智慧教学管理软件系统说明

目录

一、设计要求...............................................................................................................3 二、实施方案..............................................................................................................3 三、具体设计..............................................................................................................4

3.1.功能模块图.......................................................................................................4

3.2 各个模块详细的功能描述...............................................................................5 四、所用算法及函数..................................................................................................7

2.2.1 基本功能算法................................................................................................7 2.2.2 考勤管理的实现..........................................................................................11

2.2.2 考勤管理的实现..........................................................................................13

2.2.2 邮件发送......................................................................................................13

附录：源程序..................................................................................................................

## 一、设计要求

根据教与学活动过程中所涉及的教师任课、学生学习情况，设计并实现基于教与学的智慧教学管理软件系统。

要求：

1. 掌握数据结构的输入/输出；
2. 设计并实现涉及教与学活动过程中的相关多功能算法；其中，包括但不限于如下主要功能：
3. 教师授课、作业批改，答疑/辅导、学生考勤记录、平时成绩管理等；学生各科成绩及其有无不良记录、获奖、竞赛、学术交流情况等；
4. 经手机或邮件等不同通信方式、手段，可将学校/院相关教学活动及时通知与告知教师与学生；

（3）根据教师、学生成绩信息，模糊查找与精确查找，并显示所查找条目的相关内容；（4）教师、学生信息的有序排列（如按课程排序、教师授课效果评价排序、学习绩点排序、不良记录严重程度排序等）；

（5）同一门课程的任课教师不少于 3 人，学生人数不少于 70。

## 二、实施方案

本项目采用 Python+Html 框架设计，使用 Amaze UI 模板进行静态网页的设计，采用 Python 当中的 Flask 框架构建项目主体，连接前端与后端进行数据传输。Flask 是一个基于 Python 开发且依赖 jinja2 模板和 Werkzeug WSGI 服务的一个微型框架。因为 Flask 上手比较方便且操作简单不需要将整个 web 应用放入 Python 文件，所以我们决定采用这个框架进行项目设计。

项目通过 Flask 中的@app.route(‘/’)将前端与后端相连接，实现在静态网页中的操作可以通过后端算法展现在前端。比如@app.route(‘/login’,methods=[‘POST’])是连接到网页端的登录界面，通过 def login(): 中所写的函数进行登录操作，判断用户输入的不同信息返回不同的静态网页，实现登录：

@app.route('/login', methods=['POST']) def login():

if request.form['identity'] == "admin":

ifrequest.form['username']=="admin"andrequest.form['password'] == "admin": return render\_template('admin\_index.html')

if request.form['identity'] == "student":

if request.form['username'] == "2" and request.form['password'] == "222": stwid(request.form['username']) return render\_template('student\_index.html')

if request.form['identity'] == "student":

if request.form['username'] == "3" and request.form['password'] == "333":

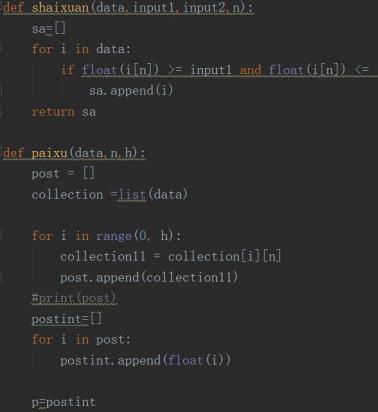
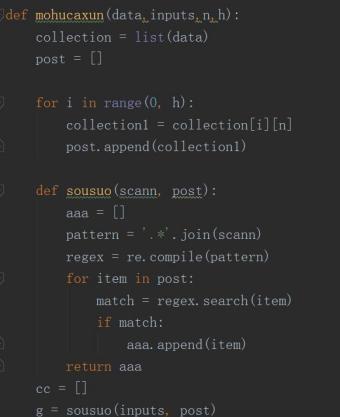
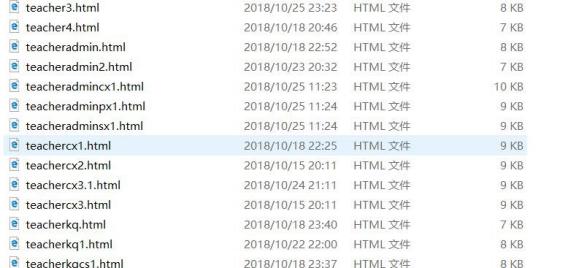
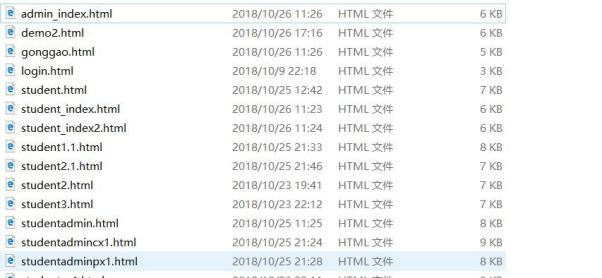
stwid(request.form['username']) return render\_template('student\_index2.html')

if request.form['identity'] == "teacher":

if request.form['username'] == "111" and request.form['password'] == "111":

return render\_template('teacher\_index.html')

项目目前总共设计了大约 60html 个界面，30 个相关函数，部分展示如下：



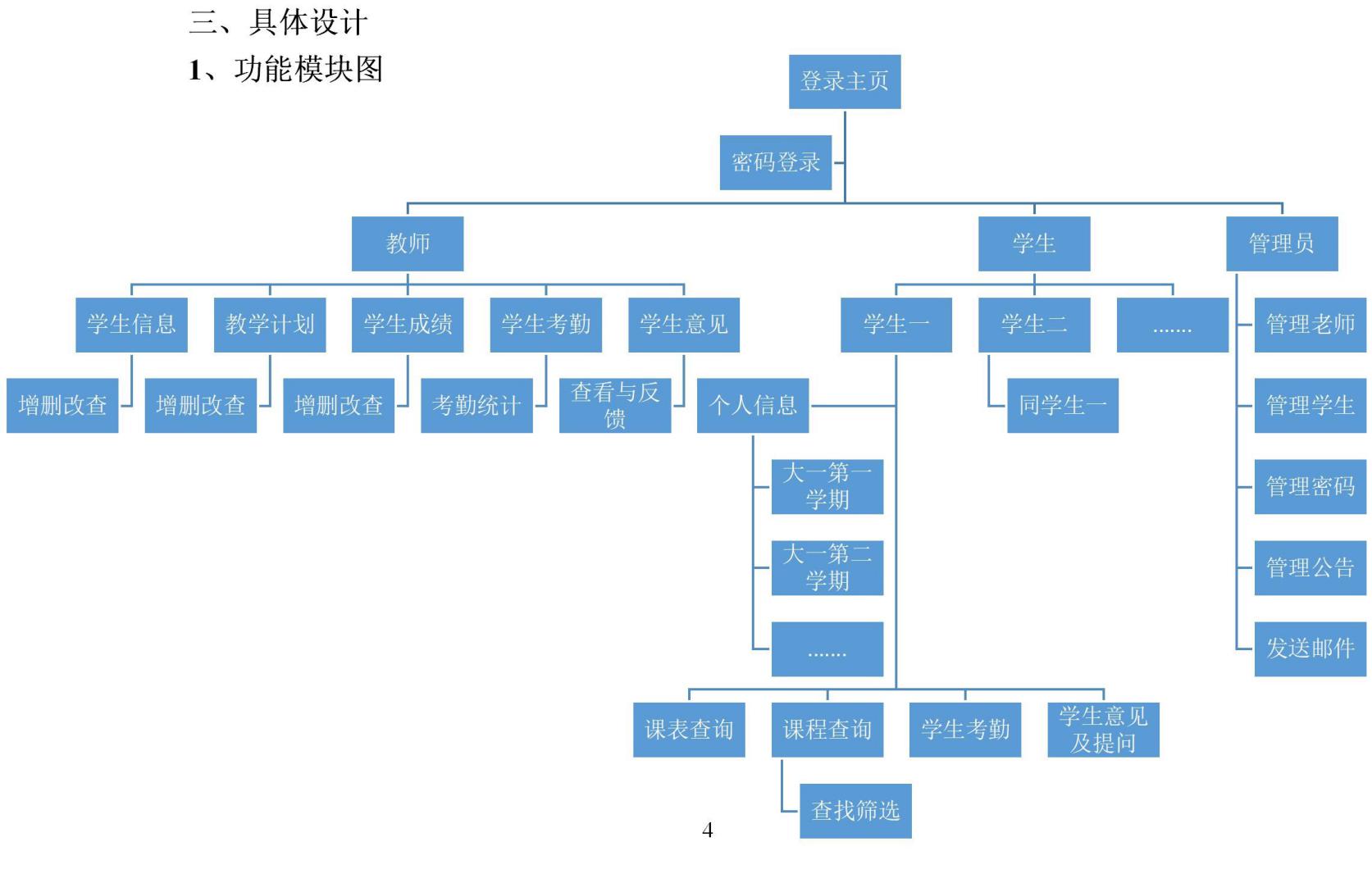
通过 app.route()函数进行前后端连接，将存放代码中的数据显示到界面当中，并进行操

作实现功能。

三、具体设计

**1**

、功能模块图



1. 功能模块详细介绍本项目总共总共有三个登录端：1、教师，2、学生，3、管理员。



* 1. 教师管理系统：

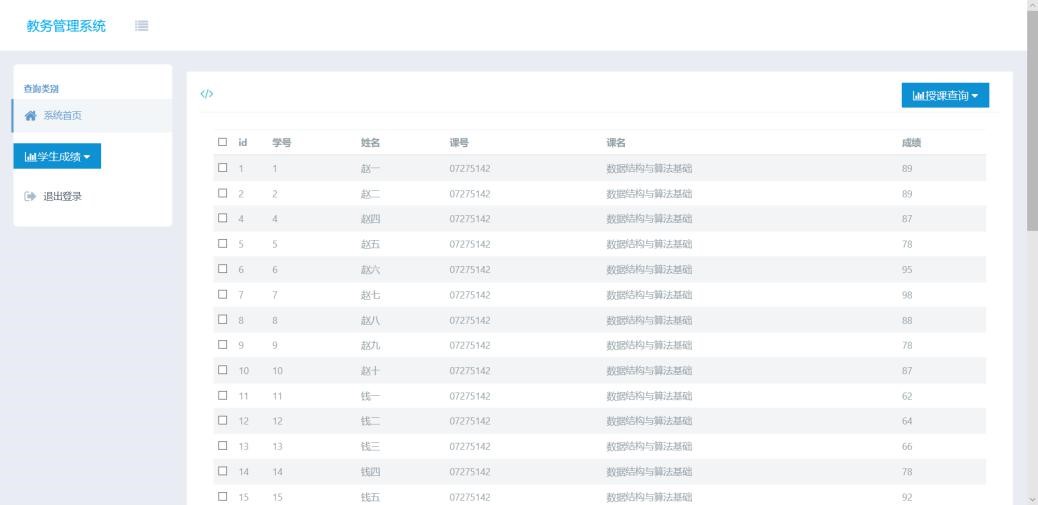
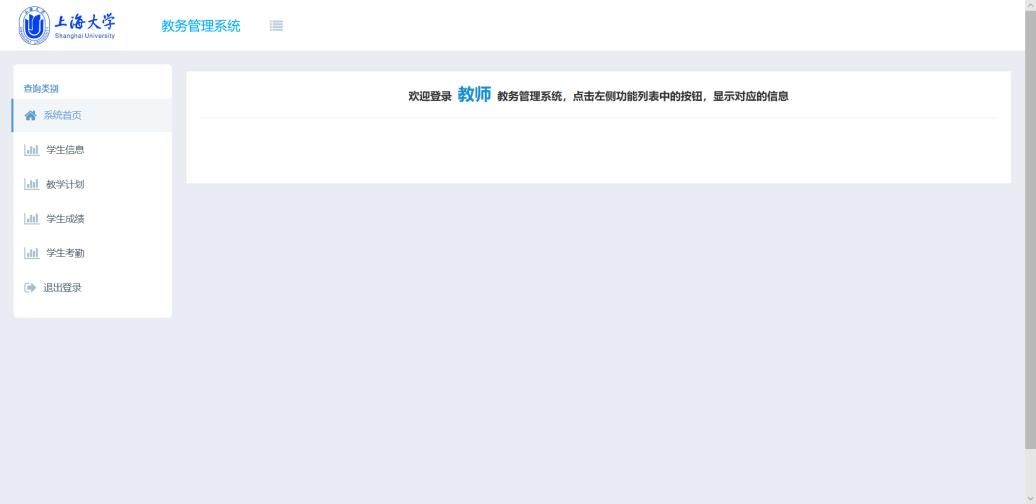
·管理自己班上学生的平时成绩；

·根据平时成绩进行排序；

·精确或模糊查找学生学生信息和课程；

·管理班级学生考勤情况：发送验证码，学生登录系统输入验证码和学号进行签到；

·对学生提出的意见和问题进行反馈；



* 1. 学生管理系统：

我们设计了两个学生登录，进入后显示不同的信息

·个人不同学期的成绩查询

·本学期课表查询

·课程查询：可以进行模糊查询，精确查询，按条件筛选

·学生签到

·提意见，提问题



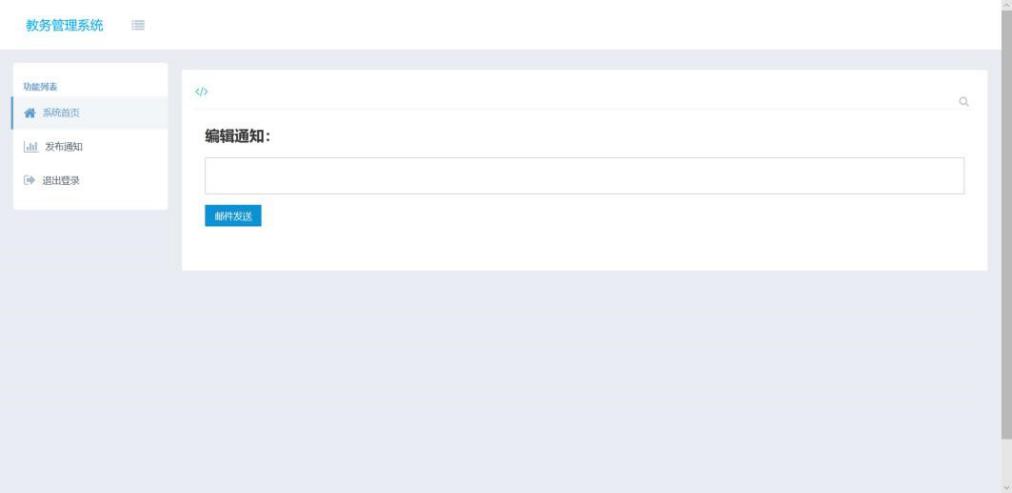
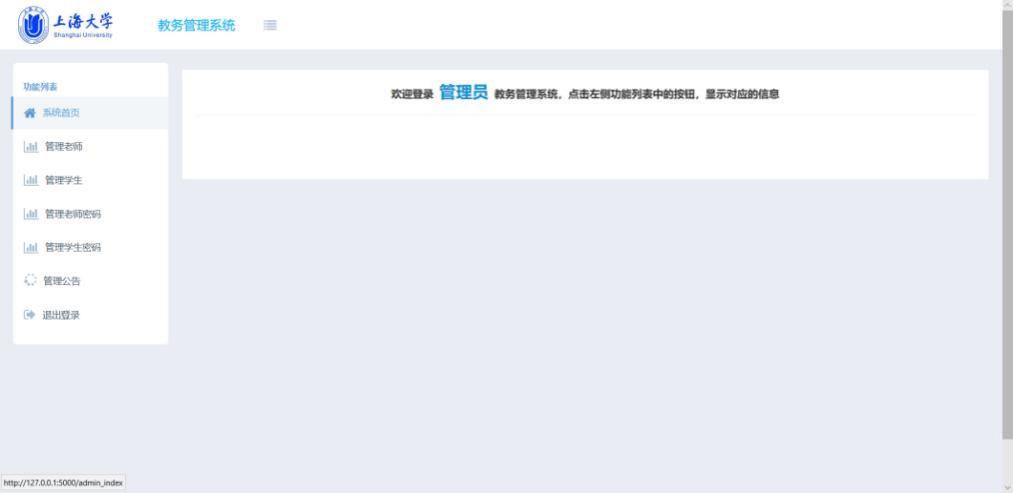
* 1. 管理员系统：

·对老师信息的增删改查

·对学生信息的增删改查

·对密码的管理

·学校通知编写及邮件发送



## 四、所用算法及函数

**1**、基本功能的算法

（1）数据存储：在代码中用列表存储数据，然后将操作后的列表使用.json 文件或.txt 文件存储起来，比如在进行修改、删除等操作后，用文件存储起来，下次再对文件中的内容进行操作。以下代码：将列表 collection 写入名为 kqlv.json 的文件。

def stwkql(collection) file = open('kqlv.json', 'w') for i in collection:

json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

下面为读取文件名为 kqlv.json 的内容代码： def strkql():

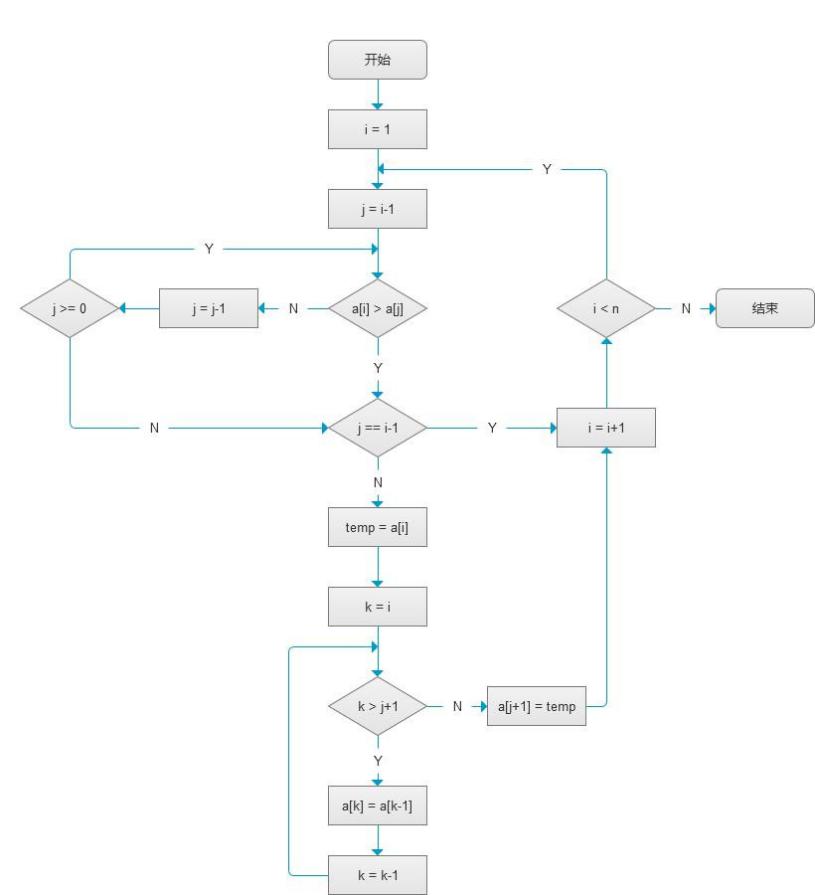
risk\_result = [] with open('kqlv.json', 'r') as f:

collection = f.read().split('\n')[:-1] for x in collection:

json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result

（2）排序插入排序流程图：



代码实现： for index in range(1, len(postint)): current\_val = postint[index] position = index while position > 0 and postint[ position - 1] > current\_val:

postint[position] = postint[position - 1] position -= 1

postint[position] = current\_val

ind=[] for i in range(0,len(postint)):

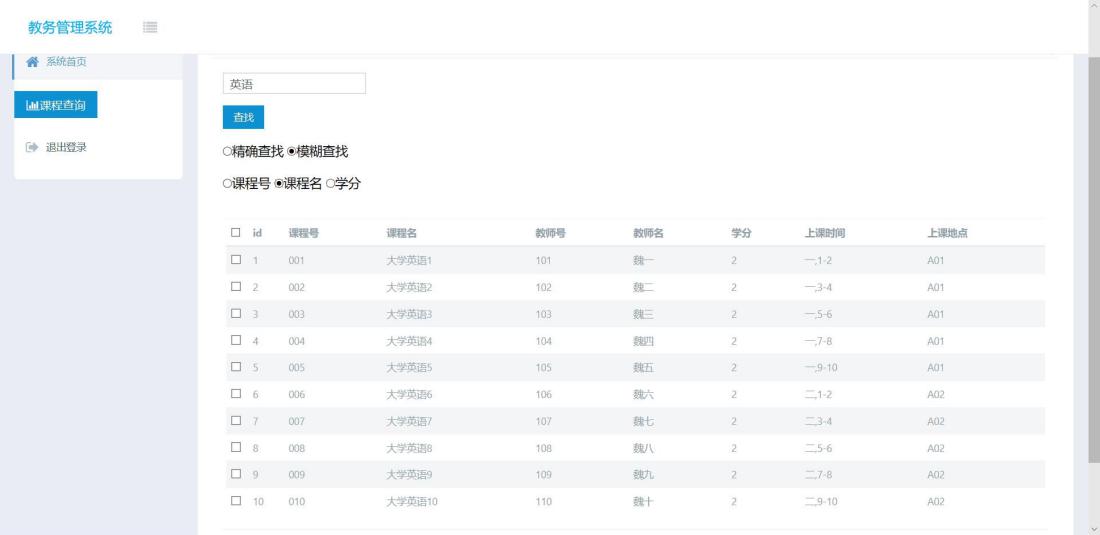
for j in range(0,len(p)): if postint[i] == p[j]:

ind.append(j)

od=[] for i in ind:

od.append(collection[i])

（3）查找：模糊查找：使用正则匹配，正则匹配依赖于正则表达式，正则表达式由一些普通字符和一些元字符构成，本项目中使用的元字符为.\* ，作用为匹配前面的子表达式任意次，例如 xk 能匹配 x，也能匹配 xk，xkk…… 在此次项目中，当在文本框输入查找目标后，选择查找类型（模糊查找或精确查找），再选择目标类型点击确定，就会遍历列表中与目标匹配的表达式任意次，并显示出来，如图所示



精确查找：在模糊查找的前提下，将模糊查找的结果与输入相对比，输出与模糊查找结果相同的数据。如图所示



代码实现：（核心代码） def sousuo(scann, post):

aaa = []

pattern = '.\*'.join(scann)

regex = re.compile(pattern) for item in post:

match = regex.search(item) if match:

aaa.append(item) return aaa

（4）增加、删除：增加即在已有的列表后增加空的元素，再对空元素进行赋值

删除即删除想要删除行的一行元素。如下 data 代表要操作的列表，h 代表第几行。

def deletehang(data,h):

d=[]

if h==len(data):

for i in range(0,len(data)-1):

d.append(data[i])

else:

for i in range(h,len(data)-1): data[i]=data[i+1]

for i in range(0, len(data) - 1):

d.append(data[i])

return d

1. 修改：data 为要操作列表，h 为行，n 为列，inputs 为修改内容，即要将列表 data 中的为 h 行 n 列的元素修改为 inputs



如下修改第 0 行第 2 列的数据改为 2015 年



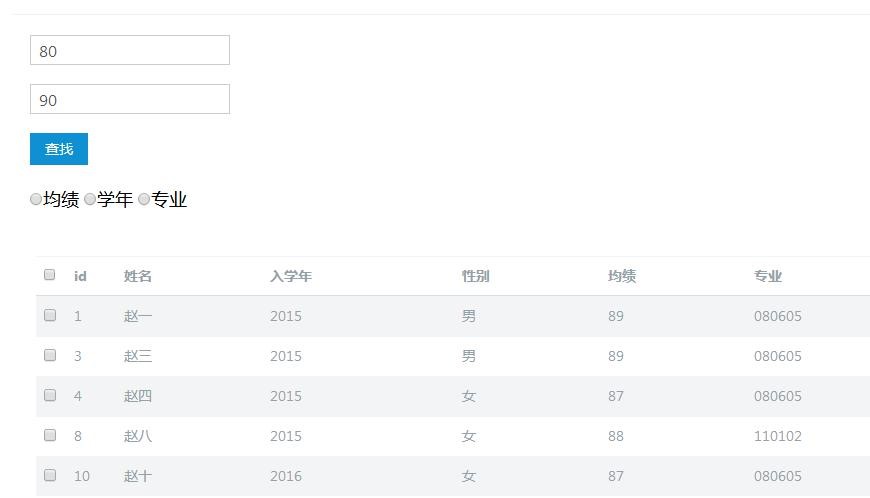
def xiugai(data,h,n,inputs):

collection=[] for i in data:

collection.append(list(i))

collection[h][n]=inputs return collection

1. 筛选：主要针对于将成绩居于一个区间的内容输出，比如输出成绩为 80—90 分的学生信息：



相应代码： def shaixuan(data,input1,input2,n):

sa=[] for i in data:

if float(i[n]) >= input1 and float(i[n]) <= input2:

sa.append(i)

return sa

data 为操作列表，input1 和 input2 为上下区间，n 为筛选第几列的数据

### **2**、考勤管理的实现

由于我们无法做到网页之间的相互传值，我们运用文件的相互传值实现网页中老师界面

和学生界面的传值。

签到：老师登录后在学生考勤网页中，在文本框中随便输入一个密码，然后告诉学生。学生登录学生界面相应的考勤网页输入密码和自己的学号则代表签到。此时老师界面中的名单中会删除掉相应学生的信息，即留下来的为未签到的学生信息。（类似于微信面对面建群，需要建群人的密码才可入群，‘建群人’为老师，‘入群人’为来上课的学生）



比如：在老师考勤界面输入 **321**

然后登录学号为2的学生网页分别输入321、2后



回到老师界面点击名单，学号为 2 的学生信息被删除，即代表 2 号已签到



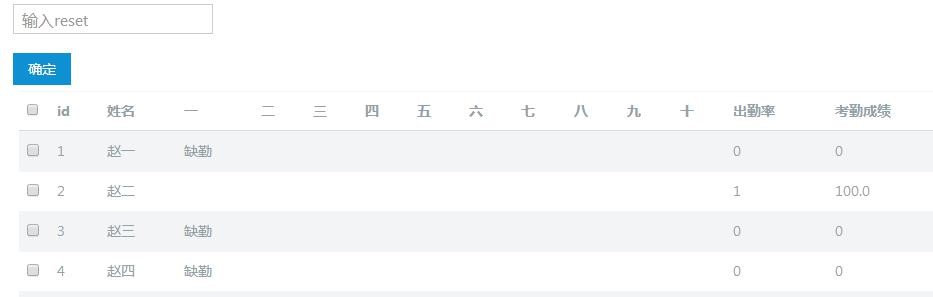
为了防止学生A帮其他人签到，我们将签到的人的id与登录界面的id联系起来，即 id为2的学生输入2后只会取消掉id为2的学生信息，输入3并不会取消掉id为3的学生信息，也就是说学生A只能帮自己签到，不能帮别人签到。

**2.1** 考勤统计：

在考勤统计界面中点击‘确定’按钮后会显示学生的出勤信息，同时表示考勤结束，即点击之后，学生再签到了也是显示缺勤（比如 8 点开始签到，9 点截止，9 点时老师点击上面所说的‘确定’按钮后，学生再进行签到也不会改变结果）。

考勤统计界面：一、二……分别代表第一次考勤，第二次考勤等。当签到已经十次后，可以输入 reset 后界面重置为没有考勤的状态。

接着上面的演示结果，我们得知学号为 2 的学生已签到，故下图显示学号为 2 的学生没有显示‘缺勤’的信息，且出勤率为 1（我们刚才只进行了一次签到）



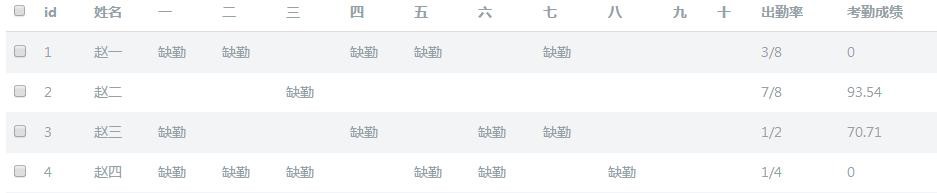
其中出勤率和考勤成绩是自动生成

例如在进行了登录四个学生账号进行了八次考勤后：

关于考勤成绩的计算，考虑到非线性的因素，比如出勤率低于 50%（具体值老师可自己设置）的考勤成绩就为 0，如下图出勤率为 3/8 和 1/4 的学生成绩均为 0，而高于 50%的使用 y=100\*√x，x 为出勤率，y 为成绩。因为这样非线性化更符合老师给分情况，比如在十次签到中，十次全到得到的成绩和签到九次、八次、七次得到的成绩分别为 100 和 94.9、 89.4、83.7（百分制），每缺勤一次，成绩差距并不会很大，不会出现像线性情况中缺勤一次成绩就少 10 分这样分差导致的两极分化情况。

考虑到学生请假或系统 bug 等情况，我们提供修改功能，即可修改是否缺勤（此时会

根据修改后的结果计算出勤率和成绩）。当然，出勤率和考勤成绩老师也是可以修改的。

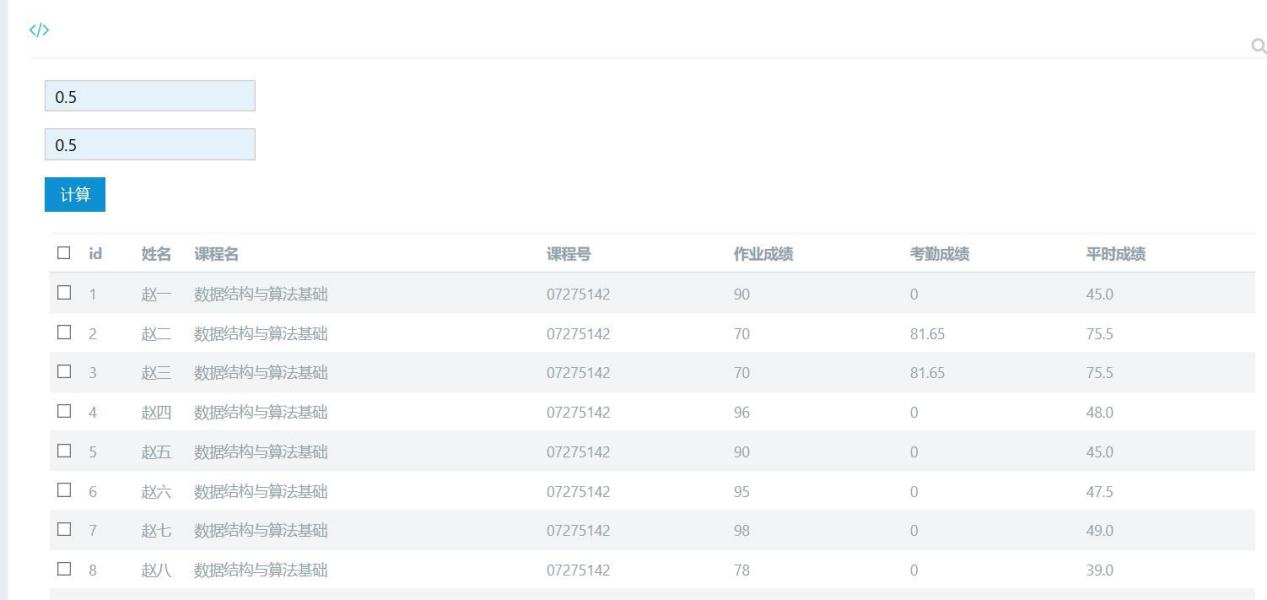


**2.2** 平时成绩管理：

平时成绩由考勤成绩和作业成绩构成

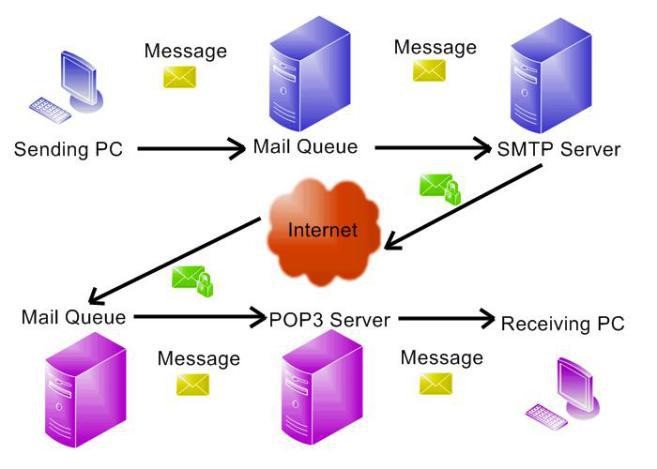
当老师输入考勤成绩和作业成绩的各自占比，网页会自动显示平时总成绩其中作业成绩是来自我们初始建立的数据，可以随意修改。

考勤成绩来自于上述考勤系统生成的考勤成绩。



### **3**、邮件发送模块

邮件发送主要使用 TCP/IP 中的 SMTP 和 POP3 服务，它使用由 TCP 提供的可靠的数据传输服务把邮件消息从发信人的邮件服务器传送到收信人的邮件服务器。跟大多数应用层协议一样，SMTP 也存在两个 端：在发信人的邮件服务器上执行的客户端和在收信人的邮件服务器上执行的服务器端。SMTP 的客户端和服务器端同时运行在每个邮件服务器上。当一个邮件服 务器在向其他邮件服务器发送邮件消息时，它是作为 SMTP 客户在运行。



因为 python 本身支持 SMTP 协议，所以我们只需要进行相应的参数设定即可使用邮件

发送服务。具体代码端配置过程如下：

app.config.update(

DEBUG = True,

MAIL\_SERVER='smtp.qq.com',#以qq邮箱为例

MAIL\_PROT = 465,##端口号

MAIL\_USE\_TLS = True,

MAIL\_USE\_SSL = False,

MAIL\_USERNAME = '2236902736@qq.com',

MAIL\_PASSWORD = 'uwvoqelnedfhdjgc',#qq邮箱生成的授权码

MAIL\_DEBUG = True

) mail = Mail(app)

相应邮件端：



### **Python** 中引入的模块：

from flask import Flask from flask import render\_template from flask import request from flask\_mail import Mail, Message import re import json from fractions import Fraction 所定义的函数：

def strpscj(): risk\_result = [] with open('pscj.json', 'r') as f:

collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result def strpscj2(): risk\_result = [] with open('pscj2.json', 'r') as f:

collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result

def stwpscj(collection): file = open('pscj.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def stwpscj2(collection): file = open('pscj2.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def stw1(collection): file = open('studentINF1.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def stw2(collection): file = open('studentcesi.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def stwtj(collection): file = open('tongji.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def stwcount(collection): file = open('count.txt', 'w') file.write(collection) file.close()

def stwid(collection): file = open('id.txt', 'w') file.write(collection) file.close()

def strid():

with open('id.txt', 'r') as f:

result = f.read()

return result

def strcount():

with open('count.txt', 'r') as f:

result = f.read()

return result

def stwbiaoji(collection): file = open('kqbiaoji.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def stwkql(collection): file = open('kqlv.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def strkql(): risk\_result = [] with open('kqlv.json', 'r') as f:

collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result

def stwkqcj(collection): file = open('kqcj.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def strkqcj(): risk\_result = [] with open('kqcj.json', 'r') as f:

collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result def strtj(): risk\_result = [] with open('tongji.json', 'r') as f: collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result

def str1():

risk\_result = [] with open('studentINF1.json', 'r') as f: collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result

def strbiaoji(): risk\_result = [] with open('kqbiaoji.json', 'r') as f: collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result

def wrtans(collection): file = open('answer.txt', 'w') file.write(collection) file.close()

def readxgcz(): risk\_result = [] with open('xueshengxx.json', 'r') as f: collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result

def readans():

with open('answer.txt', 'r') as f:

result = f.read()

return result

def biaoji(h,c): k=strbiaoji() for i in range(0,len(k)): for j in range(0,len(k[i])):

if k[i][0] == h : g='缺勤' k[i][c] = g break

stwbiaoji(k)

def biaojicz(collection): file = open('biaojicz.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close()

def readbjcz(): risk\_result = [] with open('biaojicz.json', 'r') as f: collection = f.read().split('\n')[:-1] for x in collection: json\_x = json.loads(x) risk\_result.append(json\_x)

return risk\_result

def chuqing(): global c1 global c2 global r global cj cj = []

r=[] x=strbiaoji() for i in x: c1=0 for j in i: if j=='缺勤': c1=c1+1

c2=int(strcount())-1 b1=c2-c1 b2=c2 f=Fraction(b1,b2)

r.append(str(f)) i[12]=str(f) if float(f)<0.5: y=0 cj.append(y)

if float(f)>=0.5: y=round(100\*float(f)\*\*0.5,2) cj.append(y)

i[13]=str(y)

stwkqcj(cj) stwkql(r) stwbiaoji(x)

x=[] for i in strpscj():

x.append(i[4])

stwpscj2(x) def pinshicj(m,n):

A=[]

B=[]

D=[] for i in strkqcj():

A.append(int(i)\*m) for i in strpscj2():

B.append(int(i)\*n) for i in range(0,69):

C=A[i]+B[i]

D.append(round(C,3)) return D

#print(pinshicj(0.5,0.5)) def deletexx(data,h):

d=[]

hang = []

print(data) if h==len(data):

for i in range(0,len(data)-1):

d.append(data[i])

else: for i in data:

hang.append(i[0])

hh=hang.index(h) for i in range(hh,len(data)-1): data[i]=data[i+1]

for i in range(0, len(data) - 1):

d.append(data[i])

return d

def shaixuan(data,input1,input2,n):

sa=[] for i in data:

if float(i[n]) >= input1 and float(i[n]) <= input2:

sa.append(i)

return sa def paixu(data,n,h): post = []

collection =list(data)

for i in range(0, h):

collection11 = collection[i][n] post.append(collection11)

#print(post) postint=[] for i in post:

postint.append(float(i))

p=postint #print(p) postint=list(set(postint))

for index in range(1, len(postint)): current\_val = postint[index] position = index

while position > 0 and postint[position - 1] < current\_val:

postint[position] = postint[position - 1] position -= 1

postint[position] = current\_val

ind=[] #print(postint) for i in range(0,len(postint)):

for j in range(0,len(p)):

if postint[i]==p[j]: ind.append(j)

#print(ind) od=[] for i in ind:

od.append(collection[i])

print(od) return od

def mohucaxun(data,inputs,n,h): collection = list(data) post = []

for i in range(0, h):

collection1 = collection[i][n] post.append(collection1)

def sousuo(scann, post): aaa = []

pattern = '.\*'.join(scann) regex = re.compile(pattern) for item in post: match = regex.search(item) if match:

aaa.append(item)

return aaa

cc = []

g = sousuo(inputs, post) news = [] for i in g:

if i not in news:

news.append(i)

print(news)

for x in collection:

for j in x:

for k in news:

if j == k: if x.index(j)==n:

cc.append(x)

print(cc) return cc

def addNone(data):

add=['None','None','None','None','None','None','None','None','None','No ne'] data.append(add) return data

def xiugai(data,h,n,inputs):

collection=[] for i in data:

collection.append(list(i))

collection[h][n]=inputs return collection

def xiugair1(h,n,inputs):

risk\_result = [] with open('xueshengchengji1.json', 'r') as f: risk\_new\_list = f.read().split('\n')[:-1] for x in risk\_new\_list: json\_x = json.loads(x) risk\_result.append(json\_x)

f.close() collection=[] for i in risk\_result:

collection.append(list(i))

collection[h][n]=inputs

file = open('xueshengchengji1.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close() return collection

def xiugair2(h,n,inputs): risk\_result = [] with open('xueshengchengji2.json', 'r') as f: risk\_new\_list = f.read().split('\n')[:-1] for x in risk\_new\_list: json\_x = json.loads(x) risk\_result.append(json\_x)

f.close() collection=[] for i in risk\_result:

collection.append(list(i))

collection[h][n]=inputs

file = open('xueshengchengji2.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close() return collection

def xiugair3(h,n,inputs): risk\_result = [] with open('js.json', 'r') as f: risk\_new\_list = f.read().split('\n')[:-1] for x in risk\_new\_list: json\_x = json.loads(x) risk\_result.append(json\_x)

f.close() collection=[] for i in risk\_result:

collection.append(list(i)) collection[h][n]=inputs

file = open('js.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close() return collection

def xiugair(h,n,inputs): risk\_result = [] with open('xueshengxx.json', 'r') as f: risk\_new\_list = f.read().split('\n')[:-1] for x in risk\_new\_list: json\_x = json.loads(x) risk\_result.append(json\_x)

f.close() collection=[] for i in risk\_result:

collection.append(list(i))

collection[h][n]=inputs

file = open('xueshengxx.json', 'w') for i in collection: json\_i = json.dumps(i) file.write(json\_i + '\n')

file.close() return collection

def deletehang(data,h): d=[] if h==len(data):

for i in range(0,len(data)-1):

d.append(data[i])

else:

for i in range(h,len(data)-1): data[i]=data[i+1]

for i in range(0, len(data) - 1):

d.append(data[i])

return d

def jinquecaxun(data,inputs,n,h): collection = list(data) post = [] for i in range(0, h):

collection1 = collection[i][n] post.append(collection1) print(post) def sousuo(scann, post):

aaa = []

pattern = '.\*'.join(scann) regex = re.compile(pattern) for item in post: match = regex.search(item) if match:

aaa.append(item)

return aaa

cc = []

hh=[] g = sousuo(inputs, post)

#查重 news = [] for i in g:

if i not in news: news.append(i)

print(news) for o in news:

if o == inputs: hh.append(o)

print(hh) for x in collection:

for j in x:

for k in hh:

if j == k:

if x.index(j) == n: print(x) cc.append(x)

return cc