后端实现：

一、登录(LoginServlet)

对于所有除了index.jsp外的jsp，以及各种教师、学生、root功能的请求在登录前会被Filter过滤，重定向到index.jsp。

登录时，根据传入的userType确定用户是教师、学生或root，去对应的teacher或student表下查询，确定密码是否正确。如果密码正确，则在HTTPSession中设置对应用户名。

二、教师(TeacherServlet)

1.查看课程

教师通过getSectionTaught.tch的url来查看所有由他教授的课程。getSectionTaught函数通过HTTPSession获取当前教师的id，调用SectionUtil中的sectionTaught函数获取开课列表。教师能看到的信息包括课程id，课程段id，学期，年份，最大人数（记录在section表中），课程名称，学分，课时（记录在course表中，在course表中查询对应course\_id的记录），已选人数（在teaches表中应用count函数），上课时间（记录在time\_slot中，通过section,sec\_time\_slot,time\_slot自然连接后查询相应section的上课时间段），上课教室（通过section,sec\_class,classroom自然连接后查询），考试类型（记录在exam中，查询对应开课的考试类型，所有course\_id一样的开课考试类型相同），如果考试类型是考试，则还将得到考试时间和考试教室（通过exam,exam\_time,exam\_time\_slot自然连接和exam,exam\_class,classroom自然连接后查询）。

2.查看花名册

教师可以通过getStudentTake.tch查看由自己教授的开课的选课学生的信息，包括学生id，姓名和专业。每次查看一门开课的选课名单，开课信息从request中获取。

3.管理选课事务申请

教师可以通过getApplicationBySection.tch查看由自己教授的开课的选课申请。

对每个状态为未处理的选课申请，教师可以选择同意或拒绝，所以调用updateApplication函数时，会先查询选课申请是否存在，状态是否为未处理。接下来，如果拒绝，将application表中对应记录的状态改为1（即拒绝）。如果同意，首先会查询已选人数是否达到了教室容量的上限，如果已达到，则处理失败。接下来获取申请的学生的id，更新application表中记录的状态，向takes中插入记录。如果在同意申请后已选人数达到教室容量上限，查询对相同开课的所有申请，递归调用updateApplication，拒绝所有未处理申请。

4.登分

教师可以通过updateGrade.tch（手动登分）或updateGradeList.tch（自动登分）录入自己开课的成绩。

手动登分时一次录入一个成绩。提供学生id，开课信息，以及分数（***分数是否合法在前端校验***）。updateGrade函数检查course是否已被删除，已被删除的课程无法登分。之后检查学生是否选了这门课（或学生、开课两者中有的不存在），以及分数是否已经录入，已录入分数的无法再次录入。更新成绩后，重新计算学生的总学分、平均绩点以及非P/NP课程的总学分。

自动登分时，先指定开课，通过sectionExist函数判断开课是否存在，如果不存在直接返回操作失败。从前端获取一个csv的路径，该csv中记录了学生id以及分数。读取csv的每一行时，检查列数是否为2以及分数是否合法。其余登分前的检查和更新后的重新计算同手动登分，最后返回操作成功的记录数、操作失败的记录数以及失败的记录信息。

5.修改密码

教师可以通过changePassword.tch修改密码。用户名从HTTPSession中获得，所以只能修改自己的密码。

三、学生(StudentServlet)

1.查看开课

学生可以通过名字或id搜索开课（searchSection.stu）。

当request中courseName不为空时，根据课程名字搜索。调用searchByName查询。首先搜索出所有名字符合的开课（以及最大人数、学分、课时、开课名称），再获取每个开课的已选人数、上课老师。

当request中courseId不为空时，根据id搜索。调用searchById查询。首先搜索出所有具有该course\_id的开课，其余同通过名字搜索。

2.查看课程表

学生可以查看自己所选的所有课（sectionTaken.stu）。

学生id通过HTTPSession获得。调用sectionTaken查询。首先通过takes查询出该学生上的所有开课（以及最大人数、学分、课时、开课名称），然后查出每个开课的上课时间、上课地点、上课老师、考试地点、考试时间。

3.选课

学生可以选课（selectSection.stu）

学生id通过HTTPSession获得，开课信息从request获得，调用selectSection函数。首先检查课程是否存在，已选人数是否达到课程最大人数。然后检查学生没有选同样course\_id的开课，上课时间是否冲突（检查要选的课和已选的课有没有相同的time\_slot\_id）。然后获取考试类型，如果是考试，检查考试时间段是否冲突（要选的课和已选的课的exam\_slot\_id是否相同）。最后向takes中插入记录并删除drops中可能存在的记录，最后在drops中查询一边确定已经删去。

4.退课

学生可以退课（dropSection.stu）

学生id通过HTTPSession获得，开课信息从request获得，调用dropSection函数。***退课期限在前端检测****。*先检查该学生是否选了该开课，如果没有则操作失败，否则从takes中删除记录并在drops中添加对应记录。

5.选课事务申请

学生可以创建选课事务申请（addApplication.stu）

学生id通过HTTPSession获得，开课信息和申请理由从request获得。首先在application和stu\_app的自然连接后的表中查询学生是否已有对该开课的申请，如果有，状态是否为拒绝。如果有状态为未处理或通过的申请，则不能添加申请。然后从drops中查询学生是否退过该门开课，如果有，不能添加申请。从takes中查询学生是否选了该门开课或有相同course\_id的开课，如果有，不能添加申请。在takes表上使用count获得已选人数，查询section获得最大人数（同时可以检查section是否存在）。如果已选人数小于最大人数，则不能添加申请。向application表插入记录，状态默认为未处理，application\_id为自增的键，通过Statement.RETURN\_GENERATED\_KEYS获取生成的id，用以添加stu\_app和app\_sec的记录。

6.查看选课申请状态

学生可以查看由自己创建的选课申请（getApplicationByStudent.stu）

学生id通过HTTPSession获得。调用getApplicationsByStudent函数。在stu\_app，application和app\_sec，course表的自然连接中查找stu\_id为该学生的记录，并获取申请对应的section的course\_id,sec\_id,semester,year和course\_name信息，application的state,application\_id,excuse信息。

7.查看成绩

学生可以查看自己每门课的成绩及平均绩点（getGrades.stu）

学生id通过HTTPSession获得。在takes中查询stu\_id为该学生的记录，如果已经登入成绩（grade不为空）则加入列表。每次录入成绩都会更新student的avg\_grade和total\_credit，平均绩点和总学分从student表中获取

8.更改密码

学生可以通过changePassword.stu修改密码。用户名从HTTPSession中获得，所以只能修改自己的密码。

四、管理员(RootServlet)

1.教师信息管理

管理员可以手动输入（addTeacher.root）或自动导入（addTeacherList.root）教师信息，可以查看所有教师信息（allTeacher.root），包括tch\_id,tch\_password,tch\_name,dept\_name。

手动输入时，教师的信息从request获取，***各项的格式在前端校验***。调用addTeacher函数，如果出现重复的id，则添加失败。

自动导入时，从request获取csv路径，该csv中记录了tch\_id,tch\_name,tch\_password,dept\_name。检验每一行长度是否为4，通过正则表达式检验tch\_id是否符合格式，及其余三项是否为空。接下来同手动输入，调用addTeacher函数，如果检验不通过或addTeacher返回0（添加失败），则在message中添加这一行信息，最后返回操作成功的记录数、操作失败的记录数以及失败的记录信息（message）。

查看所有教师信息时，调用allTeacher函数，获取teacher表中所有记录。

2.学生信息管理

管理员可以手动输入（addStudent.root）或自动导入（addStudentList.root）学生信息，可以查看所有学生信息（allStudent.root），包括stu\_id,stu\_name,stu\_password,major,avg\_grade,total\_credit,grade\_credit。后三项在导入或输入是不用提供，默认为0.

手动输入、自动导入和查看信息类似教师信息管理。

3.管理课程信息

管理员可以手动添加（addCourse.root）或自动导入（addCourseList.root）课程信息，可以查看所有课程信息（allCourse.root），包括course\_id,course\_name,credit,class\_hour。可以删除课程（deleteCourse.root）

手动输入时，课程信息从request获取，***各项的格式在前端校验***。调用addCourse函数，如果出现重复的id，则添加失败。

自动导入时，从request获取csv路径，该csv中记录了course\_id,course\_name,credit,class\_hour。检验每一行长度是否为4，课程id是否符合要求，name是否为空。credit和class\_hour不为整数通过catch NumberFormatException处理。调用addCourse函数。最后返回成功操作数、失败操作数和失败记录信息。

查看信息时，调用allCourse函数，获取course表中所有记录。

删除课程时，如果是在选课时间内删除，通过在takes上使用count计算选课人数，如果大于某一值（studentNumber）则不能删除，否则删除course中的记录，并删除section中对应记录，由于外键约束设置为cascade，所以其他相关记录也会被删除。如果是登分后删除，先检查是否完成了登分（takes中该course\_id存在grade不为空的记录），只删除course中记录，保留section中及其相关的记录

4.管理开课信息

管理员可以手动添加（addSection.root）或自动导入（addSectionList.root）开课信息，可以查看所有开课信息（allSection.root）

手动输入时，课程信息从request获取，***各项的格式在前端校验***。自动导入时，csv中包含course\_id,sec\_id,semester,year,exam\_type,exam\_slot\_id,time\_slot\_id,tch\_id,exam\_building,exam\_room\_number,building,room\_number,max\_amount，检验semester和year不为空，sec\_id,exam\_type,exam\_slot\_id不是整数的错误通过catch NumberFormatException处理。tch\_id和time\_slot\_id都可能有多个值，值之间由；分割。

调用addSection函数。首先检查course\_id是否存在，其次在exam表中查找有相同course\_id的记录，两者的考试类型要相同。如果考试类型为考试，则在exam\_time表中查找course\_id相同的记录，检查考试时间是否相同，以及检查同一教室在这个时间段是否已经安排了考试。

然后检查任课老师时间上的冲突。对每一个任课老师的每一个已经用去的时间段，检查是否与要添加课程的时间段有重合（timeSlotConflict）。检查上课教室的冲突，对上课教室已经用去的每一个时间段，检查是否与要添加课程的时间段有重合。

然后检查上课教室和考试教室的容量，如果要添加开课的最大人数大于教室容量，则添加失败。

最后向section,exam,sec\_class,sec\_time\_slot,teaches添加记录，如果考试类型为考试，向exam\_time和exam\_class添加记录。

查看所有开课信息时，对section中每条记录，获取最大人数、学分、课时、，然后查出每个开课的名称、上课时间、上课地点、上课老师、（如果类型为考试）考试地点、考试时间。

5.管理成绩信息

管理员能手动导入（updateGrade.root），自动导入（updateGradeList.root）成绩信息。

手动输入时，学生id和开课信息从request获取，随后调用updateGrade函数，函数的实现在教师导入分数部分已经提及。

自动导入时，csv中记录了course\_id,sec\_id,semester,year,stu\_id,grade。检查每一行长度是否为6，以及grade是否合法。返回操作成功数，操作失败数，失败记录信息。

***6.选课事务信息管理***

***管理员能查看所有选课事务信息***

***7.选课、退课信息***

***管理员能查看takes,drops所有信息***