凯里学院”智慧微校园“项目开发文档

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文档修改记录 | | | | |
| 序号 | 日期 | 修改人 | 修改内容简介 | 备注 |
| 001 | 2018.11.03 | 李韦鹏 | 创建文档 |  |
| 002 | 2018.11.12 | 李韦鹏 | 开始写接口标准 |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[凯里学院”智慧微校园“项目开发文档 1](#_Toc529346612)

[1项目简介 3](#_Toc529346613)

[2需求分析 3](#_Toc529346614)

[2.1 学生服务 4](#_Toc529346615)

[2.2 教师服务 5](#_Toc529346616)

[2.3 主管服务 6](#_Toc529346617)

[2.4 实验室/实训室管理 6](#_Toc529346618)

[2.5 校园服务 7](#_Toc529346619)

[2.6 后台数据管理 7](#_Toc529346620)

[2.7 其他 8](#_Toc529346621)

[3项目架构与细节澄清 8](#_Toc529346622)

[3.1 前端微信小程序 8](#_Toc529346623)

[3.2 后端 8](#_Toc529346624)

[3.2.1 基本解决方案 8](#_Toc529346625)

[3.2.2 数据库设计 8](#_Toc529346626)

[3.2.3 数据智能分析 9](#_Toc529346627)

[3.2.4 分布式数据库 9](#_Toc529346628)

[4团队成员与分工记录 9](#_Toc529346629)

[4.1 前端 9](#_Toc529346630)

[4.2 后端 9](#_Toc529346631)

[5开发计划 9](#_Toc529346632)

[6 项目进度记录 9](#_Toc529346633)

# 1项目简介

凯里学院“智慧微校园”项目由之前的省级科研项目“高校微校园智能化云平台”转化而来。项目初衷是为我校打造一个服务师生校园工作、学习和生活的移动平台。实事求是地讲，本相比发达地区高校，我校目前的信息化建设是很落后的，因此本项目开工的目的无外乎两个：

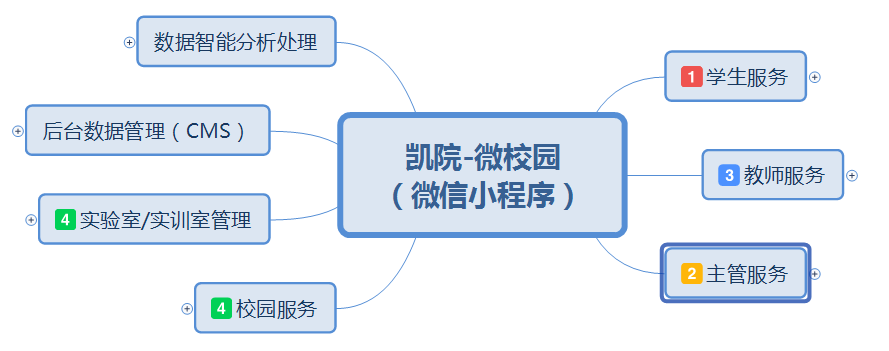
(1)不再依靠外包买软件卖服务，根据学校师生的实际需要，创建自己的校园服务平台。

(2)锻炼学生开发团队，让喜欢学习技术，钻研技术的同学有一个实战的机会和目标，重点是希望通过本项目的开发，建立起一支有年级梯队的学生队伍，为今后校企合作以及学校教改转型提供抓手。

# 2需求分析

总体原则：先满足学生需求，再同时满足教师需求和管理者关心的数据呈现于上报需求，最后是其他通用服务和易用性提升。

一句话：先对事后对人。需求大致划分见下图



整个项目的需求分为7个部分：

（1）学生服务 （2）教师服务 （3）主管服务 注：这三大服务是项目初期的工作核心

（4）校园服务（通用类服务集合）（5）实验室/实训室管理（分院特色）

（6）后台数据管理（CMS） （7）数据智能分析处理（星辰大海的梦想）

## 2.1 学生服务



学生服务核心思想是：帮助学生成长，方便学生学习和生活，因此我们划分功能为大致4个方面：

（1）成长关注。这里涵盖了学生的学习情况（包括理论和实操技能）、成绩（考试安排和成绩查询）、比赛、获奖与证书、社团、实习/就业/创业，以及好书推荐等诸多方面。

（2）申请服务。这里主要是指学生的申请，其中

* 请假：放在学生界面专栏中的，同辅导员/班主任版面的信息有数据关联，相关数据也会反应在主管特有界面。
* 借用设备：关联到实验室管理模块和设备管理模块，方便学生申请设备使用

（3）个人任务追踪：因为很多时候学校要求统计学生各类数据，但是学生却有自己的事情需要优先完成，必然产生矛盾；由此我们将

1）开学报到、放假到家（或目的地）报平安

2）上课签到/会议签到（这个功能其实是整合在校园服务中的，可以理解为综合的“扫一扫”）

3）重要消息阅读（阅读后自动标记，不会在群里一堆“收到”，反复确认回复）

4）民主评议投票

等需要统计数据（特别是实时上报和分阶段上报）场景都交给平台去有序管理，避免了重复低效工作，核心思想：有规律的东西都创建标准的流程，让流程的制度化科学管理逐步替代低效的人工管理，帮助教师腾出精力和资源去做更有意义的事情。

（4）个人发展目标与综合素质测评。实事求是地讲，美国一流大学（哈佛、斯坦福，甚至密西西比大学等知名学府）早在10年前就已经开始采集和分析学生在校期间的各项发展数据的了。但与我们现阶段不同的是：他们沿用的是“默默在背后悄悄的采集”的工作策略，这导致整个10年中没有师生抱怨，反而很配合这项工作。我认为这是我们这个项目应该学习的，但是实际情况比较复杂，当然具体问题具体分析，渐进式改革嘛。

个人发展目标是让学生自己编辑的；而综合素质测评，除了需要辅导员/班主任在后台CMS中录入的评价，其他数据尽量从教务系统抓取；学生获奖信息需要学生自己上传到CMS中。这样一来开发团队的工作量和学生科的统计工作量都能得到极大减轻，师生抱怨理应减少。

在实际施工中采用“雷达图”的多维度描述，也是参考国内信息化建设走前沿的高校开发经验而来，也是符合后期数据分析、数据展示通用标准的，尽量不改动。

## 2.2 教师服务

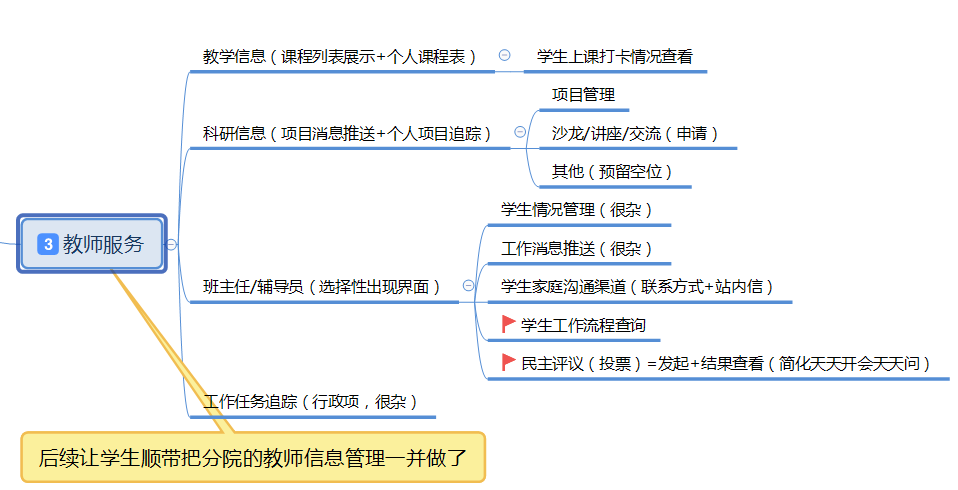


图2-1

教师服务就很简单了，但是在建立数据库表格的时候，我们建议尽可能将教师信息健全，方便后续平台在教师档案管理中发挥作用。教师服务主要包含

（1）教学信息。教师课程表，附带查看当前上课学生打卡情况。

（2）科研信息。主要是科研信息的推送，包括项目信息、讲座等活动的申请，这块可以后续完善，非平台初期核心功能。

（3）班主任/辅导员功能。这里的事物非常繁杂，很可能在项目中后期反复修改，且会有大的改动。而学生工作流程和民主评议两个功能比较常用，且容易实现，优先完成：

1）学生工作查询，就是将现在又的涉及到学生事物的工作流程（如奖学金、毕业条件、授予学位条件、休学申请等等相关的工作标准和流程）以卡片的形式呈现出来，方便学生在掌上查看。这样就能减少学生反复去学生科确认消息的情况。但是这里需要注意一个情况：相关信息的更新和录入是后期维护工作中较为繁重的任务。

2）民主评议。也就是常说的网络投票，这些投票结果是会保存在服务器上的，这些数据也可以提供给分管学生工作的其他主管看到。

## 2.3 主管服务

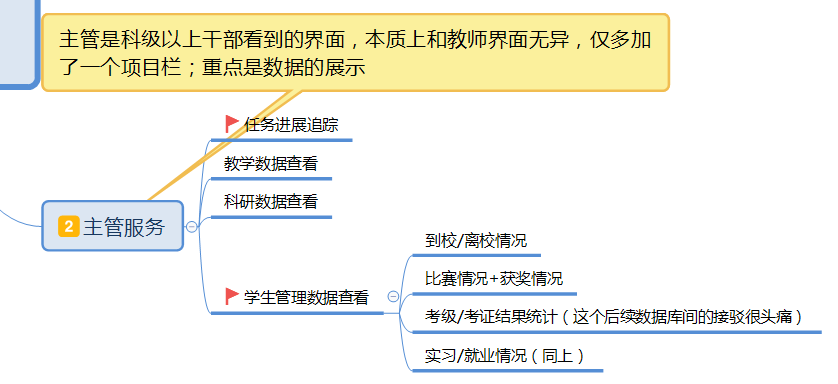


图2-2

如图，主管服务的核心无外乎两点：

（1）任务进展追踪。这个功能是后期需要联合很多模块攻克的，属于另一个小项目，项目初期可以先标记为//TODO。任务追踪是很多OA软件的必备功能，咱们慢慢来实现。

（2）各类管理数据的查看。作为一款掌上平台，我们的微信小程序是重要的数据展示窗口，在实现此需求时，任务的难点可能在于采取何种方式展现数据让用户体验更加舒适。

## 2.4 实验室/实训室管理

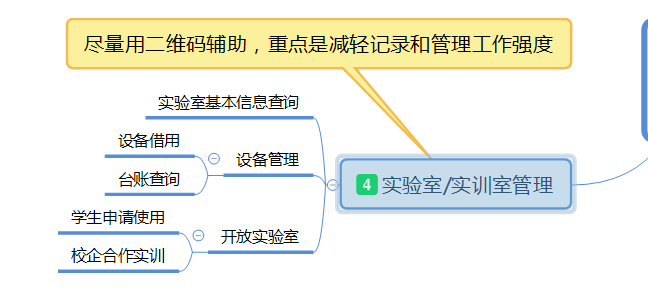


图2-3

该模块主要功能如下：

（1）实验室信息的存储于展示，即实验室基本信息查询。

（2）实验室设备管理，即分院自己的设备数据库，携带设备借用流程功能。

（3）开放实验室管理，同上主要包含申请流程。

实际上实验室在管理方面比较容易实现，主要是“使用留痕”，方便档案建设；另外我们将通过条形码和二维码扫描用于清点实验设备。

## 2.5 校园服务



图2-4

校园服务的范畴很大，项目前期主要解决扫码问题（扫码签到）、普通咨询推送、流程办事指南等基础功能；后期重点解决消息的“精准推送”问题，需要依靠后台的算法来实现。

## 2.6 后台数据管理

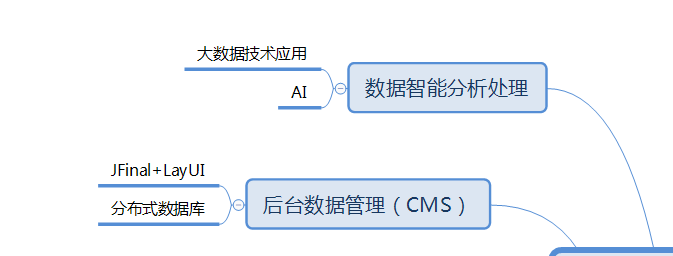


图2-5

借助JFinal和LayUI开源项目搭建我们微校园小程序的数据管理后台，将各种数据有效管理（分布式数据库）和分析。

## 2.7 其他

我们还在项目后期规划了数据智能分析和处理模块，这个模块主要借助大数据和人工智能相关技术解决实际的数据统计和分析问题。

# 3项目架构与细节澄清

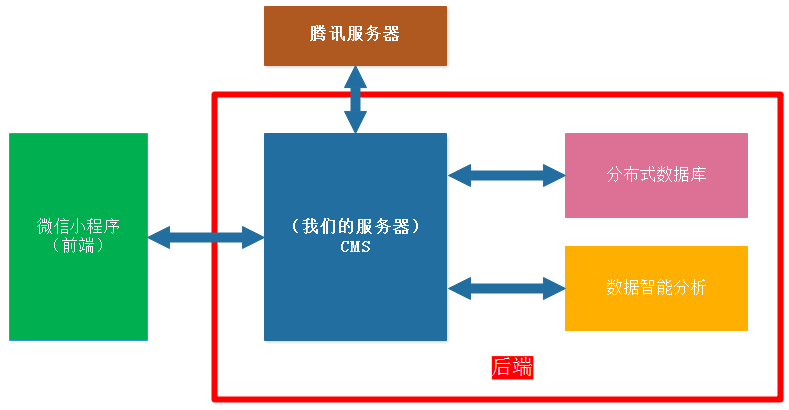


图3-1

上图为总体的项目框架简图，项目第一个版本发布后会再完善。

## 3.1 前端微信小程序

## 3.2 后端

## 3.2.1 基本解决方案

## 3.2.2 数据库设计

（1）权限控制和用户

表3-x 用户表 user\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | user\_id | id | INT | 10 | 是 |
| 2 | num | 编号 | INT | 10 |  |
| 3 | type | 种类 | INT | 2 |  |
| 4 | level | 等级 | INT | 1 |  |

初步设想用户权限控制如上表，type用于区分是教师、主管、校工、管理员等不同角色，

level可以限定当前用户能够查看和修改的数据的权限。

（2）学生信息实体和相关数据结构

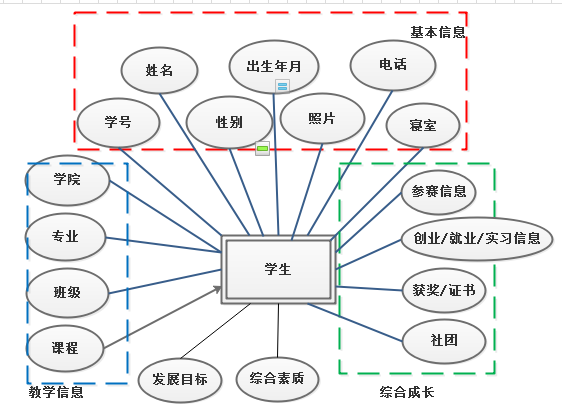


图3-x

如图，相关表项和类型说明见学生信息表（stu\_info）

表3-x 学生信息stu\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | stu\_no | 学号 | INT | 10 | 是 |
| 2 | name | 姓名 | VARCHAR | 10 |  |
| 3 | sex | 性别 | TINYINT | 1 |  |
| 4 | birth | 出生年月 | DATE |  |  |
| 5 | tel | 电话 | BIGINT | 13 |  |
| 6 | room | 寝室 | VARCHAR | 15 |  |
| 7 | img | 照片 | MEDIUMBLOB |  |  |
| 8 | school | 学院 | INT |  |  |
| 9 | major | 专业 | INT |  |  |
| 10 | class | 班级 | INT |  |  |
| 11 | target | 发展目标 | VARCHAR | 200 |  |

target字段是学生的发展目标

课程信息需要专门列表：

表3-x 课程信息 course\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | cou\_no | 课程号 | INT | 10 | 是 |
| 2 | name | 课程名称 | VARCHAR | 50 |  |
| 3 | hour | 学时数 | INT | 3 |  |
| 4 | score | 学分 | INT | 1 |  |
| 5 | stu\_num | 选课人数 | INT | 3 |  |
| 6 | tea\_no | 任课教师 | INT | 10 |  |
| 7 | room | 上课教室 | INT | 10 |  |

考试成绩的查询需要专门再生成新表

表3-x 课程信息 score\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | score\_id | ID | INT | 10 | 是 |
| 2 | cou\_no | 课程号 | INT | 10 |  |
| 3 | stu\_no | 学号 | INT | 10 |  |
| 4 | score | 分数 | INT | 3 |  |

这样一条一条的查询可能比较慢，但是存储的时候可以讲一个科目的成绩按照学号连续存储一小段，还需要进一步优化。

教室信息单独列表

表3-x 教室信息room\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | r\_no | 房间编号 | INT | 10 | 是 |
| 2 | name | 教室名 | VARCHAR | 20 |  |
| 3 | region | 校区 | INT | 3 |  |
| 4 | capacity | 容量 | INT | 3 |  |
| 5 | type | 种类 | INT | 2 |  |
| 6 | status | 状态 | INT | 2 |  |
| 7 | latitude | 纬度 | DECIMAL | M=10 D=5 |  |
| 8 | longitude | 经度 | DECIMAL | M=10 D=5 |  |

对于type和status两个属性要做说明：

（1）room实质上不仅仅是教室，还可以是实验室、实训室、活动室，所以通过type来设置不同的类型。

如：0--教室 1--实验室 2--实训室 3--活动室

也可以用占位方法标记 000--教室 001--实验室 010--实训室 011--活动

这种方式可以快速的解析，还能丰富的添加其他信息，但是数据库中就要变成varchar类型存储了，

反而检索变慢。

（2）status能代表当前教室的状态

0--教室空闲 1--教室上课 考虑这里还需要标记是否有保修情况

此外还有一个很重要的属性就是位置信息，即经纬度，根据资料

|  |  |
| --- | --- |
| 赤道周长（米） | 度数（度） |
| 40076000 | 360 |
| 111322.2222 | 1 |
| 11132.22222 | 0.1 |
| 1113.222222 | 0.01 |
| 111.3222222 | 0.001 |
| 11.13222222 | 0.0001 |
| 1.113222222 | 0.00001 |
| 0.111322222 | 0.000001 |
| 0.011132222 | 0.0000001 |

得到当前设置的经纬度采用DECIMAL(10,5)即可，我们只用推算到10米内打卡

参赛信息如下：

表3-x 竞赛信息competition\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | com\_id | 比赛id | INT | 10 | 是 |
| 2 | name | 赛事名称 | VARCHAR | 40 |  |
| 3 | description | 赛事信息 | TEXT |  |  |
| 4 | name\_list | 参赛人员 | TEXT |  |  |

竞赛信息中，赛事信息是需要后台调整显示格式的，而参赛人员采用如下方式编辑：

同学1 学号1，同学2 学号2，....每个学生之间用逗号隔开，在项目初期我们就这样约定，后续可能还要改进。

创业、就业、和实习信息，按照预想也是和学校现有系统的数据库对接的，但考虑到实现这个目标比较难，我们暂时采用如下表格逐一存放记录，每一个专业、班级按照学号大小连续存储。

表3-x 实习统计表 internship\_job\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | job\_id | id | INT | 10 | 是 |
| 2 | stu\_no | 学号 | INT | 10 |  |
| 3 | content | 内容 | TEXT |  |  |
| 4 | date | 日期 | DATE |  |  |
| 5 | type | 种类 | INT | 1 |  |

其中type用来区分是实习信息还是就业信息，这样节省一张表。

获奖以及证书信息

表3-x 获奖证书信息 award\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | award\_id | id | INT | 10 | 是 |
| 2 | stu\_no | 学号 | INT | 10 |  |
| 3 | content | 内容 | TEXT |  |  |
| 4 | date | 日期 | DATE |  |  |
| 5 | img | 图片 | MEDIUMBLOB |  |  |
| 6 | type | 种类 | INT | 1 |  |

其中type用来区分是获奖证书还是职业技能证书

表3-x 社团信息 community\_info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 属性名 | 中文描述 | 类型 | 宽度 | 是否主键 |
| 1 | com\_id | id | INT | 10 | 是 |
| 2 | stu\_no | 学号 | INT | 10 |  |
| 3 | content | 内容 | TEXT |  |  |

至此，除了消息推送相关的信息，基本上所有的学生服务涉及的属性都设计完成了。综合素质测评相关项目，有待学校的规则稳定后再添加。

## 3.2.3 数据智能分析

## 3.2.4 分布式数据库

## 3.3 业务流程

表3-xxx

|  |  |  |
| --- | --- | --- |
| 业务编号 | 业务名称 | 简要描述 |
| 100xx | Xxx | 学生操作函数 |
| 10001 | 用户登录 | 验证用户身份，无需显式区分用户种类，返回用户真实姓名。 |
| 10002 | 用户登出 | 当前用户登出，服务器端记录，本地小程序清空缓存。 |
| 10003 | 查看资讯 | 点击底端的“资讯”tab页，进入资讯列表界面 |
| 10004 | 成长信息 | 点击首页中成长栏后进入流程，项目较多后续考虑 |
| 10005 | 阅读 | 点击首页中看书栏进入流程，展示推荐书籍 |
| 10006 | 事务 | 点击进入学生事务相关申请环节 |
| 10007 | 课表 | 个人专属课表，分单双周显示，有调课后显示 |
| 10008 | 竞赛 | 根据学生所在分院、专业、技能以及特长，智能推送竞赛信息 |
| 10009 | 实习/就业 | 向求职求实习学生推送求职信息链接，后期考虑智能推送 |
| 10010 | 社团 | 放置学生所在社团的简单信息 |
| 10011 | 成绩查询 | 查询各类考试成绩查询，后期考虑加入资格证考试成绩查询 |
| 200xx | Xxx | 教师操作函数编号 |
|  |  |  |
| 300xx | Xxx | 主管操作函数编号 |
|  |  |  |
| 400xx | Xxx | 校工操作函数编号 |
|  |  |  |
| 500xx | Xxx | 所有用户通用行为函数编号 |
| 50001 | 扫一扫 | 通用扫描入口（签到，教室/实验室信息查询等等） |

业务详情（每个业务有对应编号，业务所属函数也有编号）

* 【10001】用户登录



微信小程序访问函数10001-1

* + 函数名称 **userLogin( )**
  + 访问模式 post
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx', //左边key，右边value**

**password : 'xxxxx'**

**}**

* + 服务器返回数据Json形式

**data : {**

**user\_name : 'xxxx', //用户真实姓名**

**user\_type : 'yyyyy' //用户种类：教师、学生、主管？**

**}**

（1）用户在手机上用微信小程序登录，用post模式上传账号和密码（**userLogin( )**）

（2）微校园服务器端验证账号和密码正确性，并判断当前用户类型。

比如：0--学生 1--教师 2--主管 3--校工

从目前对学生学号和教师工号（以后简称“学工号”）的梳理，可以写一个简单的判断方法区分两种用户，主管用户单独列表判断；小程序中通常使用的都是user\_id。

（3）小程序获取到用户登录是否成功的验证信息，并根据用户类型加载各类用户对应的界面，这样就不用在登录时要求用户进行类似“勾选学生或者教师”的操作了，提高用户体验；

在显示信息时需要使用真实姓名，所以有3个主要的登录数据缓存本地：

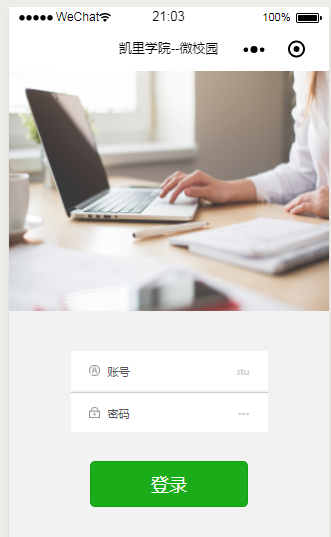
user\_id 常用的用户识别码，即学工号

user\_name 用户真实姓名

user\_type 用户类型，用一个int型表示

（4）后续可能需要完善的几点：

* 利用小程序本地缓存和小程序open\_id关联，只要用户登录一次，后续就自动识别了；当然这个点是存在安全风险的，待深入研究。
* 用户的登出问题，将在个人首页（就是每个用户登录成功后跳转的默认界面）的右上方提供一个Setting（设置）入口，在设置界面完成登出作业。
* 登录界面的背景图尺寸太大，需要裁剪或者换图。



* 【10002】退出登录

微信小程序访问函数 10002-1

* + 函数名称 **userLogout()**
  + 访问模式 post
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**logout\_status : 'xxxx', //传回一个返回码，默认0为正常退出，**

**//其他返回码待定**

**}**

（1）服务器接收到用户退出的请求后，在服务端完成用户登出的记录和相关操作，如果没有异常，直接返回状态0；如果发生异常情况，需要返回错误码；小程序端将根据错误码在界面上给出相应的提示；方便日后维护和测试中定位bug。

（2）小程序如果收到登出正常的返回码，将做两件事情：

1）主动清空本地小程序缓存。这点很重要，小程序缓存上限就10MB

2）根据场景决定是否重新加载登录界面，甚至需要考虑小程序直接关闭的场景（后续完善）。

* 【10003】查看资讯

微信小程序访问函数 10003-1

* + 函数名称 **getInfoList() //获取消息列表，点击tab时调用**
  + 访问模式 post
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx', //用于服务器端精准投放消息**

**info\_type : 'xxxxx', //这个值用于区别资讯中4类消息列表**

**}**

* + 服务器返回数据Json形式

**data : {**

**info\_id\_up : 'xxx', //最上面一条消息的id**

**info\_id\_down : 'xxx', //最下面一条消息的id**

**infos[**

**{ //一条消息的Json结构**

**info\_id : 'xxx', //消息id**

**title : 'xxx', //消息标题**

**time : 'xxx', //发布时间**

**image : 'url' //消息图片的url，有可能为空，通知、讲座不需要图片**

**decs : 'xxxx', //列表中的简要描述（30个字以内，简明扼要）**

**detail : 'xxxx', //消息内容**

**},**

**.... // 返回最多20条消息，10条用于直接显示，10条用户向下滑动时**

**] // 显示备用，以提高用户体验**

**}**

微信小程序访问函数 10003-2

* + 函数名称 **getInfoListMore() //获取消息列表，滑动列表到底时调用**
  + 访问模式 post
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx', //用于服务器端精准投放消息**

**info\_type : 'xxxxx', //这个值用于区别资讯中3类消息列表**

**info\_id\_down : 'xxx', //最下面一条消息的id，让服务器返回后续的消息**

**}**

* 服务器返回数据Json形式

同上**getInfoList，但是仅返回10条**

微信小程序访问函数 10003-3

* + 函数名称 **getInfoListLastest() //获取消息列表，获取最新消息，顶部下拉时调用**
  + 访问模式 post
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx', //用于服务器端精准投放消息**

**info\_type : 'xxxxx', //这个值用于区别资讯中4类消息列表**

**info\_id\_up : 'xxx', //最上面一条消息的id，让服务器返回最新消息**

**}**

* 服务器返回数据Json形式

同上**getInfoList**，初步定规则是：服务器对比**info\_id\_up**

1）如果当前数据库中新消息>=10条，就返回从**info\_id\_up**开始计算的最新10条。

2）如果当前数据库中新消息<10条,就返回**info\_id\_up**开始计算的最新的所有消息。

一共三个函数，稍微复杂，欢迎指正给出更好的解决方案 O(∩\_∩)O

（1）界面大致效果见下图，“资讯”中新闻/消息的种类被分为3种：

0讲座：学校举办的学术讲座，或者其他论坛及讲座

1通知：与用户相关的重要通知

2分院：分院的要闻

（2）资讯列表不可能一下子加载很多条，所以我们采用了一些常见的加载列表策略，详情见函数描述。

（3）小程序端本地将缓存3个列表，分别对应这4类消息；每个列表缓存60条消息，应该已经足够；开发的难点可能在于列表追加（从底部向上滑动），插入（从顶部下拉刷新）。

**且对于没有分院归属，或者不上分院课程的管理人员、校工等不应该显示分院tab页。**

（4）服务端的难点在于消息的分类投放：

1）建议本地先缓存讲座、学校新闻两个消息列表数据，因为这两组数据不涉及精准投送，都是通用消息，与分院无关。

2）分院新闻在设计CMS时可考虑到这个问题，提前准备好数据。

3) 可能包含校级通知，也可能是分院通知，同样如果能提前在CMS供管理员录入信息时分类处理（通俗的讲就是每个部门有自己的列表，自己管理自己的消息录入）那么问题就很好解决了；或是在录入消息时，做分院标记，重新整理传给微信小程序。

4) 班级通知的推送可能是转发自分院的，不重复考虑减轻开发工作量，这个工作在“待办”tab中完成推送和监督。

**  **

* 【10004】成长信息

微信小程序访问函数 10004-1

* + 函数名称 **getStuRadarChart ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**radar\_data : [{ //学生素质雷达图，共6项，名称待定**

**title : 'xxxx', //雷达图单项名称string**

**value: xxxx, //对应数值**

**}, ... //共6项，剩余5项就不写出来了**

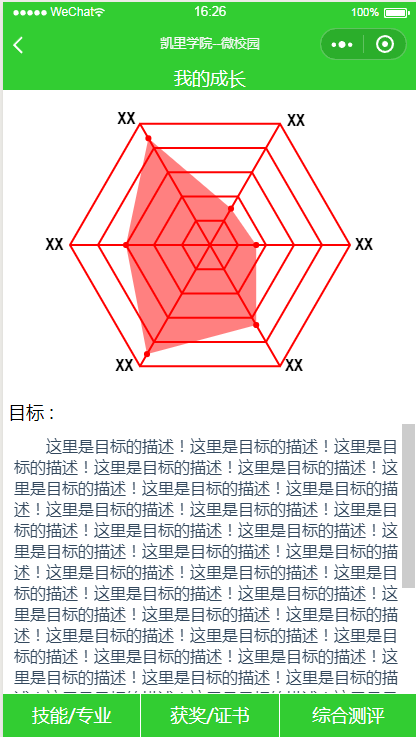
**],**

**goal : 'xxx' //个人发展目标，string**

**}**

（1）这个函数用于进入“成长”页面时获取和刷新数据用，服务端不用理会小程序本地数据刷新机制，只需要将数据按照约定格式返回即可。

（2）返回的雷达图数组共6项数据，名称可以后续再定下来，每项数据包含两个值：该项数据的名称（title）和数值。



微信小程序访问函数 10004-2

* + 函数名称 **getStuSkills ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**skills: [{ //学生当前有的技能和对应描述，比如学会了JSP技术**

**title : 'xxxx', //String 技术名称**

**level: xxxx, // int值 0-初步了解 1-熟练掌握 2-灵活应用**

**}, ... //数组不限制个数**

**]**

**}**

该函数从服务器获取当前学生掌握的各种专业技能和这些技能掌握情况（初步了解、熟练掌握、灵活应用），并以卡片方式显示在单独的界面中。后期可以考虑将技能的等级用颜色区分： 白色🡪绿色🡪蓝色🡪紫色（level 0~4，最高级表示已经达到行业技术岗用人水准）



微信小程序访问函数 10004-3

* + 函数名称 **getStuCertificate ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**certificates: [{ //学生当前有的证书和对应描述，比如CET-6**

**desc : 'xxxx', //String 证书名称和简要介绍**

**type : xxxx, // int值 0-资格证 1-获奖证书 2-培训讯结业证书**

**date : 'xxxx年xx月', //获得证书时间，格式任意，用String也可以**

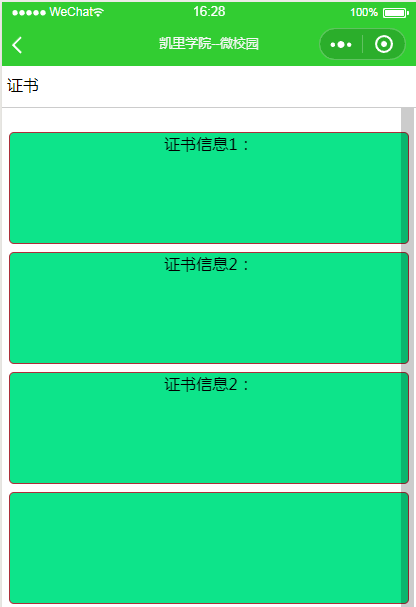
**}, ... //数组不限制个数**

**]**

**}**

（1）获奖/证书这个tab是用来展示学生当前的获奖状态的，通过不同的type值区分证书的种类，同时在小程序端显示不同的颜色。

（2）对服务端和小程序的主管、班主任用户（在教师页面的一栏）来讲是十分重要的一个数据项目，可以反应当前学生获奖拿证的统计数据；我们在设计之初将这个项目加入学生页面有两点考虑：一个是为后续2/3期工程做技术准备，二是让学生有晒“荣誉墙”的机会，正面激发学生的竞争意识。



微信小程序访问函数 10004-4

* + 函数名称 **getStuMultiData ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**stu\_mulit\_data: [{ //学生综合素质评价**

**desc : 'xxxx', //String 综合素质评价（大一）**

**}, {**

**desc : 'xxxx', //String 综合素质评价（大二）**

**}, {**

**desc : 'xxxx', //String 综合素质评价（大三）**

**},**

**] //数组暂定3项：大一、大二、大三**

**}**

综合素质测评也是主管用户比较关心的一个数据展示环节，但是内容目前暂定为全文字描述，因为这个界面是在小程序的layer4，已经比较深入了（小程序官方标准最多5层）；如果以后该需求地位提升，考虑转移到学生首页（layer2）



* 【10005】阅读

微信小程序访问函数 10005-1

* + 函数名称 **getBookPush ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**recommendation: [{ //推荐书籍列表**

**desc : 'xxxx', // String 推荐书籍列表，就是文字段落**

**tilte : xxxx, // String 如“2018年4月”推荐书籍**

**}, ... //数组初步定为6个月，也就是大概一个学期的样子**

**]**

**}**

（1）推荐书籍以月为单位，成条状信息呈现在界面中（后续做好了补图）

（2）每一条的标题为“xxxx年xx月 好书推荐”，可以点击打开查看推荐书单

（3）原则上这些书单是由服务端编辑好后加入desc字段的，项目前期采用统一推送模式，项目后期采用根据专业区分的精准推送。

（4）原则上建议每次从服务端取最近6个月的书单，因为一个学期也就4个月，这样前期研发编码压力会小很多。

（5）初步考虑这样的UI设计：12个月用12种颜色渐变，每种颜色代表当时月份的季节颜色，如12月就是白色的，10月就是金色的。

* 【10006】事务

微信小程序访问函数 10006-1

* + 函数名称 **getFreeClassroom ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**build\_id : 'xxxx', //String 或者int 是约定好的教学楼或者实训楼编号**

**time : “xxx” //自习节次，用于找到空闲房间**

**}**

* + 服务器返回数据Json形式

**data : {**

**free\_room: [ //空闲教室数组**

**“xxxx”, //String 教室1**

**“xxxx”, //String 教室2**

**... //根据选定地点和时间，列出所有可以用教室**

**]**

**}**

（1）显示结果放在界面上，按照房间编号从小到大排序，因为编号小的教室肯定在低楼层；

（2）（相关界面设计后续补上）这里类似于成绩查询的UI实现，用下拉条给学生选择，每次点击查询按钮后会检测当前查询内容是否和在小程序缓存中上次查询内容一致，如果一致就不调用函数访问服务器，减轻潜在访问压力。

微信小程序访问函数 10006-2

* + 函数名称 **askForLeave()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**}**

* + 服务器返回数据Json形式

**data : {**

**}**

请假申请（待完成）

微信小程序访问函数 10006-3

* + 函数名称 **askForLab()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**}**

* + 服务器返回数据Json形式

**data : {**

**}**

开放实验室申请（待完成）

微信小程序访问函数 10006-4

* + 函数名称 **askForDevice()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**}**

* + 服务器返回数据Json形式

**data : {**

**}**

设备申请（待完成）

微信小程序访问函数 10006-5

* + 函数名称 **askForRepair()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**}**

* + 服务器返回数据Json形式

**data : {**

**}**

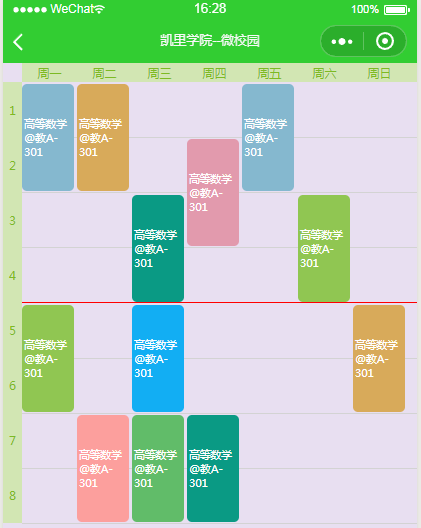
报修（待完成）

说明：

（1）各种申请实际上都可以使用同一个模板，就像列表一样，目前设计方案已经下发给学生，正在施工中，需要耐心等待小程序端的实现，最后才能同服务端协商确认接口形式

（2）因为申请是有一个流程模板的，这件事情需要认真的商议。

* 【10007】课表



微信小程序访问函数 10007-1

* + 函数名称 **getTimeTable ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**time\_table: [[//课程表数组，是一个二维数组**

**{ // 每一组括号代表每周的一天**

**start : xxx, //int第几课开始**

**len : xxx, //int 占几节课**

**desc : “xxxx”//课程名称和教室，如：高等数学@教A-301**

**},{**

**//今天的下一节课**

**}**

**],[ // 注意这里加的是方括号**

**…**

**],**

**…**

**]**

**}**

（1）这里我们使用二维数组存放课程信息，每条项目就是一个数组，每个数组代表学生一天的课程（按照周一~周天的顺序排列，如果周六、周天没有课程，就直接跳过；假若周一到周五某天没课，也是需要添加一个空选项的，相当于站位符，当然这里后续还需要解决可能出现的bug）

（2）对每一天的课程，都是服务器根据个人课表、单双周情况和所在时间给出的信息：也就是说如果本周是单周，课表上就会显示单周课和每周都上的课程，而不会显示双周才有的课程，且学生个人选修课（包括公选课，我们将设计得松散些，适应之后需求变更）也需要正常显示。

（3）每天的课程数组，按照常理全天满课（包括晚上）的标定应该有10节课的，但实际情况一般是8节课，且有的时候第一节课可能没有，所以依靠数组中**start（**从第几节开始上课）和**len**（要上几节课）两个字段正确显示

（4）这个模块是非常通用的，因为后续教师课表，考试时间教室查询也可以借用这个模块，小程序端最好能将这个功能做成模板或者组件。

* 【10008】竞赛

微信小程序访问函数 10008-1

* + 函数名称 **getCompetition()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**list : [{**

**title : “xxx”, //string 竞赛信息题目**

**date : “xxxx-xx-xx”， //发布时间，最好是String，格式自定义**

**url : “xxxxx” //竞赛信息发布的网页地址**

**},{ //第二条信息**

**….**

**},…. //有很多条信息**

**]**

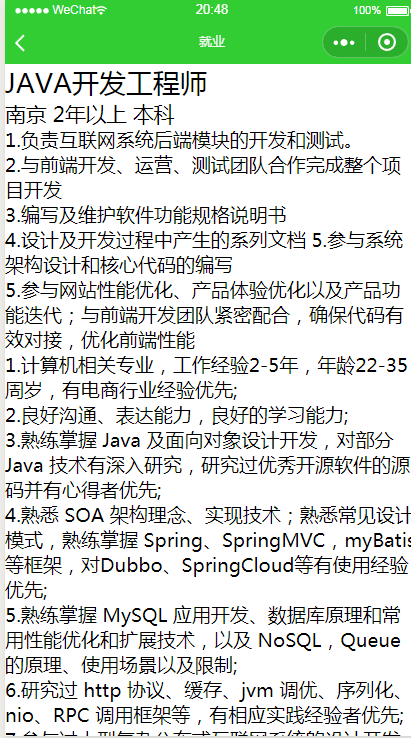
**}**

(1) 我们采用列表的方式将有关竞赛的信息推送给用户

（2）用户点击列表项，由列表项存放的url跳转到发布信息的网址，这样就不用我们二次编辑文字了，省去了很多问题。

（3）从外网网址跳转回小程序时，可能存在bug，需要修正。

* 【10009】实习/就业 （类似竞赛）

微信小程序访问函数 10009-1

* + 函数名称 **getRecruitment ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**list : [{**

**decs : “xxx”, //string 招聘信息抬头简介，不超过20个字**

**date : “xxxx-xx-xx”， //发布时间，最好是String，格式自定义**

**url : “xxxxx” //招聘信息发布的网页地址**

**},{ //第二条信息**

**….**

**},…. //有很多条信息**

**]**

**}**

* 【10010】社团

微信小程序访问函数 10010-1

* + 函数名称 **getSociety ()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**}**

* + 服务器返回数据Json形式

**data : {**

**list : [{**

**title : “xxx”, //string 社团名称**

**url : “xxxxx” //社团主页地址**

**},{ //第二条信息**

**….**

**},…. //有很多条信息**

**]**

**}**

UI布局上风格任意：卡片式、网格式、列表式都行；用户点击选项后，直接连接到相关网页上；在前期我们考虑给出描述，再做一层页面，类似资讯新闻的展示方式；需要注意显示的社团信息是根据当前用户上传的id决定的。

* 【10011】成绩查询

微信小程序访问函数 10011-1

* + 函数名称 **getExamResult()**
  + 访问模式 get
  + 小程序上传数据Json形式

**data : {**

**user\_id : 'xxxxx',**

**exam\_name : ‘xxxx’, //String考试名称，支持模糊查找**

**exam\_date : ‘xxxx’ //String考试时间，让用户通过下拉条选择**

**}**

* + 服务器返回数据Json形式

**data : {**

**exam\_name : ‘xxxx’, //String考试官方名称**

**exam\_result : ‘xxxx’ //考试结果，为了兼容各类等级和资格证考试，**

**} //采用String 类型的文字描述**

（1）这个界面正在开发中，预想有单科查询和全部查询模式

（2）单科成绩的查询主要是要求服务端通过单科名称做在对应时间内进行模糊查找

（3）全部查询仅核对考试时间。

## 3.4 团队开发工具

# 4团队成员与分工记录

## 4.1 前端

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 编号 | 目标简介 | 负责人 | 完成进度 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## 4.2 后端

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 编号 | 目标简介 | 负责人 | 完成进度 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 5开发计划

后面补齐，大致定在12月中旬第一个版本能够试用。

# 6 项目进度记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 编号 | 目标简介 | 完成情况 | 备注 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |