需求分析报告

设计背景

大学中每位同学每天都有学校课程、自选课程、课外活动、临时事务等多类活动, 每类活动的特点各不相同, 因而, 需要针对学生们的特殊需求, 将每天的多类活动进行有效的管理和提醒.

项目综述

本项目技术选型，拟定后端采用**java**语言，**B/S 结构**来完成服务主体；数据库视需求，灵活使用**MySQL**与**redis**进行开发。完成的选做任务是**导航图形化**，以及**课表图形化**，这两者的前端采用**html+css+js**来完成

创新点

- 后端采用主流的**Spring boot**框架完成主体功能，并以时下流行的**Spring Cloud Alibaba 微服务**架构结合了部分分布式服务的实现来探索创新。 在加强项目性能与可靠性的同时，解决了多成员间分模块开发如何运行的的协调问题。 此外，也让大家接触新技术、新知识，在项目开发的过程中锻炼了自己的能力
- 前端使用**vue3+js**

总体设计要求

选做项目

- 设计导航功能的图形界面, 包括地图展示和输出路径展示;
- 能够使用课表图形界面方式进行课程管理和查询;

基本需求

设计一款学生日程管理系统可以帮助学生管理自己的课程和各种活动, 具备课程类日程管理、课外活动日程管理和临时事务日程管理等。每天晚上系统会提醒学生第二天的所有日程; 快要到活动时间时, 系统会根据活动的类型进行相应的提醒和规划; 也可以查看一个学期的所有日程.

对于课程类, 邻近上课时间时会进行提醒：如果是线下课程, 会输出去教室的路线, 如果是线上课程, 会输出课程在线平台和链接; 对于课外活动类, 如果是线下活动, 会输出到活动地点的路线 (校内) , 如果是线上活动会输出活动的在线平台和链接; 对于临时事务, 可以规划几个并行事务的最佳路线, 并输出.

课程类日程管理需求

- **用户可以查询某个课程相关的信息**（如：该课程上课时间/地点、考试时间/地点）；
- 将用户进行分组, 并分别赋予不同的权限, 拥有管理员权限的用户可以随时**发布或修改**课程信息;
- **课程为周期或单次**，有早八~晚八的时间限制，建议以某时刻到 0 点的分钟作为单位，然后小时显示再转换;
- 在每天晚上系统会提醒学生第二天的所有课程, **高效查找算法**;
- 临近上课时也要进行提醒。
- 若线下，**输出导航路线**，且为最优解（见地图导航模块）；若线上，**输出链接**;
- 单个同学的课程之间不允许冲突, 不同同学的同一门课程的上课时间地点应当相同, **校验**算法.

课外活动日程需求

- 学生可以**输入**课外活动信息, 有**个人活动**和**集体活动**;
- 学生可以根据**时间**、**活动类型**查询活动, 并且是**高效**的查找/排序;
- 在**有效的时间 (6: 00-22: 00)** 内 (只能持续**1 个小时**), 在**整点**开始或结束,为**周期**/**单次**活动;
- 学生可以设置活动闹钟, 闹钟可以是**单词、每天一次、每周一次。**若线下，**输出导航路线**，且为最优解（见地图导航模块）；若线上，**输出链接**;
- 每天晚上系统会提醒学生第二天的所有课外活动, 所以需要高效的**查找**算法;
- 活动不得与课程冲突。 返回合理的活动安排 (注意补救措施)

(核心算法为检测时间冲突算法；当用户输入课外活动时进行冲突检测，如果出现冲突需要为个人活动和集体活动提供当天可行的三个时间；如果没有可性时间，个人活动提示失败，集体活动给出当天冲突最少的三个时间).

临时事物日程管理

- 学生可以输入临时事务信息, 临时事务类型包括: 购物、洗澡、取外卖、取快递、送取东西等;

- 学生可以根据**时间、临时事务类**型进行查询, 并对查询的多个结果进行**排序**, 所以需要高效的**查询**和**排序**算法;
- 临时事务可以指定某个小时完成, 含义为该小时内完成即可, 临时事务**不考虑**持续时间 (有效临时事务完成时间为**6：00-22：00**);
- 学生可以设定临时事务闹钟, 闹钟是**一次性的**, 用于事务提醒: 输出去临时事务地点的路线, 所以需要高效的**查找路径**算法;
- 临时事务不能与课程和课外活动冲突, 如果发生冲突则输入失败, 所以需要合理的**冲突检测**算法;
- 多个临时事务可以**同时进行**, 此时需要根据这些临时事务的地点和用户所处位置规划**最佳完成路线**, 所以需要设计出一种**途径多个地点的最短距离路径**算法;

日程导航

- 学生向系统**输入**某课程/活动/事务的名称，或根据他们的提醒**跳转而来**，则查询该任务**当天/本周/以后**的时间及地点并给出**导航**（优先级递减，不是本周以内的需要提示确认。导航内容参照前面描述的）；
- **最短距离策略**要求: 距离最短即可, 输出相应的路线; 路线可以是图形展示, 也可以是文字描述; (优先队列 dijstra)
- 当可以同时进行多个临时事务时, 需要使用**途经最短距离策略**: 从起点出发经过多个地点并回到起点的最短距离路径. (搜索+剪枝+限制搜索深度)

模拟系统时间

- 系统依据时钟向前推进, 时间精度为小时, 且以计算机的 10 秒作为模拟系统的 1 小时 (技术待定，考虑每个事件（课程）使用一个定时器，然后一个线程来定时自检,削所有的定时器，当某一个到时间的时候就给用户发出提示);
- 推进模拟的内容是各个事件到时间的提醒，不是模拟用户导航的移动（应该？）
- 人机交互时暂停系统时间推进 (例如用户输入信息时); 可以通过加入时钟暂停按钮或者命令来实现.

建立日志文件

- 记录学生课程、课外活动和临时事务的状态变化, 系统提醒的信息, 输出的导航信息, 以及学生输入的信息和各种查询操作.

选做项目需求

- 在前端页面绘制一张地图, 当输出导航路径时则在地图上面进行路线绘制;
- 前端页面应包含登录界面, 用户信息, 课程管理和查询入口等模块;
- 使用排序算法和冲突检测自动对课外活动进行排期, 并返回合理的可选时间.

概要设计报告

程序架构

主模块

- **模拟时间推进功能（一个线程，计时并检测 提醒 推送）**

用户信息

- **用户注册/登录**
- **权限认证，管理员操作**

课程表

- **课程类日程管理**
- **课外活动类日程管理**
- **临时事务类日程管理**

地图导航

- **导航线路设计和输出**

数据结构

由于还未开始真正编写，所以这里讨论的数据结构不可避免的可能会在后期根据需求和编程情况发生改变，这里给出的只是大致参考

用户信息

```
class User{
    /**
     * 用户名
     */
    private String name;
    /**
     * 用户密码
     */
    private String password;
    /**
     * 邮箱
     */
    private String mail;
    /**
     * 用户类别 (student/admin)
     */
    private String type;
}
```

日程信息

```
class Event {
    /**
     * 日程名称
     */
    private String name;
```

```
    /**
     * 日程形式 (online / offline)
     * online 的话就不用导航
     */
    private String type;
    /**
     * 日程地点
     */
    private String position;
    /**
     * 日程日期
     */
    private String date;
    /**
     * 日程起止时间
     * 均为整点
     */
    private int startTime;
    /**
     * 持续时间
     */
    private int duration;
    /**
     * 日程的周期
     * 0: 不循环
     * x: 每x天循环一次
     */
    private int cycleTime;
}
```

课程的上课信息和考试信息

```
class Lesson extends Event{
    /*super args*/

    /**
     * 考试信息
     */
    Event exam;
}
```

个人活动和集体活动信息

```
class Activity extends Event{
    /**
     * 活动性质 (personal / group)
     */
```

```
    private String activityType;
}
```

临时事务信息

```
class Temporary extends Event{
    private String activityType;
}
```

用户设置的定时闹钟信息

```
class Clock extends Event{
    /**
     * 事务类别
     */
    private String eventType;
}
```

建筑物的位置信息, 各条道路的距离等信息

```
class Buliding{
    /**
     * 建筑名称
     */
    private String name;
    /**
     * 建筑物坐标
     */
    private int x,y;
}
```

```
class Point{
    /**
     * 交叉点的id
     */
    int id;
    /**
     * 都是像素坐标：相对于地图左上角的位置
     */
    int x,y;
}
```

```
// 边集仅仅作为存贮点集地图的辅助信息，用来在初始化地图的时候构建，
// 实际运算都使用上面点集标识的地图
class Road {
    /**
     * 道路两端坐标
     */
    private Point from, to;
    /**
     * 道路长度
     */
    private int length;
    /**
     * 拥挤程度————待定
     */
    // private int busyLevel;
}
```

导航策略信息

```
class Compass{
    /**
     * 点集，描述当前的位置到目标的位置经过的点，
     * 展示时把它们的xy坐标依次连起来
     */
    private List<Point> paths;
}
```

其他：模拟推进的功能考虑每个事件（课程）使用一个定时器，然后一个线程来定时自检，削所有的定时器，当某一个到时间的时候就给用户发出提示，但是待解决一个怎么主动向客户提示的问题

```
class Timer{
    /**
     * 事件的id,
     */
    private int EventId;
    /**
     * 事件的时间
     */
    private long EventTime;
    /**
     * 更多的待定
     */
}
```

另，此页面的基本类型都将转换为包装类

整体计划

计划于第十周完成主体功能框架（不一定可实际运行，但是有代码可以说明功能）

第 11-12 周组员之间相互检查，补齐未完成功能。确保划分的模块可以正常运行（通过 postman 等 api 测试软件无误）

第十三周测试前端获取信息之后的交互显示有无异常并修改

之后时间留作机动

时间表

时间	郭晨旭任务安排	叶沛鑫任务安排	郭泽远
第五周	约定开发规范, 商讨并确定开发所使用的具体技术	确定前端开发所使用的技术，学习 html、css、js、vue 框架等前端知识	补充完成文档
第六周	对"事务"模块进行详细的架构, 包括但不限于确定最终的数据结构, 具体函数接口等	前后端充分沟通，确定前后端接口的规范	完成地图点集标注，学习途经最短路算法
第七周	创建数据库, 完成环境配置和框架的搭建(建立数据表, 使 java 代码和数据库相连等)	逐步实现主体框架功能	完成导航路径的实现，协助完成课程部分数据库设计
第八周	实现对学生事务增加和修改的操作(包括对课程冲突的检验)	逐步实现主体框架功能	学习、完成模拟时间推进的功能
第九周	对增加修改的代码进行完善, 实现对事务的删除操作, 开始构思查询事务的算法	借助 postman 等接口测试工具对接口进行测试和性能分析，确保划分的模块可以正常运行	继续完成时间模拟；与组员共同完成课程表部分
第十周	实现对事务查询的操作	基本完成主体框架功能	与组员共同完成课程表部分
第十一周	完善对事务查询的操作, 进行代码整合测试	组员之间相互检查，补齐未完成功能	检查功能实现，查漏补缺
第十二周	进行代码整合测试	进行代码整合测试	机动

目前进度

后端技术学习大致完成；已初步完成用户登陆注册的后端程序及界面