**项目文档**

**项目成员**

17302010024 俞继涛 [17302010024@fudan.edu.cn](mailto:17302010024@fudan.edu.cn)

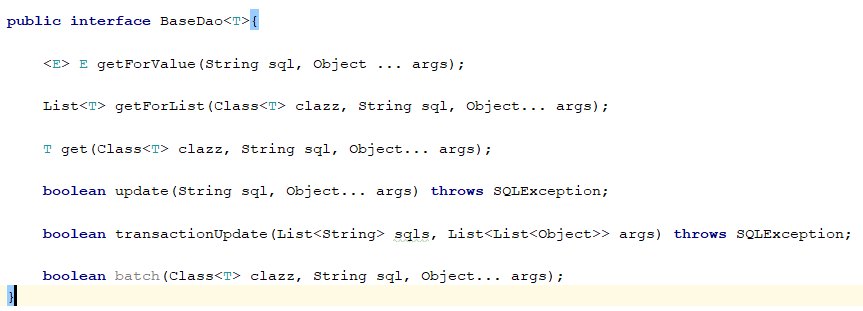
17302010026 邓朋 17302010026@fudan.edu.cn

**项目结构**

本次项目的实现使用的技术是基础Java Web技术，包含了Servlet和Jsp的基本使用。项目结构主要采用MVC的结构。Servlet作为控制器控制系统基本运行。Dao作为数据访问层的封装，主要实现对数据的增删改查。Jsp最为页面显示的部分，负责展示前端页面。由于课程主要考察的是数据库的设计和使用，我们将项目结构的重点放在Dao层。

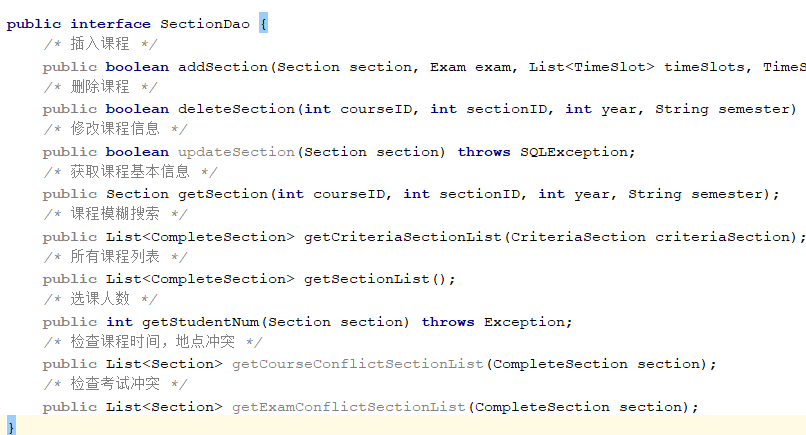
1. **Dao层的设计**

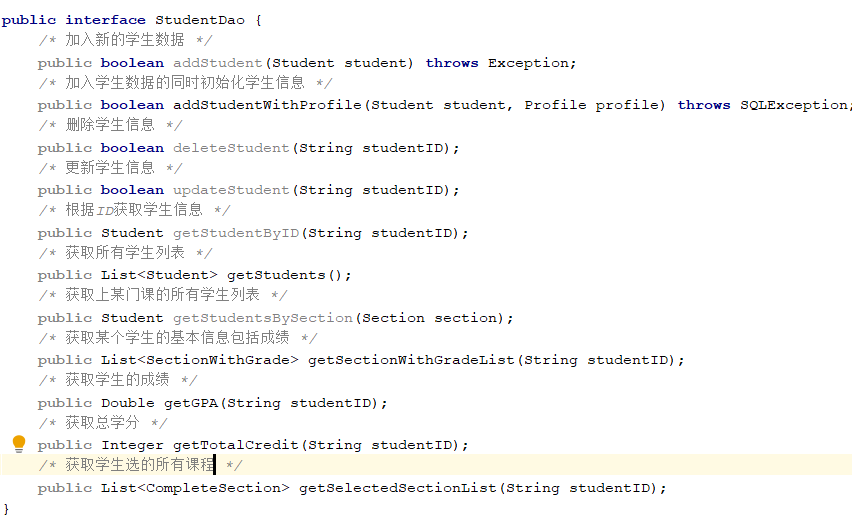
Dao（Data Access Object）层的作用是对数据处理的封装。基本功能包括数据库连接的建立，数据库查询操作的封装，读取结果的封装与对象化。为了将不同的表格操作区分开，我们将Dao分成不同的模块。下面逐一介绍各个Dao模块的接口以及对应的功能。

1. **Base Dao**

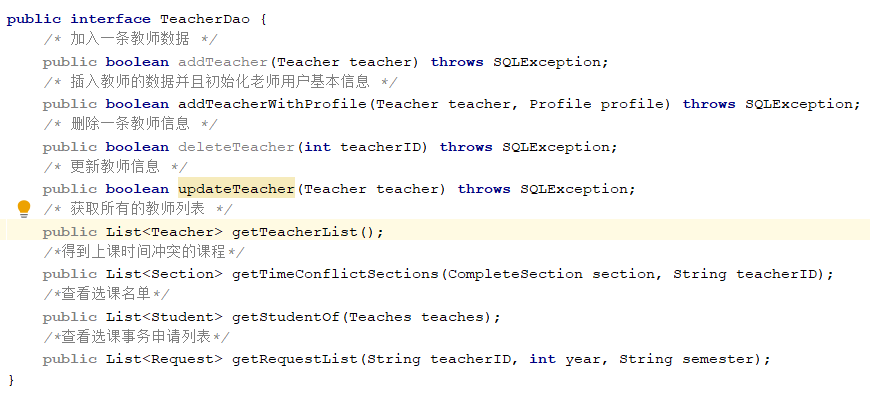
Base Dao负责连接数据库以及数据查询结果的封装。数据库的连接采用JDBC库连接数据库。这里我们使用了数据库连接池的连接方式提高数据库连接的效率。

1. **Section Dao**

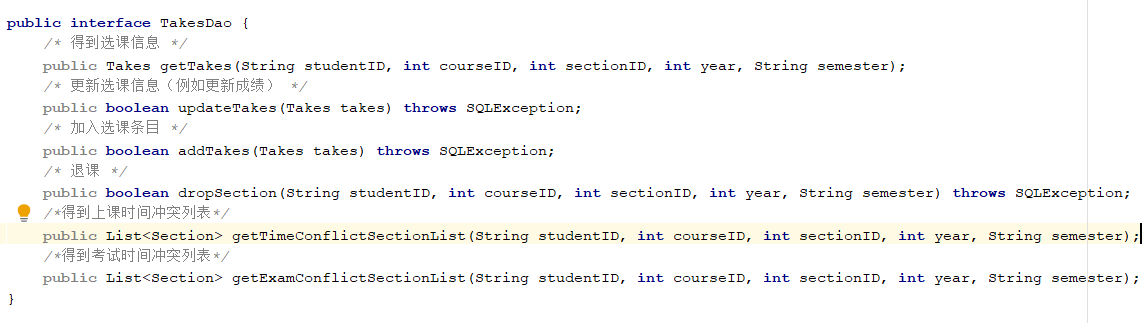
全部接口如上图所示。每一个函数都封装了对应功能的SQL语句。值得一提的是，在加入一门课的时候，需要插入多个表格的数据（例如上课时间，考试信息）。为了保证操作的原子性，发生错误时需要将之前的所有操作回滚（rollback），因此需要采用事务的处理解决原子性的问题。

1. **Student Dao**

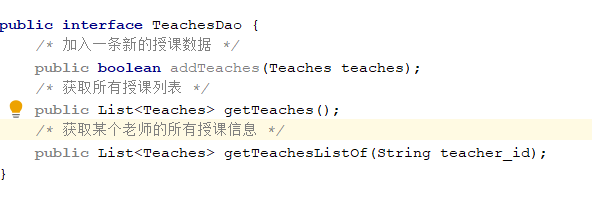
Student Dao封装了所有与学生有关的数据库操作。有一些操作在设计时考虑到了但是实际中并没有使用。

1. **Teacher Dao**

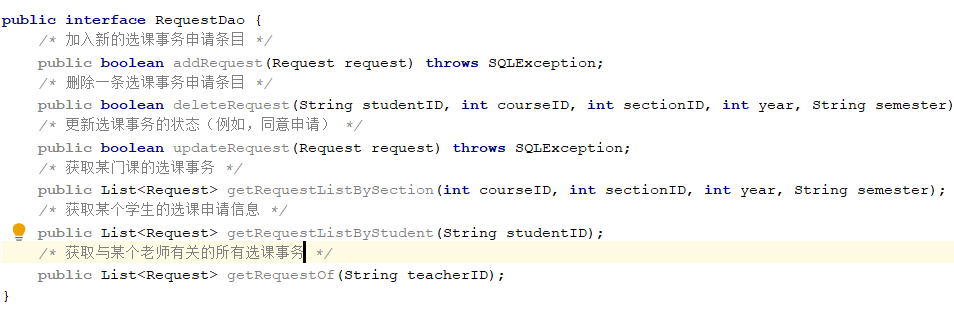
Teacher Dao封装了所有与教师相关的数据库操作。

1. **Takes Dao**

Takes Dao处理与选课相关的数据库操作。获取时间冲突的列表和获取地点冲突的列表用于判断选课是否有时间地点冲突。

1. **Teaches Dao**

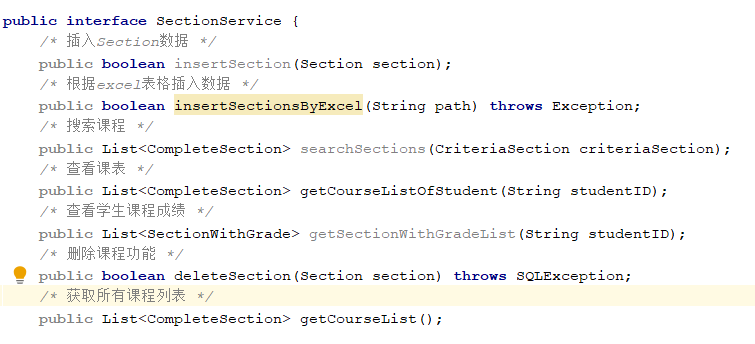
Teaches Dao处理所有与授课信息相关的数据库操作。

1. **Request Dao**

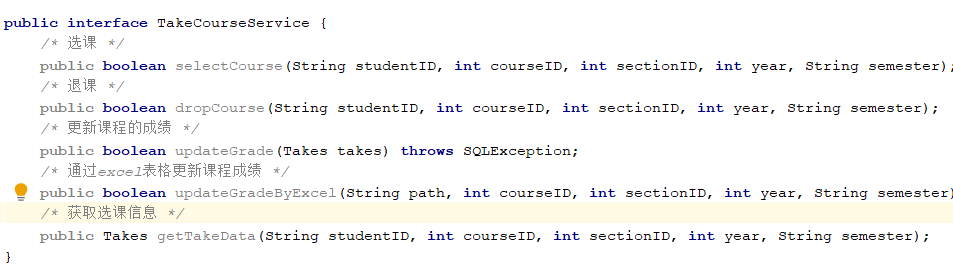
Request Dao处理与选课事务相关的数据库操作。

1. **Service的结构**

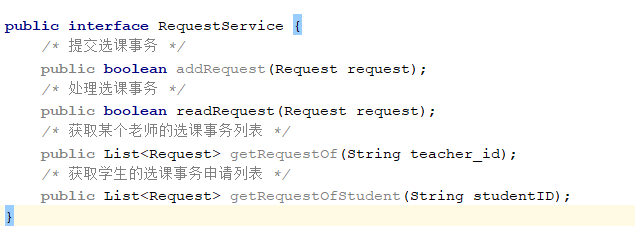
Service层处理基本业务。例如，选课功能需要在选课前进行选课冲突的判断，选课人数的判断。Service的功能主要是调用Dao层进行业务功能的实现。由于不是重点考察对象，我们只简单介绍一些重要的Service。

1. **Section Service**

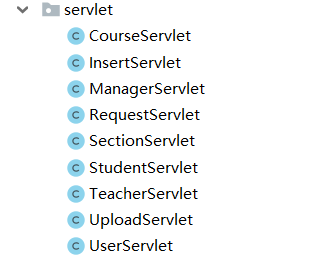
Section Service主要实现与课程相关的服务。值得一提的是，插入excel表格时，由于可能有多条数据有误，所以具体处理方式有所差异。我们的处理方案是先捕捉所有出错的行，记录出错行的数据。最后再统一抛出错误提示上层哪些数据有误。对于那些正确的数据，我们会正确地将数据放在数据库中，而不是整个表格的数据全部插入失败。

1. **Takes Service**

除了基本的选课退课，还包括更新学生的成绩信息等操作。

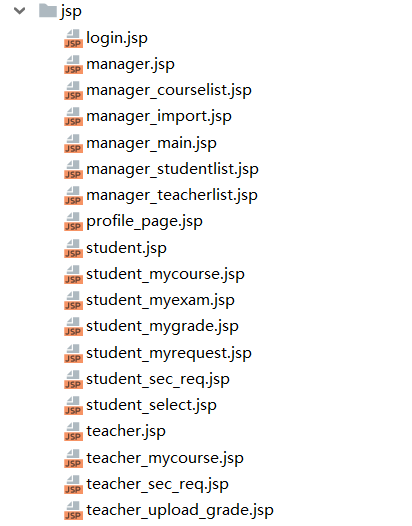
1. **Request Service**

Request Service处理与选课事务有关的事务。

1. **Servlet的结构**

Servlet是项目的控制器。主要处理来自前端的AJAX请求。这里我们将AJAX请求进行了分类，不同的Servlet处理不同的AJAX请求。InsertServlet处理与插入新数据有关的请求。RequestServlet处理与选课事务相关的请求。SectionServlet处理与课程相关的AJAX请求。UploadServlet处理上传excel表格的请求。UserServlet处理登录的请求。

1. **JSP的结构**

**** 对于每一种类型的用户，我们都有对应业务的jsp页面。Jsp页面负责数据整合和最后的内容拼接。需要注意的是，为了防止某些用户跨权限访问其他业务，我们用Filter对请求进行过滤。不同类型的用户只能访问特定的页面，从而实现权限控制。

**数据库设计**

**数据库ER图**

**数据库schema**

**设计思路**

本项目的数据库设计参考了书籍《Database system concepts》中的思路，并做了相关改进。

本次项目的关键实体大致有四个，分别为：学生、老师、管理员、课程。数据库设计的中心思想便是方便对几类实体信息以及她/他们之间关系的表示和储存。

首先实现的是对于这几个实体的信息储存。学生、老师和管理员均可以抽象称为系统的用户，只是具有不同的权限，因此我们选择将她/他们的用户名、密码、权限等信息储存在一个名为profile的数据库表当中，学生和老师的用户名即为相应的学号和密码。这样做的优点为用户登录系统时只需要查询一个数据库表即可，同时也方便更改用户的权限。对于学生、老师、管理员，为了方便处理她/他们和课程之间的关系，我们选择另外建立了独立的数据库表，并分别对profile做外键引用。

由于一门课程包含的信息较多，同时一门名称相同的课在相同学期内可能开设多门，因此我们参照书籍建立了course和section两个表，学校的教室信息需要提前导入名为classroom的数据表中，section表对其做了外键引用。为了避免section表中储存信息过多，我们另外建立了exam表和time\_slot表来储存考试信息和上课时间信息，section表对它们做外键引用。

对于以上几类实体之间的关系分析。学生和课程之间存在选课、退课、申请关系，同时选择的课程还包括课程成绩，课程的申请信息中包括申请理由和处理结果等。老师和课程之间存在教课的关系。因此我们建立了takes表、sec\_request表和teaches表，作为关系集储存相关信息。关于课程成绩，我们额外建立了level\_to\_grade表来储存等级和绩点之间的对应关系（比如我旦A对应4.0，而西南某高校A可能对应5.0）。

在数据库的设计中，我们充分考虑了扩展性，有些扩展点的优势在本次项目中可能无法体现，但是作为数据库的设计者，我们充分考虑了一个选课系统在实际应用中可能遇到的问题，并预留了扩展空间。

**需求实现**

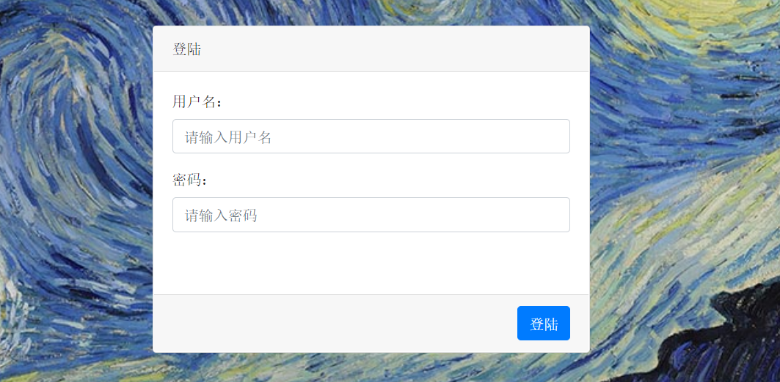
**教师、学生信息管理**

系统管理员在管理界面可以管理教师、学生信息。

**开设课程**

系统管理员在管理界面可以导入课程信息、删除课程。导入课程信息时会进行冲突数据检查，并会进行提示，冲突数据不会导入数据库。

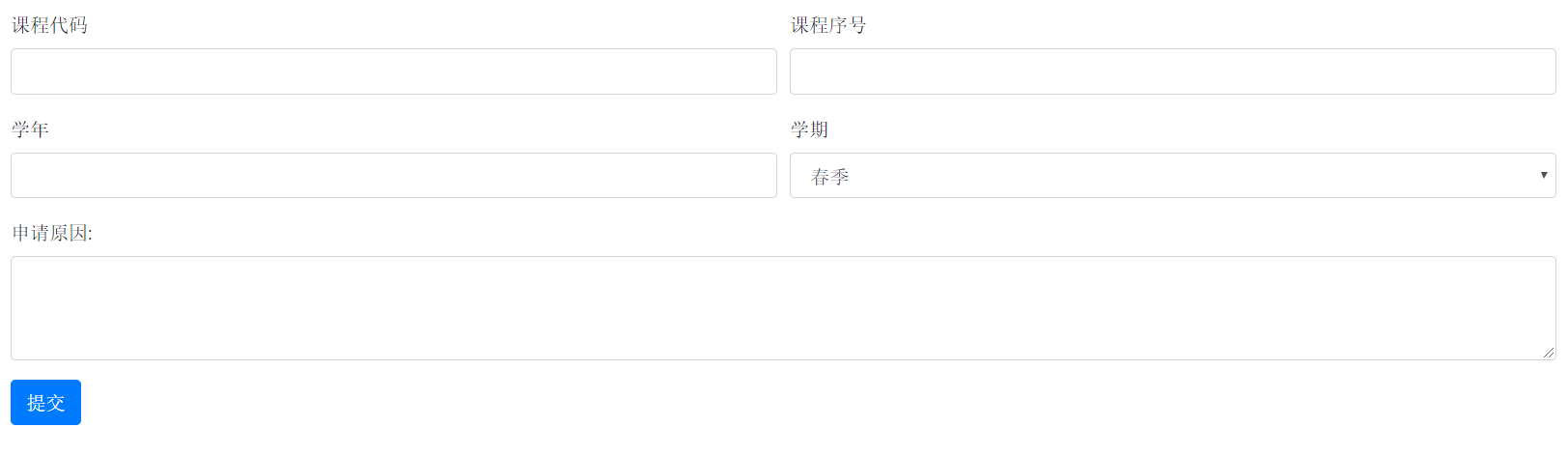
**登录/登出系统**

****学生、教师、管理员均可以使用自己的账号和密码登陆系统，学生和教师的初始密码由系统设置。成功登录系统后，具有相应的权限，学生可以选课、查看课表、查看成绩、查看考试信息、选课申请、查看申请结果；教师可以查看开始课程、查看课程花名册、管理选课事务申请。

**学生选/退课**

学生登录系统后可以搜索课程、查看课程详情并进行选课。选课时系统会检查课程冲突，主要包括上课时间冲突、考试时间冲突，如若存在冲突，则无法选课。选课成功或失败均会给出相关提示。学生可以在自己的课表里退掉已选课程，退掉后可以重选但是无法申请。

**选课事务申请**

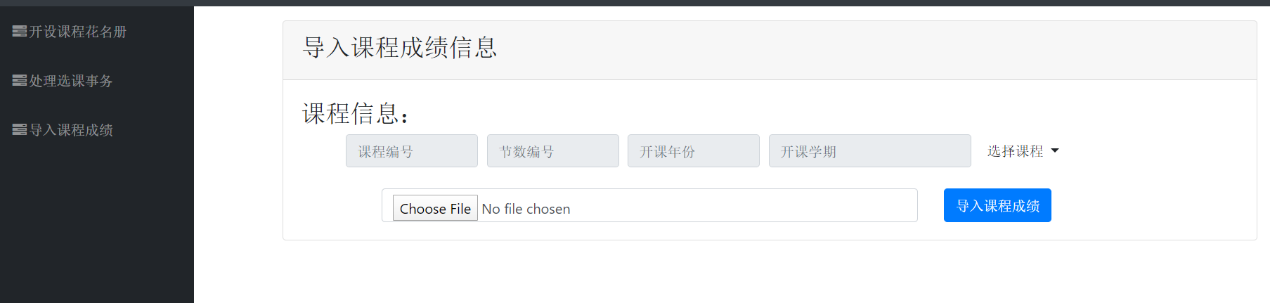
 学生可以申请当前选课人数已达课程容量的课程，但若当前选课人数已超过教室容量，则无法申请，同时也无法申请已退课程、已选课程、和已选课程在上课时间或考试时间上存在冲突的课程。申请提交后，会由课程的开设教师处理。

**考试**

 考试信息在导入课程信息时导入，学生可以查看自己的考试信息。

**登分**

教师登录后可以导入自己所开设课程的成绩信息。



**项目技术栈**

数据库连接池

单元测试

……