重构学生关怀系统计划

目的及时间点

2017年8月5号截止，需要能做出一些东西，可以应付下学校反复催促的需求。

计划

2017-7-20 – 2017-7-22

1. 梳理清晰当前学生关怀系统数据库现状，理清楚到底有哪些数据字段可用、可用的字段现状（是否有None，是否可靠）
2. 梳理清晰当前学生关怀系统真正核心的功能有哪些，优先将这些功能完成（边缘或者不那么重要的功能延后）

2017-7-23 – 2017-7-24

1. 定制前端、后端开发框架。
    - 前端开发框架主要为文件结构（特别是公共组件文件、局部页面文件架构）、交互API
    - 后端开发框架包括类设计、数据库orm设计、.py文件结构设计等
2. 针对每一个交互api，制作对应的.json文件，供前后端开发使用

2017-7-25 – 2017-7-29

1. 前后端同时开发，完成代码开发任务
   （我先做后端开发，完成后迅速投入到前端开发中）

2017-7-30 – 2017-8-5

1. 测试，迅速解决可能影响软件正常使用的功能缺陷
2. 做演示准备

进展

2017-07-21

规划

本次重构需要优先完成的页面：

1. 首页
    - 学院重点关注学生增长量（两个图：折线、柱状）
    - 首页下面的列表，其具体项可与永信的一致（最后一类 只有放大镜）
2. 个人页
    - 在主页上单击可进入个人页。
    - 个人页包含（不做特殊说明，均可与永信保持一致）：
        - 学生个人信息
        - 重点关注–理由
        - 该学生成绩统计
        - 该学生出行记录
        - **该学生一卡通消费记录，表项改成：商户名称、金额（只留下消费金额，其他全都屏蔽）、交易时间、操作类型**
        - 心理状态测试（直接剔除掉）
        - 该学生时间记录
    - 个人页其他帮助
        - 新建关注、取消关注、新建事件均包含
        - 事件线（直接去掉）
3. 办公页面
   数据导出

> 这部分比较麻烦的是筛选部分，筛选部分先不管，只允许用户导出所有内容。

意见反馈

4. 系统管理
    - 用户组管理

> 罗列出当前的所有用户组名称、描述（一段话）、更改权限按钮、删除按钮。
> 单击更改权限按钮后，展示一个小界面，允许勾选该用户组所能查看的班级（可能之前已经勾选过，所以这个小界面上有的勾已经勾选过了）

    - 角色组管理

> 罗列出当前的所有角色组名称、描述（一段话）、更改权限按钮、删除按钮。
> 单击更改权限按钮后，展示一个小界面，允许勾选该用户组所能查看的页面（可能之前已经勾选过，所以这个小界面上有的勾已经勾选过了）

- 用户管理

> 罗列出当前所有用户名称、描述（一段话）、更改按钮、删除按钮。
> 单击删除，需要弹出确认框，确认后，向后端发送删除命令。
> 该页面上还有添加用户的功能（应是一个比较大的按钮），单击添加，弹出一个框。框体内让输入一个用户名、密码、确认密码，还有两个选择框，分别让选择该用户对应的用户组、角色组。单击确定按钮执行校验，确保均已输入及选择，向后端发送添加命令。

新增页面

5. 数据维护
    - 学生基本数据增量导入
    - 学生成绩数据增量导入

> 上述页面上都只有一个按钮加一个上传控件。按钮单击后导出我们的excel模板，上传的文件也是一个excel文件，该excel文件的数据交给后端进行详尽的纠错，返回success或者fail+错误信息

其他重要功能：

1. 设定用户组功能，用户组内的用户登录，能看到其对应的数据权限。
2. 设定角色组功能，角色组内的用户登录，能看到其对应的权限的页面。

会议内容

1. 技术难点讨论、交流
2. 前端架构
    - 按照做恶意域名分析那一套，以页面划分文件结构
    - 建立common文件夹，将所有可复用的.vue放在里面
    - 建立api文件夹，所有需要与后端交互、JS功能类的函数，均定义在该文件夹下
    - 按需建立sass文件，将可复用的颜色、位置等定义成函数
    - Vue文件的命名统一为驼峰式
3. 后端架构
    - 按照数据库具体数据字段，建立orm.py
    - 建立api_define.py，将所有与前端交互的API定义写在这里。
    - 按照前端页面，建立文件夹。在文件夹下按需建立.py文件实现该页面下的所有功能需求，API对接的class需要继承api_define中的对应class
    - 可复用的功能，按需定义实现于common_func.py中

- 后端需要使用logging模块，将

4. 交互API、json数据文本
5. 任务分配及时间节点

纪要：

1. 技术难点讨论、交流
2. 前端架构

    - 按照做恶意域名分析那一套，以页面划分文件结构
    - 建立common文件夹，将所有可复用的.vue放在里面
    - 建立api文件夹，所有需要与后端交互、JS功能类的函数，均定义在该文件夹下
    - 按需建立sass文件，将可复用的颜色、位置等定义成函数
    - Vue文件的命名统一为驼峰式

3. 后端架构

    - 按照数据库具体数据字段，建立orm.py
    - 建立api_define.py，将所有与前端交互的API定义写在这里。
    - 按照前端页面，建立文件夹。在文件夹下按需建立.py文件实现该页面下的所有功能需求，API对接的class需要继承api_define中的对应class
    - 可复用的功能，按需定义实现于common_func.py中
    - 后端需要使用logging模块，将

1.交互API、json数据文本
一、首页
API:
XX/index/grow-line:

XX/index/grow-bar

XX/index/focus-table

二、个人页
API:
XX/person/static-info

XX/person/trip

XX/person/card

XX/person/cancel-focus

XX/person/add-focus

XX/person/add-event

三、办公页面
XX/office/export
XX/office/suggestion

四、系统管理
XX/system/get-total-user-team
XX/system/get-one-user-team
XX/system/set-one-user-team
XX/system/del-one-user-team

XX/system/get-total-role-team
XX/system/get-one-role-team
XX/system/set-one-role-team
XX/system/del-one-role-team

XX/system/get-total-user
XX/system/get-one-user
XX/system/set-one-user
XX/system/del-one-user
XX/system/add-one-user

五、数据维护
XX/data/update-basic
XX/data/update-score

其他：
登录：XX/login/if-pass
Session: XX/login/session

json样例文件已当场搞定。

任务分配及时间节点：

1. 技术盲点
    - 上传文件及下载文件（下载模板、上传excel）
    - 页面hidden
    - 针对上传数据的数据清洗、数据规整判断
    - 用户组权限
    - 加解密安全问题
2. 其他任务：
    - 前端页面开发任务
    - 后端功能开发任务
    - 交互测试
3. 要点：
    - 重视权限的处理，不允许看到的页面，不能通过手工在浏览器上敲网址跳转

具体安排如下：
上传、下载（前后端均包含）以及数据规整判断（后端）交给zzh做
后端框架建设、开发交给zc和lwj做
前端框架建设、开发、hidden判断交给jyc做
期间任何问题、困难随时沟通，复杂问题交给zzh断后。

第一次交流时间：2017-07-24

至少完成的任务：前后端框架完成。

第二次交流时间: 2017-07-26

至少完成的任务：基础开发任务完成。

第三次交流时间：2017-07-28

至少完成的任务：重、难点攻坚完成，系统可运行。

2017-07-22

1. 前端上传控件代码：

```
            <el-upload class="upload-demo" drag action="http://127.0.0.1:
8001/test" multiple show-file-list name="file">
                <i class="el-icon-upload"></i>
                <div class="el-upload__text">将文件拖到此处，或
                    <em>点击上传</em>
                </div>
            </el-upload>
```

2. 前端将列表table内的内容转成excel下载到本地的方法：
   通过jquery的table插件实现，教程：http://www.cnblogs.com/Alex80/p/6371750.html
3. 前端通过后端给的json转成excel下载到本地的方法：
   http://blog.csdn.net/educast/article/details/52775559

2017-07-23

关于用户部分的三张表：

1. 用户表
   字段：
   - id #用户编号，自增int
   - username #用户名，varchar
   - userpass #用户密码，varchar
   - userteamid #该用户对应用户组id

- userroleid #该用户对应角色组id

2. 角色组表

    - id #角色组编号
    - permission #该角色组权限，varchar

3. 用户组表

    - id #用户组编号
    - permission #用户组权限，varchar

```
class MyBaseModel(Model):
    class Meta:
        global my_database
        database = my_database

    @classmethod
    def getOne(cls, *query, **kwargs):
        """
        为了方便使用，新增此接口，查询不到返回None，而不抛出异常
        """
        try:
            return cls.get(*query,**kwargs)
        except :
            return None

    @classmethod
    def returnList(cls,Model = None, key = None):
        """
        将结果返回成一个列表嵌套字典的结构返回
        """
        if not type(Model) == SelectQuery:
            return None
        list = []
        for con in Model:
            if type(con) == dict:
                if not key == None:
                    list.append(con[key])
                else:
                    list.append(con)
            else:
                list.append(to_dict(con))
        return list

class new_users(MyBaseModel):
    username = CharField(null = True)      #用户名，varchar
    userpass = CharField(null = True)      #用户密码，varchar
    userteamid = IntegerField(null = True)    #该用户对应用户组id
    userroleid = IntegerField(null = True)    #该用户对应角色组id

class new_user_role(MyBaseModel):
    userrolename = CharField(null = True)
    permission  = CharField(null = True)      #角色组权限，varchar

class new_user_team(MyBaseModel):
    userteamname = CharField(null = True)
    permission  = TextField(null = True)      #用户组权限，text
```

关于用户部分的公共函数：

```
getTeam(userid):
"""给一个用户编号，函数返回该用户对应的角色组权限""

return {
        "index": 0,
        "person": 0,
        "officeExport": 0,
        "officeSuggestion": 0,
        "dataUpdateBasic": 0,
        "dataUpdateScore": 0,
        "systemUserTeam":0,
        "systemRoleTeam":0,
        "systemUsers":0,
    }

getUserRole(userid):
"""给一个用户编号，函数返回该用户对应的用户组权限"""

return {
    "1101401":0,
    "1104102":1,
    ……
}

getTotalClass():
"""函数返回当前用户组可能存在权限，即：返回当前数据库内所有班级"
return ["1104101", "1104102", "1104103"]

getTotalPage():
"""函数返回当前角色组可能存在权限，即：返回当前系统的所有页面名"
return ["index", "person", "officeExport"]

judgeIfPermiss(userid, stuid = None,  page = None, mode = 0):
"""
mode为0时：给一个用户编号加学生学号，函数通过该用户对应用户组的权限，返回True或者False，True代表该用户拥有该学生的权限，False代表该用户没有权限操作该学生，如果输入的学生id或者userid不存在，函数仍返回False
mode为1时：给一个用户编号加页面名（'index'、‘person’、‘officeExport’等），函数通过该用户对应角色组权限，返回True或者False，True代表该用户可以看该页面，否则表示不可以。如果输入的页面名不存在，函数仍返回False
"""
return True
```