|  |  |  |
| --- | --- | --- |
| 学 号： |  | 2013-2014学年第2学期第19～20周 |

**软件设计报告**

题 目**： 教务精细化管理系统**

班 级**： B111108**

姓 名**： 江如意**

指导教师**： 徐润森**

成 绩**：**

管理学院

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2014 | 年 | 06 | 月 | 23 | 日 |

南京邮电大学

**软件设计指导教师成绩评定表**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 题　　目 |  | | | | | | | | | |
| 学生姓名 |  | 班级学号 | |  | | 专业 | 信息管理与信息系统 | | | |
| 评分内容 | 评分标准 | | | | | | | 总分 | | 评分 |
| 平时成绩 | 认真对待软件设计，遵守实验室规定，上机不迟到早退，不做和设计无关的事。 | | | | | | | 20 | |  |
| 设计内容 | 设计内容丰富，符合大纲要求 | | | | | | | 10 | |  |
| 界面外观漂亮、大方 | | | | | | | 10 | |  |
| 功能符合大纲要求 | | | | | | | 10 | |  |
| 数据分析设计正确合理 | | | | | | | 10 | |  |
| 文档写作 | 文档反映设计规划分析过程 | | | | | | | 10 | |  |
| 文档反映软件设计流程 | | | | | | | 10 | |  |
| 答辩 | 简练、准确阐述设计内容，能准确有条理回答各种问题，系统演示顺利。 | | | | | | | 20 | |  |
| 总评分 |  | | | | | | | | | |
| 指导教师  评阅意见 | 该生对待软件设计的态度 □认真 □良好 □一般 □比较差。  设计的系统界面外观美工布局情况 □很好 □较好 □一般 □比较差  系统前后台功能和脚本编写情况 □很好 □良好 □一般 □比较差  文档书写符合规划与设计流程程度 □很好 □良好 □一般 □比较差  系统演示顺利情况和答辩流畅准确情况□很好 □良好 □一般 □比较差  完成的软件设计特色之处： | | | | | | | | | |
| 成绩 |  | | 指导教师签名 | | 徐润森 | | 日期 | | 2014-07-5 | |

**教务精细化管理系统**

**课题简介**

**研究背景**

精细化管理又称精细管理、精益管理、零缺陷管理，是指管理活动的精确化、标准化、程序化和细致化。作为现代工业化时代的一个管理概念，最早是由日本企业在20世纪50年代提出来的。在我国，上世纪60年代的大庆油田、本世纪的海尔等一批企业都是实行精细化管理的优秀典范。清华大学可以算得上是国内精细化教务管理的典范之一。

有效的学籍管理是保证高校正常教学秩序，形成良好学风、教风和校风的重要保障，是体现学校管理水平的一项重要指标。

从宏观上来说，教务精细化管理研究或实现精细化管理是高等学校在人才培养中落实科学发展观的内在要求；是提高学校品位，依法治教，推进本科教学管理改革持续发展的动力；是以人为本，求真务实，构建服务型和谐校园的客观需要；是克服粗放管理、经营大学，创建现代大学教学管理制度的需要。在教务管理中，努力用细节体现特色、用细致弥补缺陷和用创新推动改革。没有合理完善的流程与程序，没有量化细节，就没有执行力、控制力和竞争力。

**项目研究的现状**

目前，我国大多数高校都实行二级管理制度，教务处负责统筹协调，二级学院负责具体事情的实施。由于教务处有多个科室，如教务科、教学科、实践科等，每个科室虽有交叉，但都有各自的行事履历，而二级学院通常只配备一个教务人员，这就导致教务人员有时会接到多个通知，直接体现出教务工作的最大特点：杂。如何能在繁杂的事务中做到按部就班、清楚准确，而不是顾此失彼、应付将就，这是提升教务管理工作质量的当务之急。

教务管理工作作为高校中最基本的管理工作，长期以来得不到应有的重视。由于教务工作是一项繁琐的、不起眼的服务和管理工作，尽管做的事情很多，但工作绩效不明显，经常会被上级领导忽视。同时，教务工作被认为是附属工作，在薪酬待遇、职称评审、培训等方面都比不上教师、辅导员，使教务人员潜意识地认为自身和教务工作不重要，极大地打击了工作的积极性。这一方面会导致教务熟练人员的流失，另一方面会使教务人员对待自己的工作随意，不会积极创新，这也是为什么教务管理没有形成系统的管理理论的最大原因。

鉴于此，本项目拟以教务精细化管理为背景，为减轻教务管理人员的工作繁杂性，提高准确性，为学校教务教务处设计与开发教务精细化管理系统。

# 网站开发工具

## 1.1BBedit与Sublime Text2简介

BBEdit是全球领先的专业HTML和文本编辑器用于Macintosh。具体响应的Web作者和软件开发商，这个奖项的需要，成功的产品制作编辑提供了一个高性能的功能丰富，搜索和文本操作。智能接口可方便地到达BBEdit最好的一流的功能，包括GREP图样匹配，搜索和替换多个文件（即使未开启的远程服务器上的文件），项目定义的工具，功能导航和众多的源代码语言的语法着色，代码折叠，FTP和SFTP打开和保存，AppleScript 的，Mac OS X的Unix的脚本支持，文字和代码完成，当然还有一个强大的HTML工具配套。

Sublime Text 是一个代码编辑器（Sublime Text 2是收费软件，但可以无限期试用），也是HTML和散文先进的文本编辑器。Sublime Text是由程序员Jon Skinner于2008年1月份所开发出来，它最初被设计为一个具有丰富扩展功能的Vim。

Sublime Text 是一个代码编辑器（Sublime Text 2是收费软件，但可以无限期试用），也是HTML和散文先进的文本编辑器。Sublime Text是由程序员Jon Skinner于2008年1月份所开发出来，它最初被设计为一个具有丰富扩展功能的Vim。

Sublime Text具有漂亮的用户界面和强大的功能，例如代码缩略图，Python的插件，代码段等。还可自定义键绑定，菜单和工具栏。Sublime Text 的主要功能包括：拼写检查，书签，完整的 Python API ， Goto 功能，即时项目切换，多选择，多窗口等等。[1] Sublime Text 是一个跨平台的编辑器，同时支持Windows、Linux、Mac OS X等操作系统。

2012年6月26日推出新版本的Sublime Text 2.0，与之前版本相比主要有较大的改善：支持 Retina 视网膜屏、快速跳到下一个、文本拖放、改善构建系统、CSS 自动完成和高亮设置等。

## 1.2Python简介

Python是一种解释型、面向对象、动态数据类型的高级程序设计语言。自从20世纪90年代初Python语言诞生至今，它逐渐被广泛应用于处理系统管理任务和Web编程。Python[1] 已经成为最受欢迎的程序设计语言之一。2011年1月，它被TIOBE编程语言排行榜评为2010年度语言。自从2004年以后，python的使用率是呈线性增长[2] 。

由于Python语言的简洁、易读以及可扩展性，在国外用Python做科学计算的研究机构日益增多，一些知名大学已经采用Python教授程序设计课程。例如麻省理工学院的计算机科学及编程导论课程就使用Python语言讲授。众多开源的科学计算软件包都提供了Python的调用接口，例如著名的计算机视觉库OpenCV、三维可视化库VTK、医学图像处理库ITK。而Python专用的科学计算扩展库就更多了，例如如下3个十分经典的科学计算扩展库：NumPy、SciPy和matplotlib，它们分别为Python提供了快速数组处理、数值运算以及绘图功能。因此Python语言及其众多的扩展库所构成的开发环境十分适合工程技术、科研人员处理实验数据、制作图表，甚至开发科学计算应用程序。

说起科学计算，首先会被提到的可能是MATLAB。然而除了MATLAB的一些专业性很强的工具箱还无法替代之外，MATLAB的大部分常用功能都可以在Python世界中找到相应的扩展库。和MATLAB相比，用Python做科学计算有如下优点：

● 首先，MATLAB是一款商用软件，并且价格不菲。而Python完全免费，众多开源的科学计算库都提供了Python的调用接口。用户可以在任何计算机上免费安装Python及其绝大多数扩展库。

● 其次，与MATLAB相比，Python是一门更易学、更严谨的程序设计语言。它能让用户编写出更易读、易维护的代码。

● 最后，MATLAB主要专注于工程和科学计算。然而即使在计算领域，也经常会遇到文件管理、界面设计、网络通信等各种需求。而Python有着丰富的扩展库，可以轻易完成各种高级任务，开发者可以用Python实现完整应用程序所需的各种功能。

## 1.3Web.py简介

web.py 是一个轻量级Python web框架，它简单而且功能大。web.py是一个开源项目。该框架由美国作家、Reddit联合创始人、RSS规格合作创造者、著名计算机黑客Aaron Swartz开发。web.py目前已被很多家大型网站所使用。

web.py简单易学，只要有[1] ，掌握web.py就非常容易。

Aaron Swartz于2013年1月11日自杀身亡，享年26岁。

## 1.4Mysql简介

MySQL[1] 是一个关系型数据库管理系统，由瑞典MySQL AB公司开发，目前属于Oracle公司。Mysql是最流行的关系型数据库管理系统，在WEB应用方面MySQL是最好的RDBMS(Relational Database Management System：关系数据库管理系统)应用软件之一。MySQL是一种关联数据库管理系统，关联数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。MySQL所使用的SQL语言是用于访问数据库的最常用标准化语言。MySQL软件采用了双授权政策（本词条“授权政策”），它分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择MySQL作为网站数据库。由于其社区版的性能卓越，搭配PHP和Apache可组成良好的开发环境。

## 1.5DWZ简介

DWZ富客户端框架(jQuery RIA framework)，是中国人自己开发的基于jQuery实现的Ajax RIA开源框架。 DWZ富客户端框架设计目标是简单实用、扩展方便、快速开发、RIA思路、轻量级.

DWZ框架支持用HTML扩展的方式来代替JavaScript代码，只要懂HTML语法， 再参考DWZ使用手册就可以做Ajax开发。

开发人员不写JavaScript的情况下，也能用Ajax做项目和使用各种UI组件。 基本可以保证程序员不懂JavaScript， 也能使用各种页面组件和Ajax技术。 如果有特定需求也可以扩展DWZ做定制化开发。

做Ajax项目时需要写大量的JavaScript才能达到满意的效果， 国内很多程序员javascript不熟， 大大影响了开发速度。使用DWZ框架自动邦定JavaScript效果， 不需要开发人员去关心JavaScript怎么写，只要写标准HTML就可以了。DWZ简单扩展了HTML标准，给HTML定义了一些特别的class和attribute。 DWZ框架会找到当前请求结果中的那些特别的class和attribute, 并自动关联上相应的js处理事件和效果。

DWZ基于jQuery，可以非常方便的定制特定需求的UI组件， 并以jQuery插件的形式发布出来，如有需要也可做定制化开发。

## 1.6JQuery简介

Jquery是继prototype之后又一个优秀的Javascript框架。它是轻量级的js库 ，它兼容CSS3，还兼容各种浏览器（IE 6.0+, FF 1.5+, Safari 2.0+, Opera 9.0+），jQuery2.0及后续版本将不再支持IE6/7/8浏览器。jQuery使用户能更方便地处理HTML（标准通用标记语言下的一个应用）、events、实现动画效果，并且方便地为网站提供AJAX交互。jQuery还有一个比较大的优势是，它的文档说明很全，而且各种应用也说得很详细，同时还有许多成熟的插件可供选择。jQuery能够使用户的html页面保持代码和html内容分离，也就是说，不用再在html里面插入一堆js来调用命令了，只需定义id即可。

jQuery是一个兼容多浏览器的javascript框架，核心理念是write less,do more(写得更少,做得更多)。jQuery在2006年1月由美国人John Resig在纽约的barcamp发布，吸引了来自世界各地的众多JavaScript高手加入，由Dave Methvin率领团队进行开发。如今，jQuery已经成为最流行的javascript框架，在世界前10000个访问最多的网站中，有超过55%在使用jQuery。

jQuery是免费、开源的，使用MIT许可协议。jQuery的语法设计可以使开发者更加便捷，例如操作文档对象、选择DOM元素、制作动画效果、事件处理、使用Ajax以及其他功能。除此以外，jQuery提供API让开发者编写插件。其模块化的使用方式使开发者可以很轻松的开发出功能强大的静态或动态网页。

## 1.7Jinja2简介

Jinja2是基于python的模板引擎，功能比较类似于于PHP的smarty，J2ee的Freemarker和velocity。 它能完全支持unicode，并具有集成的沙箱执行环境，应用广泛。jinja2使用BSD授权。

Jinja2是Python下一个被广泛应用的模版引擎，他的设计思想来源于Django的模板引擎，并扩展了其语法和一系列强大的功能。其中最显著的一个是增加了沙箱执行功能和可选的自动转义功能，这对大多应用的安全安全性来说是非常重要的。

他基于unicode并能在python2.4之后的版本运行，包括python3。

## 1.8Nginx简介

Nginx ("engine x") 是一个高性能的 HTTP 和 反向代理 服务器，也是一个 IMAP/POP3/SMTP 代理服务器。 Nginx 是由 Igor Sysoev 为俄罗斯访问量第二的 Rambler.ru 站点开发的，第一个公开版本0.1.0发布于2004年10月4日。其将源代码以类BSD许可证的形式发布，因它的稳定性、丰富的功能集、示例配置文件和低系统资源的消耗而闻名。2011年6月1日，nginx 1.0.4发布。

Nginx（发音同 engine x）是一款轻量级的Web 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器，并在一个BSD-like 协议下发行。由俄罗斯的程序设计师Igor Sysoev所开发，供俄国大型的入口网站及搜索引擎Rambler（俄文：Рамблер）使用。其特点是占有内存少，并发能力强，事实上nginx的并发能力确实在同类型的网页服务器中表现较好，中国大陆使用nginx网站用户有：新浪、网易、 腾讯等。

## 1.9Git简介

Git是用于Linux内核开发的版本控制工具。与CVS、Subversion一类的集中式版本控制工具不同，它采用了分布式版本库的作法，不需要服务器端软件，就可以运作版本控制，使得源代码的发布和交流极其方便。Git的速度很快，这对于诸如Linux内核这样的大项目来说自然很重要。Git最为出色的是它的合并追踪（merge tracing）能力。

实际上内核开发团队决定开始开发和使用Git来作为内核开发的版本控制系统的时候，世界上开源社区的反对声音不少，最大的理由是Git太艰涩难懂，从Git的内部工作机制来说，的确是这样。但是随着开发的深入，Git的正常使用都由一些友善的命令稿来执行，使Git变得非常好用。现在，越来越多的著名项目采用Git来管理项目开发，例如：wine、U-boot等。

作为开源自由原教旨主义项目，Git没有对版本库的浏览和修改做任何的权限限制，通过其他工具也可以达到有限的权限控制，比如：gitosis、CodeBeamer MR。原本Git的使用范围只适用于Linux/Unix平台，但在Windows平台下的使用也日渐成熟，这主要归功于Cygwin、msysgit环境，以及TortoiseGit这样易用的GUI工具。Git的源代码中也已经加入了对Cygwin与MinGW编译环境的支持且逐渐完善，为Windows用户带来福音。

# 需求分析阶段

## 2.1 网站建设目的

设计并实现通达的教务精细化管理系统（以下简称系统），该系统为通达学院量身定制，满足以下要求：低培训成本，界面简洁友好实用，使用习惯与已有的教务系统相符；具有完善的人员管理、日程管理、任务管理、工作计划管理和时间设置；具有安全的权限管理；可扩展性高，便于推广。

## 2.2 研究目标与思路

系统架构采用B/S(浏览器/服务器)架构，用户打开网页即可开始操作，简便易行，前端页面布局采用流行的现代化简洁设计，所有功能一目了然，而且参考了已有的创新管理系统、教务处等系统样式，使用户的学习成本降到最低。

完善的人员管理可以通过账号与用户分离的策略来实现，任务只与账号相关，即使用户更换，只是账号和用户的对应关系改变而已，并不影响工作任务的继续进行。

完善的日程管理可通过引入日程表和时间进度条来实现，所有的任务按照月份排列在日历上，时间进度条可将任务体现在一维直线上，时间段可选月、周、日，方便工作人员和主管查看日程的繁简。

完善的任务管理根据用户的权限不同而赋予用户不同的任务功能，干事可以创建、查看自己和其他同事的任务，主管还拥有审核任务、指派任务的权力。任务管理是系统的核心部分，

系统还具有制定、修改工作计划的功能，方便用户管理自己的日程。

系统运行需要提前初始化学期开始的时间，使日程管理、任务管理等系统模块正常启动。

安全的权限管理，给账号设置不同的权限，在登录系统后根据权限不同提供不同的功能，达到安全性要求。

系统各个模块、数据库设计都采用弱耦合关系，方便日后优化、修改。

系统产生的所有信息包括任务信息、人员信息、日程信息都存储在数据库中，便于新人员参考能够快速上手工作，也有利于分析总结学期任务完成情况。

## 2.3 项目研究具体内容

系统主要围绕着任务来运行。

在学期开始之前，用户可能需要制定自己的工作计划，使工作变得有调理，并在学期中各个时段更新自己的工作计划，在学期结束时可以看到自己完成的任务情况，便于下一年的计划制定。

首先是创建任务，填写完整的任务信息，包括任务标题、内容、标签、起始时间、预计完成时间等。这个功能所有账号都可以使用，干事可以为自己创建新任务，主管可以创建新任务指派给某个干事去做。创建任务后任务状态改为待审核。

接下来是审核任务，这个功能需要上级主管来审核任务是否通过，如果不通过则把信息返回给任务创建者还可附上批示，修改过后可再提交审核，如果通过任务状态变为已通过审核，表明该任务已经开始，任务可能被指派给任务创建者也可能指派给其它干事。

在任务执行过程中，可能会需要修改任务，不论是任务接受者还是主管修改了任务，都需要让双方都了解任务信息的改变，体现在系统中就是任务状态的改变，当任务状态改变时该任务自动排列到顶端提醒用户查看。

任务执行到一定阶段可能会上传一些文件，系统需要具备文件管理功能，且权限控制只能删除修改自己的文件而不是所有文件，任务一旦完成普通账户将不能修改或删除任务文件。考虑到一个任务可能需要多个账号共同完成，所以文件管理允许下载其它账号的任务文件。

任务开始后，将按照任务的审核通过时间和预计结束时间或者必须完成时间纳入日程管理，在时间表上直观的显现出来，按照距离任务截止时间的长短来排列任务，提醒用户完成最紧要的任务。

当任务完成时，任务接受者提交完成的材料，填写任务总结，待主管审核，如果不通过则可以批示继续完成任务，如果通过主管可写下评价供任务接受者总结经验教训。任务完成时任务状态变为已完成，保存所有的任务信息和任务文件，供日后参考。

对于主管来说，需要看到各个下属一定时间段内任务的繁重情况，来调节工作量，这需要系统提供良好的搜索功能，能够通过任务搜索任务接受者，通过人员搜索该人员一定时间段内的工作量，通过时间搜索所有任务的执行情况，为监督、激励、提醒工作人员提供方便的信息获取途径。

## 2.4项目拟解决的关键问题及项目的重点、难点、创新点

项目的重点是系统的在教务管理中的可用性、易用性和定制性。

项目的难点是如何满足系统各项详细的功能需求，使系统无严重漏洞的应用到实践中。

项目的创新点有运用了成熟的系统设计思想，从数据库设计到后端功能代码设计，再到前端的布局，都采用弱耦合关系，使得可扩展性非常高，便于修改后应用到其它学院或学校管理系统中，并且都先设计出模型再应用到实践中，系统是经过科学的分析再实践出来的。

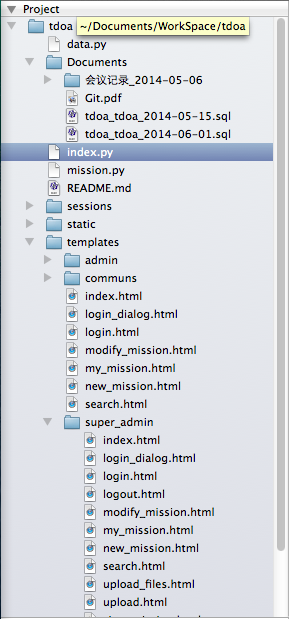
# 总体设计

## 3.1 系统功能模块划分及系统结构图

该网站系统分为：首页、登录注销、任务管理、部门（科室）管理、用户管理、账户管理、附件管理等模块。

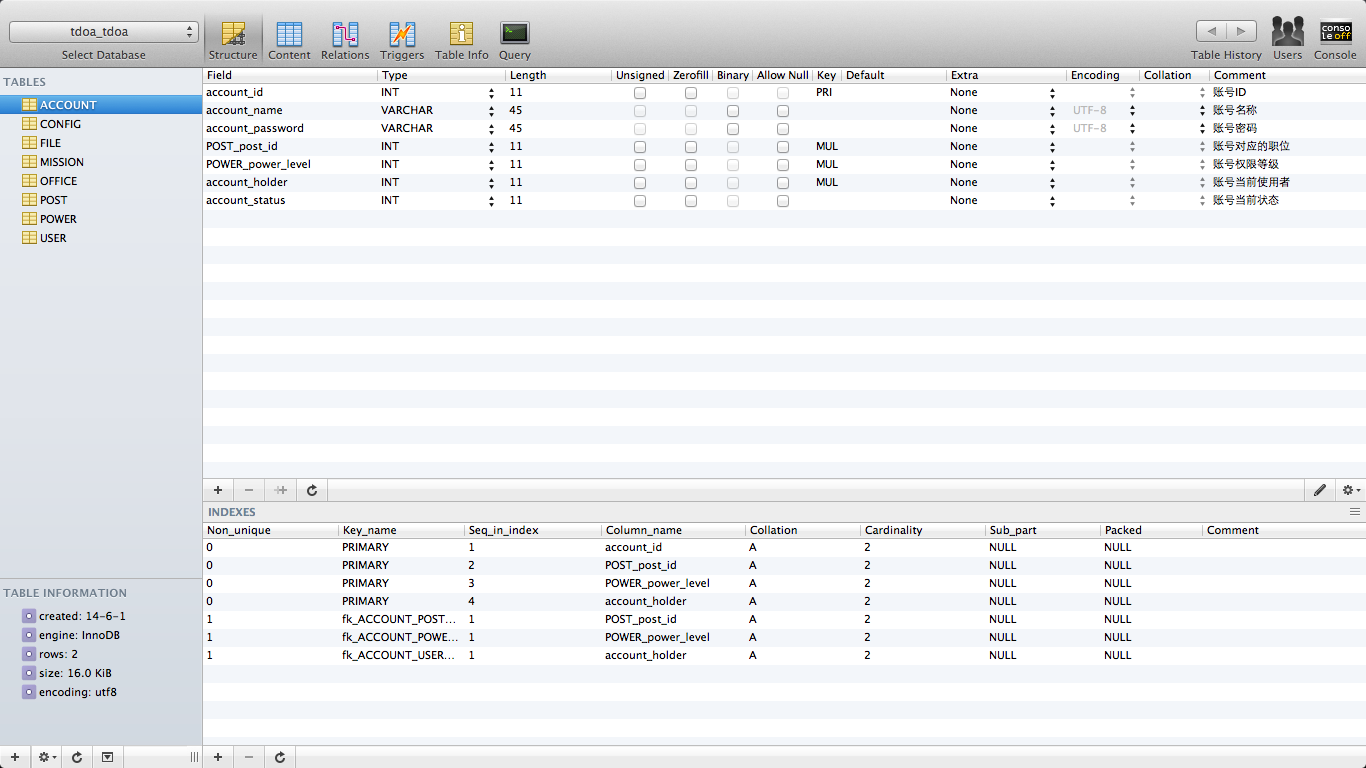
## 3.2 数据流程分析

## 3.3 文件结构

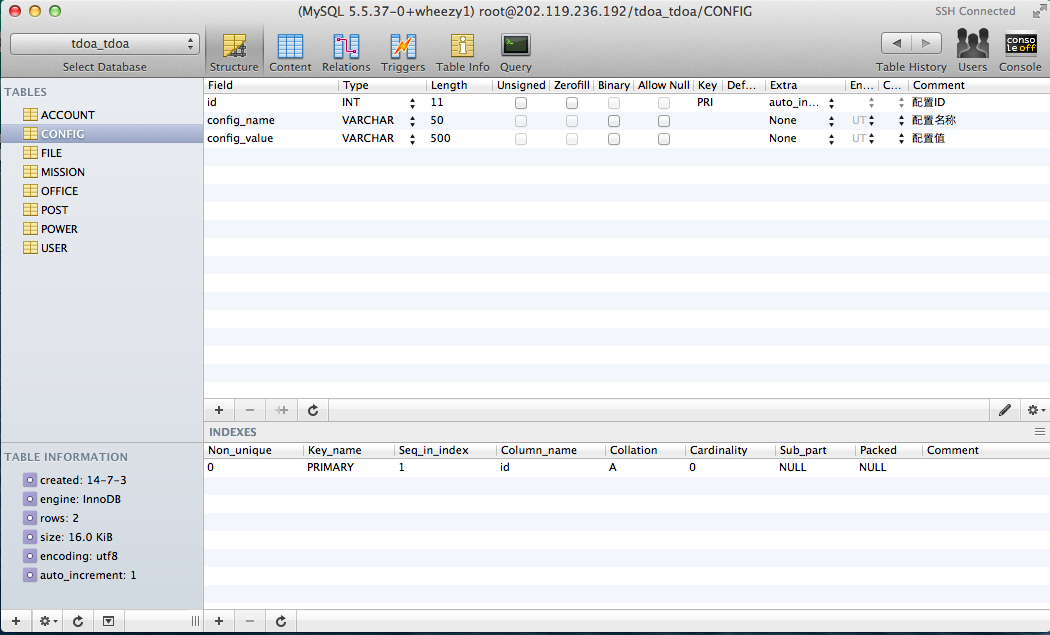


## 3.4 各模块的数据结构设计

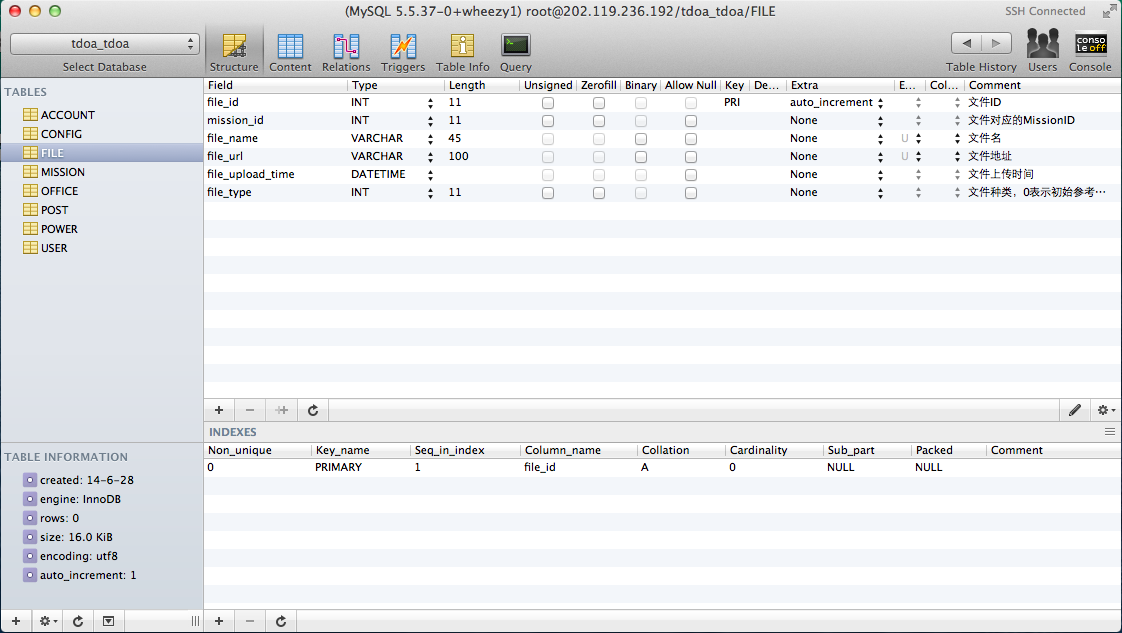
1.账户表



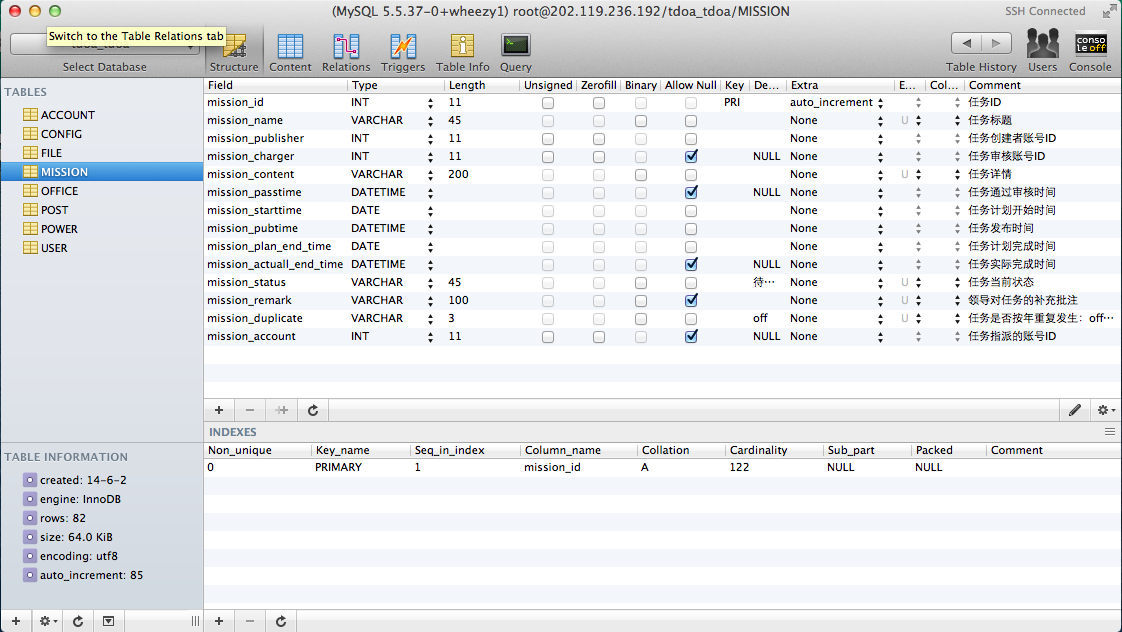
2.配置表



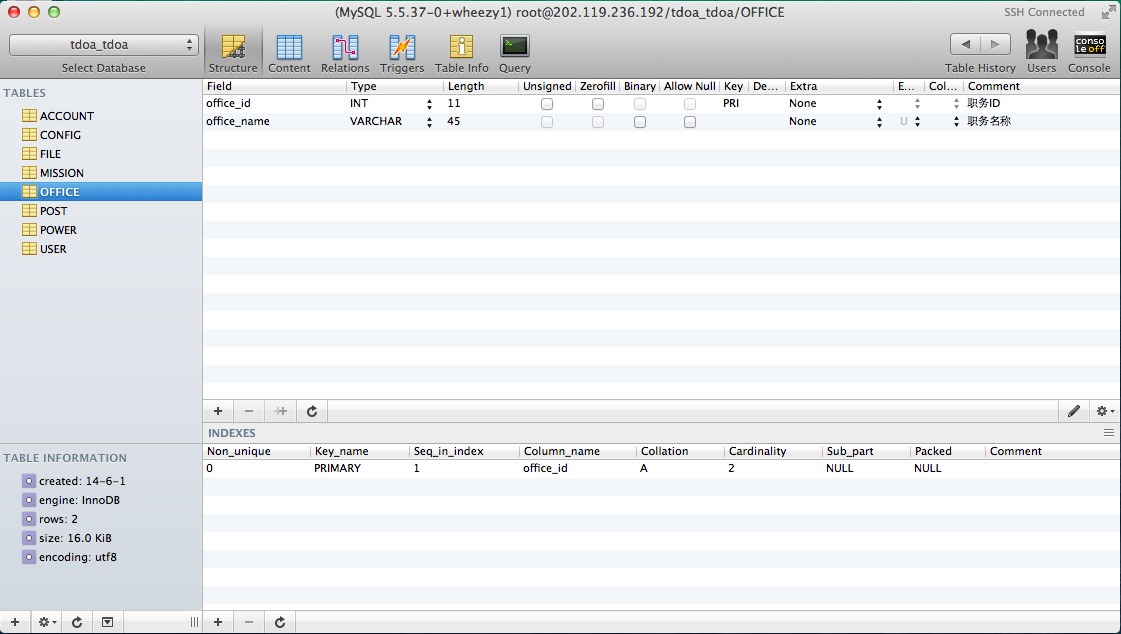
3.附件表



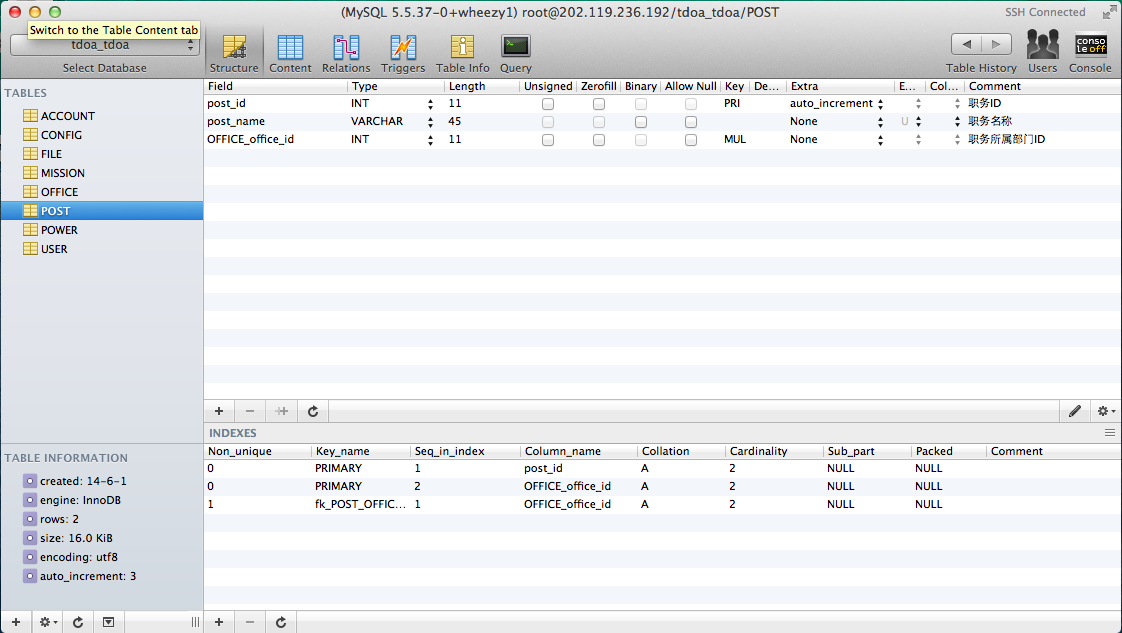
4.任务表



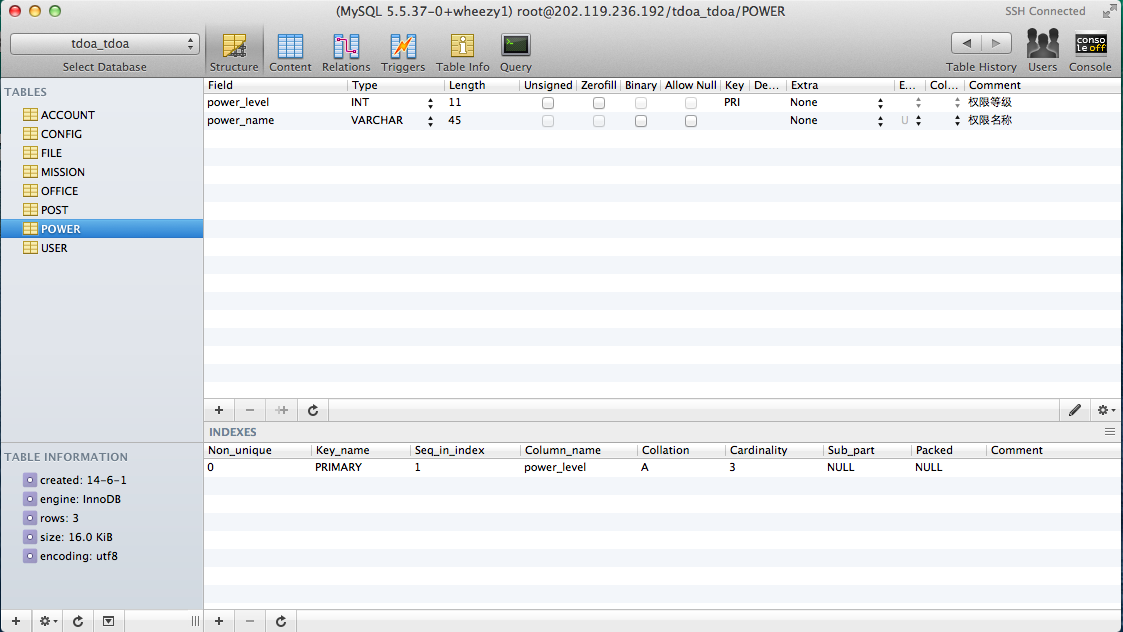
5.部门表



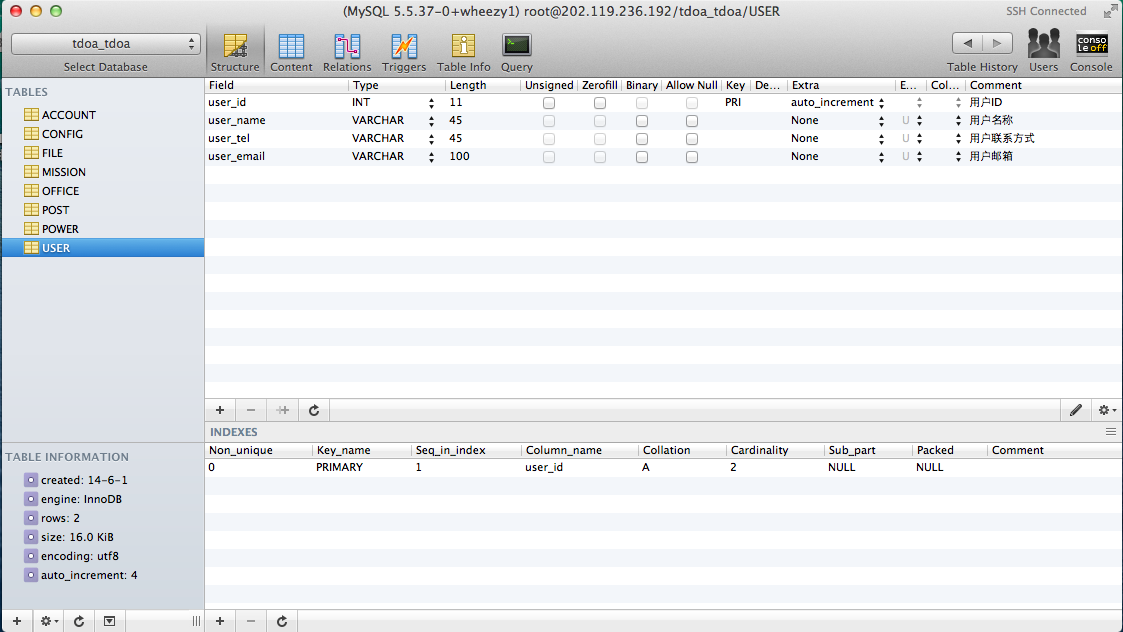
6.职务表



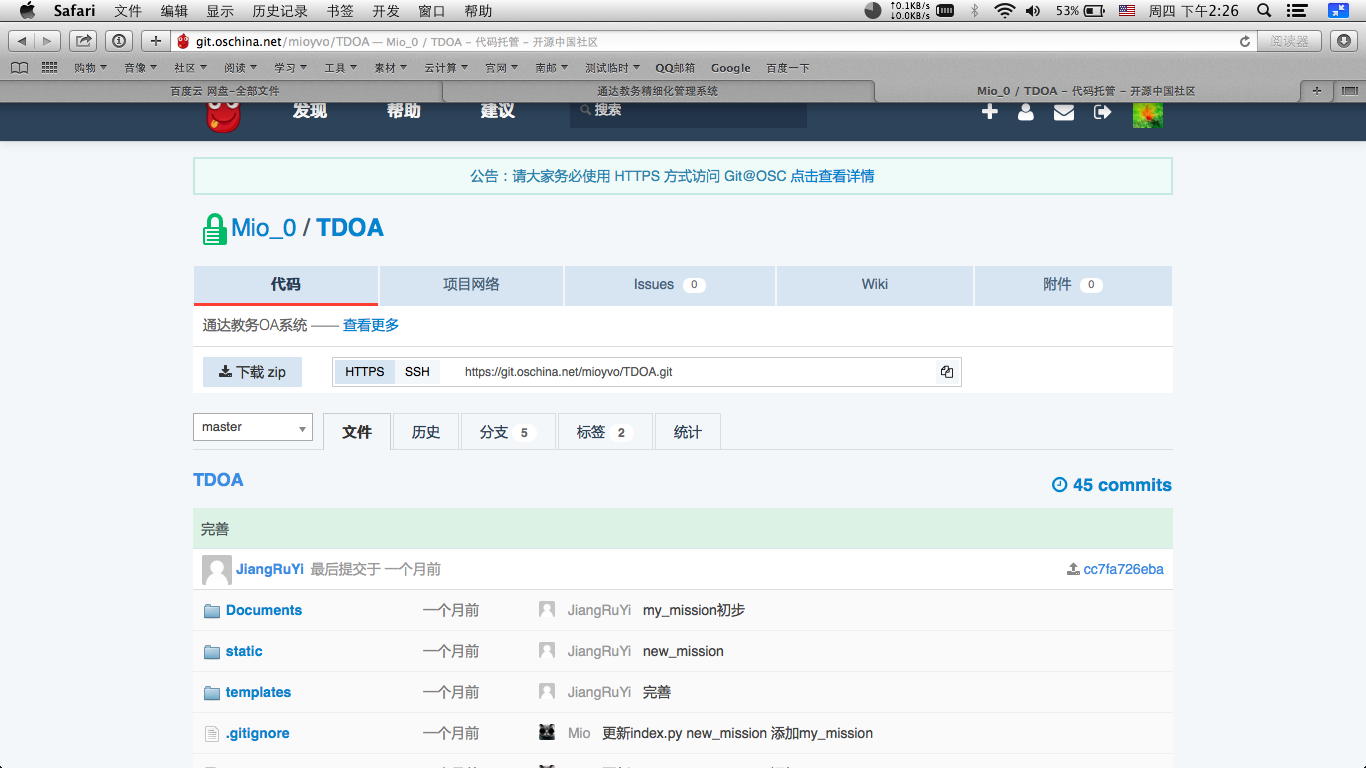
7.权限表



8.用户表



## 3.5 Git版本控制



# 详细设计

## 4.1 登录注销模块

### 4.1.1 流程图

### 4.1.2 登录界面



说明：通过此界面，可以实现用户登录。

### 4.1.3 代码

class login(object):

"""登陆"""

def GET(self):

if session.login==1:

#render = creat\_render(level=session.level)

raise web.seeother('/index')

#return render\_template(level=session.level,template\_name='index.html',user=session.user)

else:

#render = creat\_render(level=0)

return render\_template(level=0,template\_name='login.html',error="")

def POST(self):

user = web.input().user

#md5加密存储密码

passwd = data.md5(web.input().passwd)

ident = data.checkin(user)

try:

if len(ident)==0 :

session.login = 0

#render = creat\_render(level=0)

return render\_template(level=0,template\_name='login.html',error="用户名不存在")

#return render.login(error="用户名不存在")

elif passwd == ident[0]['account\_password']:

session.login = 1

session.user = user

session.level = ident[0]['POWER\_power\_level']

#render = creat\_render(session.level)

return render\_template(level=session.level,template\_name='index.html',user=session.user)

#return render.index(session.user)

else:

session.login = 0

#render = creat\_render(level=0)

#return render.login(error="密码错误")

return render\_template(level=0,template\_name='login.html',error="密码错误")

except :

session.login = 0

#render = creat\_render(level=0)

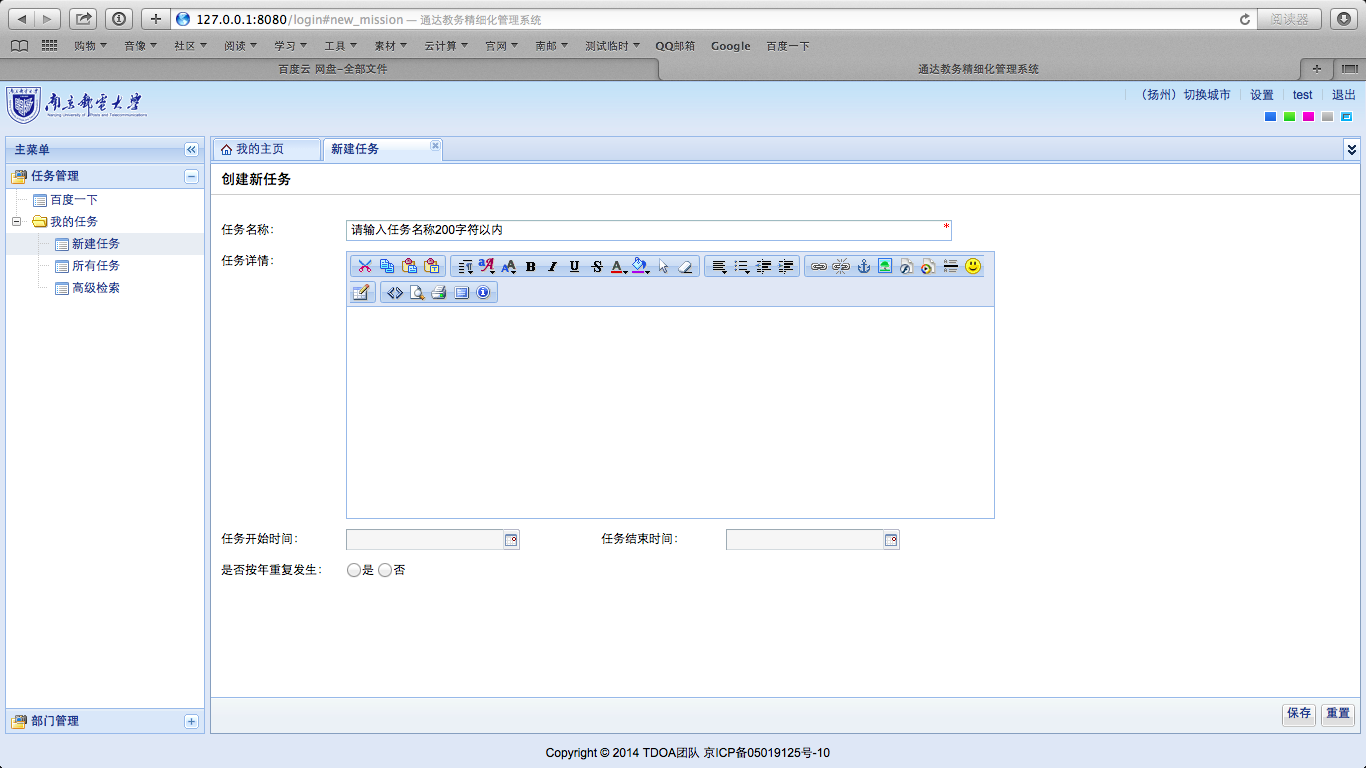
#return render.login(error="系统错误")

return render\_template(level=0,template\_name='login.html',error="系统错误")

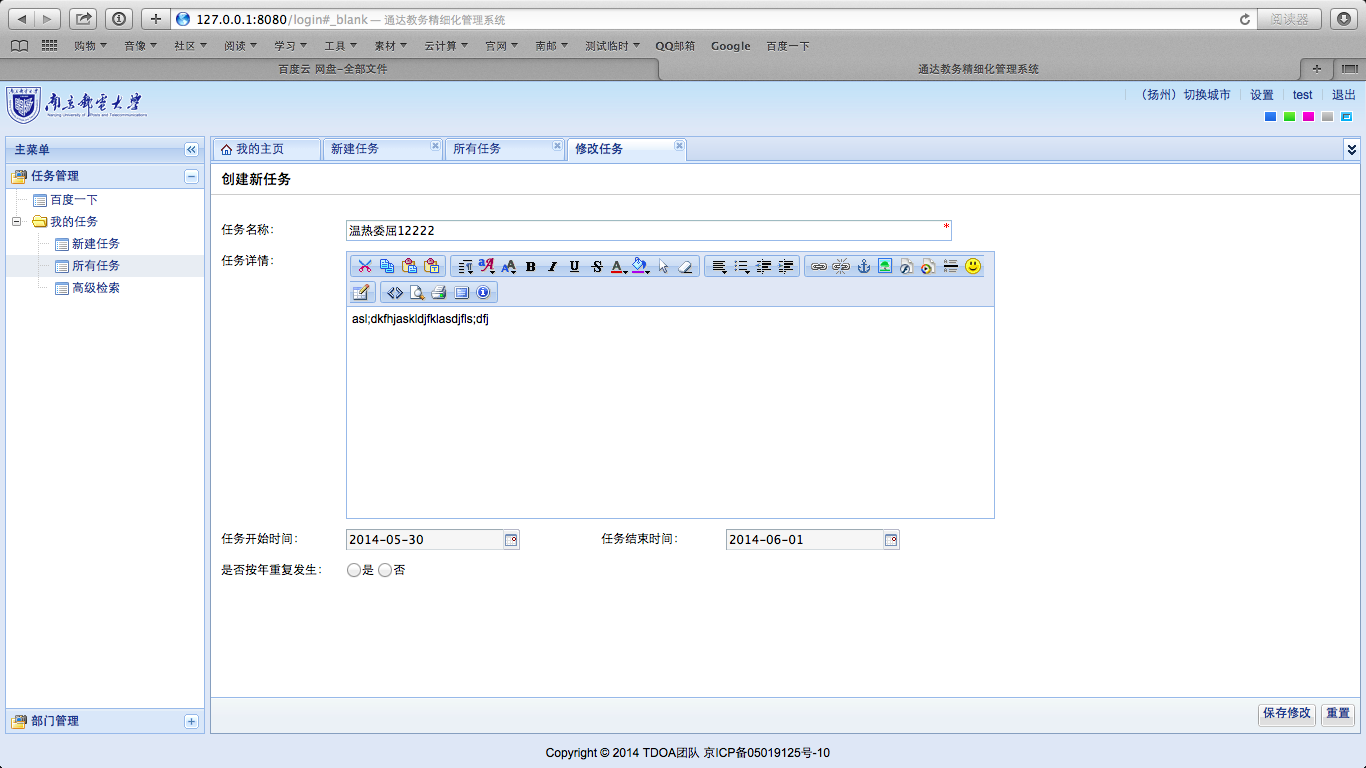
## 4.2任务管理模块

### 4.1.1 流程图

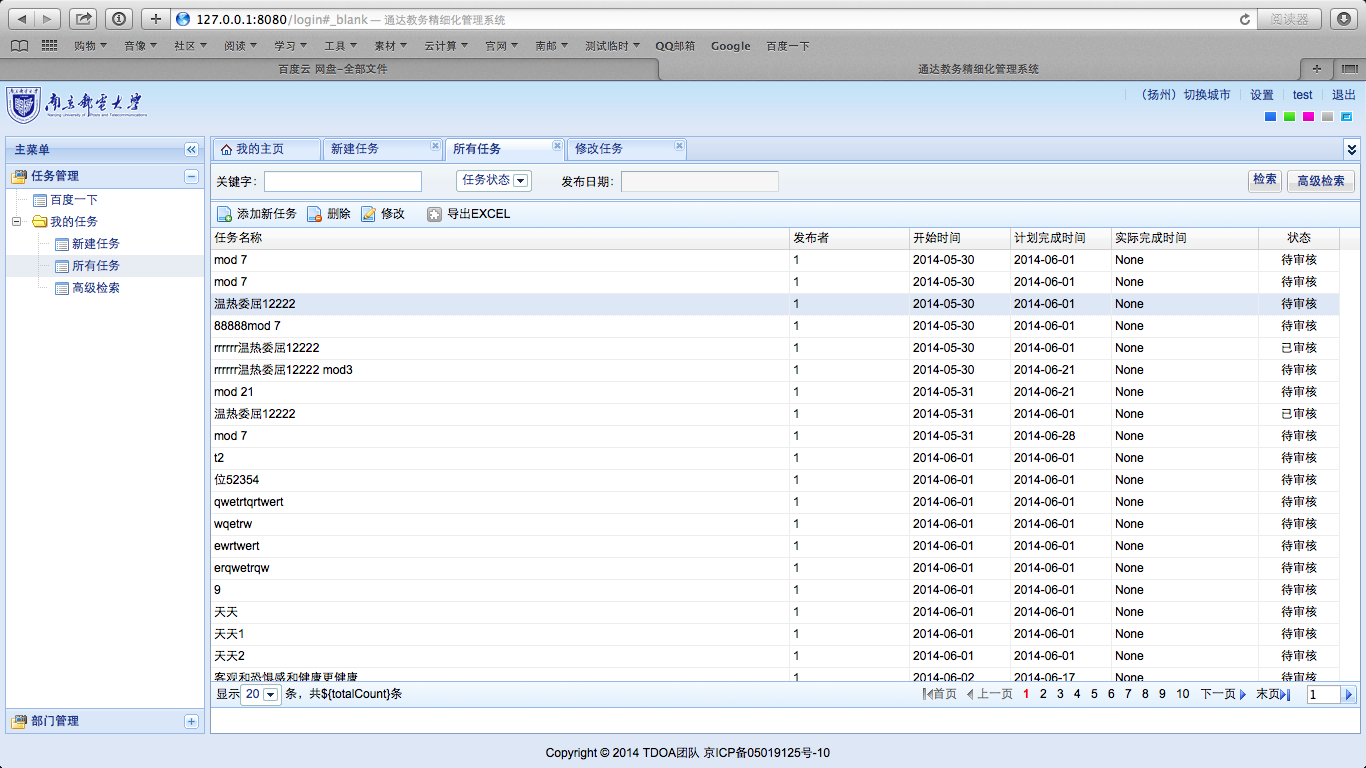
### 4.2.2界面



说明：新建任务界面



说明：修改任务界面



说明：查看所有任务界面



说明：查看某一任务界面

### 4.1.3 代码

class my\_mission(object):

"""任务列表"""

def GET(self):

if session.login == 1:

if session.user:

#找到当前账户所有任务

mission\_list = mission.mission\_list(account\_name=session.user, role='mission\_publisher')

#render = creat\_render(level=session.level)

#return render.my\_mission(session.user, mission\_list)

return render\_template(level=session.level, template\_name='my\_mission.html',

user=session.user, mission\_list=mission\_list)

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

class new\_mission(object):

"""创建任务"""

def GET(self):

if session.login==1:

if session.user:

#render = creat\_render(level=session.level)

#return render.new\_mission(session.user,"","","","","")

return render\_template(level=session.level, template\_name='new\_mission.html',

user=session.user,mission\_name="",mission\_content="",mission\_starttime="",

mission\_plan\_end\_time="",error="")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

def POST(self):

#登陆控制

if session.login == 1:

if session.user:

mission\_name = web.input().mission\_name

mission\_content = web.input().mission\_content

mission\_starttime = web.input().mission\_starttime

mission\_plan\_end\_time = web.input().mission\_plan\_end\_time

try:

mission\_duplicate = web.input().mission\_duplicate

except:

mission\_duplicate = "off"

#检查任务信息是否合法

result = mission.mission\_check(mission\_name, mission\_content, mission\_starttime, mission\_plan\_end\_time)

ajax\_result = {"statusCode":"300", "message":result}

web.header('Content-Type', 'application/json')

#如果任务合法，将任务信息存储进MISSION表

if result == "no error":

mission.mission\_save(mission\_name, session.user, mission\_content, mission\_starttime, mission\_plan\_end\_time, mission\_duplicate)

mission\_list = mission.mission\_list(account\_name=session.user, role='mission\_publisher')

ajax\_result = {"statusCode":"200", "message":"任务新添加成功", "callbackType":"closeCurrent"}

return json.dumps(ajax\_result)

#return render\_template(level=session.level, template\_name='my\_mission.html',

#user=session.user, mission\_list=mission\_list)

#如果任务不合法，把已填写的表单数据返回给new\_mission页面

"""

else:

return render\_template(

level=session.level, template\_name='new\_mission.html',

user=session.user,

mission\_name=mission\_name,

mission\_content=mission\_content,

mission\_starttime=mission\_starttime,

mission\_plan\_end\_time=mission\_plan\_end\_time,

error=result

)

"""

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

class delete\_mission(object):

"""删除任务"""

def POST(self):

mission\_id = web.input().mission\_id

if session.login==1:

if session.user:

result = mission.mission\_delete(mission\_id)

mission\_list = mission.mission\_list(account\_name=session.user, role='mission\_publisher')

return render\_template(level=session.level, template\_name='my\_mission.html',

user=session.user, mission\_list=mission\_list)

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

class modify\_mission(object):

"""修改任务 ,modify\_mission.html跟new\_mission.html相同,

区别是post提交到 apply\_modify\_mission, 而且post信息中需要添加mission\_id"""

def GET(self,arg):

if session.login == 1:

if session.user:

arg = web.input()

m = mission.mission\_view(arg.mission\_id)

#render = creat\_render(level=session.level)

#return render.modify\_mission(session.user,m,"")

return render\_template(

level=session.level,

template\_name='modify\_mission.html',

user=session.user,

mission\_view=m,

error="")

#返回m，m[0]['mission\_name'], m[0]['mission\_content']等等

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

def POST(self):

"""接收mission\_id,"""

if session.login == 1:

if session.user:

try:

m = mission.mission\_view(web.input().mission\_id)

return render\_template(

level=session.level,

template\_name='modify\_mission.html',

user=session.user,

mission\_view=m,

error="修改任务")

except Exception, e:

#render = creat\_render(level=session.level)

#return render.modify\_mission(session.user,)

print e

return render\_template(

level=session.level,

template\_name='modify\_mission.html',

user=session.user,

mission\_view=m,

error="出现未知错误，请重试"

)

#通过mission\_id获取任务信息，作为表单的默认值返回给modify\_mission.html

#最后多了一个参数mission\_id

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

class apply\_modify\_mission(object):

"""应用任务修改, 成功返回my\_mission.html，失败返回modify\_mission.html继续修改"""

def POST(self):

#需要接收mission\_id

if session.login == 1:

if session.user:

mission\_id = web.input().mission\_id

#先添加任务，再把原来的任务删除

#创建任务

try:

mission\_duplicate = web.input().mission\_duplicate

except Exception, e:

mission\_duplicate = "off"

#检查输入信息

result = mission.mission\_check(web.input().mission\_name, \

web.input().mission\_content, web.input().mission\_starttime, web.input().mission\_plan\_end\_time)

if result == "no error":

#存储任务信息

mission.mission\_save(web.input().mission\_name, session.user, \

web.input().mission\_content, web.input().mission\_starttime, \

web.input().mission\_plan\_end\_time, mission\_duplicate)

mission\_list = mission.mission\_list(account\_name=session.user, role='mission\_publisher')

#return render.my\_mission(session.user, mission\_list)

ajax\_result = {"statusCode":"200", "message":"任修改成功", "callbackType":"closeCurrent"}

try:

result\_del = mission.mission\_delete(mission\_id)

except Exception, e:

print e

finally:

mission\_list = mission.mission\_list(account\_name=session.user, role='mission\_publisher')

return json.dumps(ajax\_result)

else:

m = mission.mission\_structure(mission\_name=web.input().mission\_name, \

mission\_content=web.input().mission\_content, \

mission\_starttime=web.input().mission\_starttime, \

mission\_plan\_end\_time=web.input().mission\_plan\_end\_time, \

mission\_id=mission\_id)

#render = creat\_render(level=session.level)

#return render.modify\_mission(session.user, m, result)

return render\_template(

level=session.level,

template\_name='modify\_mission.html',

user=session.user,

mission\_view=m,

error=result

)

#删除任务

"""

try:

result\_del = mission.mission\_delete(mission\_id)

except Exception, e:

print e

finally:

mission\_list = mission.mission\_list(account\_name=session.user, role='mission\_publisher')

#render = creat\_render(level=session.level)

#return render.my\_mission(session.user, mission\_list)

return render\_template(level=session.level, template\_name='my\_mission.html',

user=session.user, mission\_list=mission\_list)

"""

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

class view\_mission(object):

"""任务详情

链接放在my\_mission对应的任务名中, mission\_id直接跟在/view\_mission

/view\_mission/(.\*)

"""

def GET(self,arg):

if session.login == 1:

if session.user:

arg = web.input()

m = mission.mission\_view(arg.mission\_id)

#render = creat\_render(level=session.level)

#return render.view\_mission(session.user,m)

return render\_template(

level=session.level,template\_name='view\_mission.html',

user=session.user,

mission\_view=m)

#返回m，m[0]['mission\_name'], m[0]['mission\_content']等等

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

else:

return json.dumps({"statusCode":"301", "message":"会话超时，请重新登录"})

#return render\_template(level=0,template\_name='login.html',error="请重新登录")

# 总结

转眼之间，近两周的课程设计就要结束了，这是大学之中最重要的一个课程设计。专业实习的课程设计是考验大学者几年来的所学，它要求我将大学这几年来所学到的知识能够融会贯通、熟练应用，并要求我能够理论联系实际，培养我的综合运用能力以及解决实际问题的能力。

在这期间，我不断学习、不断积累并且不断提高。Pyton是一门非常健壮的解释性语言，在国内应用确实不多，在使用的过程中遇到无数问题，在指导老师们的悉心指导和同学们的帮助下，我从最初的选定课题内容开始做起，进行设计方案的确定，以及之后的网站设计。这次的课程设计，是对我们这几年来所学的专业知识是否踏实的检验，让我对这几年中所学的知识进行了综合，也温习了一些已经快要忘掉的知识，并且还学到了一些实际操作经验。与此同时，我也充分认识到了自身的许多不足：基础知识学的不够扎实，缺乏综合运用及理论联系实际的能力等。最后，感谢我们的指导老师。

在此，我们衷心感谢我们的指导老师，在他们的精心指导和帮助下，我的课程设计才得以顺利完成，也感谢在网站设计过程中帮助过我们的同学们，在知识和合作方面，都得到了很大的提高！

# 参考

[1] DWZ 富客户端框架使用手册

[2] Git Community Book 中文版