桂林电子科技大学2019-2020学年 第2学期

**数据库系统原理综合 实验报告**

**课号： 1922159 任课老师：唐麟**

|  |  |  |  |
| --- | --- | --- | --- |
| 项目名称 | **课程选择系统** | | |
| 院 系 | **计算机与信息安全学院** | 专业 | **软件工程** |
| 实验日期 | **2021年6月20日** | | |

团队任务分工表

|  |  |  |  |
| --- | --- | --- | --- |
| 题目 | 课程选择系统 | | |
| 负责人 | 1900301236 谢浚霖 | | |
| 合作  方式 | 使用git和github进行多人远程协作 | | |
| 序号 | 学号 | 姓名 | 个人任务描述 |
| 1 | 1900301236 | 谢浚霖 | 起草项目需求文档、项目接口文档、关系模型；  负责设计数据库；  负责学生、教师登录、用户退出的后端逻辑实现；  负责学生端、教师端后端逻辑编写；  协助组员完成教师端的前端逻辑测试；  协助组员完成用户信息、课程信息模块； |
| 2 | 1900300819 | 卢林军 | 负责数据库的表创建，生成测试所需数据；  负责设计数据库；  负责教师登录、用户退出的前端逻辑实现；  负责前端界面的编写；  负责学生端、教师端后端逻辑编写；  协助组员完成用户信息、课程信息模块；  协助组员完成教师端的后端逻辑测试； |

目录

[1、引言 4](#_Toc75104872)

[1.1、社会背景 4](#_Toc75104873)

[1.2、用户需求 4](#_Toc75104874)

[1.2.1、学生端 4](#_Toc75104875)

[1.2.2、教师端 4](#_Toc75104876)

[2、功能分析 5](#_Toc75104877)

[3、数据库关系 5](#_Toc75104878)

[3.1、数据库关系分析 5](#_Toc75104879)

[3.2、概念结构与逻辑结构设计 6](#_Toc75104880)

[3.3、 物理结构的设计与实现 7](#_Toc75104881)

[3.3.1、学生表(student) 7](#_Toc75104882)

[3.3.2、教师表(teacher) 8](#_Toc75104883)

[3.3.3、学院表(college) 9](#_Toc75104884)

[3.3.4、班级表(class) 9](#_Toc75104885)

[3.3.5、课程计划表(schedule) 10](#_Toc75104886)

[3.3.6、课程表(course) 11](#_Toc75104887)

[3.3.7、讲授关系表(teach) 12](#_Toc75104888)

[3.3.8、选课关系表(selection) 12](#_Toc75104889)

[4、详细设计 13](#_Toc75104890)

[4.1、登录界面 13](#_Toc75104891)

[4.1.1、登录模块的设计与实现 13](#_Toc75104892)

[4.2、学生端 15](#_Toc75104893)

[4.2.1、查看个人信息的模块 15](#_Toc75104894)

[4.2.2、 查看已选课程信息模块 17](#_Toc75104895)

[4.2.3、查看所有的课程计划模块 17](#_Toc75104896)

[4.2.4、课程选择模块 17](#_Toc75104897)

[4.2.5、学生退课模块 20](#_Toc75104898)

[4.3、教师端 20](#_Toc75104899)

[4.3.1、教师个人信息查询模块 20](#_Toc75104900)

[4.3.2、教师课程安排查询 21](#_Toc75104901)

[4.3.3、教师课程计划查看和添加模块 21](#_Toc75104902)

[4.3.4、教师删除课程计划 22](#_Toc75104903)

[5、测试 22](#_Toc75104904)

[5.1、登录界面测试 22](#_Toc75104905)

[6、关键技术 24](#_Toc75104906)

[6.1、前端关键技术总结 24](#_Toc75104907)

[6.2、后端关键技术总结 24](#_Toc75104908)

[7、存在的问题及改进思路 24](#_Toc75104909)

[8、总结 24](#_Toc75104910)

# 1、引言

## 1.1、社会背景

在传统的选课模式中，选课记录大多数都由人工手动完成，这样既限制了信息流通，又浪费了时间，甚至可能造成一些错误的发生，不能很好地使用户快速解决自己也业务。随着科学技术的发展，网络web技术得到不断发展，现在该项技术已经成熟，人们视线了帮助学生和教师高效，快速地完成选课排课需求的平台——课程选择系统。

先如今的课程选择系统开启了新的模式，摆脱了手工操作的过程，优化了用户的操作，通过用户友好型的人机交互，使用简洁地操作完成复杂的功能，同时由于线程同步的出现，避免了“脏数据”的出现。根据不同的用户不同的需求，可以有针对性得改变着平台提供的服务，轻松便捷地引导用户完成业务，即教师和学生根据现有课程进行合理排课和选课。但是在信息飞速发展的今天海量信息每天都在扑面而来，不断进行更新和迭代，为了是用户快速了解业务的状态，并高效地完成任务，选课系统仍然需要完善相关技术。因此，在切实提高用户业务质量的基础上改善，加强用户信息化条件的建设，利用现今流行和高速发展的web技术将选课系统逐渐完善是当前的首要任务。综上所述，课程选择系统的开发和发展是有必要的。

## 1.2、用户需求

1.2.1、学生端

学生能够在课程选择系统中，查看自己的个人信息，能够查看所有的课程计划安排。提供可以选择的课程列表，在点击选课后会展示出该课程在对应学期的所有课程计划。学生可以通过点击选课进行课程选择。在选择课程结束后还可以对已选课程进行退课。学生还可点击查询自己的已选课程。

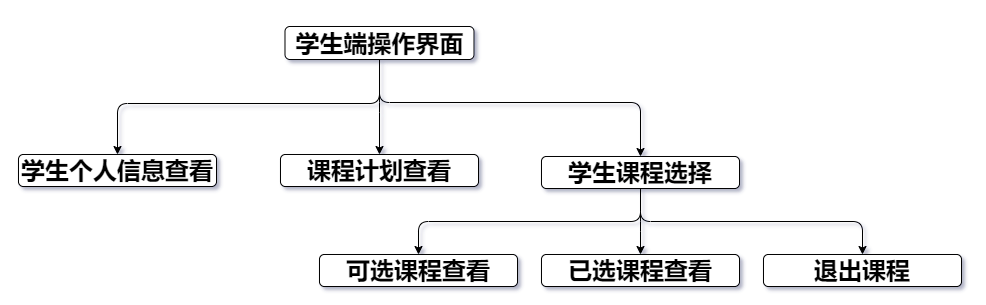
1.2.2、教师端

教师能够在课程选择系统中，查看自己的个人信息，可以查看自己的课程计划，能够添加自己的课程计划，根据自己添加的课程计划，生成自己已经选择要教的课程信息，通过获取到的教师课程计划，在对应的课程计划中会显示出学生当前课程已经选择的人数，教师可以有选择地删除自己的课程计划。

# 2、功能分析

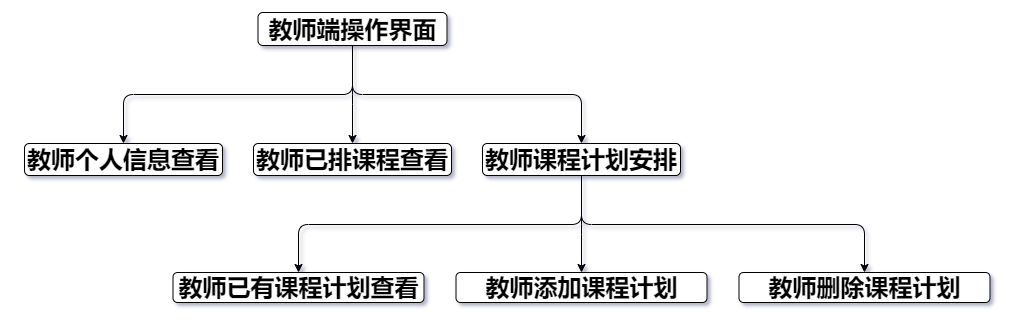
根据需求可以得知，课程选择系统应该提供两个不同的操作端，一个为学生端，为学生提供课程选择服务，另一个为教师端，为教师提供排课服务。为了鉴别用户是否为学生与教师，还需要一个登陆界面，通过单选框的方式确认用户是登陆学生端还是教师端，根据不同的选择，分别到不同的数据库表中进行验证。

对于登陆界面，应该提供登陆功能模块，通过用户对客户端的不同选择到不同的数据库中进行验证，并通过不同的登陆行为进入不同的客户端。对于学生端的设计，学生端应该具备的功能如下图2-1所示：



**图2-1 学生端界面功能图**

对于教师端的设计，教师端应该具备的功能如下图2-2所示：



**图2-2 教师端界面功能图**

# 3、数据库关系

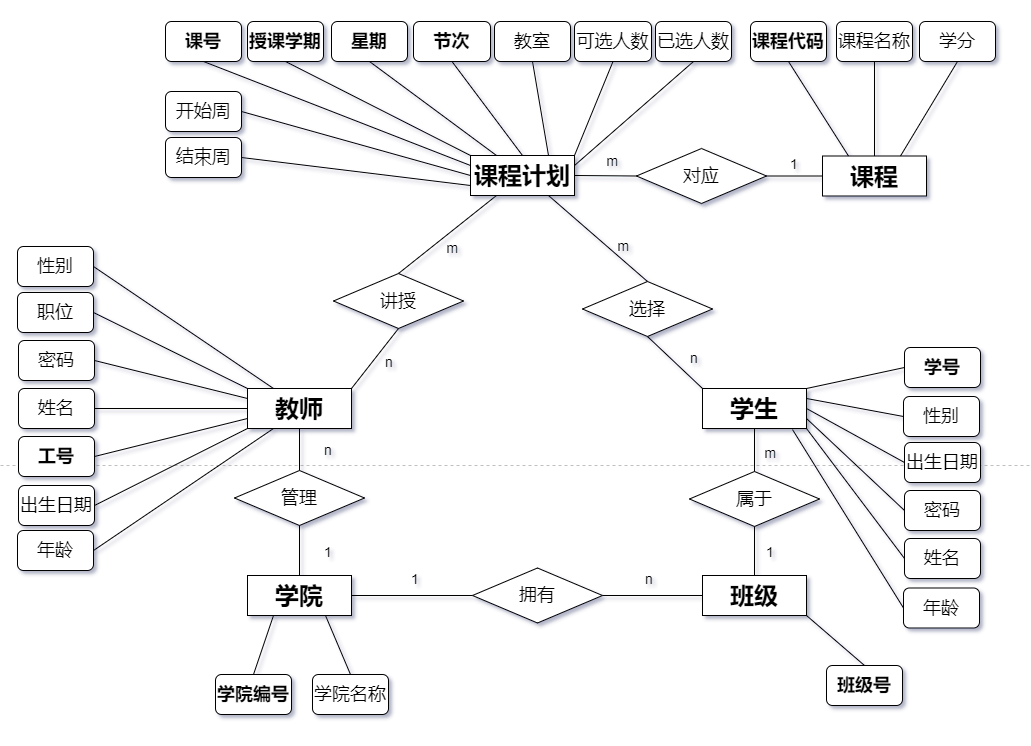
## 3.1、数据库关系分析

根据上述需求调研，对构建的数据库的关系模式抽象描述如下：

一个学院管理多个教师，管理多个班级；一位老师只能属于一个学院，一个班级也只能属于一个学院。学院属性为：学院编号，学院名称，班级属性为：班级号，教师的属性为：工号，姓名，性别，职位，年龄，出生日期以及教师端的登陆密码。一个学生属于一个班级，一个班级可以有多名学生。学生的属性为：学号，姓名，性别，年龄，出生日期以及学生端的登陆密码。一个门课程因为课号的不同可能对应多个课程计划，一个课程计划只能对应一门课程。课程计划的属性为：课号，授课学期，星期，节次，教师，可选人数，已选人数，开始周以及结束周。课程的属性为：课程代码，课程名称，学分。一位老师可以讲授多门课程，一门课程也可以由多个老师讲授。一位学生可以选择多门课程，一门课程也可以被多个学生选择。

## 3.2、概念结构与逻辑结构设计

根据关系模式的抽象描述，可以绘制出E-R图，如下图3-1所示：



**图3-1 课程选择系统E-R图**

根据上述E-R图的设计以及E-R图之间的转化关系模型的规范，可以将关系模型转化成如下表3-1所示的关系，根据BC范式的定义，该关系模式为BC范式。

**表3-1 课程选择系统的关系模式**

|  |
| --- |
| 学生 (学号，姓名，性别，年龄，出生日期，班级号，登录密码)  教师 (工号，姓名，性别，出生日期，年龄，职位，学院编号，登录密码)  学院 (学院编号，学院名称)  班级（班级号，学院编号）  课程计划 (课号，授课学期，星期，节次，教室，可选人数，已选人数，开始周，结束周，课程代码)  课程 (课程代码，课程名称，学分)  选择 (学号，课号)  讲授(课号，工号) |

## 3.3、 物理结构的设计与实现

依据上述的关系模式，使用OpenGauss数据库建立了八张关系表，他们的表明以及对应的名称如表3-2所示：

**表3-2 运用OpenGauss数据库建立的八张关系表**

|  |  |
| --- | --- |
| 表名 | 备注 |
| student | **学生表** |
| teacher | **教师表** |
| college | **学院表** |
| class | **班级表** |
| schedule | **课程计划表** |
| course | **课程表** |
| teach | **讲授关系表** |
| selection | **选课关系表** |

3.3.1、学生表(student)

| **名称** | **字段** | **类型** | **说明** |
| --- | --- | --- | --- |
| 学号 | sno | varchar(50) | 主码 |
| 姓名 | name | varchar(50) |  |
| 密码 | passwd | varchar(50) |  |
| 性别 | sex | char(1) |  |
| 生日 | birthday | smalldatetime |  |
| 年龄 | age | int |  |
| 班级号 | classnum | varchar(50) | 班级表的外码 |

在学生表用来存储学生的个人信息，学生的学号sno能唯一标识一个学生，因此将学生的学号作为学生表的主码，根据完整性约束原则，通过设置班级号classnum的外码用来获取学生对应的班级信息。如表3-3所示：

**表3-3 学生表**

3.3.2、教师表(teacher)

在教师表中工号能够唯一标识教师的身份，所以将教师的工号tno作为教师表的主码，同时根据完整性约束原则，设置学院编号collegenum作为学院表的外码，用来获取教师所处学院的信息。如表3-4所示：

**表3-4 教师表**

| **名称** | **字段** | **类型** | **说明** |
| --- | --- | --- | --- |
| 工号 | tno | varchar(50) | 主码 |
| 姓名 | name | varchar(50) |  |
| 密码 | passwd | varchar(50) |  |
| 性别 | sex | char(1) |  |
| 生日 | birthday | Smalldatetime |  |
| 年龄 | age | int |  |
| 职位 | position | varchar(50) |  |
| 学院编号 | collegenum | int | 学院表的外码 |

3.3.3、学院表(college)

学院的编号能够唯一标识学院的信息，因此设置学院编号num为主码。学院表构成如图表3-5所示：

**表3-5 学院表**

|  |
| --- |
|  |
| **名称** | **字段** | **类型** | **说明** |
| 学院编号 | num | int | 主码 |
| 学院名称 | name | varchar(50) |  |

3.3.4、班级表(class)

班级表只有班级号num一个属性，设置其为主码，并且根据参照完整性，设置班级表中的学院编号collegenum为对于学院表的外码，如表3-6所示：

**表3-6 班级表**

| **名称** | **字段** | **类型** | **说明** |
| --- | --- | --- | --- |
| ID | id | int | 自增 |
| 班级号 | num | varchar(50) | 主码 |
| 学院编号 | collegenum | int | 学院表的外码 |

3.3.5、课程计划表(schedule)

课程计划表的属性较多，经过我们的分析，课程计划表记录的是一次课的记录，比如说一节课可能在第一周星期一上午上一次，这时候在课程计划表中有一个记录，同一个课号还可能在第一周的星期四上午第二节上一次课，因此我们会发现课号显然无法作为课程的唯一标识符，因此需要引入联合主码。我们将课号cno，授课学期semester，星期day，节次index作为主码。我们看看这样能否唯一标识一条记录，如果有了授课学期，我们就可以确定具体是哪一个学期开始，但是这个学期仍然有很多课；通过课号，可以确定是哪一门课，但是他可能在不同的星期有不同的一次课；通过星期，可以确定该课具体开设在哪一个星期，但是可能在这个星期里面的四节里都有可能；最后通过节次，就可以唯一确认每一次上课的具体记录了。同时根据参照完整性将，课程代码coursecode作为课程表和课程表之间的外码。如表3-7所示：

**表3-7 课程计划表**

| **名称** | **字段** | **类型** | **说明** |
| --- | --- | --- | --- |
| 课号 | cno | varchar(50) | 主码 |
| 授课学期 | semester | varchar(50) | 主码 |
| 星期 | day | int | 主码 |
| 节次 | index | int | 主码 |
| 教室 | classroom | varchar(50) |  |
| 可选人数 | optional | int |  |
| 已选人数 | selected | int |  |
| 开始周 | startweek | int |  |
| 结束周 | endweek | int |  |
| 课程代码 | coursecode | varchar(50) | 课程表的外码 |

3.3.6、课程表(course)

在课程表中记录每一门课程的课程信息，课程代码coursecode能唯一标识一门课因此将课程代码设置为主码。课程表如表3-8所示：

**表3-8课程表**

|  |
| --- |
|  |
| **名称** | **字段** | **类型** | **说明** |
| 课程代码 | coursecode | varchar(50) | 主码 |
| 课程名称 | name | varchar(50) |  |
| 学分 | credit | float |  |

3.3.7、讲授关系表(teach)

讲授关系需要结合课程计划表和教师表的主码，联合作为一个新的关系表，主码为教师工号tno和课号cno的联合主码，但是该关系并没有其他的附加属性，因此得到如表3-9所示的讲授关系表。

**表3-9讲授关系表**

| **名称** | **字段** | **类型** | **说明** |
| --- | --- | --- | --- |
| 教师工号 | tno | varchar(50) | tno与cno的联合主码 |
| 课号 | cno | varchar(50) | tno与cno的联合主码 |

3.3.8、选课关系表(selection)

选课关系需要结合课程计划表和学生的主码，联合作为一个新的关系表，主码为学号sno和课号cno的联合主码，并且该关系也没有其他多余的附加属性，因此可以得到如表3-10所示的选课关系表。

**表3-10选课关系表**

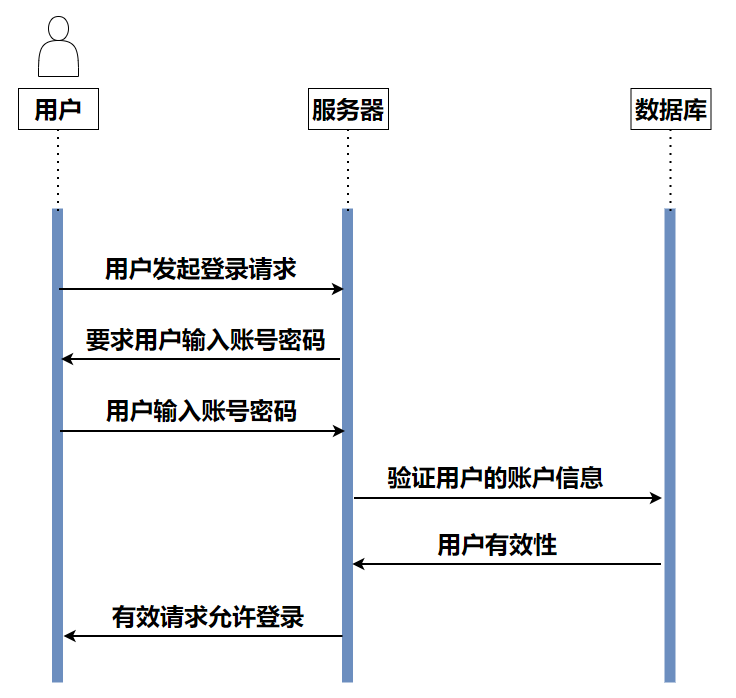
| **名称** | **字段** | **类型** | **说明** |
| --- | --- | --- | --- |
| 课号 | cno | varchar(50) | sno与cno的联合主码 |
| 学号 | sno | varchar(50) | sno与cno的联合主码 |

# 4、详细设计

## 4.1、登录界面

4.1.1、登录模块的设计与实现

登录界面是进入学生端和教师端所必须经过的一扇们。为了防止不良人士对数据进行篡改以及机器自动修改后台信息，登录模块是非常必要的。模块的时序图如图4-1所示。



**图4-1登录模块时序图**

由于用户分别存放在student表(学生表)和teacher表（教师表）中，用户的信息包含了密码字段(passwd)，因此用户在输入时会根据输入的内容调用登录接口，同时根据选择的登录端不同向不同的表中进行查询。由于考虑到前端发送请求时发送的是原密码，因此在数据库中存储的密码是经过MD5加密过的密码，使得登录的过程中能够保持较高的安全性。先获取对应账号的密码(已经经过MD5加密)，然后对输入的密码进行加密，判断两个MD5码是否相同，若相同则允许登录，否则会提示密码错误。

由于学生端和教师端的登录接口基本一致，因此只展示学生端的关键代码：

m = md5()

userid = None

db\_passwd = None

# 确认输入类型

if not isinstance(sno, str):

    sno = str(sno)

if not isinstance(passwd, str):

    passwd = str(passwd)

try:

    # 链接数据库(密码不方便展示)

    conn = psycopg2.connect(database="CourseSelectionSystem", user="\*\*\*\*\*\*\*\*",

                            password="\*\*\*\*\*\*\*\*", host="\*\*\*\*\*\*\*\*", port="\*\*\*\*\*\*\*\*")

    cursor = conn.cursor()

    # 获取输入学号的密码

    cursor.execute(f"select passwd, userid from student where sno={sno}")

    rows = cursor.fetchall()

    for row in rows:

        db\_passwd = row[0]

        userid = row[1]

    cursor.close()

    conn.close()

    # 对输入的密码进行加密

    m.update(passwd.encode("utf-8"))  # md5加密密码

    md\_passwd = m.hexdigest()

    print(db\_passwd, md\_passwd)

    # 判断两个密码是否相同

    if db\_passwd == md\_passwd:

        return {"status": "success", "data": str(userid)}

    else:

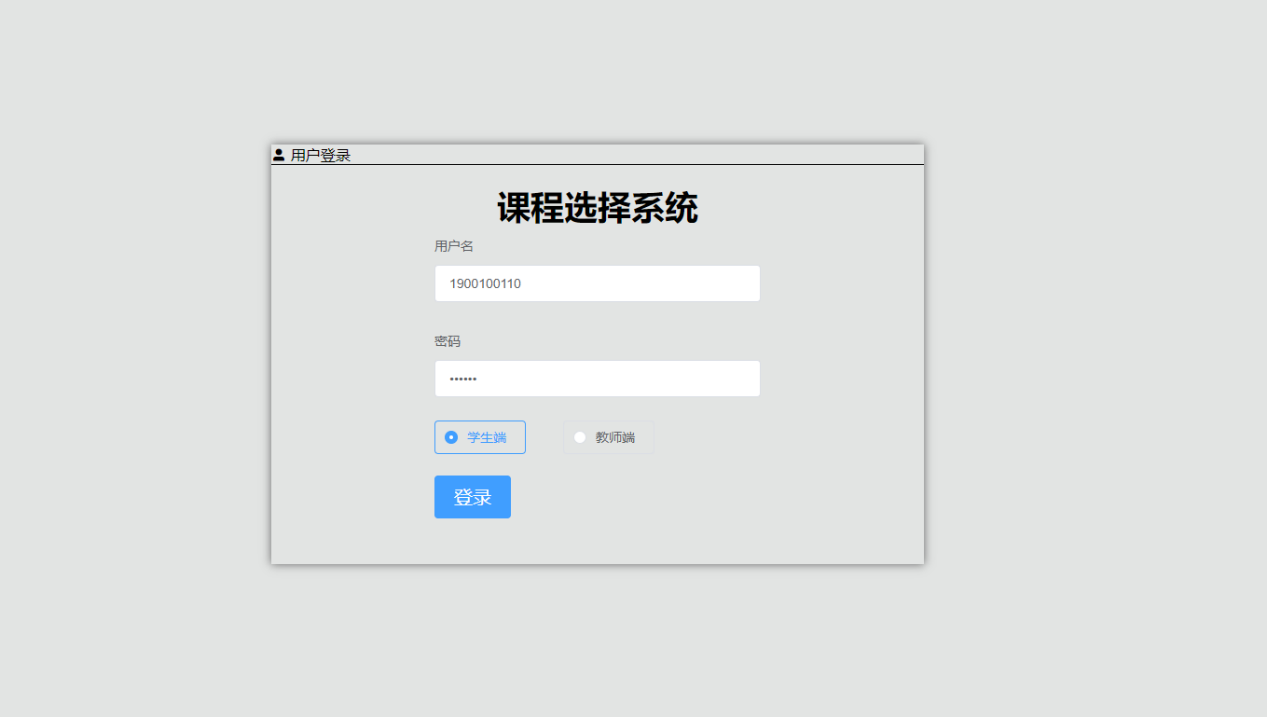
        return {"status": "failure", "data": []}

except Exception as e:

    traceback.print\_exc()

    return {"status": "error", "data": []}

录界面设计如图4-2所示：



**图4-2登录界面**

## 4.2、学生端

4.2.1、查看个人信息的模块

通过查看个人信息模块，学生可以在前端发出请求，后端根据学生的身份信息到后端中寻找该学生的信息，然后将数据返回到前端，前端通过合适的方式展示给学生，界面设计如图4-3所示：



**图4-3学生查看个人信息**

用户在登录后会产生对应的userid，并将其存储在浏览器缓存中，使用stu/getStuInfo/userid=<userid>的接口可以获得用户用户的个人信息，并将其存储在js变量中，在vue的data函数将js变量返回，使得这个存储着用户信息的变量将变成相应式变量在vue创建页面的视图上使用，对于信息的展示我们选择以表格的方式展示，涉及的vue代码如下所示：

data() {

var tableData;

// 通过axios发出请求，并通过对应的接口获取对应的数据信息

    this.$axios

      .get("stu/getStuInfo/userid=" + window.localStorage.getItem("userid"))

      .then((res) => {

        var temp = res.data.data[0];

        this.tableData = [

          {

            date: "姓名",

            name: temp.name,

          },

          {

            date: "工号",

            name: temp.sno,

          },

          {

            date: "性别",

            name: temp.sex,

          },

          {

            date: "年龄",

            name: temp.age,

          },

          {

            date: "出生日期",

            name: String(temp.birthday.split(" ")[0]),

          },

          {

            date: "班级",

            name: temp.classnum,

          },

        ];

      });

    return {

      tableData,

    };

  },

4.2.2、 查看已选课程信息模块

用户登录后点击已选课程，就会向服务器发起请求，请求获取在数据库中存储的用户选课信息，之后服务器会在数据库中寻找对应的信息，并返回数据给后端，后端将数据封装好后响应前端的请求，在前端中获得后端返回的数据，并以表格形式进行展示，查看已选课程模块的界面如图4-4所示：



**图4-4学生查看已选课程**

4.2.3、查看所有的课程计划模块

用户登录后可以查看所有的课程计划，并且提供了搜索功能，以便让用户能够更好地找到自己需要课程计划信息，规划好自己的选课过程，该模块的界面如图4-5所示：



**图4-5学生查看所有的课程计划**

4.2.4、课程选择模块

在改模块中可以说是本系统的一大难点，首先是可选课程展示，这个过程中的接口会先过滤掉已经被选择的的课程，然后提供可以选择课程。可以选择课程可能比较多，不过我们的系统也提供了搜索功能，可以迅速找到用户所需要的课程内容。我们的选课一级界面如图4-6所示：



**图4-6学生选课界面(1)**

当我们点击选课按钮后会进入二级界面， 在二级界面中会向后端传入选中课程的课程代码，后端需要向前端返回对应课程代码的所有课程计划，并以弹窗表格的形式展示在用户界面中，二级界面如图4-7所示：



**图4-7学生选课界面(2)**

最后该部分为本模块实现难度较大的地方，为学生添加选课。该模块表面上就是将对应的课程序号，添加到选择表中，但是仔细想想，学生如果有课程冲突，这是不是就不能单纯地对数据进行添加，而是需要进行一个排课冲突查询呢？

答案是显然的，因此我们的解决方案是，先判断在学生已经选择的课表中是否存在已经和正在选择课程的星期节次相同的星期节次，如果不存在，我们可以明白，不同的星期或不同的节次，不论开课时间教师是否相同，都不会有冲突，因此这种情况就可以直接插入。

如果存在，接着需要判断相同的课程在开课时间是否可以兼容，这里我们采用了交叉对比的方法，假设正在选择的课为a，冲突的课为b，如果a的开始时间比b的结束时间晚或者如果a的结束时间比b的开始时间早，那么这两个课程就是可以兼容的，因此可以直接插入，但是，如果不兼容怎么办呢？那就说明他是真的冲突了，所以就要给前端返回课程冲突。添加成功后还需要对课程计划表中的内容进行+1操作。

关键代码如下：

## 获取学生的已选课程

cursor.execute(

    f"select schedule.cno, semester, day, index, classroom, optional, selected, startweek, endweek, teach.tno, teacher.name, course.name  from schedule join course on course.coursecode=schedule.coursecode join teach on teach.cno=schedule.cno join teacher on teach.tno=teacher.tno where schedule.coursecode='{coursecode}'")

rows = cursor.fetchall()

for row in rows:

    ## 获取星期节次不同的记录

    cursor.execute(

        f"select day, index from schedule join selection on selection.cno=schedule.cno where day={row[2]} and index={row[3]}")

    course\_rows = cursor.fetchall()

    ## 没有记录就直接添加

    if not len(course\_rows):

        choosible\_class\_list.append(

            {"cno": row[0], "semester": row[1], "day": row[2], "index": row[3], "classroom": row[4],

                "optional": row[5], "selected": row[6], "startweek": row[7], "endweek": row[8], "tno": row[9],

                "tname": row[10], "cname": row[11]})

    else:

        # 如果有，就进行交叉判断

        cursor.execute(

            f"select startweek, endweek from schedule join selection on selection.cno=schedule.cno where day={row[2]} and index={row[3]}")

        time\_rows = cursor.fetchall()

        for item in time\_rows:

            # 起始时间比结束时间晚，结束时间比起始时间早

            if int(row[7]) > item[1] or int(row[8]) < item[0]:

                choosible\_class\_list.append(

                    {"cno": row[0], "semester": row[1], "day": row[2], "index": row[3], "classroom": row[4],

                        "optional": row[5], "selected": row[6], "startweek": row[7], "endweek": row[8],

                        "tno": row[9],

                        "tname": row[10], "cname": row[11]})

cursor.close()

conn.close()

4.2.5、学生退课模块

在退课模块中，根据用户已经选择的课程进行选择，点击退课前端向后端后端传递课程序号，根据课程序号，删除删除选择表中的记录，同时到课程计划表中对对应的课号已选人数-1，该模块的界面如图4-8所示：



**图4-8学生退课界面**

## 4.3、教师端

4.3.1、教师个人信息查询模块

用户在登录后，点击信息查看，前端会向后端发送请求，并传递用户的userid，根据userid用户返回对应的个人信息，这里和学生端使用的是不同的接口，但是信息的展示方式是一样的，如图4-9所示：



**图4-9教师个人信息界面**

4.3.2、教师课程安排查询

在这项功能中将会将当前用户的课程计划们进行简化，并展示在我的课程中，主要展示老师的课号，以及课程名称和选择人数等信息。界面如图4-10所示：



**图4-10教师课程界面**

4.3.3、教师课程计划查看和添加模块

在这个功能中用户可以查看自己的课程计划，并且可以点击右上角的课程计划进行添加课程计划。一级界面的展示如图4-11所示：



**图4-11教师课程计划界面**

点击添加课程计划后，可以填写对应的信息，然后生成自己的课程计划，和学生选课的过程差不多，也需要根据老师的课程安排，进行排课冲突判断，在学生判断冲突的基础上添加了教室冲突的判断，如果开始周，起始周，星期，节次均冲突的情况下，在看看教室是否冲突，若不冲突即可添加课程计划。在点击课程名称时会弹出可以选择的课程，选择完成后会自动生成对应的课程代码。教师添加课程计划的界面如图4-12所示：



**图4-12教师课添加程计划界面**

4.3.4、教师删除课程计划

根据一般教务系统，只有教师完成课程计划后学生才可以开始选课，所以删除了课程计划对学生的选课没有太大的影响，因此可以直接在对应的表格中进行删除，删除界面如图4-13所示：



**图4-13教师删除课程计划**

# 5、测试

## 5.1、登录界面测试

(1) 不输入用户名和密码，结果如图5-1所示：



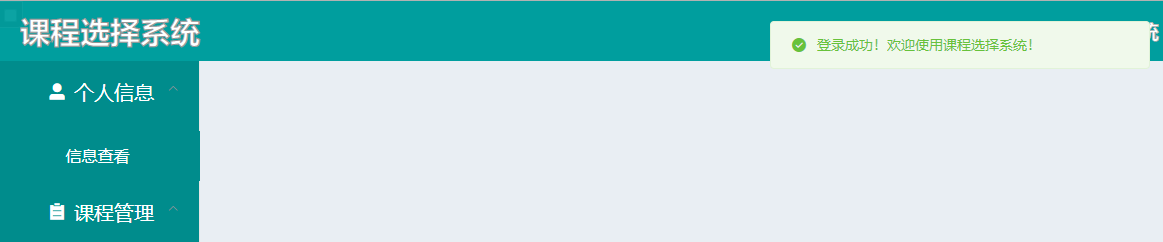
**图5-1登录界面不输入**

（2）密码或用户名错误，或客户端选择错误如图5-2所示：



**图5-2用户名，密码输入错误或客户端选择错误**

(3)用户名，密码以及客户端正确，如图5-3所示：



**图5-3用户名，密码输入正确以及客户端正确**

# 6、关键技术

## 6.1、前端关键技术总结

(1) 使用现在流行的Vue前端框架，搭配上Element-UI的组件，使前端在设计过程中,更加具有项目规范化，Vue是使用Node.js的环境上搭建的，具有项目打包和局域网部署功能。同时Vue也是视图级别的前端框架，对数据的动态变化十分友好。Element-UI是现在最火爆的UI组件，通过调用这些UI组件使得前端界面的设计更加美观。

(2) 使用了JavaScript中AJAX异步通信技术，通过AJAX通信，从后端获取对应的数据信息，加上Vue框架对动态数据有极高的贴合性，使得课程选择系统的Web设计更加灵活。

## 6.2、后端关键技术总结

(1) 使用企业级微型后端框架Flask搭建后端，该框架对后端常用的方法进行高度封装，简化搭建后端的过程，对于一个小型项目而言，能够快速实现一个负责数据处理的后端，具有很高的易上手度。

(2) 使用Python作为后端开发语言，能够在极短时间内上手并实现，代码具有较强的可读性和健壮性。通过设计规定好的数据传输格式，可以实现前后端分离的设计模式，使得开发更具备灵活性。同时，通过Python对应数据库相关的功能包能够快速实现数据库连接以及对数据库进行操作，直接通过代码操作数据库，只需要一次连接数据库即可操作，且连接速度快，更新效率高。

# 7、存在的问题及改进思路

7.1 存在的问题

1. 在开发过程中，前期开发时没有意识到文档的重要性，导致在进行前后端对接的时候出现了很多问题，比如，前端没有正确解析后端数据、后端逻辑出错导致前端错误等问题，降低了开发的效率。

2. 开发过程中，因为没有对要查询的目标分析清楚导致SQL语句出错导致选出了错误的数据。

3. 本项目使用SQL语句进行查询时因为关系模式设计满足BCNF，导致在查询时需要频繁使用自然连接，降低查询效率。

4. 高并发读写数据库时会有较大的延迟。

7.2 改进思路

1. 进一步完善该系统的整体功能以及用户体验，优化用户细节。

2. 规范化文档书写，完善文档内容以及描述。

3. 对使用自然连接SQL语句进行优化，将一些自动化的操作使用触发器来解决。

4. 将数据库吞吐量较大的数据存储到分布式数据库中，并且引入并发支持。

# 8、总结

本次实验为团队的综合实验，考验的不仅仅是技术实力更是团队成员之间的沟通能力以及写作能力。最初设计课程选择系统的时候把系统想得过于简单，导致在中期实行的时候处处碰壁。在完成本次综合实验报告后，我们团队第一次完成了前后端分离开发，即使是前后端分离，但接口的具体实现，数据库中调度数据的方式也是团队成员在不断的讨论中才最终确定的。每一次讨论都是对课程选择系统的一次完善。而每一个课程选择系统功能的实现也是对自己技术和能力的提升。

就比如系统中排课冲突和选课冲突的解决方案，也是在团队的讨论中不断寻找最优的解决方案和算法最后得以落实的，这就是我们从团队合作中得到的益处。在系统设计过程中，其实比较艰难的一段时间是最初的系统设计上。系统设计可以说是很重要的一环，最初就是在这上面的时间过少了，问题考虑过于简单，导致后期要进行极大的修改，但是最终还是在不懈地努力中完成了课程选择系统的重构。通过本次系统设计，明白了需求分析的重要性，在实现的过程中总是动不动就把接口的名字和功能忘记了，如果在设计初期能够有一个比较完善的文档，就可以减少没有必要的重复过程。

最后团队的合作，合理分工以及对系统实现的热情，对工作的负责，都是成功完成系统必不可少的因素，因此，在本次实验中，我们对合作有了更深的认识，也体会到了合作的重要性，沟通的重要性，已经文档编写的重要性，同时在这个过程中也巩固了个人的软件开发能力，让我们受益匪浅。