

早睡早起组

# 高考招生系统

软件工程项目总结文档

段清楠 白云仁 侯禹凡 李昊阳 邵韵秋

2017-1-15

## 目录

|       |                   |    |
|-------|-------------------|----|
| 1     | 前言 .....          | 3  |
| 1.1   | 系统概述 .....        | 3  |
| 1.2   | 开发工具 .....        | 3  |
| 1.3   | 文档概述 .....        | 3  |
| 1.4   | 概念解释 .....        | 4  |
| 2     | 系统功能介绍 .....      | 4  |
| 2.1   | 系统结构 .....        | 4  |
| 2.2   | 用户管理 .....        | 5  |
| 2.2.1 | 注册 .....          | 5  |
| 2.2.2 | 登录与退出登录 .....     | 6  |
| 2.3   | 教师端 .....         | 6  |
| 2.3.1 | 发送微信推送 .....      | 7  |
| 2.3.2 | 编辑个人信息 .....      | 7  |
| 2.3.3 | 管理学生 .....        | 8  |
| 2.3.4 | 管理志愿者 .....       | 9  |
| 2.3.5 | 管理活动问卷 .....      | 10 |
| 2.3.6 | 管理估分测试： .....     | 11 |
| 2.4   | 学生端 .....         | 12 |
| 2.4.1 | 修改个人信息 .....      | 12 |
| 2.4.2 | 在线估分 .....        | 13 |
| 2.4.3 | 查看排名 .....        | 14 |
| 2.4.4 | 录取情况 .....        | 14 |
| 2.4.5 | 消息 .....          | 14 |
| 2.4.6 | 联系我们 .....        | 15 |
| 2.5   | 志愿者端 .....        | 15 |
| 2.5.1 | 账户生成 .....        | 16 |
| 2.5.2 | 账户登录 .....        | 16 |
| 2.5.3 | 查看、修改个人信息 .....   | 16 |
| 2.5.4 | 查看学生列表 .....      | 17 |
| 2.5.5 | 查看学生详细信息 .....    | 18 |
| 2.5.6 | 填写日期问卷 .....      | 18 |
| 2.6   | 微信模块 .....        | 19 |
| 2.6.1 | 接口配置 .....        | 20 |
| 2.6.2 | 关注欢迎提示 .....      | 21 |
| 2.6.3 | 关键词自动回复 .....     | 22 |
| 2.6.4 | 自定义菜单 .....       | 22 |
| 2.6.5 | 网页授权 .....        | 24 |
| 2.6.6 | 通过微信公众号登录网站 ..... | 25 |

---

|        |                       |    |
|--------|-----------------------|----|
| 2.6.7  | 绑定微信号自动登录网站.....      | 26 |
| 2.6.8  | 推送发布与查看.....          | 27 |
| 2.6.9  | 关于 access_token ..... | 28 |
| 2.6.10 | 其他.....               | 28 |
| 2.7    | 数据设计 .....            | 28 |
| 2.8    | 用户界面设计 .....          | 31 |
| 2.8.1  | 主界面 .....             | 31 |
| 2.8.2  | jQuery UI.....        | 32 |
| 2.8.3  | FusionCharts.....     | 33 |
| 2.8.4  | jQuery Growl .....    | 33 |
| 2.8.5  | 移动端界面简介 .....         | 34 |
| 3      | 开发与评价 .....           | 35 |
| 3.1    | 开发过程 .....            | 35 |
| 3.2    | 单元测试 .....            | 35 |
| 3.2.1  | 测试内容.....             | 35 |
| 3.2.2  | 测试结果.....             | 36 |
| 3.2.3  | 测试过程.....             | 37 |
| 4      | 问题与改进.....            | 37 |
| 5      | 下一步工作的方向.....         | 39 |
| 6      | 收获 .....              | 40 |
| 7      | 致谢 .....              | 42 |

# 1 前言

## 1.1 系统概述

本项目为高考招生辅助系统，是基于微信公众号开发的 web 应用，在软件工程课程中隶属 PRJ8。

高考招生工作主要分为高三下学期对特定学生进行信息收集以及宣传工作，考后初期对考生进行估分收集，考分公布之后与考生约谈通知录取情况这三个部分。而其中繁琐的数据收集和协调时间工作会花费老师和志愿者大量的时间精力。

在此实际背景下，本项目的开发目的为借助此系统帮助老师在各时期发布对应信息，收集，跟踪，管理考生的资料，统筹协调志愿者的工作时间，同时使考生及时获取信息。

本项目的用户分为 3 类：老师、志愿者、学生，对应着不同的功能和权限。譬如，老师可以查看跟踪学生信息，审核估分，管理志愿者，发布消息等。志愿者主要是协助老师记录学生相关情况，反馈有空的时间段。而学生主要是可以通过该系统进行估分自测，以及及时查看录取情况等。

## 1.2 开发工具

本系统开发过程中使用过的工具（框架、库、插件）有：

|                                |
|--------------------------------|
| Django 1.10                    |
| jQuery                         |
| jQuery UI                      |
| CLEditor                       |
| Python xlrd, xlwt, PIL, pytz 库 |
| jQuery Validation              |
| jQuery Growl                   |
| Font Awesome                   |

## 1.3 文档概述

本文档分为前言、系统功能介绍、开发过程评价和总结部分。系统功能介绍部分将介绍系统的模块结构，并分别介绍每个模块的功能。开发过程评价部分将回顾本组同学的开发过程，并总结经验和教训。最后的总结部分分为问题与改进、下一步工作、收获与致谢几部分

展开。

## 1.4 概念解释

超级管理员：可以直接操作 admin 后台的管理人员，拥有全系统最高的权限，能更改所有信息，目前由开发人员担任。超级管理员是拥有添加或删除教师用户的唯一操作者。

完全（账户管理权限）：可以新建账户，可以查看并编辑账户所有信息，并可以删除账户。

只读（账户管理权限）：可以查看部分账户信息，但不能编辑，也不能删除账户。

无（账户管理权限）：不能查看到账户信息。

正常（账户状态）：用户可以在系统允许的范围内进行各项操作。

挂起（账户状态）：用户只能查看、修改个人信息，不能进行其他操作。教师用户可以挂起志愿者和学生，禁止他们访问系统。

学生组：本系统引入了学生组的概念，允许教师用户建立学生组，将同一类的学生加入组中进行管理。教师发放通知时，也可选择发到某一个组。

# 2 系统功能介绍

## 2.1 系统结构

我们开发的高考招生系统主要由四个模块组成：教师端、学生端、志愿者端和微信模块。其中，教师端是面向使用本系统的教师用户，为他们提供多样的管理功能。学生端主要是面向使用本系统的高中生用户。志愿者端面向的用户群体则是每年暑期的招生志愿者（一般为清华大学在校学生）。微信模块用来将我们开发的网站与微信连接，使得学生可以在微信上完成基本的登录、估分、查看录取情况等功能。

招生系统对信息保密性有着特殊的要求。教师、学生和志愿者三类用户的信息会分别存储。用户间只能通过系统中已有的功能进行交互。

教师、学生、志愿者的用户管理权限如下：

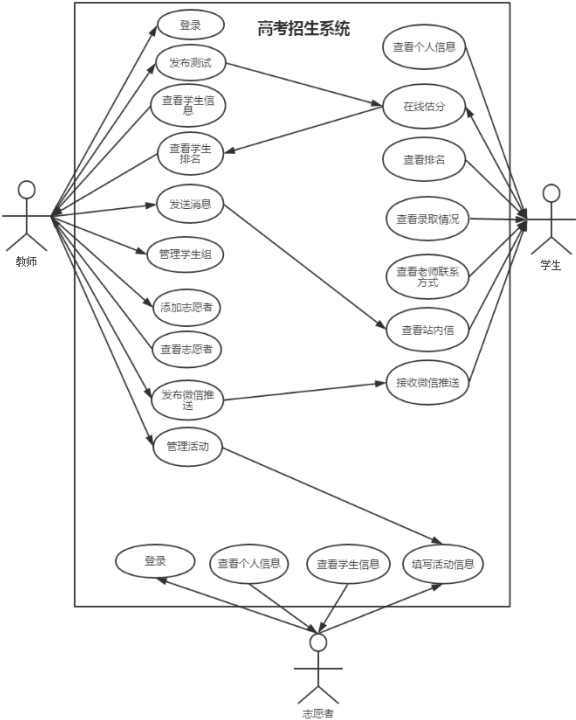
| 管理对象<br>操作者 | 教师* | 学生* | 志愿者* | 自己 |
|-------------|-----|-----|------|----|
| 教师          | 无   | 完全  | 完全** | 完全 |
| 学生          | 无   | 无   | 无    | 完全 |

|       |    |    |    |    |
|-------|----|----|----|----|
| 志愿者   | 无  | 只读 | 无  | 完全 |
| 超级管理员 | 完全 | 完全 | 完全 | -  |

\* 不包含该教师/学生/志愿者自己的用户

\*\* 部分信息（如志愿者的姓名、学号）不能被教师编辑，因为通常情况下教师不必编辑这些信息。

本系统中用户之间的交互形式较多，下面以用例图加以分析：



本系统的特点是不同身份的用户之间交互较多，身份相同的用户之间交互较少，甚至几乎没有。招生系统的特性就是信息的不对称，只有教师才拥有完整的信息。所以很多通知性的操作要由教师发出。相比之下学生之间、志愿者之间的交流需求很有限。教师之间的交流很重要，但往往是以招生会议等形式进行的，因此本系统中也没有再加入相应的功能。

## 2.2 用户管理

### 2.2.1 注册

学生在本系统中注册后，方能正常使用。注册时，系统能够在前端验证所填写的用户名、邮箱、密码以及邀请码的合法性，不合法则无法成功注册。一个激活码成功使用之后即失效，不能被用来注册其他账户。登录注册界面如下图所示



学生在注册界面中填入用户名、电子邮箱、密码、注册码等即可注册。注册成功后页面弹出注册成功的提醒，并且系统会向所填的邮箱自动发送一封邮件，如下图所示



清华欢迎你  
欢迎报考清华大学！

## 2.2.2 登录与退出登录

学生可以在主页输入用户名、密码、验证码，然后点击“学生登录”按钮来登录。另外，也可以使用我们的微信公众号提供的链接登录。第一次使用微信登录之后，我们会在数据库中记录用户的 OpenID，与账号绑定。当学生再次使用微信打开我们的网页时，我们会验证其 OpenID 是否与数据库中相符，对于相符的用户可以跳过登录过程直接访问网站。这样学生在手机端无需再次输入用户名和密码来登录。本文档微信端部分也有相关说明。

当用户登录进去以后，会生成一个 session，用于以后的服务器与客户端的交流。老师、学生、志愿者对应的 key 是不同的，所以不同角色的用户可以在同一台电脑上登录，但是相同角色的用户只能登录一个。当用户点击退出登录时，相应的 session 将会被删除，当关闭浏览器时，相应的 session 也会被删除。

## 2.3 教师端

| 页面标题    | URL                               | 页面功能          |
|---------|-----------------------------------|---------------|
| 首页      | /teacher/                         | 教师端首页, 欢迎界面   |
| 添加学生    | /teacher/add_student/             | 老师生成注册序列号     |
| 查看学生    | /teacher/search_student/          | 老师检索学生        |
| 学生查看排名  | /teacher/rank_student/            | 老师查看一个省内的学生排名 |
| 学生分组    | /teacher/list_group/              | 老师给学生分组       |
| 发送消息    | /teacher/new_message/             | 老师给某组成员发送消息   |
| 添加志愿者   | /teacher/add_volunteer/           | 老师添加一个志愿者     |
| 查看志愿者   | /teacher/search_volunteer/        | 老师检索志愿者       |
| 管理活动问卷  | /teacher/manage_activity/         | 老师向志愿者发放活动问卷  |
| 管理测试    | /teacher/manage_test/             | 老师管理测试题       |
| 发送微信图文  | /teacher/wechat_push_stack/       | 老师给学生发送微信图文   |
| 用户信息    | /teacher/profile/                 | 查看和编辑自己的个人信息  |
| 学生详情    | /teacher/student_info/?id=        | 老师查看某一学生详情    |
| 修改学生详情  | /teacher/student_info_edit/?id=   | 老师修改某一学生详情    |
| 志愿者详情   | /teacher/volunteer_info/?id=      | 老师查看某一志愿者详情   |
| 修改志愿者详情 | /teacher/volunteer_info_edit/?id= | 老师修改某一志愿者详情   |

### 2.3.1 发送微信推送

我们用一个类似于栈的结构来维护微信消息, 每次将当前的数据插入到栈顶。需要获取最新的消息则直接取栈顶的数据, 获取旧的消息则获取从栈顶开始的若干条数据, 不过此处没有进行退栈的操作。

页面校验:

| 校验域    | 校验规则       |
|--------|------------|
| 图片链接   | 必填、必须选择非空项 |
| 图文消息链接 | 必填、必须选择非空项 |
| 标题     | 必填、必须选择非空项 |
| 摘要     | 必填、必须选择非空项 |

### 2.3.2 编辑个人信息

老师可以随时进入个人信息一栏维护自己的个人信息, 该信息中的联系方式可以被学生查询得到。向服务器传递编辑信息时, 我们禁用了保存按钮, 当服务器给予响应反馈后再解除禁用, 防止用户重复提交。



页面校验：

|        |  |
|--------|--|
| 校验域    | 校验规则   |
| 姓名     | 必填   |
| 手机     | 必填、长度为 11、全是数字   |
| 办公地址   | 必填   |
| Email  | 必填、必须符合电子邮箱的规范   |
| 修改密码   | 必须是合法的密码格式(字母、数字、+ - ? ! @ # \$ % < > , . ^ & * [ ] , 长度 4~20) |
| 确认修改密码 | 必须和修改密码所填内容完全一样  |
| 个人描述   | 必填   |

### 2.3.3 管理学生

#### 2.3.3.1 添加学生

老师可通过添加学生功能，批量生成注册码，并通过 excel 表的形式导出到本地。在输入框中输入具体数量，点击 add 按钮则生成相应的注册码。一键导出则导出 excel 到本地。在批量生成注册码时带有一定随机性，使得用户不能猜出哪些是合法的注册码。一个注册码只能供一位学生注册。

页面校验：

|                |       |
|----------------|-------|
| 校验域            | 校验规则  |
| 请输入您想新建的学生账户数量 | 必填、数字 |

#### 2.3.3.2 检索学生

当学生注册后，老师可以进入检索学生界面，检索特定学生。我们对检索结果进行了分页显示，每页 15 条记录。此处，直接获取所有特定学生的所有信息，然后进行分页，没有实现只获取仅含某页数据的功能。当前的技术从空间和时间上来说都有很大的优化空间，但是实现的复杂度略大，故暂未实现。在输入框中输入姓名点击检索则可查看结果，点击全选就可将所有条目选择，通过复选框将某条记录选中或取消选中，可以将选中的记录加入到某一个特定的组。

页面校验：

|     |      |
|-----|------|
| 校验域 | 校验规则 |
| 姓名  | 必填   |

#### 2.3.3.3 查看/编辑学生信息

在检索的学生结果中，老师可以查看某一已选择的学生的详情并进行编辑。在查看信息时，所有信息均为不可编辑状态。

页面校验：

| 校验域   | 校验规则           |
|-------|----------------|
| 姓名    | 必填             |
| 身份证号  | 必填、18 位身份证格式   |
| 手机    | 必填、长度为 11、全是数字 |
| Email | 必填、必须符合电子邮箱的规范 |
| 学号    | 必填、必须全是数字      |
| 父亲姓名  | 必填             |
| 父亲手机  | 必填、长度为 11、全是数字 |
| 母亲姓名  | 必填             |
| 母亲手机  | 必填、长度为 11、全是数字 |
| 班主任姓名 | 必填             |
| 班主任手机 | 必填、长度为 11、全是数字 |
| 成绩、排名 | 数字             |

#### 2.3.3.4 管理学生组

老师可以进入管理学生组建立新组，为志愿者和学生分组。页面上的分组列表采用 bootstrap 的 collapse 模块，以折叠导航形式呈现。如果要查看某一组内的志愿者和学生成员详情，点击组名展开即可。每个志愿者和学生都有一个组号字段（一个数字集合），每个学生最多隶属于 5 个小组，志愿者没有限制。按照组号递增的顺序显示每个组，如果每个志愿者或学生的组号字段含当前组号，则显示该人。显示顺序上，先显示所有志愿者，再显示所有学生。

如果某个志愿者和某个学生的组号字段有交集，则该志愿者可以查看该学生的信息。新建一组时用 javascript 直接插入相应的一块代码，并采用 AJAX 技术传递给服务器相应信息；在一个组内添加新成员同理。通过使用 AJAX 手段在不刷新的情况下更新页面内容，提升用户体验。点击修改可以进入学生的具体界面修改详情，点击删除可以从该组中删除该学生。目前我们没有提供删除组功能。

#### 2.3.3.5 发送消息

老师可以进入发送消息功能，向某一学生组发送消息。老师编辑好标题和正文后，选择相应的学生组发送该信息即可。使用了 CLEditor 插件，让正文编辑支持带 HTML 样式的文本编辑，包括表情、字体等多种功能。

页面校验：

| 校验域 | 校验规则 |
|-----|------|
| 标题  | 必填   |

#### 2.3.4 管理志愿者

### 2.3.4.1 添加志愿者

添加志愿者账户时，我们对输入的用户名、密码进行了检查，限定用户名由字母和数字组成，长度 4 到 20 位；限定密码含有字母、数字部分特殊字符，长度 4 到 20 位。前端传递给服务器数据后，服务器会进行再次检查。

页面校验：

| 校验域    | 校验规则   |
|--------|--|
| 预置用户名  | 必填、仅含字母  |
| 预置密码   | 必须是合法的密码格式(字母、数字、+ - ? ! @ # \$ % < > , . ^ & * [ ] , 长度 4~20) |
| 确认预置密码 | 必填、必须和修改密码所填内容完全一样   |

### 2.3.4.2 检索志愿者

当志愿者注册后，老师可以进入检索志愿者界面，检索特定志愿者。对于检索结果超过一定数量者，此处没有分页显示，因为考虑到志愿者数量不多，所以没有必要分页。

页面校验：

| 校验域 | 校验规则   |
|-----|--------|
| 姓名  | 仅含汉字或空 |

在检索的志愿者结果中，老师可以查看某一已选择的志愿者的详情并进行编辑。此处与编辑学生信息类似，因此不再重复阐述。在查看信息时，所有信息均为不可编辑状态。

### 2.3.5 管理活动问卷

老师可以进入管理活动界面进行发放活动问卷。新建活动时，日期选择采用 jQuery UI 的 datepicker 组件，便于填写时间和进行校验。对于已发布的活动，教师可以查看已填信息的志愿者详情，将已有的结果以 excel 格式导出到本地。我们对日期进行了校验，防止用户选择不合法的时间（开始时间不得晚于结束时间）。



页面校验：

|     |             |
|-----|-------------|
| 校验域 | 校验规则        |
| 日期  | 开始日期不晚于结束日期 |

2.3.6 管理估分测试：

老师可以进入管理测试界面进行测试发布、编辑、撤回。点击发布按钮后，该试题在数据库中就是已发布状态，点击撤回后，该试题处于未发布状态，该套试题的所有成绩都会清空。需要重新发布后，学生才能进行估分测试。点击编辑按钮后，可以进入到具体的界面，对该套试题进行编辑。点击删除按钮后，该套试题就会被删除，所有学生的该套试题成绩都会被删除。

新建试卷时，教师需要对时间、地区、科目进行选择框形式选择，这里我们已经列出了可能的选项，只需选择即可。确认以后将会创建一套空的试卷，即里面还没有任何题目。然后进入试题编辑界面进行试题编辑。

在试题编辑界面中，通过对浏览器的特别处理，将上传的试题图片显示到浏览器上，让用户看见上传的试题。然后对题型、科目等项目进行选择，避免产生不必要的错误填写。对分数等用户需要自己输入的项目进行错误检查，避免输入不合法的字符。

当学生提交了自己的考试测试时，老师会接受到提醒，然后点击提醒即可进入估分审核界面。估分审核界面使用了开源的 fusioncharts，老师可以选择某一个学生某一个科目，查看其估分时间的柱状图。老师也可以凭借总估分时间进行评判，判断该生的估分是否有效。对于有效的成绩纳入该学生的总分，并参与排名，对于无效的成绩让同学再次重新估分。

## 2.4 学生端

| 页面标题 | URL               | 页面功能             |
|------|-------------------|------------------|
| 首页   | /student/         | 学生端首页，欢迎界面       |
| 在线估分 | /student/score/   | 选择试题，进行估分，提交估分结果 |
| 查看排名 | /student/rank/    | 查看各科以及总分的成绩和排名情况 |
| 录取情况 | /student/admit/   | 查看老师录入的自己的录取情况   |
| 消息   | /student/message/ | 查看老师发送的站内消息      |
| 联系我们 | /student/contact/ | 查看老师和志愿者的联系方式    |

考生利用学生端与本系统交互。根据需求，学生端需要实现以下功能：

- 1) 考生需要通过给出特定的注册码方能注册成为认证的考生，享受后续的功能。
- 2) 注册考生可以查看自己的基础个人信息，以及后续的录取情况等。
- 3) 注册考生登录后可以看到老师，志愿者的联系方式。
- 4) 注册考生可以查看教师发送的信息和表格。

除此之外，我们还添加了权限管理功能，即老师可以指定学生是否具备完整的访问权限。学生默认拥有正常的权限，但教师可以通过编辑学生信息将其账户置于挂起状态（见本文档 1.4 节）。对于没有权限的学生，不可以进行估分、查看估分结果和老师联系方式或站内消息，仅能够查看和修改个人信息。对于无权限的学生，若访问带有权限的链接，我们能够阻止访问并将页面重置到首页。另外，学生端还要求页面能够自动适配手机（网页、微信等）。

### 2.4.1 修改个人信息

我们允许学生修改和完善个人信息，供老师和志愿者查看。学生点击右上方的铃铛按钮，选择修改个人信息，即可进入修改个人信息界面。其中有些项（如学校）使用文本框供学生填写，有些项（如性别）使用下拉框供学生选择。另外，出生日期这一项我们使用了专门的日期选择框供学生选择。整个界面的所有字段前段均加入了校验，对于不合法的信息修改，在用户点击“保存”时能弹出对话框进行提醒，并且不向后端提交本次修改。

页面校验：

| 校验域   | 校验规则           |
|-------|----------------|
| 姓名    | 必填             |
| 身份证号  | 必填、18 位身份证格式   |
| 手机    | 必填、长度为 11、全是数字 |
| Email | 必填、必须符合电子邮箱的规范 |
| 学号    | 必须全是数字         |
| 父亲姓名  | 汉字             |
| 父亲手机  | 长度为 11、全是数字    |

|       |             |
|-------|-------------|
| 母亲姓名  | 汉字          |
| 母亲手机  | 长度为 11、全是数字 |
| 班主任姓名 | 汉字          |
| 班主任手机 | 长度为 11、全是数字 |

### 2.4.2 在线估分

点击侧边栏的估分测试按钮，进入界面，我们从数据库中获取并列出了这个学生可以进行估分的所有试题。学生可以查看试题的名称（年份、省份、科目）、状态（是否估分过、估分结果），并从中选择要估分的试题。

学生点击“开始测试”按钮，前端将使用 GET 请求链接到试题界面，学生开始对这套题进行估分。估分时，前端根据试卷的名称和正在估分的题号，使用 POST 请求通知后端，后端将给前端返回本道题的全部数据，其中图片是以 URL 的形式给出。在该界面上方正中的位置，有一个 LED 风格的计时器，用来显示学生在本道题上已经用掉的时间。实现原理是用 JS 在前端进行计时，并时刻把计时的数据送给 LED 时钟控件来显示。本界面能显示本题的题号、题型、题目的图片、满分等信息，学生估计好分数后，在下拉框中选择分数（客观题只能选择 0 分或满分，主观题可以选择从 0 分到满分之间的任意整数），点击“提交本题估分结果”按钮，完成本题估分。之后，系统将自动记录下本题的分数和用时，并进入下一题，计时器清 0。



整套题估分完成后，页面会弹出对话框显示总计的用时，将整套题中每道题的的估分值和用时使用 POST 提交到后端，然后使用 GET 返回到之前的试题列表界面，服务器更新估分数据。若学生在弹出估分完成之前退出估分，将不会提交任何估分数据，即本次估分无效。

### 2.4.3 查看排名

点击侧边栏的查看排名按钮，进入界面可以查看自己的估分和排名信息，包括科目，分数，自己的排名，参加排名的总人数等信息。我们提供总成绩和单科的排名信息。



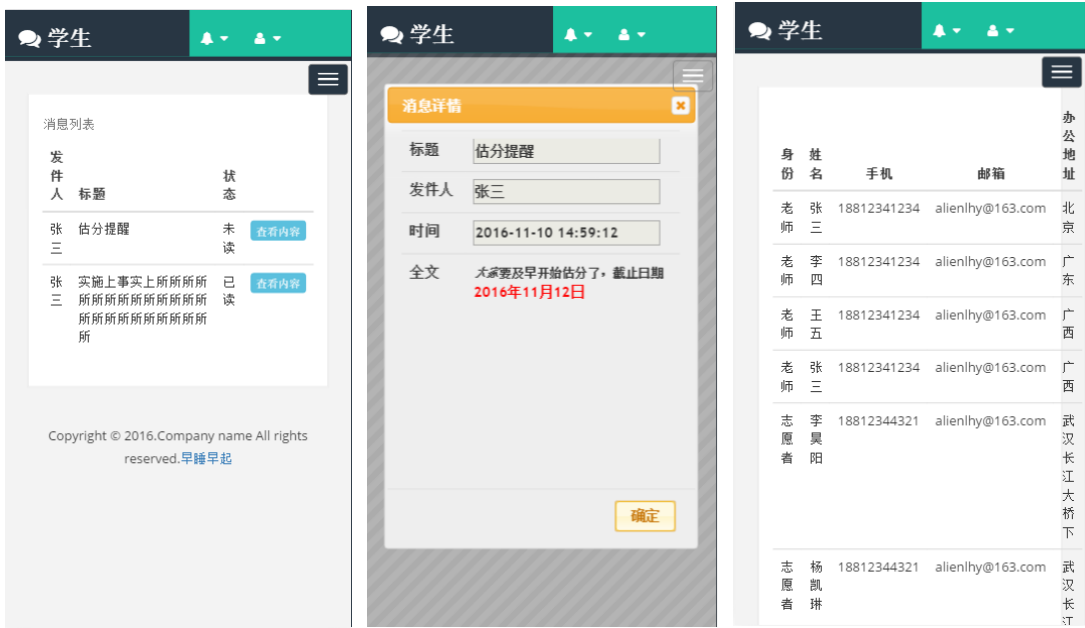
### 2.4.4 录取情况

点击侧边栏的录取情况按钮，进入界面可以查看自己的录取情况，如上右图所示。

### 2.4.5 消息

点击侧边栏的消息按钮，进入界面可以查看老师给自己发送的消息的列表，列表里显示消息的标题，发送者，阅读状态（未读、已读），和查看内容按钮。

点击查看内容按钮，打开对话框，可以查看消息的详细情况，包括标题、发件人姓名、时间和全文，其中全文的文字是支持显示 HTML 效果的（由发件人录入带效果的文字），如下方左 1、左 2 图所示。查看完毕，点击确定返回到消息列表界面。



### 2.4.6 联系我们

点击侧边栏的联系我们按钮, 进入界面可以查看老师和相关志愿者的信息以及联系方式, 包括身份、姓名、手机、邮箱、地址, 如上方右 1 图所示。

## 2.5 志愿者端

概述

| 页面标题 | URL                        | 页面功能             |
|------|----------------------------|------------------|
| 首页   | /volunteer/                | 志愿者端首页           |
| 个人信息 | /volunteer/profile/        | 查看修改个人信息, 填写工作日志 |
| 查看学生 | /volunteer/search_student/ | 查看可见的学生信息        |
| 选择日期 | /volunteer/date_choose/    | 填写老师发布的时间问卷      |
| 退出登录 | /volunteer/logout/         | 志愿者用户退出登录        |

志愿者利用志愿者端与本系统交互, 根据需求, 志愿者端需要实现以下功能:

- 1) 志愿者登录后能够修改自己的信息供老师统计。需要统计的信息包括, 志愿者姓名, 年级, 专业, 能够参与的时间段, 并能生成报表。
- 2) 志愿者能够每天对工作填写总结, 主要是在实际工作中的行程、住宿等进行统计。
- 3) 能够查看老师指定的考生的信息。

我们额外实现了一项功能, 即志愿者可以填写教师发放的日期问卷。同时, 我们仿照学生端, 在志愿者端也加入了权限管理的功能: 若教师限制了志愿者的权限, 则志愿者只能查看和修改个人信息, 访问其他界面时我们会将页面定向到个人信息界面。另外, 志愿者端的



界面无需适配手机，在浏览器中能正常使用即可。

### 2.5.1 账户生成

志愿者用户的账户和初始密码只能由老师用户生成，无其他生成途径。详见本文档教师端部分的说明

### 2.5.2 账户登录

志愿者通过登录界面输入用户名、密码、验证码，点击“志愿者登录”按钮即可登录。这部分与教师端基本相同。

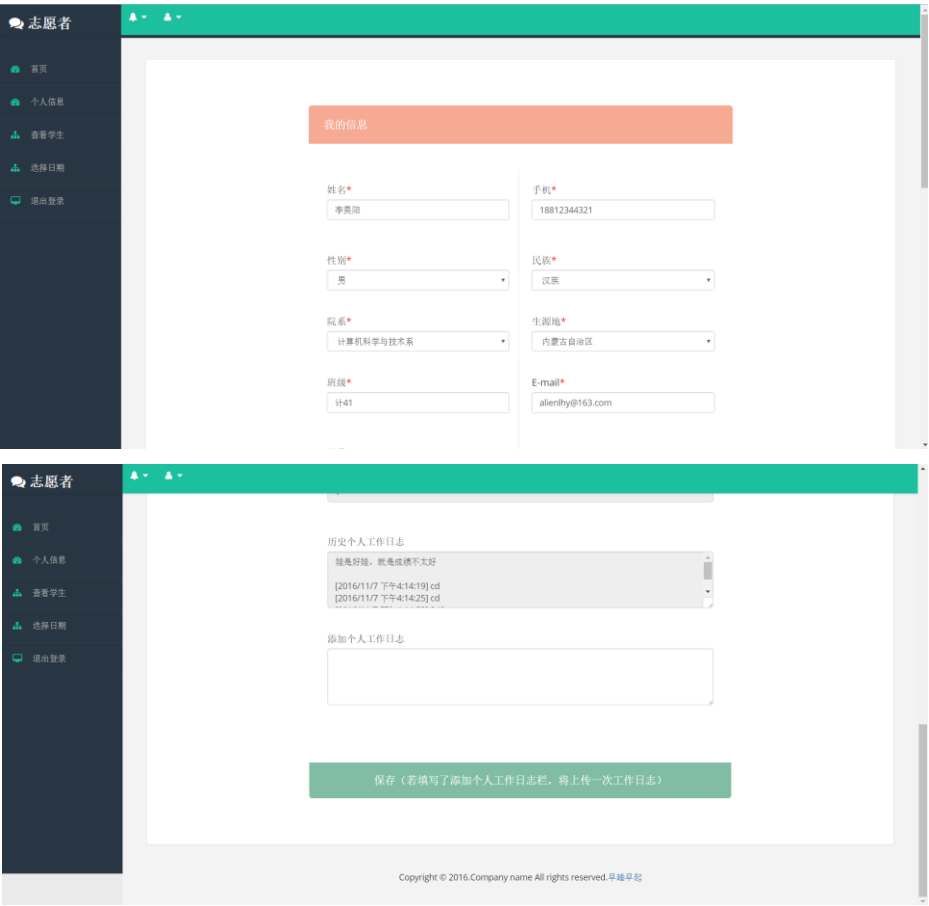
### 2.5.3 查看、修改个人信息

点击侧边栏中的“首页”或者“个人信息”都能链接到个人信息界面，在这个界面可以查看、修改个人信息。个人信息使用界面中的文本框和下拉框列出，打\*的为必填或必选项，点击最下方的保存按钮可以修改个人信息。对于志愿者，我们还支持添加个人工作日志。若在该栏填写了内容，在点击“保存”按钮之后会添加一次工作日志，系统会自动在日志内容前面加上当前的时间。上方也有不可修改的文本框来显示历史工作日志。整个个人信息界面，前端对每个字段都加入了表单校验。当用户在完成一个文本框或下拉框的输入后，若输入的信息不合法，则下方立即显示一行文字提醒用户该字段不合法。这时，即使点击“保存”按钮也不向后端提交本次修改的数据。若通过了前端校验并点击了“保存按钮”，前端会使用 POST 请求，向后端传送整个表单的数据，进而更新数据库。

页面校验：

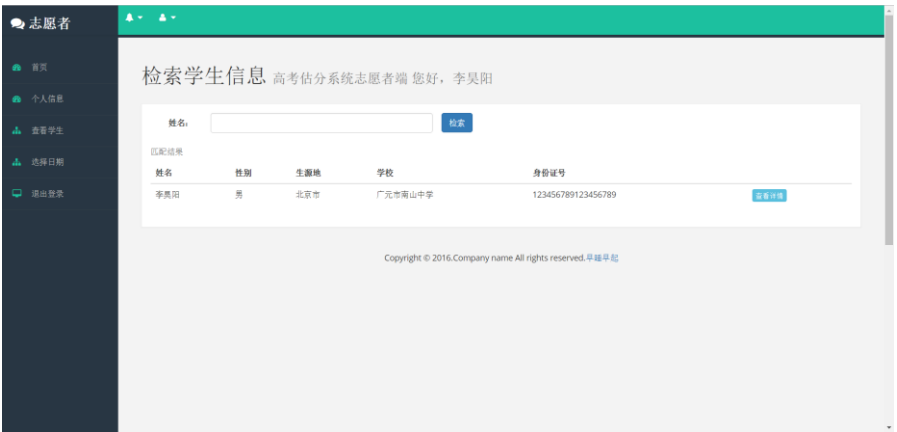
| 校验域    | 校验规则   |
|--------|--|
| 姓名     | 必填   |
| 性别     | 必须选择非空项  |
| 院系     | 必须选择非空项  |
| 民族     | 必须选择非空项  |
| 生源地    | 必须选择非空项  |
| 手机     | 必填、长度为 11、全是数字   |
| 班级     | 必填   |
| Email  | 必填、必须符合电子邮箱的规范   |
| 学号     | 必填、必须全是数字  |
| QQ     | 必须全是数字   |
| 微信     | 无  |
| 修改密码   | 必须是合法的密码格式(字母、数字、+ - ? ! @ # \$ % < > , . ^ & * [ ]，长度 4~20) |
| 确认修改密码 | 必须和修改密码所填内容完全一样  |

|          |   |
|----------|---|
| 添加个人工作日志 | 无 |
|----------|---|



2.5.4 查看学生列表

点击侧边栏的“查看学生”，志愿者可以查看自己所处的学生分组里的全部学生，并可以通过在上方的文本框输入学生姓名，点击“检索”按钮检索出名字相符的学生。此时前端会使用 POST 请求，向后端发送该姓名字符串，后端向前端返回搜索结果的列表，供前端显示。若文本框不输入直接点击“检索”按钮，则将列出所有可见的学生。



### 2.5.5 查看学生详细信息

在查看学生界面，点击要查看详细信息的学生右边的“查看详情”按钮，前端将使用 GET 请求进入学生详细信息界面，如下图。志愿者只能看到学生的姓名、性别、电话等基本信息，而无法查看志愿、分数等敏感信息。

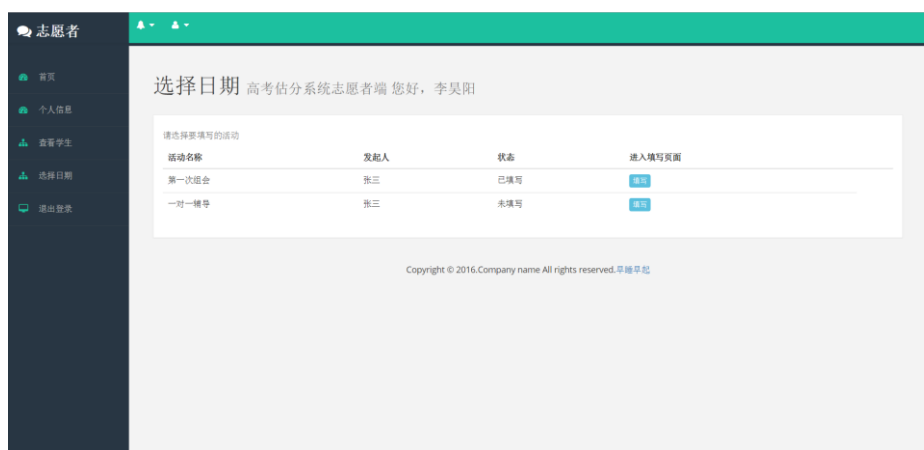


学生详情

|       |                 |
|-------|-----------------|
| 用户名:  | lihy1           |
| 姓名:   | 李昊阳             |
| 注册码:  |                 |
| 性别:   | 男               |
| 民族:   | 汉族              |
| 出生日期: | 2015-01-12      |
| 文理科:  | 文科              |
| 地区:   | 北京市             |
| 手机:   | 18812344321     |
| 邮箱:   | alienhy@163.com |
| 地址:   | 广元市广元镇广元村       |

### 2.5.6 填写日期问卷

点击侧边栏的“选择日期”按钮，进入选择日期界面，前端使用 POST 请求向后端获取全部的活动问卷，并在网页上显示。此时，志愿者可以查看教师发布的活动问卷列表，包括活动的名称、发起人、填写状态信息。



选择日期 高考估分系统志愿者端 您好, 李昊阳

请选择要填写的活动

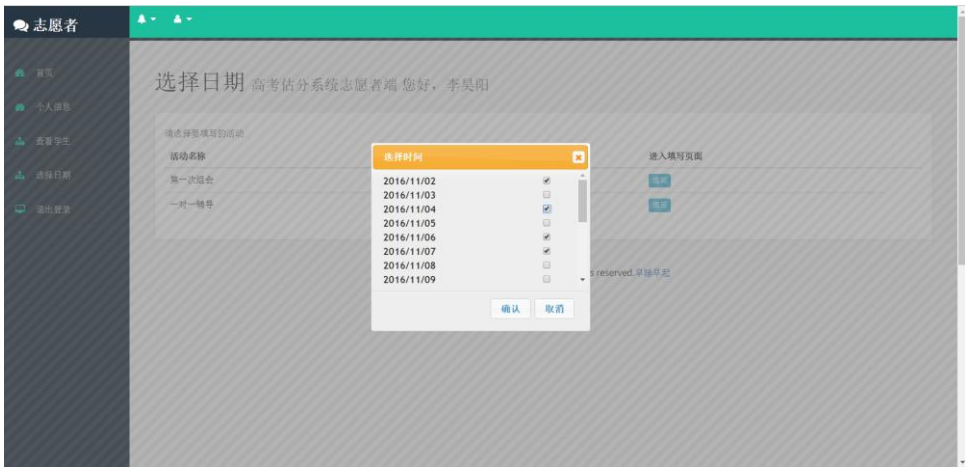
| 活动名称  | 发起人 | 状态  | 进入填写页面             |
|-------|-----|-----|--------------------|
| 第一次班会 | 张三  | 已填写 | <a href="#">填写</a> |
| 一对一辅导 | 张三  | 未填写 | <a href="#">填写</a> |

Copyright © 2016. Company name All rights reserved. 早睡早起

点击某一活动对应的“填写”按钮，将根据该活动的 ID 向后端查询对应的活动日期范围，在对话框中以复选框的形式列出供志愿者进行选择。对于每一个活动，数据库中存储的是起止日期（因为需求中表明活动时间一定是连续的一段时间）。这里我们使用了模式对话框，以保证用户不操作页面上其他元素。

同一个活动可以多次填写时间，系统保留最后一次填写的结果。对已填写时间过的活动

再次点击“填写”按钮，对话框中默认显示的是上一次的填写结果，志愿者可以进行修改并确认提交。每一次确认提交都会让活动的发起人收到一次提醒。



2.6 微信模块

微信端的代码实现主要在工程的 wechat 文件夹中。主要参考微信提供的开发者文档进行开发。

对应文件及函数的功能概述如下表：

wechat/views.py(主要负责与微信公众号的交互)

| 函数名                | 功能  |
|--------------------|---|
| wechat_main        | 与微信服务器交互的主函数，微信服务器的请求都进入该函数，通过 get 和 post 方法进行判断，进行接口校验或分配给 responseMsg 函数处理，最后处理的结果也通过该函数返回给微信服务器。 |
| responseMsg        | 处理传入的 post 请求的内容，请求内容里的 MessageType 对应的信息将请求分配给 handleEvent 或者 handleText 处理，返回上述函数所给出的返回值。         |
| handleEvent        | 处理传入的请求信息 (字典类型)，判断事件是订阅还是取消，分别进行处理，返回对应的 xml 字符串信息。  |
| handleText         | 处理传入的请求信息，实现关键词自动回复。返回对应的 xml 字符串信息。  |
| createMenu         | 创建自定义菜单，并且返回对应的 HttpResponse 对象、  |
| send_pic_text      | 根据传入的类型信息，实现回复单条图文消息的功能，返回对应的 xml 字符串信息。  |
| Send_pic_text_many | 实现回复多条图文消息的功能，主要对应回复历史消息的功能，返回对应的 xml 字符串信息。  |
| get_token          | 获取公众号调用各接口时必须使用的唯一凭证 access_token   |
| token              | 定时调用刷新 access_token   |
| xml2Dic            | 辅助转换函数，请求中 xml 格式的信息转换为字典类型   |

wechat/wechat\_api/py (主要负责网页授权部分)

| 函数名               | 功能  |
|-------------------|---|
| authority         | 根据传入的类型信息, 将对应的 url 进行处理, 返回静默网页授权页面对应的 url   |
| add_authority     | 对传入的 url 进行加工, 返回对应的网页授权页面的 url.  |
| get_code          | 从静默网页授权返回的请求中获取 code 字段对应的 code 码, 返回 (是否获取成功, code 码) 元组                             |
| Get_openid_byCode | 根据输入的 code, 获取用户微信号与该公众号关联的标识 open_id, 并返回一个元组 (与 get_code) 类似。之后会将 open_id 用于自动登录功能。 |

wechat/urls.py 实现 url 与函数的关联, 将对以 wechat 结尾的请求交给 wechat\_main 函数处理。

### 2.6.1 接口配置

此次使用的为微信公众平台提供的测试号, 采用开发者模式进行开发。首先需要在微信公众平台测试号申请的网站上填写接口配置信息 (url 以及自定义的 token) 微信服务器就是通过此处填写的 url 与网站服务器进行交互的。



图 1 接口配置信息

再填写相关信息之后, 微信会给填写的链接发送一个 request 请求, 网站服务器必须正确响应这个请求才能完成配置。在 wechat/views.py 中 wechat\_main 函数中实现。(需要在 urls.py 中对上述 url 与 wechat\_main 函数进行绑定) 判断如果此处为 get 请求, 则通过对 get 请求中传递的 signature 进行校验, 判断该请求来自微信, 并原样返回 get 请求中的 echostr 参数, 才能配置成功。

此处需要注意的是: 处理接口配置请求外, 其他所有微信请求也都是通过访问接口配置时所填的 url 来实现的, 区别是其他都使用 post 方法, 而接口配置使用 get 方法, 以及所传参数不同。所以 wechat\_main 函数为处理微信请求的入口函数, 以及返回值的最终出口。需要再该函数中对请求是 post 还是 get 方法进行分类, 分别传给对应函数进行处理, 最后再

通过 wechat\_main 函数返回给微信服务器实现交互。

## 2.6.2 关注欢迎提示

扫一扫下图二维码，点击关注，则可以收到欢迎信息。（如图 2 所示）



图 2 测试号二维码

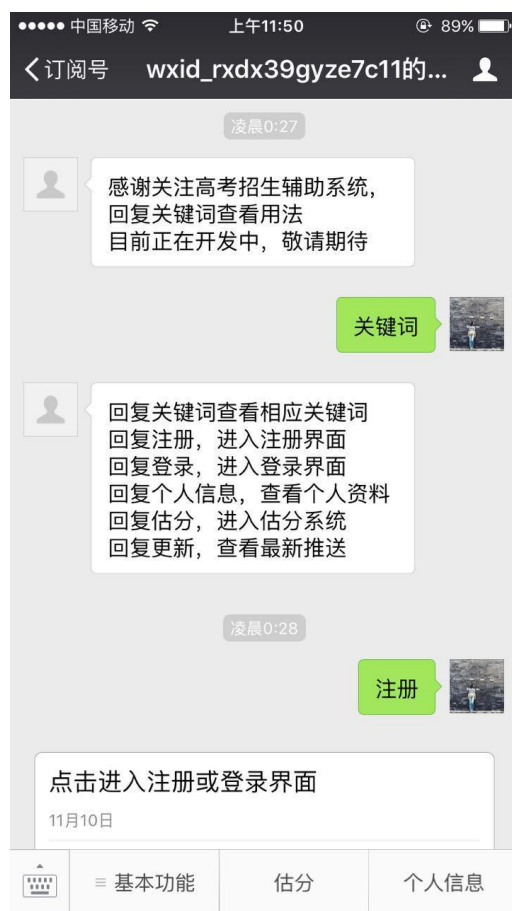


图 3

实现方式如下：

用户点击关注公众号，微信会发送一个 post 请求给服务器，在 wechat\_main 函数中判断为 post 方法，将请求信息传给 responseMsg 函数，先将其转换为字典格式，通过 Msgtype 的判断传给 handleEvent 函数，handleEvent 函数判断 event 类型为订阅事件，就将欢迎信

息包装在开发者文档提供的 xml 格式中，以字符串的形式返回，由 wechat\_main 生成 httpResponse 发给微信服务器即可。

### 2.6.3 关键词自动回复

输入“关键词”，公众号会回复相关关键词自动回复的用法。根据提示输入关键词则会获得相应的网站链接或者推送消息等。例如：输入“注册”，会返回对应的注册界面的链接，点击即可进入网站。

实现方式如下：

用户向公众号发送一条消息，微信会将其打包成一个 post 请求发送给服务器，wechat\_main 函数将其传递给 responseMsg 函数，responseMsg 函数通过对 Msgtype 判断将其传给 handleText 函数，handleText 函数将根据文本内容通过 send\_pic\_text 产生对应的图文消息的回复，最后再封装在开发者文档提供的 xml 格式中作为字符串返回，由 wechat\_main 生成 httpResponse 发给微信服务器即可。

### 2.6.4 自定义菜单

如图 3 所示，公众号最下面一栏即为自定义菜单，从学生使用的角度考虑，将最常用的几个功能放在自定义菜单处，方便直接点击使用。支持的菜单项有基本功能（注册、登录），估分，个人信息。（图 4，图 5，图 6 分别展示了点击不同的菜单项所进入的网页）



图 4 登录界面



图 5 估分界面



图 6 个人信息



实现方式如下：

通过 createMenu 函数创建，将菜单的名字，对应的链接响应参数封装在字典中，将其转为 json 格式发送给微信服务器即可创建。此处选择在响应用户关注或者取消关注该微信公众号时调用 createMenu 函数，实现自动创建自定义菜单的功能。

## 2.6.5 网页授权

想要能从微信流畅的使用网站，则必须经过网页授权，否则，在每一次跳转都会都到微信服务器的警告信息（图 7 为未经授权的效果）。我们的项目进行了静默网页授权，访问时会和直接用浏览器打开一样，跳转流畅，不会收到警告信息。

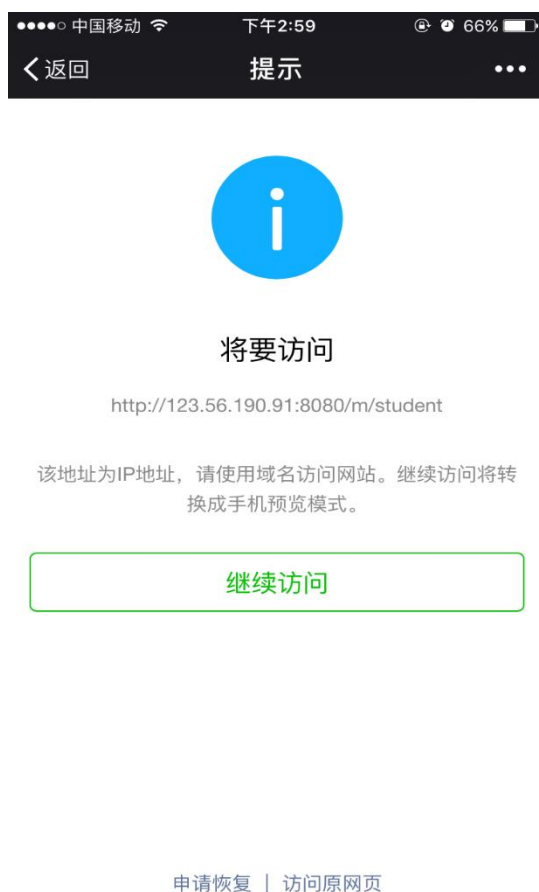


图 7 未经授权的网页

实现方式如下：

现在微信测试号管理页面修改授权回调域名（如图 8 所示），进行该配置后，所有以该域名开头的网址都可以进行网页授权。此处采取静默网页授权，即不需要用户点击确认，就可以自动跳转到对应网页。通过静默网络授权可以获得用户关联该公众号的唯一身份标识

open\_id。只需在自动回复和自定义菜单栏中奖返回给用户的 url 网址，经过 authority 函数的处理，加上 OAuth2.0 网页授权信息的内容返回给用户，引导用户进入加入授权信息 url 对应的网页即可完成静默网页授权。

|      |              |     |                    |
|------|--------------|-----|--------------------|
| 网页帐号 | 网页授权获取用户基本信息 | 无上限 | <a href="#">修改</a> |
|------|--------------|-----|--------------------|

OAuth2.0网页授权

授权回调页面域名:

用户在网页授权页同意授权给公众号后，微信会将授权数据传给一个回调页面，回调页面需在此域名下，以确保安全可靠。沙盒号回调地址支持域名和ip，正式公众号回调地址只支持域名。

图 8 修改授权回调域名

## 2.6.6 通过微信公众号登录网站

回复“注册”/“登录”关键词，可得到登录链接，或点击自定义菜单的按钮，可以进入登录界面。



图 9 登录界面

实现方式如下：

此处通过关键词自动回复以及自定义菜单的包装，给用户提供网站的网址（经过 authority 函数静默授权之后的网址），用户通过点击事件即可进入网站。即实现从微信上进入网站的功能。

### 2.6.7 绑定微信号自动登录网站

此处需要注意的是，因为我们进行了权限检查，未登录的用户如果点击估分或者个人信息的链接会转移到登录界面，必须进行登录才能获取个人信息进行相关操作。此处考虑到，每次都要输入用户名和密码很麻烦，利用微信网络授权的功能，将微信号与学生账号进行绑定。学生只需要通过该平台登录过一次网站，之后再使用时，可以直接通过相应链接或菜单进入自己的功能页面进行操作，而不需要再次输入用户名和密码。

实现方式如下：

当用户通过静默授权的 url 进入网站时，可以在处理访问对应的回调 url 的 get 请求中通过 get\_code 和 get\_openid\_byCode 函数获得用户微信号关联此微信公众号的唯一身份标识 open\_id。在用户第一次登录网站时，将获取的 open\_id 作为用户信息存入数据库中。具

体细节是，第一次用户通过微信客户端打开微信内嵌浏览器，然后成功登陆进去之时，系统会将获取到的 open\_id 和用户的 username 关联，如果同一个 open\_id 登陆不同的用户，则会关联最近一次登录的用户。因为 open\_id 和用户名都具有唯一性，所以之后再次通过微信公众号访问网站中的任一网页时，以 open\_id 作为索引，若查询到对应用户，则该用户就直接成功登录，并获取相关权限和信息，如果该 open\_id 未查找到任何对应的用户，则进入到登录界面通过密码验证登录。

## 2.6.8 推送发布与查看

教室在网站的发布信息界面输入推送链接图片等。学生在微信公众号回复“更新”可获得老师最新发布的推送。回复“历史消息”可获得近期发布的消息（不超过 10 条）。(如下图 10, 11 所示)



图 10 获得最新消息



图 11 获得历史消息

实现方式如下：

教师通过发布消息的界面，填写对应的图片链接，推送链接，推送标题等信息，服务器将其存入数据库中。因为测试号不提供主动发布消息的功能，所以我们此处以关键词自动回复的形式来更新推送和查看历史消息。与关键词自动回复的处理流程类似，此处需要注意的是当判断出关键词为“更新”时，返回的内容为数据库中存储的最新的一条推送（以 xml 格式

包装)，当关键词为“历史消息”时，转到 send\_pic\_text\_many 函数中，将数据库中最新的 10 条推送信息返回。若数据库中推送数目少于 10 条，则返回已有的所有推送。（用对应的 xml 格式包装）

### 2.6.9 关于 access\_token

access\_token 是调用公众号接口的唯一凭证，有效时间最长为 2 小时，需要定期刷新。所以每一次要与微信服务器进行交互需要用到 access\_token 值的时候都调用 token 函数获取。Token 函数会记录距离上一次刷新时间的距离，若小于 1 小时，则返回原值，否则调用 get\_token 函数进行刷新。

### 2.6.10 其他

微信端开发时将微信公众号的关注者（及用户群体）定位为学生，所以微信公众号所提供的主要功能是面向学生的。而教师主要作为微信公众号的管理员，是通过项目的网页对公众平台进行管理，进行消息的发布以及估分的发布等操作，而非从微信公众平台进入项目网页进行操作。

## 2.7 数据设计

本项目数据库中共包含 9 张表，以下介绍每个表的功能及字段含义。

### 1. Teacher

Teacher 表中储存了教师用户的各项信息。

| 字段名           | 含义     | 类型     |
|---------------|--------|--------|
| account       | 用户名    | string |
| password      | 密码     | string |
| realName      | 姓名     | string |
| phone         | 手机     | string |
| fixedPhone    | 固定电话   | string |
| email         | 邮箱     | string |
| area          | 负责地区   | string |
| volunteerList | 管理的志愿者 | list   |
| wechat        | 微信号    | string |
| comment       | 备注信息   | string |

### 2. Student

Student 表中存储了学生用户的各项信息。

| 字段名      | 含义  | 类型     |
|----------|-----|--------|
| account  | 账户名 | string |
| password | 密码  | string |

|                      |                  |          |
|----------------------|------------------|----------|
| realName             | 姓名               | string   |
| birth                | 出生日期             | datetime |
| idNumber             | 身份证号             | string   |
| type                 | 文理科              | int      |
| province             | 省份               | int      |
| sex                  | 性别               | int      |
| nation               | 民族               | int      |
| school               | 学校               | string   |
| classroom            | 班级               | string   |
| address              | 家庭地址             | string   |
| phone                | 手机               | string   |
| email                | 邮箱               | string   |
| dadName              | 父亲姓名             | string   |
| dadPhone             | 父亲电话             | string   |
| momName              | 母亲姓名             | string   |
| monPhone             | 母亲电话             | string   |
| tutorName            | 班主任姓名            | string   |
| tutorPhone           | 班主任电话            | string   |
| major                | 所选专业志愿           | string   |
| testScoreList        | 模拟考试成绩           | list     |
| rankList             | 排名结果             | list     |
| sumNumberList        | 总排名              | list     |
| estimateScore        | 估分结果             | int      |
| realScore            | 实考分              | int      |
| admissionState       | 录取状态             | string   |
| comment              | 教师评论             | string   |
| registercode         | 注册码              | string   |
| teacherList          | 对应的管理老师          | list     |
| volunteerAccountList | 志愿者列表            | list     |
| readed               | 已读信息列表           | int      |
| groupList            | 所在组号             | list     |
| wechat               | 微信号              | string   |
| openid               | 微信 openid, 自动登录用 | string   |
| isRegistered         | 是否注册             | int      |

### 3. Volunteer

Volunteer 表中存储了志愿者用户的各项信息。

| 字段名      | 含义   | 类型       |
|----------|------|----------|
| account  | 账户名  | string   |
| password | 密码   | string   |
| realName | 姓名   | string   |
| birth    | 出生日期 | datetime |

|                      |                 |        |
|----------------------|-----------------|--------|
| idNumber             | 身份证号            | string |
| province             | 省份              | int    |
| sex                  | 性别              | int    |
| nation               | 民族              | int    |
| classroom            | 班级              | string |
| phone                | 手机              | string |
| email                | 邮箱              | string |
| comment              | 教师评论            | string |
| volunteerAccountList | 管理学生列表          | list   |
| groupList            | 所在组号            | list   |
| wechat               | 微信号             | string |
| openid               | 微信 openid, 自动登录 | string |

#### 4. RegisterCode

RegisterCode 存储了注册码的各项信息。

| 字段名          | 含义                        | 类型     |
|--------------|---------------------------|--------|
| registerCode | 注册码                       | string |
| state        | 注册码状态 (0 : 未注册 ; 1 : 已注册) | int    |
| account      | 该注册码对应的用户名                | string |

#### 5. Picture

Picture 表保存试题的各项信息。

| 字段名         | 含义                         | 类型  |
|-------------|----------------------------|-----|
| year        | 试题年份                       | int |
| province    | 试题省份                       | int |
| subject     | 试题科目                       | int |
| number      | 试题编号                       | int |
| score       | 试题分值                       | int |
| isDelivered | 试题是否发布 (0 : 未发布 ; 1 : 已发布) | int |
| category    | 试题类型                       | int |

#### 6. Notice

Notice 表保存了站内信的各项信息。

| 字段名         | 含义     | 类型       |
|-------------|--------|----------|
| title       | 标题     | string   |
| text        | 内容     | string   |
| teacher_id  | 发布人    | string   |
| send_date   | 发布时间   | datetime |
| receive_stu | 接收学生列表 | list     |

## 7. Group

Group 表保存小组的各项信息。

| 字段名      | 含义      | 类型     |
|----------|---------|--------|
| name     | 组名      | string |
| vol_list | 组内志愿者列表 | list   |
| stu_list | 组内学生列表  | list   |

## 8. Timer

Timer 表保存了统计问卷的各项信息。

| 字段名           | 含义      | 类型       |
|---------------|---------|----------|
| teacher_id    | 发布人     | string   |
| name          | 活动标题    | string   |
| start_time    | 活动开始时间  | datetime |
| end_time      | 活动结束时间  | datetime |
| volunteer_dic | 志愿者填写信息 | string   |

## 9. WechatUrl

WeChatUrl 保存了发布推送的各项信息。

| 字段名         | 含义         | 类型     |
|-------------|------------|--------|
| title       | 推送标题       | string |
| text        | 推送摘要       | string |
| picture_url | 推送封面图片 url | string |
| message_url | 推送图文消息 url | string |

## 2.8 用户界面设计

在界面设计方面，本项目使用了 Bootstrap 的框架，并在此基础上使用了网上一个开源的主题 CSS。字体选择上，使用了谷歌和微软的字体。对于学生端的界面，则对该主题进行了适当的调整，使之能完全适应市场上大部分的手机型号。本项目的登录与注册界面又使用了另外一个开源主题。

### 2.8.1 主界面

本项目的主界面如下：





左侧的为功能导航栏，固定在左侧，点击相应的按钮进入到具体的功能界面。

上方为个人信息和个人提醒的导航栏，固定在上方，点击进入到具体的功能界面。

右下方为功能具体内容显示模块，此处只有欢迎辞和页脚。

## 2.8.2 jQuery UI

jQuery UI 是 jQuery 的一个插件，主要用于美化页面。我们的项目中使用了两个组件：datepicker 和 dialog。

我们的对话框使用的是 jQuery UI 的 dialog 组件。为了防止用户在有对话框弹出时与页面上其他元素交互，我们的项目中全部使用了模式对话框，这样可以保证同一时刻只有一个对话框实例存在。

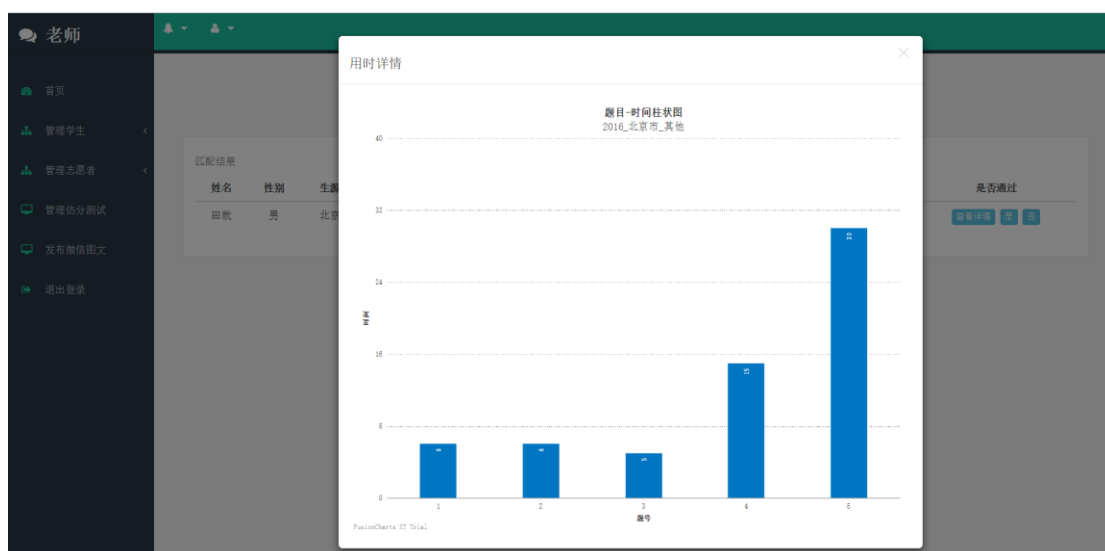


在用户选择日期的输入框中，我们使用了 jQuery UI 的 datepicker 组件来美化日历，功能也更加丰富。



### 2.8.3 FusionCharts

为了使学生估分所用时间更加形象地展示给老师，我们使用了 FusionCharts 插件，以优美的图表形式展示给老师，一目了然。



### 2.8.4 jQuery Growl

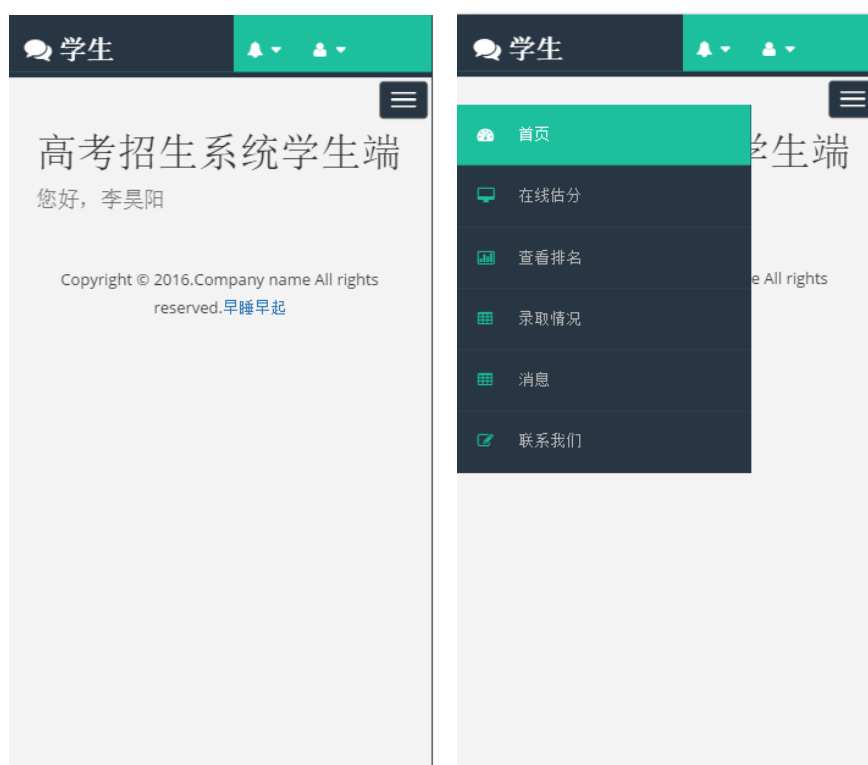
jQuery Growl 也是我们项目中使用的插件之一。JS 自带的 alert 函数可以弹出一个对话框

框对用户进行提示，但这个效果在不同浏览器上的样式不同，也谈不上美观。使用 jQuery Growl 插件后，提示信息会出现在页面右上角，在一定时间后会自动消失，用户不用频繁地点击确定按钮。这个效果有点像安卓开发中的 toast 提示。



### 2.8.5 移动端界面简介

学生的界面和老师界面大体类似，但是为了学生能够在手机上操作，我们进行了手机端的适配。后面部分的所有截图都来自手机端显示的结果。手机端上的学生用户界面也有侧边栏，但是需要通过点击右上角的按钮对侧边栏进行展开和收缩，如下图



移动端布局中，将左侧的导航栏隐藏，通过点击右上方的按钮显示。移动端布局基于 bootstrap 的响应式布局，使用 CSS3 的 Media Query 实现。

## 3 开发与评价

### 3.1 开发过程

本次软件工程项目历时九周，采用迭代开发的模式进行。一个迭代的时间为 2 周。我们每两周会进行一次大组会，验收一个迭代的成果。在大组会当天晚上小组开会制定下一迭代的计划。

我们组内将这个项目的工作分成前端和后端两部分。大致来说，所有的网页页面归前端开发，所有“跑在服务器上”的代码都归后端，包括数据库和它的各种接口等等。最初我们的分工是 3 人前端，2 人后端。后来由于中期前端工作量显著大于后端，对人员分工做出了一些微调。最终分配是邵韵秋负责微信模块和单元测试工作，白云仁、段清楠、侯禹凡三人负责前端的开发，李昊阳负责后端。

我们在开发中的一点感受是需要学习的东西太多了。网站开发有许多框架、许多库和插件，这些对我们来说都是全新的。我们之前关注的都是老师出的作业题，不曾有过这样“真刀真枪”的实践。学习使用这些工具是相当耗时的，就像在给自己补课，更不要提从若干个陌生的插件中选择一个了。这对前端开发的同学带来了很大的挑战。与此同时，后端也要学习用 python 操控 SQLite 数据库以及 excel 文件。现在我们已经完成了这个项目的开发，每个人也都从中学会了很多技能，再做同样的事情应该会熟练很多吧。

助教要求每一个迭代应该做出一些可以检查的成果，因此我们在前三个迭代中分别完成了教师端、学生端和志愿者端。最后一个迭代周期主要做一些检查、修补工作，并将项目部署到服务器上。

### 3.2 单元测试

#### 3.2.1 测试内容

单元测试部分主要是对涉及到读写数据库的函数进行测试，使用 django 自带的 unittest 模块实现。测试代码在 database/tests.py 中。主要对 database 文件夹下的 backend.py, image\_backend.py, register\_backend.py, student\_backend.py, teacher\_backend.py, volunteer\_backend.py, models.py 文件进行测试。命令行运行 `python manage.py test database` 即可运行单元测试。最终，所有单元测试均通过，结果如下所示。

```
-----
Ran 117 tests in 1.065s
OK
Destroying test database for alias 'default'...
```

(图 3.2.0 单元测试结果)

### 3.2.2 测试结果

使用 coverage 工具评测单元测试覆盖率，结果如下表所示。

| 文件名                  | 覆盖率 |
|----------------------|-----|
| Image_backend.py     | 88% |
| Backend.py           | 68% |
| Resgister_backend.py | 88% |
| Student_backend.py   | 50% |
| Teacher_backend.py   | 82% |
| Volunteer_backend.py | 49% |
| Models.py            | 79% |
| Total                | 72% |

表中所列文件是涉及读写数据库的较为底层函数的文件。从覆盖率的角度来看，对上述文件的平均覆盖率在 72%左右，部分文件的测试覆盖率较好，将近 90%，部分覆盖率不佳，仅在 50%左右。对上述覆盖率结果做出如下几点解释：

- 1) 有些函数产生的结果是随机的，无法使用该单元测试框架进行测试。如生成注册码时所用到的函数，打乱字符串的函数之类。
- 2) 部分函数涉及到的环境较为复杂，难以简单的在单元测试的环境中构造出来。如涉及到学生志愿者分组相关的函数。（这也解释了为什么 student\_backend.py 和 volunteer\_backend.py 这两个文件的覆盖率较低）
- 3) 这个项目是信息系统，保存的信息种类很多，例如学生一张表，涉及到了几十个表项，而这些表项的属性和特点有很多是相似的，所以在单元测试时没有一一进行测试，而是将其分类，选取每类中有代表性的进行测试，并进行边界检查。
- 4) 如果对考虑工程中所有文件的覆盖率，此时输出的总的覆盖率大约在 41%左右，产生这样的结果的原因是只对操作数据库相关的函数和文件进行了测试，并没有对 views.py 这类视图文件进行测试，因为这类函数的环境较难构造，以及测试输出结果不够直观，有些也难以输出，所以此项目中采用直接测试的方法，而没有进行单元测试。

### 3.2.3 测试过程

项目的单元测试主要从 sprint3 开始, 延续了 sprint3 和 sprint4 的两个迭代周期。Sprint1 和 sprint2 阶段主要是在熟悉 django 框架, 以及构造数据库表项, 前后端进行磨合协议匹配的阶段, 后端代码完成度不高, 也一直处于调整改动阶段, 所以此时并没有开始单元测试。从 sprint3 开始, 后端进度稳步推进, 单元测试也随之跟进。

我们组采用的方式是写代码和测试的人分开, 这样能增加发现问题的概率, 减少隐藏错误的可能性, 也可以换个角度思考, 思考代码的合理性。为了通过单元测试, 测试人员会尽可能的加入较小的改动, 暂时使测试通过, 并在旁边进行标注, 代码的原作者根据这些记录进行修改, 有的是直接采用修改, 有时候是重构代码, 并将修改的情况及时反馈给测试人员, 再构造测试, 重新进行测试, 如此反复, 直至通过单元测试。

在测试过程中也发现了单元测试与后端代码编写同步推进的好处, 例如在一开始测试时发现了边界检查出现了漏洞, 此类情况在之后可能出现的频率也很高, 及时发现可以在之后写代码时避免这个问题, 减少对代码的修改。

从结果来看, 虽然单元测试的代码最终是 1060 行左右, 但因为经历了多次修改再测试的过程, 所以实际的测试量要比最后显示出来的结果多很多。

## 4 问题与改进

目前我们的系统已经可以正常使用, 但我们认为本系统中还有一些值得改进之处:

### ➤ 多条件检索

目前在检索方面(教师检索学生、教师检索志愿者、志愿者检索学生)仅支持按姓名查找。这样可以完成基本功能, 但可能不够方便。在实际应用中可能有更多的需求, 比如教师希望将所有江苏考生加入一个组, 就可能需要检索所有江苏省的考生。

重要性: 比较重要, 尤其是考虑到学生数量较多的时候, 一个一个添加到学生组会相当麻烦。

难度: 不难。前端方面, 界面上只需在表单中新增输入框即可, 再加一些相应的校验条件; 后端接收条件之后, 筛选出非空的条件作为检索条件, 相当于在 SQL 的 SELECT 语句中使用复合的 WHERE 子句, 多个条件之间是与的关系。

### ➤ 批量操作

如果教师可以对学生进行批量操作, 一定会大大缩短教师用户的操作时间。目前我们支持批量添加学生到学生组。以后可以添加批量删除学生用户的操作, 或是其他扩展功能。

重要性：不重要。批量删除操作的应用场景基本在每一年招生工作结束之后，这时候需要清空学生数据。即便老师不方便删除，可以由超级管理员进行批量删除。

难度：不难。我们已有批量添加到组的操作，原理大致类似，只需要后端提供批量删除的接口。

#### ➤ 延迟加载

目前我们使用了分页的页面上，仍然是一次加载完所有信息，我们希望在用户点击跳转到某一页时，再向服务器请求该页的数据。

重要性：比较重要。在学生数量较多的时候，延迟加载可以缩短页面加载时间，给用户更好的体验。

难度：困难。目前，分页与后端完全无关，前端收到所有的数据后再进行切分，并计算出页数。用户点击跳转到某一页后也是由/前端来计算应该显示哪些数据。如果要改为延迟加载，总页数、当前页等信息可能需要后端计算并传输给前端。同时前后端必须明确每页显示的行数，如果这一参数允许用户进行设置将带来更大的复杂度。尽管实现过程可能需要达成一系列新的协议，总体来说该功能是可以实现的。

#### ➤ 分页

目前我们仅在教师检索学生的页面上应用了分页，我们希望推广到网站的其他界面上。

重要性：一般。分页可以提高用户体验。我们目前在检索学生的页面使用分页是因为学生用户可能数量相对较多。

难度：不难。这部分的工作只是重用已有的代码。

#### ➤ 一对一站内信

目前教师可以对某个学生组发送消息。如果添加了任意用户之间点到点的站内信机制，可以方便用户之间相互交流，尤其是在招生时不同的教师和志愿者可能在不同的地理位置。

重要性：一般。站内信可能只是电话、邮件、微信之余的一个补充。

难度：困难。需要新建一张表来存储站内信，需要维护发件方、收件方。界面上需要增加编辑和浏览站内信的页面，增加站内信的消息提醒。

#### ➤ 内置 Everyone 学生组

通过一个内置的 Everyone 学生组，教师给学生发通知的时候可以方便地发送给所有人，而不用先将所有人加入到一个组内。

重要性：一般。教师可以更方便地发送一个所有人可见的通知。

难度：一般。每次新建学生账户时，将其加入内置的 Everyone 学生组。每次删除一个学生账户时也将其从 Everyone 学生组移除。Everyone 学生组不可由用户编辑，因此在查看学生组的页面需要对其进行特判，不允许用户向 Everyone 学生组增删成员。

#### ➤ 消息提醒

目前消息提醒在二级菜单内，用户必须先点开才能看到提醒。我们应该增加更明显的提示方式（例如有新的提醒时让按钮图表变红）。

同时，我们目前只在加载页面时检查有无新的提醒。如果用户在一个页面停留太久，他将收不到这一期间的提醒。我们希望用户即便没有切换页面也能及时收到提醒。

重要性：比较重要。提醒机制直接影响用户体验。

难度：一般。前一个需求可以通过在页面上用 JS 进行判断。后一个则可以通过引入定时器并定期向后端查询新的提醒来解决。

#### ➤ 使用 https 协议传输

使用 https 替代 http 可以给网站带来更强的安全性。

重要性：非常重要。招生系统含有大量的重要、敏感信息，信息安全十分重要。

难度：困难。准确地说对此方面我们的了解还很少，预计可能需要修改 Django 的设置和部署配置，同时也要打开服务器的相应端口。

## 5 下一步工作的方向

虽然我们力求最大限度地给使用这个系统的用户以最佳的体验，但是我们还是客观地认为，这个项目依然有可以改进的地方

#### ➤ 优化数据库结构设计

目前数据库中表的设计与排布可能会在保存某些用户信息的时候效率略显低下，在用户较多或者操作大量信息的时候存在着等待时间略长的现象，所以之后可以尝试对表的设计进行优化，采用更多合理的高级设计。

#### ➤ 进一步加强安全性

目前用户的账户名和密码是明文发送和存储的，虽然我们做了较多权限认证方面的工作，但仍存在着被黑客攻击入侵的危险。可以使用 https 替代 http 可以给网站带来更强的安全性。

#### ➤ 进一步加强系统内各级别用户的信息沟通



目前系统支持发送站内信的功能，考虑到功能更多的是对一个组发信，所以目前系统并不支持对单人发信，后面可以继续增加这方面的功能。

➤ 加强异步和延迟操作的功能

在用户量大或者数据库内信息量大的情况下一一次性统一把数据返回给用户是比较费时的，在网络差的情况下更是如此，后续可以增加延迟加载机制只返回当前用户需要的信息。

➤ 增加检索多样性与灵活性

目前系统仅支持根据用户姓名的检索，后续可以加入多条件检索，比如按照性别，省份等。

➤ 增加分页功能

当考生数量大时分页可以增强老师的用户体验，避免出现所有考生信息集中在一页上的情况。

## 6 收获

经过两个多月的开发，我们最终完成了自己的项目。在这个过程中，我们也有很多收获。开发的成果固然十分重要，而我们从这过程中学到的，也同样弥足珍贵。

### 1) 学会了熟悉使用 git

《软件工程》这门课第一个小作业就是在计蒜客上学习 git，其实以前也简单接触过 git，不过对它的理解基本停留在“保存代码，方便版本迭代”这个阶段。在真正开始写项目之后，我们才意识到了 git 对一个团队合作开发项目的重要性，每个人在项目中分别负责不同的模块，可以在主分支上开出不同的分支分别进行版本迭代，之后将所有的分支合并到主分支上即可，非常便于管理。有时候发现某次修改出了问题，又可以通过 commit 信息方便的回滚到之前的版本。可以说，正是因为有了 git，我们的项目开发才得以如此高效，版本控制才如此合理。

### 2) 认识到了团队合作的重要性

之前课程的大作业更多的是个人大作业，需要团队合作的作业很少，而这次软工大作业让我们充分认识到了团队合作的重要性。也许有些时候自己一个人完成比将任务分配下去分别完成再整合要快速得多，也许有时候如何进行项目的划分让更多的人参与到开发中对我们来说是一个头疼的事，但是我们必须意识到，实际的开发项目必定是多人合作完成的，我们

有必要提前接触并适应这种开发模式。

### 3) 养成了良好的代码习惯和代码风格

软工项目的代码量很大,如果不能在一开始保持很好的代码风格,将会给后期的维护带来很大的困难。尤其是 Tab 和空格如果混用会对 Python 代码造成很大的影响。我们始终致力于保持良好的代码风格。在本次开发过程中,我们也取得了很多经验和教训。

### 4) 加深了对 Django 开发框架的理解和运用能力

在开发时选择 Django 框架,是因为我们之前大一夏季学期曾简单学习过它的使用,但是后来才发现框架还有很多我们不了解的地方。经过这次开发之后我们对这个框架的使用更加熟悉,对 Django 的配置和部署等也更加熟悉。

### 5) 较为熟练地掌握了前端开发的一些知识

之前我们做的网页基本上都是只有功能但并不美观,但是一个软件的外观其实对用户体验也有不小的影响。因此在这次完成项目的过程中我们学习并使用了 Bootstrap, JavaScript 脚本,以及 Ajax 等可以增强用户体验的前端技术,让前端页面既美观又好用。

### 6) 充分锻炼了快速开发的能力

本次的软工项目由于是开发,每个迭代只有一两周的时间。这要求我们将开发计划明确到每一天,对我们快速开发的能力提出了很高的要求。经过这次项目,我们充分锻炼了快速开发的能力,学到了很多贴近实际工作的专业技能。

### 7) 提升信心

在上这门课之前,我们并不知道自己是否有能力完成这样大型、综合的开发项目,然而在真正做起来之后,我们才意识到其实自己是具有这方面的水平和潜力的。如果说之前的收获是知识水平上的提高,那么这个收获就是信心上的提升。

### 8) 增强学习能力

本次项目中,我们接触了很多新知识、新技能。这些都是课堂上没有讲过的。然而为了我们项目的良好效果,我们投入了很多时间学习各种框架、库、插件的使用。在这个过程中我们阅读了很多教程和文档,比如 JavaScript 教程、Django 1.10 Documentation、PyUnit 文档、微信开发者文档,还有各种插件的官方教程与示例。起初我们感觉学习新知识很难,尤其是没有老师教,全凭自己摸索。后来我们在网上学习新知识的能力越来越强,一个新的插件花不了几小时就能学下来了。互联网是最好的老师,我们今后还会遇到很多新事物,这种自学能力对我们来说是十分必要的。

老师上课时曾讲过软件工程的核心就是对于软件开发的 5 个重要组成部分：需求分析，设计，编码，调试，维护，如何组织这 5 个部分的工作，以及如何完成每一个工作。开始我们对软件工程处于一知半解的状态，分工也比较混乱。在划分模块后明确了各自分工，形成了良好的前后端交互模式，渐渐形成良性循环。在学习过程中，知道了团队合作十分重要，争议固然存在，但通过讨论、协商，群策群力，在不断磨合中能够达成一致与默契。团队成员中能力各有高下，各取所长，不宜妄自菲薄。组长多加协调，组员积极配合，才能合作愉快。学习能力体现在能尽快接受新的知识，顺应变化，学为所用。

正如我们的用户代表李山山老师评价道，我们的项目是所有项目中最具系统性和完整性的一个，涉及到复杂的权限管理以及不同身份之间用户的交流，在开发过程中需要一一处理。在这次大作业实践中，我们将白老师课堂上软件开发理论与自身实践相结合，在不断磨合中有所感悟、有所成长、有所收获。我们觉得，这必定会成为大学生活中最难忘的一门课程，也将成为我们人生中难忘的一次经历。

## 7 致谢

感谢白晓颖老师在我们整个项目中给予的教导、支持与帮助。感谢余翔助教以及用户方代表李山山老师耐心地与我们沟通，每周定时与我们交流开组会。正是在老师和助教的支持与帮助下，我们才得以顺利完成这个项目。同时也感谢同一项目内其他组同学在开发过程中给予我们的启发和帮助。