式。

## 主要功能模块实现 (Implementation of Main Functional Modules)

### 管理员后台 (Admin Dashboard):

1. 用户管理: 实现了对系统中所有用户的完整CRUD（创建、读取、更新、删除）操作。界面包含用户列表、搜索功能和健壮的分页。
2. 医生注册审批: 为管理员提供了一个专门的界面，用于审核新注册医生的请求，可以一键批准或拒绝，并提供清晰的操作反馈。
3. 动态表单: “添加用户”和“编辑用户”的表单会动态地出现在主列表页面上，提供了无缝的操作体验。

### 医生工作台 (Doctor Dashboard):

1. 主看板与患者历史: 医生可以查看所有已关联患者的最新预测记录列表。通过点击“查看详情”，可以进入单个患者的详细历史页面，该页面包含一个风险概率随时间变化的趋势图 (st.line\_chart) 和详细的历次预测数据表格。
2. 癌症风险预测: 一个核心功能页面，医生可以选择一位已关联的患者，并填写一个包含多项医学指标的表单。提交后，系统会加载预训练的机器学习模型，对输入数据进行预处理和预测，并立即将预测结果（风险等级和概率）存入数据库。
3. 患者关联请求 (Patient Requests): 医生可以查看并处理来自患者的关联请求，实现对医患关系的自主管理。

### 患者工作台:

1. 个人历史记录: 患者登录后可以查看自己所有的历史预测记录，了解自己的健康风险趋势。
2. 查找并请求关联医生: 提供了一个可搜索、可分页的医生列表。患者可以向列表中的医生发送关联请求，以便后续由该医生为其进行风险预测。

# 参考文献

Carlos, C. T. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794). ACM.

Chawla, N. V. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 321–357.

Developer, S. (2025, 5 31). *Datatypes In SQLite*. From SQLite: https://sqlite.org/datatype3.html

Developer, S. (n.d.). *Streamlit API reference*. From Streamlit: https://docs.streamlit.io/develop/api-reference

developers, x. (n.d.). *XGBoost Parameter*. From XGBoost Docs: https://xgboost.readthedocs.io/en/release\_3.0.0/parameter.html#parameters-for-tree-booster

*SMOTE for Imbalanced Classification with Python*. (2024, 5 23). From GeeksforGeeks: https://www.geeksforgeeks.org/machine-learning/smote-for-imbalanced-classification-with-python/

# 附录

代码上传至 Github 仓库：