重庆三峡学院毕业设计（论文）

题目：基于Python的毕业设计系统

专 业：软件工程

年 级：2016级

学 号：201606084128

作 者：周雅培

指导老师：邹细涛

完成时间：2020年4月

目录

[摘要 I](#_Toc38512379)

[Abstract II](#_Toc38512380)

[第1章 绪论 1](#_Toc38512381)

[1.1 课题背景及意义 1](#_Toc38512382)

[1.2 国内外研究现状 1](#_Toc38512383)

[1.3系统设计目标 1](#_Toc38512384)

[1.4 本文结构 1](#_Toc38512385)

[第2章 可行性分析 2](#_Toc38512386)

[2.1 技术可行性分析 2](#_Toc38512387)

[2.2 操作可行性 2](#_Toc38512388)

[2.3 经济可行性 2](#_Toc38512389)

[第3章 需求分析 3](#_Toc38512390)

[3.1系统总体需求分析 3](#_Toc38512391)

[3.2系统详细需求分析 3](#_Toc38512392)

[3.3 系统数据库需求分析 3](#_Toc38512393)

[3.3.1 数据流图 3](#_Toc38512394)

[3.3.2数据字典的描述 9](#_Toc38512395)

[第4章 系统开发工具及使用技术介绍 17](#_Toc38512396)

[4.1 系统开发工具 17](#_Toc38512397)

[4.1.1 PyCharm 17](#_Toc38512398)

[4.1.2 MySQL 17](#_Toc38512399)

[4.1.3 NaviCat Premium 17](#_Toc38512400)

[4.2 系统使用技术 17](#_Toc38512401)

[4.2.1 Python编程语言 17](#_Toc38512402)

[4.2.2基于MVT架构的Django框架 18](#_Toc38512403)

[4.2.3 FastDFS分布式存储 18](#_Toc38512404)

[4.2.4 Nginx 19](#_Toc38512405)

[第5章 系统设计 20](#_Toc38512406)

[5.1系统软硬件环境 20](#_Toc38512407)

[5.1.1 系统硬件环境 20](#_Toc38512408)

[5.1.2 系统软件环境 20](#_Toc38512409)

[5.2系统总体设计 20](#_Toc38512410)

[5.2.1系统概述及功能 20](#_Toc38512411)

[5.3系统详细设计 21](#_Toc38512412)

[5.3.1 注册管理 21](#_Toc38512413)

[5.3.2 登录管理 21](#_Toc38512414)

[5.3.3 题目管理 22](#_Toc38512415)

[5.3.4 选题管理 23](#_Toc38512416)

[5.3.5 开题管理 23](#_Toc38512417)

[5.4系统数据库设计 23](#_Toc38512418)

[5.4.1 概念设计 24](#_Toc38512419)

[5.4.2 逻辑设计 24](#_Toc38512420)

[5.4.3 物理设计 25](#_Toc38512421)

[第6章 系统实现 28](#_Toc38512422)

[6.1毕业生角色实现 28](#_Toc38512423)

[6.1.1选题实现 28](#_Toc38512424)

[6.1.2 开题实现 28](#_Toc38512425)

[6.1.4 答辩实现 28](#_Toc38512426)

[6.2 毕业指导老师功能实现 29](#_Toc38512427)

[6.3 教学秘书功能实现 29](#_Toc38512428)

[第7章 系统测试 30](#_Toc38512429)

[结束语 31](#_Toc38512430)

[参考文献 31](#_Toc38512431)

[致 谢 31](#_Toc38512432)

基于Python的毕业设计系统

周雅培

（重庆三峡学院计算机科学与工程学院软件工程专业2016级 重庆万州 404000）

摘要：毕业设计流程繁琐，指导老师与学生之间沟通不够灵活，各种毕业设计文档管理混乱。本毕业设计系统采用了现代的技术方式，提升了质量效率，降低了沟通成本和时间成本。

本系统经过实际需求分析，基于B/S架构，采用Django框架、MySQL数据库、FastDFS分布式文件存储系统、Bootstrap前端框架，设计并完成了包括题目管理子系统、选题管理子系统、开题管理子系统、答辩管理子系统、成绩管理子系统、用户权限管理子系统和注册、登录、反馈等各模块，其设计均按照简单灵活，安全稳定的原则来实现。

关键词：内容管理系统；B/S模式；Django框架；MySQL数据库；FastDFS分布式系统

Graduation design system based on Python

Zhou Ya-pei

(Grade 2016, software engineering, School of Computer Science and Engineering,

Chongqing Three Gorges University, Wanzhou, Chongqing 404000 )

Abstract： Considering the intricate feature of graduation design process, inflexible communication between tutor and student, and discombobulated management of various design files, this graduation design system adopts the counting way of our time. Thus, quality is ameliorated, communication cost reduced, and time line prioritized.

This system, been through practical demand analysis, is basded on Browser/Server Architecture, and adopts various technics, such as Django framework, MySQL database, FastDFS distributed file storage system, Bootstrap front frame. This program has been designed and completed arduously, including management subsystems like Subject, Topic selection, Opening, Reply, Score, User authority, and modules like Registration, Login, Feedback, all of them employing the principles of simplicity, flexibility, security and stability.

**Key words:** CMS；B/S mode；Django framework；MySQL；FastDFS

# 第1章 绪论

## 1.1 课题背景及意义

随着社会的发展，科技的进步、互联网的高速发展，计算机在各个领域中都得到了广泛应用。当今社会已经步入了数字化时代。计算机正在潜移默化地改变着人们的生活学习，人们也习惯了信息共享带来的便利。在高校的日常工作中，每年临近毕业，都有大量应届毕业生需要进行毕业设计。在这一过程中，选题、开题等各个阶段都可能因为指导老师带领的学生过多导致选题混乱，指导老师难以统计学生选题等问题。不但效率低、工作繁琐，且容易出错。而毕业设计作为本科阶段的关键环节,非常有必要对其施行网络化管理。现代计算机技术是实现信息化的良好手段与载体。

毕业设计系统作为内容管理系统（Content Management System，简写为CMS），其主要目的是以信息共享为核心，面向大量的信息处理，集信息数字化、分布式存储、管理、查询、传播于一体，从内容的创建、采集到传递等的完整整合。毕业设计系统通过分布式的文件存储和数据库实现资料和信息的管理、存储操作。

毕业设计系统应包括题目管理、选题管理、开题管理等核心业务。合适的毕业设计系统能够简化用户在选题、开题、题目管理、答辩、成绩判定、反馈过程中的操作，降低毕业生、指导老师、教学秘书之间的沟通成本，提供更高效的服务。

## 1.2 国内外研究现状

当前国内已有很多高校建立了毕业设计管理系统,提升了管理的效率,方便了教师和学生。然而,部分毕业设计管理系统在设计时存在一些问题,主要表现在:(1)功能存在缺陷,部分系统只关注选题管理,对整个毕业设计过程没有管理措施;(2)采用的技术较为落后,有很多系统采用ASP技术结合Access数据库设计而成，在执行效率、可维护性、安全性等方面已不能满足新的要求;(3)缺少针对多用户并发访问和系统安全问题的解决方案。因此,采用新技术和更完善的解决方案来设计毕业管理系统显得很有必要。

## 1.3系统设计目标

毕业设计系统设计之初的要求

1. 设计并实现一个基于python的毕业设计交流网站。
2. 实现毕业设计全过程（包括选题、开题、答辩、反馈等）的管理。

## 1.4 本文结构

本文的结构分为：绪论、可行性分析、需求分析、开发工具及使用技术介绍、系统设计、系统实现、总结、参考文献、致谢。

# 第2章 可行性分析

本章旨在分析毕业设计系统开发的可行性和需求。

## 2.1 技术可行性分析

技术可行性分析主要分为软、硬件两方面：

软件方面：本系统基于Python语言进行开发，采用流行的Django框架，前端采用HTML语言开发。上述技术开源，无学习门槛，其应用场景广泛，完全可以完成本系统的开发任务。开发工具采用PyCharm，可兼容使用以上技术。

硬件方面：本系统采用B/S架构，没有对客户端环境的硬性要求，浏览器接入网络即可使用，现今绝大多数PC都可以完成上述任务；服务端的架设也只需要满足配置要求的计算机或者云服务器即可完成。

故技术上不构成障碍。

## 2.2 操作可行性

对于操作部分，各个功能模块已经有了详细的划分，其流程清晰，功能明确，相应的，界面简洁大方，操作简单易懂，只需要极其基础的电脑操作水平，就可以完全操作本系统。

## 2.3 经济可行性

本系统功能相对简单，开发难度小，技术工具皆来自于开源，仅有服务器部署和开发中过程的人力消耗会产生费用，但不论是云服务器还是局域网服务器，其成本都很低，而人力成本因为开发难度小所以也相对低，就经济上而言是完全能达到要求的。

# 第3章 需求分析

## 3.1系统总体需求分析

本系统面向用户群体为教学人员和应届学生，分为教师、教学秘书和学生，学生和教师在前台执行业务，教学秘书在后台部署业务。学生能进行登录、选题、查看个人信息、提交文档和评价等操作，教师能进行出题、上传文档等操作，教学秘书则能进行相关数据信息的管理和部署。

## 3.2系统详细需求分析

学生主要功能**：**

（1）登录：根据用户名密码进行登录。

（2）选题：查看所有题目并选择毕业设计题目。

（3）开题：下载题目相关文档模板，上传开题报告。

（4）上传中期报告：上传中期报告相关文档。

（5）答辩：答辩申请、上传相关文档。

（6）反馈：对指导老师的评价。

教师主要功能：

（1）题目管理：对题目进行增删查改。

（2）题目状态管理：修改题目确认状态。

（3）中期报告管理：下载中期报告相关文档

（4）答辩：上传相关文档，记录答辩过程。

（5）成绩管理：对毕业生进行评分，计算答辩总成绩。

（6）反馈：对毕业生进行评价。

教学秘书主要功能：

（1）登录：根据用户名密码进行登录。

（2）用户信息，用户权限，题目，题目状态的管理。

## 3.3 系统数据库需求分析

### 3.3.1 数据流图

（1）题目管理的数据流图

教学秘书在管理员界面界面将相关的题目信息进行输入，然后查询得到结果，再使用查询过的信息，进行更改和删除操作。

教师在教师页面输入相关题目信息，得到查询结果（或者添加新的题目），再使用查询过的信息，进行更改和删除操作。具体过程如图3-2和图3-3：

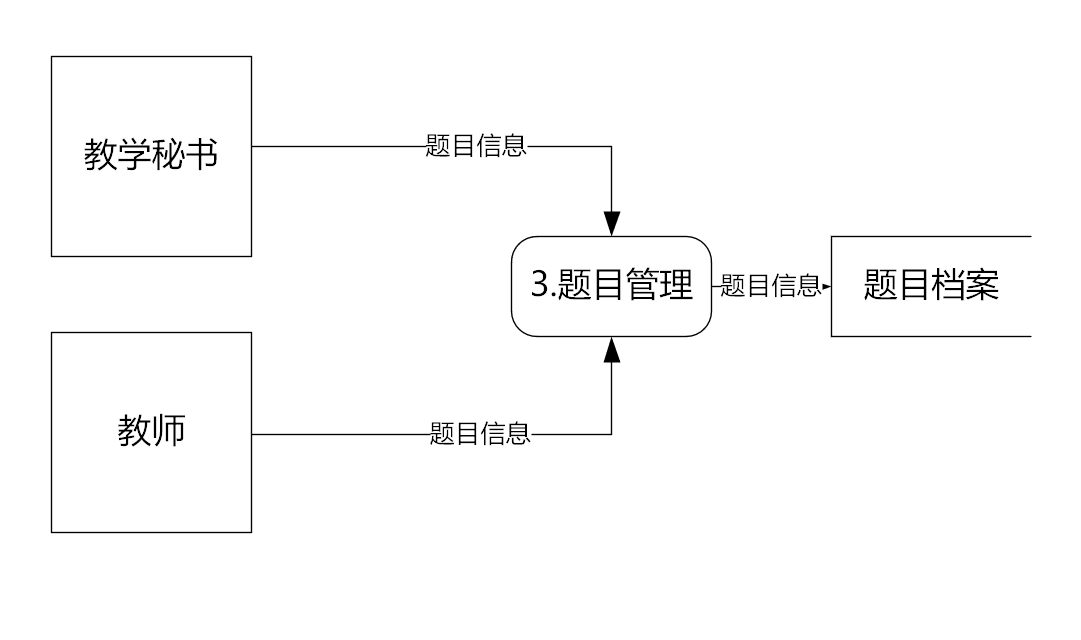


图3-2 题目管理1层数据流图

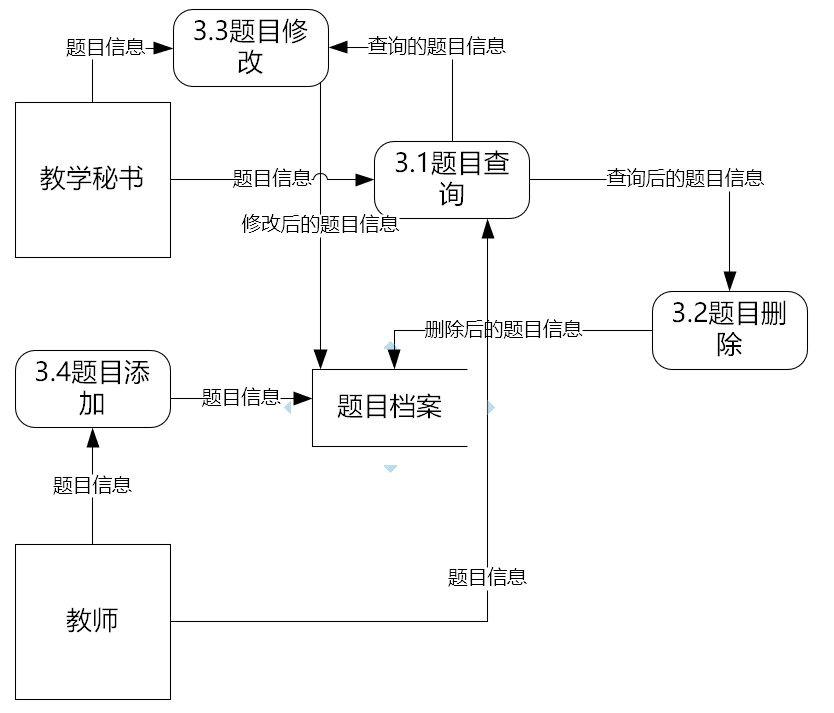


图3-3 题目管理2层数据流图

（2）选题管理的数据流图

学生选择题目，教师同意学生选题，教学秘书最终确认题目信息。随后对选题状态进行更改并更新用户与题目的关系。具体过程如图3-4和图3-5：

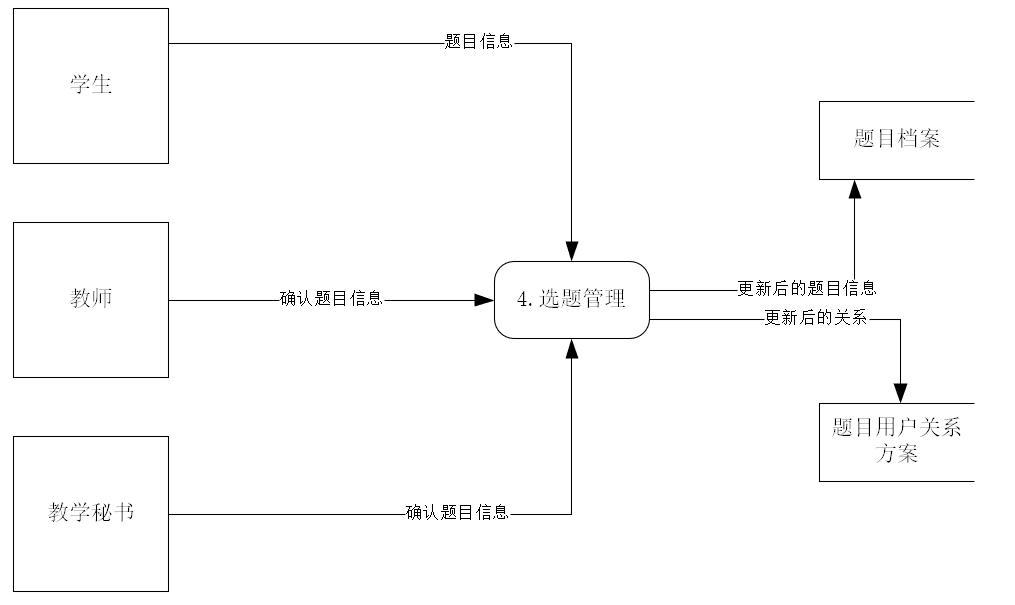


图3-4 选题管理1层数据流图

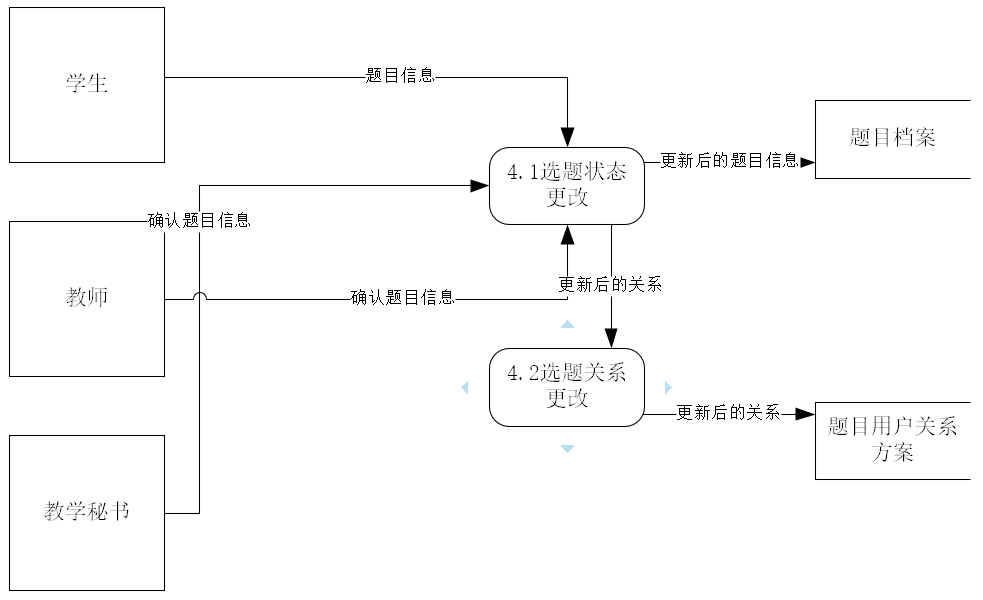


图3-5 选题管理2层数据流图

（3）开题管理的数据流图

教师/学生在操作界面将相关的文件进行上传，再使用查询过的信息，进行下载操作。具体过程如图3-6和图3-7：

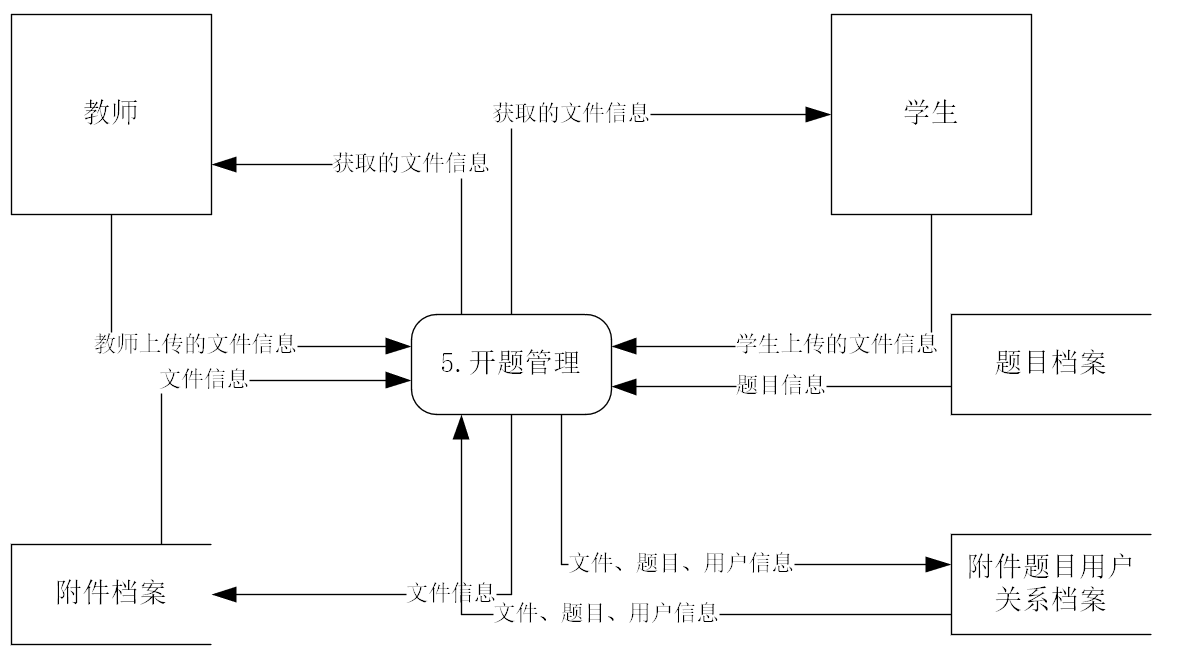


图3-6 开题管理1层数据流图

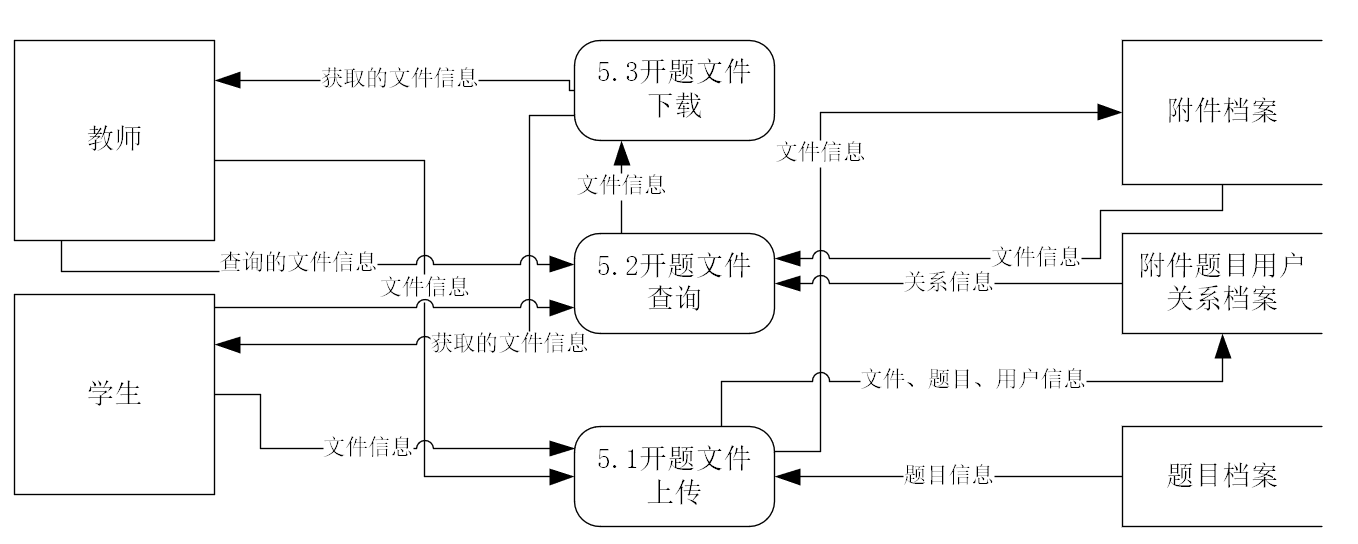


图3-7 开题管理2层数据流图

（4）答辩管理的数据流图

学生申请答辩，上传相关文档信息到数据库，教师确认答辩，记录答辩过程，生成文档并上传到数据库。具体过程如图3-8、图3-9和图3-10：

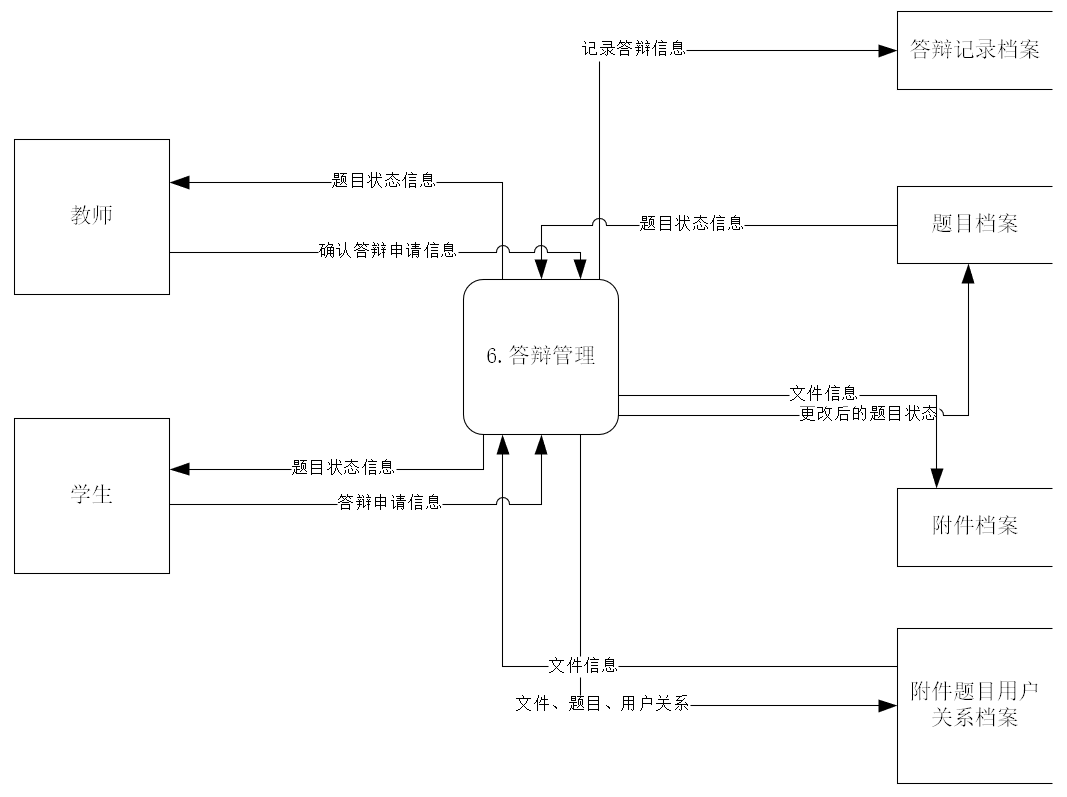


图3-8 答辩管理1层数据流图

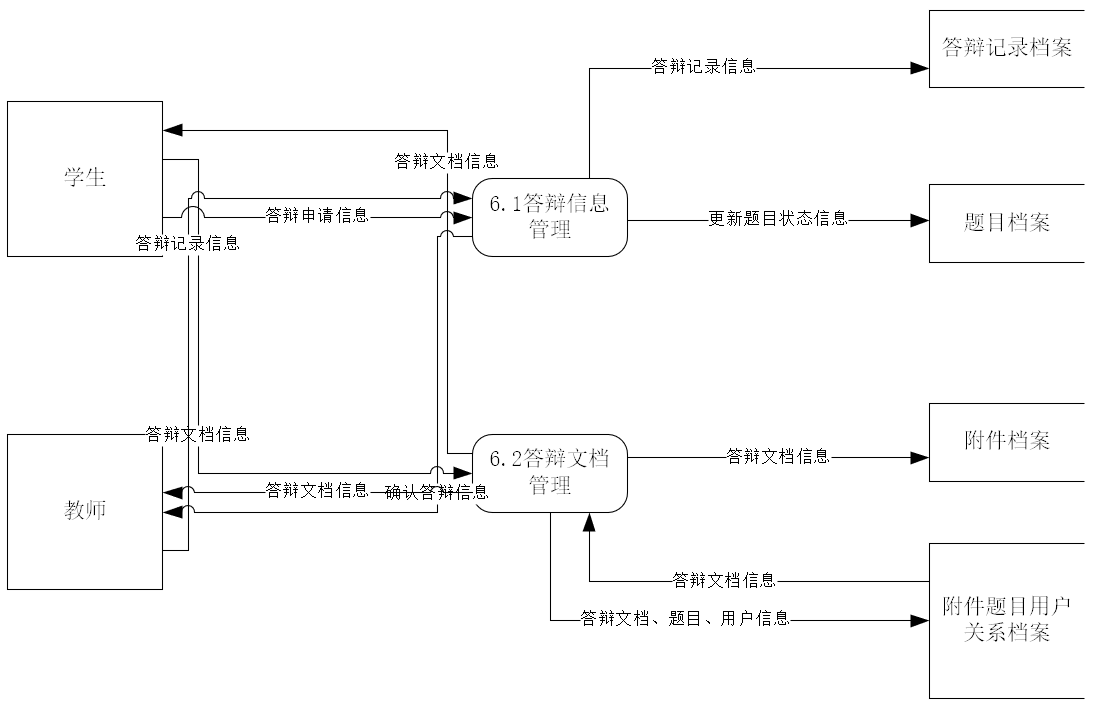


图3-9 答辩管理2层数据流图

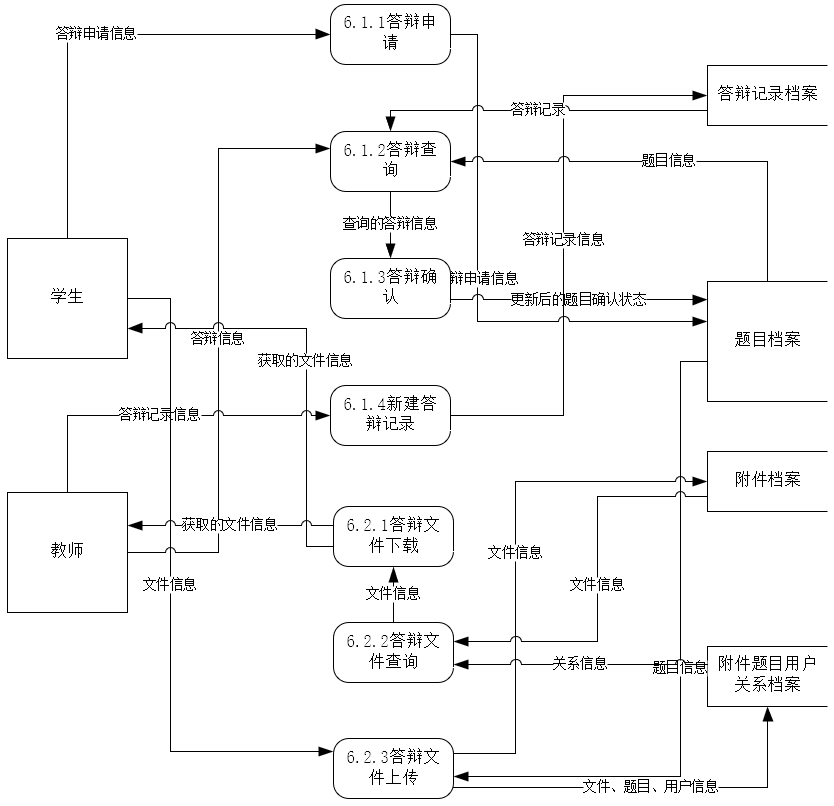


图3-10 答辩管理3层数据流图

（5）反馈管理的数据流图

学生、教师使用反馈管理将反馈信息存储到数据库，具体过程如图3-11和图3-12：

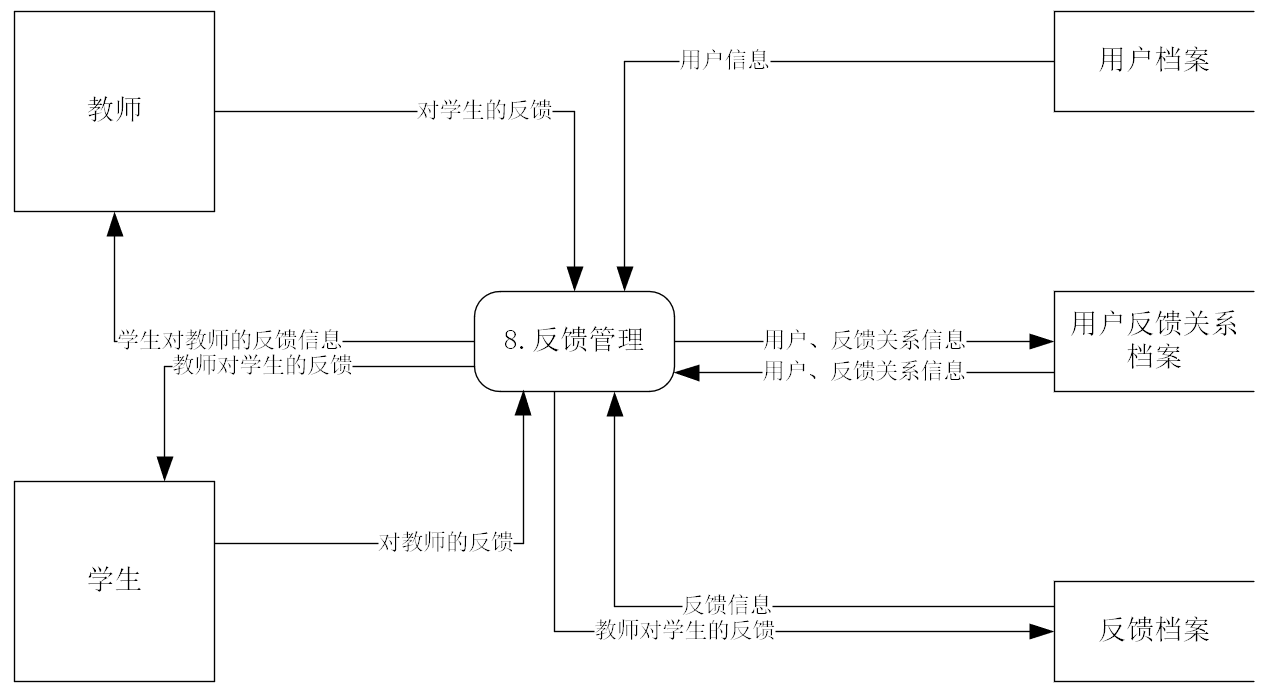


图3-11 反馈管理1层数据流图

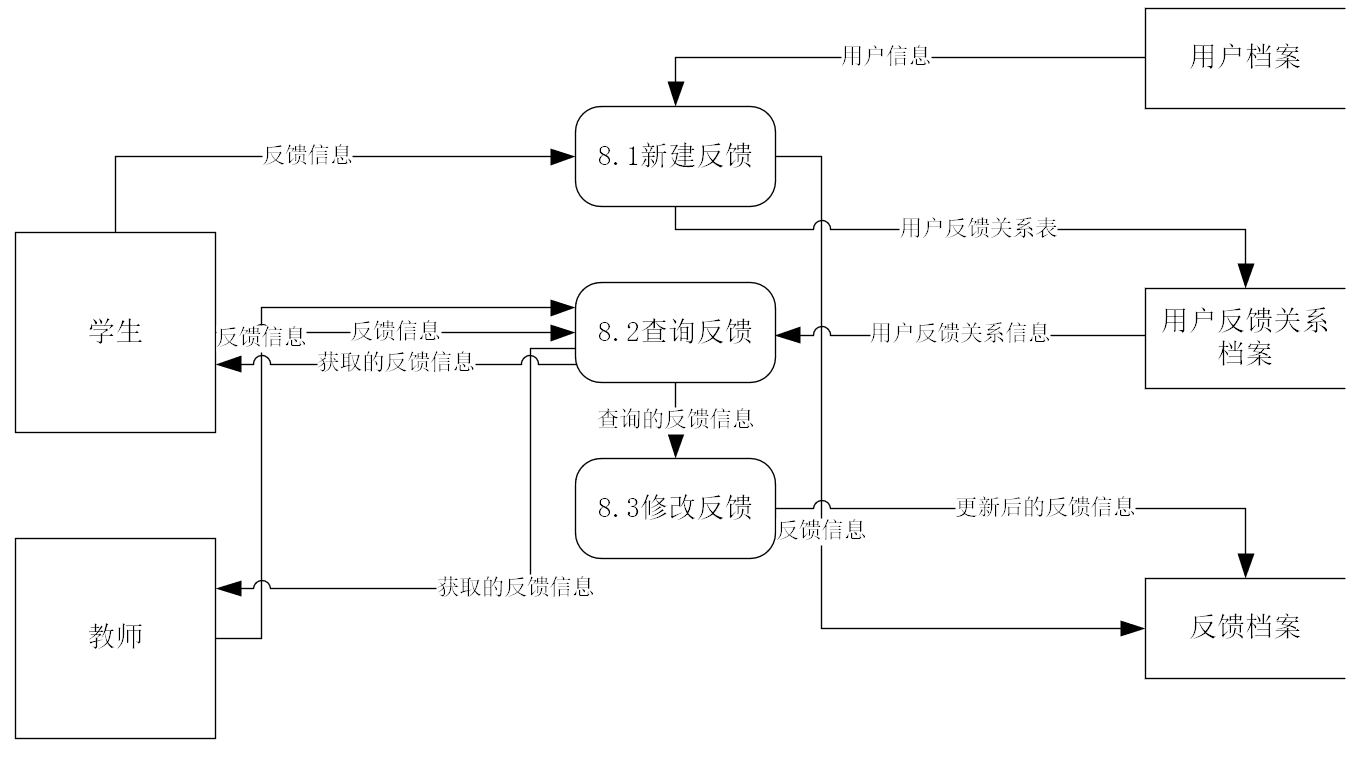


图3-12 反馈管理2层数据流图

### 3.3.2数据字典的描述

数据字典是关于数据的信息的集合，也就是对数据流图包含的所有元素的定义的集合。任何字典最重要的用途都是供人查阅对不了解的条目的解释，数据字典的作用也正是在软件分析和设计的过程中给人提供关于数据的描述信息。

下面给出了本系统的数据字典描述：

（1）登录信息

表3.1“登录信息”描述

|  |
| --- |
| 名 字：登录信息  说 明：登录系统的认证信息  来 源：用户档案  去 向：无  数据结构：  －－用户名  －－用户密码 |

（2）.注册信息

表3.2“注册信息”描述

|  |
| --- |
| 名 字：注册信息  说 明：注册系统的信息  来 源：教学秘书、教师、学生的操作  去 向：用户档案  数据结构：  －－用户编号  ­－－用户密码（加盐的哈希算法）  －－上次登录时间  －－是否是管理员  －－用户名  －－真实姓名  －－Email  －－是否是职工  －－是否激活  －－注册日期  －－学校  －－院系 |

（3）.题目信息

表3.3“题目信息”描述

|  |
| --- |
| 名 字：题目信息  说 明：题目的相关记录  来 源：教师，教学秘书，题目档案  去 向：题目管理、选题管理、开题管理、成绩管理、答辩管理、反馈管理  数据结构：  －－题目编号  －－题目名称  －－题目详情  －－限选人数  －－已选人数  －－上次编辑时间  －－选题状态  －－确认状态  －－成绩 |

（4）选题确认信息

表3.4“选题确认信息”描述

|  |
| --- |
| 名 字：选题确认信息  说 明：选题确认的情况  来 源：学生、教师、教学秘书  去 向：选题管理、题目档案  数据结构：  －－题目编号  －－选题状态 |

（5）答辩确认信息

表3.5“答辩确认信息”描述

|  |
| --- |
| 名 字：答辩确认信息  说 明：答辩确认的状况  来 源：学生、教师  去 向：答辩管理、题目档案  数据结构：  －－题目编号  －－答辩状态 |

（6）题目用户关系信息

表3.6“题目用户关系信息”描述

|  |
| --- |
| 名 字：题目用户关系信息  说 明：题目与用户之间的关系  来 源：题目管理、选题管理、开题管理、题目用户关系档案  去 向：选题管理、题目用户关系档案  数据结构：  －－关系编号  －－题目编号  －－用户编号 |

（7）文件信息

表3.7“文件信息”描述

|  |
| --- |
| 名 字：文件信息  说 明：上传或下载文件的信息  来 源：教师、学生、附件档案  去 向：教师、学生、附件档案  数据结构：  －－文件编号  －－文件地址  －－文件原始名称 |

（8）文件题目用户关系信息

表3.8“文件题目用户关系信息”描述

|  |
| --- |
| 名 字：文件题目用户关系信息  说 明：文件、题目与用户之间的关系信息  来 源：开题管理、答辩管理、文件题目关系档案  去 向：开题管理、答辩管理、文件题目关系档案  数据结构：  －－关系编号  －－题目编号  －－文件编号  －－用户编号 |

（9）成绩信息

表3.9“成绩信息”描述

|  |
| --- |
| 名 字：成绩信息  说 明：题目对应的成绩  来 源：教师  去 向：成绩档案  数据结构：  －－成绩编号  －－题目编号  －－成绩 |

（10）反馈信息

表3.10“反馈信息”描述

|  |
| --- |
| 名 字：反馈信息  说 明：教师和学生之间的相互反馈内容  来 源：教师、学生  去 向：反馈档案  数据结构：  －－反馈编号  －－反馈内容 |

（11）用户反馈关系信息

表3.11“用户反馈关系信息”描述

|  |
| --- |
| 名 字：用户反馈关系信息  说 明：反馈内容与用户之间的关系  来 源：反馈管理  去 向：用户反馈关系档案  数据结构：  －－关系编号  －－用户编号  －－反馈编号 |

（12）角色信息

表3.12“角色信息”描述

|  |
| --- |
| 名 字：角色信息  说 明：用户的角色  来 源：管理员  去 向：角色档案  数据结构：  －－角色编号  －－角色名称 |

（13）用户角色关系信息

表3.13“用户角色关系信息”描述

|  |
| --- |
| 名 字：用户角色关系信息  说 明：用户的角色  来 源：用户角色管理  去 向：用户角色关系档案  数据结构：  －－关系编号  －－用户编号  －－角色编号 |

（14）权限信息

表3.14“权限信息”描述

|  |
| --- |
| 名 字：权限信息  说 明：所有的权限  来 源：教学秘书  去 向：权限档案  数据结构：  －－权限编号  －－权限名称 |

（15）角色权限关系信息

表3.15“角色权限关系信息”描述

|  |
| --- |
| 名 字：角色权限关系信息  说 明：不同角色对应的权限  来 源：用户权限管理  去 向：角色权限关系档案  数据结构：  －－关系编号  －－角色编号  －－权限编号 |

（16）用户权限关系信息

表3.16“用户权限关系信息”描述

|  |
| --- |
| 名 字：童虎权限关系信息  说 明：不同用户对应的权限  来 源：用户权限管理  去 向：角色权限关系档案  数据结构：  －－关系编号  －－用户编号  －－权限编号 |

（17）用户档案

表3.17 “用户档案”描述

|  |
| --- |
| 数据存储名：用户档案  说 明：说明用户的相关信息  组 成：用户信息=用户编号+用户名+用户密码+上次登录时间+是否为管理员+真实姓名+是否为职工+是否激活+加入时间+学校+院系+Email  存 取 方 式：按用户编号存取 |

（18）题目档案

表3.18 “题目档案”描述

|  |
| --- |
| 数据存储名：题目档案  说 明：说明题目的相关信息  组 成：题目信息=题目编号+题目名称+题目详情+限选人数+已选人数+选题确认状态+答辩确认状态+上次编辑时间+出题时间  存 取 方 式：按题目编号存取 |

（19）题目用户关系档案

表3.19 “题目用户关系档案”描述

|  |
| --- |
| 数据存储名：题目用户关系档案  说 明：说明题目与用户之间关系的信息  组 成：题目用户关系=关系编号+题目编号+用户编号  存 取 方 式：按关系编号存取 |

（20）附件档案

表3.20 “附件档案”描述

|  |
| --- |
| 数据存储名：附件档案  说 明：说明上传文件的相关信息  组 成：文件信息=文件编号+文件地址+原始文件名  存 取 方 式：按文件编号存取 |

（21）附件题目用户关系档案

表3.21 “附件题目用户关系”描述

|  |
| --- |
| 数据存储名：附件题目用户关系  说 明：说明上传文件、题目、用户之间的关系  组 成：附件题目用户关系信息=关系编号+文件编号+题目编号+用户编号  存 取 方 式：按关系编号存取 |

（22）答辩记录档案

表3.22 “答辩记录档案”描述

|  |
| --- |
| 数据存储名：答辩记录档案  说 明：记录答辩内容  组 成：答辩记录信息=答辩记录编号+题目编号+答辩内容  存 取 方 式：按答辩记录编号存取 |

（23）题目成绩档案

表3.23 “题目成绩档案”描述

|  |
| --- |
| 数据存储名：题目成绩档案  说 明：说明每个题目的成绩  组 成：题目成绩信息=题目成绩信息编号+题目编号+成绩  存 取 方 式：按题目成绩信息编号，题目编号存取 |

（24）反馈档案

表3.24 “反馈档案”描述

|  |
| --- |
| 数据存储名：反馈档案  说 明：记录反馈内容  组 成：反馈信息=反馈编号+反馈内容  存 取 方 式：按反馈编号存取 |

（25）用户反馈关系档案

表3.25 “用户反馈关系档案”描述

|  |
| --- |
| 数据存储名：用户反馈关系档案  说 明：记录用户与反馈内容的关系  组 成：用户反馈关系信息=用户反馈关系编号+用户编号+反馈内容编号  存 取 方 式：按用户反馈关系编号存取 |

# 第4章 系统开发工具及使用技术介绍

## 4.1 系统开发工具

### 4.1.1 PyCharm

PyCharm是一款强大的Python集成开发环境，由JetBrains公司开发。

（1）智能编码辅助

它的智能代码编辑器为Python、JavaScript、TypeScript、HTML、CSS以及其它常用模板语言提供了一流的支持。PyCharm的智能搜索能够跳转到任意类、文件等，只需单击一下即可切换到声明、测试、用法、实现等。使用PyCharm能够快速安全的进行重构，如安全的重命名、删除方法，引入变量、或方法来重构代码。

（2）内置开发人员工具

PyCharm内置的开箱即用的工具集，如调试器、内置终端、VCS以及数据库工具继承、远程开发功能、SSH终端等。

（3）网络开发的支持

除了Python，PyCharm还对现代的Web开发框架如Django、Flask、Google App Engine等提供了出色的支持。

### 4.1.2 MySQL

MySQL是一款关系型数据库管理系统，使用最常用的SQL语言访问数据库。MySQL数据库体积小、速度快、代码开源、应用广泛。作为网站的数据库再适合不过。

### 4.1.3 NaviCat Premium

NaviCat是一款数据库开发工具，它能够进行数据传输、数据同步和结构同步，以此快速地迁移数据，减少开销。可视化的SQL生成器能够快速创建、编辑和运行SQL语句，而无需担心命令语法是否正确。

## 4.2 系统使用技术

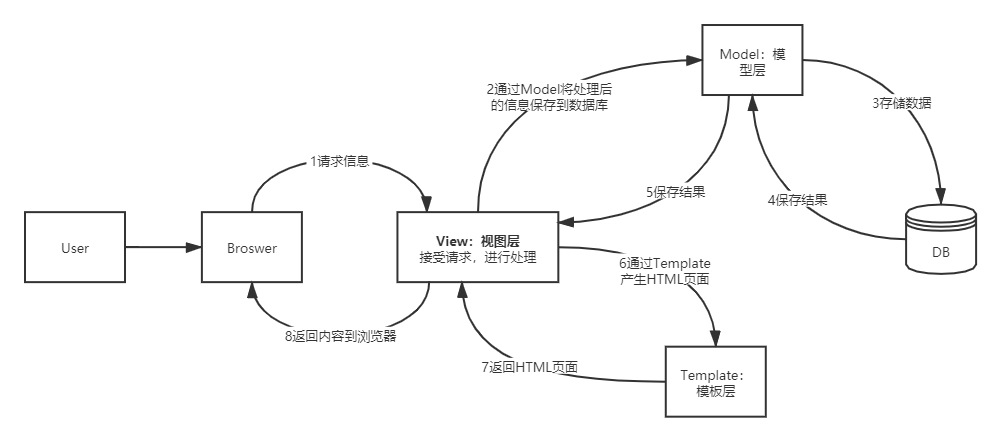
### 4.2.1 Python编程语言

Python 是一种具有动态语义的可解释、面向对象的高级编程语言。其高级内置数据结构，结合动态类型和动态绑定，使其对快速应用程序开发非常有吸引力，并用作脚本或胶合语言，将现有组件连接在一起。Python 简单易学的语法强调可读性，从而降低程序维护成本。

### 4.2.2基于MVT架构的Django框架

本系统后端编码采用了MVT架构，即模型层（Model），视图层（View），模板层（Template）。模型层用于构建和操作Web应用的数据，每一个模型都映射为数据库中的一张表。视图层负责处理用户的请求并返回响应。它对外接收用户请求，对内调度模型层和模版层，统合数据库和前端，最后根据业务逻辑，将处理好的数据，与前端结合，返回给用户。模板层由所需HTML输出的静态部分以及一些描述如何插入动态内容的特殊语法组成。

图4-1 MVT架构运行流程



### 4.2.3 FastDFS分布式存储

使用了FastDFS分布式文件存储并结合Nginx提高网站上传下载文件效率。FastDFS 架构包括Tracker server和Storage server。客户端请求Tracker server进行文件上传、下载，通过Tracker server调度最终由Storage server完成文件上传和下载。Tracker server作用是负载均衡和调度，通过 Tracker server在文件上传时可以根据一些策略找到Storage server提供文件上传服务。可以将tracker称为追踪服务器或调度服务器。Storage server 作用是文件存储，客户端上传的文件最终存储在Storage服务器上，Storageserver没有实现自己的文件系统而是利用操作系统的文件系统来管理文件。可以将storage称为存储服务器。

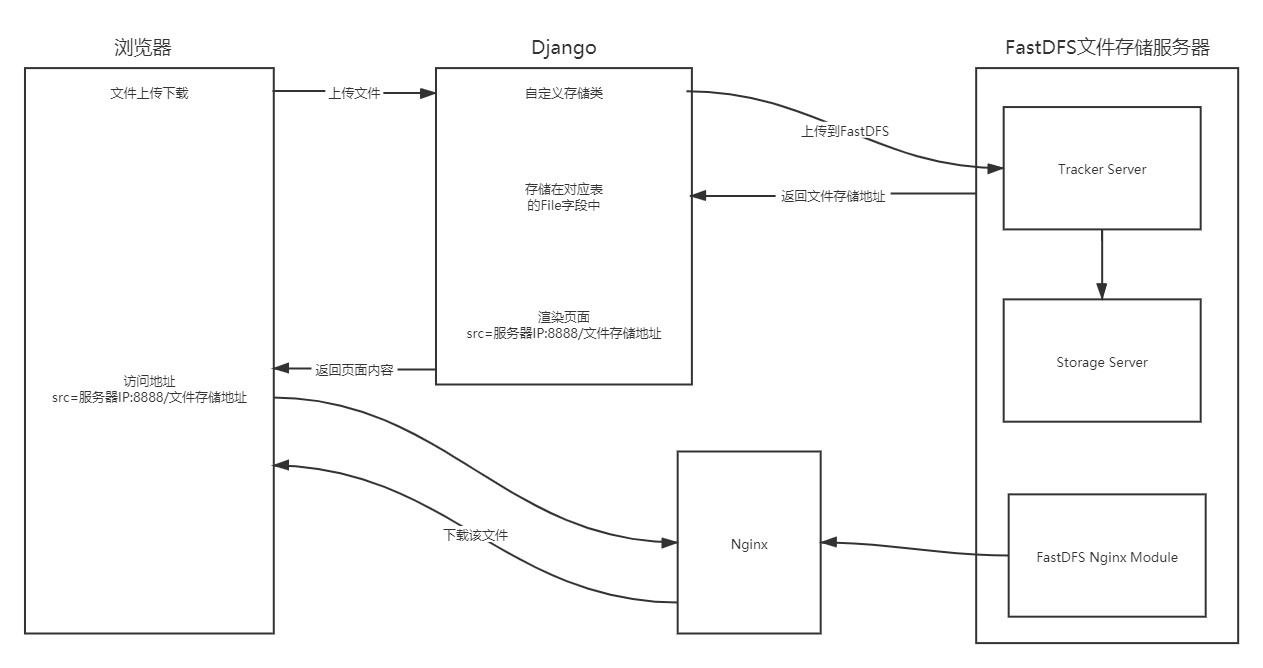


图4-2 FastDFS

### 4.2.4 Nginx

Nginx是用于 Web 服务、反向代理、缓存、负载平衡、媒体流等的开源软件。它最初是一个 Web 服务器，旨在实现最大的性能和稳定性。除了 HTTP 服务器功能外，NGINX 还可以充当电子邮件（IMAP、POP3 和 SMTP）的代理服务器以及 HTTP、TCP 和 UDP 服务器的反向代理和负载均衡器。

# 第5章 系统设计

## 5.1系统软硬件环境

### 5.1.1 系统硬件环境

CPU：双核2.0ghz及以上。

内存：2G及以上。

磁盘空间：20G及以上。

### 5.1.2 系统软件环境

操作系统：Windows 7,8,10、Linux、MacOS。

浏览器：Chrome、FireFox、Opera、Safari、Edge。

## 5.2系统总体设计

### 5.2.1系统概述及功能

（1）学生子系统

以学生身份登录系统的用户将进入学生子系统。可以对自身档案信息，成绩信息进行查看，可以使用选题管理和答辩管理对题目档案的状态进行修改，可以使用开题管理对文件档案进行新增和查看，可以使用反馈管理系统对对应指导老师的反馈进行新增和查看。

（2）教师子系统

以教师身份登录系统的用户将进入教师子系统。教师可以对自身档案进行查看，通过题目管理对题目档案进行增删改查，可以使用选题管理对题目档案的状态进行修改，可以使用开题管理和选题管理对文件档案进行新增和查看，可以使用答辩管理对答辩档案进行查询和新增，可以使用成绩管理对学生成绩进行查询和新增，可以使用反馈管理系统对对应学生的反馈进行新增和查看。

（3）教学秘书子系统

以教学秘书身份登录系统的用户进入管理员子系统 ，可以对题目档案进行删查改，对文件档案、用户、角色、权限进行增删查改。

（4）注册模块

管理用户的注册，对用户档案进行新增。

（5）登录模块

管理用户的登录。

以上5个主要组成模块所具有的主要功能模块如图5-1所示。

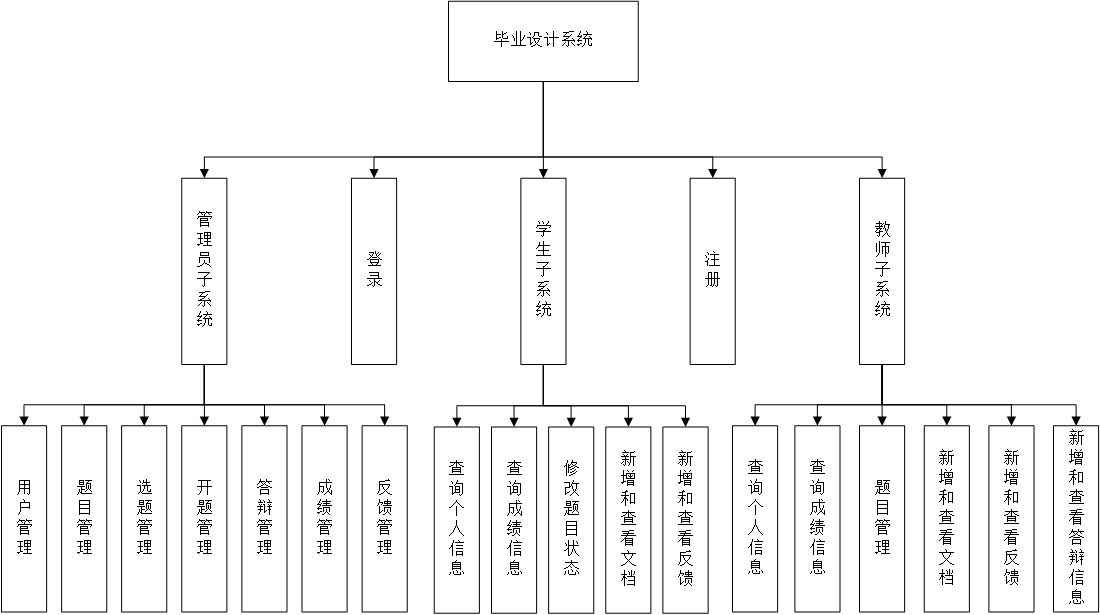


图5-1 软件结构层次图

## 5.3系统详细设计

### 5.3.1 注册管理

注册管理负责用户档案的增加以及用户角色的初始化。

模块的处理流程如下图：

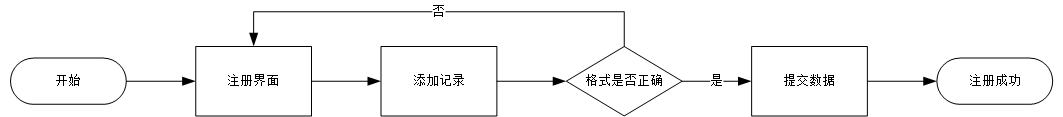


图5-3 注册管理

### 5.3.2 登录管理

登录管理负责用户的登录验证。

模块的处理流程如下图：

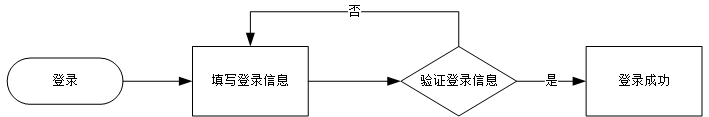


图5-5 登录管理

### 5.3.3 题目管理

题目管理是对题目档案进行增删查改，对于学生，能够查看所有题目；对于教师，能够增删改查题目。

模块的处理流程如下图：

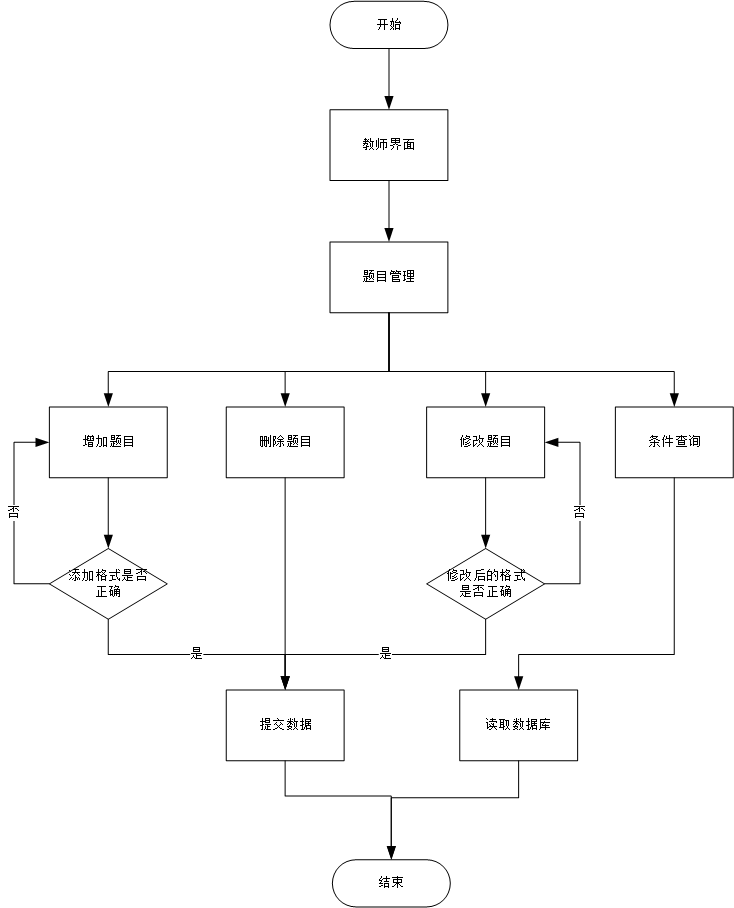


图5-10 教师-题目管理（增删查改）

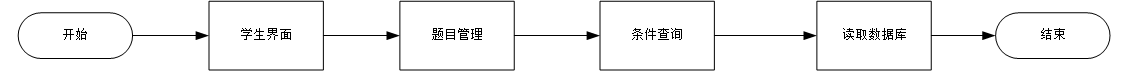


图5-11 学生-题目管理（查找）

### 5.3.4 选题管理

选题管理是由学生选择题目，老师和教学秘书修改题目状态两部分组成，学生选择由教师发布的题目后，须由教师和教学秘书确认。

模块的处理流程如下图：

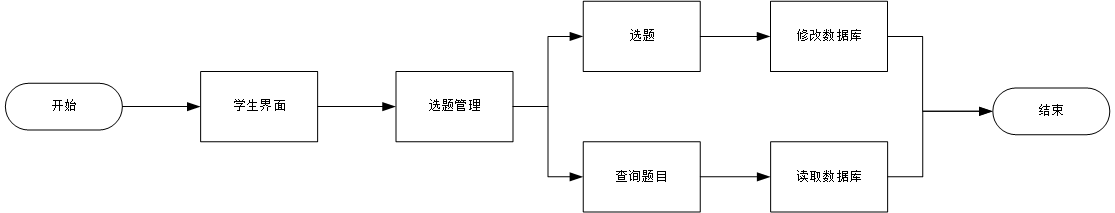


图5-16 学生-选题（选取题目）

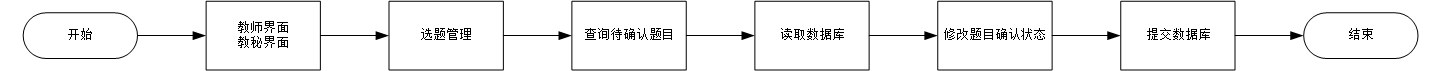


图5-17教师、管理员-确认选题（修改对应题目状态信息）

### 5.3.5 开题管理

开题管理需要教师上传开题相关文档模板，学生下载该模板并上传已完成的开题相关文档。

模块的处理流程如下图：

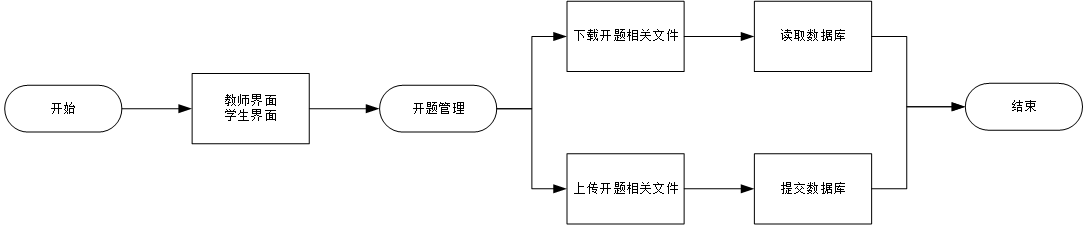


图5-20 教师、学生-上传下载开题相关文档

## 5.4系统数据库设计

本毕业设计系统采用了体积小、速度快、总体拥有成本低的MySQL数据库。MySQL数据库引擎为关系型数据和结构化数据提供了安全可靠的存储功能，可以使用常用的SQL语句进行读写。数据库使用简单，易学易用，研发成本大幅减少。以下是概念设计、逻辑设计和物理设计：

### 5.4.1 概念设计

通过调查分析，了解到系统中的实体类型有用户、角色、权限、题目、文件、答辩、反馈等，这些实体之间的相互关系有：

1. 用户与角色存在“拥有”关系，是多对多的；

2. 角色与权限存在“拥有”关系，是多对多的；

3. 用户与题目存在“拥有”关系，是多对多的；

4. 用户与文件存在“提供”关系，是一对多的；

5. 题目与文件存在“拥有”关系，是一对多的；

6. 用户与答辩存在“参与”关系，是一对一的；

7. 用户与反馈存在“提出”关系，是一对多的；

每个实体的属性分别是：

1. 用户：用户编号、用户密码、上次登录时间、是否是管理员、用户名、真实姓名、Email、是否是职工、是否激活、注册日期、学校、院系；

2. 角色：角色编号、角色名称；

3. 权限：权限编号、权限名称；

4. 题目：题目编号、题目名称、题目详情、限选人数、已选人数、出题时间、确认状态、答辩状态、上次编辑时间；

5. 文件：文件编号、文件存储地址、原始文件名称；

6. 答辩：答辩编号、答辩内容；

7. 反馈：反馈编号、反馈内容；

ER图如图5-21所示。

### 5.4.2 逻辑设计

逻辑设计的任务是根据DBMS的特征把概念结构转换为相应的逻辑结构。将图5-21转换为规范的关系模式为：

用户：(用户编号，用户名，用户密码，上次登录时间，是否为管理员，真实姓名，电子邮件，是否为职工，是否激活，注册时间，上次编辑时间)

题目：(题目编号，题目名称，题目详情，已选人数，限选人数，出题时间，开题状态，答辩状态)；

角色：(角色编号，角色名称)；

权限：(权限编号，权限名称)；

反馈：(反馈编号，反馈名称)；

文件：(文件编号，文件存储地址，原始文件名称，阶段)；

答辩：(答辩编号，答辩名称)；

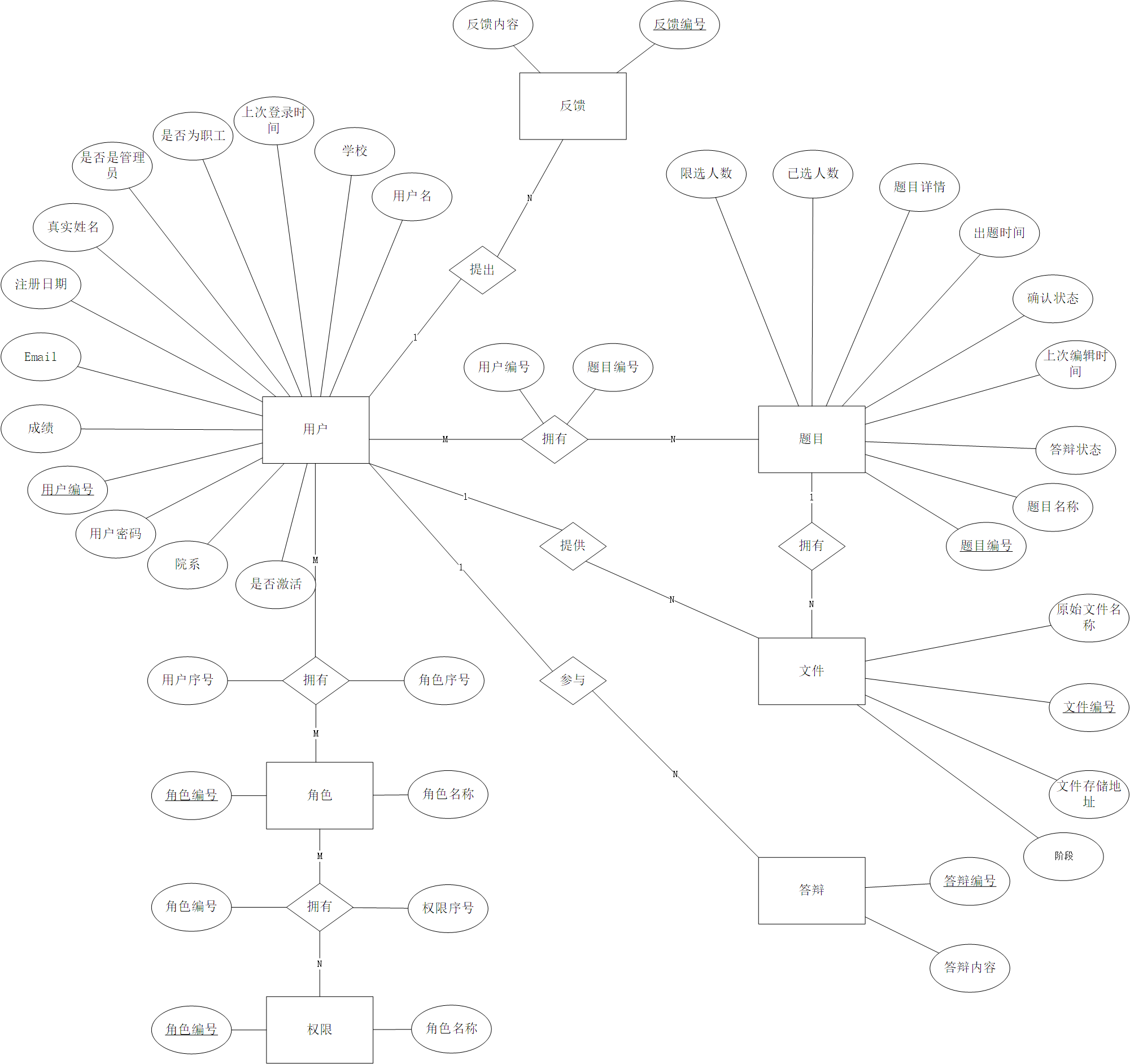


图5-21 数据库E-R图

### 5.4.3 物理设计

物理设计的目的是根据具体DBMS的特征，确定数据库的物理结构，确定数据库的物理结构（存储结构）。数据库中表的结构如表3-1到表

表5.1用户信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 用户编号 | int | 11 |
| 2 | username | 用户名 | varchar | 150 |
| 3 | password | 用户密码 | varchar | 128 |
| 4 | last\_login | 上次登录时间 | datetime | 255 |
| 5 | is\_superuser | 是否为管理员 | bool | 1 |
| 6 | full\_name | 真实姓名 | varchar | 30 |
| 7 | Email | 电子邮件 | varchar | 255 |
| 8 | is\_staff | 是否为职工 | bool | 1 |
| 9 | is\_active | 是否激活 | bool | 1 |
| 10 | date\_joined | 注册时间 | datetime | 256 |
| 11 | school | 学校 | varchar | 100 |
| 12 | department | 院系 | varchar | 50 |
| 13 | score | 成绩 | varchar | 10 |

表5.2题目信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 题目编号 | int | 11 |
| 2 | title | 题目名称 | varchar | 100 |
| 3 | detail | 题目详情 | text |  |
| 4 | chosen\_num | 已选人数 | smallint | 2 |
| 5 | limit\_num | 限选人数 | smallint | 2 |
| 6 | release\_time | 出题时间 | datetime | 256 |
| 7 | accept\_status | 开题状态 | smallint | 2 |
| 8 | defence\_status | 答辩状态 | smallint | 2 |
| 9 | last\_edit\_time | 上次编辑时间 | datetime | 256 |

表5.3角色信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 角色编号 | int | 11 |
| 2 | role\_name | 角色名称 | varchar | 150 |

表5.4权限信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 权限编号 | int | 11 |
| 2 | role\_permission | 权限名称 | varchar | 255 |

表5.5文件信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 文件编号 | int | 11 |
| 2 | file | 文件存储位置 | varchar | 255 |
| 3 | raw\_name | 原始文件名 | varchar | 255 |

表5.6答辩信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 答辩编号 | int | 11 |
| 2 | defense\_content | 答辩内容 | text |  |

表5.7反馈信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 反馈编号 | int | 11 |
| 2 | feedback\_content | 反馈内容 | text |  |

表5.8用户角色关系信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 关系编号 | int | 11 |
| 2 | user\_id | 用户编号 | int | 11 |
| 3 | role\_id | 角色编号 | int | 11 |

表5.9角色权限关系信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 关系编号 | int | 11 |
| 2 | role\_id | 角色编号 | int | 11 |
| 3 | rpermission\_id | 权限编号 | int | 11 |

表5.10用户答辩关系信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 关系编号 | int | 11 |
| 2 | user\_id | 用户编号 | int | 11 |
| 3 | defense\_id | 答辩编号 | int | 11 |

表5.11用户反馈关系信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 关系编号 | int | 11 |
| 2 | user\_id | 用户编号 | int | 11 |
| 3 | feedback\_id | 反馈编号 | int | 11 |

表5.12题目文件关系信息

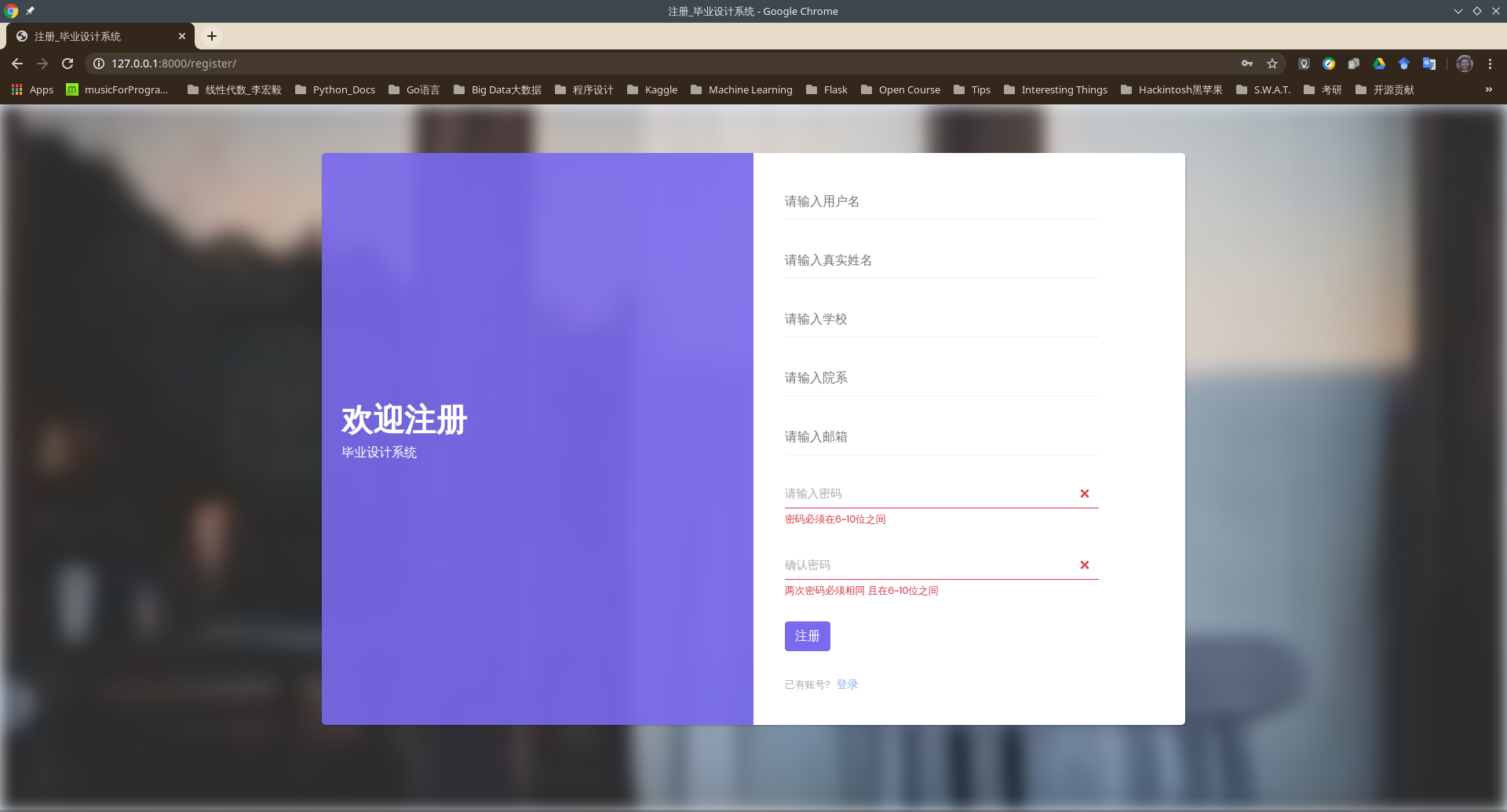
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 别名 | 类型 | 长度 |
| 1 | id | 关系编号 | int | 11 |
| 2 | topic\_id | 题目编号 | int | 11 |
| 3 | file\_id | 文件编号 | int | 11 |

# 第6章 系统实现

## 6.1 注册与登录实现

6.1.1 注册

（1）界面截图



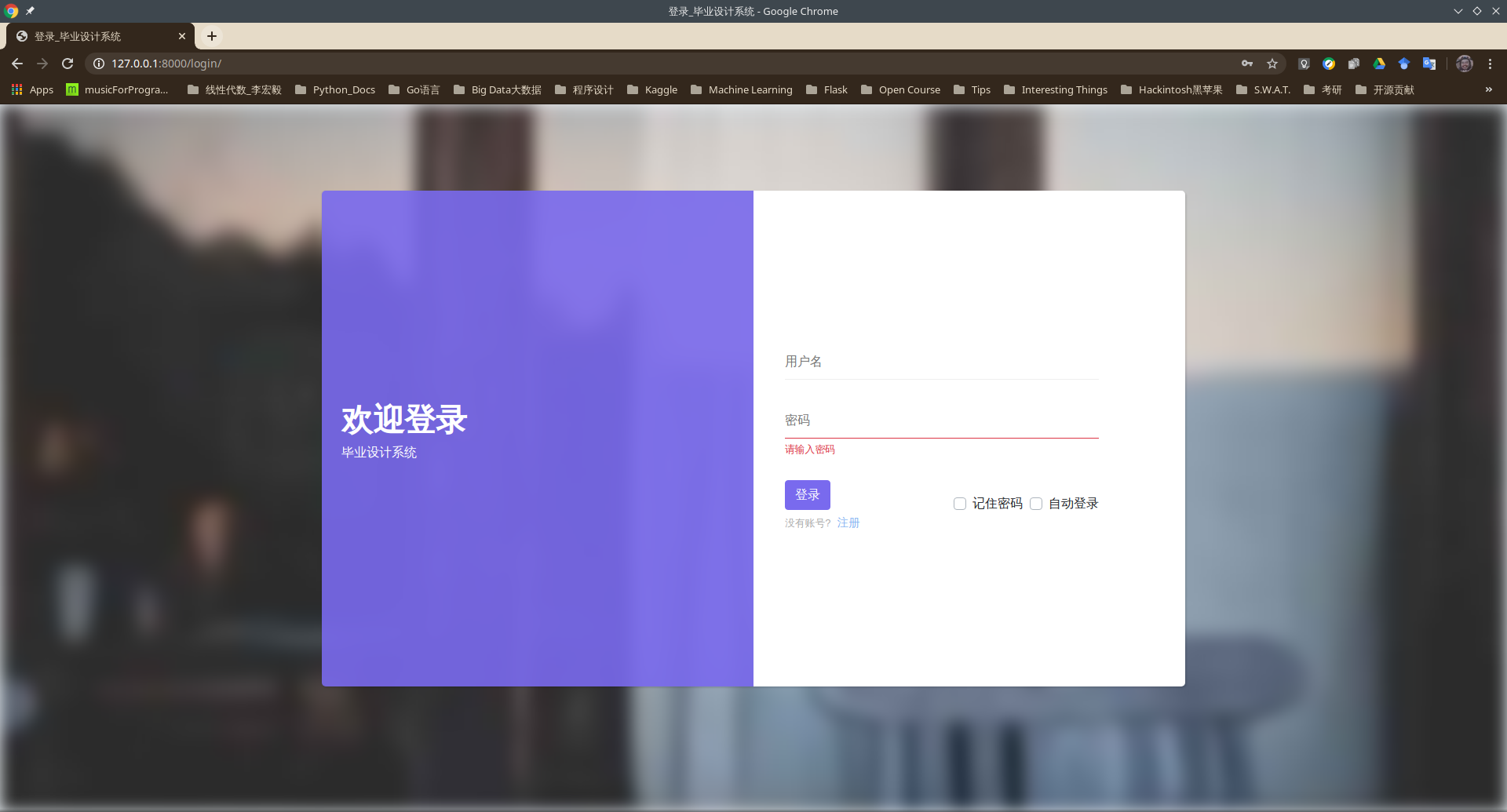
（2）核心代码

from django.shortcuts import render, redirect, reverse  
from django.http import HttpResponse  
from django.views import View  
from user.models import User  
from django.contrib.auth import authenticate, login, logout  
from rolepermissions.roles import assign\_role  
from utils.fdfs.storage import FDFSStorage  
from django.conf import settings  
import os

# /register/  
class RegisterView(View):  
 @staticmethod  
 def get(request):  
 # 显示注册页面  
 return render(request, 'register.html')  
  
 @staticmethod  
 def post(request):  
 name = request.POST.get('registerName', None)  
 username = request.POST.get('registerUsername', None)  
 password = request.POST.get('registerPassword', None)  
 school = request.POST.get('registerSchool', None)  
 department = request.POST.get('registerDepartment', None)  
 email = request.POST.get('registerEmail', None)  
 # print(name)  
 # print(username)  
 # print(password)  
 # print(school)  
 # print(department)  
 # print(email)  
  
 if not all([username, password]):  
 print("数据不完整")  
  
 user = User.objects.create\_user(  
 name=name,  
 username=username,  
 password=password,  
 school=school,  
 department=department,  
 email=email  
 )  
 user.save()  
 # 默认角色为学生  
 assign\_role(user, 'student')  
 return redirect(reverse('login'))

6.1.2 登录

（1） 界面截图



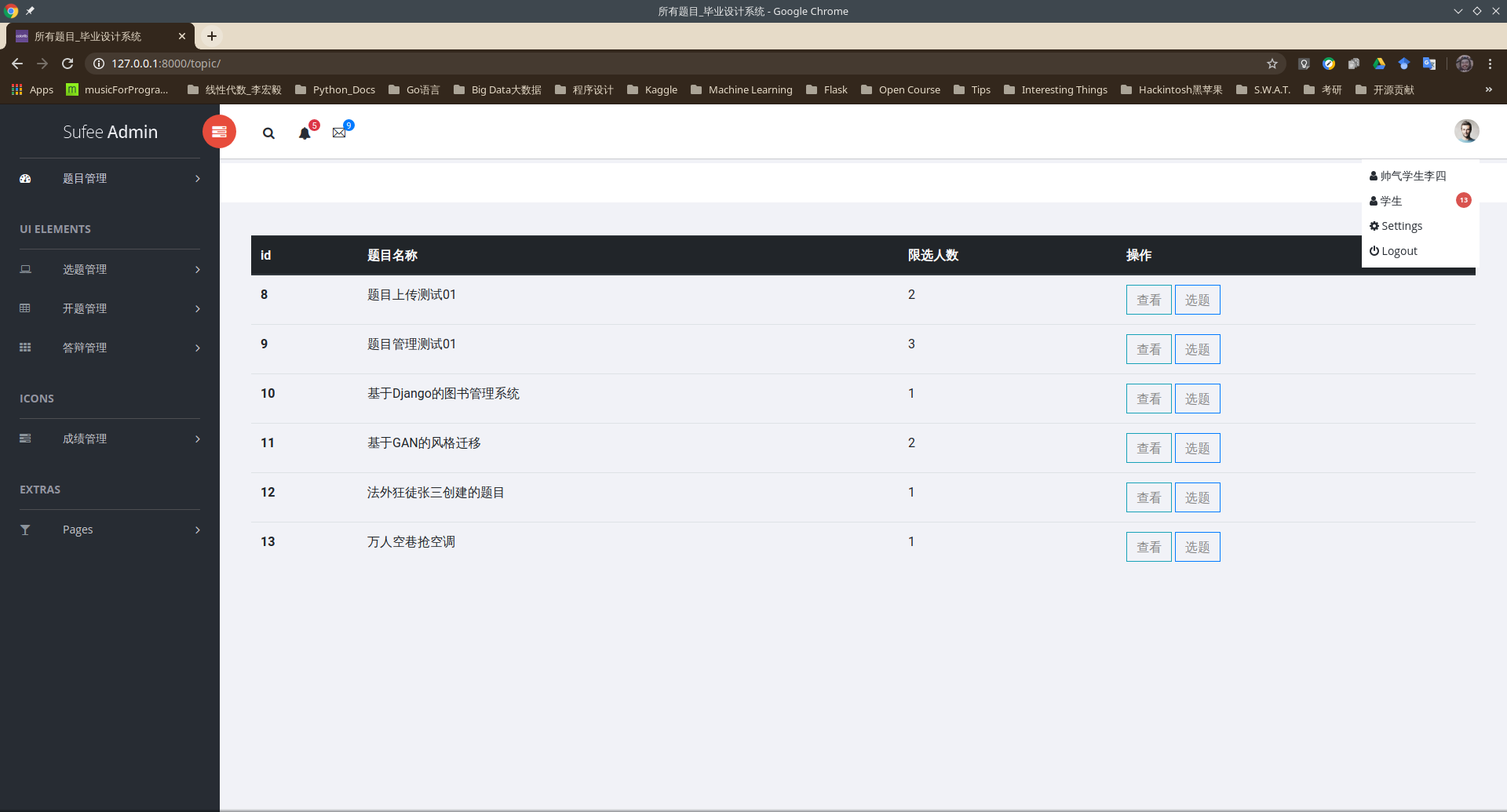
（2）核心代码

# /login/  
class LoginView(View):  
 @staticmethod  
 def get(request):  
 return render(request, 'login.html')  
  
 @staticmethod  
 def post(request):  
 username = request.POST.get('userName', None)  
 password = request.POST.get('passWord', None)  
 print(username)  
 print(password)  
  
 user = authenticate(username=username, password=password)  
 print(user)  
  
 if user is not None:  
 login(request, user)  
 next\_url = request.GET.get('next', reverse('topic:show\_all\_topic'))  
 response = redirect(next\_url)  
 return response  
 else:  
 return render(request, 'login.html')  
  
  
class LogoutView(View):  
 '''退出登录'''  
  
 def get(self, request):  
 '''退出登录'''  
 # 清除用户session信息  
 logout(request)  
 # 跳转到首页  
 return redirect(reverse('login'))

## 6.1毕业生角色实现

### 6.1.1选题实现

（1）界面截图

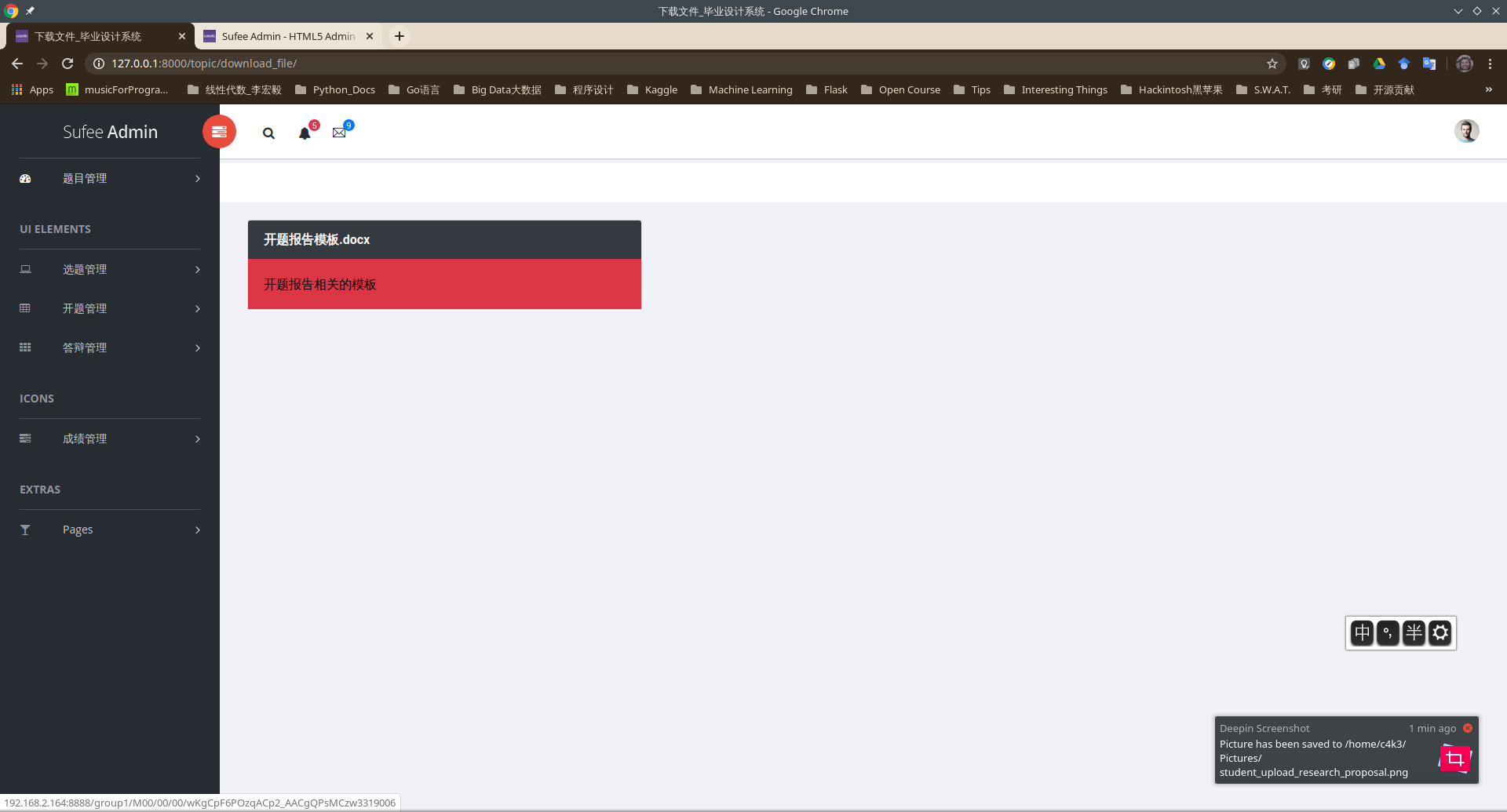
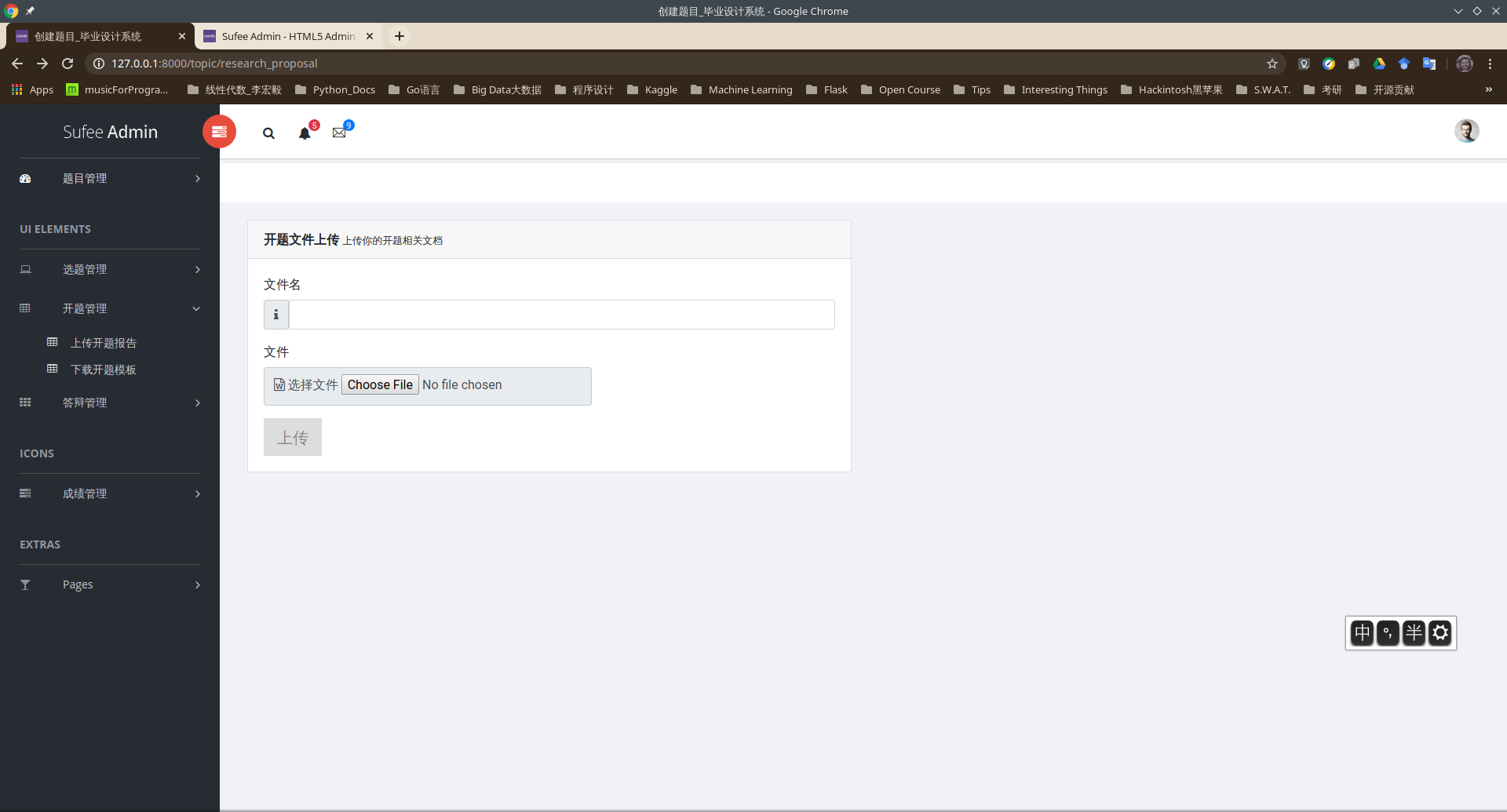


（2）核心代码

# /topic/choose\_(\d+)  
# 选择题目  
def choose\_topic(request, topic\_id):  
 user = request.user  
 topic = TopicRecord.objects.get(id=topic\_id)  
 Topic2User.objects.create(topic\_id=topic, user\_id=user)  
  
 return redirect(reverse('topic:show\_chosen\_topic'))  
  
  
# /topic/chosen\_topic  
# 展示已选题目  
def show\_chosen\_topic(request):  
 user = request.user  
 t2u\_list = Topic2User.objects.filter(user\_id=user)  
 topic\_list = list()  
 user\_list = list()  
 teacher\_list = list()  
 for item in t2u\_list:  
 topic = item.topic\_id  
 user = item.user\_id  
 user\_list.append(user)  
 topic\_list.append(topic)  
  
 for user in user\_list:  
 if has\_role(user, ['teacher']):  
 teacher\_list.append(user)  
 try:  
 role = get\_user\_roles(user)[0].get\_cls\_name()  
 except IndexError:  
 role = ''  
  
 try:  
 teacher = teacher\_list[0]  
 except IndexError:  
 teacher = ''  
 context = {  
 'result': topic\_list,  
 'name': user.name,  
 'role': role,  
 'teacher': teacher  
 }  
 return render(request, 'chosen\_topic.html', context)

### 6.1.2 开题实现

（1）界面截图



1. 核心代码

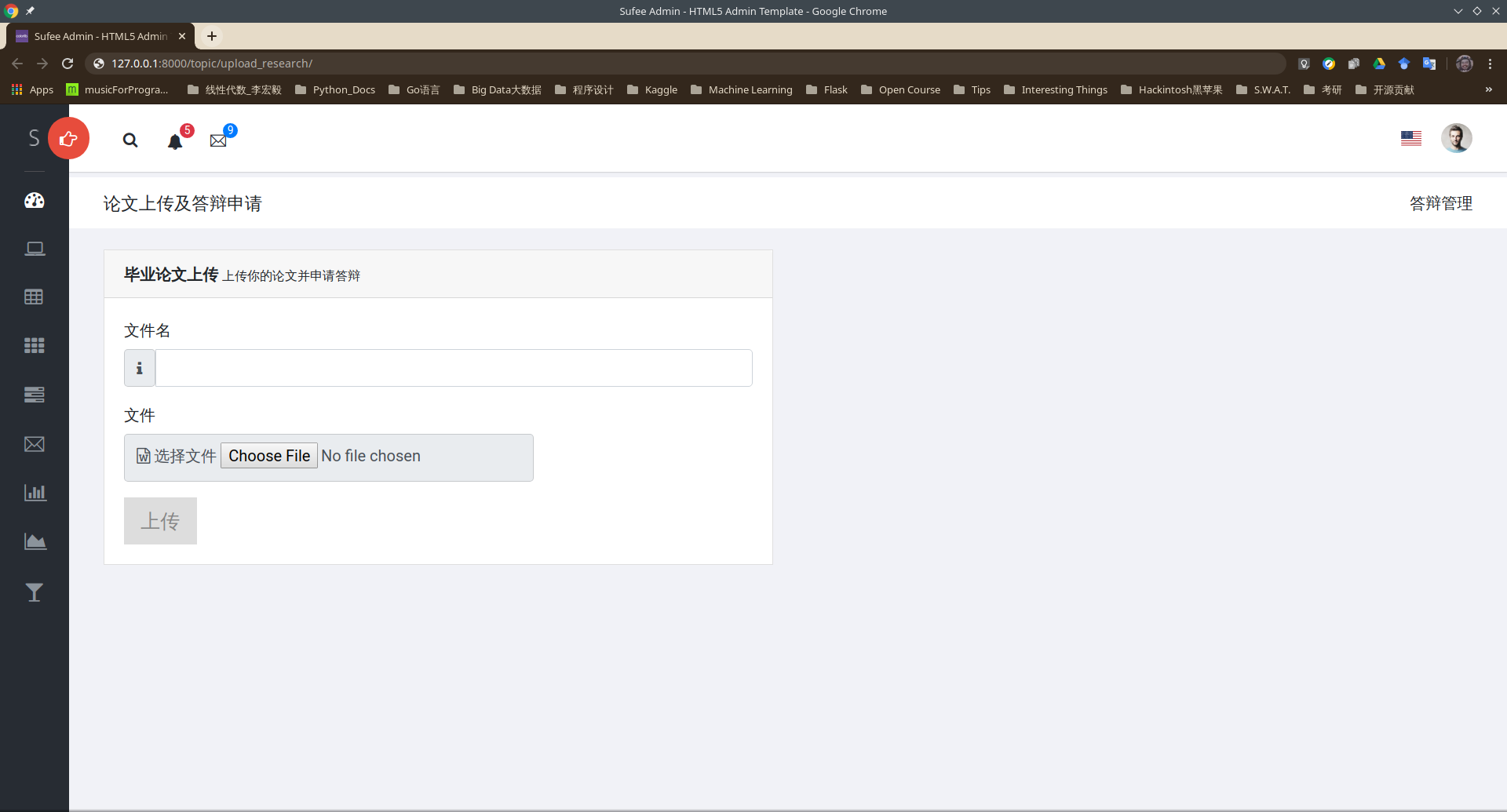
Topic.upload\_research\_proposal.py

def upload\_research\_proposal(request):  
 user = request.user  
 try:  
 # 获取用户角色  
 role = get\_user\_roles(user)[0].get\_cls\_name()  
 except IndexError:  
 role = ''  
 if request.method == 'POST':  
 form = AnnexForm(request.POST)  
 file = request.FILES.get("file")  
 temp\_file = "%s/%s" % (settings.MEDIA\_ROOT, file.name)  
 with open(temp\_file, 'wb') as upload\_files:  
 for chunk in file.chunks():  
 upload\_files.write(chunk)  
  
 # 将文件写入临时文件  
  
 file\_path = os.path.abspath(temp\_file)  
 print(file\_path)  
 fdfs\_storage = FDFSStorage()  
 remote\_file\_id = fdfs\_storage.upload(file\_path)[1]  
 if form.is\_valid():  
 form.save()  
 user = request.user  
 user\_id = user  
  
 if has\_role(user, ['student']):  
 t2u = Topic2User.objects.get(user\_id=user\_id)  
 topic = t2u.topic\_id  
 topic\_id = TopicRecord.objects.get(id=topic.id)  
  
 annex\_phase = Phase.objects.get(phase="开题文档")  
 Annex.objects.create(file=remote\_file\_id, annex\_phase=annex\_phase)  
 annex\_id = Annex.objects.get(file=remote\_file\_id)  
 Annex2Topic2User.objects.create(annex\_id=annex\_id, topic\_id=topic\_id, user\_id=user\_id)  
  
 return redirect(reverse('topic:show\_my\_topic'))

context = {  
 'name': user.name,  
 'role': role  
}  
return render(request, "research\_proposal.html", context)

### 6.1.4 答辩实现

（1）界面截图

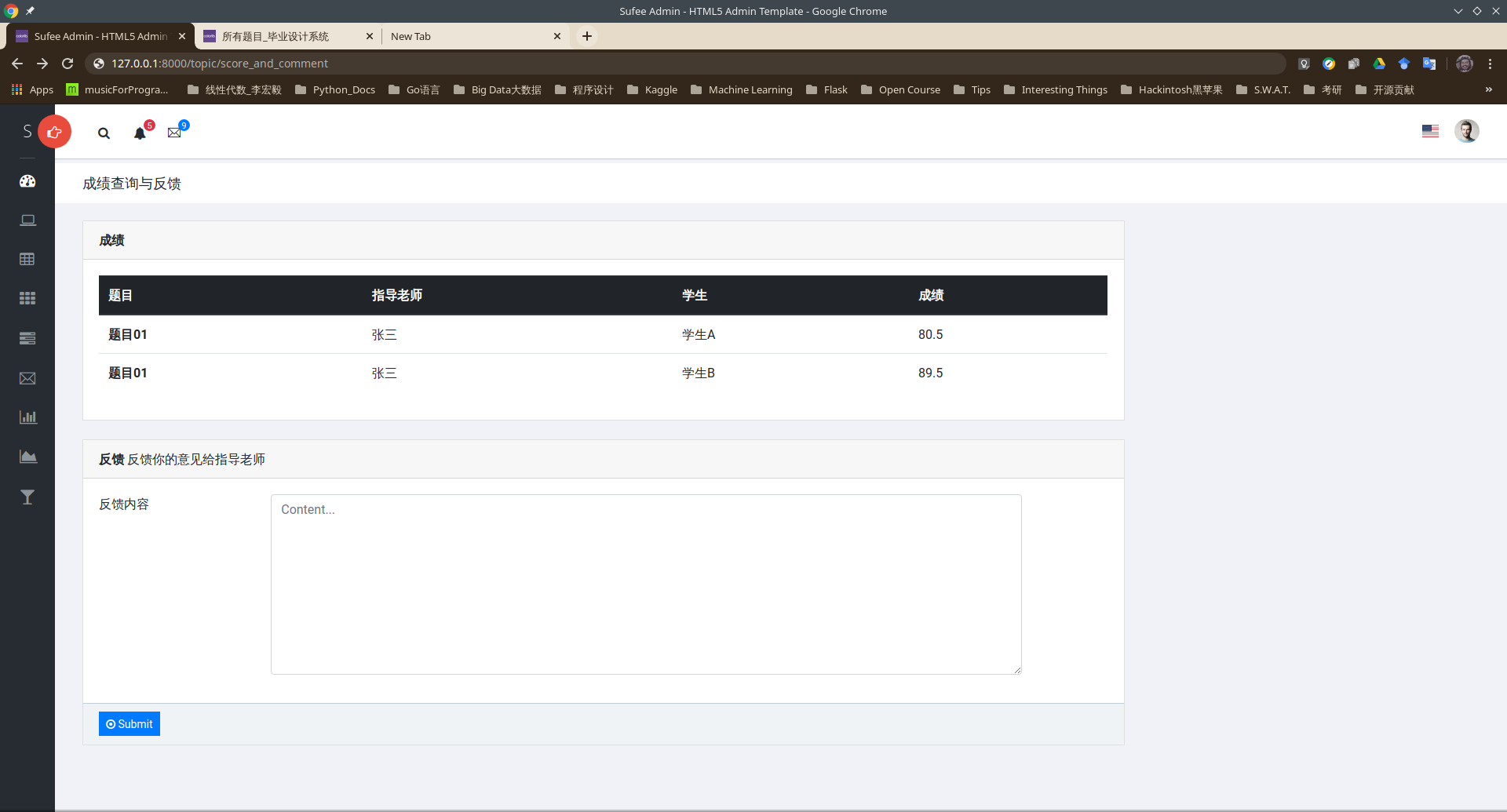


（2）核心代码

def upload\_research\_proposal(request):  
 user = request.user  
 try:  
 # 获取用户角色  
 role = get\_user\_roles(user)[0].get\_cls\_name()  
 except IndexError:  
 role = ''  
 if request.method == 'POST':  
 form = AnnexForm(request.POST)  
 file = request.FILES.get("file")  
 temp\_file = "%s/%s" % (settings.MEDIA\_ROOT, file.name)  
 with open(temp\_file, 'wb') as upload\_files:  
 for chunk in file.chunks():  
 upload\_files.write(chunk)  
  
 # 将文件写入临时文件  
  
 file\_path = os.path.abspath(temp\_file)  
 print(file\_path)  
 fdfs\_storage = FDFSStorage()  
 remote\_file\_id = fdfs\_storage.upload(file\_path)[1]  
 if form.is\_valid():  
 form.save()  
 user = request.user  
 user\_id = user  
  
 if has\_role(user, ['student']):  
 t2u = Topic2User.objects.get(user\_id=user\_id)  
 topic = t2u.topic\_id  
 topic\_id = TopicRecord.objects.get(id=topic.id)  
  
 annex\_phase = Phase.objects.get(phase="开题文档")  
 Annex.objects.create(file=remote\_file\_id, annex\_phase=annex\_phase)  
 annex\_id = Annex.objects.get(file=remote\_file\_id)  
 Annex2Topic2User.objects.create(annex\_id=annex\_id, topic\_id=topic\_id, user\_id=user\_id)  
  
 #  
 # if has\_role(user, ['teacher']):  
 #  
  
 return redirect(reverse('topic:show\_my\_topic'))  
  
 context = {  
 'name': user.name,  
 'role': role  
 }  
 return render(request, "research\_proposal.html", context)

6.1.5 成绩与反馈实现

（1）界面截图



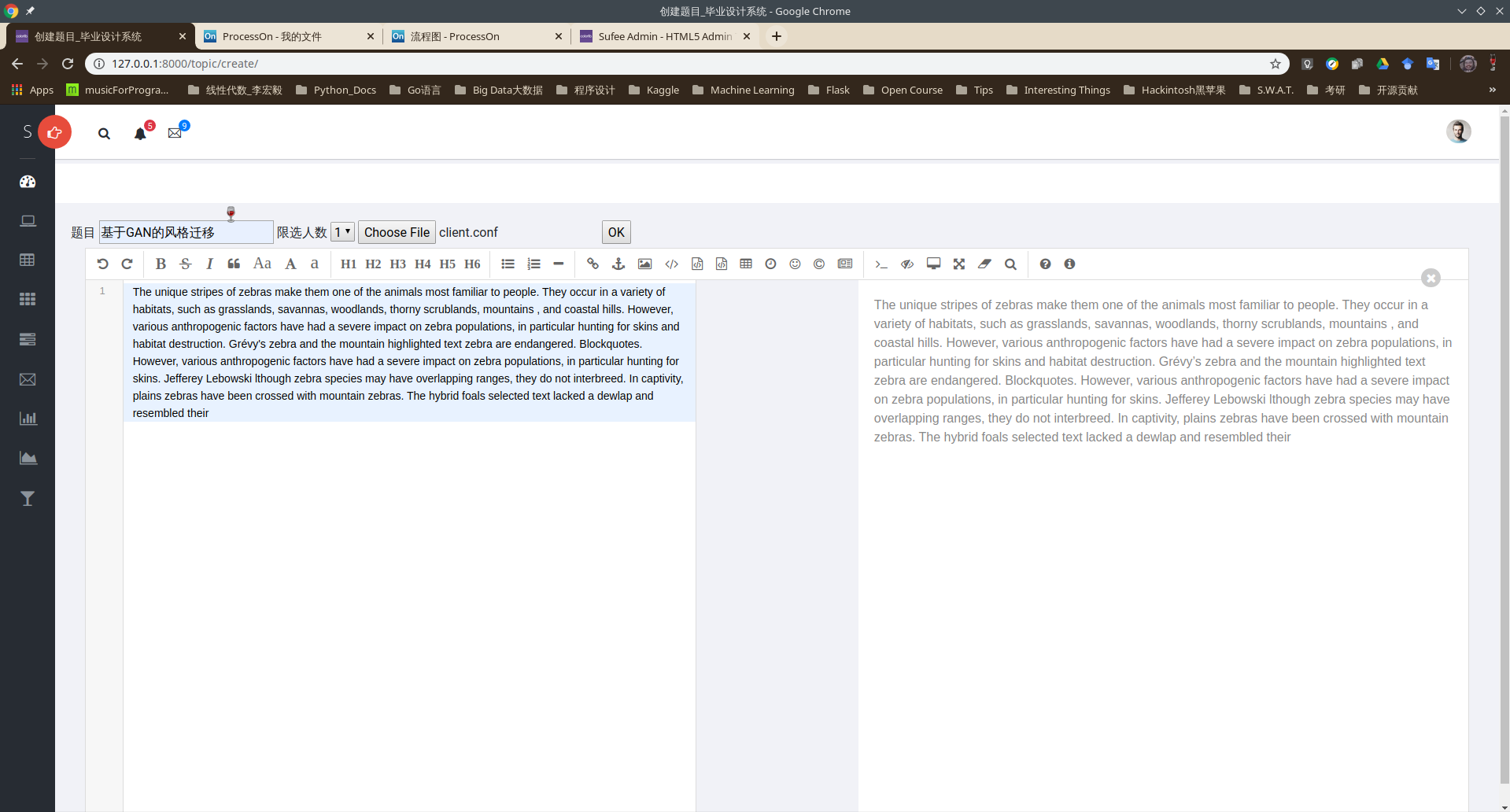
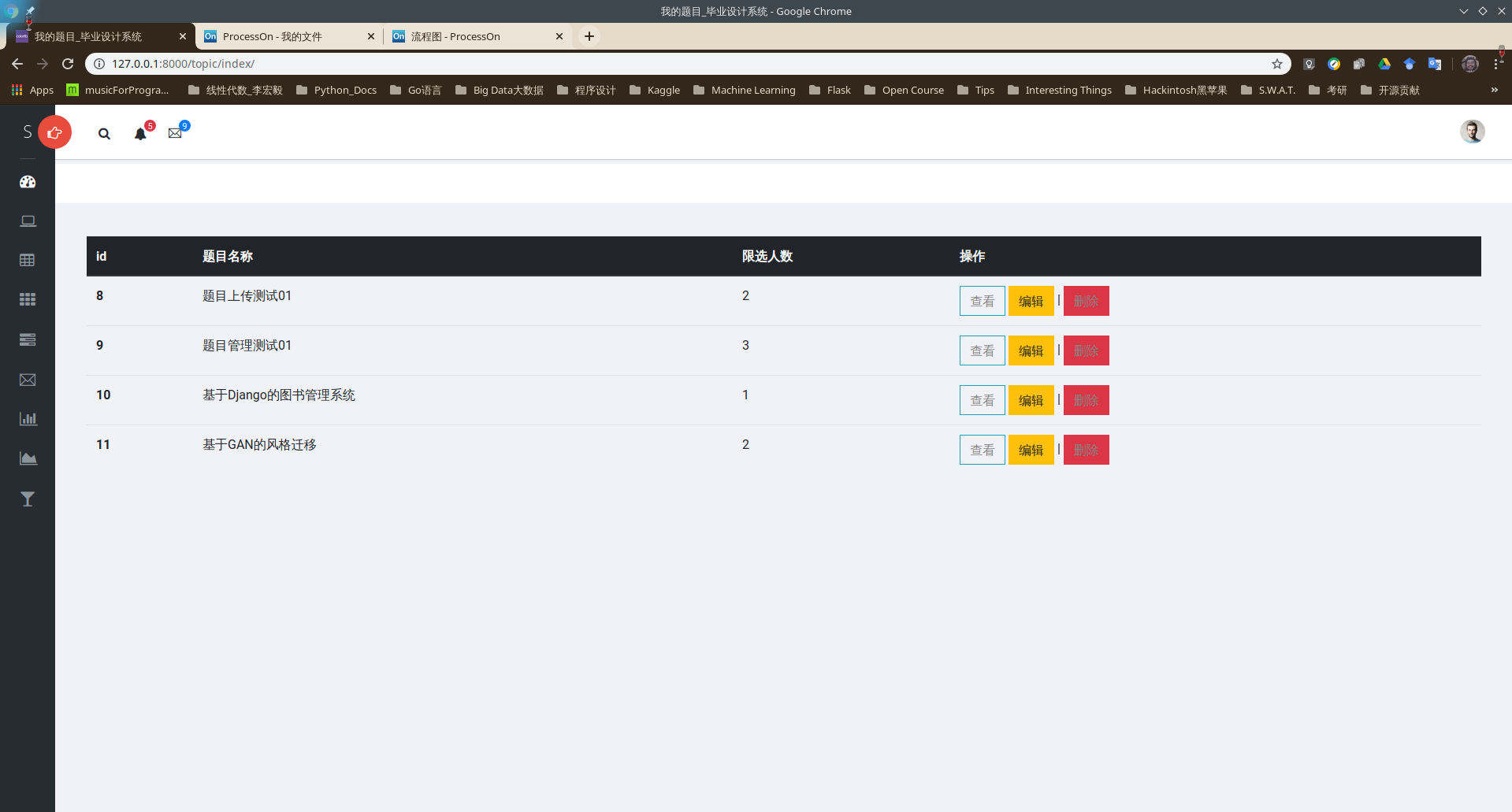
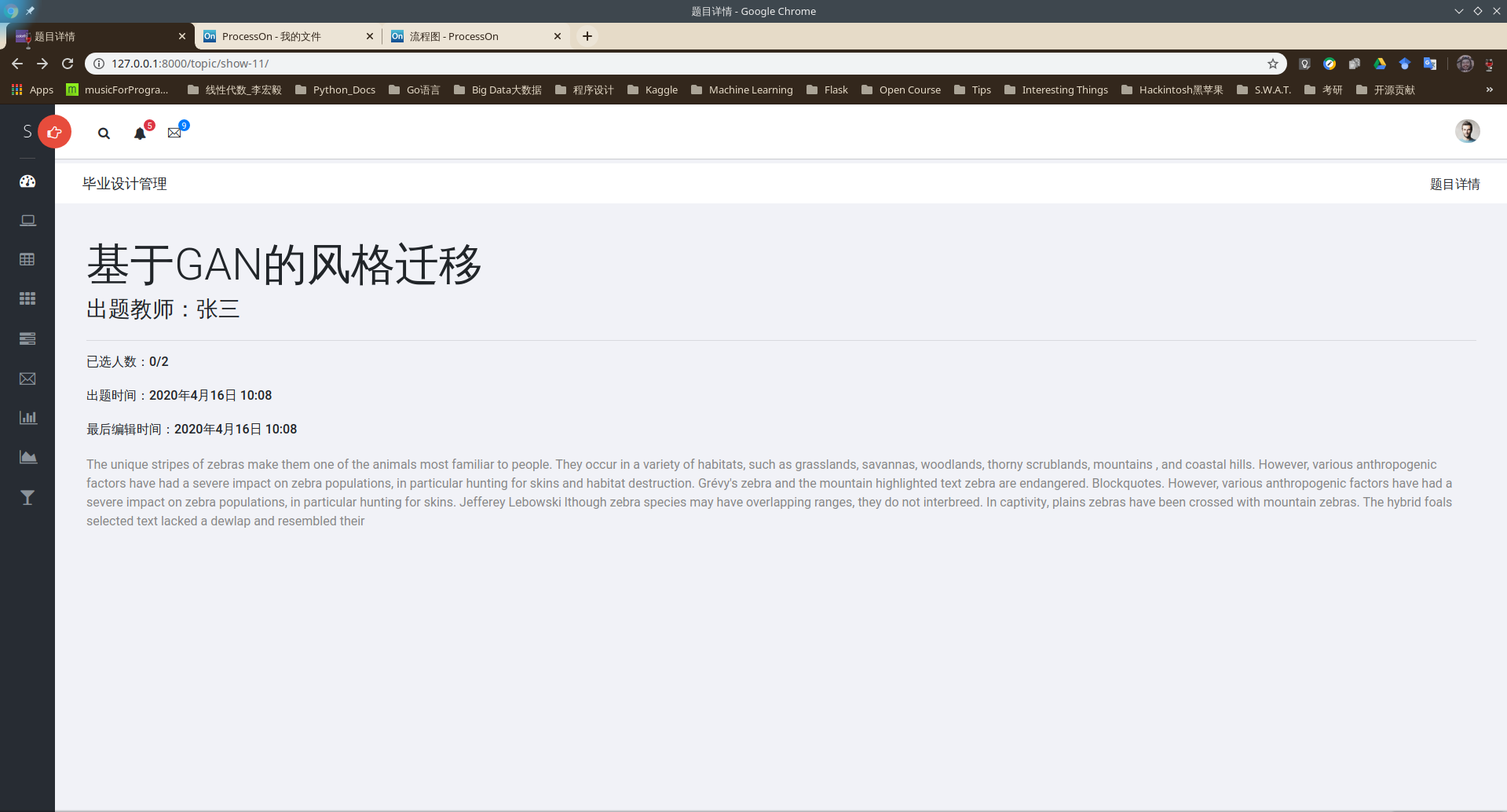
（2）核心代码

def get\_score\_and\_set\_comment(request):  
 global teacher  
 if request.method == 'GET':  
 user = request.user  
 # 获得题目  
 user\_id = user.id  
 t2u = Topic2User.objects.get(user\_id=user\_id)  
 topic = t2u.topic\_id  
 title = topic.title  
  
 user\_list = Topic2User.objects.filter(topic\_id=topic)  
 for user in user\_list:  
 if has\_role(user, 'teacher'):  
 teacher = user  
 break  
 score = user.socre  
  
 context = {  
 "user": user,  
 "topic": title,  
 "teacher": teacher,  
 "score": score  
 }  
  
 if request.method == 'POST':  
 user = request.user  
   
   
 feedback = request.POST.get('comment', None)  
   
 feedback\_obj = Feedback.objects.create(feedback=feedback, setter = user, getter = teacher)  
 feedback\_obj.save()

## 6.2 毕业指导老师功能实现

6.2.1 题目管理

（1）界面截图

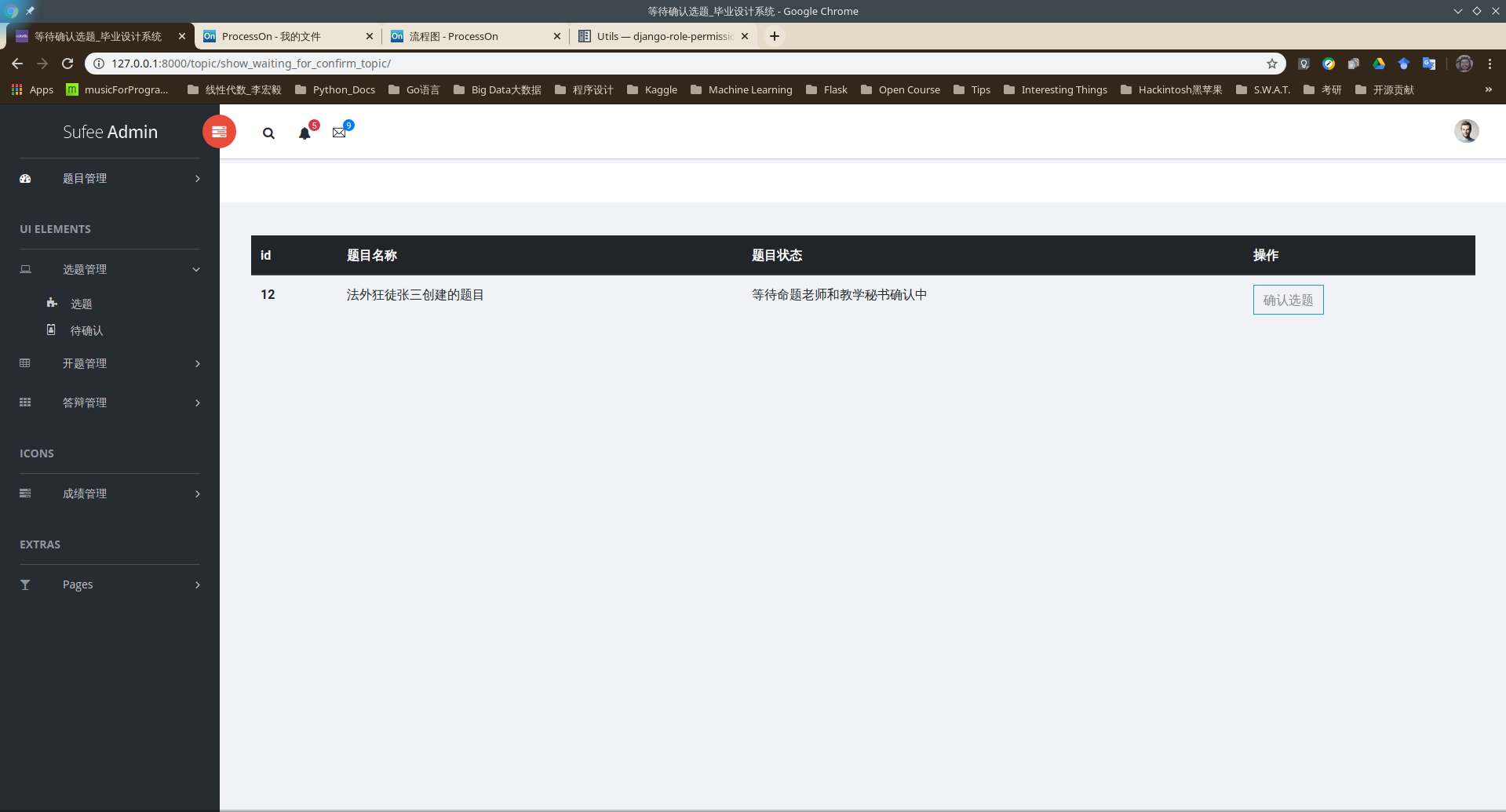


1. 核心代码

# /topic/create/  
# 创建题目（role==teacher）  
@login\_required  
def test\_create\_topic(request):  
 # 获取当前用户对象  
 current\_user = request.user  
 try:  
 # 获取用户对应角色  
 role = get\_user\_roles(current\_user)[0].get\_cls\_name()  
 except IndexError:  
 role = ''  
 # 获得form，并写入数据库  
 if request.method == 'POST':  
 form = TopicRecordForm(request.POST)  
 # file = request.FILES.get("file")  
 # temp\_file = "%s/%s" % (settings.MEDIA\_ROOT, file.name)  
 # with open(temp\_file, 'wb') as upload\_files:  
 # for chunk in file.chunks():  
 # upload\_files.write(chunk)  
 #  
 # # 将文件写入临时文件  
 #  
 # file\_path = os.path.abspath(temp\_file)  
 # print(file\_path)  
 # fdfs\_storage = FDFSStorage()  
 # remote\_file\_id = fdfs\_storage.upload(file\_path)[1]  
  
 print("-------TOF")  
 print(form.is\_valid())  
 if form.is\_valid():  
 form.save()  
 title = form.clean().get("title")  
 print("-----title------")  
 print(title)  
 topic\_id = TopicRecord.objects.get(title=title)  
 user = request.user  
 user\_id = user  
  
 Topic2User.objects.create(topic\_id=topic\_id, user\_id=user\_id)  
 # Annex.objects.create(file=remote\_file\_id)  
 # annex\_id = Annex.objects.get(file=remote\_file\_id)  
 # Annex2Topic2User.objects.create(annex\_id=annex\_id, topic\_id=topic\_id, user\_id=user\_id)  
  
 return redirect(reverse('topic:show\_my\_topic'))  
  
 context = {  
 'name': current\_user.name,  
 'role': role  
 }  
 return render(request, "test\_create\_topic.html", context)  
  
  
# /topic/show-(\d+)  
# 查看题目详情  
@login\_required  
def topic\_detail(request, topic\_id):  
 if request.method == 'GET':  
 # 获取当前topic\_id对应的题目对象  
 obj = TopicRecord.objects.get(id=topic\_id)  
 # if not obj:  
 # return HttpResponse('已被学生选择或教秘确认完成的题目无法被编辑')  
  
 teacher\_id = Topic2User.objects.filter(topic\_id=topic\_id)  
 print(teacher\_id)  
 title = obj.title  
 # 通过Topic2User表获取teacher\_id  
 teacher = Topic2User.objects.get(topic\_id=topic\_id)  
 teacher = teacher.user\_id  
 # 通过teacher\_id获取教师姓名  
 teacher = User.objects.get(username=teacher)  
 teacher = teacher.name  
 chosen\_num = obj.chosen\_num  
 limit\_num = obj.limit\_num  
 release\_time = obj.release\_time  
 last\_edit\_time = obj.last\_edit\_time  
 detail = obj.detail  
 context = {  
 "title": title,  
 "teacher": teacher,  
 "chosen\_num": chosen\_num,  
 "limit\_num": limit\_num,  
 "release\_time": release\_time,  
 "last\_edit\_time": last\_edit\_time,  
 "detail": detail  
 }  
 return render(request, "topic\_detail.html", context)  
  
  
# /topic/delete-(\d+)  
# 查看题目详情  
@login\_required  
def delete\_topic(request, topic\_id):  
 TopicRecord.objects.get(id=topic\_id).delete()  
 return redirect(reverse("topic:show\_my\_topic"))

6.2.2 选题管理

（1）界面截图

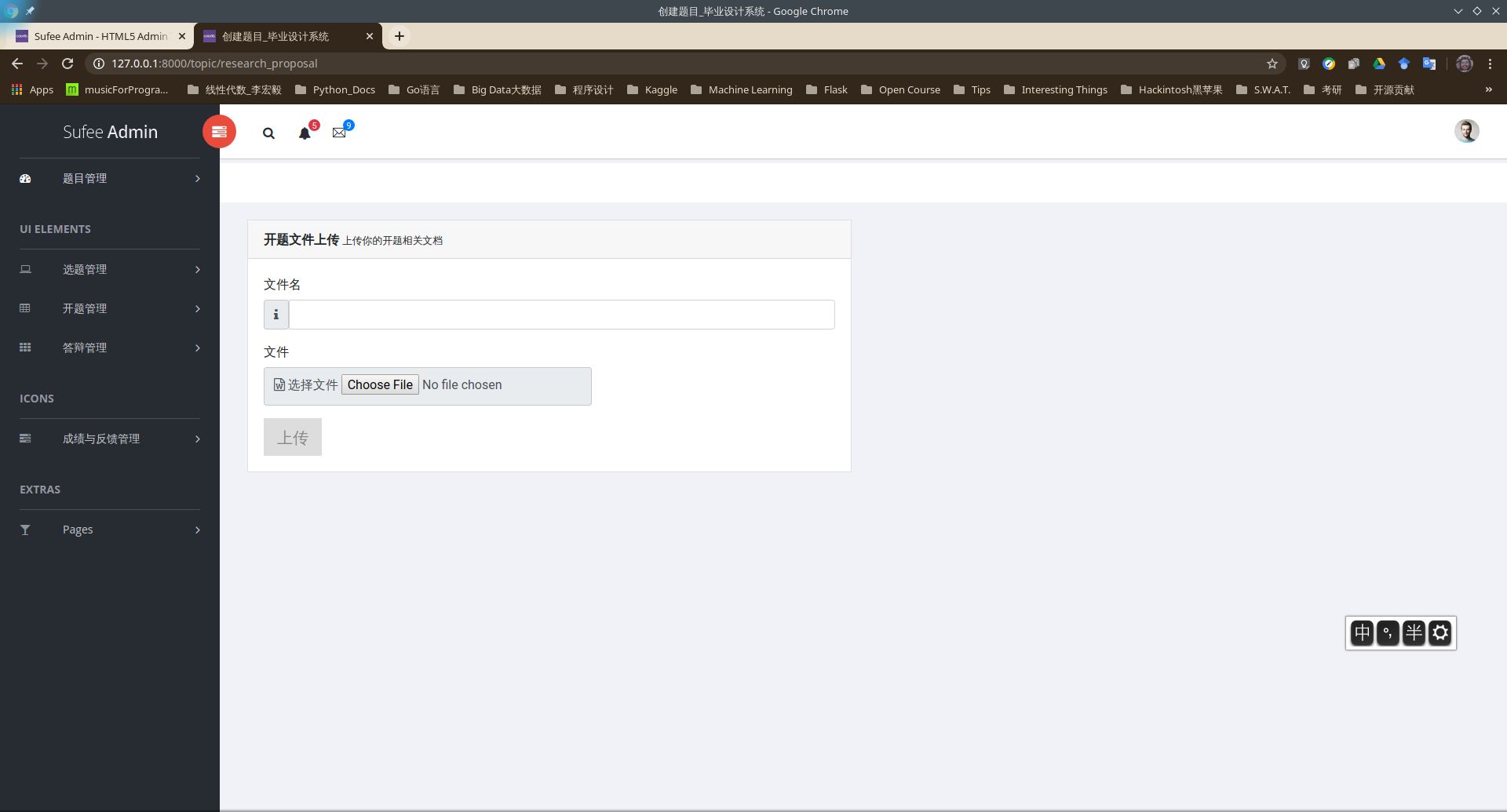


（2）核心代码

def waiting\_for\_confirm(request, topic\_id):  
 topic = TopicRecord.objects.get(id=topic\_id)  
 topic.accept = 1  
 topic.save()  
 return redirect(reverse('topic:show\_waiting\_for\_confirm\_topic'))  
  
  
def show\_waiting\_for\_confirm\_topic(request):  
 user = request.user  
 t2u\_list = Topic2User.objects.filter(user\_id=user)  
 topic\_list = list()  
 for item in t2u\_list:  
 topic = item.topic\_id  
 if topic.accept == 0:  
 topic\_list.append(topic)  
 print(topic\_list)  
 try:  
 role = get\_user\_roles(user)[0].get\_cls\_name()  
 except IndexError:  
 role = ''  
  
 context = {  
 'result': topic\_list,  
 'name': user.name,  
 'role': role,  
 }  
 return render(request, 'waiting\_for\_confirm.html', context)

6.2.3 开题管理

（1）界面截图

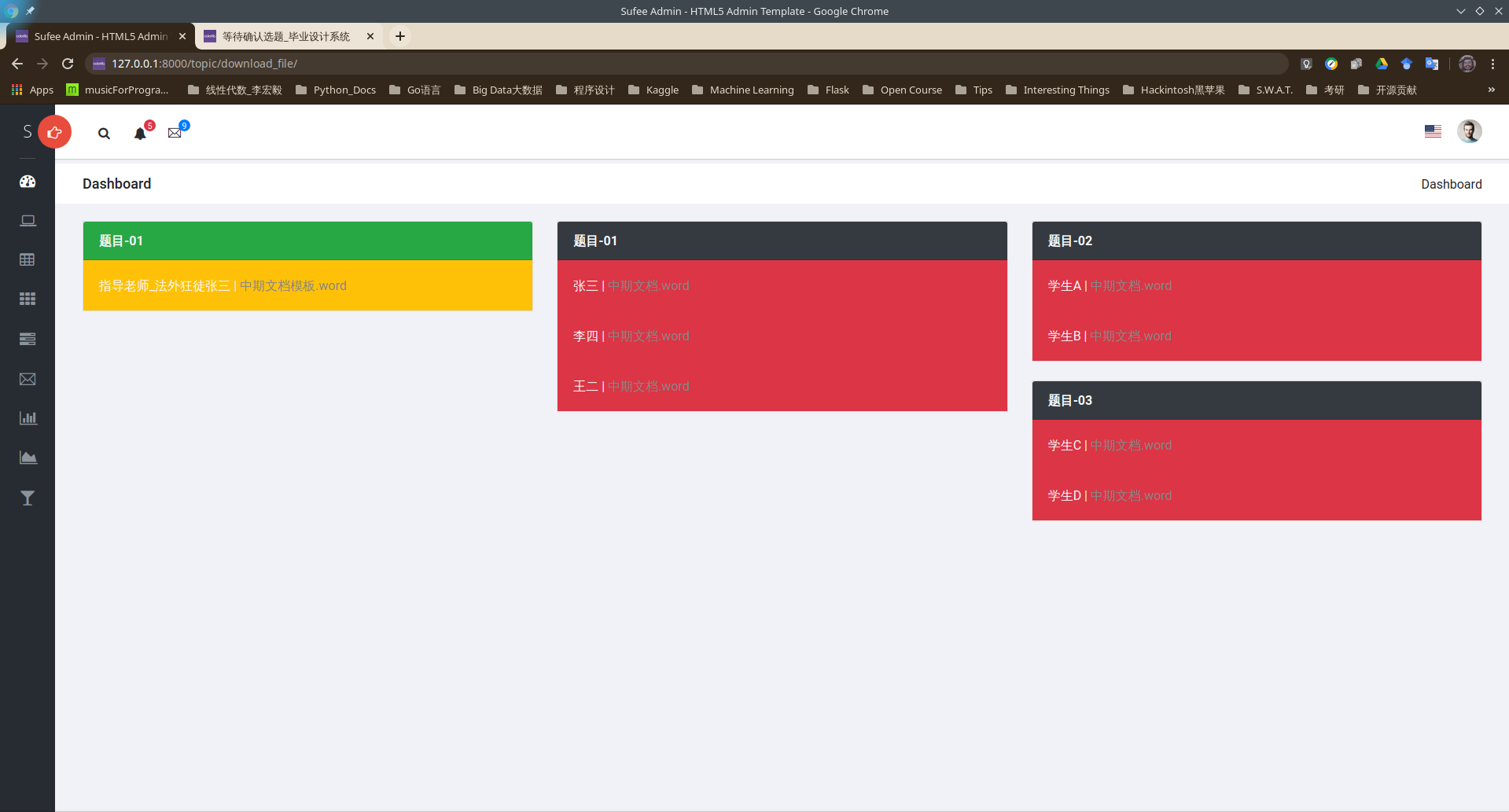


（2）核心代码

def upload\_research\_proposal(request):  
 user = request.user  
 try:  
 # 获取用户角色  
 role = get\_user\_roles(user)[0].get\_cls\_name()  
 except IndexError:  
 role = ''  
 if request.method == 'POST':  
 form = AnnexForm(request.POST)  
 file = request.FILES.get("file")  
 temp\_file = "%s/%s" % (settings.MEDIA\_ROOT, file.name)  
 with open(temp\_file, 'wb') as upload\_files:  
 for chunk in file.chunks():  
 upload\_files.write(chunk)def upload\_research\_proposal(request):  
 user = request.user  
 try:  
 # 获取用户角色  
 role = get\_user\_roles(user)[0].get\_cls\_name()  
 except IndexError:  
 role = ''  
 if request.method == 'POST':  
 form = AnnexForm(request.POST)  
 file = request.FILES.get("file")  
 temp\_file = "%s/%s" % (settings.MEDIA\_ROOT, file.name)  
 with open(temp\_file, 'wb') as upload\_files:  
 for chunk in file.chunks():  
 upload\_files.write(chunk)  
  
 # 将文件写入临时文件  
  
 file\_path = os.path.abspath(temp\_file)  
 print(file\_path)  
 fdfs\_storage = FDFSStorage()  
 remote\_file\_id = fdfs\_storage.upload(file\_path)[1]  
 if form.is\_valid():  
 form.save()  
 user = request.user  
 user\_id = user  
  
 if has\_role(user, ['student']):  
 t2u = Topic2User.objects.get(user\_id=user\_id)  
 topic = t2u.topic\_id  
 topic\_id = TopicRecord.objects.get(id=topic.id)  
  
 annex\_phase = Phase.objects.get(phase="开题文档")  
 Annex.objects.create(file=remote\_file\_id, annex\_phase=annex\_phase)  
 annex\_id = Annex.objects.get(file=remote\_file\_id)  
 Annex2Topic2User.objects.create(annex\_id=annex\_id, topic\_id=topic\_id, user\_id=user\_id)  
  
 #  
 # if has\_role(user, ['teacher']):  
 #  
  
 return redirect(reverse('topic:show\_my\_topic'))  
  
 context = {  
 'name': user.name,  
 'role': role  
 }  
 return render(request, "research\_proposal.html", context)

6.2.4 中期文档管理

（1） 界面截图



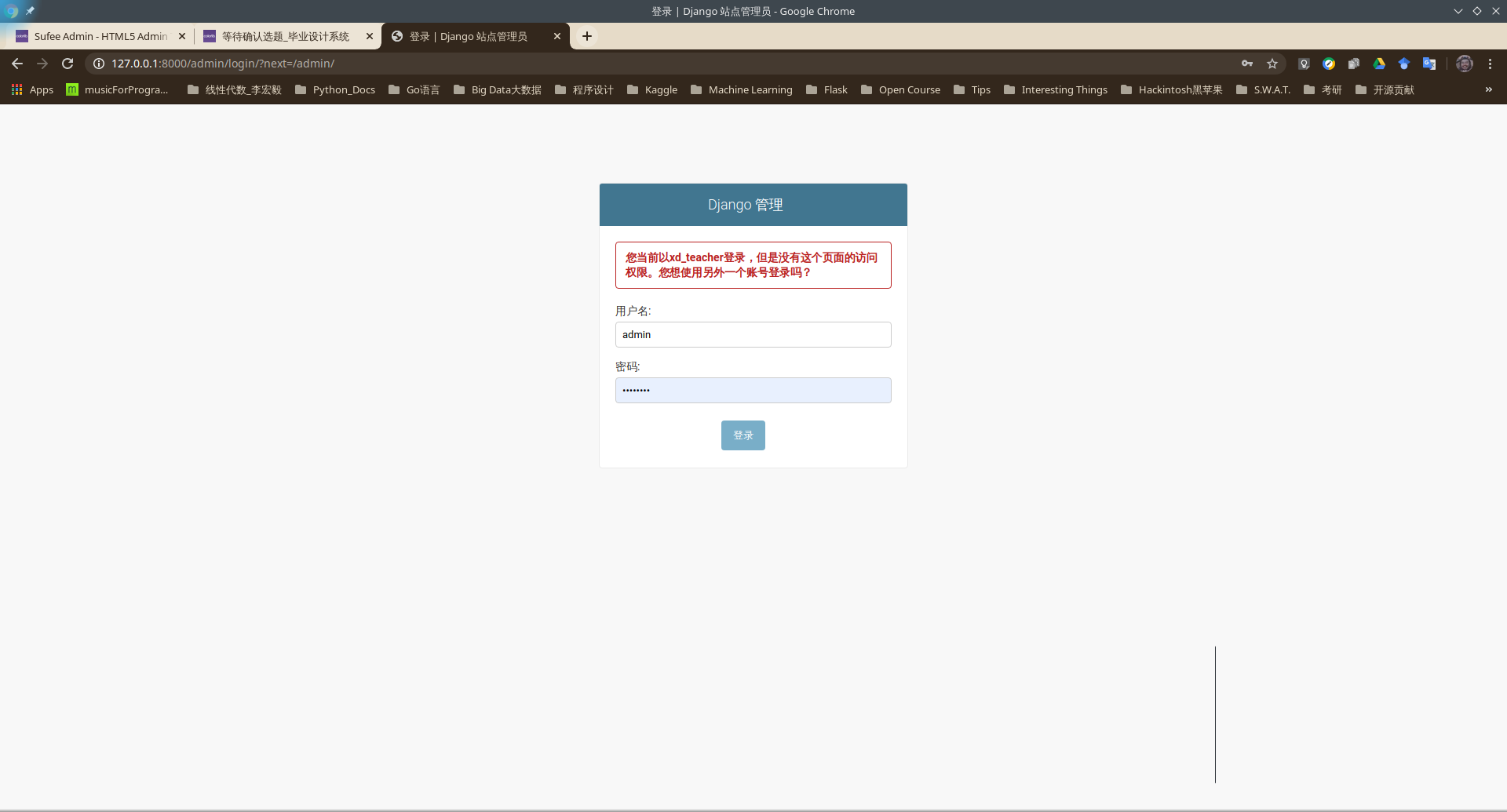
（2）代码

def show\_research\_proposal\_files(request):  
 # user = request.user  
 # t2u = Topic2User.objects.get(user\_id=user)  
 #  
 files = Annex.objects.get(id=14)  
 context = {  
 'files': files  
 }  
 return render(request, 'download\_file.html', context)

## 6.3 教学秘书功能实现

6.3.1 登录

（1）界面截图

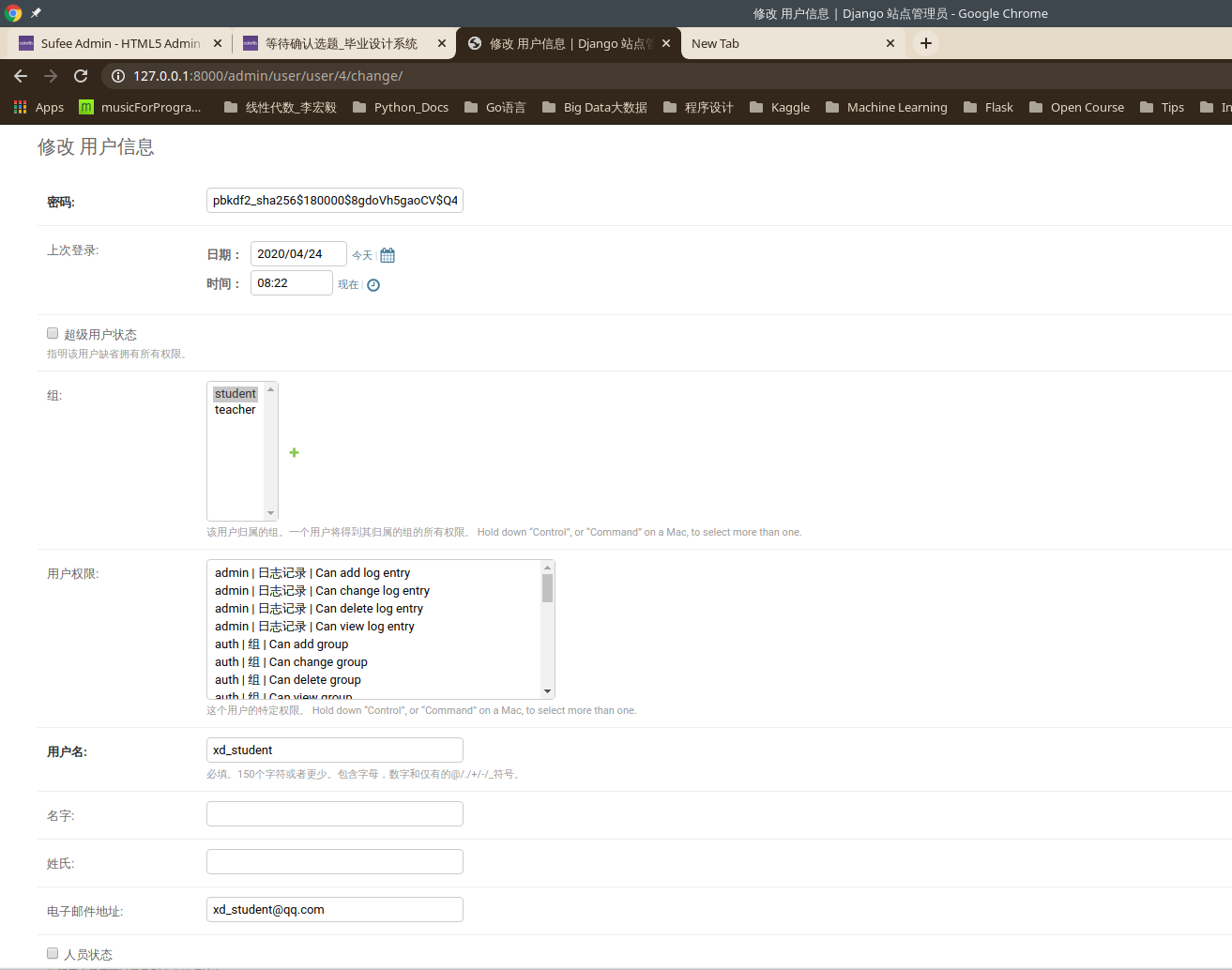


（2）核心代码

def login(self, request, extra\_context=None):  
 """  
 Display the login form for the given HttpRequest.  
 """  
 if request.method == 'GET' and self.has\_permission(request):  
 # Already logged-in, redirect to admin index  
 index\_path = reverse('admin:index', current\_app=self.name)  
 return HttpResponseRedirect(index\_path)  
  
 from django.contrib.auth.views import LoginView  
 # Since this module gets imported in the application's root package,  
 # it cannot import models from other applications at the module level,  
 # and django.contrib.admin.forms eventually imports User.  
 from django.contrib.admin.forms import AdminAuthenticationForm  
 context = {  
 \*\*self.each\_context(request),  
 'title': \_('Log in'),  
 'app\_path': request.get\_full\_path(),  
 'username': request.user.get\_username(),  
 }  
 if (REDIRECT\_FIELD\_NAME not in request.GET and  
 REDIRECT\_FIELD\_NAME not in request.POST):  
 context[REDIRECT\_FIELD\_NAME] = reverse('admin:index', current\_app=self.name)  
 context.update(extra\_context or {})  
  
 defaults = {  
 'extra\_context': context,  
 'authentication\_form': self.login\_form or AdminAuthenticationForm,  
 'template\_name': self.login\_template or 'admin/login.html',  
 }  
 request.current\_app = self.name  
 return LoginView.as\_view(\*\*defaults)(request)

6.3.2 用户角色与权限管理

（1） 界面截图



（2）核心代码

from rolepermissions.roles import AbstractUserRole  
  
  
class Teacher(AbstractUserRole):  
 available\_permissions = {  
 'create\_topic\_record': True,  
 'edit\_topic\_record': True,  
 }  
  
 @staticmethod  
 def get\_cls\_name():  
 return "教师"  
  
  
class Student(AbstractUserRole):  
 available\_permissions = {  
 # 'edit\_patient\_file': True,  
 }  
  
 @staticmethod  
 def get\_cls\_name():  
 return "学生"  
  
  
class SystemAdmin(AbstractUserRole):  
 available\_permissions = {  
 'create\_topic\_record': True,  
 'edit\_topic\_record': True,  
 'delete\_topic\_record': True,  
 }

6.5 FastDFS配置

Settings.py

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\_\_)))  
  
# Quick-start development settings - unsuitable for production  
# See https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/  
  
# SECURITY WARNING: keep the secret key used in production secret!  
SECRET\_KEY = '1c\*+e-!q0v$$l2+e%il\*(fv\_zo\_6reryyl%+\*l4nn\_o1qy^l+!'  
  
# SECURITY WARNING: don't run with debug turned on in production!  
DEBUG = True  
  
ALLOWED\_HOSTS = []  
  
# Application definition  
  
INSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
 'rolepermissions',  
 'user.apps.UserConfig',  
 'topic.apps.TopicConfig',  
]  
  
MIDDLEWARE = [  
 'django.middleware.security.SecurityMiddleware',  
 'django.contrib.sessions.middleware.SessionMiddleware',  
 'django.middleware.common.CommonMiddleware',  
 # 'django.middleware.csrf.CsrfViewMiddleware',  
 'django.contrib.auth.middleware.AuthenticationMiddleware',  
 'django.contrib.messages.middleware.MessageMiddleware',  
 'django.middleware.clickjacking.XFrameOptionsMiddleware',  
]  
  
ROOT\_URLCONF = 'bysjms\_rbac.urls'  
  
TEMPLATES = [  
 {  
 'BACKEND': 'django.template.backends.django.DjangoTemplates',  
 'DIRS': [os.path.join(BASE\_DIR, 'templates')]  
 ,  
 'APP\_DIRS': True,  
 'OPTIONS': {  
 'context\_processors': [  
 'django.template.context\_processors.debug',  
 'django.template.context\_processors.request',  
 'django.contrib.auth.context\_processors.auth',  
 'django.contrib.messages.context\_processors.messages',  
 ],  
 },  
 },  
]  
  
WSGI\_APPLICATION = 'bysjms\_rbac.wsgi.application'  
  
# Database  
# https://docs.djangoproject.com/en/3.0/ref/settings/#databases  
  
# DATABASES = {  
# 'default': {  
# 'ENGINE': 'django.db.backends.sqlite3',  
# 'NAME': os.path.join(BASE\_DIR, 'db.sqlite3'),  
# }  
# }  
  
DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.mysql',  
 'NAME': 'bysj\_rbac',  
 'HOST': '172.16.43.128',  
 'USER': 'root',  
 'PASSWORD': 'qq990110',  
 'PORT': 3306,  
 }  
}  
  
# Password validation  
# https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators  
  
AUTH\_PASSWORD\_VALIDATORS = [  
 {  
 'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',  
 },  
]  
  
# Internationalization  
# https://docs.djangoproject.com/en/3.0/topics/i18n/  
  
LANGUAGE\_CODE = 'zh-hans'  
  
TIME\_ZONE = 'Asia/Shanghai'  
  
USE\_I18N = True  
  
USE\_L10N = True  
  
USE\_TZ = True  
  
# Static files (CSS, JavaScript, Images)  
# https://docs.djangoproject.com/en/3.0/howto/static-files/  
  
STATIC\_URL = '/static/'  
STATICFILES\_DIRS = (  
 os.path.join(BASE\_DIR, "static"),  
)  
  
ROLEPERMISSIONS\_MODULE = 'bysjms\_rbac.roles'  
  
AUTH\_USER\_MODEL = 'user.User'  
  
LOGIN\_URL = '/login/'  
  
MEDIA\_URL = '/media/'  
  
MEDIA\_ROOT = 'media/'  
  
# 设置Django文件存储类  
DEFAULT\_FILE\_STORAGE = 'utils.fdfs.storage.FDFSStorage'  
  
# 设置fdfs使用的client.conf文件路径  
FDFS\_CLIENT\_CONF = './utils/fdfs/client.conf'  
  
# 设置fdfs存储服务器商Nginx的IP和Port  
FDFS\_URL = 'http://172.16.43.128:8888/'

6.5 models.py

6.5.1 topic.models.py

from django.db import models  
from user.models import User  
  
# Create your models here.  
class TopicRecord(models.Model):  
 limit\_num\_choices = (  
 (1, '限1人'),  
 (2, '限2人'),  
 (3, '限3人'),  
 )  
  
 status\_choices = (  
 (1, '未选'),  
 (2, '已选'),  
 )  
  
 accept\_choices = (  
 (0, '等待命题老师和教学秘书确认中'),  
 (1, '命题教师已确认选题，等待教秘确认'),  
 (2, '选题通过'),  
 )  
  
 title = models.CharField(max\_length=100, blank=False, verbose\_name='题目')  
 detail = models.TextField('描述')  
 chosen\_num = models.SmallIntegerField(verbose\_name="已选人数", default=0)  
 limit\_num = models.SmallIntegerField(choices=limit\_num\_choices, blank=False, verbose\_name='限选人数')  
 release\_time = models.DateTimeField("出题时间", blank=True, null=True, auto\_now=True)  
 last\_edit\_time = models.DateTimeField("最后编辑时间", blank=True, null=True, auto\_now\_add=True)  
 status = models.SmallIntegerField(choices=status\_choices, default=1, verbose\_name='选题状态')  
 accept = models.SmallIntegerField(choices=accept\_choices, default=0, verbose\_name='确认状态')  
  
 class Meta:  
 verbose\_name = '题目'  
 verbose\_name\_plural = verbose\_name  
  
 def \_\_str\_\_(self):  
 return self.title  
  
  
class Phase(models.Model):  
 '''  
 0-开题文档  
 1-中期文档  
 2-论文、答辩文档  
 3-Other  
 '''  
 phase = models.CharField(verbose\_name="文件所属阶段", max\_length=50)  
  
  
class Annex(models.Model):  
 file = models.FileField(upload\_to="fdfs", blank=True, null=True, verbose\_name="附件")  
 raw\_name = models.CharField(max\_length=128, blank=True, null=True, verbose\_name="原始文件名")  
 detail = models.TextField('描述', default=" ")  
 annex\_phase = models.ForeignKey(Phase, on\_delete=models.CASCADE, blank=True, null=True)  
  
  
class Annex2Topic2User(models.Model):  
 annex\_id = models.ForeignKey(Annex, on\_delete=models.CASCADE)  
 topic\_id = models.ForeignKey(TopicRecord, on\_delete=models.CASCADE)  
 user\_id = models.ForeignKey(User, on\_delete=models.CASCADE)  
  
  
class Topic2User(models.Model):  
 topic\_id = models.ForeignKey(TopicRecord, on\_delete=models.CASCADE)  
 user\_id = models.ForeignKey(User, on\_delete=models.CASCADE)  
  
 class Meta:  
 verbose\_name = '题目用户对应关系'  
 verbose\_name\_plural = verbose\_name

6.5.2 user.models.py

from django.db import models  
from django.contrib.auth.models import AbstractUser  
  
  
# Create your models here.  
class User(AbstractUser):  
 name = models.CharField(max\_length=30, verbose\_name="真实姓名")  
 school = models.CharField(max\_length=100, verbose\_name="学校", blank=True, null=True, default="")  
 department = models.CharField(max\_length=50, verbose\_name="所属院系", blank=True, null=True, default="")  
  
 class Meta:  
 verbose\_name = '用户信息'  
 verbose\_name\_plural = verbose\_name

# 第7章 系统测试

# 结束语

本系统基于Python，运用流行框架Django，有着良好的稳定性和可维护性，前端采用Bootstrap框架，界面简洁美观，图形化操作界面无入门门槛可直接使用。其特色在于Django+MySQL+FastDFS的结合，配合Nginx能在海量的文件存储与数据操作做到从容应对。帮助提高教学秘书、教师与学生之间的沟通效率，降低时间成本。

经过近四个月的忙碌与工作，本次毕业设计至此已接近尾声，作为一名本科毕业生，由于经验不足，难免有诸多考虑不周的地方。如若没有邹细涛老师的督促与指导，想要完成这次毕业设计是难以想象的。

而随着毕业论文的完成，我的大学生活也已进入尾声。春梦秋云，聚散真容易。

参考文献

[1] Elman J , Lavin M . Lightweight Django[M]. O'Reilly Media, Inc. 2014.

[2] 余庆. 分布式文件系统FastDFS架构剖析[J]. 程序员, 2010(11):65-67.

[3] Chun W. Core python programming[M]. Prentice Hall Professional, 2001.

[4] 殷锋. 软件工程[M]. 天津科学技术出版社. 2014.

[5]王珊, 萨师煊. 数据库系统概论(第5版)[J]. 中国大学教学, 2018, No.333(05):100.

[6] Filipe Ximenes. django-role-permissions [EB/OL]. https://django-role-permissions.readthedocs.io/

[7] Sanner M F . Python: a programming language for software integration and development.[J]. Journal of Molecular Graphics & Modelling, 1999, 17(1):57-61.

[8] Python Software Foundation. Python [EB/OL]. https://www.python.org/

致 谢

首先感谢邹细涛老师在各方面的指导和帮助，他平日工作繁忙，同时也忙于学业，能抽出时间给带领的应届毕业生指导毕业设计也相当不易，正是在他的细心指导下，我才能完成此次的毕业设计与毕业论文。虽然过程中遇到了很多问题，但是通过搜索引擎、询问老师、同学，我的问题都或多或少得到了解决。

感谢我的学校， “万丈高楼从地起”，若是没有学校近四年的培养，没有夯实的基础，那么我也无法完成毕业设计。

最后要感谢开源社区，本项目中的Python语言有许许多多的开发者贡献了他们的开源库，其中也包括了Django框架以及FastDFS，这给个人开发者在整个开发过程提供了诸多方便。