**Python大作业**



系统名称： 竞赛管理系统

姓 名： 张泰民

班 级： 计科1708

完成日期： 2019年12月27日

**分工情况**

|  |  |  |
| --- | --- | --- |
| **学号** | **姓名** | **分工** |
| **17110501127** | **张泰民** | **前端** |
| **17110501125** | **雷洪润** | **后端** |

**成绩评定表**

|  |  |
| --- | --- |
| **姓名** | **成绩** |
| **张泰民** |  |
| **雷洪润** |  |

**评语：**

**签名：**

**时间：**

目录

[一、背景介绍 1](#_Toc28504528)

[二、功能介绍 1](#_Toc28504529)

[三、详细使用说明 2](#_Toc28504530)

[1. 用户操作 2](#_Toc28504531)

[1.1进入网站 2](#_Toc28504532)

[1.2 报名 2](#_Toc28504533)

[1.3查看报名结果 6](#_Toc28504534)

[1.4查看报名结果 7](#_Toc28504535)

[2. 管理人员 8](#_Toc28504536)

[2.1后台登陆 8](#_Toc28504537)

[2.2新增比赛 9](#_Toc28504538)

[2.3查看管理当前比赛 10](#_Toc28504539)

[四、总结 1](#_Toc28504540)

[附：系统源代码 1](#_Toc28504541)

# 背景介绍

学校学科竞赛是高校培养应用型和创新型人才的重要载体,对培养大学生的综合素质及实践能力有着至关重要的作用。每个学校都会举办各种各样的竞赛活动，通过举办各个专业领域的竞赛，能够调动学生的积极性，锻炼学生的智力，意志力，培养学生良好的自学能力，创新意识。然而，在当前学科竞赛种类繁多的情况下,依然有部分院校在报名时,采用现场报名的方式。这存在着工作人员工作量大、效率低,学生不便等问题,有效的解决方式之一就是使报名过程网络化。这就需要一个高效稳定的系统来辅助报名工作，功能包括但不限于创建比赛，展示比赛信息，学生报名信息提交，Excel表格的导入，报名结果展示，将数据导出至Excel表格。通过这样一个报名系统，使竞赛从报名到数据导出都在电脑端完成，简化管理人员的操作，提高了工作效率，尽可能减少因为人为因素而产生的错误，从而推进竞赛活动在学校内的开展。

# 二、功能介绍

竞赛管理系统共分为四个个主要功能：

1. 比赛信息管理：
2. 创建比赛：可以设置比赛的名称、具体信息，比赛起始时间，参赛人数限制等。
3. 竞赛列表：可以查看所有的竞赛信息，并进入查看详细的报名列表
4. 报名列表：可以查看当前比赛的所有报名信息，并进行比赛信息的导入导出，报名的审核工作等。
5. 比赛报名
   1. 学生可以通过网站进行报名信息的提交，根据不同的比赛类型，填写相应的报名表，报名表提交的信息会自动导入数据库；
   2. 信息收集过程中能对收集的信息格式进行检测，并及时反馈给用户。
6. 查询/修改信息：可以查询并修改各个队伍的信息
7. 导入/导出比赛信息：可以随时将比赛信息按照规定的格式导入到数据库，或导出到Excel表格中。
8. 排名展示：每场比赛的竞赛排名可以通过相应网页地址进行展示。

# 详细使用说明

## 用户操作

### 1.1进入网站

输入网址进入比赛列表界面，浏览当前存在的比赛信息，比赛状态信息与结束时间，如图1所示。



图 1 系统主页

### 1.2 报名

选择要报名的比赛，点击标题进入，这里以山东理工大学网络安全比赛为例，进入页面后，会首先查询当前比赛的所有信息，页面会默认展示团队赛，参赛人数默认为最少参赛人数，用户可以自行进行人数的增加，但最大人数不得超过比赛设置最大人数。如图2所示



图 2 报名页面

点击新增成员按钮即可新增队员，如图3所示。



图 3 新增成员

当前新增人数大于比赛所设置最大人数后，就会提示不能继续新增成员。如图4



图 4 新增成员超出人数提示

鼠标移入相应队员会出现删除按钮，点击可以删除改队员信息，最少比赛人数必须大于比赛所设置最少人数，否则报错。如图5所示



图 5 删除人员及提示

也可以选择报名个人赛，输入信息点击提交即可报名成功



图 6 个人比赛报名

当用户填写内容不全或格式不正确时，系统会提示错误并拦截提交信息，如图7。



图 7 错误提示

### 1.3查看报名结果

在比赛列表界面点击查看报名按钮即可查看当前比赛的队伍报名情况，显示各个队伍的名称，人员信息以及通过情况，如图8。



图 8 报名结果界面

用户也可通过搜索框进行搜索，查找所需的队伍信息，如图9所示。



图 9 报名信息搜索

### 1.4查看报名结果

报名截止的比赛，管理员上传成绩后即可通过主界面相应按钮进入报名结果界面，如图10， 图11所示

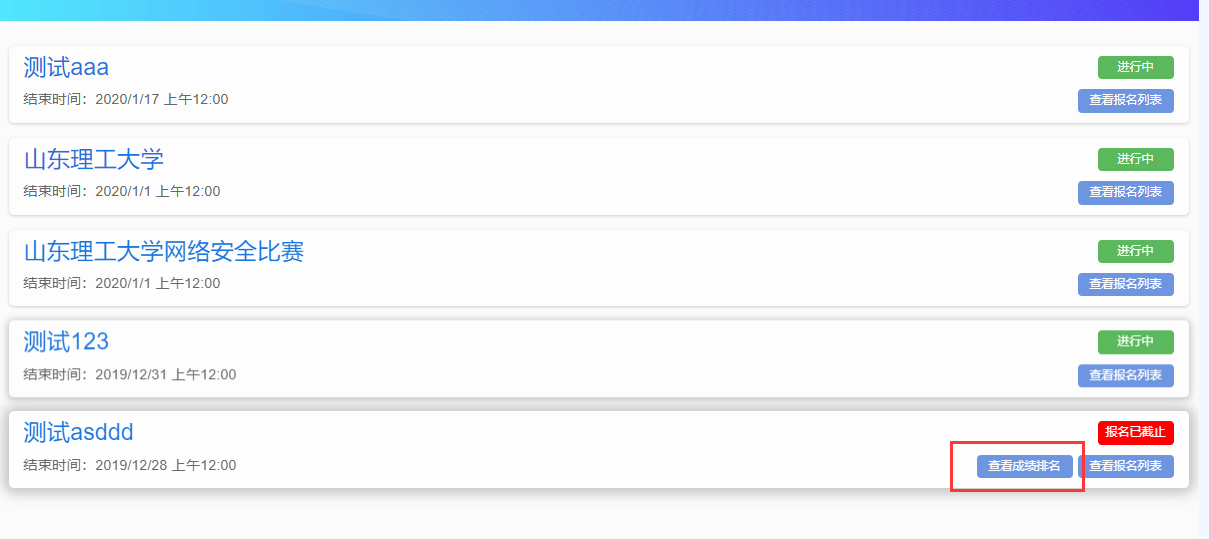


图 10 查看排名



图 11 排名列表

## 2. 管理人员

### 2.1后台登陆

输入后台网址即可进入后台管理登陆界面，输入管理员账户密码点击登陆，即可进入后台管理主界面，如图12，图13。

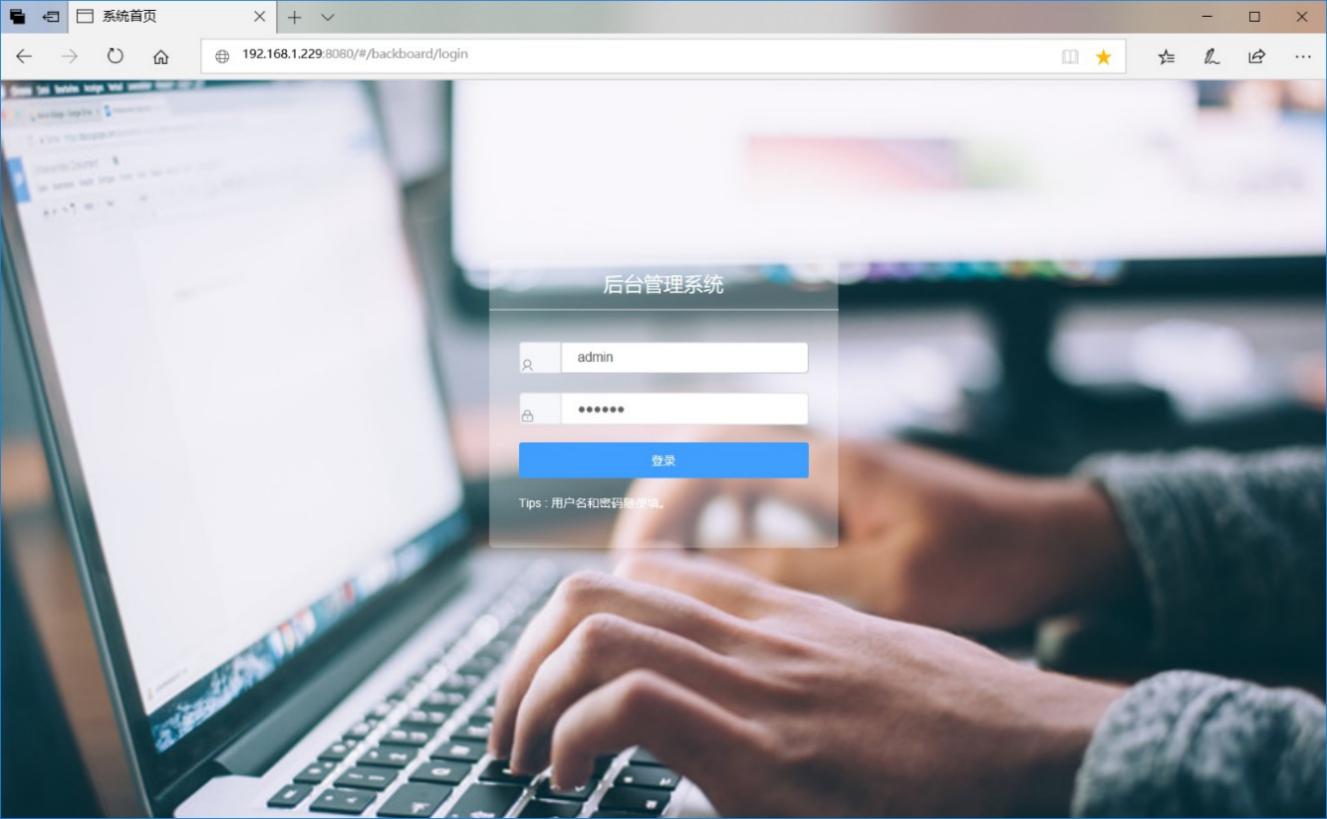


图 12 后台登陆页面

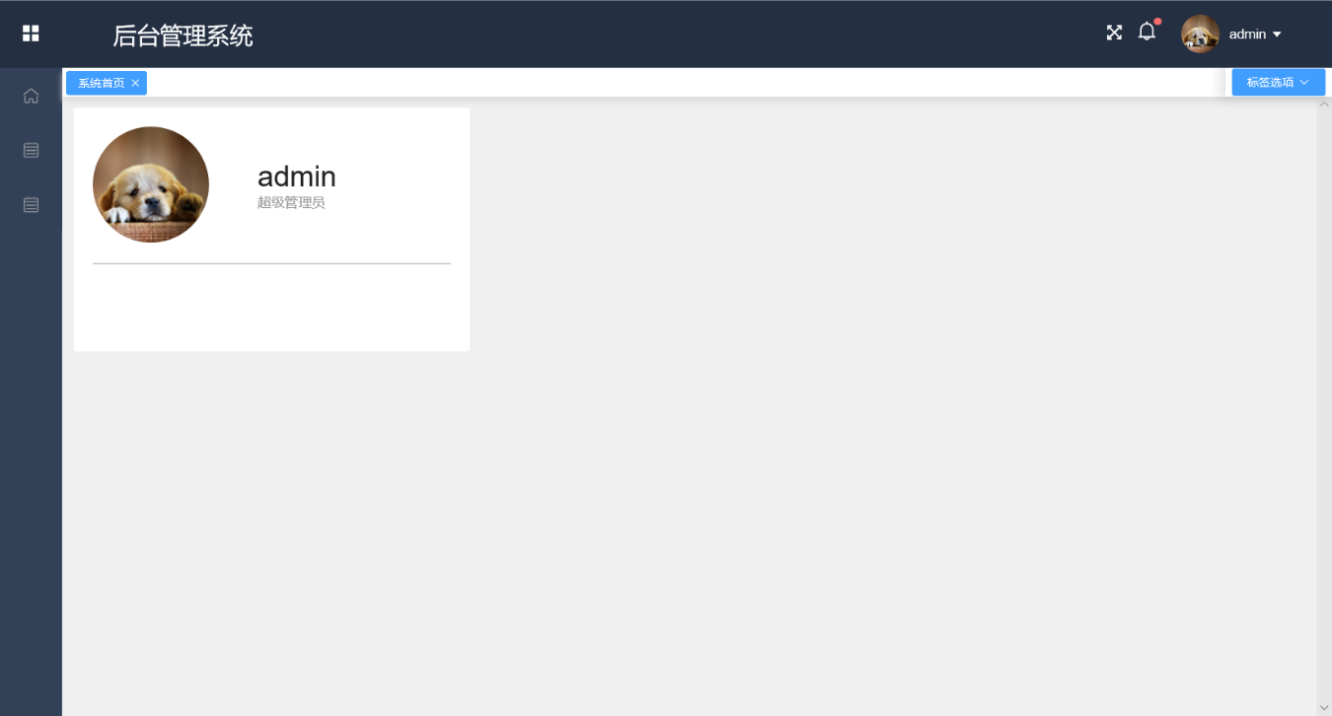


图 13 登陆主界面

### 2.2新增比赛

点击左侧的工具栏中新增比赛按钮，即可进入新增比赛页面输入比赛相关信息，点击提交即可创建比赛，点击重置重新输入信息，如图14

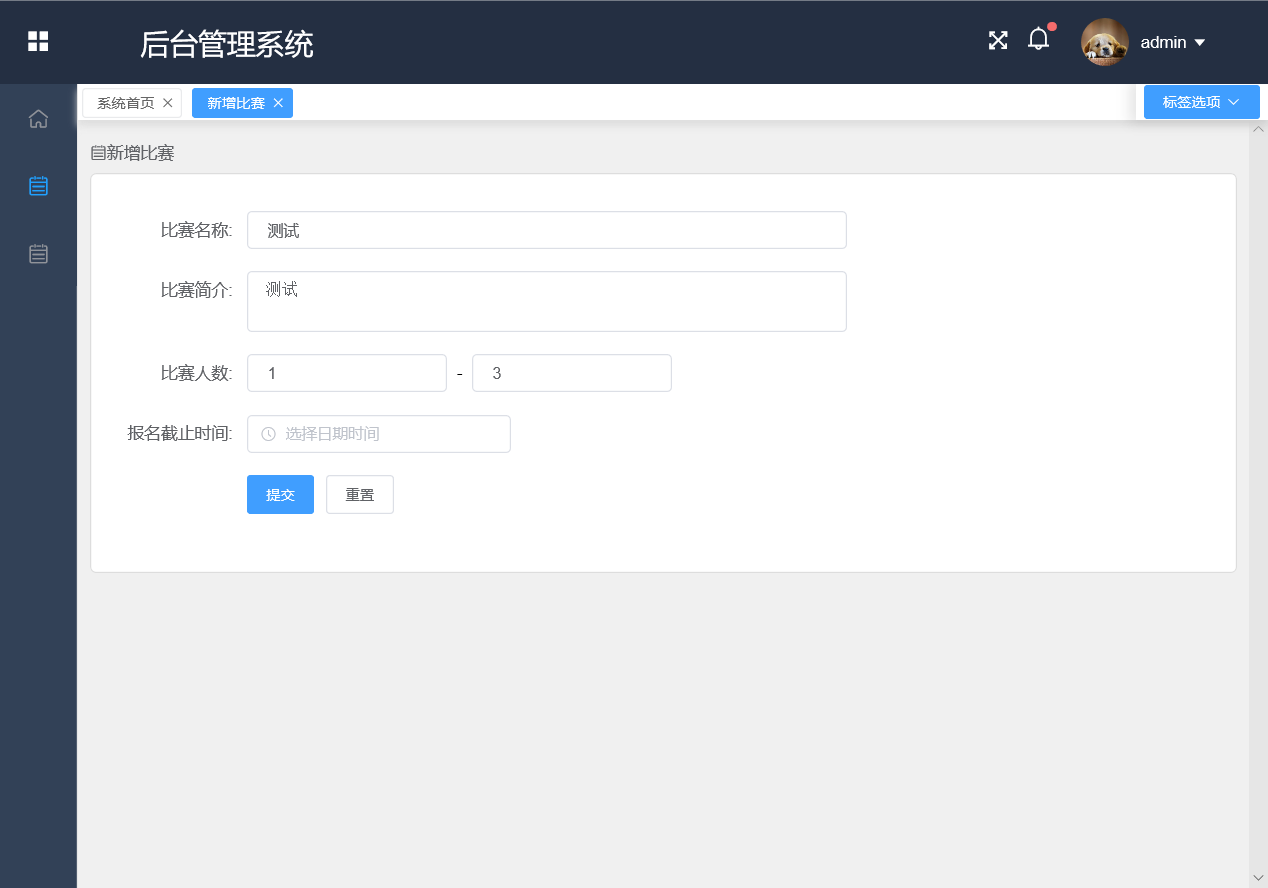


图 14新增比赛界面

### 2.3查看管理当前比赛

点击左侧工具栏的比赛列表按钮即可进入比赛列表界面查看当前存在的比赛的相关信息，点击删除按钮即可进行删除，点击查看报名列表，即可查看对应比赛的报名情况，如图15， 图16。

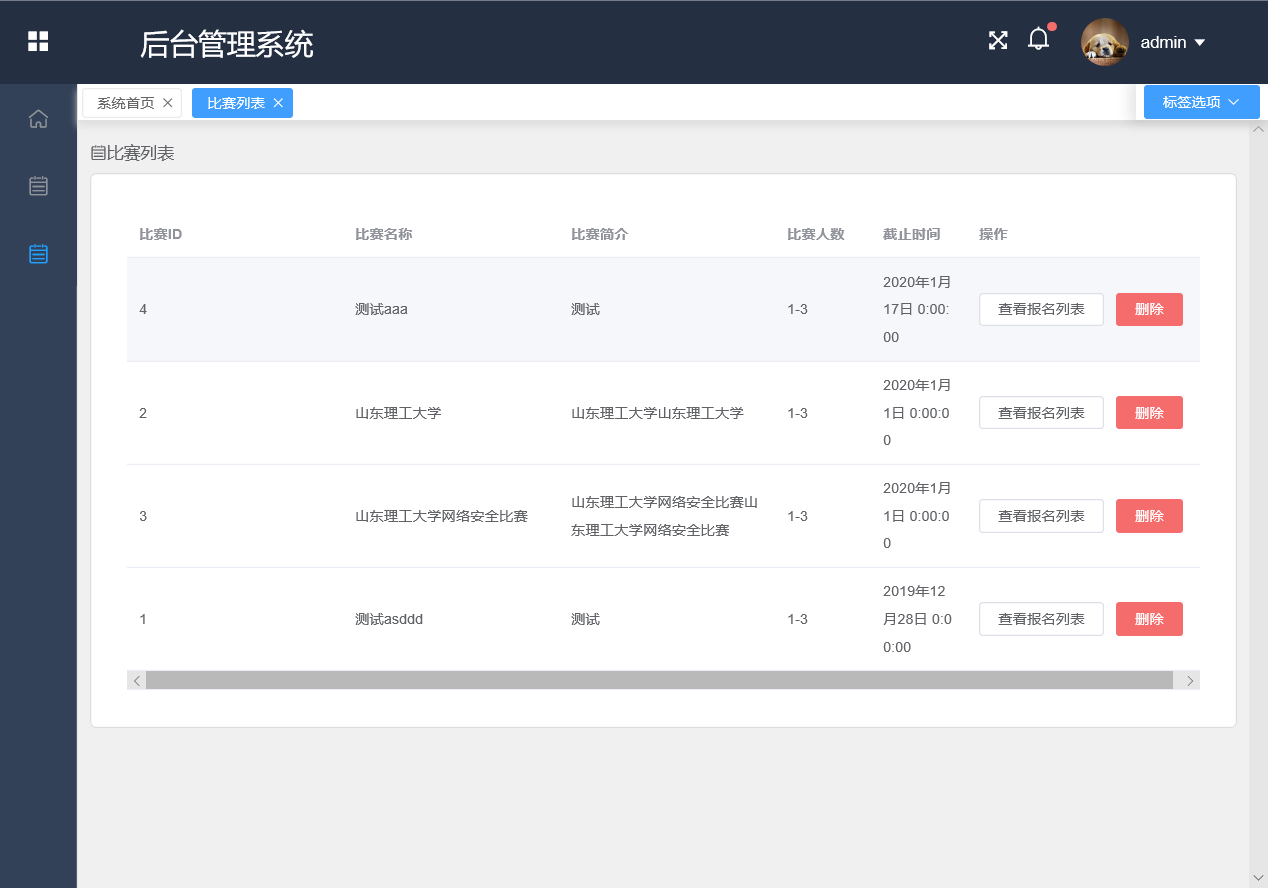


图 15 比赛列表界面

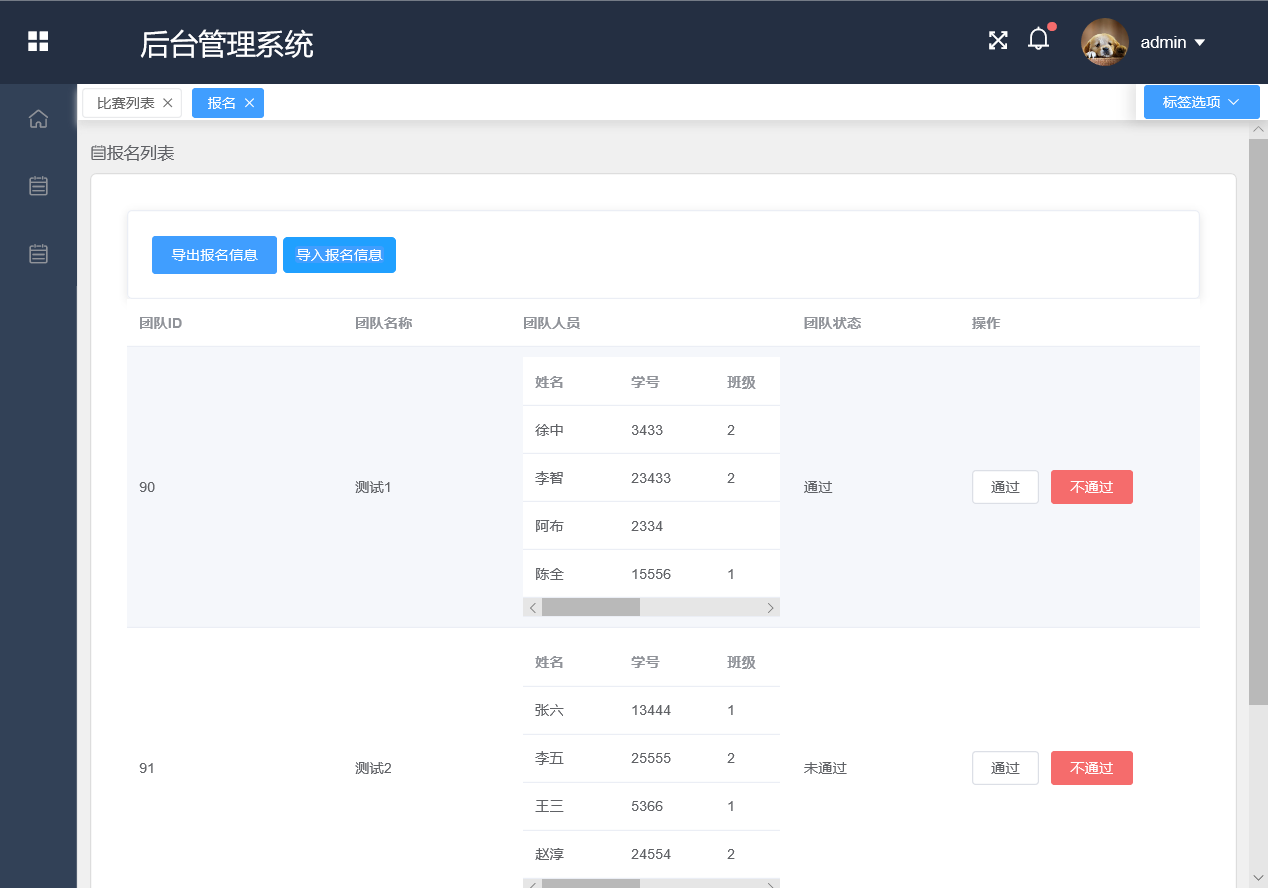


图 16比赛报名情况

点击上方导入报名信息，即可选择Excel文件导入报名系统

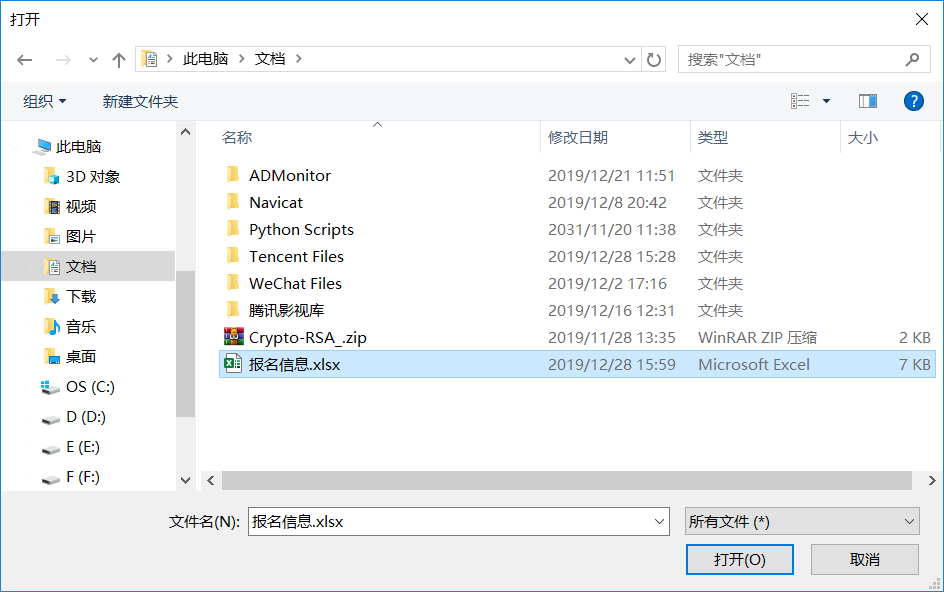


图 17 报名信息导入

点击导出报名信息即可以Excel格式下载当前比赛的报名信息，如图18

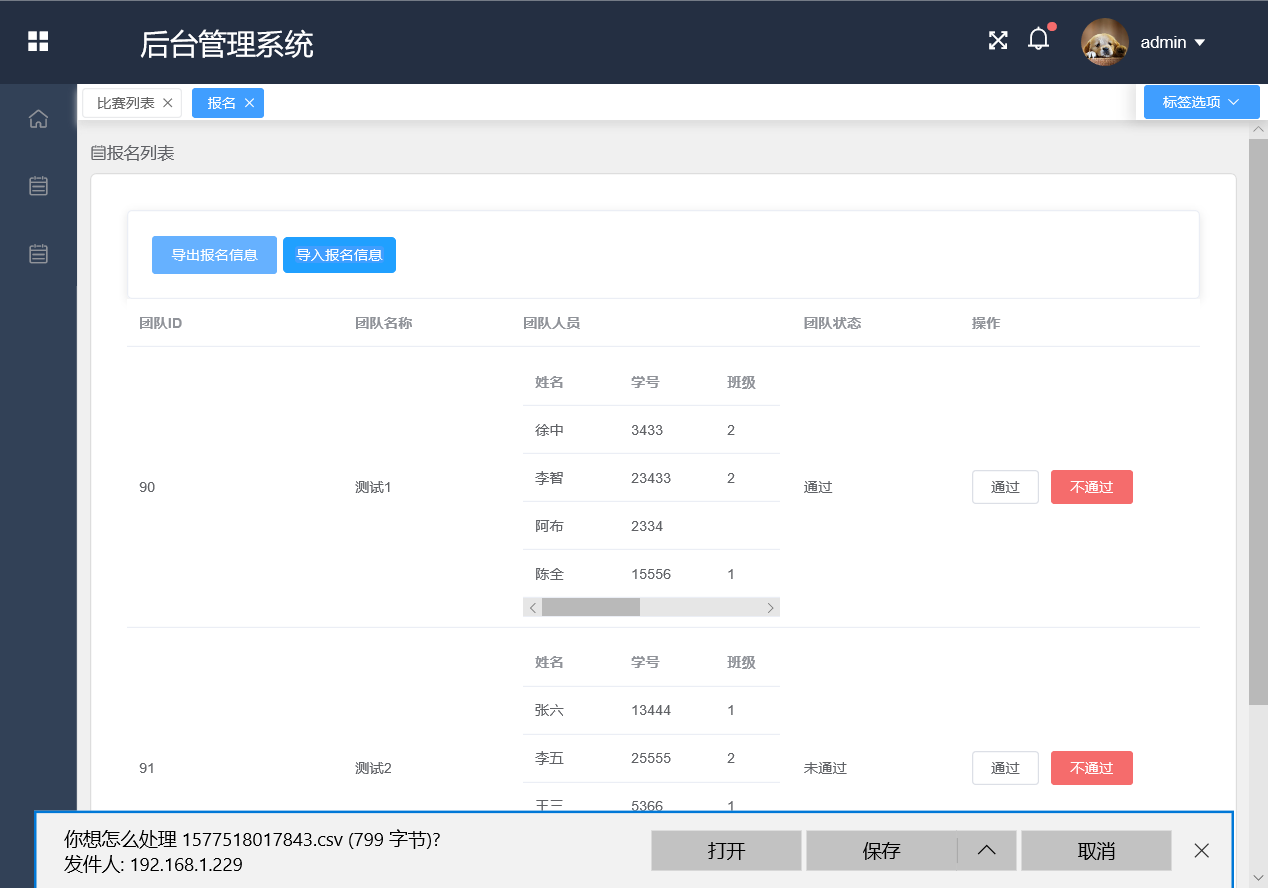


图 18 比赛信息导出

点击团队右侧的通过与不通过按钮即可选择队伍的通过情况。

# 总结

竞赛报名系统时学校智能化发展的重要组成部分，本系统通过Django+vue前后端分离技术，实现了网页端的竞赛报名平台，实现对学生报名信息的收集，比赛的发布，管理，学生的数据保存，成绩导入导出等功能。减少了传统报名的复杂度，提高了工作人员的工作效率。

# 附：系统源代码

|  |
| --- |
| # -\*- coding:utf-8 -\*-  from django.http **import** HttpResponse  **import** time  **import** json  **import** MySQLdb  from django.views.decorators.csrf **import** csrf\_exempt  def connect():  db = MySQLdb.connect("localhost", "root", "123456", "registration", charset='utf8', port=3306)  **return** db    @csrf\_exempt  def team(request): #团队赛  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  tname = res['name'] #团队名称  mid = res['mid'] #比赛id  type = 1 #团队类型，个人为0，团队为1  cur.execute('insert into team (tname, type) values(%s, %s)', (tname, type)) #执行插入操作，先插入团队  con.commit()  cur.execute('select tid from team where tname = %s',(tname,))  count = cur.fetchone()  tid = **int**(count[0])  identity = 0  **for** student in res['personItems']:  sname = student['stuName'] #姓名  sno = student['stuNum'] #学号  c = student['class'] #班级  qq = student['qqNum'] #qq  tel = student['phoneNum'] #电话  college = student['college'] #专业  identity = identity + 1 #身份1为队长，其它为队员  cur.execute('insert into user values(%s, %s, %s, %s, %s, %s, %s, %s, %s)', (sname, sno, tid, identity, tel, qq, c, mid, college))  con.commit()  con.close()  **return** HttpResponse(1)  def get(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  type = res['type']  con = connect()  cur = con.cursor()  cur.execute('select sname from team')    def personal(request): #个人赛  res = json.loads(request.body.decode('utf-8'),strict=False)  print(time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time())))  con = connect()  cur = con.cursor()  tname = res['name'] #团队名称  type = 0 #团队类型，个人为0，团队为1  mid = res['mid']  print(time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time())))  cur.execute('insert into team (tname, type) values(%s, %s)', (tname, type)) #执行插入操作，先插入团队  con.commit()  cur.execute('select tid from team where tname = %s',(tname,))  count = cur.fetchone()  tid = **int**(count[0])  identity = 0  sname = res['stuName'] #姓名  sno = res['stuNum'] #学号  c = res['class'] #班级  qq = res['qqNum'] #qq  tel = res['phoneNum'] #电话  college = res['college'] #专业  identity = identity + 1 #身份1为队长，其它为队员  print(time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time())))  cur.execute('insert into user values(%s, %s, %s, %s, %s, %s, %s, %s, %s)', (sname, sno, tid, identity, tel, qq, c, mid, college))  con.commit()  con.close()  print(time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time())))  **return** HttpResponse(1)    def CreateM(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  mname = res['mname']  des = res['describe']  begin = res['begin']  end = res['end']  min = res['min']  max = res['max']  print((mname, des, begin, end, min, max))  cur.execute('insert into m (mname,des,end,min,max) values(%s, %s, %s, %s, %s)', (mname, des, end, min, max))  con.commit()  con.close()  **return** HttpResponse(1)  def MatchList(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  cur.execute('select \* from m order by end DESC')  key = ["mname", "mid", "describe", "begin", "end", "min", "max"]  result = cur.fetchall()  teamList = []  **for** mdata in result:  match = dict(zip(key,mdata))  teamList.append(match)  a=json.dumps(teamList,ensure\_ascii=False)  **return** HttpResponse(a)  def UserList(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  mid = res['mid']  cur.execute('select team.tname,team.tid,team.status from team,user where user.tid = team.tid and mid = %s group by user.tid',(mid,))  team = cur.fetchall()  key = ["sname", "sno", "tid", "identity", "tel", "qq", "class", "mid", "colege"]  teamkey=["tname", "personItems","tid","status"]  teamList = []  **for** t in team:  tname = t[0]  cur.execute('select \* from user where tid = %s and mid = %s',(t[1],mid))  result = cur.fetchall()  status = t[2]  tid = t[1]  a = []  a.append(tname)  b = []  **for** udata in result:  mid = udata[7]  user = dict(zip(key,udata))  b.append(user)  a.append(b)  a.append(tid)  a.append(status)  data = dict(zip(teamkey,a))  teamList.append(data)  c=json.dumps(teamList,ensure\_ascii=False)  **return** HttpResponse(c)    def test(request):  **return** HttpResponse(1)    def Mid(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  cur.execute('select mid from m')  result = cur.fetchall()  teamList = []  **for** mdata in result:  teamList.append(mdata)  a=json.dumps(teamList,ensure\_ascii=False)  **return** HttpResponse(a)    def Accept(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  tid = res['tid']  status = res['status']  con = connect()  cur = con.cursor()  cur.execute('update team set status = %s where tid = %s', (status,tid))  con.commit()  **return** HttpResponse(1)    def DeleteM(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  mid = res['mid']  cur.execute('delete from m where mid = %s', (mid,))  con.commit()  con.close()  **return** HttpResponse(1)    def GetMatch(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  mid = res['mid']  cur.execute('select \* from m where mid = %s',(mid,))  key = ["mname", "mid", "describe", "begin", "end", "min", "max"]  result = cur.fetchall()  teamList = []  **for** mdata in result:  match = dict(zip(key,mdata))  teamList.append(match)  a=json.dumps(teamList,ensure\_ascii=False)  **return** HttpResponse(a)      def InsertTeam(request): #批量插入团队赛信息  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  mid = res['mid']  count = (len(res['header']) - 1)/6  **for** per in res['body']:  tname = per[res['header'][0]]  type = 1  cur.execute('insert into team (tname, type) values(%s, %s)', (tname, type)) #执行插入操作，先插入团队  con.commit()  cur.execute('select tid from team where tname = %s',(tname,))  r = cur.fetchone()  tid = **int**(r[0])  i = 0  p =[]  p.append(tid)  p.append(mid)  **for** key ,items in per.items():  **if** i != 0:  p.append(items)  **if** i%6 == 0 and i != 0:  cur.execute('insert into user (tid,mid,sname,sno,tel,qq,class,college) values(%s,%s,%s,%s,%s,%s,%s,%s)', tuple(p)) #执行插入操作，先插入团队  con.commit()  p.clear()  p.append(tid)  p.append(mid)  i = i + 1  con.close()  **return** HttpResponse(1)    def SearchTeam(request):  res = json.loads(request.body.decode('utf-8'),strict=False)  con = connect()  cur = con.cursor()  tname = res['tname']  mid = res['mid']  cur.execute('select team.tname,team.tid,team.status from team,user where user.tid = team.tid and mid = %s and team.tname = %s group by user.tid',(mid,tname))  team = cur.fetchall()  key = ["sname", "sno", "tid", "identity", "tel", "qq", "class", "mid", "colege"]  teamkey=["tname", "personItems","tid","status"]  teamList = []  **for** t in team:  tname = t[0]  cur.execute('select \* from user where tid = %s and mid = %s',(t[1],mid))  result = cur.fetchall()  status = t[2]  tid = t[1]  a = []  a.append(tname)  b = []  **for** udata in result:  mid = udata[7]  user = dict(zip(key,udata))  b.append(user)  a.append(b)  a.append(tid)  a.append(status)  data = dict(zip(teamkey,a))  teamList.append(data)  c=json.dumps(teamList,ensure\_ascii=False)  **return** HttpResponse(c) |