山东大学软件学院

高级程序开发(双语) 课程实验报告

题目名称： 学生信息管理平台的设计与开发

指导老师： 武蕾

组员： 202100300281 贾凤鸣

202000111131 陈书瑄

202100300059 程子函

202100300244 曹云培

202100300063 李彦浩

目录

1. 系统总体要求
   1. 系统总体功能要求
   2. 开发平台介绍
   3. 开发进度表
2. 系统需求分析

2.1 系统需求分析

1. 系统设计

3.1 系统概要设计

3.2 系统详细设计

1. 系统实现

4.1 主干部分的实现

4.2 课程部分的实现

4.3 学生成绩部分的实现

4.4 学生日志信息部分的实现

4.5 荣誉管理的实现

4.6 课程资源部分的实现

4.7 创新实践部分和日常活动部分的实现

4.8 学生作业信息部分的实现

4.9 学生基本信息部分的实现

4.10 个人简历部分的实现

第五章 总结与体会

5.1 各组员总结与体会

1. 系统总体要求
   1. 系统总体功能要求

本次设计要求利用Java实现一个学生信息管理平台（PC版，应用于校内网有线网络访问，暂不开发移动端）。主要功能为：

（1）学生基本信息、联系方式、入学前信息、家庭信息、社会关系等基本信息的管理

（2）学习信息管理，包括课程基本信息，课程中心（教材、课件、参考资料等）选课信息、考勤信息、作业信息、成绩信息等

（3）学生社会实践、学科竞赛、科技成果、培训讲座、创新项目、校外实习等创新实践信息管理

（4）学生荣誉信息管理，包括获得的各种称号奖励等

（5）学生体育活动、外出旅游、文艺演出、聚会、等日常活动管理。

（6）学生外出请假信息和生活学习消费等日志信息管理

（7）学生个人信息的统计汇入统计数据库

（8）学生各种信息的查询统计、综合绩分的计算（可自行设计公式）和学生个人画像、个人简历的生成打印

1.2 开发平台介绍

硬件环境：

贾凤鸣：

处理器：Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz 2.20GHz

机带RAM 8.00GB

陈书瑄：

处理器：Intel(R) Core(TM) i7-6500U CPU @2.52GHz

内存：12GB

程子函：

处理器：11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz 2.30GHz

机带RAM：16.0GB

曹云培：

处理器：11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz 2.30GHz

机带RAM：16.0GB

李彦浩：

处理器：Intel(R) Core(TM) i5-8257U CPU @ 1.40GHz 1.40GHz

机带RAM：8.00GB

软件环境：

操作系统：Win11

开发工具：IDEA

数据服务：SQLite

数据设计：Navicat

图表设计：procession

技术选型：

后端框架：springboot框架（三层 Models DAO Controller)

数据服务：SQLite

前端：由教师方提供

1.3开发进度表

|  |  |
| --- | --- |
| 实验时间/周 | 内容纪要 |
| 第二周 | 完成分组。进行了首次组会，为了更好地了解整个框架的各部分(models,repository,controller,yaml)的运作逻辑与规范，进行了首次分工。各组员分别学习不同的内容 |
| 第三周 | 学习实体类与JpaRepository，了解两者在springboot中的作用。组会期间确定models开发的分工，初步开发系统实体类 |
| 第四周 | 学习DAO层。组会期间分工完成了所有实体类的开发。初步了解JpaRepository的各类方法，以及自定义接口的书写规范；初步了解并学习SQL增删改查语句与JPQL查询语句 |
| 第五周 | 学习SQL语句与JPQL语句，仿照代码样例进行自定义接口的初步开发。 |
| 第六周 | 阅读样例代码并学习Controller与springboot注解。组会期间集体阅读并解析了样例代码中的TeachController部分，并附带阅读了DataResponse、DataRequest、CommonMethod等开发过程中必须用到的类，分析了他们各自附带的方法，对框架的数据传输等环节有了更进一步的认识 |
| 第七周 | 阅读样例代码。组会期间进行了Controller部分的分工，并首次尝试开发系统需要的Controller。同时也重新规划了实体类的各部分属性并作修改。 |
| 第八周 | 阅读yaml配置文件。组会期间反复实验，了解yaml中的编写规范，并编写了各自负责的Controller所需要的yaml代码，调试并成功运行。 |
| 第九周 | 各组员完成了各自负责部分的编写，调试代码，解决bug。此外，学习并了解了超链在springboot中的应用，按照程序设计要求编写各个页面间的超链接，并完成调试 |
| 第十周 | 为了让整个应用程序的各页面产生联系，重构了代码。增加了一些必要的实体类例如专业、年级、班级，完善了实体类的属性，使得学生实体以及其外键类与新增类发生关联。重写了yaml以及Controller中的代码，利用超链将各个页面有机地联系在一起，初步形成了最终成品的原型 |
| 第十一周 | 各个成员完成各自部分的代码重构，进行了代码整合。并协作完成UML类图，ER图，行为图的设计与绘制。完成了实验报告的编写。将成品源码、类图以及实验报告提交至FTP。 |

1.4 小组成员分工表及个人贡献比例

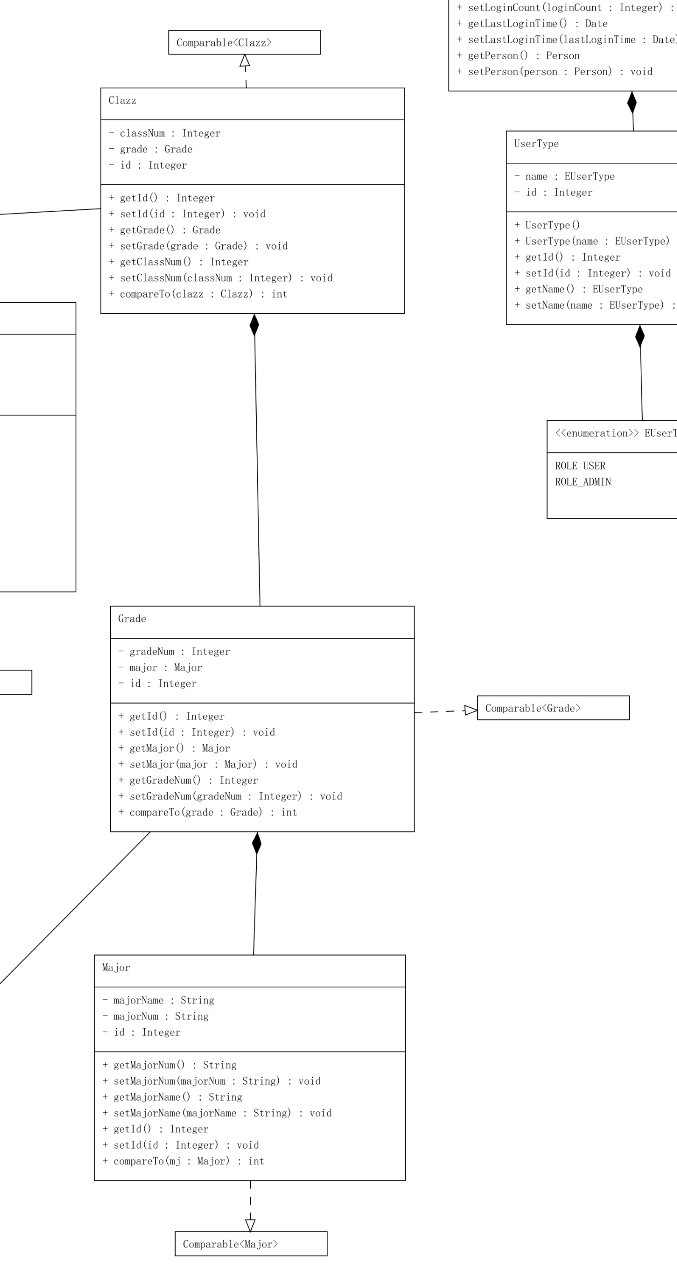
|  |  |  |
| --- | --- | --- |
| 小组成员（学号+姓名） | 负责模块（分工内容） | 个人贡献比例 |
| 202100300281 贾凤鸣 | 学生消费请假信息与学生日志信息（包含models、repository、controller、yaml等部分)和UML类图、ER图、行为图、论文的绘制与编写 | 1.00 |
| 202000111131 陈书瑄 | 学生荣誉信息与课程中心(包含models、repository、controller、yaml等部分)和UML类图、ER图、行为图、论文的绘制与编写 | 1.00 |
| 202100300059 程子函 | 学生基本信息与学生作业成绩信息（包含models、repository、controller、yaml等部分)和UML类图、ER图、行为图、论文的绘制与编写 | 1.00 |
| 202100300244 曹云培 | 学生创新实践信息管理和学生日常活动管理（包含models、repository、controller、yaml等部分)和UML类图、ER图、行为图、论文的绘制与编写 | 1.00 |
| 202100300063 李彦浩 | 专业、年级、班级、学生信息与课程信息、学生成绩信息（包含models、repository、controller、yaml等部分)和个人简历。以及论文的编写 | 1.00 |

第二章 系统需求分析

2.1 系统需求分析

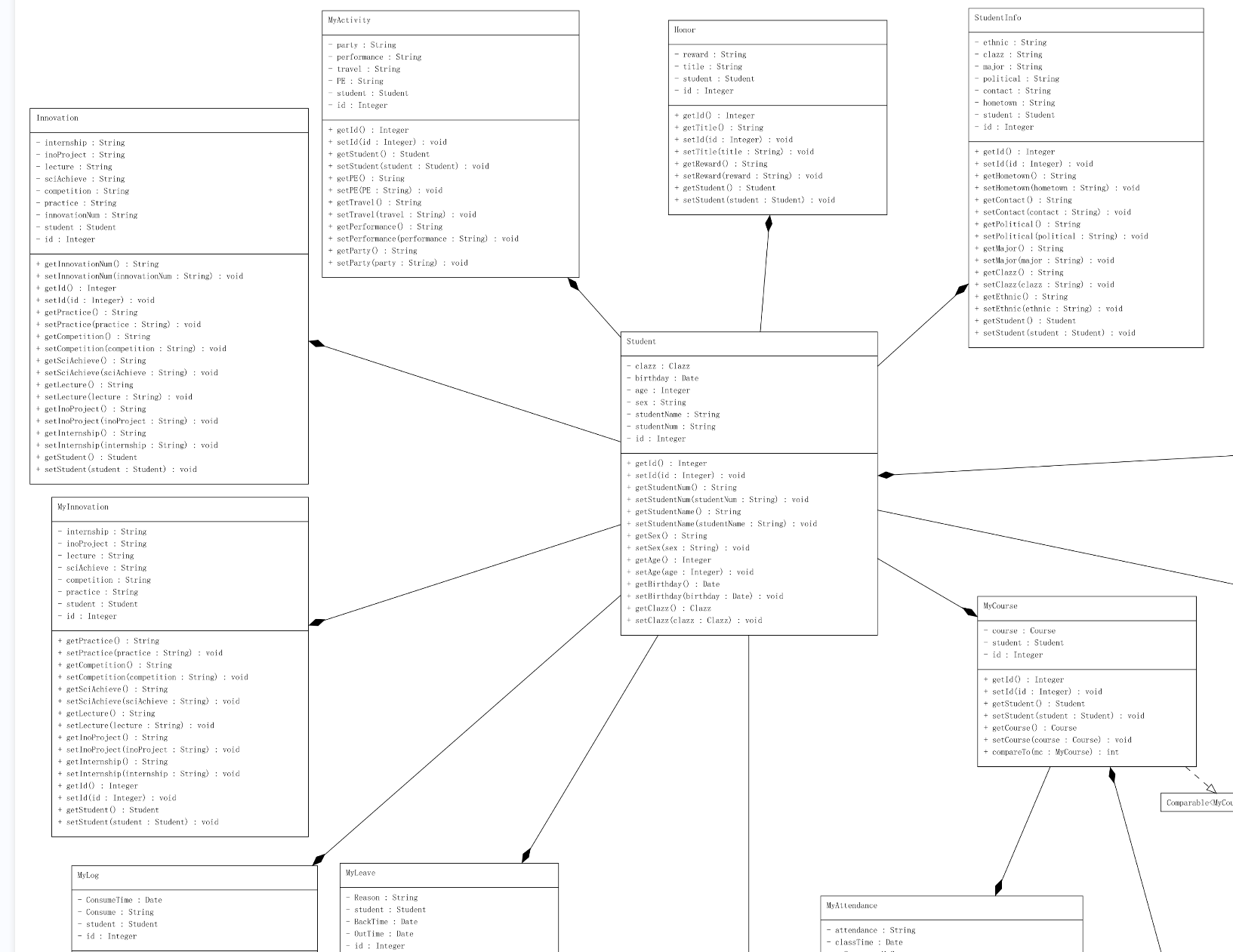
2.1.1 系统面向对象

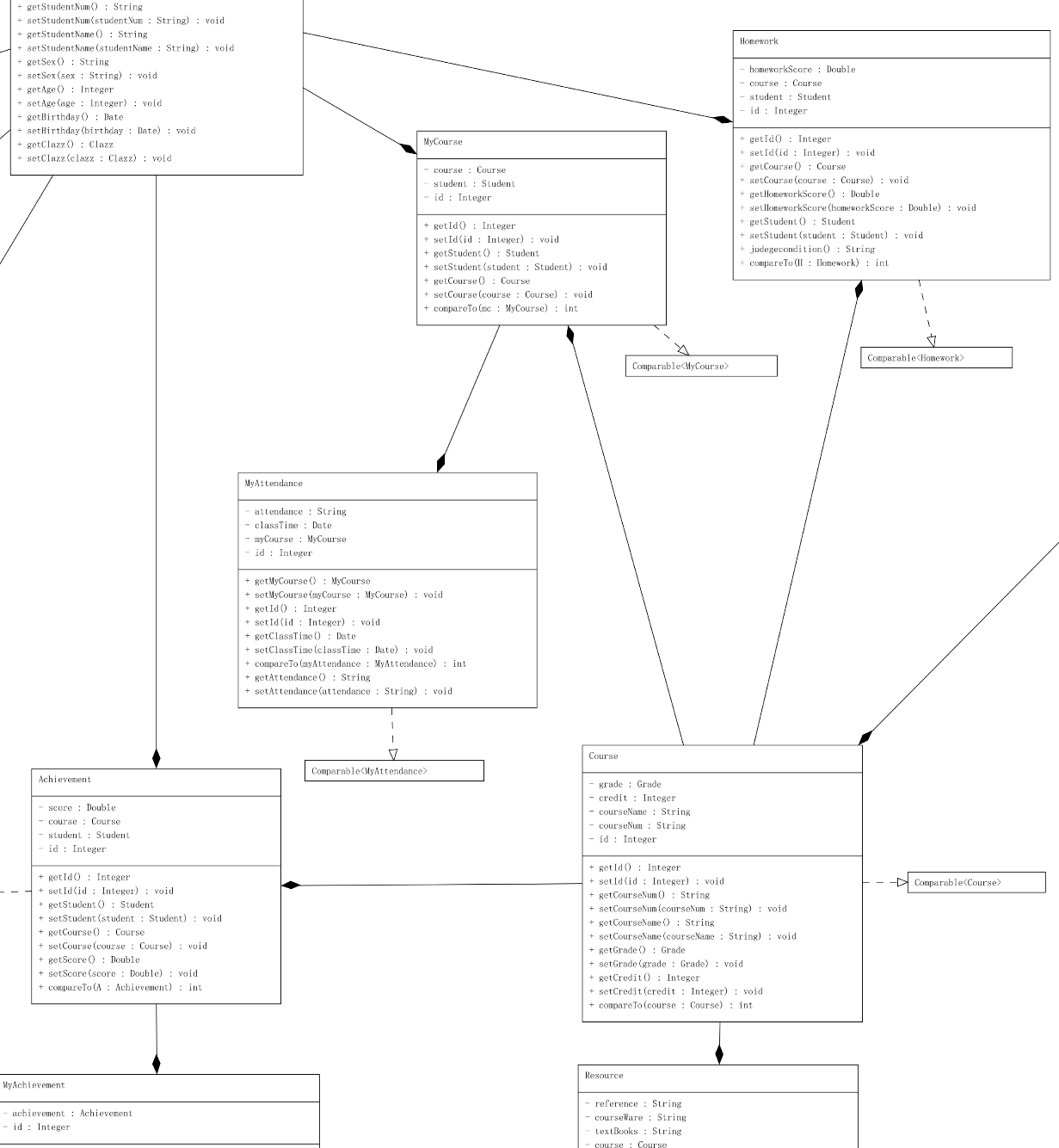
学生信息管理系统，应该是以教师\管理员为用户主体，以学生实体为主要管理对象的系统。而在一般建制下，大学生会被分入不同的专业；由于年龄的差异，他们也会有不同的年级；为了便于管理，会被分入不同的班级；每个班级里都存在着适当数量的学生。因此，本系统的需求主干便清晰了，即，专业->年级->班级->学生。如下图所示。主干部分的每一级都以上一级作为外键。



在主干部分完成后，就可以考虑某个学生的信息情况了。一般来说，一个大学生会有这样一些相应的信息：基本信息，如联系方式，祖籍，政治面貌等；学习信息，例如选课信息，考试成绩信息，作业成绩信息，考勤信息等；创新实践信息，例如社会实践，创新项目，实习经历等；荣誉信息，例如获得的荣誉或称号等；日常活动信息，例如体育活动，文艺演出等；日志信息，如外出请假与日常消费等。

以上描述的这些信息实际上是主干外的分支，即围绕专业->年级->班级->学生的最后一环，即学生展开的。因此，在完成这些分支需求时，我们需要将信息管理系统的分支页面绑定到某一指定学生上。如下图所示，以学生实体为中心，其余类均以前者为外键。



但此处需要注意的是，对于一个学生管理系统而言，势必要有一个下属的课程系统。学生的选课、课程成绩、作业成绩等都来源于此。因此，在系统设计时，需要有一个旁支，即课程类，来完成这些功能。同时，这个类的几个下级类需要同时关联该类与学生类。这是因为某一条成绩记录需要知道它的主人（哪个学生）以及它是关于哪一方面的（哪门课程）。如下图所示。

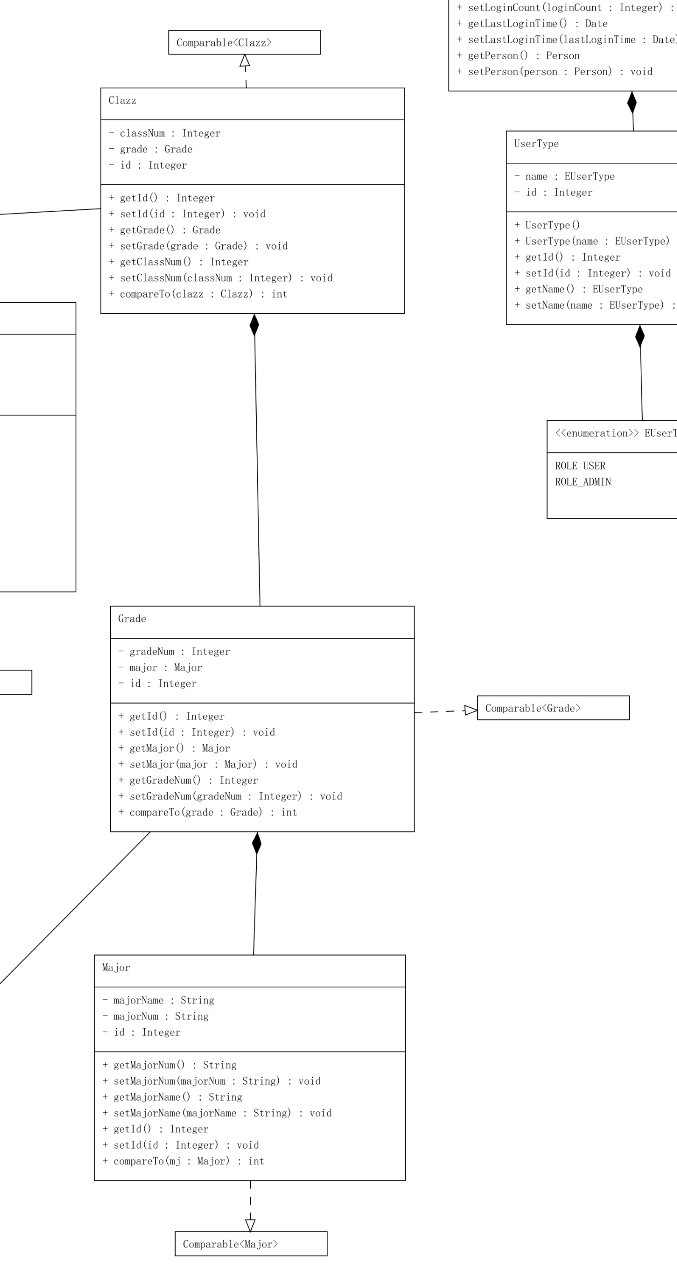
这样整个系统的需求便明确了。附全图如下。



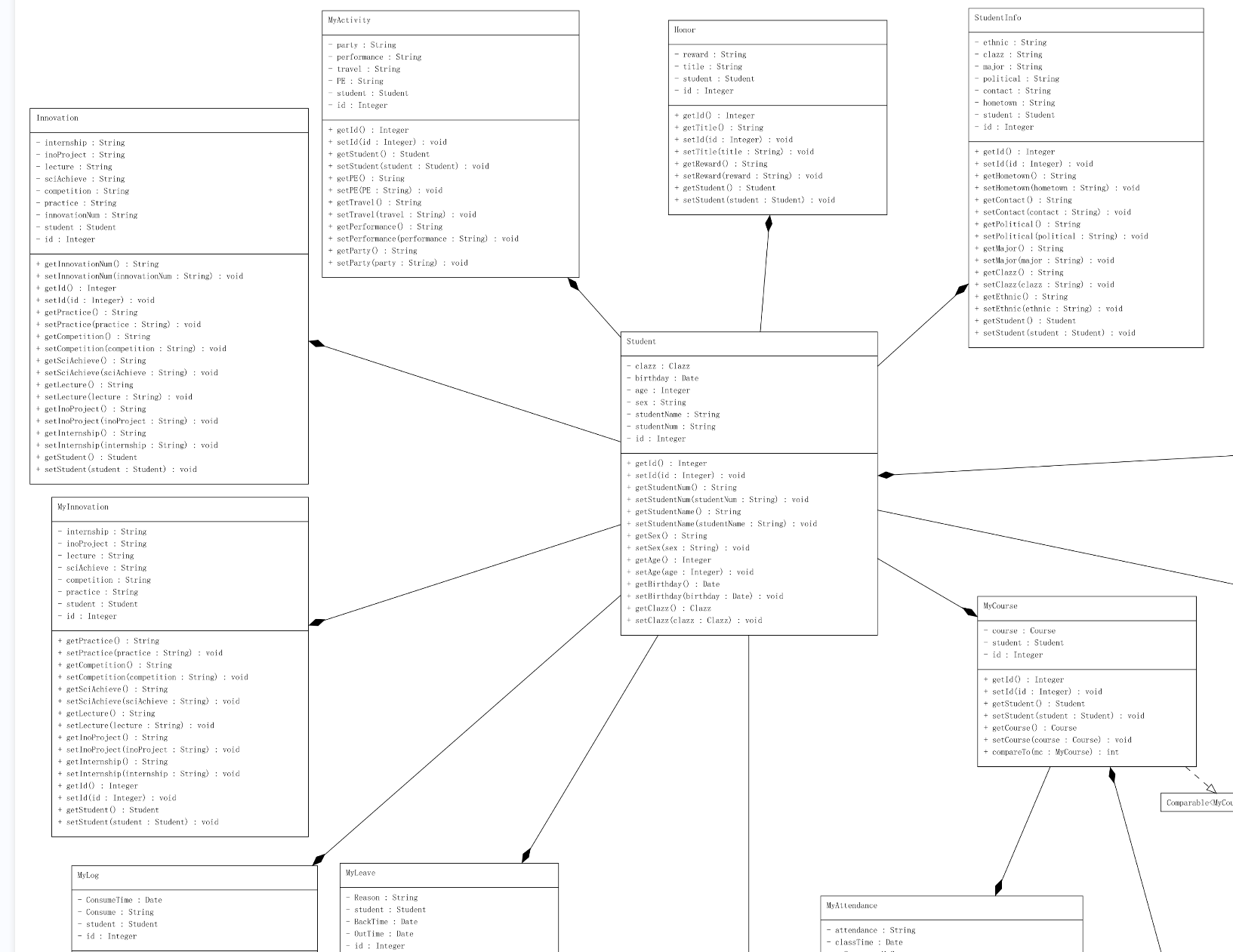
第三章 系统设计

3.1 系统概要设计

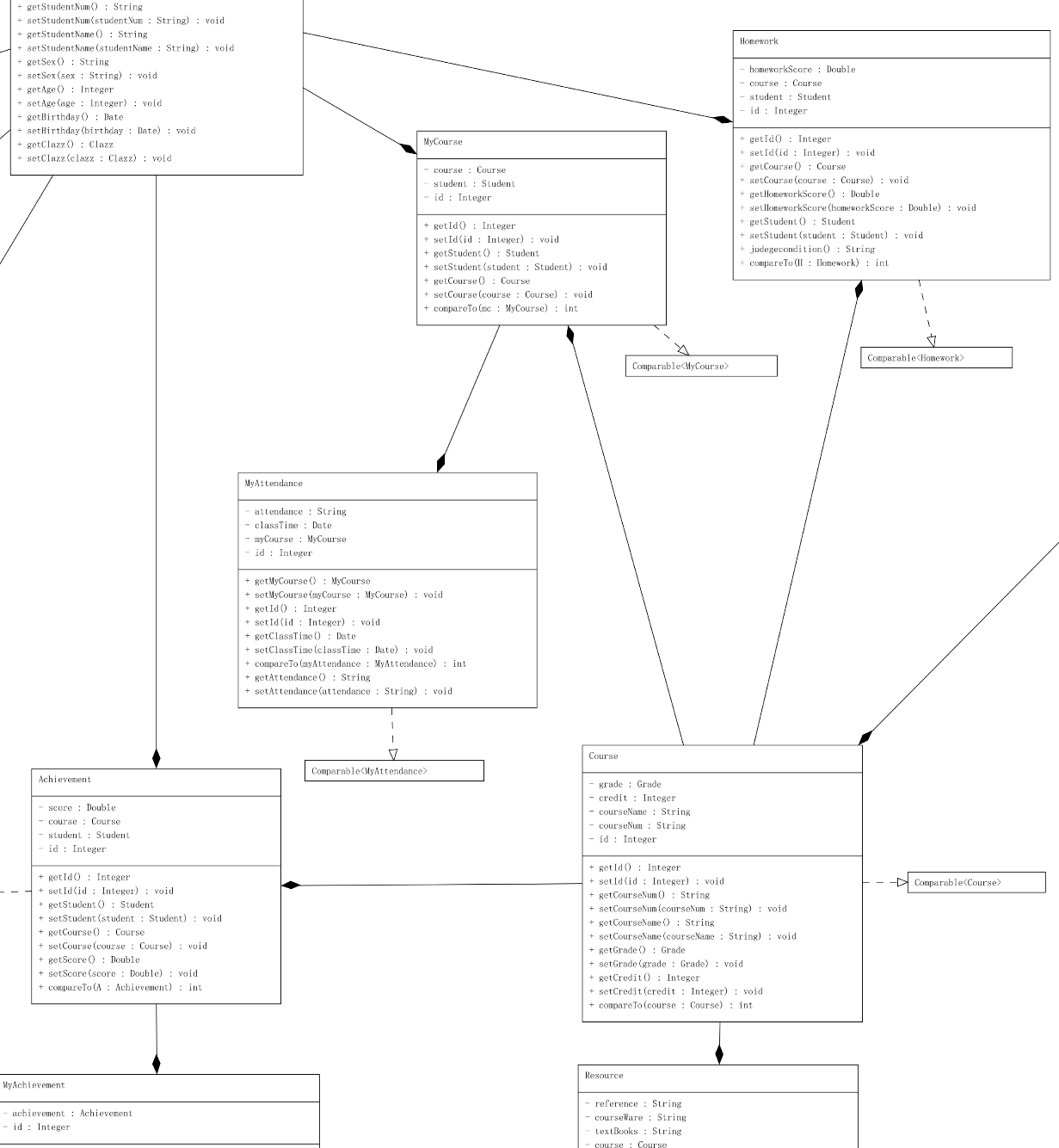
上文提到过，本系统遵循专业->年级->班级->学生的主干，课程为旁支的需求。因此设计上也是依照该架构形成的。首先展示除了统计功能外最基本的功能。



随后，以学生类为中心，完善分支功能。

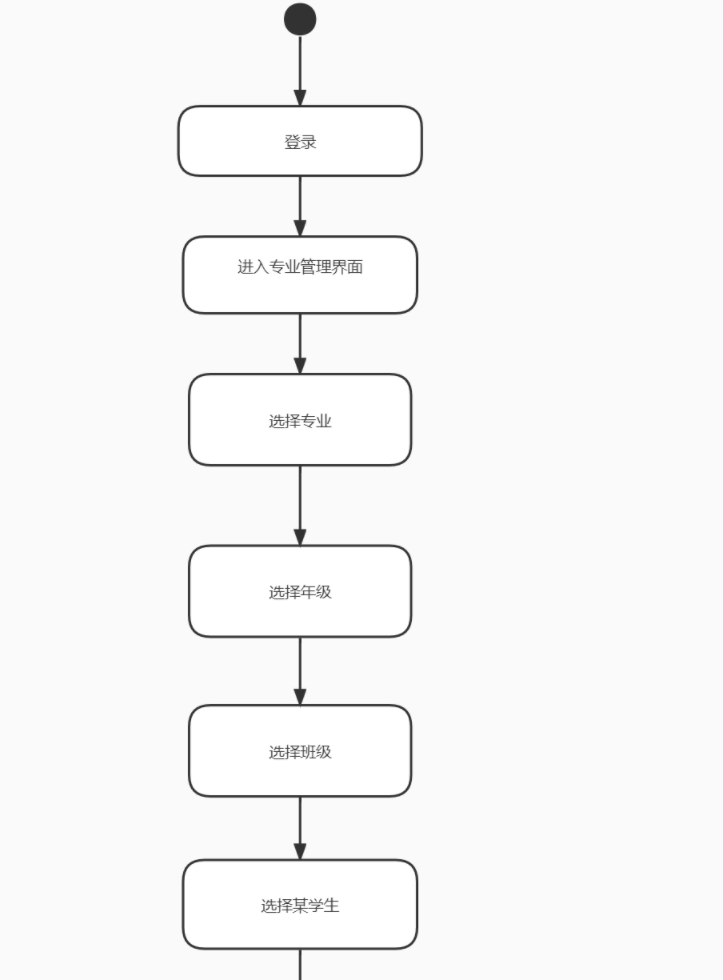


之后，完成课程类旁支的设计，下属类均以学生与课程为外键。

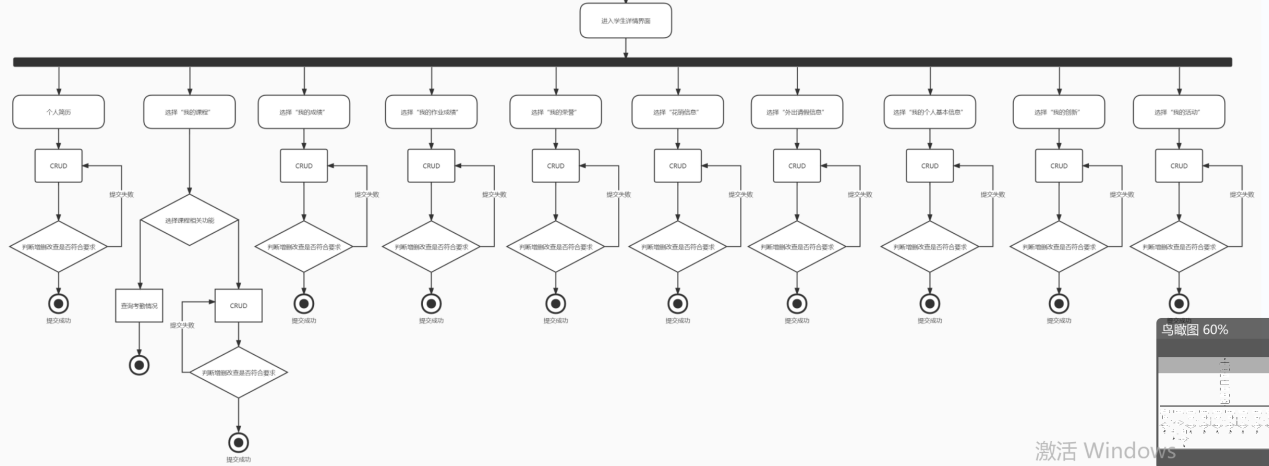


3.2 系统详细设计

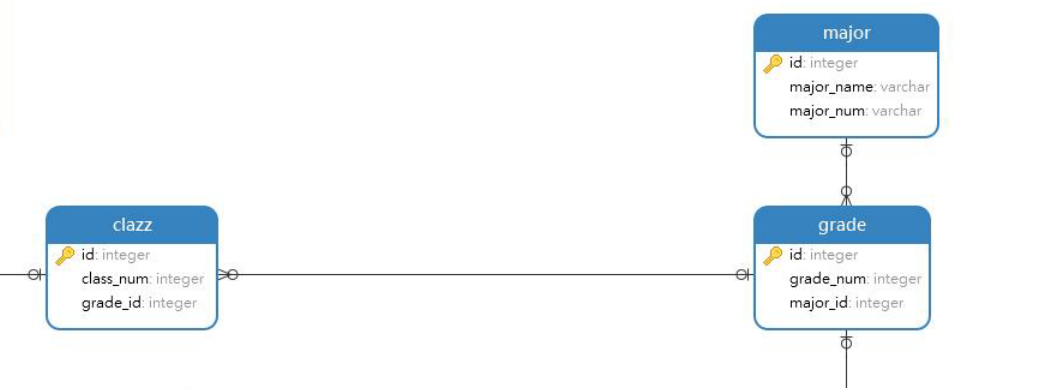
首先是主干部分。如图所示，可以通过页面的跳转在前端获得某一学生的信息。



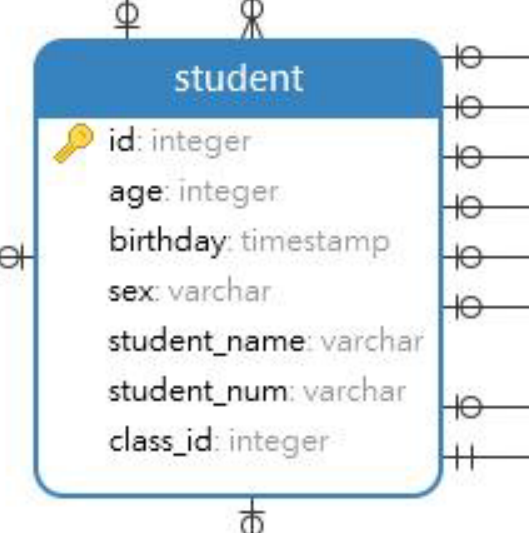
在选择学生完毕后，做一个过渡页，存放所有关于该学生信息的超链接，通过过渡页超链接导航至该学生的其他信息。



这样主干部分的设计就完成了。在主干部分中，专业，年级，班级，学生均是表关联的，除专业外的任何一个都以前者为外键。除外键外，在专业中，涉及专业名majorName与专业编号majorNum属性;在年级中，仅有gradeNum年级编号这一属性；在班级中，仅有classNum这一属性。

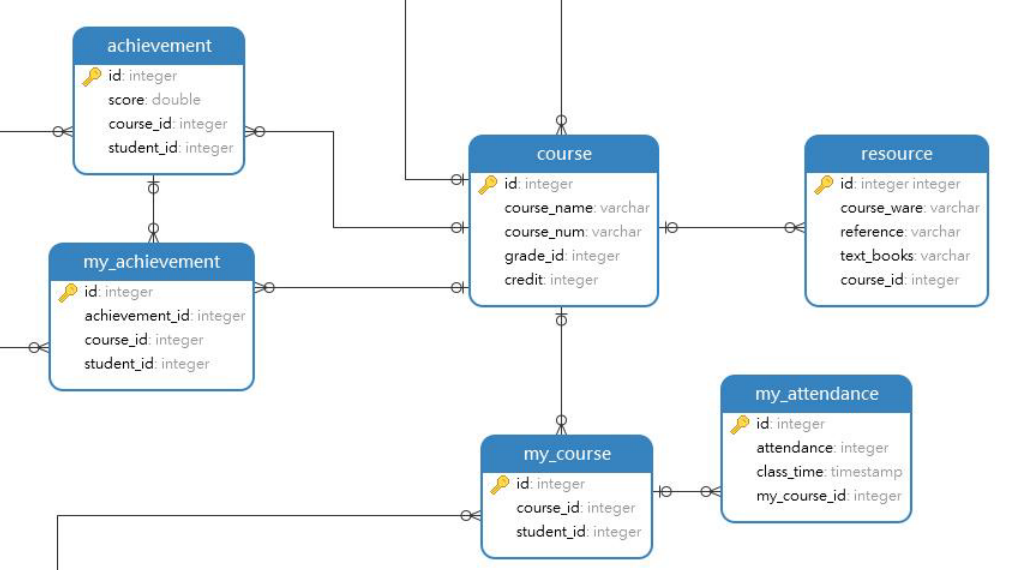


而在学生类中，除外键以外，涉及到学生的一些在校基本信息。例如，学生的姓名，学号，年龄，生日等。如下图所示。



围绕学生展开的详细信息，有学生基本信息StudentInfo,创新实践信息MyInnovation,请假信息MyLeave,日志信息MyLog,荣誉信息Honor,日常活动信息MyActivity。在设计过程中，开发人员已经做到了变量名即注释，这里各属性功能不再一一赘述（否则文章篇幅将过长，超过50页）。

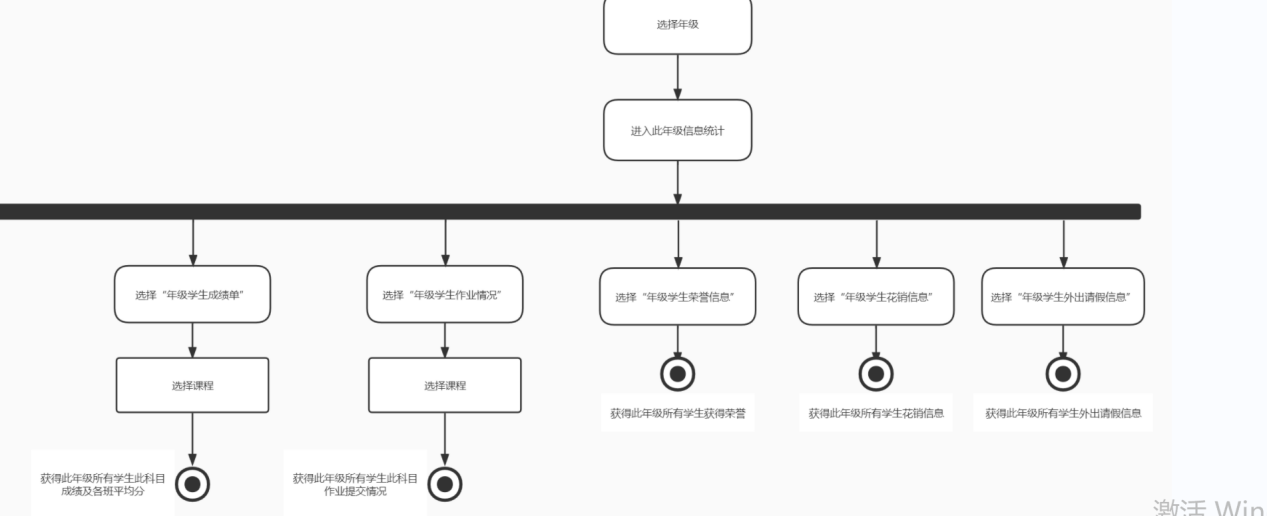
之后完成课程类的设计。课程类的下属类成绩类Achievement,我的选课类MyCourse均为两表关联，即关联学生类与课程类。而MyAttendance作为课程考勤信息，直接关联至MyCourse类，也相当于关联了两张表。



另外的课程资源Resource类由于跟学生实体没有关系，仅关联课程类这一张表。

这样，基本功能的设计就实现了。下面展示统计功能。

在年级页面中，提供了全年级学生各方面信息的统计，如下图。



进入年级页面后，通过一个过渡页，将年级学生所有信息的超链放置在过渡页中，点击后即可进入这一方面信息的统计。

在班级页面中，提供了某一年级所有班级学生各方面信息的统计，如下图。



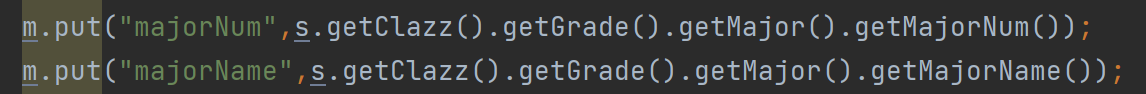
进入班级页面后，通过一个过渡页，将班级学生所有信息的超链放置在过渡页中，点击后即可进入这一方面信息的统计。

1. 系统实现

4.1 主干部分的实现

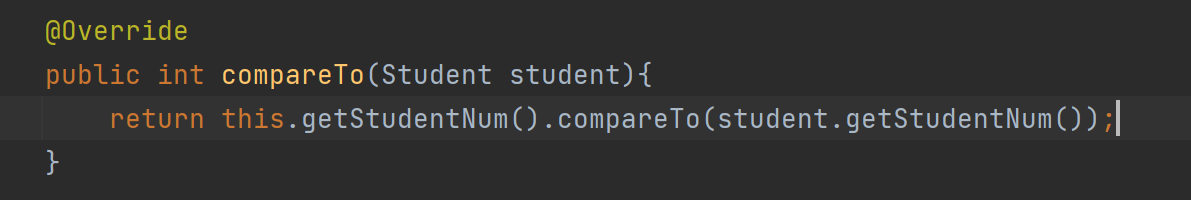
4.1.1 主干部分的实体类

主干部分的实体类包括了Major(专业)、Grade(年级)、Clazz(班级)、Student(学生)四部分。它们按照上文的顺序依次将上一级的对象作为自己的外键属性。例如，Grade实体类中，将Major作为自己的外键。实体类中提供了get方法以供开发人员准确的定位各类对象的信息。例如，如果要确定一个学生的专业，可以对于学生对象进行getClazz().getGrade().getMajor()来获取该对象对应的专业内容。



此外，为了让各实体类对象在前端显示时有序，我们选择在实体类中实现Comparable接口，重写compareTo方法，在controller中利用java.util包下的Collections类，进行排序。







其他部分均为实体类中必要的内容，例如注解、表关联、get、set方法等，不再赘述。

4.1.2 主干部分的控制逻辑

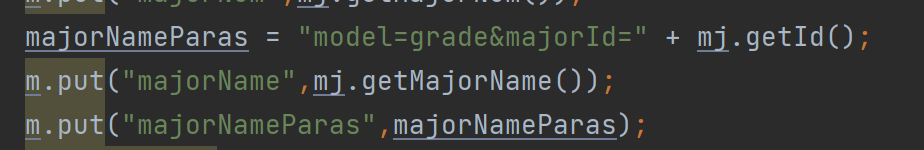
先前提到，在系统设计时，存在这么一条“专业->年级->班级->学生”的主干部分。在应用程序中，体现为同顺序的多个超链页面间的跳转。如下图。





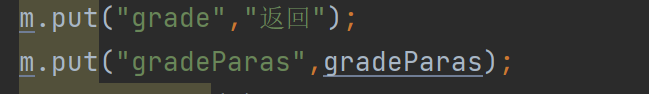
这里只展示其中两个作为样例。这些页面中，超链与提交是主要的技术难点。而在超链中，主要包含了两部分超链：1.进入到下级页面中的超链（例如专业到年级）2.返回到上级页面中的超链

对于第一部分的超链，利用超链字符串，传递下一个页面需要的参数，随后将标签名称与超链字符串一同放入映射集中返回给框架处理



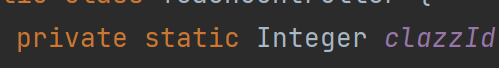
对于第二部分的超链，对于任意一个页面，如果已知它初始化需要的参数，那么我们无论在哪个页面，只要在超链字符串中设置导航到这个页面所需的参数，就可以导航到该页面并完成初始化。而超链的标签其实只是一个形式，是可以自由选择的，因此我们可以选择令其名为“返回”，这样在形式上就实现了返回超链。

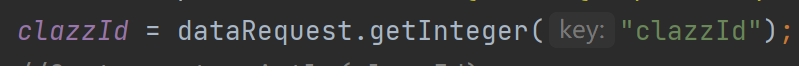




（以上代码截图用于从班级界面返回到年级界面）

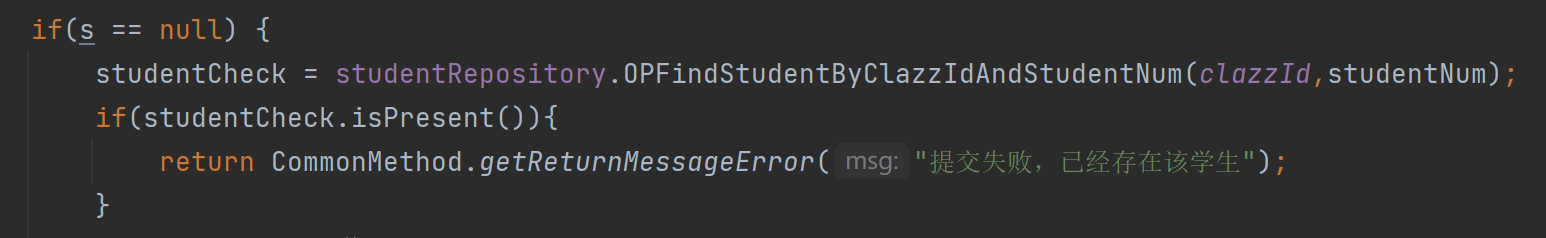
这里还要说明一个保存数据的方法。例如我们在做Student对应的页面时，在提交方法中，需要查询某一班级下的某个学生是否存在来做判重。这时查询就需要联合班级与学号进行查询。但是在编辑或添加学生时，我们仅输入学号，而不会输入班级，则打包回来的数据里没有班级的信息。那么我们怎么获得当前学生的班级信息呢？这里选择利用全局变量实现。在页面初始化时，由于该页面是利用超链导航的，因此，它的上级页面会向它传递一个页面初始化需要的参数，也即，班级页面会向学生页面传递班级的主键id。那么我们在接收这个参数的时候，开一个全局变量，起到延长数据生命周期、保存数据的作用，以便后续的使用。





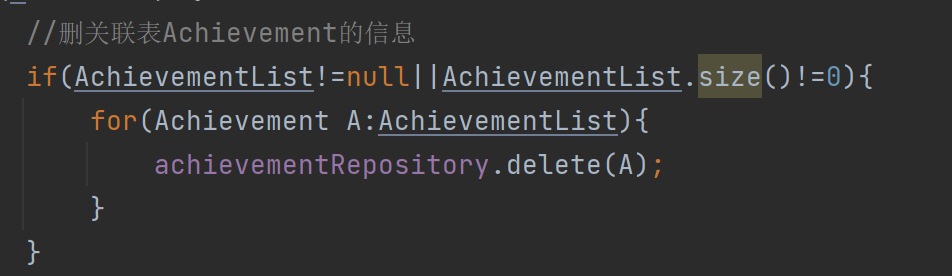


此外，在提交方法中，由于我们输入的数据时输入的是学号，但是一个班级不会有两个学号相同的人，因此我们要做判重。这里的思路是：接收完前端打包传回的参数后，如果id不为空，根据id查询，将查询结果通过.get方法赋值给最终的目标对象。随后，如果对象为空，通过全局变量clazzId(班级id）和前端传回的studentNum（学号)进行联合查询，如果存在该对象，返回”提交失败“的信息。否则，进行手动主键自增。之后，set各属性，最后调用save。

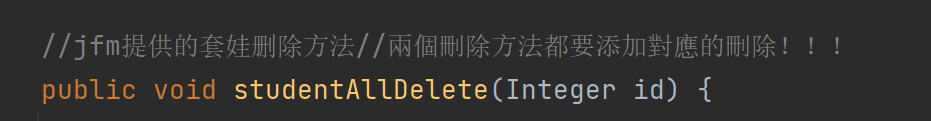


联合查询进行判重是因为，一个班级不能同时存在两个学号相同的学生。

最后，对于删除而言，如果一个学生被删除，那么这个学生对象在数据库表中的其他相关数据也要全部予以删除。这个道理同样适用于专业、年级、班级等实体类。



为了便于编写主干部分对下级数据的删除，在每个controller中都提供了一个封装性良好的“套娃”删除方法。



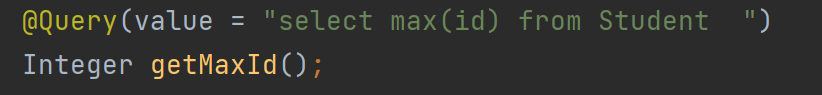
4.1.3 主干部分的数据表维护与查询

鉴于主干部分都各自对应了一个实体类，它们会分别在数据库中新建自己对应的表。于是在DAO层我们需要为每一张表设置增删改查的方法。其中“增”与“删”，我们利用JpaRepository中自带的save与delete方法实现；“改”在controller中利用set方法操作修改对象属性实现。此两者不再赘述。

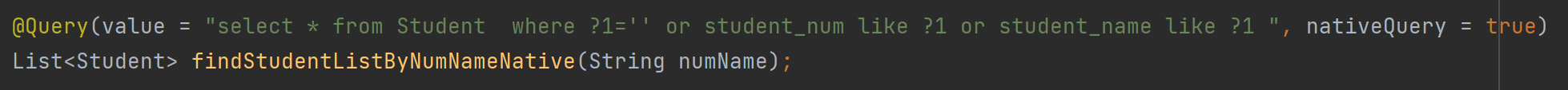
而对于“查”，需要开发人员自行根据需求，于repository包下新建自定义接口，继承JpaRepository，利用命名规范或编写SQL或JPQL语句来实现数据库表查询。

此处以学生实体类(Student)的对应接口为例。

由于我们使用的嵌入式数据库软件SQLite并没有主键自增功能，因此无法利用实体类中的主键自增注解来实现该功能。于是在接口中利用JPQL提供获取当前表中最大主键Id的方法，查询得到后在controller中手动设置主键值。

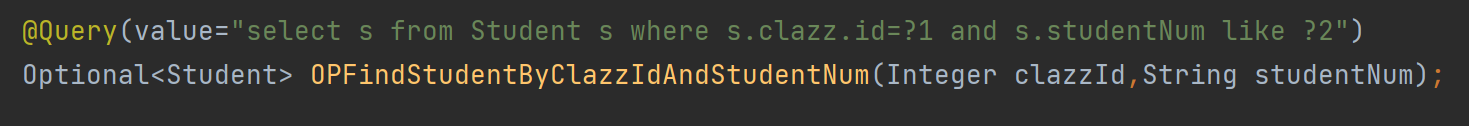


有时我们想通过某一个或多个属性来查询对象在数据库表中的信息，这时利用命名规范去查询可能略显乏力，可以选择用SQL语句或者JPQL查询。



这里是根据学生的学号或者姓名进行查询的方法。”?1=’’”代表可以进行缺省查询。nativeQuery=true代表这是SQL原生语句。

或者我们想通过多个条件进行联合查询，同样地，利用SQL或JPQL查询。



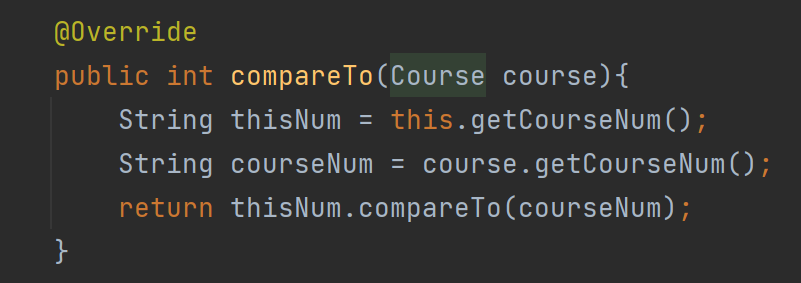
这里是利用JPQL进行查询的方法。JPQL一般不会用select，除非指定了对象。这里s就是指定的对象，”from Student s”意味着从Student实体类对应的表进行查询。”s.clazz.id”与”s.studentNum”代表这个学生的属性（或属性的属性）。”?1”和”?2”代表这是有两个形参的方法。

4.2 课程部分的实现

4.2.1 课程部分的实体类

课程部分的实体类包括Course(课程)与MyCourse(我的课程)。其中，Course代表了某一年级下的所有课程，因此，它包含了Grade的外键。而MyCourse是某一学生的所有选课情况，因此，它包含了Student与Course的外键。此外，为了使得前端显示的课程有序，类似地，实现Comparable接口，重写compareTo方法，在controller中Collections.sort。







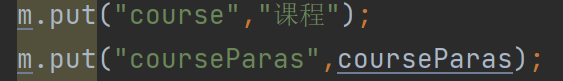
MyCourse也是如此。

其余代码包括注解、表关联、get\set方法均为实体类必要的内容，不再赘述。

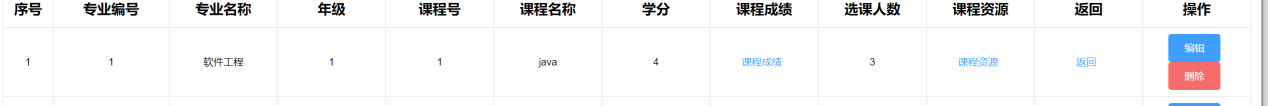
4.2.2 课程部分的控制逻辑

鉴于Course实体类代表某一年级下所有课程，本系统在设计时将其对应的页面超链置于年级页面下，便于查看该年级所有课程。这一点与其Grade外键不谋而合。

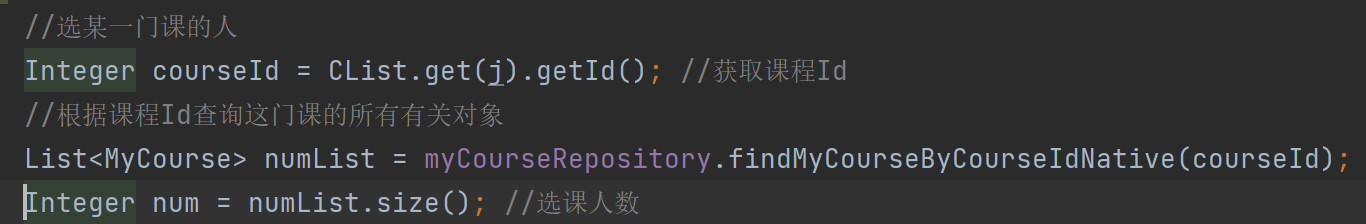




在课程界面中，包括了一些与课程相关的超链。例如，课程成绩、课程资源等，这些将在其他的模块中予以介绍。



此外，该界面还实现了一个统计功能，即选课人数。这是利用MyCourseRepository实现的。实际上，如果要统计选课人数，就是统计在“我的选课”这张数据库表中有着对应课程id的数据数量。根据MyCourseRepository提供的查询方法，查询出所有符合的数据，组成的List的size就是选课人数。

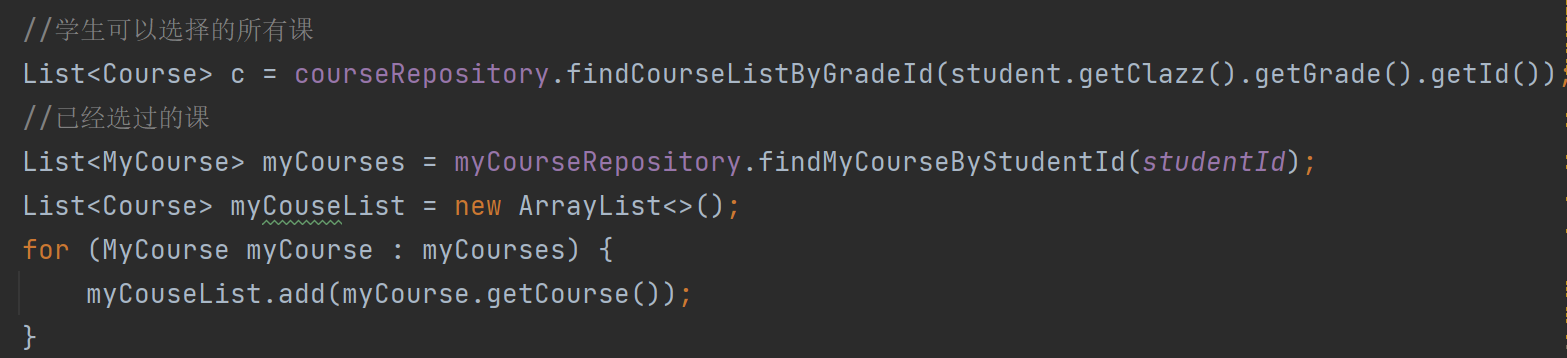


Course方面的控制逻辑与上文中的Student的控制逻辑大同小异（仅是变量名与超链的区别），这里不多阐述。下面来介绍MyCourse方面的控制逻辑。

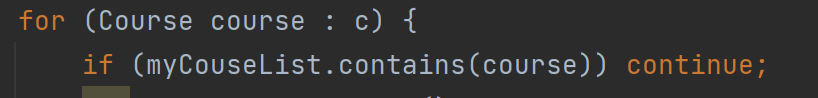
MyCourse的很多部分与Course是类似的，除了编辑页面。在MyCourse的编辑页面中，通过后端代码实现了一个动态的，可供选择的列表。也即，当前学生可供选择的课程只能在该学生所属年级下的所有课程中挑选。并且，一旦一门课程已经被该学生选择，下一次选择中的列表中将不再出现这门课程。如下图。



思路是：首先我们查出所有该学生所在年级的课程（也即他可以选择的所有课程），然后，根据全局变量studentId查出该学生所有选过的课程。随后遍历第二个List，如果第一个List中包含了该值，那么跳过（也即，该学生已经选择了这门课程，那么他可以再选择的课程中就应该忽视这门课）。



这里的手动复制List是因为类型不一样，需要Course类型的List。



调用java.util.List中提供的contains方法判断是否选择过该课程。

做出动态列表的好处在于：1.更加直观，用户体验更佳 2.后端无需判重，因为动态列表里不会显示重复的课程。

MyCourse的其他功能的实现与上述的几个类几乎相同，例如，删除中需要删除关联表的数据。这里就不再赘述了。

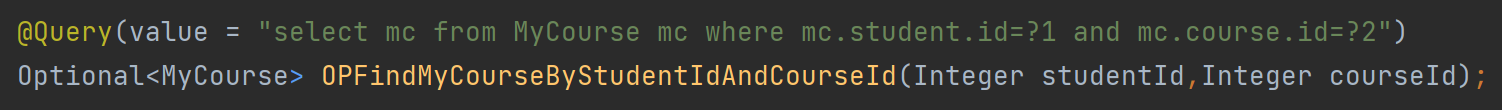
4.2.3 课程内容的数据表维护和查询

在这一方面，实际上，几乎所有类都是相同的。 采用的方法均为继承JpaRepository，随后利用SQL或者JPQL自定义一些必要的方法。这里仅就几个方法进行展示。

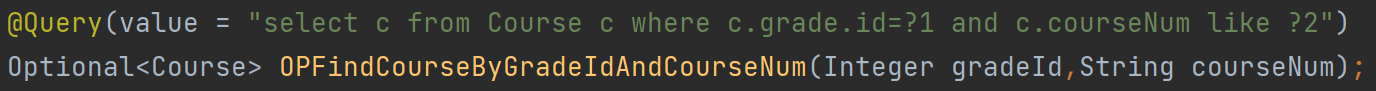




主键自增需要的方法：获取当前主键最大值。



联合学生主键id和课程主键id进行查询。由于这样可以唯一确定一条数据，因此数据量至多为1，返回值为Optional。



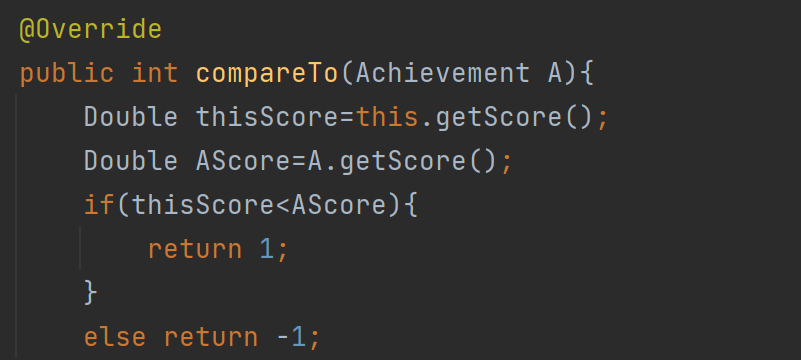
联合年级主键id和课程号进行查询。同样地，结果具有唯一性。

4.3 学生成绩部分的实现

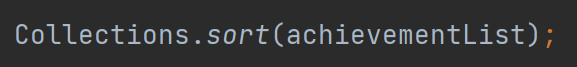
4.3.1 学生成绩部分的实体类

学生成绩部分包括的实体类仅有一个：Achievement(成绩)。但是这个类衍生出的controller数目相当客观。显然，一个学生的成绩由多门课程的成绩组成，那么在Achievement类中包括了Student和Course的外键也是很自然的。此外，为了使得前端显示的数据有序，按照分数大小进行排序。





这里return的1和-1是反的，因为开发人员想让分数高的人排在前面。



4.3.2 学生成绩部分的控制逻辑

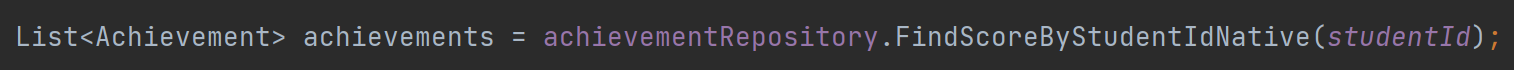
学生成绩部分被分为了几个模块。一个是统计了指定年级中指定课程的所有学生成绩的CourseAchievement；再一个是统计了指定班级中指定课程的所有学生成绩的ClassStudentAchievementCourse；还有一个是统计了指定某一学生的成绩的MyAchievement。此处先从最后一个讲起。

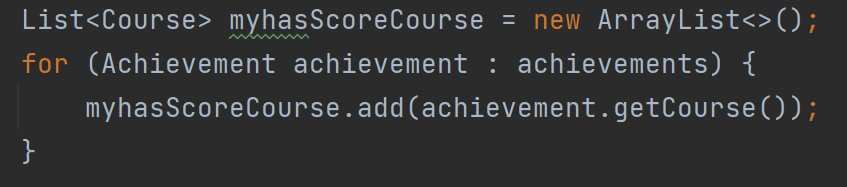
在MyAchievement(我的成绩)中，类似于MyCourse(我的课程)，开发人员希望做出有后端代码控制的动态列表。即，如果一个学生选择的某一门课程已经提交了一个成绩，那么在编辑或者添加时，动态列表中不会再出现这一门课程。如下图



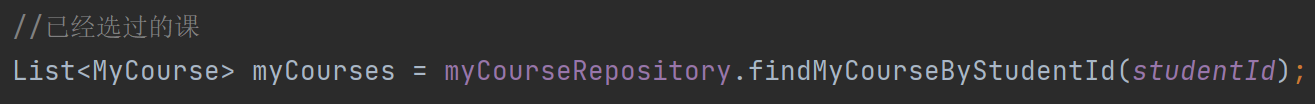
思路是：分类讨论。

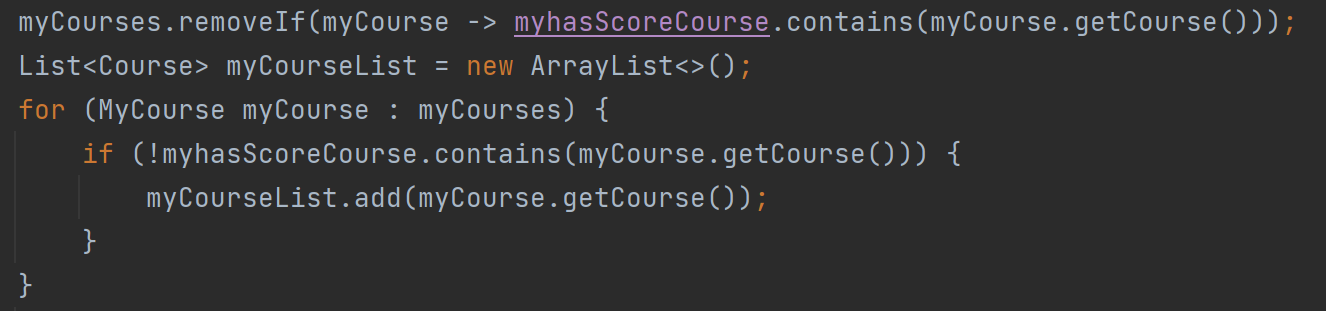
第一种情况是编辑，先根据全局变量studentId查出所有这个学生已经有的成绩。然后查出这个学生所有的选课，取差集，则该差集显然为该学生没有成绩的课程，动态列表即为这些课程组成的List。





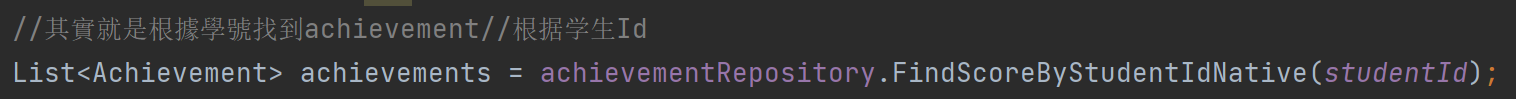
手动复制一遍是因为类型不同，需要Course类型的动态列表。

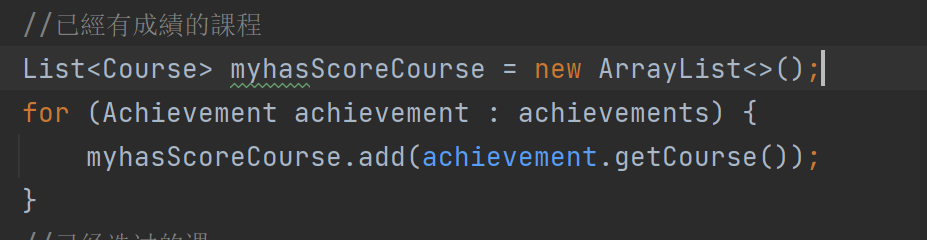




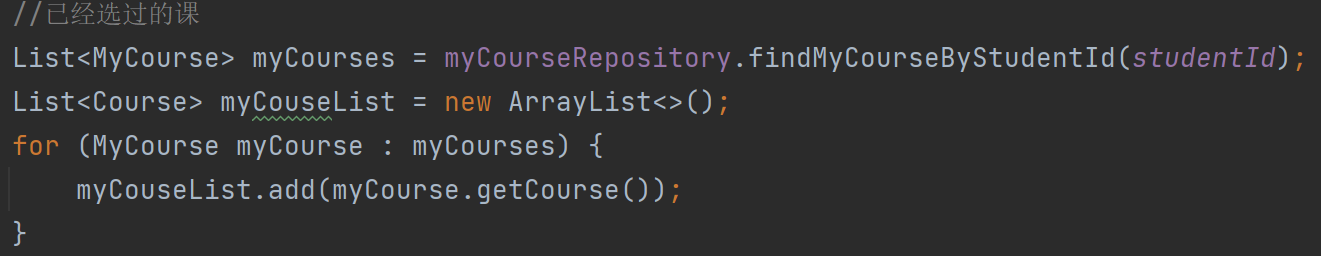
上面那条是用lambda表达式取了差集。或者像下面循环中编写的一样，遍历List的同时，判断是否已经包含了该课程，没有包含再进行手动复制。

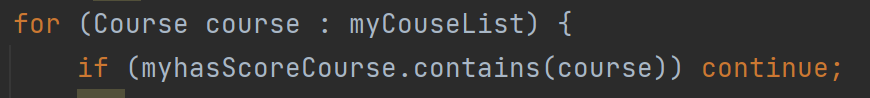
第二种情况是添加，这种情况和MyCourse的动态列表相似。即查出所有某一指定学生的选课和所有的成绩数据，遍历前者，如果后者不包含前者中的某一元素，再添加到最终目标List中。否则，跳过。





同样的道理，手动复制是因为类型需要。



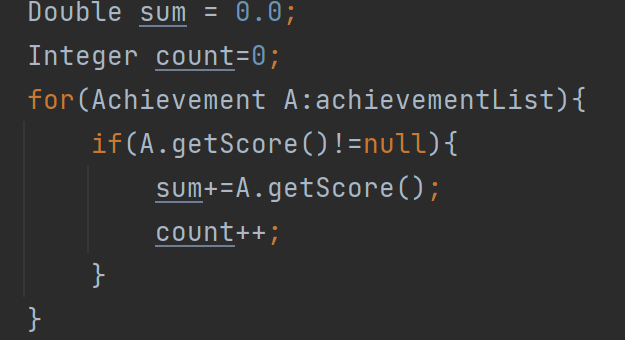


随后的两个模块，实现的方式和思路都是相同的，因此这里为了节省篇幅，选择统计某一指定年级的所有成绩的CourseAchievement(课程成绩)进行展示。

由于某一年级的所有成绩在编辑或者添加时需要知道是哪个班级哪个学生的成绩，因此就不需要动态列表了，而是用户手动键入。因此编辑、提交、删除这些方法和之前提到的处理过程是相同的。这里来展示一下实现的统计功能。

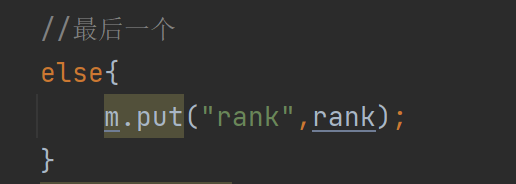
课程成绩的统计功能包括了平均分与排名。思路是：首先，因为Achievement类已经实现了Comparable接口，排序规则是按照成绩高低进行排序，高者在前。因此我们可以直接在controller中sort获取的所有成绩数据。随后初始化所有人的总分sum为0.0，初始化计数器count为0。遍历整个成绩数表，加和sum，count自增。以此求出平均分。而排名，则可以利用数表下标实现。但是考虑到可能有重分的成绩数据，因此排名可能有相同的情况。所以判断本条成绩数据是否比下一条大。如果较大，则下一条的排名需要在先前的基础上加1；否则，保留当前排名。此外还有一些细节处理，比如防止下标越界。

以下为平均分的实现。



以下为排名的实现。





4.3.3 学生成绩部分数据库表的维护和查询

这一模块和之前相同。但是值得注意的是，由于只有Achievement类，所以只需要提供一个接口即可，简化了很多步骤。

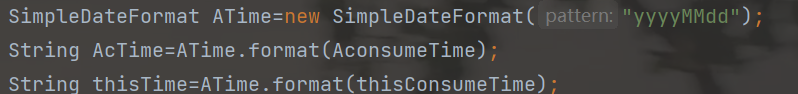
1. 4 学生日志信息部分的实现

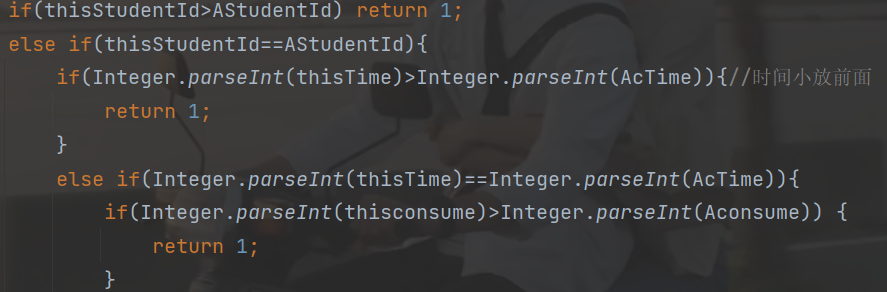
4.4.1学生日志信息部分的实体类

日志信息的实体类为MyLog(我的花销)和MyLeave(外出请假信息)。

1. MyLog实则为所有学生日志信息的总表，因此含有学生外键Student。根据实际逻辑，一个学生一天可有多次花销，所以为ManyToOne。含有属性Consume，ConsumeTime。为了是前端显示的学生花销信息有序，实现Comparable接口，重写compareTo方法，在controller中Collections.sort。为了对应**年级统计，班级统计和学生详情页面**不同的显示规则，**排序规则**为：同年级学生班级号靠前优先，同班级学生学号靠前优先，单个学生花销日期靠前优先，相同日期花销越小越优先。

转换Date为String类型



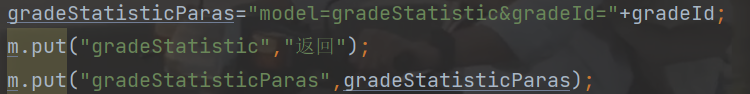


其余代码包括注解、表关联、get\set方法均为实体类必要的内容，不再赘述。

1. MyLeave是所有学生外出请假信息的总表，同理包含了Student的ManyToOne外键。属性有OutTime，BackTime和Reason。类似地实现了Comparable接口，排序规则为：同年级学生班级号靠前优先，同班级学生学号靠前优先，单个学生外出时间越近优先，外出时间相同返回时间越远优先。
2. 4.2 学生日志信息部分的控制逻辑

教师可通过年级信息统计，班级信息统计和学生详情页面通过超链分别查看该年级，该班级和该学生的花销和外出请假信息。精心设计的排序规则使得前端显示更有规律。

通过年级信息统计超链进入到年级学生花销信息超链，对应的是gradeStudentLogController，排序规则不再赘述。显示页面返回做了一个年级信息统计的超链，方便教师返回到前一个页面。





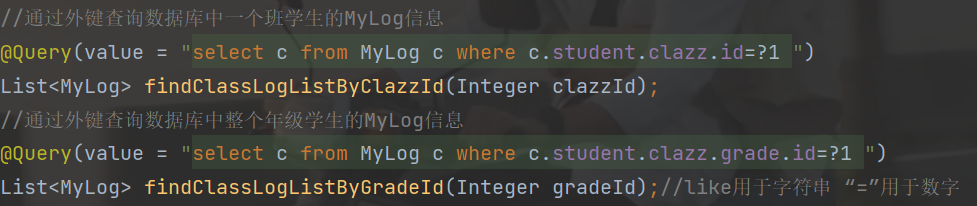
班级信息统计和学生详情页面逻辑同上。

外出请假信息在EditSubmit方法时添加了判断，如果OutTime>BackTime就无法添加成功。



统计页面与详情超链同花销信息。

4.4.3 学生日志信息部分的数据库表维护



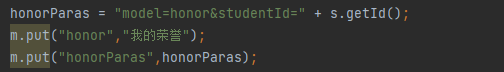
4.5 荣誉管理的实现

4.5.1 荣誉管理部分的实体类

这部分的实体类是Honor，包含了Student外键，代表了每一个学生自己的荣誉信息，在荣誉信息中包括了每个学生获得的称号以及奖励，除此之外，在代码中与其他实体类相同，包括注解，表关联，以及Getter和Setter等。

4.5.2 荣誉管理部分的控制逻辑

鉴于Honor实体类代表每个学生自己的荣誉管理信息，本系统在设计时将其对应页面的超链接放在了学生详情界面下，即StudentDetails，，以便于查找查看。



在前端展示如下



再点击进入以后，展示学生个人荣誉信息，例如荣誉称号，奖励等。



此外，此页面还实现了编辑，删除，添加以及返回功能，在实现编辑以及添加功能的时候，使用了repository中的语句进行查重，防止重复提交已经存在的内容。





其中，编辑和添加功能在使用时，都只显示了称号（title），奖励（reward）的编辑框，而不显示学生姓名、班级、学号等内容，因为编辑和添加功能是在某一确定的学生其详情页下进行的，为防止重复输入已知信息，也是为了避免在一个学生的界面中也能编辑其他学生的逻辑错误发生，这里为了实现功能使用了全局变量studentId,从页面初始界面得到保存以在Edit方法中使用



返回界面同样是使用了添加超链接的方式，在此不再赘述。

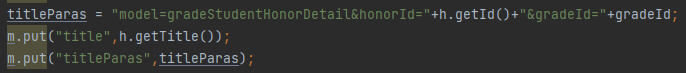
除了以上通过StudentDetails界面进入某一学生的荣誉信息界面以外，还将荣誉信息加入到年级信息统计和班级信息统计中，在这里描述一下年级荣誉信息统计的实现，班级荣誉信息统计同理，只是多添加了班级的筛选条件，以下不再赘述。

进入年级信息统计界面，会出现以下画面，点击年级学生荣誉信息的超链接会显示当前年级的所有学生的荣誉信息。



点击进入后此处只会出现当前年级已有的所有学生的姓名和荣誉称号，这里给荣誉称号添加了超链接，可点击进入，进入后显示的是当前年级所有获得此称号的学生详细信息，此功能的实现调用了honor repository中的查询方法，通过联合t称号（title）和Grade Id进行查询

超链接如下:



前端演示如下:





在详细信息中依旧使用了全局变量honor Id和grade Id来实现编辑和添加界面的简洁便利，

使使用者不需要多次输入已知的重复信息。

在以上的两个界面中都添加了超链接来返回到上一页面。

4.5.3 荣誉信息的数据表维护和查询

采用的方法均为继承JpaRepository，随后利用SQL或者JPQL自定义一些必要的方法，这里仅展示一个方法，其余的查询都只是替换了查询条件，语句类似。



4.6 课程资源部分的实现

4.6.1课程资源部分的实体类

课程资源部分包括的实体类只有一个：Resource。每一门课程有自己的课程资源信息，于是在Resource中添加了course外键，课程资源信息包括课本，课件，以及参考资料的信息，除此之外，在代码中与其他实体类相同，包括注解，表关联，以及Getter和Setter等。

4.6.2 课程资源部分的控制逻辑

鉴于Resource实体类代表每个课程的荣誉信息，本系统在设计时将其对应的超链接放在了课程界面下，即Course，以便于查找查看。



点击进入后，会看到当前课程的所对应的课程资源信息，同样在这个界面设置了编辑、添加、删除、以及返回按钮，在实现编辑以及添加功能的时候，使用了repository中的语句进行查重，防止重复提交已经存在的内容。





其中，编辑和添加功能在使用时，都只显示了课本（textBook），课件（courseWare），以及参考资料（reference）的编辑框，而不显示课程名，课程号等内容，因为编辑和添加功能是在某一确定的课程页下进行的，为防止重复输入已知信息，也是为了避免在一个课程的界面中也能编辑其他课程资源信息的逻辑错误发生，这里为了实现功能使用了全局变量courseId,从页面初始界面得到保存以在Edit方法中使用。



返回功能即在HonorController中的getMap方法加入超链接即可实现，此处不再赘述。

4.6.3 课程资源部分的数据库表维护和查询

这一模块和之前相同，采用的方法均为继承JpaRepository，随后利用SQL或者JPQL自定义一些必要的方法，这里仅展示一个方法，其余的查询都只是替换了查询条件，语句类似。



4.7.1 创新实践部分与日常活动部分的实体类

设置实体类MyActivity、MyInnovation、Innovation，建立数据库表以存储数据。MyActvity中有PE,performance,travel,party属性，分别代表学生的体育运动，文艺表演，研学旅行和聚会活动；MyInnovation和Innovation两个类中拥有共同的属性practice(社会实践)，competition(学科竞赛)，sciAchieve（科研成果），inoProject(创新项目),lecture(讲座),internship(实习经历)。并设置相应的getter和setter。

注：MyInnovation和Innovation的功能有部分重叠

4.7.2 创新实践、日常活动部分的数据库表维护和查询

在Repository包中添加对应的查找数据库表中数据的方法，通过JPQL和SQL语句来实现

例如：

@Query(value = "select a from Innovation a where a.student.id=?1 ")  
List<Innovation> findInnovationByStudentIdNative(Integer ID);

此方法是根据给定的学生主键Id来查找Innovation表中的对应的Innovation对象，并存到一个List中

还例如：

@Query(value = "select \* from my\_activity where pe=?1 ",nativeQuery = true)  
List<MyActivity> findMyActivityByPE(String PE);

此方法是根据给定的体育运动名称（String类型）来查找MyActivity表中的MyActivtiy对象，存储到List中。

4.7.3 创新实践、日常活动部分的控制逻辑

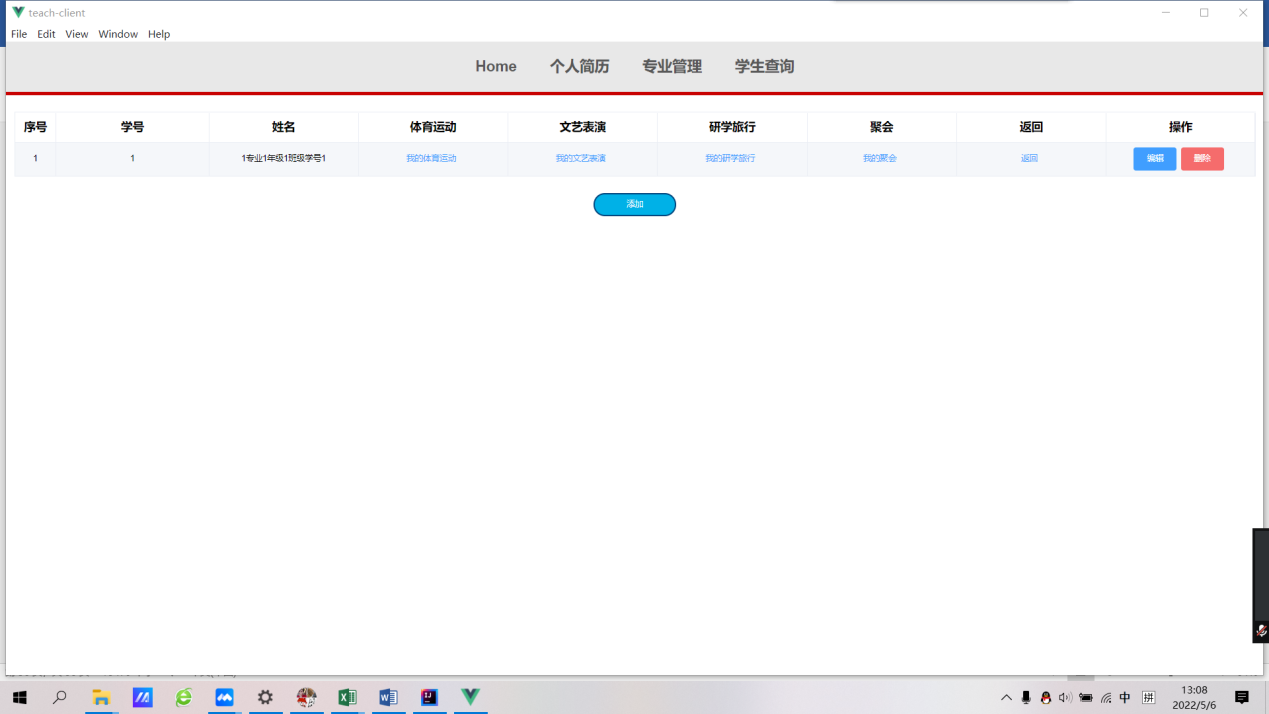


在班级学生的详细信息页面下点击“我的创新”和“我的活动”超链接，将分别进入我的创新和我的活动页面，并将该学生的学号作为参数传给下一个页面。

myInnovationParas = "model=myInnovation&studentId="+s.getId();  
m.put("myInnovation","我的创新");  
m.put("myInnovationParas",myInnovationParas);  
  
myActivityParas = "model=myActivity&studentId="+s.getId();  
m.put("myActivity","我的活动");  
m.put("myActivityParas",myActivityParas);

因为活动和创新两个页面的代码基本相同，在此以“我的活动”页面为例进行说明，而省略“我的创新页面”。

点击“我的活动”进入我的活动页面：



该页面通过调用getMyActivityListByStudentId方法来初始化，显示学号对应学生的各项信息

List<MyActivity> dataList = getMyActivityListByStudentId(studentId);

其中四个属性各分别作为一个超链接链接到对应的页面，该页面还实现了编辑、添加、删除等功能

if (id != null) {  
 actOp = myActivityRepository.findById(id);  
 if (actOp.isPresent()) {  
 act = actOp.get();  
 }  
}

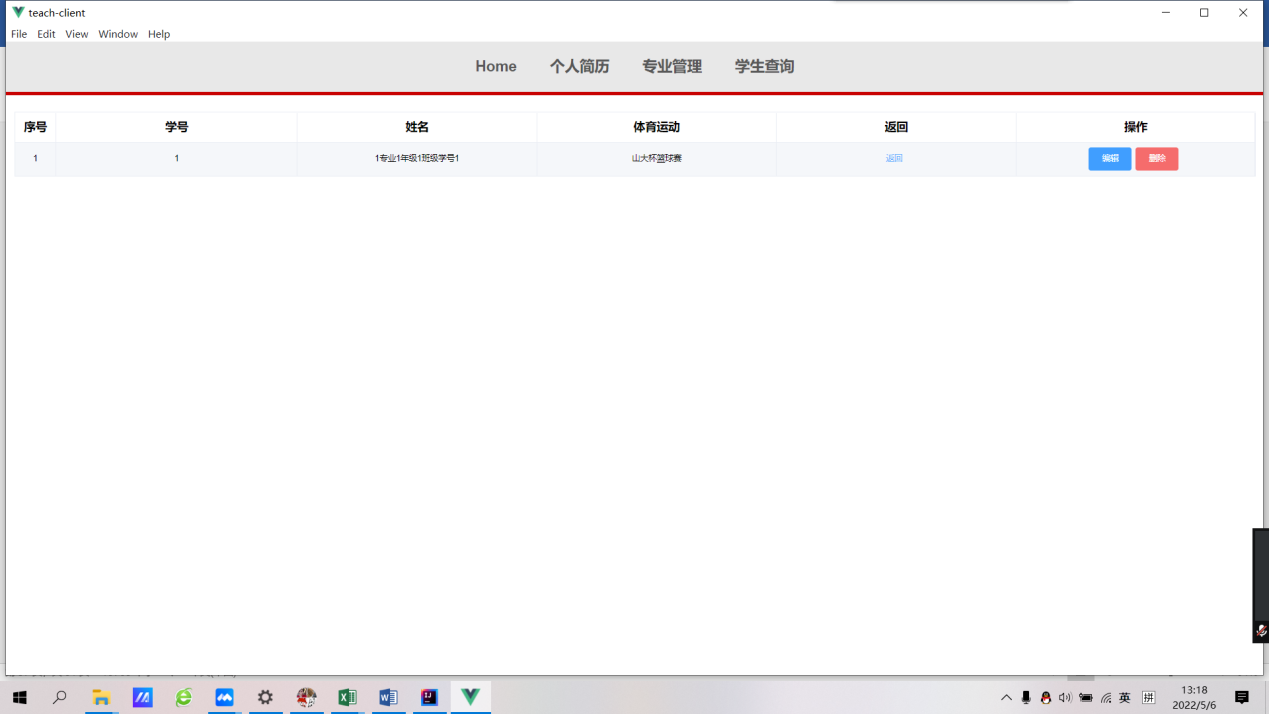
注意编辑初始化和编辑提交方法中要先判断对应的主键Id是否为空，不为空才能根据Id查找对应的MyActivity对象

if(act==null){  
 act = new MyActivity();  
 id = myActivityRepository.getMaxId();  
 if(id==null){  
 id = 1;  
 }  
 else{  
 id = id+1;  
 }  
 act.setId(id);  
}

在EditSubmit方法中还要手动设置主键自增。

四个属性子页面基本相同，在此以体育运动页面为例，其他则不过多赘述。

点击“我的体育运动”进入体育运动界面



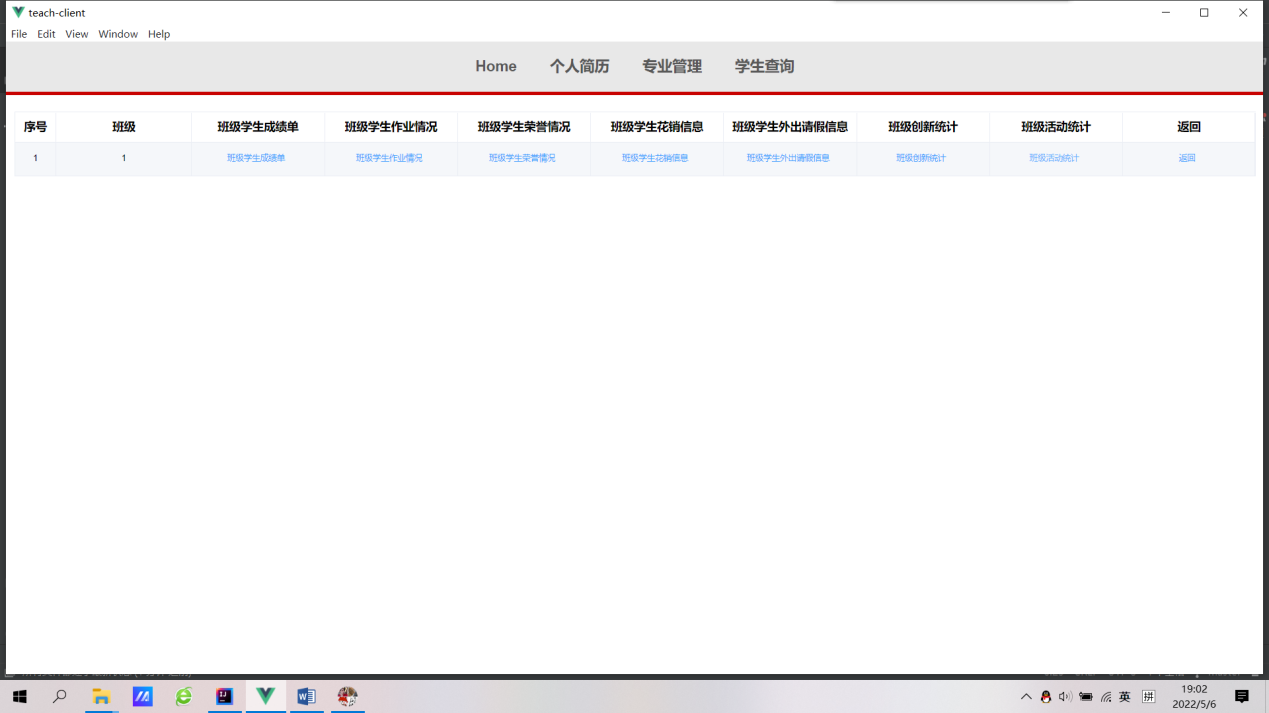
注意在getMyPEMapListByStudentId方法中，要对PE判断是否为空

if(myActivity.getPE()!=null){  
 m.put("PE",myActivity.getPE());  
}

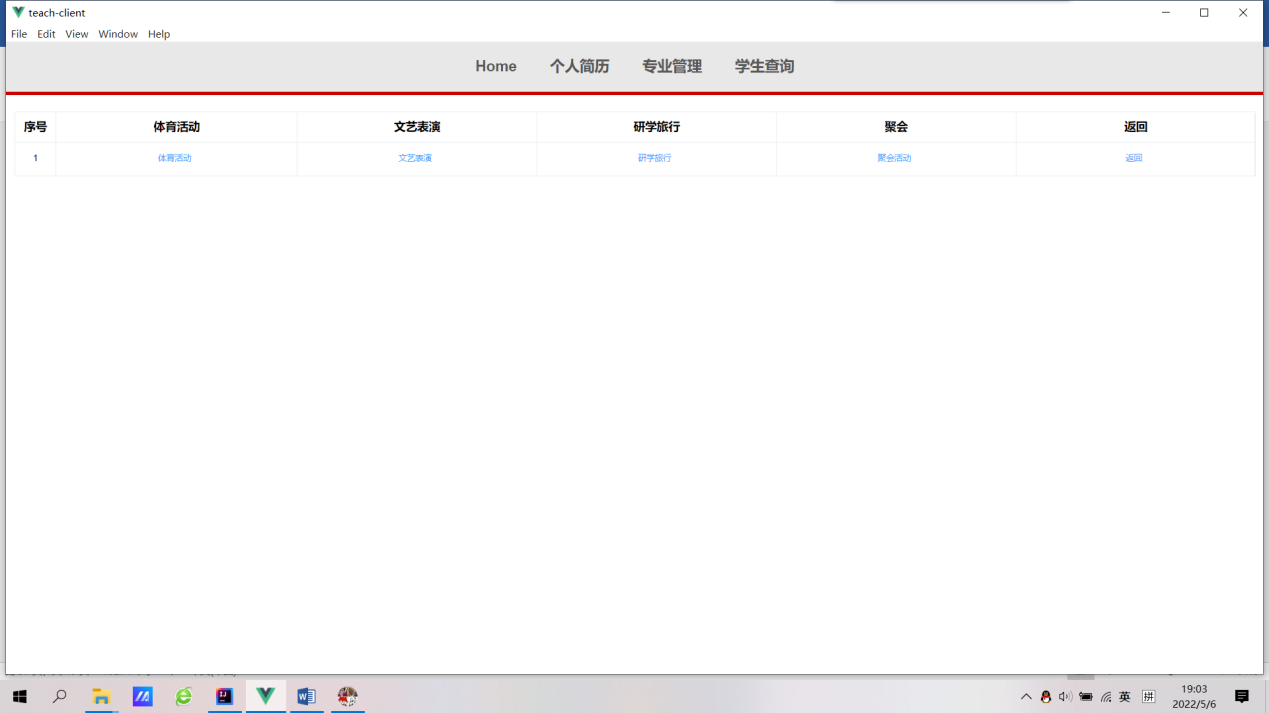
在该页面可以实现编辑学生体育运动的相关信息，删除信息，或者返回“我的活动界面”

* + 1. 班级创新、活动信息的统计功能

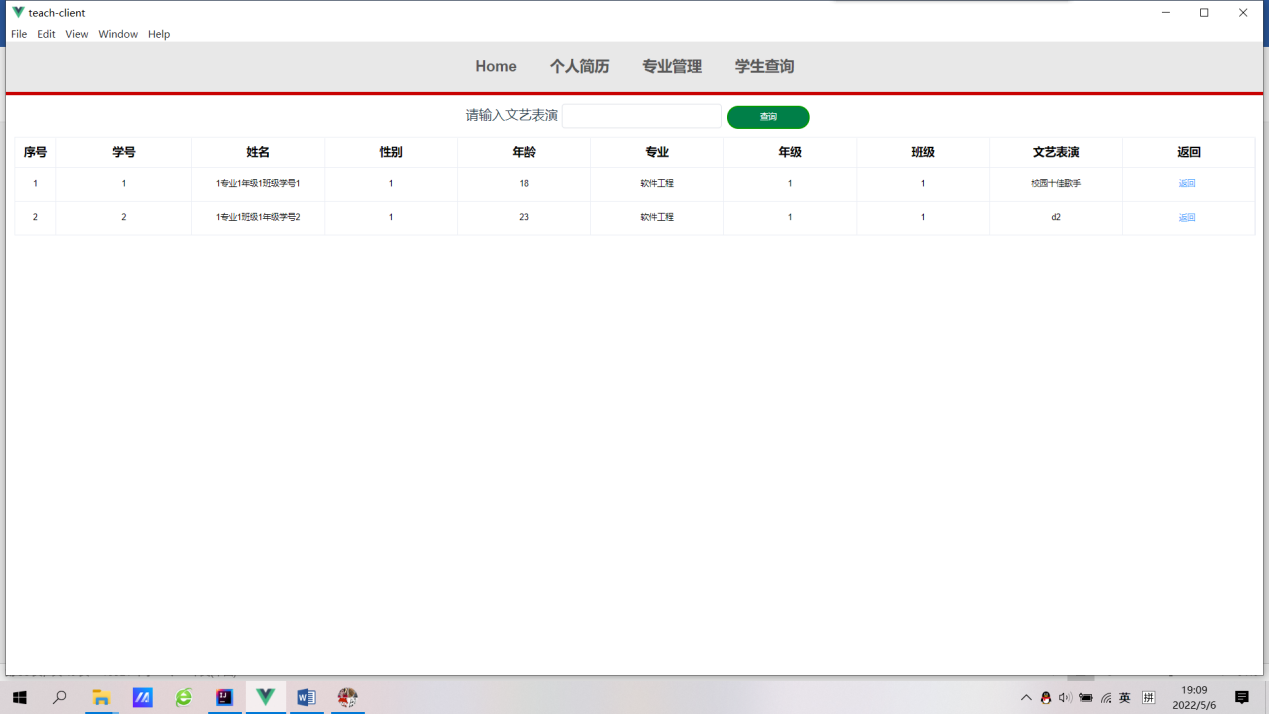
这里以班级活动信息统计功能为例（班级创新信息统计与此相同），为了更切合实际应用的功能需求，我们组考虑到使用此学生管理系统的教师可能会希望查找参加某一特定活动的所有学生对象，并查看所有满足条件的学生的属性。针对这一需求，我们组创建了相应的界面并完成了对应的功能实现.



从班级信息统计界面点击“班级活动统计”进入班级活动统计界面



其中是活动对象的四个属性的超链接和返回界面（返回到班级信息统计页面），此处以文艺表演界面为例，该页面下列出对应班级的所有学生的相关信息，比如序号、学号、姓名、文艺表演等，还有“返回”按钮返回到班级活动页面。



if(performanceList.size()==0){  
  
}  
else{

在该界面getClazzStudentPerformanceMap方法中要对获取的List对象判空

List dataList = getClazzStudentPerformanceMapByClazzId(clazzId);

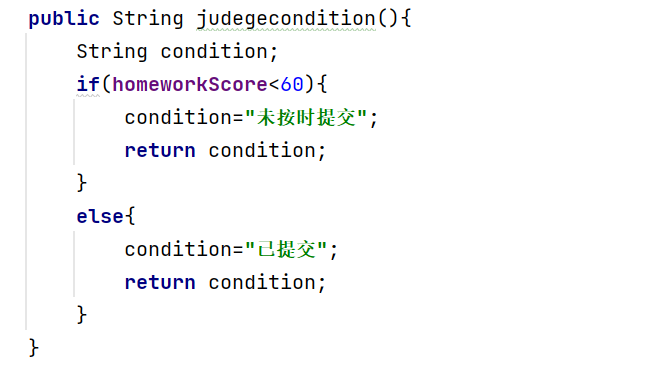
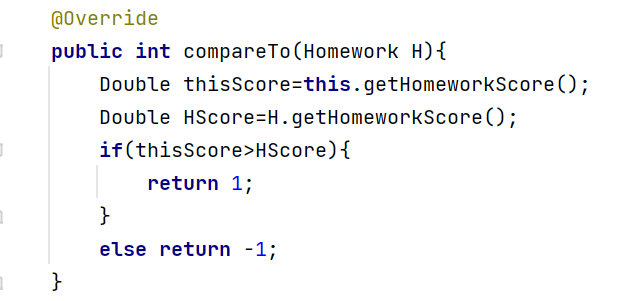
在初始化方法中将班级的主键Id传给getClazzStudentPerformanceMapByClazzId方法，获取dataList对象

4.8 学生作业情况部分的实现

4.8.1 学生成绩部分的实体类

学生作业情况的实体类为Homework(作业)。考虑到作业成绩与学生和学生所选课程两个属性密切相关，所以，在知道学生编号及课程编号的情况下，为了实现访问“某同学某课程的作业情况”采用“student”（学生）,“course”（课程）作为作业情况的外键。同时将“HomeworkScore”（作业成绩）与“submitcondition”（提交情况）作为此实体类的特有属性。

在实体类中实现compareTo接口，并在此基础上创建作业提交情况的判断。



4.8.2 学生作业情况的控制逻辑

学生作业情况部分被分为了三个模块：

1. 统计了指定年级中指定课程的所有学生成绩的GradeStudenHomeworkCourse；
2. 统计了指定班级中指定课程的所有学生成绩的ClassStudenHomeworkCourse；
3. 由学生详情页面导航的指定某一学生的成绩的MyHomework。

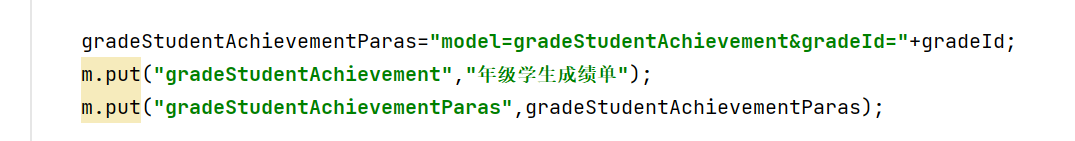
（1）&（2）：由于第一个与第二个逻辑基本一致，这里以GradeStudenHomeworkCourse为例进行展示

在年级统计(GradeStatistic)导航页中设置年级作业情况（GradeStudenHomework）的超链接

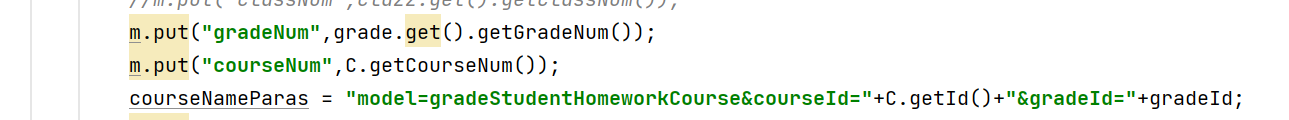
前端：



Controller:



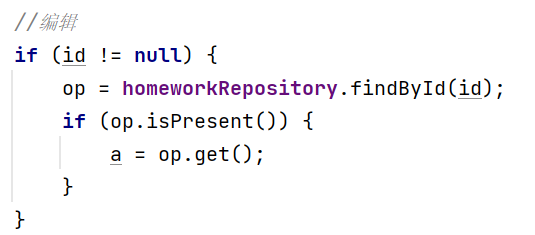
在年级作业情况中选择课程（超链接），将课程编号作为参数传入下一个页面



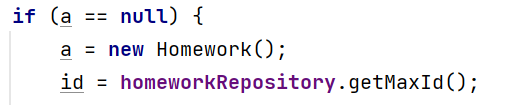
最后，在指定课程中显示统计的本年级此课程作业详情，包含学生作业分数汇总以及提交情况汇总。

（3）在MyHomework(我的作业成绩)中，类似于MyAchievement(我的成绩)，在前端显示某学生所有课程的作业成绩单。这里展示对于增删改查的系统实现。

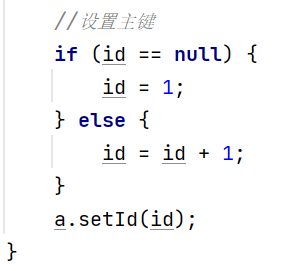
编辑：



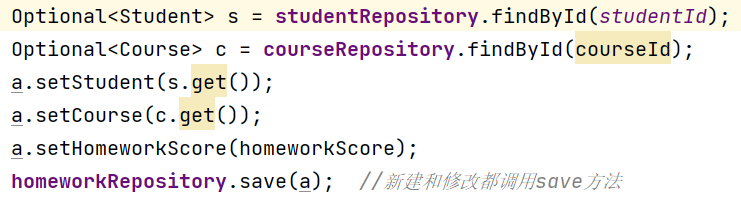
添加：



主键自增：



再进行对应属性的设置和存储。



以下是删除方法：



4.8.3 学生作业成绩部分数据库表的维护和查询

由于只有Homework类，所以只需要提供一个接口即可，简化了很多步骤。

4.9 学生基本信息部分的实现

4.9.1 学生基本信息部分的实体类

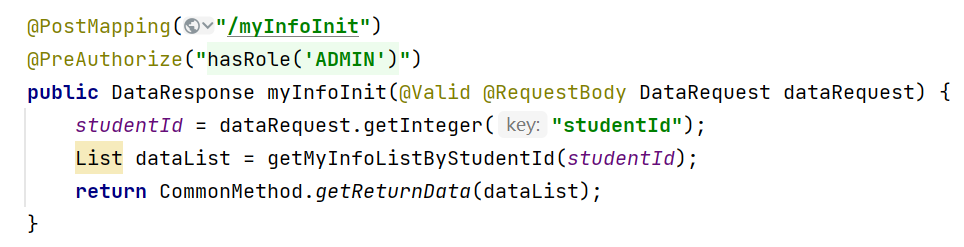
学生基本信息将student作为主键，此实体类包含诸多String类型的属性：hometown(祖籍)，contact（联系方式），political（政治面貌），ethnic（民族）等。

4.9.2 学生基本信息的控制逻辑

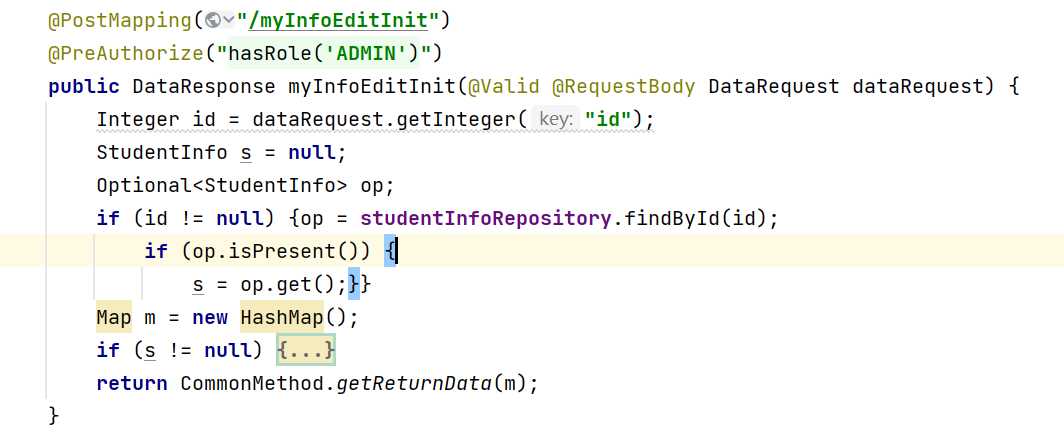
因为在学生人数较多的情况下，统计学生的基本信息，会使前端页面显得繁复冗杂，所以未实现年级下学生基本信息的统计。实现了统计指定班级中指定所有学生基本信息的ClassStudenInformationCourse；和学生详情页的MyInfo（我的基本信息）

由于功能与以上基本重复，以下仅作基本展示。

页面初始化：



编辑页面初始化：



CRUD实现：





4.10 个人简历部分的实现

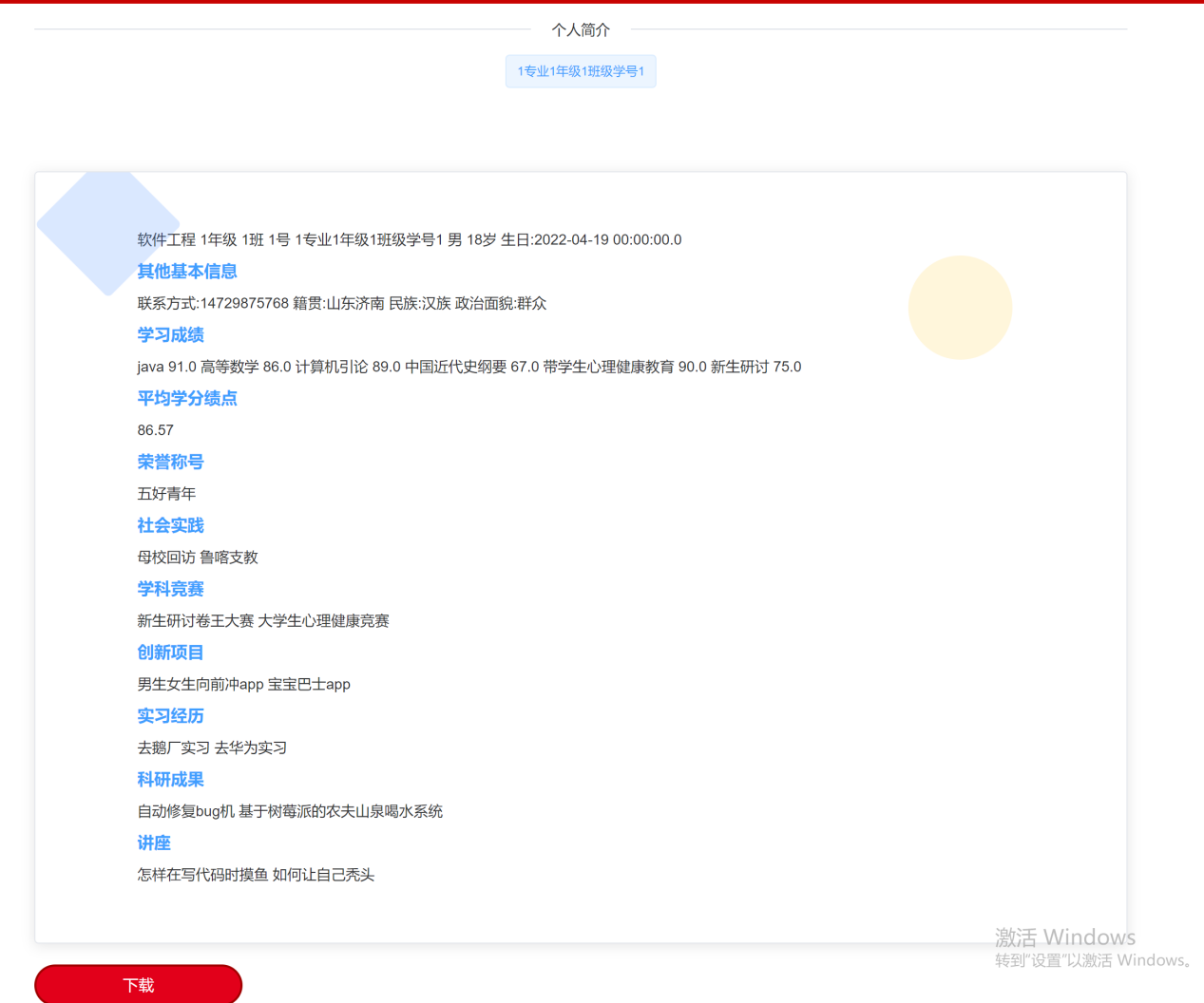
4.10.1 个人简历部分的实体类

无。只需调用其他类的方法与接口即可。

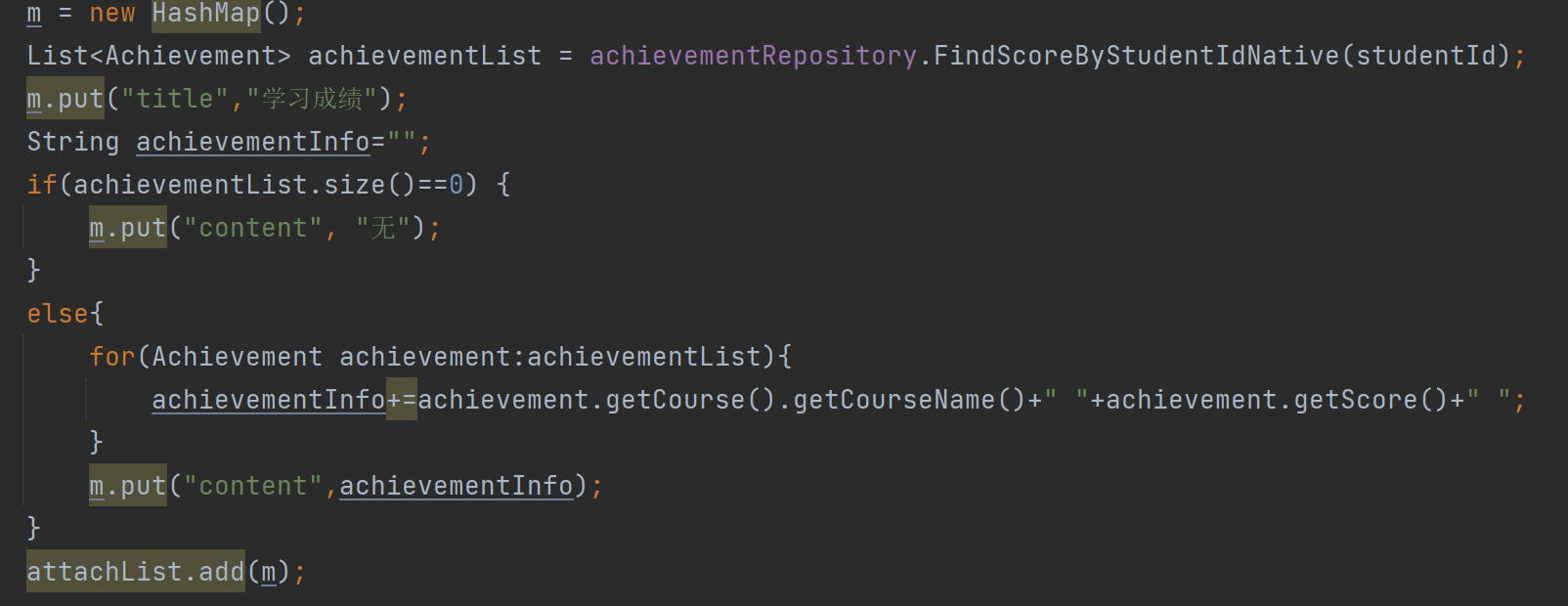
4.10.2 个人简历的控制逻辑

这一模块是做在service包下的IntroduceService中，在学生界面的controller里提供了一个获取个人简历的方法。

在个人简历中，提供了学生基本信息、学生成绩信息、学生平均学分绩点、学生荣誉信息、学生社会实践、学科竞赛、创新项目、实习经历、科研成果、参与讲座等内容。以及一个下载按钮。如下图。



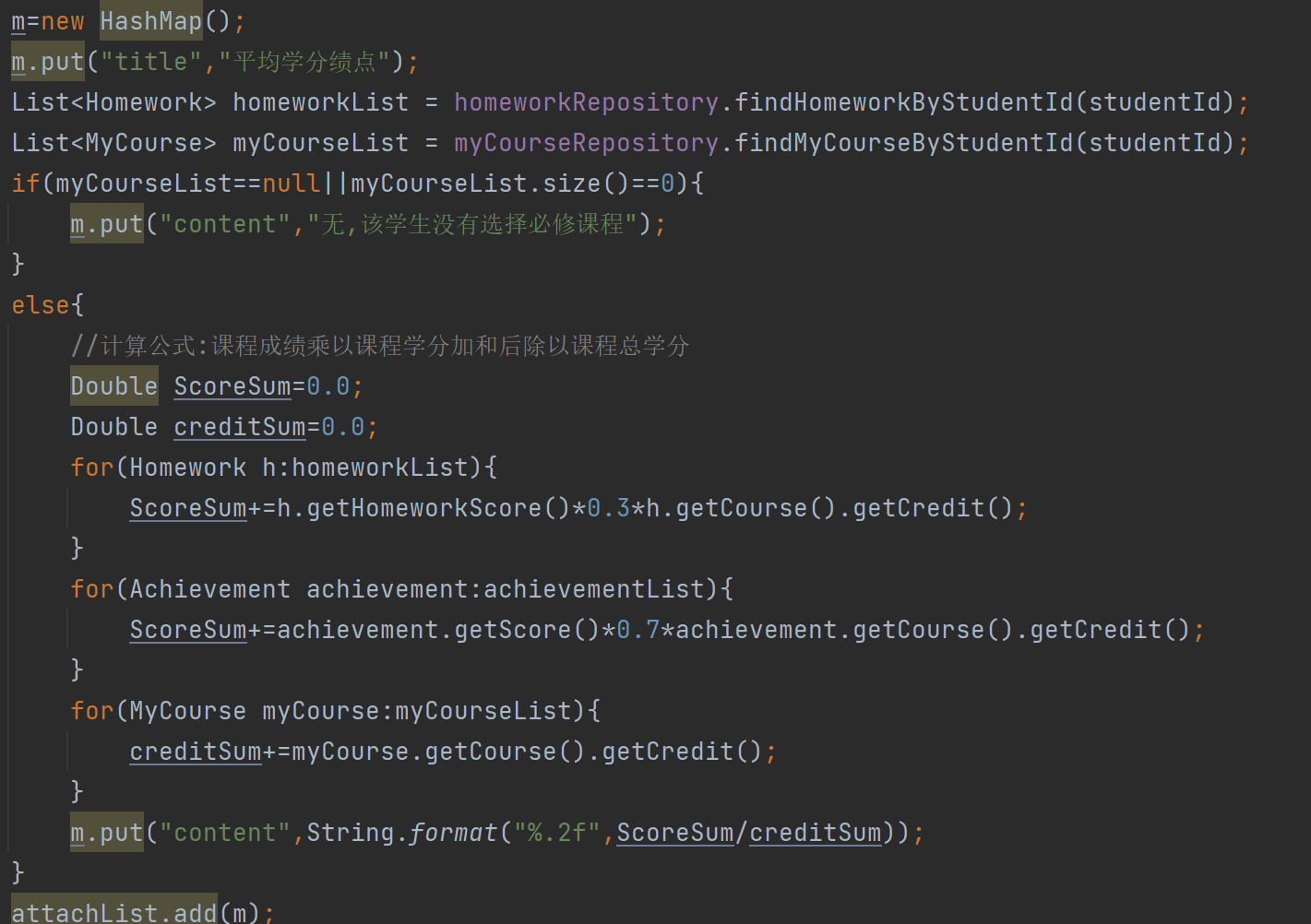
而对于除平均学分绩点外的内容，都是采用遍历查询结果，进行字符串拼接得到的，这里给出一个样例（学习成绩部分）。



下面展示平均学分绩点部分的实现。

首先是计算公式，开发人员采取了山东大学的平均学分绩点的计算公式，即课程成绩乘以课程学分加和再除以总学分。而课程成绩统一按照作业成绩占3成，考试成绩占7成处理。

随后，遍历查询结果，加和计算总分（先加和作业成绩，再加和考试成绩）和学分。随后相除，利用format方法保留两位小数。



4.10.3 个人简历部分的数据库表维护和查询

无。直接调用其他类的接口即可。

1. 总结与体会

5.1 各组员总结与体会

202100300063 李彦浩：

课设结束，细算了一下，学到不少东西。21级的java课设听说和往年不同，采用了springboot框架。但对于springboot本身的学习内容并不算多，只是在准备工作前期查阅了一些资料，结合实际代码与课堂中的讲解大致了解了框架的运作逻辑，大部分的时间和精力还是花在了“使用轮子”，而非“造轮子”上。不过即便有了轮子，作为初学者还是花费了不少时间学习如何使用轮子。从实体类开始，一点一点阅读样例代码，揣测、实验某个注解或是某段代码的作用；到DAO层，遇到了从来没接触过的SQL和JPQL，在网上看了许多不同形式的SQL语句的编写的教程资源，再仿照着样例代码中的Query，写系统需要的自定义查询语句；在Controller中涉及了一些框架传输数据的规范，反复看了很多遍DataRequest和DataResponse类，绘制了一些思维导图以便更清楚地理解这两个类以及相关的方法的运行逻辑。动手写后却又发现总不是起初想象的那么回事儿，经常要花上好几个小时的时间调试bug，上一周的代码也许要调到下周才能真正跑起来。从学习的角度来说，这次课设的内容确实是有不少困难的。

而从开发人员的角度来说，这次的开发之路也多有险阻。在开发之初，小组并没有良好的设计、规划系统的习惯。对于一个新做的模块，简单讨论后便进行分工，随后凭印象和经验予以实现。到了第十周，做出来一个很丑陋的系统，各模块在前端几乎没有联系，实体类和数据库表功能紊乱，且偶有冲突。这些都是没有事先做好规划导致的。临结束了，我们推倒重来了一次，详细地设计布局，改动了不少实体类属性，新增了主体部分，将各数据库表有机地关联在一起，并使得前端界面间的跳转更有序。

最大的体会并不来源于我学到的东西，而是来自那些我没学到的，或者说，在开发过程中“不求甚解”的地方。如果现在有人问我一些开发方面的问题，无论是宏观如“springboot是什么？”、“什么是多线程？”这样的大问题，还是细微至“如何编写代码连接到数据库？”、某一注解的具体作用是什么这样的小问题，我都没法确切地给出答案，只能根据印象泛泛而谈。但我认为这就是开发这个课设项目的过程中一个很重要的收获，即了解今后在软件工程这条道路上应该学习什么，包括学习的内容、学习的顺序等。现在不知道什么是springboot，什么是多线程，什么是IO框架，不懂什么是数据库，但我知道了自己不了解这些今后势必会用到的东西，也切身地体会过这些内容在项目开发中的应用。在今后的学习中，目标也会更加清晰。

之前听说过一些可能较本校为次的院校中的不少学生大一结束就可以自己开发一些小项目，当时还有些惶恐，现在想来也不是那么回事。东一榔头西一棒槌地学习方式也许可以在较短的时间内拼凑出一个还看得过去的成果，来粉饰自身基础的薄弱，但长此以往很可能只会浮于表面，到头来还得重新回头补课。我们作为科班出身的同学，将用至少四年时间持续地学习软件工程方面的内容，如果最后形成的知识体系只是外面培训班1、2个月速成的水平，那也不失为另一种形式的虚度光阴。课程设计是软工路上的一块敲门砖，将许多今后要系统地、有规划地学习的内容一一展现在我们面前。现在有很多不会的不打紧，好的程序员会自行设计一套完整且合理的学习路线并有计划地完成它，这是我们今后要身体力行的事。

202100300281 贾凤鸣:

在代码实现方面，我了解到了springboot JPA框架，认识了实体类与数据库的关系，对jpql和sql语言有了简要的认知并会简单运用。学习到了如何使用数据库软件sqlite和navicat，学习了yaml语言，对设计一个项目的流程有了简要清晰的认知。我也遇到了各种各样的bug，如前端点击超链但是不跳转页面，后端也不报错，debug最终发现是前端对应的yaml文件与后端controller里map里的命名不一致。Debug的过程中我逐渐学会到了通过报错定位自己错误原因，面对后端不报错的情况就考虑逻辑问题。我深刻地体会到先通读已有代码，设计好类以及规划好每周进度再去实现的重要性。

202000111131 陈书瑄:

此次的Java课设是学习编程后参与进行的第一个项目，开始接触的时候时常感到迷茫，没有方向，不知道从哪里下手，感觉和Java课程第一学期的学习有很大的不同，起先和队友们一起开会讨论要学习的内容，在学习的过程中接触了SpringBoot、JPA、SQL以及Yaml等内容的学习，对软件制作以及前后端的知识有了更加深刻的认识与理解，通过课程的不断学习，渐渐知道了此次课设的切入点，和队友们进行讨论后才正式开始了编写。

在前期models和repository的编写过程中遇到的困难只是新东西的学习，上手之后，完成的很快，在之后写controller和yaml的时候遇到了较多的问题，常常会有后端运行没有出错，但是在前端进行操作后就遇到问题的情况，或者按钮点击无效，或者想要出现的信息在前端显示为空，在这一段时间常常会陷入长时间查改Bug的情况，最后会发现问题出现的原因除了一般的逻辑错误以外，就是粗心造成的，编写代码的时候一时逞快而造成这样的后果，让我深刻的意识到细心对于编程的重要性，还有在使用新的语言或软件之前要对它有一定的认知，比如在使用数据库时要注意的操作，避免对以后使用造成影响。

在此次的课设在让我意识到了编程语言功能强大的同时，也让我认识到了它的严谨性，以及对于编程人员技术和态度的高要求，通过这次课设我学习到了很多。

202100300059 程子函：

作为第一次开发一个项目的新手，深刻体会到自己现阶段能力的不足，导致刚开始在既定框架下，不理解整个系统的运行原理，更无法自主实现想象中的功能。在开发前期，大部分时候都是照着老师的模板，进行删改以实现其他功能的开发，对于代码的来龙去脉以及具体运行过程不能完全掌握，多是照猫画虎照葫芦画瓢，实际对其中的技术细节和知识要点只是一知半解。但在多次阅读模板样例，查找资料，进行实践后，在开发的中期，也就是实现了增删改查的基本功能后逐渐理解了系统运行原理。经过不懈的努力，能在开发后期，在原有的基础上举一反三，增加功能，在改善系统的美观性实用性的同时，完善系统后端的判断逻辑，使整个系统简明且稳定。

在开发过程中遇到各式各样的bug，但大部分可归结于两点：

1. 对开发框架的理解不充分
2. 对开发过程中的格式与细节多有忽视

这也导致了类似的bug在相似的模块中反复出现，导致写代码的时间远远赶不上debug的时间，在反复检查之后，发现是细节错误而非逻辑错误时，不得不逐渐对代码的精细程度以及写代码时的准确率有所改进。

经过本次java课程设计，我也对类似的系统有了简单的认识和了解，也发现了在这些系统实践中，框架以及数据库的重要性。本次采用yaml格式的前端并已经写好，所以在做完整个系统后也感受到了自己在这几方面还有很多欠缺，缺乏系统性的学习，也为自己接下来的进修方向有了指引。

最后，程序开发作为一个团队工程，让我对协同工作有了切身体会，许多问题并不是简单的任务分配可以解决的，组内的讨论与互助可大幅提升整个项目的开发进度。我们在本次团队开发中，交流积极充分，协作热情高涨，使整个推进的过程相对高效顺利。但在开发版本的迭代以及最终成品的整合中，显得有些混乱，虽然在组员们的努力下顺利解决，但仍然花费较长的时间。在接下来的团队任务中选择团队协作开发平台作为支撑或许可以更加高效。

202100300244 曹云培：

经过我们组五个人近两个月来的不懈努力，算是基本完成了一个功能较为齐全的学生管理系统。这次的课程设计项目的经历，是我学习程序语言以来第一次进行的实际开发。在项目初期，我发现我已掌握的程序语言知识实在太过浅薄，根本不能支撑我完成整个项目的设计，所以我在进行项目工作的同时还不断学习新的知识，比如yaml页面控制的格式，sql语句的书写、数据库的相关操作等等。即便学习了许多新的知识，我在实现项目功能的过程中，也遇到了许多困难，比如超链接跳转不成功，数据库查询结果不唯一返回多个值，主键自增设置出现问题等等，有的问题经过自己上网查找资料就可以解决，有的问题比较困难，需要集合全组的力量一起商量讨论，甚至是向助教老师请求帮助才得以解决。

这次的项目经历让我学习到了很多东西，但我个人认为更为重要的是简单认识了Java项目开发的整体流程和开发框架，我认为这对于我以后的程序设计学习大有帮助，对于我将来的职业发展也颇为重要。

在我个人所完成的各项任务中，也有很多设计的不太好的地方，比如存在一些冗杂的类，如MyInnovation和Innovation这两个类的功能就比较重合；还有一些页面控制器中的方法的设计也过于繁琐，比如为了实现每个属性的查询在clazzStudentActivity和clazzStudentInnovaiton中的每个属性都单独设置一个页面来实现查询，我个人认为这有些麻烦，但目前也没找到更好的解决方案，期待随着日后的深入学习能够轻松的解决这些问题。

在接下来的学习中，我会着重加强自己的学科基础，更加深入地学习Java语言以达到熟练掌握的程度。同时，这次的课程设计项目经历也让我认识到只掌握编程语言是远远不够的，我意识到了数据结构、算法、数据库等相关知识对于实际项目开发是至关重要的，所以我也会在课余时间提前学习数据结构与算法、数据库等课程，以加强自己的开发能力和算法能力。