

郑州大学西亚斯国际学院

本科毕业论文（设计）

题 目 在线医疗平台的设计与实现

——患者模块

指导教师 邵彧 职称 副教授

学生姓名 乔梦洋 学号 2019325140239

专 业 软件工程（专升本）

班 级 2019 软件工程专升本 2 班

院（系） 电子信息工程学院

完成时间 2021 年 4 月 2 日

在线医疗平台的设计与实现——患者模块

摘 要

随着经济的迅速发展，人们对生活水平和身体健康的要求越来越高，但同时也面临着优质医疗资源紧缺，看病难，看病贵，医患关系危机等各种各样的问题。如何使用互联网技术解决当前医疗系统中存在的问题，提高效率，成为研究的热点。针对这种现状，本文设计并且实现了一个在线医疗系统，系统以在线挂号问诊为核心，设计并实现了人员管理，在线挂号排队、查询诊断结果，查询医生的一切信息等模块，为各种医疗场景提供了一套较完整的解决方案。在技术架构上，本系统业务主要基于 PyCharm 与 MySQL 实现，并针对一些特殊需求，使用了合适的技术进行实现。利用前端框架 Bootstrap，对不同尺寸的设备进行适配并测试，满足了系统要求医生和病人能够在 PC 端和移动设备都能操作的需求。对于使用互联网技术改变当前医疗资源紧缺，看病难，效率低下等现状是一个良好的尝试。

关键词 在线医疗，PyCharm，MySQL。

Design and implementation of online medical platform -- patient module

ABSTRACT

With the rapid development of economy, people's requirements for living standards and health are getting higher and higher, but at the same time, they are also faced with a variety of problems, such as the shortage of quality medical resources, the difficulty and cost of seeing a doctor, the crisis of doctor-patient relationship and so on. How to use Internet technology to solve the problems existing in the current medical system and improve the efficiency has become a research hotspot. In view of this situation, this paper designs and implements an online medical system. The system takes online registration and consultation as the core, designs and implements modules such as personnel management, online registration queuing, inquiring diagnosis results, inquiring all information of doctors, etc., which provides a set of complete solutions for various medical scenarios. In terms of technical architecture, the business of this system is mainly realized based on PyCharm and MySQL and appropriate technologies are used to realize it according to some special requirements. The front-end framework Bootstrap was used to adapt and test devices of different sizes, which met the requirements of the system that doctors and patients could operate both on PC and mobile devices. It is a good attempt to use Internet technology to change the current situation such as shortage of medical resources, difficulty in seeing a doctor and low efficiency.

KEY WORDS The online medical, PyCharm, MySQL.

目 录

中文摘要.....	
英文摘要.....	
1 绪论.....	1
1.1 开发背景.....	1
1.2 课题研究的意义.....	1
1.3 论文内容的介绍.....	2
2 系统相关技术介绍.....	3
2.1 B/S 架构.....	3
2.1.1 B/S 架构概述.....	3
2.1.2 B/S 架构与 C/S 架构比较.....	3
2.2 Python 简介.....	4
2.3 PyCharm 简介.....	4
2.4 Django 简介.....	4
2.5 MySQL 简介.....	5
2.6 jQuery 简介.....	6
2.7 Bootstrap 简介.....	6
3 系统需求分析和整体设计.....	7
3.1 系统可行性研究.....	7
3.1.1 技术可行性.....	7
3.1.2 经济可行性.....	7
3.1.3 操作可行性.....	7
3.1.4 小结.....	8
3.2 系统需求分析.....	8
3.3 系统总体功能设计.....	8
3.4 系统总体数据库设计.....	9
4 患者模块的详细设计与实现.....	11
4.1 用户管理功能的实现.....	11
4.1.1 用户注册.....	11
4.1.2 用户登录.....	11
4.2 找专家功能的实现.....	12
4.3 在线挂号功能的实现.....	13
4.3 查询诊断结果功能的实现.....	15

4.5 查询医生信息功能的实现.....	15
5 模块和系统测试.....	17
5.1 用户登录功能测试.....	17
5.2 用户注册功能测试.....	17
5.3 挂号测试.....	18
5.4 总结.....	19
结 论.....	20
致 谢.....	21
参考文献.....	22
附 录.....	23

1 绪论

1.1 开发背景

随着人民生活水平的日益提高和居民健康意识的不断提升，人们对医疗资源的需求日益增长。根据 2017 年《中国卫生和计划生育统计年鉴》和《2017 年我国卫生健康事业发展统计公报》，2017 年我国医院诊疗人次从 2010 年的 20.40 亿人次增长到了 34.40 亿人次，同比增长了 68.63%，然而我国医疗资源面临着严重紧缺的问题。医疗资源的严重紧缺和分配不均，导致我国的医疗面临着严峻的问题^[3]。在我国，医生在工作期间日均担负诊疗为 34 人次，患者通常为了咨询医生几分钟而需要等待数小时的时间，这就导致了患者和医生之间缺乏有效的沟通交流^[2]。医生准确的诊断和有效的治疗需要建立在医生与患者彼此信任和双方信息透明的基础之上，并且，紧张的医患关系会阻碍医院和医生提供高质量医疗服务。因此，如何提升医生和患者之间的协作护理、改善紧张的医患关系和提升患者的健康福祉是一个亟待解决的问题。在互联网医疗中，社交媒体在医疗领域中的应用，在线医疗平台的实现有助于提升和改善医生与患者之间沟通交互的方式，并且越来越受大众的欢迎。在线医疗将医生和患者之间的沟通交流延伸到了互联网环境中，使患者不再局限于去实体医院咨询看病，可以让患者通过在线的方式不受约束地与医生进行沟通交流，从而得到更多便利和优质的医疗服务。

1.2 课题研究的意义

近几年随着互联网的发展，各行各业信息化的空前普及，传统的医疗领域也开始跨入了互联网时代，新兴的在线问诊服务可以方便、快捷的为患者提供优质的医疗服务，2019 年，新冠疫情催化下在线医疗发展再次加速，我国在线医疗行业迎来了“春天”^[1]，居民对于在线医疗的需求呈爆发式增长，然而对于现阶段的医院行业，高度的信息化管理只存在于大型的医院当中，对于中小型医院信息的普及化还不能全面覆盖，医生与患者之间也不能畅所欲言，针对与医生与患者之间也不能畅所欲言这一问题本平台通过及时咨询、在线医患沟通平台、医师从业人员以及民众之间，建立起了基于 Internet 的有效的沟通渠道，为医生与患者搭起了一个很好的沟通桥梁。该在线医疗平台患者不仅可以在线挂号排队、查询诊断结果，而且可以查询医生的一切信息

以及上班时间等。在线医疗解决了疫情期间医护资源不足、问诊需求过高、二次感染等重要问题，同时也促进了在线医疗的广泛应用，成为居民看病问诊常态模式，加强了人们对在线医疗的了解和认可，建立对在线医疗的信任。在线医疗得到广泛应用后，我国最偏远的乡村也可以得到优质的诊疗资源，医生与患者之间也可以无缝隙的交流与沟通，更有利于医生为患者更精确的治疗，为进一步加强医疗信息的科学性、权威性、指导性、实用性、服务性、及时性，更好地为患者提供更全面更专业的优质健康服务。

1.3 论文内容的介绍

本文首先介绍了课题研究的目的及意义。其次，讲述系统所涉及的基础理论知识，以及使用到相关技术的介绍。然后对系统进行需求分析，可行性分析以及系统功能和数据库的设计。接着是分别介绍各个功能的实现，最后对系统进行测试。

2 系统相关技术介绍

2.1 B/S 架构

2.1.1 B/S 架构概述

B/S 架构即浏览器和服务器架构模式，是随着 Internet 技术的兴起，对 C/S 架构的一种变化或者改进的架构。在这种架构下，用户工作界面是通过浏览器来实现，极少部分事务逻辑在前端实现，但是主要事务逻辑在服务器端实现。这种模式统一了客户端，将系统功能实现的核心部分集中到服务器上，简化了系统的开发、维护和使用。客户机上只要安装一个浏览器，服务器安装一个数据库。浏览器通过 Web Server 同数据库进行数据交互。这样就大大简化了客户端电脑载荷，减轻了系统维护与升级的成本和工作量，降低了用户的总体成本。

2.1.2 B/S 架构与 C/S 架构比较

C/S 架构的优点是 C/S 客户端响应速度快，可以足够表现客户端 PC 的处置才能，很多工作能够在客户端处置以后再提交给服务器；操作界面漂亮、形式多样，能够足够满足客户自己的个性化要求；C/S 结构的管理信息系统拥有比较强的事务处置才能，可以完成复杂的业务过程；安全性能能够非常容易确保，C/S 通常面向相比固定的用户群，程序越发注重过程，它能够对权限实行多层次校验，提供了更安全的存取形式，对信息安全的控制才能非常强。缺点是需要在专门的客户端安装程序，分布功能弱，针对点多面广且不具备网络条件的用户群体，不可以完成迅速部署安装与配置；兼容性差，关于不一样的开发工具，拥有比较大的局限性；开发、维护费用较高，须要拥有肯定专业水准的技术人员才可以结束，发生一次升级，就全部客户端的程序全部须要更改；用户群固定，程序安装就可使用，这样不符合面向一些不可知的用户，实用面窄，常用来局域网中。

B/S 架构的优点是分布性强，客户端零维护。只需有网络、浏览器，能够随时随地实行查询、浏览等业务处理；业务扩展简单便利，通过添加网页就可以添加服务器功能；维护简单便利，只须要更改网页，就可以完成全部用户的同步更新；开发简单，共享性强。缺点是个性化特征明显减少，没办法完成拥有个性化的功能要求；在跨浏览器上，B/S 架构不尽如人意；没办法完成分页显示，给数据库访问导致较大的压力，

客户端服务器端的交互就是请求-响应形式，常常动态刷新页面，响应速度明显减少；在速度与安全性上须要花费超大的设计费用；功能弱化，难以完成传统形式下的特殊功能需要。

由此可知，虽然 C/S 响应速度快，安全性强，通常应用在局域网当中，可是开发维护费用高，但是 B/S 能够完成跨平台，客户端零维护，可是个性化才能低，响应速度较慢。所以本平台的开发使用 B/S 架构。

2.2 Python 简介

Python 是 1990 年代初由荷兰数学和计算机科学研究学会的 Guido van Rossum 设计的，是一种解释型、面向对象、动态数据类型的高级程序设计语言，它以简单明了、快速上手、功能强大等特点而著称^[5]。它不仅继承了传统语言的功能和实用性，而且还借鉴了简单的语言脚本。Python 提供了高效的高级数据结构，还能简单有效地面向对象编程。

作为一种编程语言，Python 在设计上始终贯彻简单明了、清晰划一的风格，使 Python 成为一门简单易读且易维护的语言，因此深受广大程序员欢迎的，并在实际应用中得到广泛使用。

2.3 PyCharm 简介

PyCharm 是一种 Python IDE，带有一整套可以帮助用户在使用 Python 语言开发时提高其效率的工具，PyCharm 拥有一般 IDE 具备的功能，比如调试、语法高亮、Project 管理、代码跳转、智能提示、自动完成、单元测试、版本控制。此外，该 IDE 提供了一些高级功能，以便用于支持 Django 框架下的专业 Web 开发。PyCharm 有以下几个特点：

(1) PyCharm 提供智能代码补全、代码检查、实时错误高亮显示和快速修复，以及自动化代码重构和丰富的导航功能。

(2) PyCharm 为现代 Web 开发框架提供丰富的框架针对性支持。

(3) PyCharm 与 IPYT、Jupyter Notebook 集成，提供交互式 Python 控制台，并且支持 Anaconda 和多种科学化的包。

2.4 Django 简介

Django 是一个 Web 应用程序框架的免费开放源代码，是用 Python 写成的^[8]。在众多的 Web 开发框架中，Django 是高水准的 Python 编程语言驱动的一个开源模型。使用 Django 架构，程序员可以方便、快捷地创建高品质、易维护的数据库驱动的应用程序。另外，在 Django 框架中，还包含许多功能强大的第三方插件，使得 Django 具有较强的可扩展性^[4]。

Django 采用 MTV（Model，Template，View）的设计，就是把 Web 应用分为数据模型（存取层）、模板（表现层）、视图（业务逻辑层）三层。其中，模型的职责是处理与数据相关的一切事务；视图负责处理业务逻辑；模板的职责是将前端的页面展示给用户。

2.5 MySQL 简介

就目前来看，MySQL 是最流行的关系型数据库管理系统之一，在中小型 Web 开发应用方面，是最好的 RDBMS 应用软件^[11]。MySQL 作为一种关系数据库管理系统，关系数据库会将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性^[12]。

MySQL 所使用的 SQL 语言是用于访问数据库的最常用标准化语言。MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。与其他的大型数据库例如 Oracle、DB2、SQL Server 等相比，MySQL 自有它的不足之处，但是这丝毫也没有减少它受欢迎的程度。对于一般的个人使用者和中小型企业来说，MySQL 提供的功能已经绰绰有余，而且由于 MySQL 是开放源码软件，因此可以大大降低总体拥有成本。目前每年都会有很多的用户从 MySQL AB 的官方网站下载 MySQL 的安装程序，作为一种领先的数据库，MySQL 有很多的优点，下面就做一下简单的介绍：

（1）MySQL 是开放源代码的数据库

MySQL 是开放源代码的数据库，任何人都可以获取该数据库的源代码，这样就意味着任何人都可以修正 MySQL 的缺陷，同时任何人可以以任何的目的来使用该数据库，MySQL 作为数据库完全继承了 Gun 的思想。这样就保证了 MySQL 是款可以自由使用的数据库。

(2) MySQL 的跨平台性

MySQL 不仅可以在 windows 系统上进行操作，它还可以在其它的一些操作系统上运行，所以 MySQL 的跨平台性让其在 web 的应用方面有很多的优点。虽然微软公司的 SQL Server 数据库是一款很优秀的数据库，但是这种数据库却不能跨平台操作运行，所以，MySQL 的跨平台性是一个很大的优势。

(3) 价格优势

MySQL 数据库是一款自由软件，任何人都可以到 MySQL 的官方网站去下载使用，而且这些软件都是免费的。即使是需要付费的一些附加功能，它的价格也是比较低的，相对于一些其它的费用比较高的数据库，MySQL 是具有绝对的优势的。

(4) 使用方便且功能强大

MySQL 是一个真正的多线程、多用户的数据库服务器。它是以客户机/服务器结构的实现，有一个服务器保护程序 MySQL 和很多不同的客户程序和库组成。它可以有效、快速、安全的处理大量的数据。相对一些其他的数据库，MySQL 的使用是非常的简单的，MySQL 主要的目标就是易用、健壮和快速

2.6 jQuery 简介

jQuery 是一个快速、简洁的 JavaScript 框架，是继 Prototype 之后又一个优秀的 JavaScript 代码库。jQuery 倡导写更少的代码，做更多的事情。它封装 JavaScript 常用的功能代码，提供一种简便的 JavaScript 设计模式，优化 HTML 文档操作、事件处理、动画设计和 Ajax 交互。jQuery 的核心特性可以总结为：具有独特的链式语法和短小清晰的多功能接口；具有高效灵活的 CSS 选择器，并且可对 CSS 选择器进行扩展；拥有便捷的插件扩展机制和丰富的插件。

2.7 Bootstrap 简介

Bootstrap 是美国 Twitter 公司的设计师 Mark Otto 和 Jacob Thornton 合作基于 HTML、CSS、JavaScript 开发的简洁、直观、强悍的前端开发框架，使得 Web 开发更加快捷。Bootstrap 提供了优雅的 HTML 和 CSS 规范，它即是由动态 CSS 语言 Less 写成。Bootstrap 一经推出后颇受欢迎，一直是 GitHub 上的热门开源项目，包括 NASA 的 MSNBC 的 Breaking News 都使用了该项目。国内一些移动开发者较为熟悉的框架，如 WeX5 前端开源框架等，也是基于 Bootstrap 源码进行性能优化而来。

3 系统需求分析和整体设计

3.1 系统可行性研究

可行性研究的目的是不是解决问题，而是确定问题是否值得去解决。下面便从技术可行性、经济可行性和操作可行性三方面分别研究^[14]。

3.1.1 技术可行性

该系统的开发环境和配置都是可以自行安装的，本平台使用 Python 语言开发，使用比较成熟的 MySQL 数据库进行对系统前台及后台的数据交互，根据技术语言对数据库，结合需求进行修改维护，可以使得系统运行更具有稳定性和安全性，从而完成实现系统的开发。

（1）硬件可行性分析

系统的硬件要求方面不存在特殊的要求，只需要在普通的硬件配置就能够轻松的实现，只是需要确保系统的正常工作即可，以及拥有较高的效率。如果有特别低的硬件，它可以导致系统的低性能以及效率低，从而导致整个网站的运行不顺畅。以目前普遍的个人计算机的配置而言，这是十分容易实现的。因此，本系统的开发在硬件方面是可行的。

（2）软件可行性分析

本系统将使用 PyCharm 进行开发，使用 MySQL 进行前后台的交互。因此，考虑到系统的实际情况，选择 Python 作为系统开发技术，通过以上分析，系统的设计和实现在软件中是可行的。

综上所述，我们从两个方面进行了可行性研究，可以看出系统的开发没有问题。

3.1.2 经济可行性

系统是基于 Python 语言开发的软件，采用 PyCharm 平台以及 MySQL 小型数据库，均属于开源免费产品使用，对现在的开发成本以及维护成本上来说，是比较低廉的。

3.1.3 操作可行性

本系统采用 Python 技术，利用网络就能够进行访问和操作，且界面简单易操作，用户只要平时有在用电脑，都能进行访问和操作。本系统具有易操作、易管理、交互

性好的特点，在操作上是非常简单的。因此本系统可以进行开发。系统的开发流程遵循软件开发思想，分阶段完成系统分析、设计、实现等研究内容网，系统界面友好，使用者不需要经过专门的培训即可熟练操作

综上所述，此系统开发目标已明确，在技术和经济等方面都可行，并且投入少、见效快。因此系统的开发是完全可行的。

3.1.4 小结

通过对以上对系统的经济、技术和运行方面的可行性分析，最终发现本系统的技术相当成熟，有友好的界面、操作简单、运行安全可靠。

3.2 系统需求分析

“在线医疗平台的设计与实现——患者模块”是针对于患者与医生可以在线交流设计的。该系统采用 B/S 模式，后台的数据库采用目前比较流行的 MySQL，该数据库系统在安全性、准确性、运行速度方面有绝对的优势，并且处理数据量大，效率高；前台采用 PyCharm 作为主要的开发工具，可实现与 MySQL 数据库的无缝连接。

3.3 系统总体功能设计

本在线医疗平台主要实现了患者注册登录后进行在线挂号、在线问诊、查询诊断结果等功能。

其结构功能图如图 3-3 所示：

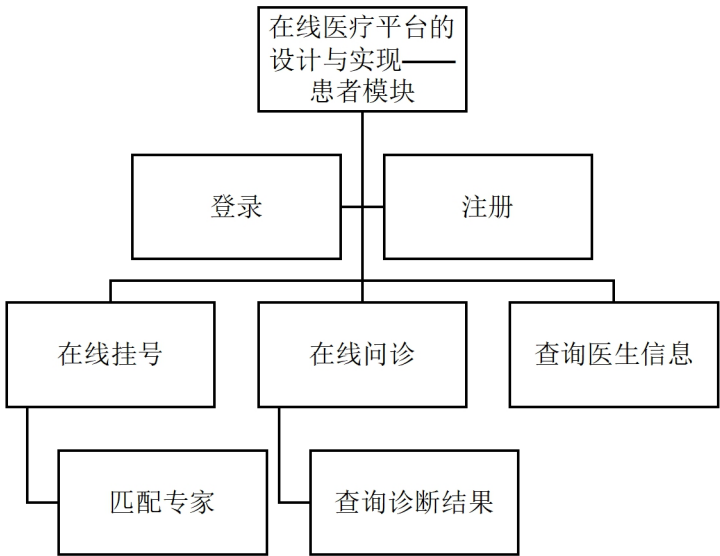


图 3-3 结构功能图

3.4 系统总体数据库设计

数据库在任何开发的过程中都具有极其重要的地位，数据库的好坏直接关乎于整个系统运行的效率，所以对于数据库的选择也一定要仔细。数据库设计的初步阶段根据该系统的功能模块以及数据信息的属性相联系进行初步的规划。该系统数据库中的信息主要包括用户信息、医生信息和聊天信息三个实体。

通过之前对系统整体的需求分析和总体模块的设计，以及本系统的数据库的详细设计，根据各实体之间的相互联系和各数据表之间的相互管理，得出了本系数据库逻辑设计。

用户账号表如图 3-4 所示：




Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户ID
 u_account	VARCHAR(20)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户账号
 u_password	VARCHAR(20)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户密码
 u_email	VARCHAR(20)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户邮箱

图 3-4 用户账号表

用户挂号信息表如图 3-4 所示：









Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户信息ID
 u_account	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户账号
 u_name	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户姓名
 u_age	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户年龄
 u_gender	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户性别
 d_department	VARCHAR(50)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户选择科室
 u_description	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户描述
 u_phone	VARCHAR(50)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户电话

图 3-4 用户挂号信息表

问诊信息表如图 3-5 所示：













Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	问诊信息ID
 d_department	VARCHAR(50)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	科室
 d_id	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	医生ID
 info_id	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	挂号ID
 is_working	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	是否进行中
 is_done	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	是否结束
 opentime	DATETIME(6)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	开始时间
 is_subsequent_visit	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	是否为复诊
 subsequent_visit_time	DATETIME(6)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	复诊时间
 done_time	DATETIME(6)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	结束时间
 u_id	INT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	用户ID
 ill	VARCHAR(150)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	电子处方

图 3-5 问诊信息表

聊天问诊表如图 3-6 所示：

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	问诊 ID
appointment_id	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	问诊信息 ID
master	VARCHAR(50)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'发送人'
message	VARCHAR(150)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	信息
flag	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	聊天锚点

图 3-6 聊天问诊表

数据库逻辑设计 E-R 图如图 3-7 所示：

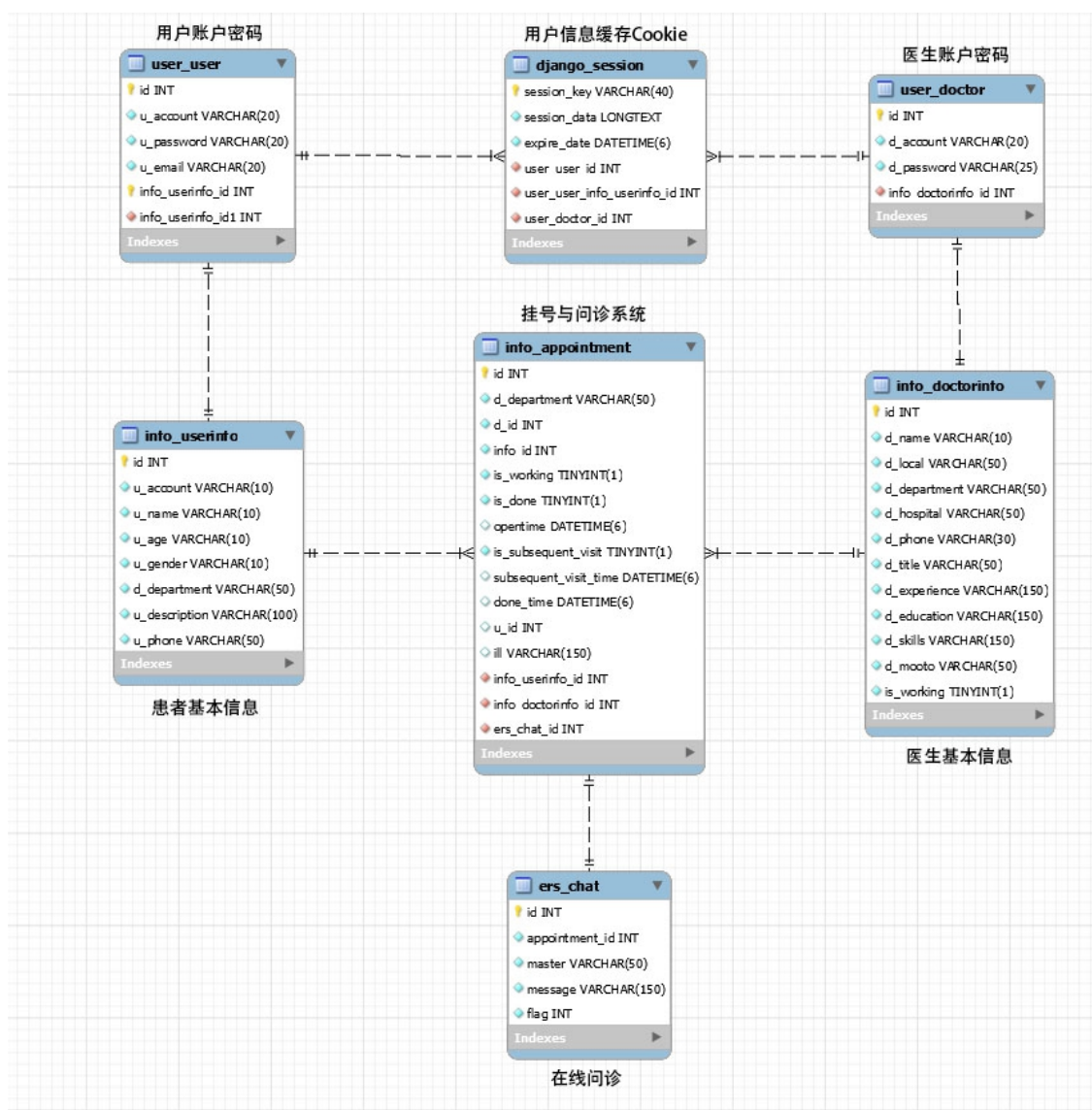


图 3-7 E-R 图

4 患者模块的详细设计与实现

4.1 用户管理功能的实现

4.1.1 用户注册

用户点击网站并进入，找到注册页面，通过输入用户名、密码、重复密码、邮箱之后点击注册按钮，其中用户名和密码都需六位以上，注册成功后自动登录。

注册界面如图 4-1 所示：



The registration form interface is displayed on a light gray background. At the top, there are two links: '登录' (Login) in blue and '注册' (Register) in red, with the latter being underlined. Below these links are four input fields: '账户' (Account) with the placeholder text 'yuanyuan', '密码' (Password) with placeholder dots, '重复密码' (Repeat Password) with placeholder dots, and '电子邮箱' (Email) with the placeholder text '1710164628@qq.com'. A large black button labeled '注册' (Register) is positioned below the input fields. At the bottom of the form, there is a link that reads '已有账号? 点此登录' (Already have an account? Click here to login).

图 4-1 注册界面

4.1.2 用户登录

用户点击网站并进入，找到登录页面，通过输入用户名、密码之后点击登录按钮即可实现登录功能。

登录界面如图 4-2 所示：

登录 注册

账户

yuanyuan

密码

登录

没有账户? 点此注册

图 4-2 登录界面图

4.2 找专家功能的实现

用户登录后，按照图标显示引导患者问诊流程，用户点击“找专家”进入填写挂号个人信息，并对病情进行描述，信息上传后系统进行医生匹配，为用户匹配一个适合自己科室的医生。

找专家界面图如 4-3 所示：

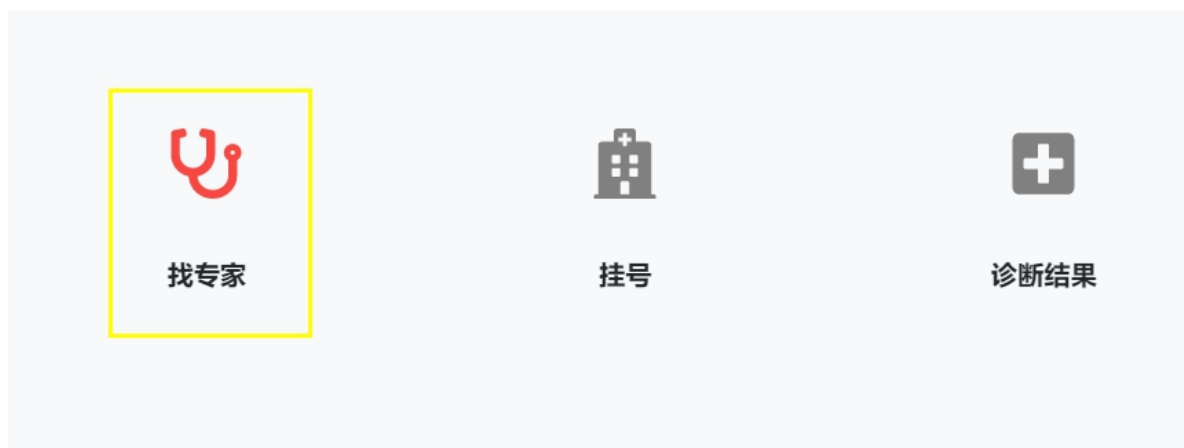


图 4-3 找专家界面图

填写挂号信息界面图如 4-4 所示：

这次挂号是为了：

yuanyuan

非账户本人

请输入患者的基本信息病况，选择科室我们帮您匹配医生

沅沅

15093114411

20

女

药剂科

感冒拿药

已同意隐私政策及免责声明
请填写真实情况，更改信息须付手续费

匹配医生

图 4-4 填写挂号信息界面图

4.3 在线挂号功能的实现

系统匹配过医生后，用户点击“挂号”，可以在该医生的门诊下进行预约排队挂号，挂号成功后，可进入聊天问诊，与医生可畅所欲言的说明自己的病况。

挂号界面图如 4-5 所示：



图 4-5 挂号界面图

用户预约排队挂号界面图如 4-6 所示：



图 4-6 用户预约排队挂号界面图

聊天问诊界面图如 4-7 所示：

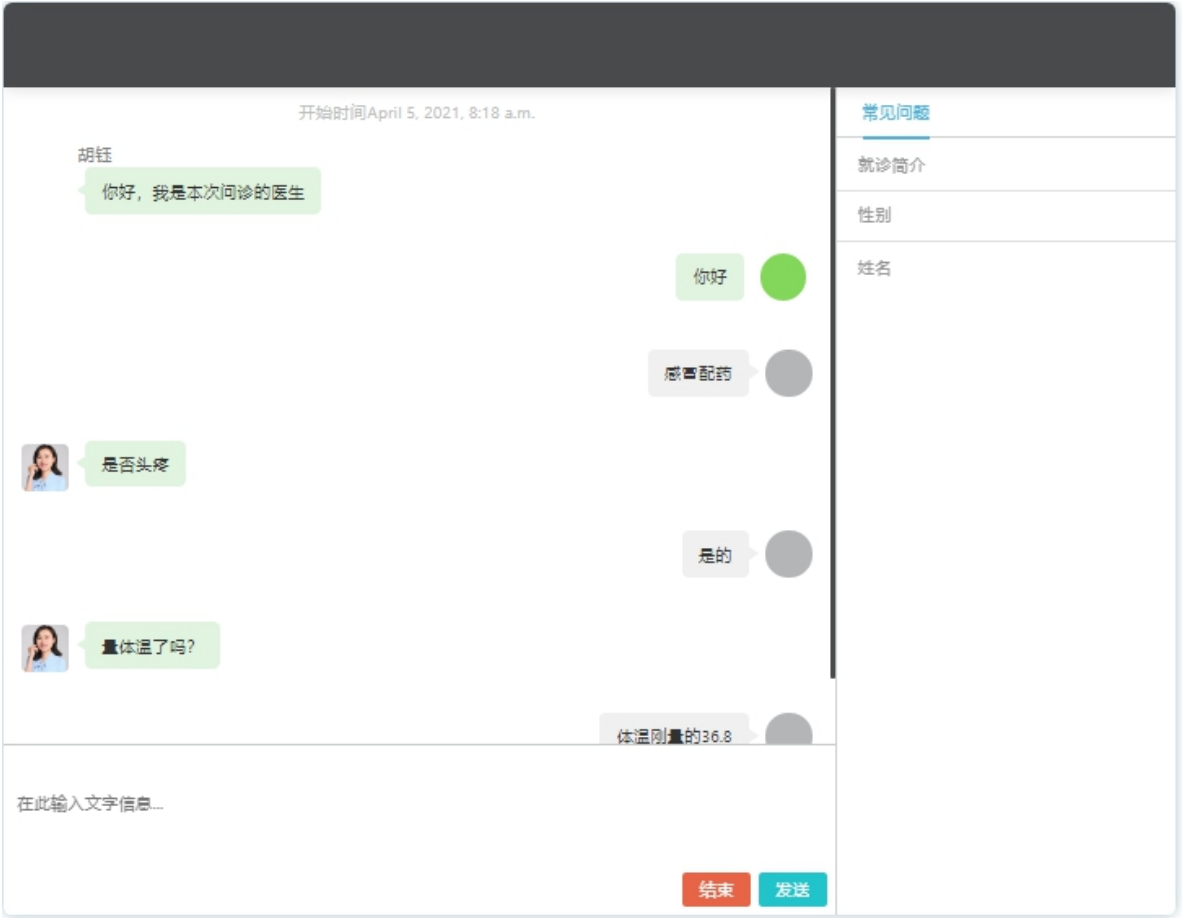


图 4-7 聊天问诊界面图

4.3 查询诊断结果功能的实现

用户问诊后，可在首页的“问诊结果”中，查看医生开的药以及医嘱。

诊断结果界面图如 4-8 所示：



图 4-8 诊断结果界面图

查看诊断结果界面图如 4-9 所示：



图 4-9 查看诊断结果界面图

4.5 查询医生信息功能的实现

用户可在首页点击医生头像查看医生个人简历。医生简历界面图如 4-10 所示：



胡钰

武汉市青山区本溪街1号 联系电话（总机）027-86868999

认真负责地从事药品的调配、审核、发药,耐心细致地为患者讲解药品的用法用量及注意事项。

图 4-10 医生简历界面

5 模块和系统测试

本在线医疗平台在开发的过程中，先是对其进行了可行性分析，在可行的基础之上对其用户功能需求做了概述，通过平时的学习 Python 技术其进行开发，该系统各个方面的设计都得到了完善。该系统的优势主要表现在以下几个方面：

1. 本在线医疗平台界面清晰，相对来说比较简洁明了，用户在浏览和使用的过程中将会得到很好的体验。
 2. 用户可以与医生随时随地畅所欲言，不用局限于在医院看病，用户会有很好的看病体验。
 3. 操作简便，用户可以根据网站图标指引进行线上看诊操作，非常的直观便捷。
- 本系统将采用黑盒测试法，对一些功能进行测试。

5.1 用户登录功能测试

在用户登录过程中只有输入正确的账号密码才会登陆成功；否则会弹出“密码错误或无此账号”的错误提示。

登录错误提示图如图 5-1 所示：

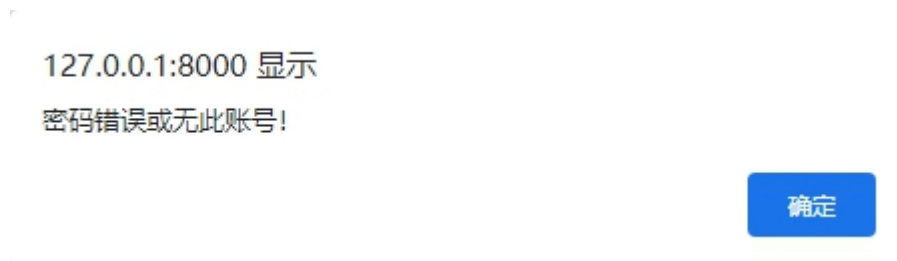


图 5-1 登录错误提示图

5.2 用户注册功能测试

在用户注册过程中不得输入已注册的账户名称，否则会弹出“账户名已存在”的错误提示。账户输入错误提示图如 5-2 所示：



图 5-2 账户输入错误提示图

在用户注册过程中要输入两次相同的密码，否则会弹出“请输入相同的密码”的错误提示。如图 5-2 所示：

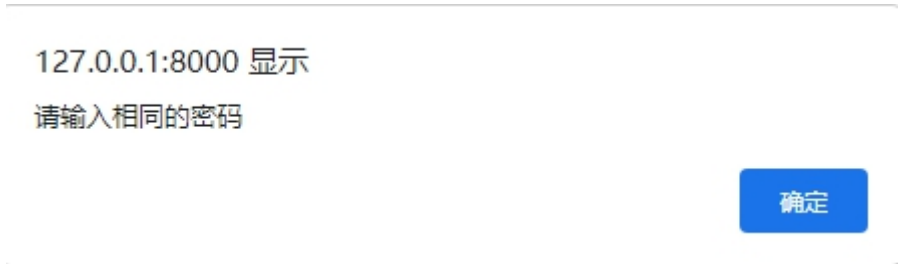


图 5-2

在用户注册过程中要输入 6 位以上的密码，否则会弹出“账户密码不可少于 6 位”的错误提示。如图 5-3 所示：



图 5-3

在用户注册过程中要填写正确的邮箱格式，否则会弹出“邮箱格式不正确”的错误提示。邮箱错误提示图如 5-4 所示：



图 5-4 邮箱错误提示图

5.3 挂号测试

在用户挂号中，需要排队等候，如果前面有人的话，会有“前方人数”的显示。挂号测试图如 5-5 所示：

预约信息

科室
药剂科

前方人数
1

病状描述
发烧

性别
男

年龄
20

姓名
王玉



胡钰

擅长

感冒 狂犬病 流感 激素避孕 疫苗接种 安

全用药

点击头像查看详细信息

还没有排到您稍等片刻

图 5-5 挂号测试图

5.4 总结

软件测试是软件开发中不可缺少的重要环节之一，不得忽视。本在线医疗平台经过以上测试，运行效果良好，达到了预期的目的。

结 论

在这几个月的不懈努力中，我总算是完成了在线医疗平台的开发。日常生活中，我们大多数人都习惯了老师讲什么我们听什么的授课方式，突然要自己做一个完整的毕业设计，刚开始来说，确实有点束手无策。在这个过程中，我难免也会遇到过不少困难，但我都没有放弃，始终咬牙坚持，走到了最后。

在整个在线医疗平台的开发过程中，我不仅学会了将课本上的知识与实际结合起来，也学会了许多课本上所不能授予我的新知识。通过此次毕业设计的完成，我很好的知道了自己不足所在，未来，我将抱着更加诚恳的态度去不断改正，通过不懈的努力去完善自己，提升自己。

在系统设计的过程中还遇到的一个问题就是，自己的英语水平还有待提高，很多关于网站开发技术的资料都是英文的文献，完全依靠自己的英语水平无法完全看懂，只能在翻译软件的实时性翻译的辅助下才勉强看懂。显然认识到英语能力的高低直接影响到系统的开发。

经过编程工作的学习让我有了更多的信心，我相信在未来的路上，我会走的更好。无论整个过程中我遇到了多少不确定因素，最终还是完成了预定的需求功能。虽然整体来说做的可能并没有十分好，该系统需要改进的地方或许还有很多，我相信通过以后的努力，我可以将其变得更加完美。

致 谢

时间过得飞快，转眼间迎来了毕业季，在这漫漫的大学时光里，每一刻都是值得珍惜回顾的。这满满的充实感，其实离不开身边很多老师，同学和朋友给了我无私的关怀和帮助，在此我想表达自己诚挚的谢意。

毕业设计帮助了我学习到了更多的知识。可以说，使用计算机来解决管理过中的任务繁重、效率低下问题是一个不错的方式。在学习理论时候，我掌握了基本知识理论课程和专业课程知识，同时也产生了一些链接和系统设计全面的测试计划，但这是第一次真正动手设计管理软件的过程。在这个项目设计阶段，我遇到过考验我各方面能力的各种难题，不过，问题最后都一一解决。在困难的过程中大大提高了我解决问题的能力，让我更好的理论联系实际，提高了专业技能。

从毕业设计的开题到结束的这段时间内，我终于完成了一个更完整，全面的系统。随着毕业设计的完成，我真正了解了软件工程，也是以前的理论知识巩固的全过程，同时也奠定了我在今后的工作中的良好基础。

最后，特别要感谢的就是此次负责我的毕业设计及论文的邵彧老师，从选题到论文撰写结束的过程中，当有许多无法解决的问题时，无论是在线上还是在线下，一直都是耐心的指导，让我顺利的完成了毕业论文的撰写，并且对大学的学习递交上一份满意的答卷。

参考文献

- [1] Journal.Exploring the freemium business model for online medical consultation services in China[J].Information Processing & Management,2021,58(3).
- [2] (巴西) Luciano Ramalho.Fluent Python[M].北京:人民邮电出版社, 2017.
- [3] 王佳丽,王嘉玲,张瑶.新型冠状病毒肺炎疫情前后民众对在线医疗服务认知态度和使用意愿的变化及影响因素[J].中国心理学杂志,2021,29(3).
- [4] 刘长龙.Python 高效开发实战——Django、Tornado、Flask、Twisted[M].北京:电子工业出版社, 2016.
- [5] [美].埃里克.马瑟斯.python 编程从入门到实践.北京.人民邮电出版社.2020
- [6] 斯维加特.Python 编程快速上手.北京.人民邮电出版社.2016
- [7] 巴里.Head First Python (中文版).北京.中国电力出版社.2012
- [8] 胡杨.Django 企业开发实战.北京:人民邮电出版社, 2019.
- [9] (美)杰佛等著.Django Web 开发指南[M].北京:机械工业出版社, 2009.
- [10] 王珊、萨师煊.数据库系统概论(第5版)[M].高等教育出版社:北京,2014.
- [11] (美)施瓦茨等著.高性能 MySQL[M].北京:电子工业出版社, 2013.
- [12] 相万让.网页设计与制作(第3版)[M].人民邮电出版社:北京,2012.
- [13] 郑娅峰,张永强.网页设计与开发[M].北京:清华大学出版社, 2016
- [14] 卢潇.软件工程.北京:清华大学出版社;北京交通大学出版社, 2005
- [15] 张友生,李雄.软件体系结构原理、方法与实践.清华大学出版社, 2008-8.

附 录

用户注册主要代码:

```
def register_view(request):
    uname = request.POST.get('uname')
    pwd = request.POST.get('password')
    pwdagain = request.POST.get('passwordagain')
    email = request.POST.get('email')
    userList = user.objects.filter(u_account=uname)
    # 判断是否存在
    if userList:
        return render(request, 'logsign.html',
                        {'code': '账户名已存在', 'sign': True})
    if pwd != pwdagain:
        return render(request, 'logsign.html',
                        {'code': '请输入相同的密码', 'sign': True})
    if len(pwd) < 6 and len(uname) < 6:
        return render(request, 'logsign.html',
                        {'code': '账户密码不可少于6位', 'sign': True})
    if not re.match(r'^[0-9a-zA-Z_]{0,19}@[0-9a-zA-Z]{1,13}\.[com,cn,net]{1,3}$', email):
        return render(request, 'logsign.html',
                        {'code': '邮箱格式不正确', 'sign': True})
    else:
        users = user.objects.create(u_account=uname, u_password=pwd, u_email=email)
        request.session['username'] = uname
        request.session['is_login'] = True
        request.session.set_expiry(24 * 60 * 60)
        return HttpResponseRedirect('/', {'username': uname})
```

用户登录主要代码:

```
def login_view(request):
    uname = request.POST.get('uname')
    pwd = request.POST.get('password')
    if uname and pwd:
        # Correct password, and the user is marked "active"
        userList = user.objects.filter(u_account=uname, u_password=pwd)

        if userList:
            request.session['username'] = uname
            request.session['is_login'] = True
            request.session.set_expiry(24 * 60 * 60)
            # request
            return HttpResponseRedirect('/', {'user_name': uname})
        else:
            return render(request, 'logsign.html',
                           {'code': '密码错误或无此账号!', 'sign': True})
```

找专家主要代码:

用户挂号信息

```
def user_resume_view(request):
    username = request.session.get('username')
    realname = request.POST.get('realname')
    phone = request.POST.get('phone')
    age = request.POST.get('age')
    gender = request.POST.get('gender')
    department = request.POST.get('department')
    desc = request.POST.get('desc')
    resume = userinfo.objects.create(u_account=username, u_name=realname,
                                     u_age=age, u_gender=gender,
                                     d_department=department,
                                     u_description=desc, u_phone=phone)

    return render(request, 'resume.html', {'flag': True})
```

在线挂号主要代码:

```
def index(request):
    username = request.session.get('username')

    flag = userinfo.objects.filter(u_account=username)
    if flag:
        userid = user.objects.get(u_account=username).id
        # 查询appointment是否创建
        infoid = userinfo.objects.filter(u_account=username).last()
        flag1 = userinfo.objects.filter(u_account=username).count()
        flag2 = appointment.objects.filter(info_id=infoid.id)
        doctor = doctorinfo.objects.get(d_department=infoid.d_department)
        if flag2:
            pass
        else:
            make_appointment = appointment.objects.create(d_department=infoid.d_department, info_id=infoid.id,
                                                            d_id=doctor.id, u_id=userid)
            is_done = appointment.objects.filter(u_id=userid, is_done=0).count()
            have_done = appointment.objects.filter(u_id=userid, is_done=1).count()

            check_queue_up = appointment.objects.filter(d_department=infoid.d_department, is_done=0).count()
            data = {'doctor_name': doctor.d_name, 'reason': infoid.u_description, 'department': infoid.d_department,
                    'doctor_id': doctor.id, 'age': infoid.u_age, 'gender': infoid.u_gender, 'name': infoid.u_name,
                    'd_title': doctor.d_title, 'd_experience': doctor.d_experience, 'd_skills': doctor.d_skills,
                    'queue_up': check_queue_up, 'edu': doctor.d_education, 'is_done': is_done, 'have_done': have_done}
            return render(request, 'index2.html', {"list": data})
```

查询诊断结果主要代码:

查询诊断结果

```
def u_center(request):
    username = request.session.get('username')
    userid = user.objects.get(u_account=username).id
    appointments = appointment.objects.filter(u_id=userid, is_done=0)
    appointments_isdone = appointment.objects.filter(u_id=userid, is_done=1)
    return render(request, 'donecenter.html',
                  {'username': username, 'appointments': appointments,
                   'appointments_isdone': appointments_isdone})
```

在线问诊主要代码:

返回聊天页面数据

```
def chat(request):
    username = request.session.get('username')
    userid = user.objects.get(u_account=username).id
    userinfos = userinfo.objects.filter(u_account=username).last()
    doctor = doctorinfo.objects.get(d_department=userinfos.d_department)
    appointments = appointment.objects.get(u_id=userid, is_done=0)
    appointments.opentime = timezone.localtime(timezone.now())
    appointments.is_working = True
    appointments.save()
    data = {'id': appointments.id, 'time': appointments.opentime,
           'doctor_name': doctor.d_name,
           'reason': userinfos.u_description, 'department': userinfos.d_department,
           'doctor_id': doctor.id, 'age': userinfos.u_age,
           'gender': userinfos.u_gender, 'name': userinfos.u_name }
    return render(request, 'chat.html', {"list": data})
```

#发送信息

```
def push_message(request):
    data = json.loads(request.body)
    appointmentid = data["appointmentid"]
    username = data["master"]
    message = data["message"]
    flag = data["flag"]
    plus_message = chats.objects.create(appointment_id=appointmentid,
                                       master=username, message=message, flag=flag)
    return HttpResponse(request.body)
```

#获得聊天信息

```
def get_message(request):
    data = json.loads(request.body)
    appointmentid = data["appointmentid"]
    other_side = data["otherside"]
    flag = data["cflag"]
    flag = flag - 1
    print(flag)
    check_new_message = chats.objects.filter(appointment_id=appointmentid,
                                             master=other_side).count()
    if flag != check_new_message:
        new_message = chats.objects.get(appointment_id=appointmentid,
                                       master=other_side, flag=flag)
        flag = flag + 1
        data = {'newmessage': new_message.message, 'flag': flag}
        return HttpResponse(json.dumps(data, ensure_ascii=False),
                           content_type='application/json')
```

#返回医生问诊页面

```
def dochat(request, aid):
    return render(request, 'dochat.html', {'id': aid})
```

查询医生信息主要代码:

医生信息

```
def doctorinfo_view(request, did):
    doctors = doctorinfo.objects.get(id=did)
    data = {'id': doctors.id, 'name': doctors.d_name,
            'local': doctors.d_local, 'department': doctors.d_department,
            'hospital': doctors.d_hospital, 'phone': doctors.d_phone,
            'title': doctors.d_title,
            'experience': doctors.d_experience,
            'education': doctors.d_education, 'skills': doctors.d_skills,
            'mooto': doctors.d_mooto}
    return render(request, 'doctor.html', {'list': data})
```