**项目研究进展情况**（包含小组成员的分工合作，项目研究内容、方法、进度及已取得的阶段性成果或收获等）

**一、小组成员的分工合作**

钱怡辰：担任项目负责人，负责人员分工、进度安排、需求分析、服务器端程序开发、客户端与服务器端数据交互。

刘凡莉：负责客户端程序开发、软件测试、财务管理。

王帅君：负责客户端UI设计、数据库设计、服务器端程序开发。

李隆岩：负责客户端UI设计、服务器搭建、服务器端程序开发。

1. **研究内容与方法**

1、时间分配统计

“TimeManager”可以收集两类事务的时间信息：日程表中已安排的事务和用户即时新增的事务。一方面，用户可以在创建计时的时候进入日程表选择已安排的日程开始计时，可以更方便地记录事务，同时方便在统计和行为模式分析中比较日程表的时间安排与实际的时间分配之间的不同；另一方面，用户可以在每天进行各项事务（比如学习，休息，运动，社团活动）的时候创建新计时，填写事务的基本信息后开始计时，完成该事务后结束计时，用户如果在进行某项事务时忘记计时，可以事后在当天的时间分配表中补记该事务，从而更加全面地收集用户的时间分配信息。“TimeManager ”对用户的时间分配分标签进行统计，并以日报表、周报表的形式反馈给用户。

时间分配统计的研究方法：在数据库事务表中添加标签字段，并建立日程表、时间分配表和事务的关系的表。用户在创建计时或者创建日程的时候选择事务的标签，然后”TimeManager”分标签统计每天/每周各种类型事务的总时间、时间比例、平均满意程度，以饼图和折线图的形式给用户直观的反馈，促进用户反思自己的时间分配并作出调整。

2、提醒设置与推送

用户安排日程时可以设置事务的计划起止时间，可以设置在事务发生前进行提醒，根据用户设置的提前提醒时间（比如1分钟，10分钟，1小时），”TimeManger”会给用户发送消息推送到通知栏。

提醒设置与推送的研究方法：”TimeManager” APP需要在用户开始计划事务前向用户推送提醒消息。实现过程中使用Android自带AlarmManager类，在用户设置提醒后立即发一条延时广播，接收器在用户安排的时间接收到广播以后再根据接收到的广播中的数据向用户推送提醒消息。

3、优秀经验分享

优秀经验分享为学业困难学生的帮扶工作提供了创新性途径，主要分为优秀学生分享时间分配表与优秀学生时间分配表展示两个部分。

优秀学生分享时间分配表：用户在注册的时候要输入自己的专业和平均学分绩（GPA），平均学分绩高于一定标准的学生可以选择自己某天的时间分配表分享。

优秀学生分配表的展示：用户可以筛选出本专业或者感兴趣的某个专业的优秀学生分享的时间分配表（通过动态搜索实现），也可以查看所有专业优秀学生分享的时间分配表。已分享的优秀学生时间分配表首先根据优秀学生的平均学分绩整体将他们分享的时间分配表排序，再以点赞数将每个学生分享的时间分配表排序，排序后以列表形式显示。用户可以查看感兴趣的优秀学生时间分配表的详细信息，并且可以点赞、收藏、评论喜欢的优秀学生时间分配表。

优秀经验分享的研究方法：在数据库中添加“已分享的时间分配表”表，并与“时间分配表”表建立外键关系。已分享的时间分配表中包含分享简介，点赞数等字段，有利于用户选择更适合自己的时间分配表，从而更好地学习时间管理的经验，改进自己的时间分配方式。优秀学生时间分配表的排序先选出平均学分绩最高的学生（五人左右）的点赞数最高的几条时间分配表（三条左右），其他时间分配表按照分享时间顺序排在其后，综合考虑了功能、热度和时效，同时比较符合用户对于列表的使用习惯。客户端获取优秀学生时间分配数据以列表-详情页的形式展出，比较灵活。

4、行为模式分析和反馈

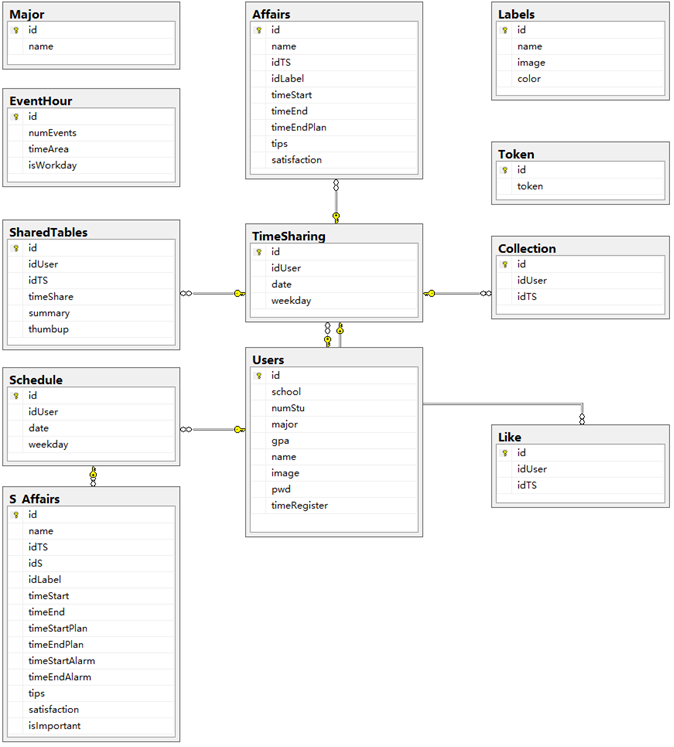
根据收集到的用户事务处理时间信息统计时间分配情况，挖掘用户的行为习惯，从日程安排完成率，事务平均推迟时间，事务超时比例以及每个时间段中的行为密集程度（一个时间段处理多少件不同事务）等角度分析用户的行为模式，帮助用户反思时间分配方面存在的问题。

行为模式分析和反馈研究方法：我们从三个方面对用户的行为模式进行分析反馈：各标签实际使用消耗时间和日程表中相对应分配时间的节约/超时情况占比、各标签所有事件中对比日程安排的节约/超时时间数占比、七个时间段各自的每小时完成事件数量，我们在设计时将在日程表中有安排的事件与普通事件分成S\_Affair和Affair两种不同的类，因此在分析时只需要从S\_Affair中获得对应用户对应标签的数据情况进行求百分比的计算，第三个方面的分析直接对S\_Affair和Affair分别分析并加和，再进行占比的求解，在客户端以柱状图、直方图等形式呈现，方便客户直观的看到自己在各个标签所划分的时事件类别上的时间分配情况以及各个类别和各个时间范围的时间利用效率情况，有利于客户据此在最高效的时间范围完成最重要的事件，并对自己的时间分配情况有大大致把握。

**三、进度及阶段性成果展示**

1. 在GitHub创建多人合作公共仓库，地址如下：<https://github.com/OUCsrdp/TimeManager>

2、对数据库设计根据功能的添加调整进行修改完善，完善后的E-R图如图2-1所示：

图2-1

1. 客户端大部分页面完成，功能完成百分之七十，前后端交互完成百分之六十，进度与预期相符。目前，已基本完全实现的操作有：用户对个人信息的修改；获得本专业和所有专业优秀学生的时间分配表推荐信息；对当前事务的计时；提前安排日程并设置提醒；查看时间分配表、补充或编辑时间分配表中的事务；对喜欢的优秀学生时间分配表进行点赞和收藏。服务器端完成的功能有：统计用户每天各类事务所占时间比的日报表展示；统计用户每周各类事务所占时间比、每类事务用时在一周内变化趋势的周报表展示；对比用户日程安排和实际时间分配情况的行为模式分析。

以下是这半年的最新进展和成果展示：

3.1 对计时功能的完善

利用了数据存储化技术，使得用户在离开计时页面/离开APP后能正常计时。用户点击右上角的”+”创建事务，点击下方的结束按钮结束计时并保存该事务的计时结果到服务器。实现效果如图3-1所示：



图3-1

3.2 推送提醒功能

在创建计时时用户可设置提醒的时间，接收器在用户设置的时间到达时，接收到广播并向用户推送提醒消息到通知栏。实现效果如图3-2所示：





图3-2

3.3 日报表、周报表

3.3.1增加日历选择页，通过右上角的按钮选择是查看日/周报表，并通过在日历上点击日期查看该日/周报表。实现效果如图3-3-1所示：

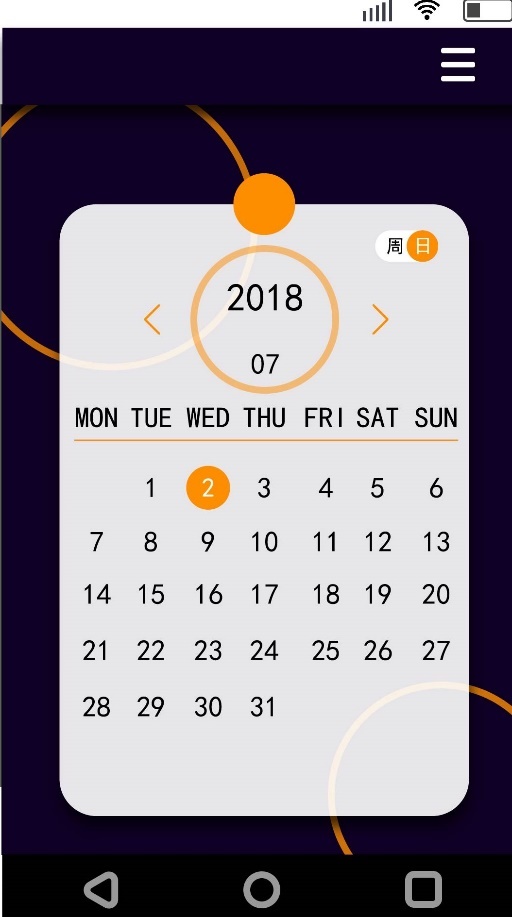


图3-3-1

3.3.2 服务器端完成日报表中统计用户每天各类事务所占时间比功能；完成周报表中统计用户每周各类事务所占时间比、每类事务用时在一周内变化趋势的功能。函数框架如图3-3-2所示：

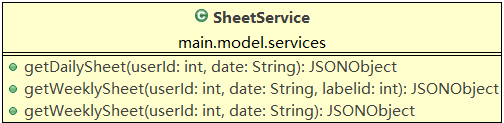


图3-3-2

该部分的核心算法：

1. 获取日报表（各类事务所占时间比）

JSONObject getDailySheet( userId, date)

{

    定义JSONObject back 用于返回当天时间分配和日程安排的相关信息;

    定义TimeSharing类动态数组 timeSharings 获取该用户时间分配表;

    if 获取不到用户分享的时间分配表

        函数中断执行，返回空;

    定义TimeSharing timeSharing 用来甄别用户的事件分配表是否为所选日期;

    for timeSharing : timeSharings

    {

        如果是所选日期的时间分配表则使用timeSharing获取，并中断循环执行;

    }

    if 没有所选日期的时间分配表

            函数中断执行，返回空;

    定义Affair类动态数组 affairs 用来获取当天的时间分配事件;

    if 当天没有时间分配事件 affairs 赋为一个空的动态数组;

    定义S\_Affair类动态数组 s\_Affairs 用来获取当天的时间分配事件;

    if 当天没有日程 s\_Affairs 赋为一个空的动态数组;

    声明一个字符串 weekday\_s 用来获取当天的日期;

    switch timeSharing.getWeekday()

    {

        获取日期，赋值给weekday\_s;

    }

    向 back 中加入日期

    定义Label类动态数组 labels 用来获取当天的时间分配事件和日程的标签

    定义JSONArray affairArray 用来获取当天所有时间分配事件的相关消息;

    定义int minutesAll 来获取当天时间分配的总时间;

    for affair : affairs

    {

        将minutesAll中逐项加上时间分配事件的时间;

    }

    for label : labels

    {

        定义获取当天各个标签的事件所占的时间比所需变量

        for affair : affairs

        {

            逐项统计每个标签的事件的平均时间，所占时间在总时间中的比例以及满意度

        }

        向 affairArray 中加入得到的相关变量的字符串

    }

    向 back 中加入当天的时间分配表的事件的相关信息 affairArray;

    定义JSONArray s\_AffairArray 用来获取当天所有时间分配事件的相关消息;

    for s\_Affair : s\_Affairs

    {

        将 minutesAll 中逐项加上日程的时间;

    }

    for label : labels

    {

        定义获取当天各个标签的日程所占的时间比所需变量

        for s\_Affair : s\_Affairs

        {

            逐项统计每个标签的日程的平均时间，所占时间在总时间中的比例

        }

        向 s\_AffairArray 中加入得到的相关变量的字符串

    }

    向 back 中加入当天的时间分配表的事件的相关信息 s\_AffairArray;

    返回 back;

}

1. 获取周报表（某类事务在一周所用时间的变化趋势）

JSONObject getWeeklySheet(userId, date, labelid)

{

    定义JSONObject back 用于返回指定周的时间分配和日程安排的相关信息;

    定义TimeSharing类动态数组 timeSharings 获取该用户时间分配表;

    定义Calendar类变量 timeOfDate 获取指定日期的年月日;

    定义Calendar类变量 timeRegister 获取用户注册的年月日

    定义int型变量 week，使用timeOfDate和timeRegister获取当前是用户使用的第几周

将 week 添加到 back;

定义String类动态数组 durationArray 用来获取当天;

将timeOfDate回到上周日;

for i = 0; i < 7; i++

{

 将timeOfDate向前推动一天

 定义TimeSharing类变量 timeSharing 获取 timeOfDate 的时间分配表;

     for timeSharing : timeSharings

     {

          选取 timeOfDate 的时间分配表。

 }

 if 这一天没有时间分配

 {

     将durationArray当天的平均时间赋值为00分00秒并跳过统计当天的信息;

 }

  获取当天的平均时间并放入durationArray;

}

将durationArray放入back;

 返回 back;

}

3.4 行为模式分析

3.4.1完成部分页面静态页面实现，图3-4-1是反映各类事务实际花费时间相比日程安排多/符合/少的比例统计图的效果实现。

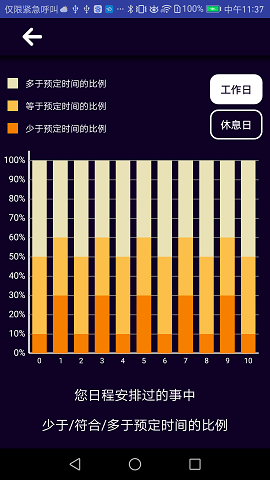


图3-4-1

3.4.2 服务器端完成行为模式分析功能，包括：获取用户使用该APP的总天数；统计日程完成率；统计事务平均拖延时间；统计行为密集程度（算法待优化）。函数框架如图3-4-2所示：

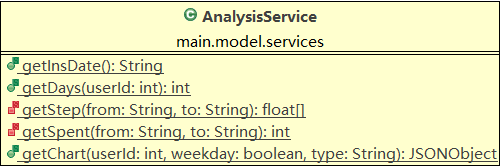


图3-4-2

getChart函数(进行行为模式分析生成统计图)的核心算法：  
1. 计算各类事务总时间相比日程安排该类事务总时间的节约/超时比例

if type == "SimpleAnalysis"

            then:

             labelList /\*ArrayList<Label>,所有标签list\*/

             for Label label:labelList /\*找到所有标签对应的\*/

                 percent //float,比例计算结果

                 truth //int,标签所有事件实际消耗时间

                 schedule //int,标签所有事件计划使用时间

                 SAList /\*ArrayList<S\_Affair>,当前标签对应的日程list\*/

                 for S\_Affair sa:SAList

                     if 当前日程是userId对应的用户的日程

                         then

                             truth += getSpent(事件开始时间,事件结束时间)

                         schedule += getSpent(事件计划开始时间,事件计划结束时间)

                 end

                    /\*计算比例\*/

                 if 计划消耗时间 != 0

                     then

                     percent = |schedule - truth|/schedule

                    endif

                    /\*判断节省/超时\*/

                 if schedule >= truth

                     将"timeSaving"存入js

                 else

                     将"timeOut"存入js

                 当前js加入数组array

             end

             当前array加入最终返回的js对象

     endif

2．计算每个标签所有事务中相比日程安排超时/节约的事务数比例

elseif type == "detailedAnalysis"

            then:

             labelList/\*ArrayList<Label>,所有标签list\*/

             for Label label:labelList

                 percent //JSONObject,比例数组，分别记录节约、相符、超时事件数量占比

                 save,equal,over //int,节约、相符、超时事件数

                 SAList /\*ArrayList<S\_Affair> ，当前标签对应日程list\*/

                 js //jsonObject,用于临时存储

                    for S\_Affair sa:SAList

                        if 当前日程是当前用户的日程

                     then:

                             truth = 0; //int,标签所有事件实际消耗时间

                             schedule = 0;//int,标签所有事件计划使用时间

                             truth = getSpent(事件开始时间,事件结束时间)

                             schedule = getSpent(事件计划开始时间,事件计划结束时间)

                             if truth > schedule then: over ++

                             else if truth == schedule then: equal ++

                             else save ++

                        endif

                    end

                 all = save + equal + over //int,所有事件数

                 per1，per2，per3 //float,用于临时存储三个占比

                 if all != 0

                    then:

                     per1 = ((float)save/all)\*100;

                     per2 = ((float)equal/all)\*100;

                     per3 = 100 - per1 - per2;

                    endif

                 三个占比分别加入percent数组

                 将计算出的占比数组percent加入js

                 当前js加入数组array

             End 当前array加入最终返回的js对象

1. 分析用户各个时间段的行为密集程度

elseif type == "densityAnalysis"

            then:

             count //float[] ，（总）所有事件在各时间段占比

             count1 //float[] ，（个）当前事件在各时间段占比

             all //long,所有事件时间数

             AList //ArrayList<Affair>，所有事件list

             for Affair a:AList

                 if 是当前用户的事件

                     then

                     count1 = getStep(开始时间,结束时间) //获得当前事件在各时间段的占比

                     for int j = 0;j < 7;j ++

                         count[j] += count1[j]

                     end

                     all ++;

                    endif

                end

            per //float[] ，每一阶段的有效利用时间和阶段总时间之比

            计算per[i]，按序存储

            per[i] = count[i] / time\_i /\*time\_i指第i时间段总时间\*/

            density = all / 用户有效使用时间 //float，平均每小时事件完成数

            //density保留两位小数

            for int i = 0;i<7;i++

                per[i] = (float)(Math.round(per[i]\*100))/100

            end

            将density加入最终返回的js对象

            将不同时间段的阶段名称以及相对应的per数组的比例数值存入新建的jsonObject

            将jsonObject存入jsonArray

            当前array加入最终返回的js对象

        end\_elseif

3.5 优秀学生分配推荐页

3.5.1 展示优秀学生分配推荐列表

用户点击上方按钮筛选条件/专业（所有专业/本专业/搜索专业），综合考虑分享者的GPA、分享时间、热度（点赞数）调整展示顺序并按序展示被分享的优秀学生时间分配页。点击其中任意时间分配表可进入该表的详情页。实现效果如图3-5-1所示：



图3-5-1

3.5.2 用户选择筛选条件

用户可以对想要查看的优秀学生时间分配表进行筛选，选择所有专业、本专业、或者其他任意感兴趣的专业（提供动态搜索，并依据相关度进行搜索结果的展示）。如图3-5-2所示：

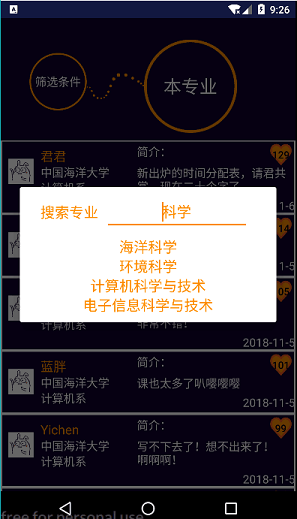


图3-5-2

3.6 优秀学生分配详情页

展示优秀学生分享的时间分配表，用户可以点击下方按钮对该时间分配表进行点赞或收藏。实现效果如图3-6所示：

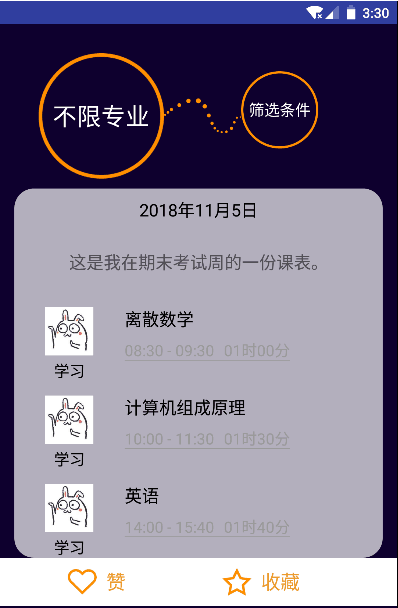


图3-6

3.7 对登录功能进行优化，给发送网络请求时的令牌进行加密解密，强化了用户密码的安全性。函数框图如图3-7所示：

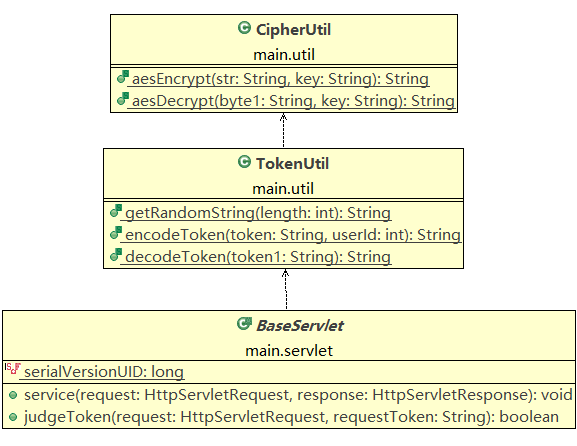


图3-7

**项目研究存在的主要问题及应对措施（如项目的研究内容及项目团队成员有变动，也请在此栏说明）**

1. 用户计时时，如果突然退出APP，再次进入APP可能会使计时不准确。

2、前后端交互和对token进行加密时，数据出现乱码。

3、对用户进行行为模式分析时，如果有大量的数据，计算速度较慢。

**应对措施：**

1. 利用数据持久化技术，将用户当前事务的开始时间存储在本地，每次进入APP计时页面的时候读取。

2、使用Base64编码规范，保证byte[]和String之间的标准转换。

3、开拓思路，对行为模式分析的算法进行优化（正在优化算法）。

**项目研究下阶段主要任务及时间进程安排**

2018年12月 完成客户端程序开发

2019年1月 完善行为模式分析

2019年2月 完成客户端与服务器端数据交互

2019年3月 优化项目细节

2019年4月 系统测试，信息学院内部试用

2019年5月 项目结题，撰写结题报告（如果时间许可，提交计算机软件著作权申请）。

**项目经费使用情况（说明购置实验材料、试剂、药品、加工测试、资料、复印、调研、交通等已开支经费数额）**

|  |  |  |
| --- | --- | --- |
| **项目** | **已支出（元）** | **支出说明** |
| 图书资料费 | 126 | 购买专业书籍 |
| 实验材料费 | 2309.9 | 购买内存条、硬盘、鼠标、键盘等计算机配件 |
| 合计 | 2435.9 |  |