

**本科毕业设计（论文）**

题 目 基于Flask框架的视频播放网站

学院名称 计算机科学与技术学院

专业名称 软件工程

年级班级 1402

学生姓名 秦亚捷

指导教师 马永强

年 月

摘 要

随着社会的各个领域正在发生着越来越大的变化，人们想要提升自身价值，就得花时间去了解和学习新鲜的事物，观看视频就是一种比较好的途径，但是网上的视频资源比较分散，需要花时间去寻找，为了让人们快速的获取感兴趣的视频资源，我们有必要开发视频播放网站给人们提供视频资源来观看。

本网站是以Python为开发语言，使用Flask框架管理后台、Bootstrap框架完成界面的搭建以及http协议的post请求实现数据交互，实现用户搜索视频、观看视频、发表评论、收藏视频、个人信息管理，管理员对视频资源的增删改查以及对用户的收藏、评论的一些基本操作和简单的爬虫爬取一些基本的视频信息及其他模块的开发。使用MySQL数据库存储数据，实现网站基本功能。

本系统基本完成了课题的要求，采用的是Windows + PyCharm环境实现网站的开发，给人们提供一个观看视频的途径。

**关键词：**Python；Flask框架；爬虫

**ABSTRACT**

As more and more changes are taking place in various fields of society, people need to spend time to understand and learn new knowledge. It is a better way to watch videos, but online video resources more dispersed, it takes time to find, in order to allow people to quickly obtain video resources of interest, we need to develop a video playback site to provide people with video resources to watch.

This website uses Python as the development language. It uses the framework of the Flask framework to manage the background, the Bootstrap framework to complete the interface, and the post request of the http protocol to achieve data interaction. It enables users to search for videos, watch videos, post comments, collect videos, and manage personal information. The CCP crawled some of the basic video information and other modules on the addition, deletion, and review of video resources, as well as basic operations and simple crawlers on users' favorites and comments. Use MySQL database to store data and realize the basic functions of the website.

The system basically completed the requirements of the project, using the Windows and PyCharm environment to achieve the development of the site, to provide people with a way to watch the video.

**Keywords：**Python; The Flask Framework; Crawler

目录

[1.引言 I](#_Toc1180660562)

[1.1课题背景 I](#_Toc683458342)

[1.2目的和意义 I](#_Toc1404688909)

[2.系统分析与开发 3](#_Toc1419931039)

[2.1 可行性分析 3](#_Toc672559075)

[2.1.1技术可行性 3](#_Toc444673897)

[2.1.2经济可行性 3](#_Toc2125765051)

[2.1.3操作可行性 3](#_Toc1417866604)

[2.1.4法律可行性 3](#_Toc729921168)

[2.2 需求分析 4](#_Toc538202329)

[2.2.1 功能需求 4](#_Toc59316716)

[2.2.2 性能需求 4](#_Toc81531936)

[2.3 开发环境与工具 4](#_Toc342763667)

[2.3.1 PyCharm 4](#_Toc936399094)

[2.3.2 MySQL 5](#_Toc2022485729)

[2.3.3 Web服务器 5](#_Toc1702565396)

[2.4 技术介绍 5](#_Toc931206565)

[2.4.1 Python 5](#_Toc559573548)

[2.4.2 Flask框架 6](#_Toc1596548466)

[2.4.3 Bootstrap框架 7](#_Toc632038230)

[2.4.4爬虫技术 7](#_Toc676547485)

[3.网站设计 9](#_Toc237545868)

[3.1 基本思想 9](#_Toc771420829)

[3.2目标 9](#_Toc1665261785)

[3.3 功能模块划分 10](#_Toc187777623)

[3.3.1 总体图 10](#_Toc866275896)

[3.3.2 网站用例图 11](#_Toc306630296)

[4.系统设计 19](#_Toc956898551)

[4.1 数据库详细设计 19](#_Toc715268692)

[4.1.1 数据库需求设计 19](#_Toc1726537758)

[4.1.2 E-R图设计 19](#_Toc622422410)

[4.1.3 数据库结构设计 20](#_Toc1895929254)

[4.2 前端功能详细设计 25](#_Toc262512452)

[4.2.1 注册登录模块 25](#_Toc2027111319)

[4.2.2 个人收藏模块 28](#_Toc1168376645)

[4.2.3 搜索模块 29](#_Toc935071527)

[4.2.4 个人评论模块 29](#_Toc324301568)

[4.3 管理员功能详细设计 29](#_Toc1146658048)

[4.3.1 登录模块 30](#_Toc205454483)

[4.3.2 标签管理模块 30](#_Toc1054222736)

[4.3.3 视频管理模块 31](#_Toc1684860377)

[4.3.4 用户管理模块 32](#_Toc264771200)

[4.3.5 日志管理模块 32](#_Toc1135754672)

[4.3.6 权限管理模块 33](#_Toc2027624045)

[4.3.7 角色管理模块 34](#_Toc1201170294)

[4.3.8 管理员管理模块 34](#_Toc1010756753)

[4.4 爬虫功能详细设计 35](#_Toc1582705793)

[5 系统测试 38](#_Toc2132376859)

[5.1软件测试目的与意义 38](#_Toc1570330302)

[5.2 测试过程 38](#_Toc1031770611)

[5.2.1 登录模块测试 39](#_Toc616931441)

[5.2.2 搜索模块测试 39](#_Toc99394139)

[5.2.3 收藏模块测试 40](#_Toc1269316479)

[5.2.4 评论模块测试 40](#_Toc1388352270)

[5.2.5 个人模块测试 41](#_Toc1764655924)

[5.2.6 标签模块测试 41](#_Toc1457094102)

[5.2.7 视频模块测试 42](#_Toc107144518)

[5.2.8 用户模块测试 43](#_Toc2071286220)

[5.2.9 权限模块测试 43](#_Toc266509005)

[5.2.10 角色模块测试 44](#_Toc822413210)

[5.2.11 管理员模块测试 44](#_Toc1650340330)

[5.2.12 爬虫模块测试 45](#_Toc888931415)

[6 总结 46](#_Toc570858816)

[参考文献 48](#_Toc1912852783)

[致 谢 49](#_Toc768559086)

# 1.引言

## 1.1课题背景

随着时代的发展，社会的进步，社会的各个领域正在发生着越来越大的变化，人们想要在多变的社会情况下来充实自己，就得花费时间去了解和学习新鲜的事物，从而来快速的适应社会的变化，就如网上的那句话说的那样：如果一个人在八小时内没有了解和学习新鲜的事物，那么可能就会被这个社会所淘汰。正是基于这样的情况，每个人不得不花时间去了解和学习新鲜的事物，尤其是对一些社会所必须的技能和对社会事件以及时事政治，都得有着充分的了解或者学习。随着互联网的迅速发展，对于了解和学习的途径来说，也变得越来越多，过程也变的越来越简单，人们由原来只能通过文字和他人教授学习变成现在不仅仅可以通过文字还可以通过音频和视频来学习自己所需要的知识；音频的话，在收听（口音）上有局限性，而视频就没有，所以对于学习来说，观看视频就是一种比较好的方法，但是，在网上的大多数的视频资源比较分散，人们得自己去寻找，这得花费大量的时间和精力，极大得降低了学习的效率。基于这样的情况，我们的视频播放网站就出现了，为人们提供大量的视频资源来了解和学习新鲜的事物从而来充实自己。

从本质上讲，视频是通过影像和声音向人们传达出信息，让人们通过类似面对面的方式来实现学习和了解新的知识，通过这样的学习，人们才能够对学习新的知识有乐趣并对学习的内容有较深的印象。所以，我们的视频播放网站的主要功能就是给人们提供一种便捷的学习方式，让人们在比较短的时间内了解新的知识，获取有用的信息，还可以通过多次的了解加深印象，学习到新的知识，不仅扩展了自己的个人知识的储备量，而且也为以后做事提供极大的效率。

## 1.2目的和意义

在现如今这样一种经济飞速发展的状态下，互联网科技的进步让越来越多的数据信息进入人们的视线，怎样在这样的环境下了解和学习到新鲜的事物变得越来越重要。因此我们制作该视频网站，让人们获取到更加有效的视频资源去了解和学习新鲜的事物，该网站相当于在用户和视频信息之家搭建了桥梁，让视频信息展现在每个用户的面前，这样，让视频资源展现在用户面前，让用户实时了解社会的变化并且还可以学习，最大限度的帮助用户解决在网上这样分散的视频资源花费的时间和精力。

这样的话，一方面将视频资源上传至网站，使得人们通过使用视频播放网站观看视频，做到快速的获取自己感兴趣的视频资源，不再为视频资源而浪费精力和时间去网上找寻，另一方面，通过这样的方式，将自己所学的知识应用到毕业设计上，提升自己的动手操作能力，弥补自己在动手实践上面的不足，并将大学学到的知识切实应用到实处。

# 2.系统分析与开发

本节主要内容在于是对网站的开发工具进行介绍，并且将对该网站的需求性进行分析，还包括系统可行性和性能分析。

## 2.1 可行性分析

### 2.1.1技术可行性

随着互联网的高速发展，带动了社会的各个领域发生了重要的变化，人们想要在这样的环境下提升自身的价值，就需要不停的去了解和学习新鲜的事物，而对于传统的方式（看报、听收音）来说，已渐渐的跟不上社会变化的节奏，而新兴的观看视频的方式就渐渐的凸显出它的地位，而人们自己在网上寻找视频需要花费大量的时间和精力，效率并不是太高，需要一个比较全面的视频网站为其提供视频资源，正是基于此，我们编写了视频播放网站，来为人们提供一种便利的方式获取视频资源。该网站是一款基于Bootstrap+Flask框架实现前后端的视频播放网站，编程语言使用的是Python，使用的持久化数据库是MySQL数据库，Web服务器采用的是Flask自带的服务器，是一种名为Web服务器网关接口（Web Server Gateway Interface,WSGI）的协议，把接收自客户端的所有请求都转交给这个对象处理。客户端（例如Web浏览器）把请求发送给Web服务器，Web服务器再把请求发送给Flask程序实例。

### 2.1.2经济可行性

编写该网站使用的工具包括一台配置并不算太高的计算机（配置Win10操作系统），Python语言运行环境，剩下的也并不是太多，编写程序的IDE用的是社区版的PyCharm（免费），Flask框架和Boostrap框架都是免费的，作为最重要的本地数据库MySQL也是开源并且也是免费的。实现经济的可行性。

### 2.1.3操作可行性

该网站的设计与开发，是基于Python语言的Flask框架技术以及简单爬虫技术的基础上并结合前端框架Bootstrap框架完成的，并且前台页面都是使用Bootstrap框架实现，在Web界面展现，比较简洁美观，从界面、功能、操作方面上说都可以说是比较简单的。用户根据所感兴趣的去搜索视频资源，然后对其进行观看，避免了花费大量的时间和精力在网上寻找资源。实现操作的可行性

### 2.1.4法律可行性

该网站是有个人对其进行编写，不存在侵权行为，本网站的用户信息和管理员信息都是是通过加密保护的，确保其安全性，不存在泄露隐私行为，同时使用的技术都是开源的的，因此不存在侵权行为所以在法律上是可行的。

## 2.2 需求分析

需求分析在软件计划阶段是非常重要的活动，也是在软件生存周期中的一个十分重要的环节，在该阶段，主要是在于分析系统在功能上需要“实现什么”，并不是考虑该如何去“实现”这些功能。需求分析的目标在于把用户对需要的软件的“要求”或“需要”进行分析并整理，然后在确认后形成描述完整的、清晰的与规范的文档，确定软件需要实现哪些功能和完成哪些工作。此外，对于软件的一些非功能性需求(如软件性能、可靠性、响应时间、可扩展性等)、软件设计的约束条件以及运行时与其他软件的关系等，这也是软件需求分析的主要目标。

### 2.2.1 功能需求

该网站是在Web上拥有视频播放功能的视频播放网站，前台界面展示的是所要播放的视频资源，需要显示视频列表让用户查看视频详情并播放。对于用户来说，需要用户的基本操作功能（登录、注册、修改个人信息和查看收藏等），对于管理员来说，需要有对用户的基本操作、对视频资源的基本操作、日志管理和各种权限的查看，数据库的需要存储所需要的视频资源以及用户和管理员的一些基本信息和操作。

### 2.2.2 性能需求

（1）该网站的爬虫功能是爬取Bilibili网站的视频资源信息，然后保存在本地数据库，在网站使用的时候进行调用。

（2）该网站只需在本地运行即可完成操作。

## 2.3 开发环境与工具

该网站的开发语言为解释型计算机程序设计语言Python，操作系统是Win10，使用的IDE是用PyCharm2017.2，利用Flask框架开发后台，用Bootstrap框架设计前台页面，利用简单爬虫爬取数据并存入数据库中，持久化数据库使用的是MySQL数据库。

### 2.3.1 PyCharm

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，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。它支持多个平台的开发，能让用户在使用时提高开发的效率。在使用时可以终止进程，支持用户在使用时无法解除控制的问题。正是基于这样强大的功能上，PyCharm一致成为程序员公认的编写Python最好的IDE，此外，该IDE提供了一些高级功能，以用于支持Django框架和Flask框架下的专业Web开发。

### 2.3.2 MySQL

本系统采用的数据库是MySQL数据库，MySQL数据库是一个关系型数据库，属于一种小型的关系型的SQL数据管理系统，采用的SQL 语言是用于访问数据库的最常用标准化语言，由瑞典MySQL AB 公司研发，后被Orcale收购，现在是Oracle 旗下产品。MySQL 是当下最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL是最好的关系数据库管理系统应用软件。由于其源码属于开源状态，使得在开源社区很受广大程序员的喜爱，对于使用其的开发人员来说，虽然MySQL没有像SQLServer等其他数据库那么大，但是在功能上面并没有因为其“体积小”而使得功能不完善，也正是由于其体积小、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。

### 2.3.3 Web服务器

Web服务器使用的Flask框架自带的服务器，这个服务器使用的是一种名为Web服务器网关接口（Web Server Gateway Interface,WSGI）的协议，把接收自客户端的所有请求都转交给这个对象处理。客户端（例如Web浏览器）把请求发送给Web服务器，Web服务器再把请求发送给Flask程序实例。

## 2.4 技术介绍

### 2.4.1 Python

Python 是一种面向对象的的计算机程序语言，具有解释性、编译性和互动性。它的设计具有很强的可读性，和其他的语言相比，在英文关键字和标点符号上面，它具有更强的特色语法结构。Python具有丰富和完善的基础代码库，包含了网络、文件、GUI、数据库和文本等大量内容，用Python作为开发语言，许多功能都是可以直接使用的。Python还有强大的第三方库。国内外很多网站都使用Python开发，比如国内的网站有豆瓣、快盘等，国外的有YouTube、Reddit。

简单来讲，使用Python有以下特点：

（1）简

Python的简有两个方面，一个方面是简单，Python是一种简单主义思想的语言的代表。阅读一个良好的Python程序就感觉像是在读英语一样，尽管这个英语的格式要求比较严格。伪代码本身是它最大的优点之一，它使你能够专注于解决问题而不是去搞明白语言本身到底是什么。另一个方面是简练，Python语言编写的代码简练，比其他主流的设计语言如JAVA，C++等编写的代码量少。Python语法简洁清晰，用Python语言能够简单编写算法，并且编写的算法代码非常的优雅，能够让人们容易理解算法实现的原理。

（2）丰富的库

Python标准库确实很庞大。它可以帮助你处理各种工作，包括正则表达式、文档生成、单元测试、线程、数据库、网页浏览器、CGI、FTP、电子邮件、XML、XML-RPC、HTML、WAV文件、密码系统、GUI（图形用户界面）、Tk和其他与系统有关的操作。记住，只要安装了Python，所有这些功能都是可用的。这被称作Python的“功能齐全”理念。除了标准库以外，还有许多其他高质量的库，如wxPython、Twisted和Python图像库等等。

（3）方便阅读

对学习过其他编程语言熟悉的程序员，或者有一定编程能力的程序员来说，Python代码是很容易读懂的，并且极其容易上手编程。对新手来说，Python作为一种极简的代码格式，也是十分方便阅读，并且容易去理解这样的代码，Python代码编写风格独特，采用缩进的方式断句，整个代码段有层次感，让人们一眼就能感受到代排列的美感。

（4）粘合性好

Python是一门胶水性脚本语言，能够和C、JAVA等主流语言混合使用。在Python的第三方库中，很多都是使用C语言进行编写的，可以和Python很好的融为一体。

（5）易学

由于Python有着极其简单的语法格式，使得Python极易上手。

### 2.4.2 Flask框架

同其他框架相比，Flask 之所以能够脱颖而出，主要的原因就在于它让开发者自己做主，使开发者能够对程序具有全面的创意控制。或许你曾经听说过“和框架斗争”这样的一种说法。应为在大多数开发框架中，当你决定去使用的解决方案不受框架官方支持的时侯就会发生这种的情况。你可能想去使用不同的数据库引擎来存储数据或者使用不同的用户认证方法。但是，这种偏离了框架开发者设定好的路线的做法往往会带给你很多不必要的麻烦。但是Flask框架就不一样了。假如你喜欢关系型数据库？那好。Flask框架支持所有的关系型数据库的扩展。或许你更喜欢使用NoSQL数据库？那么没问题，Flask也支持这样的数据库。如果都不想使用，想去使用自己开发的数据库引擎存储数据？或者根本用不到数据库存储数据？那么依然没问题。在Flask框架中，你可以自主选择适合程序的组件，如果不能够找到合适的，还可以自己去开发。就是这么简单。Flask框架之所以能够给用户提供这么大的自由度，关键在于其开发伊始就考虑到了扩展性这一功能。Flask 框架还提供了一个强健的核心，其中包含每个Web 程序都需要的基本功能，而其他功能则被交给行业系统中的众多第三方扩展来实现，当然，你也可以自行开发。

Flask框架作为由Python编写的轻量级Web应用框架。基于Werkzeug WSGI(PythonWeb服务器网关接口（Python Web Server Gateway Interface，缩写为WSGI)是处理Python应用程序或框架和Web服务器之间的一种接口，已经被广泛采纳, 基本达成它的可移植性方面的功能)工具箱和Jinja2 模板引擎。 Flask使用BSD授权。因为它使用简单的核心，用extension增加其他功能，所以也被称为“microframework”。Flask没有默认使用的数据库、窗体验证工具，但是可以通过第三方扩展模块去实现这些功能。正是由于Flask保留了扩展的弹性，所以才使得Flask可以extension的功能很多，例如：ORM、窗体验证工具、文件上传、各种开放式身份验证技术。

### 2.4.3 Bootstrap框架

Bootstrap目前是最流行、受欢迎的Web前端框架之一，它能够快速的进行网站页面的搭建。它的优点在于将常用的CSS布局组件还有JavaScript插件进行了完整并完善的封装，让你在没有前端的经验下也能很快的搭建一个比较好的页面，从而来提高效率，。Bootstrap框架非常简洁灵活，能够快速实现Web开发。它包含以下部分：

（1）丰富的组件

Bootstrap框架带有大量的组件，这些组件使没有前端开发经验的我们能够快速的完成一个网站的前端设计，并且实现一个比较美观的网站的搭建。这些组件覆盖了导航栏、下拉菜单、警告框、弹出框、表格、表单、按钮、图片、辅助类和响应式工具等更多功能。

（2）完整的CSS样式

Bootstrap提供了大量的组件的CSS样式，包括全局的 CSS 设置、定义基本的 HTML 元素样式、可扩展的 class以及一个先进的网格系统。

（3）JavaSctrip插件

Bootstrap自带了13个jQuery插件，这些插件为其中丰富的组件赋予“生命力”的插件。其中包括：模式对话框、过渡效果、滚动监听、标签页、Collaps和carousel等。这些插件，可以单个引入，也可以一次性全部引入。

### 2.4.4爬虫技术

随着网络的迅猛发展，互联网成为大量信息的载体，如何有效的将这些信息收集并从中获取有用的信息成为了一个巨大的挑战，网络爬虫应运而生，网络爬虫作为一种便捷地从网上收集信息的方式，变得越来越有用。使用Python这样的简单编程语言，你可以使用少量编程技能就可以爬取复杂的网站。

用Python语言编写的网络爬虫，可以用于数据挖掘、监测和自动化测试的实现。网络爬虫一般从一个初始网页URL开始，不断的从URL页面上抽取新的URL放到队列并抽取数据，直到满足需求从而停止。一个完整的爬虫一般需要五个模块：URL管理器、HTML下载器、HTML解析器、数据存储器和爬虫调度器。

# 3.网站设计

前面是对该网站的分析需求还有开发程序所用到的技术，这一部分是对网站的总体思路的设计以及功能模块的简介，同时画出了必要的流程图和功能图，让大家可以更加清晰明白的去了解系该网站的功能。

## 3.1 基本思想

在整个视频播放网站中，主要分为前端、后台和爬虫三个模块，爬虫作为视频资源的获取工具，属于比较重要的一部分，爬虫通过在网上爬取视频资源信息，让后将其保存在本地数据库中，然后网站后台管理员可以通过数据库访问，或者直接对其进行操作。前端包括登录注册、搜索视频资源、查看我的收藏、查看评论以及评论模块。后台包括用户管理、视频资源管理、权限管理以及管理员管理。

登录和注册这两个功能模块是用来区分游客和用户的，只有当用户登录进去才能够对自己的基本信息进行查看以及修改自己的基本信息，而游客只能观看视频，对于其他操作并不能对其进行操作，视频资源有分类，使用者可以通过分类来找到自己感兴趣的视频资源。搜索可以很直接的帮助使用者寻找到自己感兴趣的视频资源。收藏和评论是用户的权力，只用登录才能够对其进行操作。

后台管理模块的话可以分为多个管理员，每个管理员拥有着对于后台管理的不同的权限，有的可以管理标签，有的可以管理用户，有的可以管理权限，还有的可以管理视频资源。当然管理员也分为超级管理员和普通管理员，超级管理员可以对任何一种模块进行操作，具有优先权，而上述的管理员只是普通管理员，只拥有后台管理的部分权限，并且每个管理员之间互不干扰。但是每个管理员对于后台管理列表类的查看都是有权限的，通过观察日志信息能够很好的从中获取其他管理员在后台进行的操作，对于错误的操作也能够很好的做到修改，并且对失误操作能够做到很好的复原。

## 3.2目标

该网站主要是为了方便给用户提供更加有效的视频资源，主要是将通过简单爬虫爬取的视频资源放在数据库中，管理员对其进行操作，实现对数据的集中处理后，然后展示在前台页面，让用户能够在前台页面得到自己想要的视频资源。

## 3.3 功能模块划分

### 3.3.1 总体图

（1）网站总体功能图。



图3-1 网站总体功能图

（2）网站流程图。



图3-2 网站流程图

该网站主要是能够实现为用户提供视频资源，进入网站的用户只能对视频进行搜索和观看，并不能对视频进行收藏和评论，只有当用户注册并登录后，才可以对其进行收藏及评论，并可以查看自己的评论记录和登录日志，并可以观看和删除自己收藏过的电影。超级管理员登录后台的话，可以实现标签管理、视频管理、用户管理、收藏管理、权限管理、角色管理和管理员管理，对于操作日志和登录日志也能够对其进行查看。普通管理员的话，只能根据自己的操作权限对其进行操作，并不能够跨权限操作。

### 3.3.2 网站用例图

用例是代表着某些用户可见的功能，是用与实现一个客户具体的目标。整个系统功能是由多个用例构成的，所有用例被设计完成后，系统的设计工作也就基本上完成了，这就是用例驱动的方法，在软件开发的过程中，采用用例驱动方式是Jacobson对软件界最重要的贡献之一。用例图定义了系统所要的功能需求，从用户的角度描述了系统的基本功能，并指出各个功能的操作者。主要元素就是用例、执行者和通信之间的联系。然后展示出系统功能。用UML开发软件是对用例进行迭代、渐增式地构造。用例图是所有用户所期望系统具备的功能，因为其定义了系统的行为特征，描述了系统功能的使用过程。

（1）用户用例图

用户注册登录模块

用户使用本网站可以进行注册和登录，该网站的部分功能只有在用户进行登录之后才能够对其进行操作，并且对于个人信息的查看以及查看评论记录和收藏历史也得在登录之后才能够实现。用户注册登录图如图3-3所示。



图3-3 用户注册登录用例图

用户个人中心模块

可以查看自己的基本信息并且对其进行修改。用户个人中心图如图3-4所示。



图3-4 用户个人中心用例图

修改密码模块

在每个用户登录之后，用户可以在该模块对自己的密码进行修改。修改密码图如图3-5所示。



图3-5 修改密码用例图

标签查询模块

游客和用户可以通过分类标签来获取到相应的标签类视频资源，帮助游客和用户发现和得到自己感兴趣的标签类视频。标签查询图如图3-6所示。



图3-6 标签查询用例图

关键词搜索模块

用户在搜索框中输入搜索关键词，进行搜索（一般是模糊搜索），可以实现简单的搜索。关键词搜索如图3-7所示。



图3-6 关键词搜索用例图

收藏模块

本模块主要实现登录用户添加收藏视频功能以及未登录用户的模态框登录，并且能够记录用户收藏的记录，在未收藏的时候能实现收藏，已经收藏过后的视频就不能再次进行收藏。收藏视频图如图3-7所示。



图3-7 收藏用例图

评论模块

本模块主要实现登录用户对视频进行评论的功能以及提示未登录用户的进行登录，并且在评论列表里面可以看到对该视频的评论。评论视频图如图3-8所示。



图3-8 评论用例图

（1）管理员用例图

登录模块

管理员分为超级管理员和普通管理员，各个管理员登录后，能操作的权限也不一样，并且互不关联（超级管理员除外）。管理员登录模块图如图3-9所示。



图3-9 管理员登录用例图

标签管理模块

对于视频资源来说，都有各自的标签对其进行分类，标签管理模块就可以为其添加标签，在添加视频资源的时候，可以选择所要添加的标签对视频资源进行分类操作。标签管理模块图如图3-10所示。



图3-10 标签管理用例图

视频管理模块

在爬虫对视频资源进行爬取之后，管理员需要对视频资源进行操作，添加视频资源以及查看和编辑视频资源，对一些有敏感字或者不正当的字眼的进行删除。视频管理模块图如图3-11所示。



图3-11 视频管理用例图

用户管理模块

在用户管理模块，可以查看用户并对其进行删除操作，对用户的评论列表也可以进行查看及删除，对于用户收藏的视频资源也可以查看并对其进行删除。视频管理模块图如图3-12所示。



图3-12 用户管理用例图

日志管理模块

在日志管理模块中，包含着用户登录日志、管理员登录日志以及管理员操作日志，管理员可以对这三个日志进行查看，但并没有权力对这三个日志进行操作。日志管理模块图如图3-13所示。



图3-13 日志管理用例图

权限管理模块

对于管理员来说，如果只要同一个管理员负责后台全部的管理，就比较耗费精力，况且如果一个管理员操作不当，会导致系统出现问题，所以，就需要权限来对其进行划分，从而来达到“分治”操作。在权限管理模块权限管理模块图如图3-14所示。



图3-14 权限管理用例图

角色管理模块

在该模块，主要是为了添加具有不同权限的的角色，从而来为添加管理员模块提供便利。角色管理模块图如图3-15所示。



图3-15 角色管理用例图

管理员管理模块

为了实现管理员的“分治”操作，就得为其添加不同权限管理员，然后才能够对后台进行不同权限的操作。但并不是每个管理员都有权限添加管理员，只有超级管理员或者管理员管理模块的管理员才能够添加普通管理员权限，超级管理员权限只有当前管理员是超级管理员才能够对其进行添加，管理员列表是每个管理员都能够对其进行查看的。该模块主要是添加和查看管理员。管理员管理模块图如图3-16所示。



图3-16 管理员管理用例图

# 4.系统设计

在这一节，主要是对数据库的详细设计，以及对每个功能的详细设计，是对于上一节网站初步设计的深度解释。

## 4.1 数据库详细设计

在每个程序中，都会需要数据库的应用，在程序设计初始的时候，数据库的设计就会被提出，然后会被讨论。数据库的设计是指对于一个已给定的应用环境，然后构造出最优的数据库模式，从而建立数据库及其应用系统，使之能够有效地将数据存储起来，从而满足各种用户的应用需求（信息要求和处理要求）。在数据库的领域内，常常把使用数据库的各类系统统称为数据库应用系统。一般来说。系统的设计总是根据实际需求，结合系统的总体设计思想，在满足系统功能实现的同时，尽量做到控制数据库中的数据表字段设置，保证数据库的效率。

### 4.1.1 数据库需求设计

数据库的需求分析是整个系统数据库开发完成的前提，数据库的设计直接影响了整个网站的总体设计。

该网站的操作着分为两个部分，一个用户部分，用户部分分为两种，一种是游客，只能够在网站上对视频进行搜索并观看，所需要的数据库就只有视频资源的数据库；另一种就是注册过并进行登录的用户，他们不止可以在网站上搜索视频并观看（对视频资源的数据库进行操作），还可以对其进行评论和收藏（对视频评论的数据库和视频收藏的数据库进行操作），当然，在每个用户登录的时候，都会产生登录日志（对用户登录日志数据库进行操作）。这就是所有用户的数据库的分析。另一个部分就是管理员，管理员分为两种，一种是超级管理员，对后台具有所有的权限（标签、视频、用户、日志、权限、角色和管理员管理），对应的是相应的数据表，数据存储在数据库中，另一种是普通管理员，对后台的权限根据所设定的权限进行对后台管理界面的操作，然后对应操作的是相对应得数据表。对于管理员的管理，只有超级管理员和具有用户管理权限的管理员才能够对其进行操作。而超级管理员只有超级管理员才能够对其进行添加。

### 4.1.2 E-R图设计

该网站包含管理员、用户、视频、和标签四个实体，管理员分为普通管理员和超级管理员，日志分为用户登录日志、管理员登录日志和管理员操作日志，标签分为主标签和子标签。

用户和用户登录日志是一对多的关系，用户和评论是一对多的关系，用户和收藏是多对多的关系，视频和评论、收藏是多对多的关系，视频和标签是多对多的关系。管理员和管理员登录日志是一对多的关系，管理员和角色之间是一对一的关系，管理员和视频、操作日志是多对多的关系，角色和权限来说是一对多的关系。如图4-1所示。



图4-1 E-R图设计

### 4.1.3 数据库结构设计

对于逻辑结构来说，它是独立于其他数据模型的，在实际应用生产中，一般所用的数据库的环境已经给定了逻辑结构的关系（如SQL Server、Oracle和MySql)。由于现在使用的数据库大部分都是关系数据库，因此在开始的时候都需要将E-R图转换为关系模型，然后根据具体的数据库管理系统(Database Management System)的特点和限制转换为特定的数据库管理系统支持下的数据模型，最后再对其进行优化。检查当前的E-R图是否满足的了当前业务的场景。可以从任意一个实体开始，沿着当前E-R图的各个节点去模拟具体的业务场景。尤其需要和《需求规格说明书》去做校验。

对于这个网站的数据库的表来书，分为十二个表，分别是管理员表（admin）、管理员登录日志表（adminlog）、权限表（auth）、主标签表（btag）、用户评论表（comment）、管理员操作日志表（oplog）、角色表（role）、子标签表（stag）、用户表（user）、用户登录日志表（userlog）、视频资源表（video）和视频收藏表（videocol），每个表都有自己的功能，下面是对于每个表的详细设计：

管理员（admin）表是比较重要的表，对于管理员表来说，存在三个表（oplog、admin和role表）与之有外键关联关系，由字段is\_super（0是，1不是）来区分是否为超级管理员。普通管理员和超级管理员的权限不同，能实现的操作也不相同。管理员（admin）表如表4-2所示。

表4-2 管理员表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| name | varchar | 是 | 管理员账号 |
| pwd | varchar | 是 | 管理员密码 |
| is\_super | smallint | 是 | 是否为超级管理员（0是，1不是） |
| role\_id | int | 是 | 角色id（外键关联） |
| addtime | datetime | 是 | 创建时间 |

管理员登录日志（adminlog）表和管理员表具有外键关联的关系，管理员表和其是一对多的关系，每个管理员有多条登录日志的。管理员登录日志（adminlog）表如表4-3所示。

表4-3 管理员登录日志表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| admin\_id | int | 是 | 管理员id（外键关联） |
| ip | varchar | 是 | ip地址 |
| addtime | datetime | 是 | 登录时间 |

管理员操作日志（oplog）表和管理员表具有外键关联关系，是一对多的关系，每个管理员对于后台可以有多种操作，当然，对于不同权限的管理员来说，在对对应权限的后台管理进行操作的的时候，也能够产生操作日志。管理员操作日志（oplog）表如表4-4所示。

表4-4 管理员操作日志表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| admin\_id | int | 是 | 管理员id（外键关联） |
| ip | varchar | 是 | ip地址 |
| reason | varchar | 是 | 操作原因 |
| addtime | datetime | 是 | 登录时间 |

权限（auth）表是单独被拿出来存放权限的，角色表与之对应有一定的关系，当添加角色的时候，会执行查询语句，来对权限表进行遍历，寻找到所需要的权限，对角色进行设定。权限（auth）表如表4-5所示。

表4-5权限表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| name | varchar | 是 | 权限名称 |
| url | varchar | 是 | 权限路由地址 |
| addtime | datetime | 是 | 添加权限时间 |

用户（user）表是其中比较重要的表，对于用户来说，存在三个表（comment、videocol和userlog）与之有外键关联关系，基本上都是一对多的关系，每个用户都有多条数据。用户（user）表如表4-6所示。

表4-6 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| name | varchar | 是 | 用户名称 |
| pwd | varchar | 是 | 用户密码 |
| emaill | varchar | 是 | 用户邮箱 |
| phone | varchar | 是 | 用户手机号 |
| info | text | 是 | 个性签名 |
| face | varchar | 是 | 用户头像 |
| addtime | datetime | 是 | 创建时间 |
| uuid | varchar | 是 | 唯一标识符 |

用户登录日志（userlog）表和用户表具有外键关联的关系，用户表对其属于一对多的关系，每个用户一般都有多条登录日志。用户登录日志（userlog）表如表4-7所示。

表4-7 用户登录日志表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| user\_id | int | 是 | 用户id（外键关联） |
| ip | varchar | 是 | ip地址 |
| addtime | datetime | 是 | 登录时间 |

用户评论（comment）表和用户表以及视频资源表都具有外键关联的关系，用户表对其属于一对多的关系，每个用户对每个视频可以有多条评论，和视频表属于多对多的关系，对于每一个视频来说都有多条评论。用户评论（comment）表如表4-8所示。

表4-8 用户评论表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| content | text | 是 | 评论内容 |
| video\_id | int | 是 | 视频id（外键关联） |
| user\_id | int | 是 | 用户id（外键关联） |
| addtime | datetime | 是 | 评论时间 |

用户收藏（videocol）表和用户表以及视频资源表都存在外键关联的关系，用户表对其是一对多的关系，每个用户都有多个视频收藏，和视频表属于一对一的关系，对于一个视频资源来说，每个用户只能收藏一次。用户收藏（videocol）表如表4-9所示。

表4-9 用户收藏表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| user\_id | int | 是 | 用户id（外键关联） |
| ip | varchar | 是 | ip地址 |
| addtime | datetime | 是 | 收藏时间 |

视频资源（video）表在几个表中属于最重要的表，存在三个表（comment、videocol和stag）与之有外键关联的关系，和子标签表属于一对一的关系，每个子标签对应的是一个视频资源。视频资源（video）表如表4-10所示。

表4-10 视频资源表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| title | varchar | 是 | 视频标题 |
| url | varchar | 是 | 视频地址 |
| info | text | 是 | 视频简介 |
| logo | varchar | 是 | 视频封面 |
| star | smallint | 是 | 星级 |
| playnum | bigint | 是 | 播放量 |
| commentnum | bigint | 是 | 评论量 |
| stag\_id | int | 是 | 子标签id |
| addtime | datetime | 是 | 添加时间 |

子标签（stag）表和视频资源表和主表前表都存在外键关联的关系，主标签对其属于一对多的关系，每一个主标签都存在一个或者多个字标签。子标签（stag）表如表4-11所示。

表4-11 子标签表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| btag\_id | int | 是 | 主标签id（外键关联） |
| name | varchar | 是 | 子标签名称 |
| addtime | datetime | 是 | 添加时间 |

主标签（btag）表和子标签表存在外键关联。主标签（btag）表如表4-12所示。

表4-12 主标签表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 类型 | 允许为空 | 描述 |
| id | int | 否 | 主键，自增 |
| name | varchar | 是 | 主标签名称 |
| addtime | datetime | 是 | 添加时间 |

## 4.2 前端功能详细设计

在前端页面，主要实现的是用户的一些功能，主要包括注册登录、搜索功能、个人信息中心、收藏和评论等功能。

### 4.2.1 注册登录模块

在注册页面含有的注册信息有账号、邮箱、手机号、密码和确定密码这几个方面，这几个方面都是必须填写的，如果在某个字段缺填的情况下，会显示这个字段未填写，请填写的提示语句，如果两次密码输入不一致，就会阻止用户进行注册，并重新填写密码的信息。注册页面如图4-13所示。



图4-13 注册页面

注册页面通过设置表单的提交信息来达到我们所要达到的效果，在提交信息的过程中对信息进行分析并判断信息是否为空或者两次密码是否一致，实现注册的功能。

def sign\_up(request):

if request.method== 'POST':

reg\_form = UserForm(request.POST)

if reg\_form.is\_valid():

username = reg\_form.cleaned\_data['username']

password = reg\_form.cleaned\_data['password']

c\_password = reg\_form.cleaned\_data['c\_password']

email = ' '

if password!= c\_password:

return render(request, 'sign\_up.html', {'script':"alert", 'info':'两次密码不一致，请检查'})

else:

try:

user = User.objects.create\_user(username, email, password)

user.save()

infer.objects.create(username=username)

return render(request, 'sign\_up.html', {'script':"alert", 'info':'用户注册成功'})

except:

return render(request, 'sign\_up.html', {'script':"alert", 'info':'用户名已被使用，请更换用户名'})

else:

return render(request, 'sign\_up.html', {'script':"alert", 'info':'注册失败，请重试'})

return render(request, 'sign\_up.html')

在登录功能中，将实现用户的登录，在登录的过程中使用到session，在客户端用cookie来识别。登录界面如图4-14所示。



图4-14 登录界面

登录页面通过设置表单的提交来实现登录功能，在提交的时候对根据账号对密码（已加密）进行查询，判断密码是否正确，正确的话就进入。

def sign\_in(request):

if request.user.is\_authenticated:

return redirect('/')

if request.method== 'POST':

username = request.POST.get('username')

password = request.POST.get('password')

user = authenticate(username=username, password=password)

if user:

auth\_login(request, user)

return redirect('/')

else:

return render(request, 'sign\_in.html', {'script':"alert", 'wrong':'登录失败，用户名或密码不正确'})

return render(request, 'sign\_in.html')

### 4.2.2 个人收藏模块

收藏视频模块的功能主要针对的是已经登录的用户，在使用网站并没有进行登录的时候，收藏列表按钮并不会出现，用户登录后，收藏视频按钮就会出现，用户就可以对其进行收藏，并且收藏后并不能对其进行二次收藏，除非删除已经收藏的视频。收藏界面如图4-15所示。

图4-15 收藏图书界面

收藏后可以对其进行删除，当删除了已经收藏的视频资源后，可以对其再次收藏。删除收藏如图4-16所示。

图4-16 删除图书界面

删除视频收藏的代码如下：

def remove(request):

if request.method== 'POST':

username = request.POST.get('username')

b\_id = request.POST.get('id')

try:

user = infer.objects.get(username=username)

fav\_book = book.objects.get(id=int(b\_id))

user.fav.remove(fav\_book)

return HttpResponse('移除图书成功')

except:

return HttpResponse('移除失败')

return HttpResponse('收藏失败，系统出错')

可通过在用户的个人中心中查看已收藏的视频列表，便可以知道用户是否已收藏了该视频，如图4-17所示。

图4-17 查看收藏图书

### 4.2.3 搜索模块

搜索模块可以在搜索框输入所要搜索的视频标题的关键字，然后对其进行搜索，不管用户是否登录都可以对其进行搜索。

### 4.2.4 个人评论模块

评论视频模块的功能也是主要针对的是已经登录的用户，在用户使用网站并没有进行登录的时候，评论功能并不会展示出来，用户登录后，评论框和评论按钮就都会展现出来，用户就可以对视频进行评论，可以多次评论，然后再下面的评论列表中找到自己的评论记录。评论界面如图4-18所示

图4-18 个人评论界面

## 4.3 管理员功能详细设计

在管理员模块，因为Flask框架不像Django框架那样直接有管理后台工具，对于Flask框架来说，得自行去编写，在和用户模块一起可以使用蓝图进行部署及注册，实现前后台模块的分离，前台模块和后台的逻辑编写不会相互产生影响，如果在一个文件中编写，由于要导入新的模块，可能会导致后台和前台的功能出错。对于一个系统来说，如果需要多个管理员对其进行维护，那么多个管理员在同时操作一个管理功能的时候，可能会出现较大的问题，而蓝图的出现，也能够解决这样的问题。在代码的简洁性上，使用蓝图的方式，会使得代码更具有简洁性。当然，使用蓝图的话还可以使得前后台可以使用两套不同的模板，使得前后台的页面都可以变得比较美观。

蓝图/Blueprint是Flask应用程序组件化的方法，可以在一个应用内或跨越多个项目共用蓝图，蓝图允许你将不同路由分开，提供一些规范（标准），并且附带了很多好处。使用蓝图可以极大地简化大型应用的开发难度，也为Flask扩展 提供了一种在应用中注册服务的集中式机制。

### 4.3.1 登录模块

在管理员登录模块，将实现管理员登录的功能，在登录的过程中，会去数据库对其进行查询，判断是否存在这个管理员，并且会根据其字段role\_id判断该管理员所拥有的权限，能够对后台管理所进行的权限操作。管理员登录界面如图4-19所示。



图4-19 管理员登录页面

### 4.3.2 标签管理模块

在管理员标签管理模块，具有超级管理员权限或者具有标签管理权限的管理员才能够对其进行操作。在该模块，要实现四个功能，分别是添加主标签、添加子标签和标签列表项的编辑和删除。主标签的添加单独的使用输入框实现，在子标签的添加需要根据主标签的列表进行添加，编辑标签和添加子标签的页面基本一样，删除功能的实现也是根据子标签进行删除。子标签的添加如图4-20所示。



图4-20 添加子标签页面

### 4.3.3 视频管理模块

在管理员视频管理模块，具有超级管理员权限或者具有视频管理权限的管理员才能够对其进行操作。在该模块，主要实现三个功能，分别是添加视频、视频列表项的编辑和删除。添加视频资源的模块如图所示，编辑资源和添加莫资源模块基本上是相同的，删除视频资源根据视频资源的编号（id）进行删除，同时也会将该视频所属的子标签一起删除，从而来视频资源的删除。视频添加如图4-21所示。

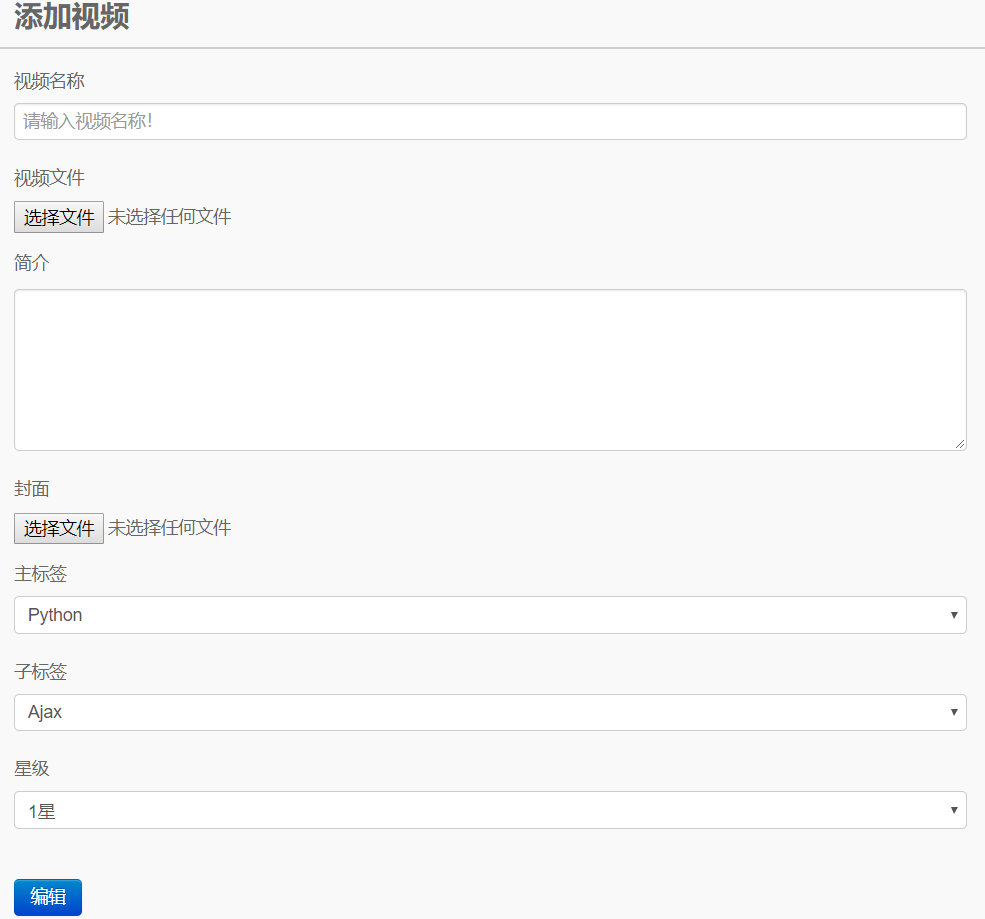


图4-21 添加视频页面

### 4.3.4 用户管理模块

在管理员用户管理模块，主要分为三部分（用户列表、评论列表和收藏列表），只有具有超级管理员权限或者具有用户管理权限的管理员才能够对其进行操作。这三部分要实现的基本相同的功能是删除功能，对于用户列表项、评论列表项以及收藏列表项都可以对其进行删除，但是要注意的是删除这三个任意一个列表项会对用户产生比较严重的影响，导致用户不能够很好的使用该系统，在对用户列表项可以实现的是查看，单独的将用户信息展现出来。

### 4.3.5 日志管理模块

在管理员日志管理模块，主要分为三个部分（操作日志列表、管理员登录日志列表和用户登录日志列表），对于所有的管理员来说，都能对其进行查看，在该模块，并没有实现操作的功能，只允许管理员对其进行查看。对于操作日志列表来说，管理员可以相互知道其他管理员对后台进行的操作，对于用户和管理员登录日志列表来说，能够获取用户和管理员登录的ip和登录时间。用户和管理员登录日志的列表基本相同，管理员操作日志存在操作原因。列表图如图4-22、图4-23所示。



图4-22 管理员/用户登录日志页面



图4-23管理员操作日志页面

### 4.3.6 权限管理模块

在管理员权限管理模块，主要分为两个部分（添加权限和权限列表），只有具有超级管理员权限或者具有权限管理权限的管理员才能够对其进行操作。在该模块，主要实现三个功能，分别是添加权限以及权限列表项的编辑和删除。在添加权限的功能上，需要对两个输入框进行输入，第一个输入框是权限名称，第二个是权限路由，编辑权限功能和添加权限的功能基本是相同的，删除权限的话，会根据权限的编号（id）对其进行删除。添加权限页面和编辑权限页面基本相同。页面如图4-24所示。

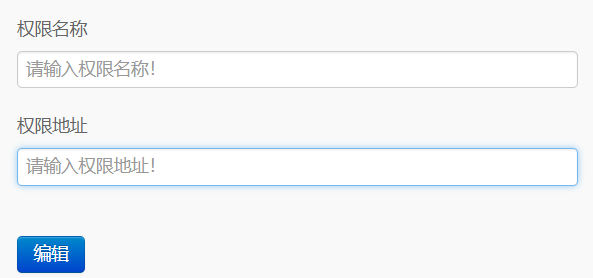


图4-24 添加/编辑权限页面

### 4.3.7 角色管理模块

在管理员角色管理模块，主要分为两个部分（添加角色和角色列表），只有具有超级管理员权限或者具有角色管理权限的管理员才能够对其进行操作。在该模块，主要实现三个功能，分别是添加角色以及角色列表项的编辑和删除。在添加权限的功能上，需要角色名称框输入角色名称和从复选框选择所要给该角色添加的权限，编辑角色的功能和添加角色的功能基本上是相同的，删除角色的话，会根据角色的编号（id）对其进行删除，添加角色页面和编辑权限页面基本相同。页面如图4-25所示。



图4-25 添加/编辑角色页面

### 4.3.8 管理员管理模块

在管理员管理模块，主要分为两个部分（添加管理员和管理员列表），只有具有超级管理员权限或者具有管理员管理权限的管理员才能够对其进行操作。在该模块，主要实现三个功能，分别是添加管理员以及管理员列表项的编辑和删除。在添加管理员的功能上，如果不是超级管理员，就不能添加超级管理员，超级管理员可以对管理员列表项进行删除和编辑（不能修改管理员的密码），删除管理员的话，会根据管理员的编号（id）对其进行删除，普通管理员并不能对其进行操作。添加管理员页面如图4-26所示。



图4-26 添加管理员页面

## 4.4 爬虫功能详细设计

视频的主要信息来自Bilibili网站视频信息，为了获取到足量的数据来展示页面，增加网站的视频量，视频简单的爬虫技术对视频信息进行爬取。下面来简述该爬虫的设计流程。 首先进入bilibili网站首页，进入分类并对其进行查看，找到所需要的分类，并直接进入该分类（https://search.bilibili.com/all?keyword=(字段)&from\_source=banner\_search，keyword代表的是所要进行查询的关键字），然后对所展示页面进行Ajax加载的数据进行分析，请求的头部为一般为：

https://search.bilibili.com/api/search?search\_type=video

&keyword=

&from\_source=banner\_search

在其中开始的语句https://search.bilibili.com/api/search?search\_type=video和&from\_source=banner\_search是固定的，改变的只有keyword的值，一般来说，keyword的值就是所要分类的字段。根据该头部可以实现对其数据信息的获取。获取的信息一般都是json数据，对于json数据的处理上面，需要做的就是将json数据通过python的json模块的loads方法将其转化为python的字典数据，对其进行获取所需要的数据的操作。在获取的时候，由于所需要的key位置有时会不相同，需要通过判断该key值在什么位置从而来获取所需要的数据。具体的步骤如下。

（1）使用python的requests从url中获取数据，对其进行加工，有关代码如下。

# 获取主要数据  
 def getData(url):  
 user\_agent = getUserAgent()

headers = {'User-Agent': user\_agent} # 伪装成浏览器的方式对其进行访问  
 datas = json.loads(requests.get(url, headers=headers).text) # 获取所需要的数据

if datas["result"]["video"] != None:  
 return datas["result"]["video"]  
 return datas["result"] #通过判断语句判断所需要的数据的位置

（2）对返回的数据列表进行加工，然后对其进行迭代，获取每个列表项中所需要的数据。

logo\_temp = source.change\_filename(data["pic"].split('/')[-1])  
 video = Video(  
 id=data["id"], # 编号  
 title=data["title"], # 标题  
 url="20180522130139447c5eb7c33b4520ac6a70d2045458ac.mp4", # 地址  
 info=data["description"], # 简介  
 logo=logo\_temp, # 封面  
 star=random.randint(1, 5), # 星级  
 playnum=data["play"], # 播放量  
 commentnum=data["video\_review"], # 评论量  
 stag\_id=data["aid"], # 标签  
 )

最后将video插入到数据库中。

# 5 系统测试

## 5.1软件测试目的与意义

软件测试的目的就是在真实系统工作环境下通过与系统的需求定义作比较，从而来检验完整的软件配置项能否能和系统正确连接，并发现软件与系统或者子系统设计文档与软件开发合同规定不符合或者与之矛盾的地方。站在不同的立场，软件测试的目的也不一样。在用户的角度上面，希望通过软件测试从而来暴露出软件所隐藏的错位和缺陷，从而考虑是否可以接收该产品，从软件开发者的角度出发来看，希望表明软件产品不存在错误，从而来验证软件是否能够实现用户的需求，确定人们对软件质量的信息。从软件管理者的角度来说，希望能够花费有限的资源从而来达到该软件用户所希望达到的质量要求，经费和进度是首要考虑的焦点。软件测试的重点是在检测缺陷上，许多的比较重要的缺陷主要是来自于对需求和设计的误解、遗漏和不正确，早期的结构化静态测试可以用于缺陷的预防，因此，检测和预防已经成为软件测试的重要目标。

系统的测试是通过将确认测试的软件，作为整个基于计算机系统的一个节点，与计算机硬件、外设、支持软件、数据和人员等其他系统节点结合起来，在实际运行（使用）环境下，对计算机系统进行的全面的测试，是为了发现产品的缺陷并度量产品的质量，按照系统的功能和性能需求对其进行的测试。而且，系统测试还要去检验系统的一系列文档是否完整、有效。其次，系统测试的测试用例需要根据需求分析说明书从而来进行设计，并且在实际使用环境中来运行。最后，系统测试一般都使用黑盒测试技术，并且全都由同一个测试人员来完成。

对于软件开发的整个周期来说，系统测试是软件研发人员参加系统的综合测试，将软件项目加入到计算机系统中进行测试。一方面为系统测试提供必要的软、硬件及资料支持，另一方面从软件测试角度提出系统测试中关于软件的测试设计。

## 5.2 测试过程

在开始测试的时候，由于数据库是新创建的，并没有数据在数据库中存储，并没有能够测试的数据使用，为了使得测试能够正常的进行，在数据库中存储了一些进行测试的有效的数据，这样的话，就能够更真实的反映出系统的功能的实现的情况。

在测试的过程中，不断发现问题，经过反复的测试、调试，把一个个的问题给解决，最终系统就能够正常的运行。

### 5.2.1 登录模块测试

测试流程：

（1）打开前台页面，进入登录页面

（2）不输入账号和密码，点击登录

（3）输入错误的账号和密码，点击登录

（4）输入正确的账号和密码，点击登录

（5）测试结果

测试表格如表5-1所示。

表 5-1 客户/管理员登录模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 客户/管理员端登录 | 账号：null  密码：null | 请输入账号  请输入密码 | 请输入账号  请输入密码 | 合格 |
| 客户/管理员端登录 | 账号：momo  密码：null | 请输入密码 | 请输入密码 | 合格 |
| 客户/管理员端登录 | 账号：null  密码：q1245 | 请输入账号 | 请输入账号 | 合格 |
| 客户/管理员端登录 | 账号：momo  密码：q1245 | 成功登录 | 成功登录 | 合格 |

### 5.2.2 搜索模块测试

测试流程：

（1）打开前台页面，点击搜索

（2）在搜索框中输入python，点击搜索

（3）在搜索框中输入C++，点击搜索

（4）在搜索框中输入科技，点击搜索

（5）在搜索框中输入国产动画，点击搜索

（6）测试结果

测试表格如表5-2所示。

表 5-2 搜索模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 搜索模块 | python | 出现关于python的视频列表 | 成功出现 | 合格 |
| 搜索模块 | C++ | 出现关于C++的视频列表 | 成功出现 | 合格 |
| 搜索模块 | 科技 | 出现关于科技的视频列表 | 出现有含有科技关键字的视频 | 合格 |
| 搜索模块 | Java | 出现关于的Java列表 | 成功出现 | 合格 |

### 5.2.3 收藏模块测试

测试流程：

1. 打开前台页面，登录
2. 随便打开一个视频，进行收藏
3. 测试结果

测试表格如表5-3所示。

表 5-3 收藏模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 收藏模块 | python语句1 | 收藏成功 | 收藏成功 | 合格 |
| 收藏模块 | WSGI讲解 | 收藏成功 | 收藏成功 | 合格 |
| 收藏模块 | 提着心吊着胆 | 收藏成功 | 收藏成功 | 合格 |

### 5.2.4 评论模块测试

测试流程：

（1）打开前台页面，登录

（2）随便打开一个视频，进行收藏

（3）测试结果

测试表格如表5-4所示。

表 5-4 收藏模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 收藏模块 | python语句1 | 评论成功 | 成功 | 合格 |
| 收藏模块 | WSGI讲解 | 评论成功 | 成功 | 合格 |
| 收藏模块 | 提着心吊着胆 | 评论成功 | 成功 | 合格 |

### 5.2.5 个人模块测试

测试流程：

（1）打开前台页面，登录

（2）打开个人中心，填写个人信息，点击修改

（3）测试结果

测试表格如表5-5所示。

表 5-5 个人中心模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 个人中心模块 | 信息都不太填写 | 提示未填写信息 | 成功 | 合格 |
| 个人中心模块 | 填写部分信息 | 提示未填写信息 | 成功 | 合格 |
| 个人中心模块 | 信息全部填写 | 提示成功修改 | 成功 | 合格 |

### 5.2.6 标签模块测试

测试流程：

（1）打开管理员页面，登录

（2）打开标签管理，

（3）添加主标签，点击编辑

（4）添加子标签，点击编辑

（5）编辑标签，点击编辑

（6）删除标签，点击删除

（7）测试结果

测试表格如表5-6所示。

表 5-6 个人中心模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 标签模块 | 添加主标签python | 添加成功 | 失败 | 不合格 |
| 标签模块 | 添加主标签科技 | 添加成功 | 成功 | 合格 |
| 标签模块 | 添加子标签爬虫 | 添加成功 | 成功 | 合格 |
| 标签模块 | 添加子标签Scrapy | 添加成功 | 失败 | 不合格 |
| 标签模块 | 编辑标签爬虫 | 编辑成功 | 成功 | 合格 |
| 标签模块 | 删除标签科技 | 删除成功 | 失败 | 不合格 |
| 标签模块 | 删除标签科技 | 删除成功 | 成功 | 合格 |

在看到添加主/子标签不合格的时候，去查看源代码，发现未对其进行提交，解决之后，发现可以成功添加，在删除标签的时候，发现传的参数不对，会对其产生影响，修改之后能够成功删除。

### 5.2.7 视频模块测试

测试流程：

（1）打开管理员页面，登录

（2）打开标签管理

（3）添加视频资源，点击编辑

（4）编辑视频资源，点击编辑

（5）删除视频资源，点击删除

（7）测试结果

测试表格如表5-7所示。

表 5-7 视频模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 视频模块 | 添加视频python语句1 | 添加成功 | 失败 | 不合格 |
| 视频模块 | 添加视频python语句1 | 添加成功 | 成功 | 合格 |
| 视频模块 | 添加视频提着心吊着胆 | 添加成功 | 成功 | 合格 |
| 视频模块 | 修改视频python语句1 | 修改成功 | 成功 | 合格 |
| 视频模块 | 删除视频python语句1 | 删除成功 | 成功 | 合格 |

在添加视频失败的时候，去查看源代码，发现在源代码设置视频资源的保存路径上出现问题，经过修改之后，可以将视频资源成功保存。

### 5.2.8 用户模块测试

测试流程：

（1）打开管理员页面，登录

（2）打开用户管理

（3）查看用户列表，查看列表项，并删除

（4）查看评论列表，点击删除

（5）查看收藏列表，点击删除

（7）测试结果

测试表格如表5-8所示。

表 5-8 用户模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 用户模块 | 查看用户列表项 | 查看成功 | 成功 | 合格 |
| 用户模块 | 删除用户列表项 | 删除成功 | 成功 | 合格 |
| 用户模块 | 删除用户评论 | 删除成功 | 成功 | 合格 |
| 用户模块 | 删除用户收藏 | 删除成功 | 成功 | 合格 |

### 5.2.9 权限模块测试

测试流程：

（1）打开管理员页面，登录

（2）打开权限管理

（3）添加权限，点击编辑

（4）编辑权限，点击编辑

（5）删除权限，点击删除

（7）测试结果

测试表格如表5-9所示。

表 5-9 权限模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 权限模块 | 添加权限 | 添加成功 | 失败 | 不合格 |
| 权限模块 | 添加权限 | 添加成功 | 失败 | 不合格 |
| 权限模块 | 添加权限 | 添加成功 | 成功 | 合格 |
| 权限模块 | 添加权限 | 添加成功 | 成功 | 合格 |
| 权限模块 | 编辑权限 | 修改成功 | 成功 | 合格 |
| 权限模块 | 删除权限 | 删除成功 | 成功 | 合格 |

在权限的添加第一次出错误的时候，通过检查发现，并没有做按钮事件，所以在添加的时候什么也没有提示，通过修改后再次添加的时候，发现还是失败，然后再次去看代码，在代码中，发现没有对其进行入库操作，实现代码的入库后，再次添加权限，就成功了。

### 5.2.10 角色模块测试

测试流程：

（1）打开管理员页面，登录

（2）打开角色管理

（3）添加角色，在权限列表选择所要添加的权限，点击编辑

（4）编辑角色，在权限列表选择所要添加的权限，点击编辑

（5）删除角色，点击删除

（7）测试结果

测试表格如表5-10所示。

表 5-10 角色模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 角色模块 | 添加角色 | 添加成功 | 成功 | 合格 |
| 角色模块 | 编辑角色 | 修改成功 | 成功 | 合格 |
| 角色模块 | 删除角色 | 删除成功 | 成功 | 合格 |

### 5.2.11 管理员模块测试

在该模块的测试方面，使用的是具有后台管理所有权限的超级管理员，可以对后台所有的内容进行管理。

（1）打开管理员页面，登录

（2）打开管理员管理

（3）添加管理员，点击编辑

（4）编辑管理员，点击编辑

（5）删除管理员，点击删除

（7）测试结果

测试表格如表5-11所示。

表 5-11 管理员模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 管理员模块 | 添加管理员 | 添加成功 | 成功 | 合格 |
| 管理员模块 | 编辑管理员 | 修改成功 | 成功 | 合格 |
| 管理员模块 | 删除管理员 | 删除成功 | 成功 | 合格 |

### 5.2.12 爬虫模块测试

（1）编写爬虫模块，执行爬虫

（2）爬出数据入库

（7）测试结果

测试表格如表5-12所示。

表 5-12 管理员模块测试表格

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模块名称 | 测试用例 | 预期结果 | 实际结果 | 是否合格 |
| 爬虫模块 | 爬取python | 爬取成功 | 失败 | 不合格 |
| 爬虫模块 | 爬取python | 爬取成功 | 成功 | 合格 |
| 爬虫模块 | 爬虫科技 | 爬取成功 | 成功 | 合格 |
| 爬虫模块 | 爬取汽车 | 爬取成功 | 成功 | 合格 |

在第一次爬取的时候，发现在爬虫获取的json数据转化成为字典的时候出现问题，发现忘记将获取的网页内容提取出来，只得再次对其提取，再次爬取就成功了。

# 6 总结

在Python的学习方面，自己是从大四开始的时候就开始对其进行学习，然后对爬虫学习的比较多，对于Python的一些Web框架只是做了简单的了解，并不能够很好的对其进行运用。在大四下班学期开始的时候就想了一下自己的毕设可能会做的东西，如果做分布式爬虫的话，对于硬件的要求比较高，并且对数据量的要求比较大，再考虑到费用的问题，就想到了做网站。然后就开始接触Flask框架和其他Python的Web框架，着重的去学习Flask框架为做网站打下基础。当然，考虑到要做网站，就需要去了解HTML5+CSS3和JavaScript。

通过对视频播放网站的开发，发现自己还是有很多的不足的地方。在开始的时候，就开始考虑数据库，然后根据E-R图创建数据表来存放数据，对于数据表的创建来说还是比较容易操作的。然后开始网页的设计，发现自己对于网页只有基础的了解，并不能将其编写出来，还是在同学的帮助下，才渐渐的了解了页面的搭建，然后根据自己的需求去设计网页，最后对其实现功能，在开始实现功能上面，发现自己并不能很好的解决各个功能之间的联系，就又去看了看大二时候的基础课程书（数据结构和Java），最后实现全部的功能并对其进行测试，发现各个功能之间能够做到正常运行。

经过一个多月的努力，基于Flask框架的视频播放网站的开发和设计就基本完成了，完成了前端Bootstrap框架、Flask框架和简单爬虫技术的开发。

# 参考文献

[1] 高铁军译.Python Cookbook中文版.北京：人民邮电大学，2010

[2] 王海鹏译.Python编程快速上手-让繁琐工作自动化.北京：人民邮电出版社，2016.7

[3] 司维译.Python基础教程.北京：人民邮电大学，2010.7

[4] 安道译.Flask Web开发. 北京：人民邮电大学，2015.1.1

[5] 范传辉. Python爬虫开发与项目实践.北京：机械工业出版社，2017.3

[6] 刘卫国. Python语言程序设计.北京：电子工业出版社，2016.5.1

[7]Bootstrap. http://baike.so.com/doc/5866541-7589096.html [EB/OL], 2017-05-05

[8] Scrapy. http://baike.so.com/doc/1853401-1960020.html [EB/OL], 2017-05-05

[9] 廖雪峰的官方网站：https://www.liaoxuefeng.com/

[10] 李辉.数据库系统原理及MySQL应用教程.北京：机械工业出版社，2016

[11] 严蔚敏，李冬梅，吴伟民.数据结构（C语言版）.北京：人民邮电出版社，2014.11

# 致 谢

时间总是在不经意间流逝，转眼间，我们已经将大学四年过完，在大学的四年中，有遗憾也有收获，大学四年的生涯中，遇到了很多帮助过我的人，有老师，也有校友。

首先感谢的就是我的指导老师马永强老师，从开始做毕设的时候，就开始为我们操碎了心，在做毕设的时候，就指导过我该如何解决功能上面的一些逻辑关系，使得各个功能能够正常的运行，在毕设做好的时候，帮我指出毕设上面的所出现的问题和可能出现的问题，提示自己应该去解决这些问题；在我编写论文结束的后，帮我查看论文，并指出我在毕业论文不合适的地方，并帮助我修改论文。马老师对待我们就像孩子一样，非常细心，花费大量的时间和精力去帮助我们，在这说声老师辛苦了。

其次感谢的是在大二的时候教我们数据结构的老师申艳梅老师，在大二刚开始的时候，有一段时间，感觉自己对于编程并没有太大的兴趣，想到过要放弃，后来在申老师的课堂上，感觉到了代码的魅力，渐渐的发现编程也不是那么的无趣，然后也不排斥编程了。当然，对于在大学四年教导过我的老师，我也从心底里对他们表示感谢。

最后，感谢在大学期间一直帮助过我的家人、老师和同学，是他们的帮助让我一步步的提升着自己的能力，并学习到足够的知识。当然，也感谢自己，并没有放弃编程。