毕 业 设 计

题目： 留学信息检索系统的设计与实现

学生姓名： 陈高松 学号： 16230238003

专业班级： 2016 级信息管理与信息系统

指导老师： 邱明辉

二级学院： 大数据与计算机学院

2020 年 5 月

留学信息检索系统的设计与实现

摘　要

随着中国经济的不断发展以及近年来人民生活水平的不断提高，留学人数呈现不断上涨的趋势。根据教育部的数据，2018 年出国留学人数达到 66.12 万，比 2017 年增长了 8.33%。留学人数的增加，导致各种留学的中介也呈现上涨趋势，但是这些中介的质量却相差甚远。这些问题导致用户申请效率低下，大多数在线上发布广告以后线下进行人工申请。这些平台的功能是发布留学信息和发布留学新闻，用户只能单方面的浏览信息。有些平台甚至连学校，专业的基本信息也无法提供，直接粗暴的让用户留下电话，然后进一步获取利益，这种模式很大程度上的依赖实体店面的运营。

在此背景下，通过对需求进行分析，本系统采用 Django 框架以及开源数据库 MySQL 来进行开发，采用 scrapy 技术从 1500 个高等院校的官网信息进行实时抓取，并实时展示在前端供用户进行检索，最终开发出一个用户能够高效的查询到自己留学所需的信息的系统；整个系统采用敏捷开发的方法，可以大大的提高开发效率，最终使得系统成功发布。

关键词： Django； Redis； MYSQL；留学；系统开发

DESIGN AND IMPLEMENTATION OF INFORMATION RETRIEVAL SYSTEM FOR STUDYING ABROAD

ABSTRACT

With the continuous development of China's economy and the continuous improvement of people's living standards in recent years, the awareness of studying abroad has been on the rise. According to data from the Ministry of Education, the number of students studying abroad in 2018 reached 661,200, an increase of 8.33% over 2017. Although the number of students studying abroad has increased and various intermediaries studying abroad have also shown an upward trend, the quality of these intermediaries is very different. In addition, the application efficiency is low, and most of them now manually apply offline after publishing the advertisement. The functions of these platforms are to publish information about studying abroad and to publish news about studying abroad. Users can only browse information unilaterally. This mode is still offline and cannot interact with users online. Some platforms even schools, professional basic information can not provide, directly and rudely let users leave a phone call, and then further obtain benefits, this model relies heavily on the operation of physical stores.

In this context, by analyzing requirements, the system was developed using the Django framework and the open source database MySQL, and eventually developed a system where users can query the information they need to study abroad; the entire system uses an agile development method. Can greatly improve the development efficiency, and ultimately make the system successful..

Key words: Django; MYSQL; Studying Abroad; System development

目　录

1 绪论

1.1 论文的背景与意义

随着中国经济的不断发展，留学人数呈现不断上涨的趋势。根据教育部的数据，2018 年出国留学人数达到 66.12 万，比 2017 年增长了 8.33％。我们可以看到，出国留学的青年潮正在逐渐上升。

比较过去五年的数据，可以发现，有“一般员工”背景的海外学生及其父母的比例逐年增加，到了 2019 年它将成为最主流的父母职位类型，所占比例为 43％。可以看出，中产阶级留学的人数稳步上升。然而，这些中产阶级的家庭确不一定愿意选择中介，因此他们可能会选择 DIY 申请，DIY 申请中最重要的莫过于选校定位了，这时候，申请人所能够掌握的信息的丰富量很大程度的决定了申请人最自己的定位是否准确。倘若出现信息不对称的情况，很可能会导致申请失败或者申请到的院校的水平远低于自己所能申请到的最好的院校的情况。

对于想要出国的深造的同学而言，在申请过程中需要需要收集很多信息：一是留学的地点。二是学校和专业。三是导师。四是成功申请过程的流程以及出国的最低要求和费用。五是文书撰写的方法。这一类问题属于一般常识性问题，我们通常需要搜索引擎来进行检索，但是传统的搜索引擎普遍存在一下缺点：□传统的搜索引擎专业的数据库接口较少，搜索出的有效信息大部分只能浏览目录信息，由于版权或者 robots 协议的限制，使得用户不能浏览详细的内容；□传统的搜索引擎对信息不具有选择和价值判断能力，导致检索出的内容排序的结果不理想，甚至会检索到一些错误的信息，这些信息不够权威□没有同一的网络信息分类标准，令网络用户无所适从，而且网络信息分类不够准确，使得信息分类体系的覆盖达不到要求、对专业较强的深度信息的查全率较低、用户在检索信息时不够能做到没有重复和遗漏；□搜索结果中广告、垃圾网站和死链接比较多；□难以搜索动态网页，查全率下降；

1.2 国内外研究现状

目前国外在选校的过程中，主要通过朋友推荐，按照距离家的距离，按照 QS 等排名进行检索，然后去官网浏览具体信息。

国内的用户除了国外用户的方法以外，选择中介的会有很多。但是传统的留学信息检索平台的弊端越来越明显，这些网站里面充斥着大量的广告，信息的质量也很差，随着互联网的高速发展，留学信息检索平台的的需求也随之产生。在百度中使用关键词“留学信息检索系统”进行检索，找到了 1640000 个结果，在谷歌中进行检索，则搜索到了 5949000 个结果，由此可见留学信息的收集网站竞争非常激烈。这些中介网站申请效率低下，大多数现在发布广告以后线下进行人工申请。目前国内外尚无专门用于高等院校的信息检索系统，国内仅有少量的留学中介的做的检索工具。这些平台的功能是发布留学信息和发布留学新闻，用户只能单方面的浏览信息，其中很多有效的信息需要用户付费才可以浏览，此外信息更新的速度不理想。这种模式依然是线下模式，无法和用户在线上进行互动。有些平台甚至连学校，专业的基本信息也无法提供，直接粗暴的让用户留下电话，然后进一步获取利益，这种模式极大的程度依赖人工，而且效率低下费用高昂。

综合目前留学系统的现状，缺点主要有一下几个方面：

学校和专业的信息不够全面，使用户不能检索到自己所需要的信息，比如学校的概况，排名，学费等。

缺少学校和用户的匹配，比如最低托福成绩，以及用户所能承受的最低学费，这些信息需要人凭借自己的经验给学生定位，不久匹配的效果不理想，而且效率低下。

学生的评价无法呈现在线上，使得学生无法了解的学校官网之外的信息。

信息不够权威，多数信息来自第三方的小机构，甚至有些数据没有注明来源。

本文所介绍的留学信息检索平台，是一个垂直的信息检索系统。留学信息检索系统是属于专业信息检索系统的一种，专注于对特定领域的信息检索，这种信息是为了满足特定人群的需要，专门给特定的人群提供权威，专业全面的留学信息，包含了大量的院校以及对应的项目和教授信息，留学指南、海外生活和签证攻略等。本系统为用户省下了很多时间和金钱，大大提高了用户检索效率，并且打破了时间和空间的限制。

1.3 论文的主要内容

本设计的目的是解决以上问题，设计一个公益平台，服务于申请国外的院校的同学，使得用户可以高效率的检索到自己所需的目标院校的信息，其主要内容

包括：

（1）由于本系统的核心功能是信息的检索，所以前端并不需要很复杂，所以本系统的前端没有采用比较流行的前端框架，而是简单的采用了HTML5+CSS3+JavaScript 的简单技术，部分地方使用了布局采用了 bootstrap，少量的使用了 Ajax 技术。

（2）系统的主要功能是检索以及对用户实现了个性化的功能，整个系统分为普通用户以及管理员用户，管理员用户拥有所有权限，普通用户只拥有部分权限，整个模块的功能分为关键词查询，排名查询，自定义查询三块。

（3）本系统来源是生活中的实际问题，其根本目的是解决用户在检索目标院校中的信息孤岛问题，各种院校的信息没有集成，使得用户的检索效率大大降低。通过本系统来检索目前院校，使得检索效率大大提高。

本系统主要实现了用户对全球排名前 1500 的高等院校进行检索，以及用户之间评论互动功能。

系统采用了 Django 框架中 MVC 的设计思想，采用 MySQL 数据库，redis 作为缓存，nginx 作为 web 服务器。通过网上调研，确定了总体模块和功能的划分，采用敏捷开发的设计方法，最终实现整个系统，提高用户的检索效率。

2 相关知识概述

本节对系统开发中使用的 python,Django，nginx，celery,mysql,redis，haystack 等技术进行解释。

2.1 Python 的概述

Python 是一种解释型、高性能的高级通用的编程语言，Python 由 Guido van Rossum 创建，并在 1991 年发布首个版本。Python 的设计理念是通过显著的使用大量的缩进来增强其代码的可读性。其语言结构和面向对象的设计方法帮助开发者在各种项目中写出简洁，富有逻辑的代码。Python 是动态的并且支持垃圾回收。它支持包括面向过程，面向对象，函数型编程等多种编程模式。由于包含众多复杂的标准库，因此也被成为胶水语言。Python 解释器支持多种操作系统。全球开发者组成的社区共同维护着 CPython。本系统中所有 web 后端实现均使用 python 实现。

2.2 Django 框架

Django 是一款基于 python 的基础 MTV 的设计理念免费并且开源的 web 框架。它由非盈利并且独立的机构- Django Software Foundation (DSF)维护。它由大量的经验丰富的开发者开发，解决了 web 开发中的很多麻烦，因此开发者可以专心的编写自己的应用程序而不用重复造轮子。Django 追求 Don't Repeat Yourself(DRY)的原则，使得这个框架更加的高效。换句话说，Django 不需要你编写已经存在的代码，因为 django 允许你想搭建乐高积木一样来构建网站。这个框架适用于高负载的系统，得益于大量的开发者的贡献的代码，django 可以大大的减少开发时间。由于采用 django 编写，django 有明了的 python 结构。刚开始作为一个 modle-view-contral 的架构，而且这种设计理念至今仍然存在最新的版本中。这种 MVC 架构允许开发者在不影响其他的情况下改变一个 app 中的视图部分并且和业务逻辑分开。事实上，开发者经常参考 djangoMTV 的架构。这三层（model，view，Template）可靠的对于不同的事情独立使用，图 2.1 展示了 Django 的 MTV 执行流程模型。

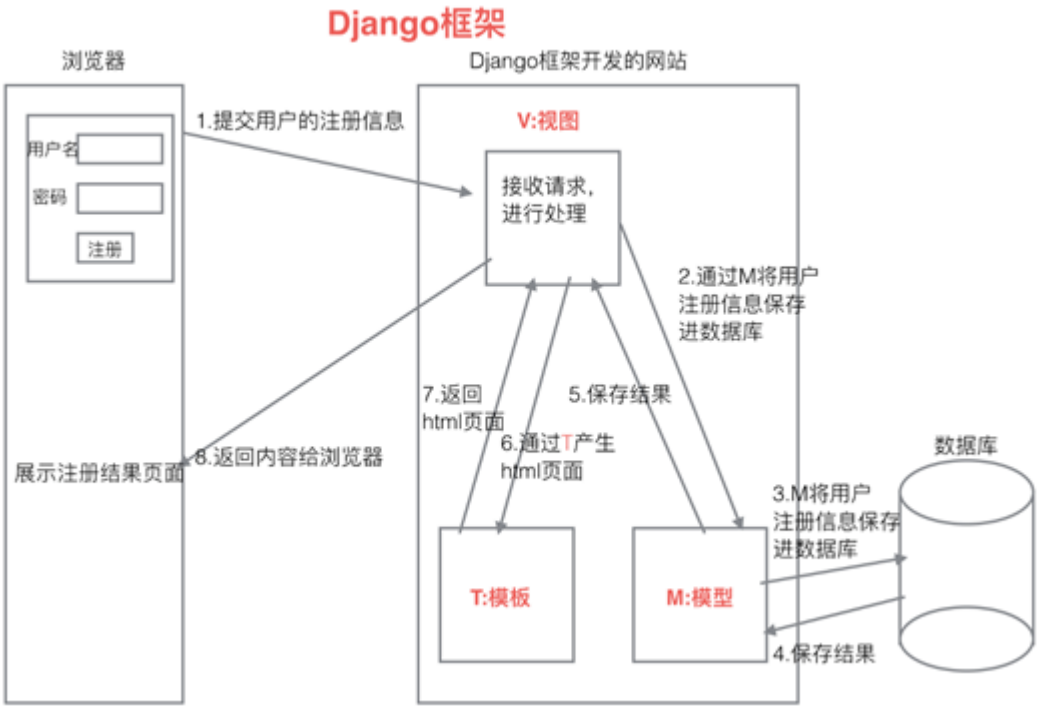

图 2.1　MTV 执行流程模型图

Django 有强大的模板引擎而且本身自带的标记语言也有很多的工具。模板是用来返回的数据的带有 HTML 代码的文件，它包含的文件可以是动态的也可以是静态的。自从模板里面没有业务逻辑开始，它只表示数据。

Django 的优点:丰富的生态，成熟，默认的管理员面板，有利于搜索引擎优化，支持大量的插件，丰富的库等。

2.3 Nginx

Nginx 是一款 HTTP 和反向代理的服务器，也可以用作邮件代理、一般的 TCP/UDP 代理服务器、负载平衡器，和其他服务器相比，nginx 具有一下优点。

1.结构简单。Nginx 和 apache 是使用的最多的 web 服务器。但是对于大多数的 web 设置来说，nginx 的表现要超过 apache。Nginx 的主机对于提交请求使用一种事件驱动的异步实现方式。这提供了负载下的可靠的性能。

2.开源。Nginx 是世界上使用的第二多的开源的 web 服务器。它兼容了包括 Windows 和 linux 在内的主流操作系统，这使得 nginx 成为最好的选择

3. 轻量，高性能。虽然 nginx 的体积很小，但是它通过为实时应用程序降低 RAM 和 CPU 的的消耗来增加性能。

4. 可拓展性。Nginx 的负载均衡，逻辑结构，简单的配置，可预测的性能以及灵活性使得它成为每天处理百万级别的访问量的大型企业和网站的解决方法。

5. 稳定性。今天超过百分之十的百万用户的网站选择 nginx，它特别为带有负载均衡支持和代理支持的高并发的网站而设计。如果你是一个网站管理员，nginx 一定是你最好的选择。

2.4 MySQL

MySQL 是一个开源的关系型数据库管理系统。MySQL 由一个被 Sun Microsystems（现在的 Oracle Corporation)）收购的瑞典公司赞助和所有。MySQL 采用 C 和 C++编写。其 SQL 解析器采用 yacc 编写，但是它使用了自制的语法解析器。MySQL 可以在很多操作系统上运行，包括 macOS, Microsoft Windows 等.

2.5 Redis

Redis 是一款开源的用作内存中的数据存储系统，可以当作数据库，缓存和消息中间件使用。它支持字符串，哈希，列表和集合等各种类型的数据结构。Redis 使用 C 语言编写，是读写速度很快的非关系型数据库。目前 redis 用在微博，GitHub，snapchat 等大型网站中。Redis 对开发者很友好，支持 JavaScript,Java，Go,python ，C++等多种主流语言。

Redis 是当今最受欢迎的分布式缓存引擎。经过生产验证，提供了许多功能，使其成为应用程序的理想分布式缓存层。Redis Cloud 和 RLEC 提供了扩展开放源代码功能的其他功能 Redis 使它更加适合缓存。

2.6 Celery

Celery 是基于分布式消息传递的异步任务队列。它专注于实时操作，同时也很好的支持调度。执行单元（也称为任务），通过多进程，Eventlet, 或者 gevent 在单个或多个服务器上同时执行。任务可以在后台异步执行或者同步执行。

3 系统分析

本系统主要申请海外院校的同学，使其在申请的时候更加快捷，高效选择到和自己匹配的学校。本系统将原本散乱、无序的信息合理的组织在一起并且结合强大的院校数据库，以及简洁的 UI，从根本上 满足用户检索院校的信息需求。

本章将分别从系统需求和系统功能两方面来进行系统分析。

3.1 系统需求可行性分析

系统需求的分析在整个系统的生命周期里面具有不可替代的重要性，这一个重要的过程对整个软件系统的后期质量、整个软件系统开发的成败有着决定性的作用，所以说系统需求分析阶段是不可缺少的一个重要环节。

3.1.1 技术可行性

本系统采用 Python 进行开发，python 具有免费、开源、跨平台的特点，并且具有可拓展性，可以调用 C 和 C++。在安装 python 环境后可以运行在 Windows，Linux 和 mac OS 上，且一次编写在进行简单的配置就可以跨平台运行。本项目采用的技术架构 django+mysql+redis。Django 框架是目前流行的 MVC 形的 web 框架，开发容易，效率高，且安全性高。MySQL 是一种关系型数据库，适用于中小型项目，对于本项目的需求完全足够。Redis 的一种 key-value 形的非关系型数据库，读写效率高，支持持久化存储，可以大大的提高读写效率、增强用户体验，而且数据结构丰富，操作指定简单，可以完美的和 django 框框兼容。

3.1.2 经济可行性

经济可行性是评估系统开发阶段和运行时期成本以及系统后期使用所产生的效益。目前，能够出国的留学的同学大多都具有较高信息素养，人均电脑超过一台，且学习能力强。用户仅仅需要使用自己的设备（pc 和手机）上的浏览器就可以顺利的运行本系统，无需其他的费用和学习成本。

本次开发的系统所使用的技术都是免费开源的，部署在 1 核的 CPU 和 2G 内存的操作系统为 Ubuntu18.04 云服务器上仅需简单的配置便可流畅运行， 因此该系统具备经济可行性。

3.1.3 社会可行性

在特定的环境下进行本系统的开发与实施就是本系统社会可行性。在用户检索目标院校的过程中，使用传统的搜索信息进行检索，只能够检索到某个院校的官网，无法对各个院校进行横向比较，使得用户无法对院校进行筛选和比较。本系统为垂直搜索，用户可以自定义条件在本系统的数据库中进行检索，以此提高检索效率。用户可以在每个院校和项目下自由评论，从而生成更多的有价值的信息。因此使用此系统来提升效率、使得数据之间可以共享是很必要的，所以本系统在社会可行性上是可行的。

3.2 系统的功能分析

本系统开发的最终目的是在用户申请海外院校的过程中，为用户提高高校的检索方式。为了将留学中介的集中数据与院校的官方数据结合,本系统主要分为用户模块和院校模块。

3.2.1 用户模块

用户注册。用户通过输入用户名，电子邮箱和密码来注册账号，成功提交此表单以后，系统会向此邮箱发送一个激活账号的邮件。

用户登录以及注销。用户通过输入用户名和密码，用户点击登录以后，系统会校验用户名和密码是否匹配，如果未能匹配到对应的用户名和密码，系统就会给出对应的错误信息，否则进入登录成功的状态并跳转到专业。

用户激活。在用户提交注册的表单以后，系统会向这个表单里的邮箱发送一个激活的链接，用户点击这个链接就可以把自己注册的账号激活。

浏览历史。用户在成功登录以后，可以查看最近浏览的信息（院校，学位项目，以及导师）。

用户收藏。用户在浏览院校，学位项目以及导师等信息时，可以点击收藏按钮来收藏对应的数据，这时收藏的数据可以在用户对应的信息里面看到。

用户中心。用户在注册的时候，只填写了最基本的信息，通过用户中心，用户可以完善姓名，意向申请国家，意向申请学校类型，学位，学位类型，意向申请的排名多少学位，本科毕业院校类型，雅思成绩，托福成绩，预算费用，年龄，性别，是否参加过竞赛，是否参加过社团活动，是否有工作经验和是否有科研经

验等。

3.2.2 院校模块

主页。主页可以通过条件来检索院校。

地图查找。此页面将全球的 1500 所高校的按照经纬度将其渲染在地图上，最终生成了一个可视化的地图，这样用户可以直观的在地图上选择自己感兴趣的院校。

条件查找。通过自定义的条件来检索院校，比如院校所在地区，平均学费等。

项目查找。在确定院校的情况下，通过学位和学位类型来查找对应的项目。

教授查找。在确定院校的情况下，根据课程或者教授名字来检索教授的信息。

3.3 系统的性能需求

留学信息检索系统是要适合中有意向留学的用户使用的，除了要实现以上的所有检索功能，系统要求性能的稳定性也是极高的，要确保系统在运行时期不出现大的问题和用户体验良好。在稳定的前提下，系统还要求 UI 简洁，使得系统易用。

3.3.1 系统响应时间

具备快速的响应时间是一个系统必然要求的属性，系统在打开网页，点击链接操作的响应时间要要求在一定时间内，进行读写的操作响应要求在三秒以内。进行数量较多的条件搜索时候响应时间要小于 5 秒，普通用户的数据交换要求控制在秒级以内，这样才能满足系统要求。

3.3.2 系统可靠性

本系统的可靠性要求只要为用户信息以及用户文件不会丢失，以及用户的数据保存在磁盘中不会因为磁盘的空间不足而导致数据保存失败，系统要求在 7*42 小时不停的正常运行。留学信息检索系统开发采用的技术较为流行和成熟，系统因为技术的问题的几率应该是没有的，该系统部署在生产环境下面是 Linux 系统，安全系数也是较高的。

3.3.3 系统可维护性

本系统是分模块开发的，每一个大模块下面都有许多小模块，以此类推，对于二次开发是有好处的。技术层面上，采用的框架全都是开源、高性能、易插

拔的，维护起来也很方便。

3.3.4 系统可保密性

保密性能需求：用户的信息具有保密性、数据库中的用户密码等信息均采用加密的方式存储、不会被他人窃取。目前，用于网络安全的主要方法是使用安全协议 SSL / TLS。 SSL / TLS 协议与数字证书相结合基本上可以保证系统用户的安全，包括数据隐私，完整性和不可否认性，但是 SL / TLS 协议与数字证书相结合并不是主密钥。 在线银行系统本身设计中的过失或各种安全风险可能成为黑客突破的大门。 该系统直接影响用户信息的安全性，因此系统的使用已逐渐成为用户关注的热点。目前采用一下方法: （1）局域网和相关服务器的操作系统的安全是最基本的保证，其中包括检测系统漏洞，扫描 Intranet 安全性和及时修补等措施。（2）确保服务器端数据和用户数据的安全性和私密性，尤其是服务器数字证书和个人数字证书的安全性； 确保即使服务器的敏感数据和密码遭到破坏，用户也需要在一定程度上确保收单方无法直接或完全获取用户的敏感信息，包括使用摘要算法 MD5，SHA -1 等获取敏感信息。

4 系统设计

4.1 系统的架构设计

留学检索信息系统利用互联网的快捷性、易用性以及方便性，帮助用户在申请海外院校的过程中，提高用户检索信息的效率，实现由传统的人工服务到自动收集数据的转变，减少人工收集信息花费的时间，以此帮助用户做出合理、正确的决策。本系统可以为用户提高效的、详细的检索服务使得用户可以申请到自己满意的院校，它的结构图如图 4.1 所示。系统在设计的时候遵循一下原则：

(1)可用性原则。

(2)易操作性原则。

(3)可靠性原则。

(4)稳定性原则。

(5)可拓展性原则。

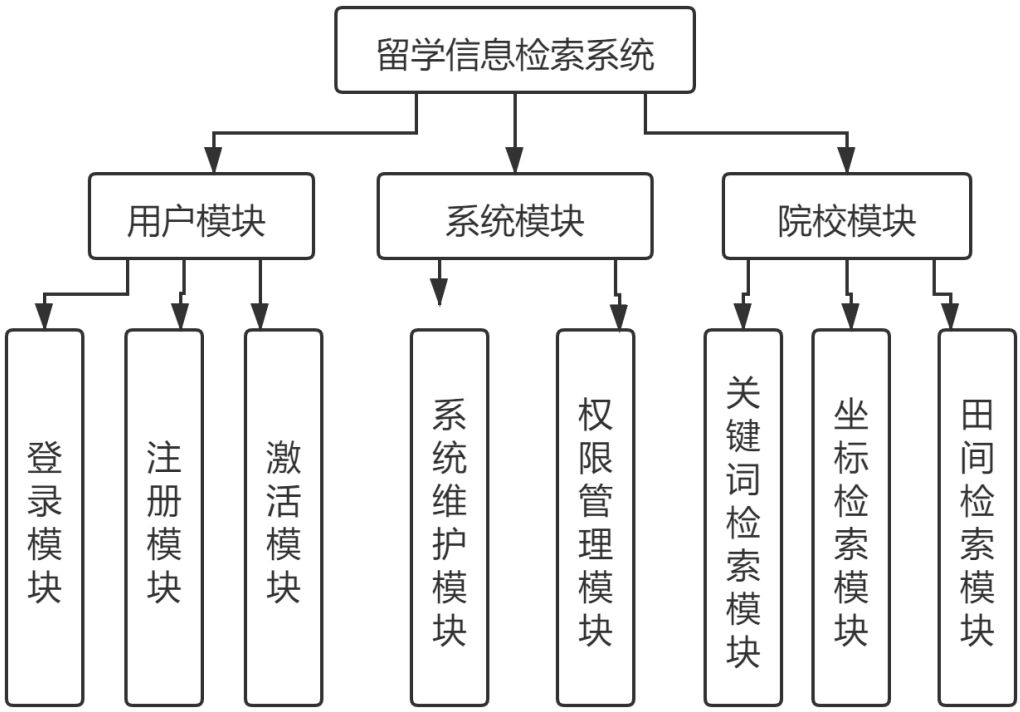

图 4.1 留学信息检索系统的结构图

4.2 系统用例图设计

本系统主要分为普通用户和管理员用户，普通用户具有浏览的权限，管理员用户具有增删查改的权限，不同类型的用户进入系统以后拥有不同的权限并仅能看到自己可以操作的界面，这很好的阐述了各司其职的要求。系统的用例图如图4.2所示。

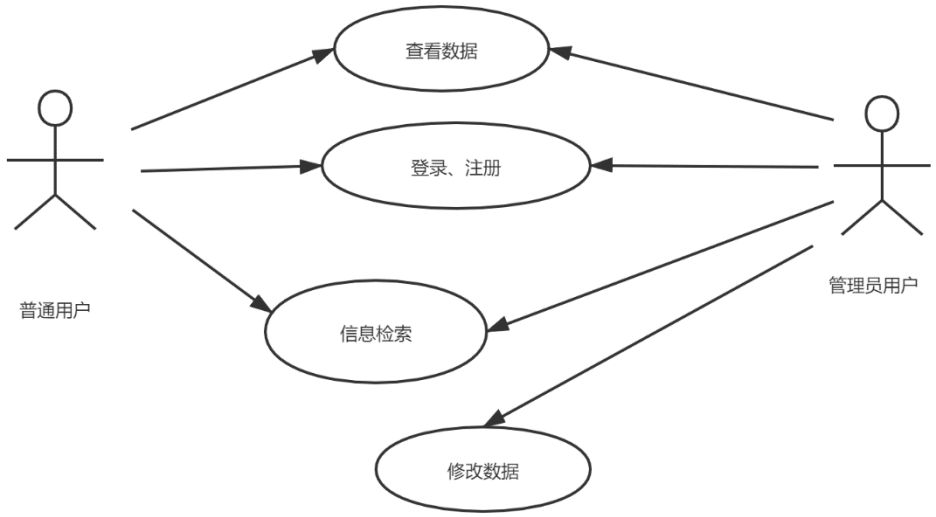

图 4.2　留学信息检索系统的用例图

4.3 系统模块模型设计

4.3.1 用户模块模型

用户包含普通用户和管理员用户。

用户登录模块。用户在输入账户密码以后，系统会在数据库中检测是否可以匹配到对应的用户名和密码,如果匹配成功，就记录登录状态然后跳转到主页，否则就会弹出对应的错误信息。

用户注册模块。用户在注册的表单里面填写自己信息，然后点击注册，系统在数据库生成一个对应的未激活的用户信息并向用户的邮箱发送一个激活邮件。

用户激活模块。用户在收到激活的邮件以后，点击邮件中的链接，便可以使用户创建的激活信息由未激活变为激活

用户信息模块。用户可以在这里修改自己的信息，查看自己发表的评论和问答信息，还可以查看自己的历史浏览记录和收藏的项目。

4.3.2 院校模块模型

搜索模块。这是留学信息检索系统中的两点功能，给用户提供海量的信息检索，这其中的内容包含，院校新闻检索，院校检索，专业检索和教授检索，提供了排名检索，坐标检索和关键词检索等功能，还提供了每个对象进行比较的功能。用户可以任意选择院校的任意属性来对比。

评论和问答模块。用户可以通过这里查看院校的评论，以及对对应的院校提出问题和做出解答。

4.3.3 系统模型

系统维护模块。主要负载系统的正常运行，其中包括流量的监控和保证运行的正常运行。

权限管理模块。负责增删权限给普通用户和管理员用户。根据每个人拥有的权限提供对应的功能。

4.4 系统数据库设计

数据库在系统中处于末端，系统中的持久层与之相互连接，它的重要性是从所周知的。数据库表结构设计的合理性直接对这个系统的性能起着决定性的作用，一个高性能的系统，其数据库表结构的设计一般会具备以下要求：SQL 语句根据项目实际需求来决定；有大数据的表，需要进行大表拆分小表，例如使用到富文本编辑器的功能；公道利用索引，索引是可以或许进步盘问效力，但索引不是越多越好；尽可能少利用内连接和子查询。系统的 E-R 图如图 4.3 所示。

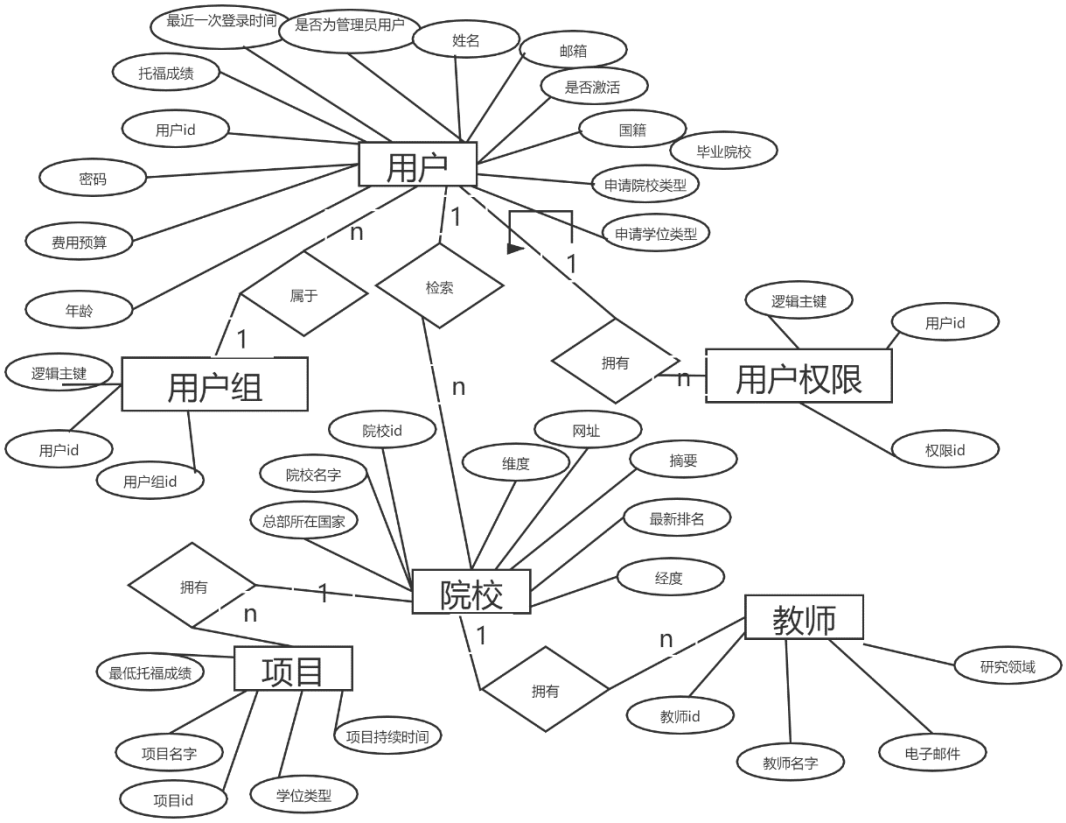

图 4.3　留学信息检索系统的 ER 图

表 4-1 到表 4-15 分别是留学信息检索的数据库各表设计结构。

表 4-1　auth_group 表用户组表

属性名称	属性类型	主键	外键	默认值	属性说明
id	int	是			逻辑主键
name	varchar(150)				用户组名称

表 4-2　auth_group_permissions 用户组权限表

属性名称	属性类型	主键	外键	默认值	属性说明
id	int	是			逻辑主键
group_id	int		是		用户组
permission_id	int		是		权限 id

表 4-3　auth_permission 用户权限表

属性名称	属性类型	主键	外键	默认值	属性说明
id	int	是			逻辑主键
name	varchar(250)				权限名
content_type_id	int		是		权限类型 id
codename	varchar(250)				编码名称

表 4-4　captcha_captchastore 验证码表

属性名称	属性类型	主键	外键	默认值	属性说明
id	int	是			逻辑主键
challenge	varchar(32)				权限
response	int				权限相应 id
hashkey	varchar				Hash 值
expiration	datetime				过期时间

表 4-5　django_admin_log 管理员日志表

属性名称	属性类型	主键	外键	默认值	属性说明
id	int	是			逻辑主键
action_time	datetime				权限
object_id	longtext		是		权限相应 id
object_repr	varchar(200)				Hash 值
action_flag	smallint				过期时间
change_message	longtext				修改的信息
content_type_id	int		是		内容种类 id
user_id	int		是		用户 id

表 4-6　django_content_type 内容类型表

属性名称	属性类型	主键	外键	默认值	属性说明
id	int	是			逻辑主键
app_label	varchar(100)				App 名称
model	varchar(100)				Model 名称

表 4-7　django_migrations 迁移表

属性名称	属性类型	主键	外键	默认值	属性说明
id	int	是			逻辑主键
app	varchar(255)				App 名称
name	varchar(255)				迁移名称
applied	datetime				迁移时间

表 4-8　django_session 登录的 session 表

属性名称	属性类型	主键	外键	默认值	属性说明
session_key	int	是			Sessiond 的 key
session_data	varchar(10)				Session 的值
expire_date	varchar(6)				过期时间

表 4-9　universitylist_continent 大洲表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	int	是			逻辑主键
name	varchar(200)				名字
short_name	varchar				简写的名字

表 4-10　universitylist_country 国家表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	int	是			逻辑主键
name	varchar(40)				名字
short_name	varchar(20)				简写的名字
continent_id	int		是		对应的洲的 id
is_delete	tinyint				是否被删除
update_time	datetime				更新时间
create_time	datetime				创建时间

表 4-11　universitylist_project 项目表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	Int	是			逻辑主键
name	varchar(500)		是		名字

short_name	varchar(10)				简写的名字
TOEFL	int				所需托福成绩
IELTS	int				所需雅思成绩
GRE	Int				所需 gre 成绩
degree	Int				学位
time_required	Int				项目时间
is_delete	Tinyint				是否被删除
update_time	Datetime				更新时间
create_time	Datetime				创建时间

表 4-12　universitylist_university 院校表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	int	是			逻辑主键
name	varchar(1000)				名字
latest_rank	Int				最近的排名
city	varchar(50)				所在的城市
country	varchar(50)				国家
address	varchar(500)				详细地址
website	varchar(500)				网址
summary	varchar(100000)				摘要
longitude	Double				经度
latitude	Double				维度
date_url	varchar(1000)				数据源地址
create_time	datetime				创建时间
update_time	datetime				更新时间

表 4-13　universitylist_university_indicator_and_subject_rankings 专业排名表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	int	是			逻辑主键
name	varchar(1000)				名字
key	varchar(500)				排名种类
value	varchar(100)				排名
create_time	datetime				创建时间

update_time	datetime				更新时间

表 4-14　universitylist_university_more 院校补充信息表表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	int	是			逻辑主键
name	varchar(1000)				名字
key	varchar(500)				分数种类
value	varchar(100)				分数
create_time	datetime				创建时间
update_time	datetime				更新时间

表 4-14　user 用户表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	int	是			逻辑主键
password	varchar(128)				密码
last_login	datetime				最近一次的登陆时间
is_superuser	int				是否为管理员用户
first_name	varchar(150)				名
last_name	varchar(30)				姓
email	varchar(150)				电子邮箱
is_staff	iinyint				是否工作
is_active	iinyint				账号是否激活
date_joined	datetime				加入时间
country	varchar		是		国家
school_type	tinyint		是		学校类型
degree	tinyint		是		学位
degree_type	varchar(1)		是		学位类型
graduate_school	varchar(1)		是		毕业院校
ielts	int				雅思成绩
toefl	int				托福成绩
fee	int				费用预算
Age	int				年龄
more_informations	varchar(100)				摘要

sex	tinyint				性别
discipline_competition	tinyint				是否有竞赛经历
club_activity	tinyint				是否有社团活动经历
research_experience	tinyint				是否有科研经历
work_experience	tinyint				是否有工作经历
create_time	datetime				创建时间
update_time	datetime				更新时间
is_delete	tinyint				是否删除

表 4-15　user_group 用户组表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	int	是			逻辑主键
user_id	int				用户 id
group_id	int				用户组 id

表 4-16 user_user_permissions 用户权限表

属性名称	属性类型	主键	外键	默认值	属性说明
Id	int	是			逻辑主键
user_id	int				用户 id
permission_id	int				权限 id

5 系统实现

5.1 系统注册入口实现

留学信息检索系统部署到服务器上面运行，通过点击注册按钮进入用户注册界面，按照要求输入帐号、密码和电子邮箱即可进入系统。主要是用来给用户提供一些个性化服务，如图 5.1 所示。

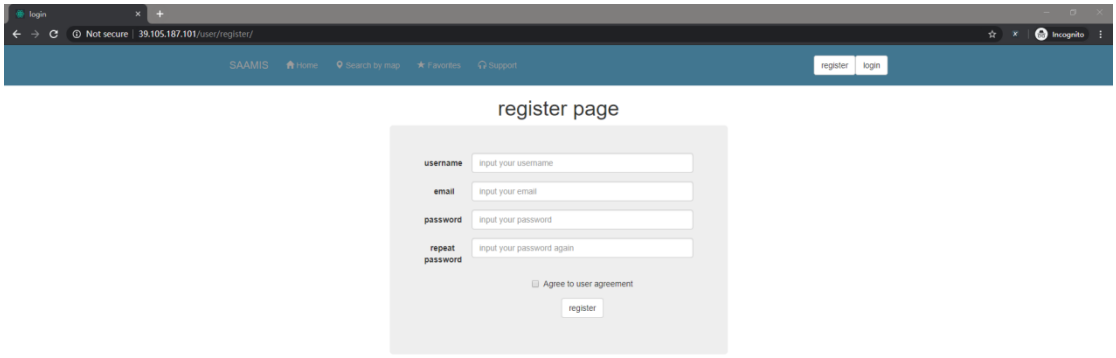

图 5.1 留学信息检索系统注册界面

5.2 系统登录入口实现

留学信息检索系统成功部署到服务器上面运行以后，通过点击登录按钮进入用户登录界面，按照要求输入用户名和密码即可进入系统，如图 5.2 所示。

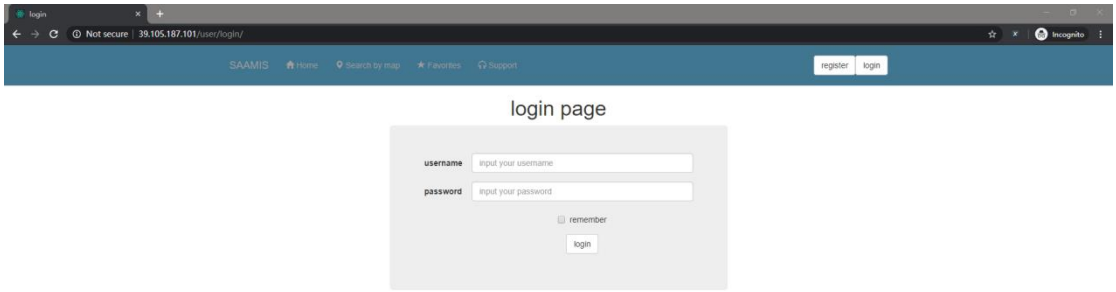

图 5.2 留学信息检索系统登录界面

5.3 系统主界面实现

留学信息检索系统成功部署到服务器上面运行以后，在浏览器中输入地址便会进入首页，在首页中，有 1500 所高等院校，这里以按照 usnews 全球排名来进

行排名，以每页 12 个院校的方法显示在首页。在院校列表的上面，有一个搜索框，用户可以输入关键词检索自己所需要的信息，搜索是一个日益重要的话题。用户越来越依赖于搜索从噪声信息中分离和快速找到有用信息。为此，这里使用了 Haystack 整合自定义搜索，使其尽可能简单的灵活和强大到足以处理更高级的用例。这里使用了 whoosh 作为搜索引擎，它可以大大的降低噪声，从而提高检索的效率。如图所 5.3 示。

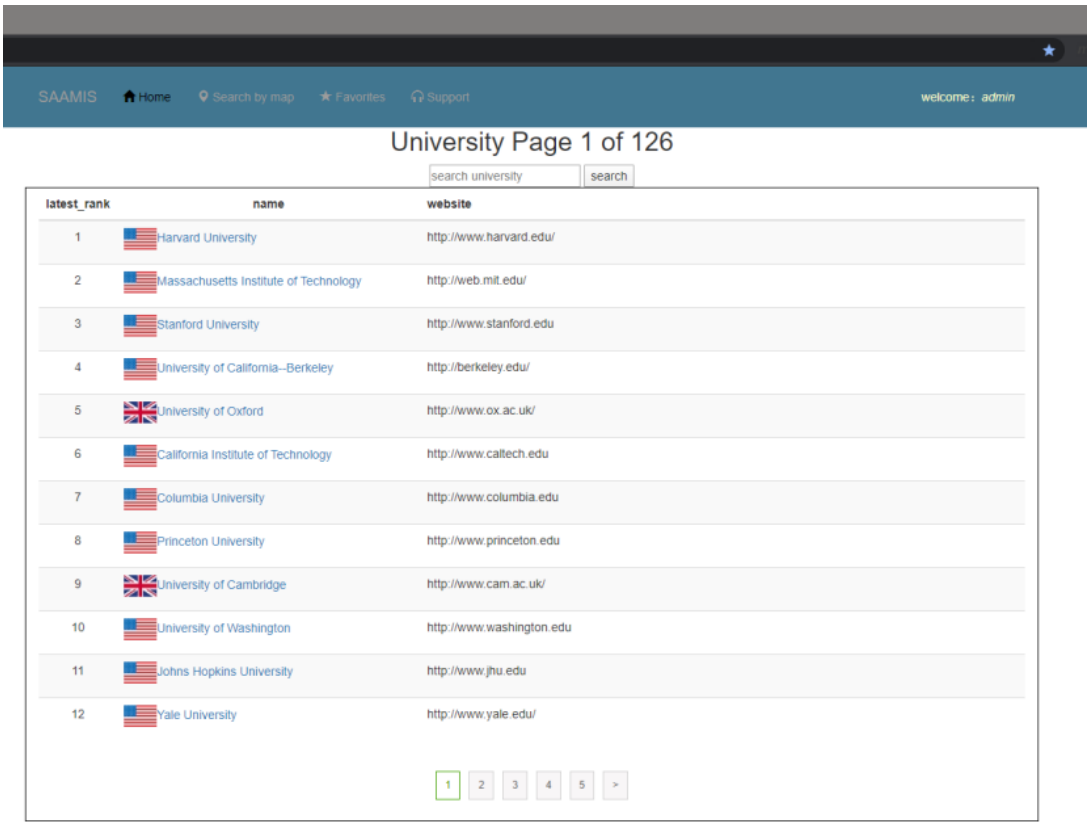

图 5.3 留学信息检索系主界面

5.4 按照地图检索实现

将 1500 院校按照经纬度渲染在题图上，使得用户可以从地图上直接的了解到院校的位置，然后可以任意点击自己感兴趣的院校然后进入详细页面。渲染结果如图 5.4 所示。

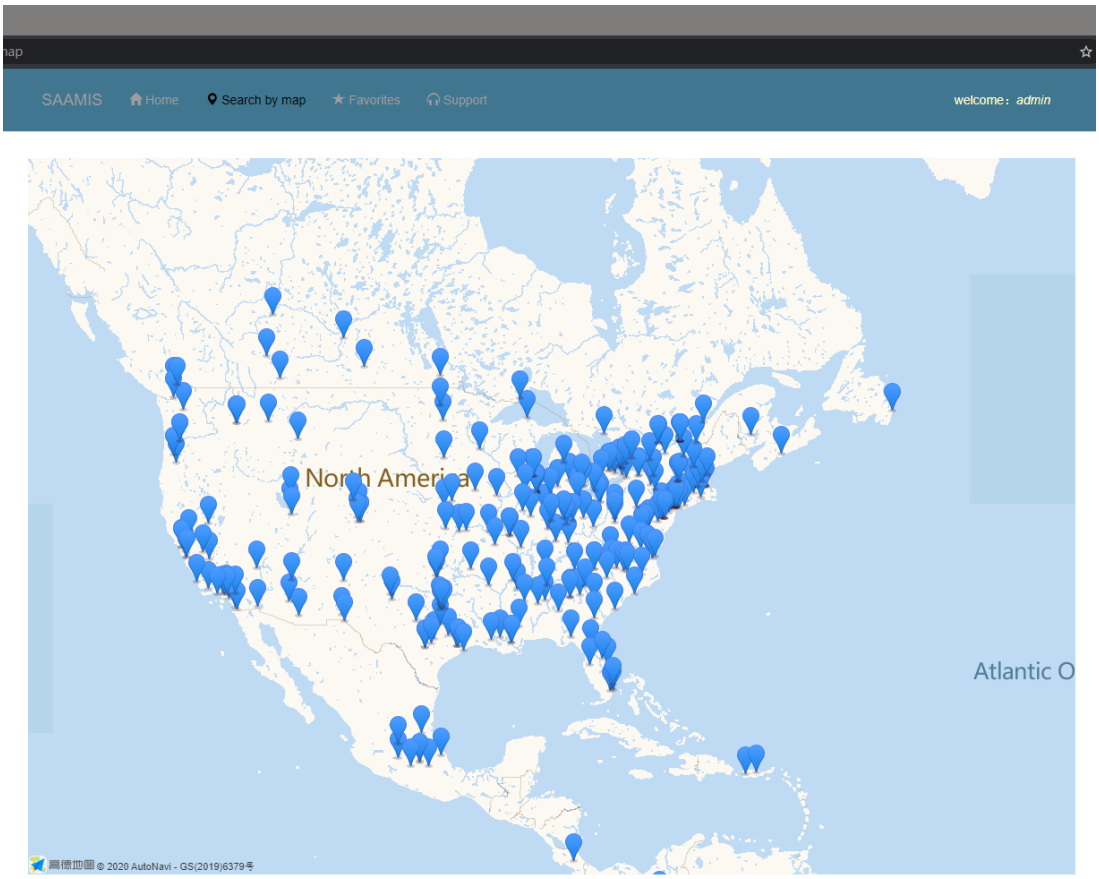

图 5.4 留学信息检索系统地图检索界面

5.5 用户详细信息设置

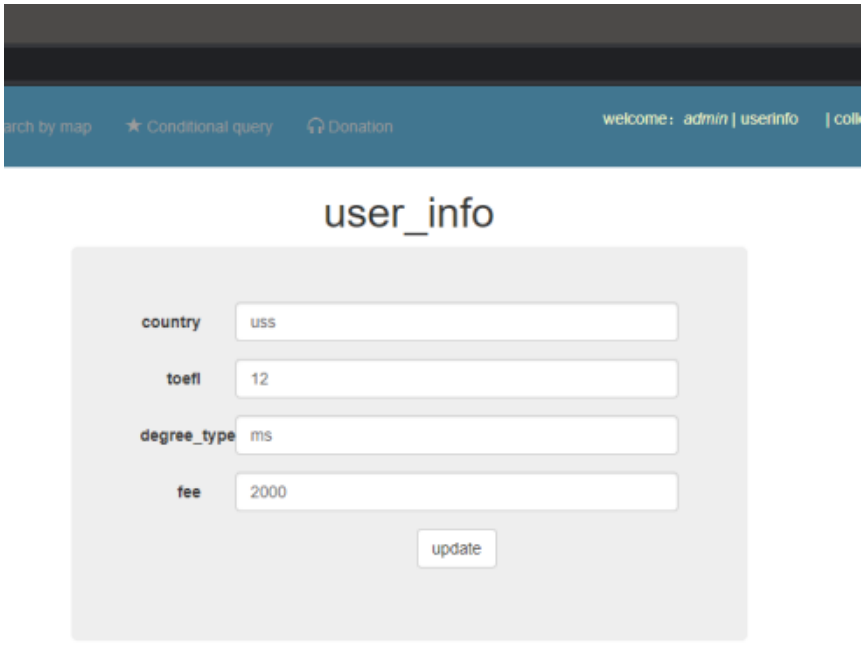

图 5.5 留学信息检索系统用户中心界面

在注册的时候，用户仅仅设置了用户名和密码以及邮箱等信息，还有许多基本信息没有设置，用户可以通过用户中心来设置用户的其他信息，如图 5.5 所示。

5.6 收藏的院校查看

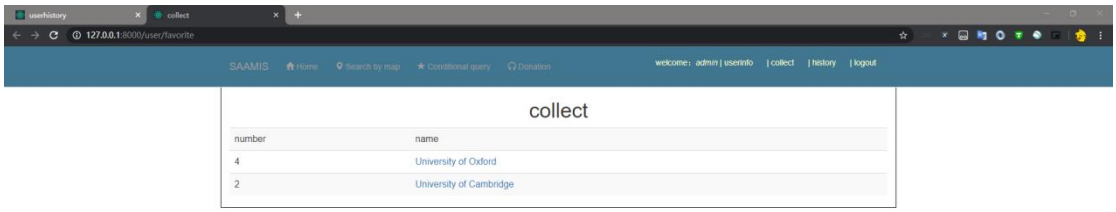

图 5.6 留学信息检索系统用户收藏界面

在浏览院校的时候，用户可以选择自己感兴趣的院校进行收藏，然后在收藏页面进行浏览，如图 5.6 所示。

5.7 浏览历史查看

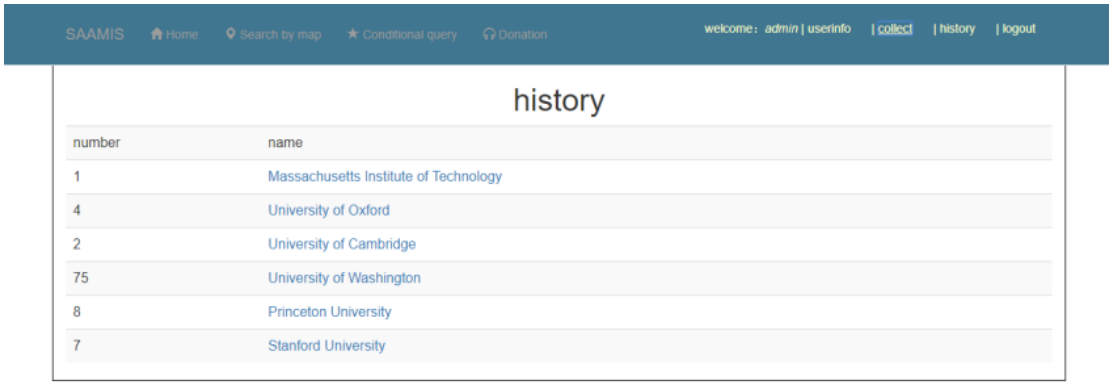

图 5.7 留学信息检索系统用户浏览历史界面

户可以通过此页面查看自己的浏览历史，如图 5.7 所示。

5.8 自定义条件查询

通过用户输入的条件进行检索，比如排名，国家的信息，结果会以列表的形式返回。

5.9 管理员功能

管理员可以增加删除以及修改数据，如图 5.8，5.9，5.10 所示。

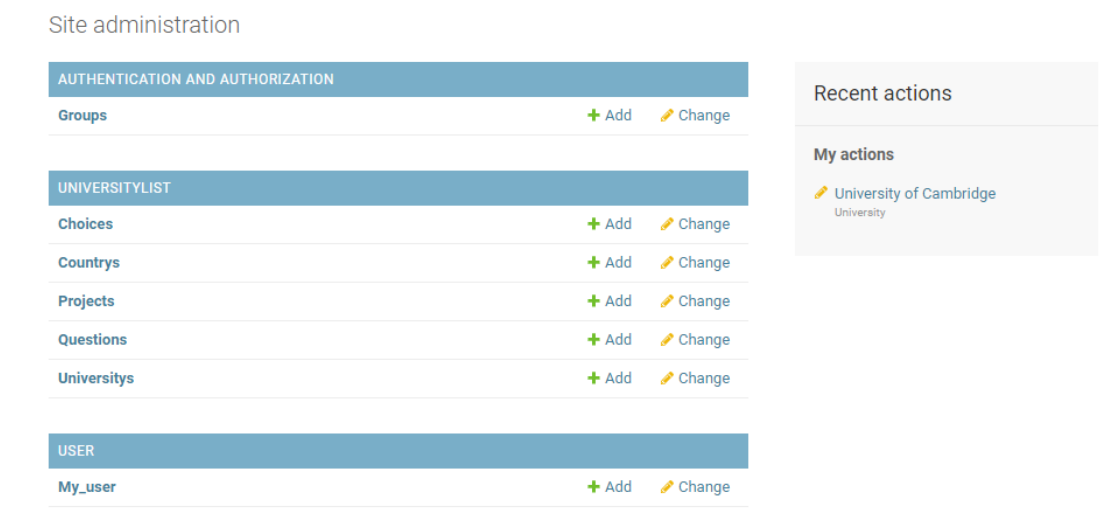

图 5.8 管理员主页面界面

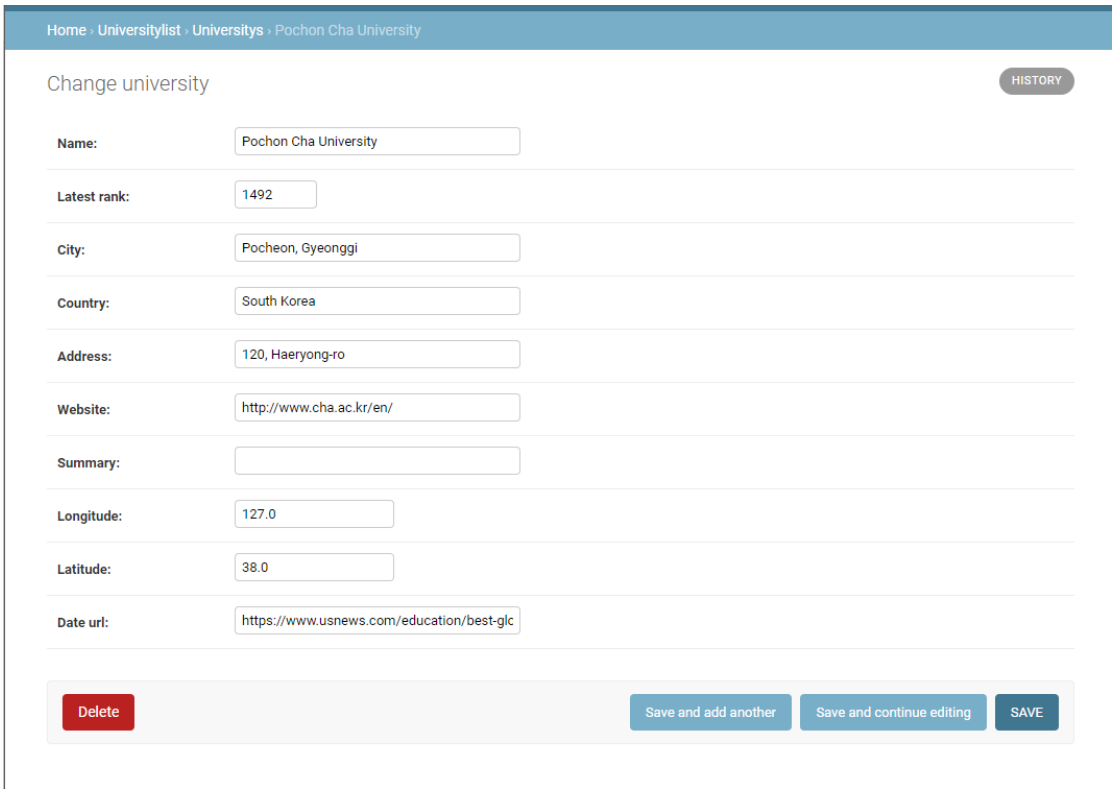

图 5.9 管理员修改删除界面

Home › Universitylist › Universitys

Select university to change

Action: --------- Go 0 of 100 selected

UNIVERSITY
Universidade Tecnologica Federal do Parana
Yangtze University
Yantai University
Anhui Normal University
Universidad Veracruzana
University of South China
Liaoning University
University of Mazandaran
Universidade Federal Rural de Pernambuco (UFRPE)
Universite Paris Seine (ComUE)
Universidad Autonoma de Baja California
Catholic University of Daegu
Government College University Faisalabad
Dokkyo University
Texas Tech University Health Sciences Center
Kaunas University of Technology
Ondokuz Mayis University
Pochon Cha University
Umm Al-Qura University
Guangdong Medical University
University of Malaysia Perlis
Shiga University of Medical Science

图 5.10 管理员院校列表界面

5.10 系统的部署架构

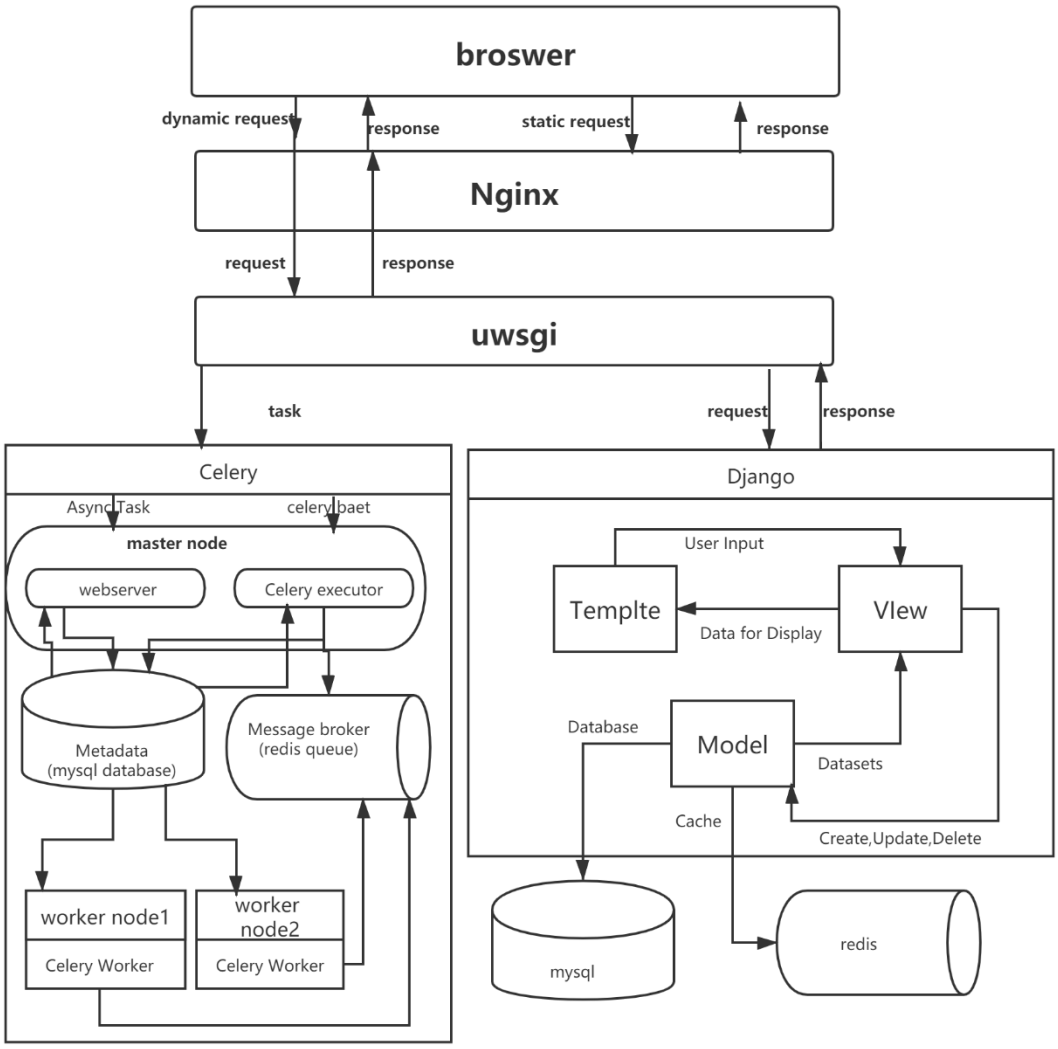

图 5.11 留学信息检索系统技术架构图

本系统采用 B-S 结构。服务器接收到从用户发过来的请求以后，先转交给 nginx 处理，如果是静态请求，nginx 会直接把静态文件返回给用户，如果是动态请求，nginx 会把请求转交给 uwsgi，然后由 uwsgi 调用 application 或者 Celery。Celery 接收到定时任务或者异步任务以后，通过消息中间件把任务传送给 worker 然后用 worker 来执行任务，最终由 nginx 把响应返回给用户。系统的技术架构图如图 5.11 所示。

6 测试

在整个留学信息检索系统开发完成以后，必须对系统进行全面的测试来验证其性能和功能的情况。本章对系统如何进行测试已经测试的方法进行介绍，通过测试系统中性能薄弱的部分和存在的错误，来修改系统的 bug，进而提高系统的稳定性和性能。

6.1 测试原理与准备

在软件设置完成后，要经过精致的测试，这样可以发现整个系统中存在的问题并加以修复，在测试的过程中分为单元测试，集成测试和系统测试，并且在测试中确定测试计划且严格按照计划执行，以此减少测试的随意性。

测试的目的是发现系统中的 bug，测试减少缺陷并控制在一定的程度内，而不是完全的消除 bug，系统测试的目的是用尽可能低的成本和较少的测试用例检测出系统中尽可能多的缺陷和错误，以此保证系统的质量。测试环境如表 6-1 所示。

表 6-1 留学信息检索系统测试环境面

名称	软件环境或版本
操作系统	Ubuntu18.04
CPU	Intel(R) Xeon(R) Platinum 8163 CPU @ 2.50GHz
内存	1G
服务器版本	nginx version: nginx/1.14.0 (Ubuntu)
Python 版本	3.7
数据库版本	mysql Ver 8.0.19 for Linux on x86_64 (MySQL Community Server - GPL)
浏览器版本	Google Chrome Version 80.0.3987.149 (Official Build) (64-bit)

6.2 测试用例及结果

6.2.1 功能测试

功能测试也成为行为测试或者黑盒测试。在测试的过程中只考虑功能能否正常使用，而不深入到系统内部的代码，在测试的过程中仅输入数据，然后等

待输出结果，最后将此结果和说明书中的结果进行对比。以此来检验程序是否正确。在测试的过程中需要使用产品说明书或者需求文档，只要知道正常情况是输入用例对应的输出结果，才能够确认测试得出的结果是否正确。所以测试者可以通过有效或者无效的输入来判断对应的输出是否正确。功能测试主要发现整个系统的功能错误。系统的测试用例表如表 6-2 所示。

表 6-2 功能测试用例表

序号	名称	预计结果	是否通过
1	用户注册	用户名已经存在，注册成功	是
2	用户登录	用户名和密码不匹配	是
3	激活激活	激活成功并成功跳转	是
4	用户注销	注销成功并成功跳转	是
5	地图查询	成功在地图上渲染出院校	是
6	排名查询	成功返回	是
7	条件查询	成功返回检索到的结果	是

6.2.2 性能测试

为了达到验证系统的可靠性定量要求而对系统进行测试的方法软件可靠性测试。测试的方法主要是按照用户实际使用系统的方式进行测试。测试的目的是发现系统在实际部署中的 bug，并排除故障使其正常运行。

对于一个 web 系统来说，网页打开的速度是系统性能测试中最重要和最直观的一部分。压力测试是性能测试的一部分，通过压力测试可以决定性能的瓶颈，当访问量过大时，性能就会明显下降。

本系统采用 loadrunner 测试，测试结果如表 6-3 所示。

表 6-3 性能测试用例表

考察项	响应时间	结果
打开主页时间	<1 秒	通过
地图检索时间	<1 秒	通过
条件检索时间	<1 秒	通过
登录时间	<1 秒	通过
注册时间	<1 秒	通过
CPU 使用率	<70%	通过
内存使用率	<70%	通过

6.2.2 可靠性测试

为了验证软件的可靠性测试而对软件进行的测试成为软件可靠性测试。测试的方法是按照用户实际使用的方法进行测试，测试的目的是发现影响软件可

靠性的因素并排除。可靠性测试结果如表 6-4 所示。

表 6-4 可靠性测试用例表

可靠性测试	
操作	长时间运行，多人在线操作系统
预期结果	系统正常运行，没有崩溃。
测试结果	通过。

7 结论

随着科技的进步以及技术的不断更新迭代，社会的信息化进度大大加快，本系统为申请人提供快捷高效、安全可信的信息检索服务，相比于传统的中介人工服务，大大的提高了学生在申请海外院校的过程中搜集信息的效率。

本系统所包含的海外院校数据库极其丰富，这也是本系统于传统的搜索引擎相比得到的优势。通过本系统，可以为用户提高院校查询，专业项目查询，导师查询，用户浏览历史，用户收藏以及学费和院校对比等信息。

本系统采用了当下流行的 MVC 的三层设计模式，设计开发了用户模块和院校模块。在开发的过程中学到了很多知识，大大的提高的编程技能，在熟练掌握 python 框架的基础上，研究了 celery 等异步组件的源码，技术架构以及实现方式，并成功的运用到本系统中。

在编写系统以及撰写论文的过程中，我对于解决代码中遇到的技术上的问题的能力和在学习生活中遇到的对问题进行分析的能力以及解决问题的能力有了大大的提高。以下是我对本次毕业设计的总结：

7.1 研究结论

本系统通过部署后运行测试，结果表明可以正常运行。本系统部署在一个 Nginx 服务器上可以满足一个服务器正常的用户访问量。输入的 URL 地址如果没有权限强行访问会跳到有权限的页面，因此安全性能有一定的保障；系统访问出错的时候会进行统一的异常处理并显示指定带有错误信息的页面给用户。这个系统最可以让申请海外院校的用户在检索海外院校的过程中提高检索效率和质量，也在一定程度上解决了信息孤岛的问题。其中数据来源均来自 1500 所院校的官网，并且采用爬虫技术实时更新数据，以保持数据的时效性。

7.2 创新之处

本系统的主要优势在于数据源质量高，且时效性强，用户界面容易操作。其数据源来自院校官网。

通过对传统的搜索引擎和中介网站以及大量用户进行调研，了解到了申请过

程中所要搜集的信息以及整个申请流程，在可行性分析的基础上做出了合理的需求分析，大大的增强了用户的体验。

在编写文档的过程中，进一步加强了对软件工程的工作流程的理解，使用了UML 对系统进行建模，并精心设计了系统模块图和用例图。

采用当下流行的框架进行开发，此外，在开发的过程中还采用了 jQuery 提高了前端的编写效率，采用 celery 根据数据的修改自动生成 html 页面，大大的减少了数据库的查询次数，大大减少了系统的响应时间。采用模板继承减少了重复代码的编写，使得最终在整个项目上大大提高了工作效率。使用账号来实现不同角色控制权对数据的增删查改，采用实时生成的图片的技术在前端动态的渲染。支持多个不同角色的用户在线进行操作；

7.3 存在问题

虽然我这次的毕业设计留学信息检索系统已经完成了，但是由于自身知识方面、技术水平和花费时间等因素，还是存在许多不足之处，如页面设计较为简陋、特别是排版不够美观，数据库中的数据质量仍然有很大的提升空间。我也在尽自己所能地最大程度解决以上问题，让留学信息检索系统更加完善。

7.4 后续研究方向

随着科技日新月异的发展，本系统作为一个长期运行的系统来说，还有以下几点可以进行改进：□系统性能。目前系统的访问量还不够高，本系统在目前的这种情况下，完全可以应对。但是随着用户的数量的增加，本系统能否应对更高的并发量以及数据库能否处理更多的数据还是个很大的问题。在今后会对本系统的各个模块进行更加全面的测试，找出系统访问的高峰期以及用户最常访问的页面及功能，并找出对策来进行优化，以此来提高系统稳定性和健壮性。□代码重构。进一步优化系统的架构和方法，引入更加成熟的开发方式、尽可能的将代码封装成模块方便使用。□增加数据量。采用自动化方法周期性对数据执行抓取，提高数据实时性和丰富程度。

参考文献

[1] 张波.自费出国留学中介研究[D].上海：华东师范大学教育学系，2009.

[2] 刘佳睿.基于瓦联网的公派留学管理信息软件平台的设计与实现[D].西安：电子科技大学，2016.

[3] 袁利.基于聚类的协同过滤个性化推荐算法研究[D].武汉：华中师范大学，2014.

[4] 颜石磊.来华留学生招生信息系统的设计与实现[D].西安：电子科技大学，2012

[5] 向学蔚.留学中介市场开发策略研究[D].武汉：华中师范大学，2015

[6] 张奇.留学申请服务系统的设计与实现[D].沈阳：辽宁大学，2016

[7] 何　俊，李慧颖.基于 Web 数据挖掘的个性化留学信息推荐系统研究与应用[J].信息与电脑，2018，11：103-107

[8] 陈 镭，张凡龙. 基于 Django 的高校人才引进系统设计与实现[J]. Computer Era,2019,7:40-42

[9] 葛宇航. 基于 Django 的留学生信息管理系统设计与实现[J]. 通信设计与应用,2019,8:35-36

[10] MAO Jun,PENG Hong,MENG Li-min, The design and implementation for download software of PDF report in the vehicle location system based on Django [J]. Zhejiang Provincial Key Laboratory of Optical Fiber Communication Technology.2012(4).

[11] 董海兰. 基于 Python 的非结构化数据检索系统的设计与实现[D].南京：南京邮电大学，2016

[12] 周鸿超 . 实时数据处理技术在网络安全管理系统中的研究与应用[D].北京：北京邮电大学，2016.

[13] 苗丹国. 我国自费出国留学政策的持续性发展与趋势研究[J].江苏师范大学学报，2013，6：1-12

[14] 刘丽.关系数据库的时态信息检索方法研究[D].西安：西安科技大学，2019.

[15] 王海彬.数学表达式检索结果文档排序[D].保定：河北大学，2019.

[16] 李文涛.基于本体的初中生物学习资源库建设及语义检索探索[D].昆明：云南师范大学，2019.

[17] 江穹峰. 面向河湖空间大数据的混合架构存储与检索系统研究及应用[D].武汉：华中科技大学，2019.

[18] 闫铁珊. 基于关键字和类型信息的实体检索方法研究[D].西安：西安理工大学，2019.

[19] 崔森. 基于中文检索纠错的航天情报系统的研究与实现[D].西安：电子科技大学，2019.

[20] 施琦. 面向知识库的知识问答技术研究[D].哈尔滨：哈尔滨工业大学，2019.

致谢

四年的本科学习生活到这里基本上结束了，回首这四年，不禁感慨万千。记得四年前来到白云学院的时候，自己的心中有着很多的想法，也给自己定下了很多目标，但是后来懵懵懂懂，但幸运的是，在各位老师的指导下，我逐渐找到了自己的人生方向。

在这四年里，我学习到了很多知识，除了与信息管理有关的知识以外，更重要的是我找到了适合自己的学习方法。与老师、同学们的相处让我学会面对苦难以及敢于承担责任。四年的学习生活中，我经历了很多，我忘不了在学习和生活中遇到的困难和迷茫，这所有的一切都将是我人生中的宝贵财富。四年来，各位老师和同学对我的鼓励和帮助铭记在我心中。

如今，四年的本科生活已经结束，我的毕业设计是我四年学习的最终成果。毕业设计包含了我这四年所学的知识。在本文中，邱老师给了我很多指导，没有邱老师，我很难完成这个设计。

最后，感谢邱老师这四年来对我的教导，向您致以最诚挚的敬意。

附录 相关代码

#用户注册发送邮件

#####异步邮件代码

```
# 使用celery
from universitylist.models import *
from django_redis import get_redis_connection
# 创建一个Celery类的实例对象
app = Celery('celery_tasks.tasks', broker='redis://127.0.0.1:6379/8')

# celery -A celery_tasks.tasks worker -l info 任务执行端使用次命令执行函数 windows： celery -A celery_tasks.tasks worker -l info -P eventlet
# 定义任务函数
@app.task
def send_register_active_email(email, username, token):
    '''发送激活邮件'''
    # 组织邮件信息
    subject = 'welcome to STUDYABROADAPPLICATION'
    message = ''
    sender = settings.EMAIL_FROM
    receiver = [email]
    html_message = '<h1>%s, welocme to join a member of STUDYABROADAPPLICATION</h1>please click the link below to active your account<a href="http://%s/user/active/%s">link</a> <br/> <h1>http://%s/user/active/%s</h1>' %(username,settings.MY_HOST,token,settings.MY_HOST,token)
    send_mail(subject,message,sender,receiver,html_message=html_message)


@app.task
def generate_static_index_html():
    '
roject.

Generated by 'django-admin startproject' using Django 3.0.2.

For more information on this file, see
https://docs.djangoproject.com/en/3.0/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.0/ref/settings/
"""
```

#索引文件更新代码

```
import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

            'ENGINE': 'haystack.backends.whoosh_backend.WhooshEngine',
            # 索引文件路径
            'PATH': os.path.join(BASE_DIR, 'whoosh_index'),
        }
}
# 当添加、修改、删除数据时，自动生成索引
HAYSTACK_SIGNAL_PROCESSOR = 'haystack.signals.RealtimeSignalProcessor'
HAYSTACK_SEARCH_RESULTS_PER_PAGE    =    15



try:
    from .settings_location import *
except ImportError as e:
            print(e.args)
#####project url
"""studyabroadapplication URL Configuration

    path('favicon.ico',RedirectView.as_view(url='/static/img/favicon.ico')),
]
```

#####model

```
from django.contrib import admin
from django.core.cache import cache
            verbose_name = 'my_user'
            verbose_name_plural = verbose_name
    def __str__(self):
            return self.username
##### search index
# 定义索引类
from haystack import indexes
# 导入你的模型类
from universitylist.models import University
class Bpi(ModelViewSet):
    # queryset是一个查询数据的查询集，存储这所有的数据库查询之后的数据
    queryset = Country.objects.all()
    serializer_class = UserInfoSerializer
    # serializer_class用来指定在当前的视图里面进行  序列化与反序列时使用的序列化器（串行器）
    # 主页视图
#@cache_page(60*15)
```

```
def index(request):
    """
    context = cache.get('index_page_data')
    if    context is None:
        #universities = University.objects.order_by('id')
        countries = Country.objects.all().order_by("-update_time")
        context={'countries_cache':countries,}
        cache.set('index_page_data',context[countries_cache], 3600)
    """
    limit = 12
    #universities = University.objects.order_by('id')
    universities = University.objects.all().order_by('latest_rank','id')
    paginator = Paginator(universities, limit)
    page = request.GET.get('page')   # 获取页码

    try:
        u
    try:
        university = University.objects.get(id=university_id)
    except University.DoesNotExist:
        return redirect(reverse('universitylist:index'))
    is_collect = False
    if user.is_authenticated:
        con = get_redis_connection('default')
        history_key = 'history_%d'%user.id
        university_ids = con.lrange(history_key, 0, -1)
        if(str(university_id).encode() in university_ids):
            con.lrem(history_key,0,university_id)
        con.lpush(history_key,university_id)
    #university = University.object
        collec_key='collect_%s' % user.id
        if(str(university.id).encode() in con.lrange(collec_key,0,-1)):
            is_collect = True
    context={'university':university,'is_collect':is_collect}
    return render(request, 'university_detail.html', context)
```

```
con = get_redis_connection('default')
history_key = 'collect_%d' % user.id
university_ids = con.lrange(history_key, 0, -1)
university_list = []
for id in university_ids:
    university = University.objects.get(id=int(id))
    university_list.append(university)
```

```
        ## 组织上下文
        context = {'university_list':university_list,}

        #除了你给模板文件传递的模板变量之外，django 框架会把 request.user 也传给模板文件
        return render(request, 'collect.html', context)

class LoginView(View):
    def get(self, request):
        '''显示登录页面'''
        # 判断是否记住了用户名
        if 'username' in request.COOKIES:
            username = request.COOKIES.get('username')

            if user.is_active:
                # 用户已激活
                # 记录用户的登录状态

                login(request, user)

                # 跳转到首页
                response = redirect(reverse('universitylist:index')) # HttpResponseRedirect

                # 判断是否需要记住用户名
                remember = request.POST.get('remember')

                if remember == 'on':
                    # 记住用户名

    #def get(self, request):
    #    return render(request, 'login.html')
    #def post(self, request):
    #    email=request.POST.get('email')
    #    password=request.POST.get('password')
    #    if not all([email,password]):
    #        return render(request,'login.html',{'errmsg':"请输入用户名和密码"})
    #    try:
    #        user=User.objects.get(email=email,is_active=1)
    #    except User.DoesNotExist:
    #        user=None
    #    if not user:
    #        return render(request,'register.html',{'errmsg':'"邮箱不存在请先<a href="/user/register/"    style="margin-top:8px" id="register-button">注册</a>'"})
```

```
        #       if(user.password!=password):

class ActiveView(View):
    '''用户激活'''
    def get(self, request, token):
        '''进行用户激活'''
        # 进行解密，获取要激活的用户信息
        serializer = Serializer(settings.SECRET_KEY, 3600)
        try:
            info = serializer.loads(token)
            # 获取待激活用户的 id
            user_id = info['confirm']

            # 根据 id 获取用户信息
            user = User.objects.get(id=user_id)
            user.is_active = 1
            user.save()

            # 跳转到登录页面
            return redirect(reverse('user:login'))
        except SignatureExpired as e:
            # 激活链接已过期
            return HttpResponse('<h1>激活链接已过期</h1>')


class UserInfoView(LoginRequiredMixin,View):
    '''用户中心-信息页'''
    def get(self, request):
        # 获取用户的个人信息
        user = request.user
        con = get_redis_connection('default')
        history_key = 'history_%d' % user.id
        university_ids = con.lrange(history_key, 0, 4)
        university_list = []
        for id in university_ids:
            university = University.objects.get(id=int(id))
            university_list.append(university)
        return render(request, 'user_info.html', context)
class CollectView(View):
    def post(self,request):
        user = request.user
        user.email = email
        user.password = password
        user.save()
```

```
            #return HttpResponse("register successful")
            return redirect(reverse('universitylist:index'))
wrapt==1.11.2
zipp==0.6.0
zope.interface==4.6.0
```

usegi

Uwsgi 配置代码

```
[uwsgi]
#使用nginx连接时使用
socket=127.0.0.1:8080
#直接做web服务器使用  python manage.py runserver ip:port
#http=127.0.0.1:8080
#项目目录
chdir=/root/studyabroadapplication
#项目中wsgi.py文件的目录，相对于项目目录
wsgi-file=studyabroadapplication/wsgi.py
#指定启动的工作进程数
processes=4
#指定工作进程中的线程数
threads=2
master=True
#保存启动之后主进程的pid
pidfile=uwsgi.pid
#设置uwsgi后台运行，uwsgi.log保存日志信息
daemonize=uwsgi.log
#设置虚拟环境的路径
virtualenv=/root/.virtualenvs/studyabroadapplication
```