V_campus 虚拟校园系统项目总结

——软件实践(JAVA)项目报告

组长： 曹邹颖　　学号：09019204　　邮箱：1799771645@qq.com

分工：软件设计说明书、小组报告的撰写，公共模块，教务管理模块

所有功能模块的整合工作；

组员：

宗琦薇　学号：09019206　　分工：整体 UI 美化用户管理模块

程瀚　　学号：09019120　　分工：网络连接模块学生信息管理模块

许志豪　学号：09019231　　分工：虚拟银行模块

汪晟宇　学号：09019230　　分工：虚拟图书馆模块

苏哲琪　学号：09019229　　分工：虚拟商店模块

一．前言

应东南大学专业技能实训课程要求，我们小组开发名为 V_campus 虚拟校园系统的软件系统。

关于我们团队所要设计的“虚拟校园系统”，随着学校的规模不断扩大，专业、班级、学生、教师等的数量急剧增加，有关校园生活的各种公共设施也成倍增长，因此，迫切需要开发基于 C/S 的虚拟校园管理系统来提高校园工作与生活的效率。**设计过程**首先根据软件需求分析得到程序系统的结构，然后规划每一个模块所要实现的功能与性能等，接着进行必要的网络模块、多线程模块与数据库设计，确定测试要求并且设定测试计划。

二．项目需求

本系统根据虚拟校园管理综合分析，便出于方便管理考虑，将虚拟校园管理系统的功能总结起来，共需要实现以下几个方面功能：

（一）用户管理

用户管理功能包括：用户注册、用户登录、用户登出、用户信息更改、用户个性化五个部分。

（二）虚拟学生管理

管理所有学生的基本信息，对学生信息的操作包括添加、修改、删除等；可以根据各种条件查询出需要的信息，比如修改，可以通过学生学号查询出学生的基本信息，然后通过对需要修改项进行修改并保存修改后的结果存入数据库的学生表中。

（三）虚拟教务

排课管理模块：管理所有课程的基本信息，包括对课程信息的添加、修改、删除等操作；可以根据各种条件查询出需要的信息，并对相应的信息进行操作，比如修改和删除，可以通过学生的学号查询出学生的相应课程信息，对相应的课程进行修改、删除操作；

选课管理模块：学生可对本人的课表信息进行查询，也可以根据培养计划进行选课；

考务管理模块：考试安排，管理员可以安排考场和考试时间；成绩录入，任课教师可以在规定的时间段内录入学生成绩；查询成绩，学生可以根据学号查询出课程成绩。

（四）虚拟图书馆

图书管理模块：管理员完成图书基本信息的管理（包括：添加、修改、查询、删除）；

借阅管理模块：学生、老师可线上进行图书的操作（包括：图书借阅、归还、查询、续借等）。

（五）虚拟商店

购物模块：创建订单，填写订单号(ID)、商品信息、订单状态、折扣等基本信息后创建订单；向订单中添加商品，进入已创建的订单中，向已创建订单中添加新的商品；删除订单，删除一个已创建的订单。

账户模块：在购物订单交易时，首先要确定用户余额足够支持交易，在交易成功后，用户账户余额会扣除相应的钱额；可通过虚拟校园的另一功能模块——虚拟银行，对账户余额进行一定相关操作。

（六）虚拟银行

老师、学生可管理自身的基本账户信息，对个人账户余额的操作包括存款、转账、查询、取款等，可以根据各种条件查询出需要的账户余额信息，甚至对于账户进行挂失操作。

管理员可管理所有师生的基本账户信息，对学生账户余额的操作包括查询、扣钱等，对于余额不足扣钱不成功的情况再将对应的学生名单存入数据库的一张表中，甚至对于账户进行封户操作。

通过以上功能的设计与实现，并实现虚拟校园管理系统的基本功能。当然在实际的虚拟校园管理系统中，其功能要比本系统软件设计的多的多，也复杂的多，本软件设计说明书仅以比较简单易懂的方式呈现—一个虚拟校园管理系统的设计与实现的流程。

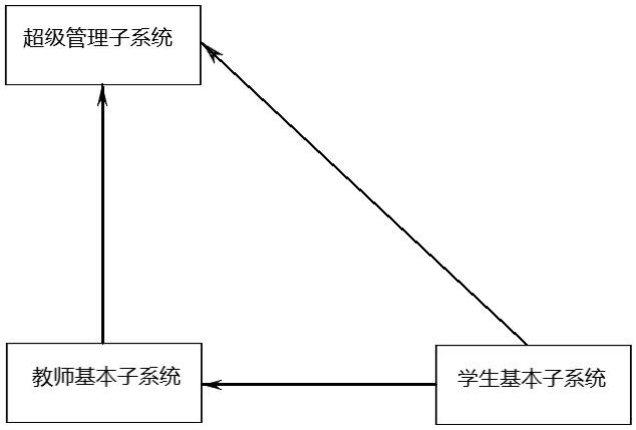
三．项目设计与开发

（一）系统设计

1. 各子系统的设计

本系统包括三个子系统：高级管理子系统，教师基本管理子系统，学生基本子系统，分别对应用户类型的三种：管理员、教师、学生。

三个子系统的关系如图 3.1：



3.1 三个子系统之间的关系

三个子系统都包括六个模块分别为：用户管理模块、学生管理模块、教务模块、图书馆模块、商店模块、银行模块。

2. 软件控制流设计

(1) 添加操作

系统将向数据库中添加一条新的记录，并修改相关表的信息（如在添加教师信息时，系统将自动修改教师的基本信息）。

(2) 查询操作

系统根据查询条件在数据库中进行查询，然后把查询结果显示在界面上。

(3) 修改操作

要进行修改操作，必须先执行查询操作。执行修改操作系统将更新数据库中的相应记录。

(4) 删除操作

要进行删除操作，必须先执行查询操作。执行删除操作系统将删除数据库中的相应记录。

3. 数据库设计

(1) 概念设计

实体——联系方法(Entity—Relationship Approach)是最常用的表示概念性数据模型的方法。这种方法使用 E-R 图来描述现实世界中的实体，而不涉及这些实体在系统中的实现方法，即使不熟悉计算机技术的用户也能理解它。根据 E-R 图中的实体，画出每个实体的属性图，如图 3.2-3.14 所示：

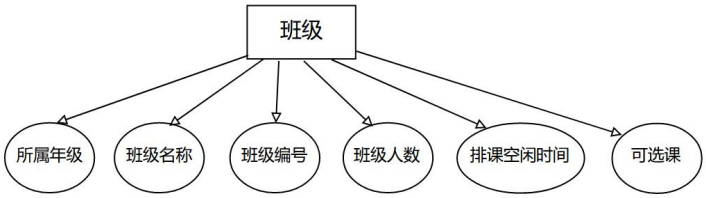


图 3.2. 班级实体属性图

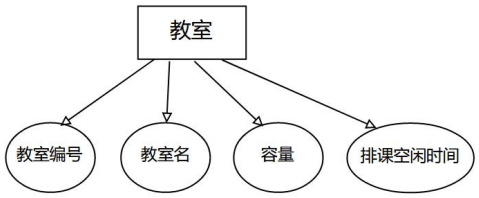


图 3.3. 教室实体属性图

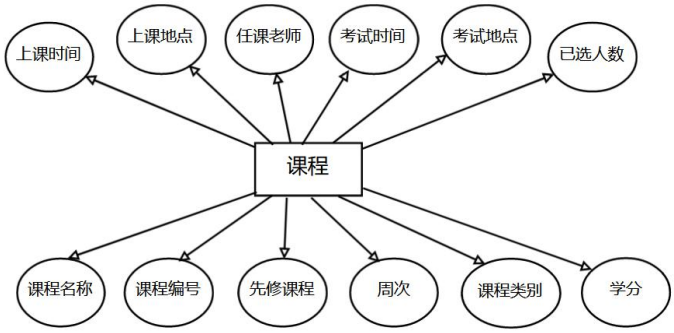


图 3.4. 课程实体属性图

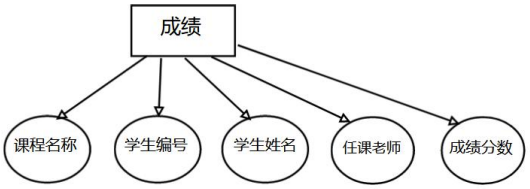


图 3.5. 成绩实体属性图

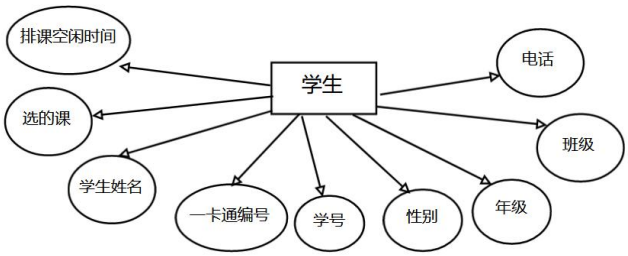


图 3.6. 学生实体属性图

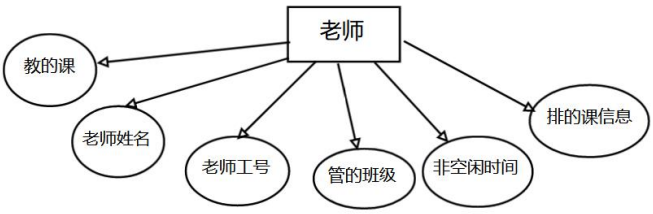


图 3.7. 老师实体属性图

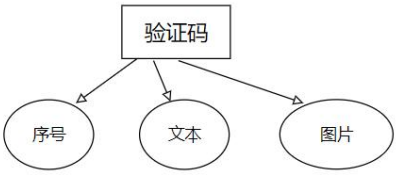


图 3.8. 验证码实体属性图

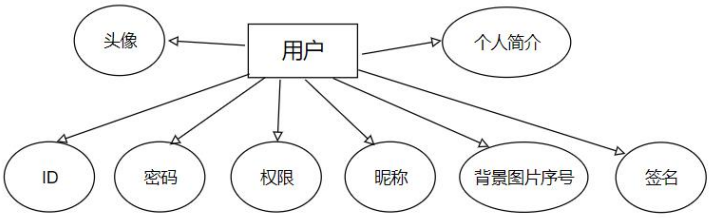


图 3.9. 用户实体属性图

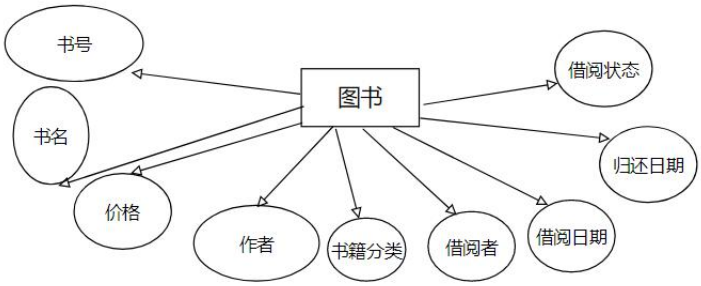


图 3.10. 图书实体属性图

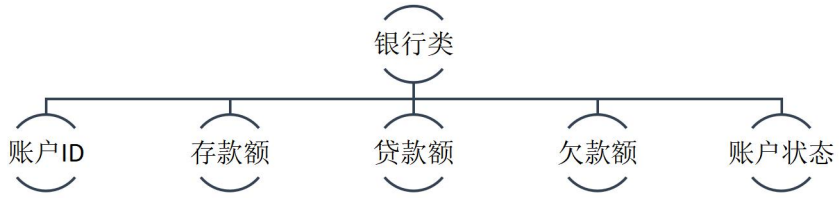


图 3.11. 银行实体属性图

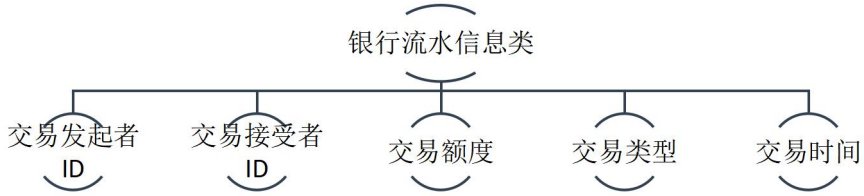


图 3.12. 银行流水信息实体属性图

图 3.13. 商店实体属性图

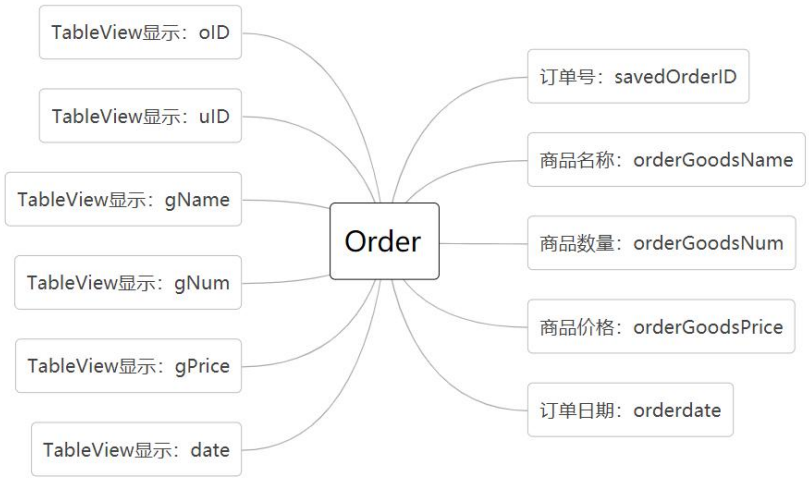


图 3.14. 交易实体属性图

(2) 数据库涉及的表

本数据库命名为 student，涉及到 14 个表，如下所示。

bookinfo (bookId, bookName, state, price, writer, address, borrower, borrowDate, returnDate)
goodsstore (goodsName, goodsNum)
salerecord (orderID, buyerID, goodsName, goodsNum, goodsPrice, date)
tb_borrow(bookId, borrower, bookName, state, borrowDate, returnDate)
tb_class (id, name, grade, sum, observed)
tb_classroom (id, name, capacity)
tb_course (id, name, timesPerWeek, preCourse, classification, credit, roomId, courseDate, teacher, testDate, testAddress, choosed)
tb_grade (id, studentID, studentName, teacherName, courseName, grade)
tb_student (id, studID, sno, name, sex, className, choosedCourse, grade, contact)
tb_teacher (id, teachId, name, course, busyName, observed, incharge)
tb_user (ID, Password, status, Username, Signature, Selfinfo, Background, Headpic, Online)
tb_vcode (AID, vcodetext, vcodepic)
tblbank (bID, bDeposit, bLoad, bDebit, bState)
tblbankrecord (originator_ID, recipient_ID, transaction_mode, transaction_amount, transaction_time)

(3) 各表的物理结构

本系统所包含的 14 张表 bookinfo，goodsstore，salerecord，tb_borrow，tb_class，tb_classroom，tb_course，tb_grade，tb_student，tb_teacher，tb_user，tb_vcode，tblbank，tblbankrecord 的物理结构如表 3.1-3.14 所示：

表 3.1 tb_class

字段名称	类型	属性
id	INT	PK，NN
grade	INT	NN
name	VARCHAR(255)	NN
sum	INT	Default:NULL
observed	VARCHAR(5000)	Default:NULL

表 3.2 tb_classroom

字段名称	类型	属性
id	INT	PK，NN
name	VARCHAR(255)	NN
capacity	INT	NN

表 3.3 tb_course

字段名称	类型	属性
id	INT	PK，NN
name	VARCHAR(255)	NN
timesPerWeek	INT	Default:NULL
preCourse	VARCHAR(255)	Default:NULL
classification	VARCHAR(45)	Default:'必修'
credit	DOUBLE	Default:'0'
roomId	VARCHAR(255)	Default:NULL
courseDate	VARCHAR(255)	Default:NULL
teacher	VARCHAR(255)	Default:NULL
testDate	VARCHAR(255)	Default:'无考试安排'
testAddress	VARCHAR(255)	Default:'无考试安排'
choosed	INT	Default:'0'

表 3.4 tb_grade

字段名称	类型	属性
id	INT	PK，NN
studentID	VARCHAR(45)	Default:'213'
studentName	VARCHAR(45)	Default:NULL
courseName	VARCHAR(45)	Default:NULL
teacherName	VARCHAR(45)	Default:NULL
grade	DOUBLE	Default:NULL

表 3.5 tb_student

字段名称	类型	属性
id	INT	PK，NN，AI
studID	VARCHAR(45)	Default:NULL
sno	VARCHAR(255)	Default:NULL
name	VARCHAR(255)	Default:NULL

sex	VARCHAR(45)	Default:NULL
className	VARCHAR(255)	Default:NULL
choosedCourse	VARCHAR(1024)	Default:""
grade	VARCHAR(45)	Default:NULL
contact	VARCHAR(255)	Default:NULL

表 3.6 tb_teacher

字段名称	类型	属性
id	INT	PK，NN，AI
teachID	VARCHAR(45)	Default:'103'
name	VARCHAR(255)	Default:NULL
course	VARCHAR(255)	Default:NULL
busyTime	VARCHAR(255)	Default:""
observed	VARCHAR(5000)	Default:NULL
incharge	VARCHAR(255)	Default:NULL

表 3.7 tb_user

字段名称	类型	属性
ID	VARCHAR(225)	PK，NN
Password	VARCHAR(225)	NN
Status	INT	NN
Username	VARCHAR(45)	Default:NULL
Signature	VARCHAR(45)	Default:NULL
Selfinfo	VARCHAR(225)	Default:NULL
Background	INT	Default:NULL
Headpic	BLOB	Default:NULL

表 3.8 tb_vcode

字段名称	类型	属性
AID	INT	PK，NN
vcodetext	VARCHAR(45)	Default:NULL
vcodepic	BLOB	Default:NULL

表 3.9 tb_bookinfo

字段名称	类型	属性
bookId	VARCHAR(45)	PK、NN
bookName	VARCHAR(45)	Default:NULL
state	VARCHAR(45)	Default:NULL
price	VARCHAR(45)	Default:NULL
writer	VARCHAR(45)	Default:NULL
address	VARCHAR(45)	Default:NULL
borrower	VARCHAR(45)	Default:NULL
borrowDate	VARCHAR(45)	Default:NULL
returnDate	VARCHAR(45)	Default:NULL

表 3.10 tb_borrow

字段名称	类型	属性
bookId	VARCHAR(45)	Default:NULL
bookName	VARCHAR(45)	Default:NULL

state	VARCHAR(45)	Default:NULL
borrower	VARCHAR(45)	Default:NULL
borrowDate	VARCHAR(45)	PK、NN
returnDate	VARCHAR(45)	Default:NULL

表 3.11 tblbank

字段名称	类型	属性
bid	VARCHAR(50)	PK，NOT NULL
bdeposit	DOUBLE	DEFAULT: 0.0
bdebit	DOUBLE	DEFAULT: 0.0
bload	DOUBLE	DEFAULT: 0.0
bstate	INTEGER	DEFAULT: 1

表 3.12 tbl_bankrecord

字段名称	类型	属性
originator_id	VARCHAR(50)	DEFAULT: NULL
recipient_id	VARCHAR(50)	DEFAULT: NULL
transaction_mode	VARCHAR(50)	DEFAULT: NULL
transaction_amount	VARCHAR(50)	DEFAULT: NULL
transaction_time	VARCHAR(50)	DEFAULT: NULL

表 3.13 goodsstore

名	类型	长度	小数点	不是 null	虚拟	键
orderID	varchar	255		☑	☐	🔑1
buyerID	varchar	255		☐	☐	
goodsName	varchar	255		☐	☐	
goodsNum	int			☐	☐	
goodsPrice	double	10	2	☐	☐	
date	datetime			☐	☐	

表 3.14 salerecord

名	类型	长度	小数点	不是 null	虚拟	键
goodsName	varchar	255		☑	☐	🔑1
goodsNum	int			☐	☐	

（二）程序设计

1.程序包的设计

Client 端：

net 包：提供网络连接模块所需要的类；

srv 包：提供工具类；

view 包：提供界面设计相关的类；

vo 包：提供实体类；

Server 端:

dao 包：数据存取类包；

net 包：提供网络连接模块所需要的类；

srv 包：提供工具类；

util 包：提供数据库工具类；

vo 包：提供实体类。

2.包中程序设计

Client 端：

Client 类：

方法：

public NetConnetcion getNetConnection() // 返回 NetConnetcion 对象
public void sendCloseMessage() // 向服务端发送中止连接消息
public int connectionDisabled() // 显示网络错误提示对话框
public void close() // 关闭客户端
public void logout(String id) // 向服务端发送退出登录消息

(1) net 包中的程序设计

ChatRecevier 类：实现 Runnable 接口

方法：

public ChatRecevier(Client s, UserSpaceViewCtrl rUserSpaceViewCtrl) // 构造方法，初始化 Client 和 UserSpaceViewCtrl 变量
public void run() // 重载，监听服务端发来的聊天消息

NetConnection 类：

方法：

public NetConnection(Client s) // 构造方法，初始化变量，调用网络连接的方法
public void netConnetion() // 和服务器连接，获得 socket，赋值输入输出流
public void chatNetConnection() // 和服务器连接，获得专门用于聊天的 socket，赋值聊天用的输入输出流
public int connectionDisabled() // 连接断开时的处理函数，对话框提示重连
public synchronized Message receive() // 监听线程监听到服务端发送消息后，同步接受消息
public synchronized void send(Message mes) // 同步向服务端发送消息
public Message chatReceive() // 聊天监听线程监听到服务端发送消息后，同步接受聊天消息
public void chatsend(Message mes) // 同步向服务端发送聊天消息

ReceiveListener 类：实现 Runnable 接口

方法：

public void run() // 重载，监听并接受输入流中的消息，根据消息类型调用相应的处理方法

(2) srv 包中的程序设计

EducationalAdminService 类：

方法：

public void updateScheduling(Message rmes){} //更新状态函数,处理服务端消息函数
public int getSchedulingState(){} // 状态 getter 函数
public String[] getObservedArray(){} // observed 数组 getter 函数
public Course getCourse() {}// course 变量 getter 函数
public Grade getGrade(){} // grade 变量 getter 函数
public ArrayList<Double> getCredit(){} // credit 变量 getter 函数
public void buttonScheduling(Client client){} // 自动排课响应函数
public void buttonShowClass(Client client,String name){} // 显示班级课表函数
public void buttonShowTeacher(Client client,String name){} // 显示老师课表函数
public void buttonCheckCourse(Client client,String name,int ID){} // 查询课程函数
public void buttonAddCourse(Client,int,String,int,String,String,double)//添加课程函数
public void buttonDeleteCourse(Client client,int id){} // 删除课程函数
public void buttonUpdateCourse(Client client,Course cCourse){} // 更新课程信息函数
public void buttonUpdateTest(Client client, Course cCourse) {} // 更新考务信息函数
public void buttonShowTest(Client client,String name, int id){} // 显示考务信息函数
public void buttonAddTest(Client client,int id,String,String){} // 增加考务信息函数
public void buttonDeleteTest(Client client, int id) {} // 删除考务信息函数
public void buttonShowGrade(Client client,String name,int index){} // 查询成绩函数

public void buttonGradePerformance(Client client, String ID){}//显示课程班成绩函数
public void buttonTotalStudent(Client client, String ID) {} // 显示所有学生函数
public void buttonTeacher(Client client) {} // 显示所有老师函数
public void buttonTotalGrade(Client client) {} // 显示所有班级函数

EducationalTeachService 类：

方法：

public void updateState(Message rmes) {} // 更新状态函数，处理服务端消息函数
public int getState() {} // state 变量 getter 函数
public Course getCourse() {} // course 变量 getter 函数
public String[] getObserved() {} // observed 变量 getter 函数
public void buttonUpdateGrade(Client client, String ,String ,double) {} //录入成绩函数
public void buttonTeachCourse(Client client, String ID) {} // 显示自己所教课程函数
public void buttonTeachStudent(Client client, String ID) {} // 显示自己所教学生函数

EducationalStudService 类：

方法：

public void updateState(Message rmes) {} // 更新状态函数，处理服务端消息函数
public int getState() {} // state 变量 getter 函数
public Course getCourse() {} // course 变量 getter 函数
public String[] getObserved() {} // observed 变量 getter 函数
public ArrayList<Course> getCourseArray(){} // courseArray 变量 getter 函数
public void buttonChooseCourse(Client client, String,String) {} // 选课函数
public String[] buttonShowCourse(Client client, int studID) {} // 显示学生课程函数
public void buttonMatchCourse(Client client, String name) {}//显示课程所选学生函数
public void buttonShowCourseForm(Client client, String ID) {} // 显示学生课表函数
public void buttonCourse(Client client, String ID) {} // 查看所选课程函数
public void buttonDeleteCourse(Client client, String ID, String name) {} // 退课函数

InformationService 类：

方法：

public InformationService(Client c) // 构造方法，初始化 Client 变量
public void quaryList(String studentID,String classID) // 向服务端发送查询学生列表的消息，并等待响应
public int getQuaryListState() // 返回查询状态
public void setList(Message mes) // 将服务端发来的消息转换为 Student 类实体并存入列表
public ArrayList<Student> getList() // 返回学生列表
public void modStu(Student stu) // 向服务端发送编辑学生信息的消息，并等待响应
private int getModStuState() // 返回编辑状态
public void modRespond(Message mes) // 更新编辑状态
public void addStu(Student stu) // 向服务端发送添加学生信息的消息，并等待响应
private int getAddStuState() // 返回添加状态
public void addRespond(Message mes) // 更新添加状态
public void deleteStu(Student stu) // 向服务端发送删除学生信息的消息，并等待响应
int getDeleteStuState() // 返回删除状态
public void deleteRespond(Message mes) // 更新删除状态
public void quaryThisTeacher(String id) // 向服务端发送查询当前老师信息的消息，并等待响应
int getQuaryTeacherState() // 返回查询当前老师状态
public void quaryTeacherRespond(Message mes) // 更新查询当前老师状态，保存当前老师信息
public Teacher getTeacher() // 返回当前老师信息的实体

public void quaryThisStudent(String id) // 向服务端发送查询当前学生信息的消息，并等待响应
private int getQuaryThisStudentState() // 返回查询当前学生信息状态
public void quaryThisStudentRespond(Message mes) // 更新查询当前学生状态，保存当前学生信息
public Student getStudent() // 返回当前学生实体

UserService 类：

int loginState=0;//登陆状态
int state=0;//获取用户信息
User user;//用户实体
String vcodetext;
Blob vcode;

方法：

public void queryChatMember(Client client,String id) // 向服务端发送查询聊天列表的消息，并等待响应
private int getQueryChatMemberState() // 返回查询聊天列表状态
public void queryChatMemberRespond(Message mes) //
更新查询聊天列表状态，从消息中获得 User 实体，并保存在列表里
public ArrayList<User> getChatMemberChatList() //
返回聊天列表
public void checkChatMember(Client client, String id) //
向服务端发送确认在线的消息，并等待响应
private int getCheckChatMemberState() //
返回确认在线状态
public void checkChatMemberRespond(Message mes) //
更新确认在线状态，并保存
public int getOnlineState() //
返回在线状态

方法：

public void updateLogin(Message rmes){}//初始化 loginState 变量
public User getUser(String rid,String rpassword,int rstatus){}//返回一个实体类 User
public void updateUser(Message rmes){}//根据消息返回类型更新变量
public void userSelf(Client client,String s) {}//用 ID 去查其他信息
public void changeinfo(Client client,String id,String field,String newstr) {}//用 ID 去查数据库修改信息，密码/昵称/签名/个人简介
public void changebg(Client client,String id,String field,int newbg) {}//用 ID 去查数据库改背景图片序号
public void buttonLogin(Client client,String s,String p){}//用于登录界面时消息响应函数，将通过界面输入的信息打包为 message 类进行通信
public void getvcode(Client client,int s) {}//用验证码序号（0-22）去查图片
public void changehead(Client client,String id,String field,String newhead) {}//用 ID 去查数据库改背景图片序号
以及相应 set、get 函数

BookService 类：

int bkState //判断操作响应状态
Book bk //book 对象的引用
ArrayList<Book> bookList //存放所有图书信息
ArrayList<Book> borrowList //存放所有借阅信息

方法：

public void updateState(Message rmes) // 读取 message 更新类变量

public void btnAddBook(Client client,String id, String name, String state, String price, String writer,String address,String borrower,String borrowDate,String returnDate) // 添加书籍时消息响应函数，将通过界面输入的信息打包为 message 类进行通信

public void buttonUpdateBook(Client client,Book book) // 用于修改图书资料时消息响应函数，将通过界面输入的信息打包为 message 类进行通信

public void buttonDeleteBook(Client client, String bookId) // 用于删除图书时消息响应函数，将 bookId 打包为 message 类进行通信

public void buttonQuery(Client client, String id) // 用于管理员查询书籍时消息响应函数，将 id 信息打包为 message 类进行通信

public void buttonBorrowQuery(Client client, String name) // 用于用户查询书籍信息时消息响应函数，将 name 信息打包为 message 类进行通信

public void buttonBorrowBook(Client client,Book book) // 用于用户借阅书籍时消息响应函数，将 book 信息打包为 message 类进行通信

public void buttonBorrowInfoQuery(Client client, String name, String borrowerID) // 用于用户查询书籍借阅记录时消息响应函数，将书籍名称，借阅者 ID 信息打包为 message 类进行通信

public void buttonReturnBook(Client client, Book book) // 用于用户还书时消息响应函数，将 book 信息打包为 message 类进行通信

public void buttonRenewBook(Client client, Book book) // 用于用户续借时消息响应函数，将 book 信息打包为 message 类进行通信

public void buttonShowBook(Client client) // 用于浏览书籍信息时消息响应函数，通过 message 类进行通信

public void buttonShowBorrow(Client client, String id) // 用于用户浏览借阅书籍记录消息响应函数，将 id 信息打包为 message 类进行通信

BankService 类：

方法：

updateanswerSelfInfoOKID (Message mes)
updateanswerSelfInfoOKDeposit(Message mes)
updateanswerSelfInfoOKLoad(Message mes)
updateanswerSelfInfoOKDebit(Message mes)

功能：在一般银行用户点击 UI 上的的个人信息 Label 后从数据库中取出它的个人信息并且用来更新。

传入参数：Socket 中传递的 Message。

方法：

answerSelfInfoOK(String bID)

功能：向 Server 端发送用户查询个人信息的 Message。

传入参数：用的 ID。

方法：

getAnswerSelfInfoID()
getAnswerSelfInfoDeposit()
getAnswerSelfInfoDebit()
getAnswerSelfInfoLoad()

功能：获得在银行类里获得银行用户的个人信息

传入参数：无

方法：

updateanswerSelfInfoOKBankRecordList(Message mes)

功能：在一般银行用户点击 UI 上的的个人信息 Label 后从数据库中取出它的个人流水并且用来更新。

传入参数：Socket 中传递的 Message。

方法：

getAnswerSelfInfoBankRecord()

功能：获得在银行类里获得银行用户的个人流水信息

传入参数：无

方法：

updateanswerDeposit_amountOKState(Message mes) ()

功能：更新在一般银行用户点击 UI 上的的确认存款按钮后从数据库中将用户输入的存款额度更新到数据库中，有没有成功存入的状态

传入参数：Socket 中传输的信息

方法：

answerDeposit_amountOK (double newDepositamount, String bID)

功能：将用户存款的信息发送到 Message 中

传入参数：用户的存款额度和用户的 ID

方法：

getAnswerDeposit_amountOKState()

功能：获得用户存款成功的信息状态

传入参数：无

方法：

updateanswerLoad_amountOKState(Message mes)

功能：更新在一般银行用户点击 UI 上的的确认贷款按钮后从数据库中将用户输入的存款额度更新到数据库中，有没有成功存入的状态

传入参数：Socket 中传输的信息

方法：

answerLoad_amountOK(double newLoadamount, String bID)

功能：将用户贷款的信息发送到 Message 中

传入参数：用户的贷款额度和用户的 ID

方法：

getAnswerLoad_amountOKState()

功能：获得用户贷款成功的信息状态

传入参数：无

方法：

updateanswerRepay_amountOKState(Message mes)

功能：更新在一般银行用户点击 UI 上的的确认还款按钮后从数据库中将用户输入的存款额度更新到数据库中，有没有成功存入的状态

传入参数：Socket 中传输的信息

方法：

answerRepay_amountOK(double newLoadamount, String bID)

功能：将用户还款的信息发送到 Message 中

传入参数：用户的贷款额度和用户的 ID

方法：

getAnswerRepay_amountOKState()

功能：获得用户还款成功的信息状态

传入参数：无

方法：

updateAdminTransactionInfoOKState(Message mes)

功能：更新在银行管理员点击 UI 上的的交易信息 Label 后从数据库中将所有用户流水信息取出有没有成功存入的状态

传入参数：Socket 中传输的信息

方法：

updateAdminTransactionInfo(Message mes)

功能：更新在银行管理员点击 UI 上的的交易信息 Label 后从数据库中将所有用户流水信息取出

传入参数：Socket 中传输的信息

方法：

adminTransactionInfoOK()

功能：将获取流水信息的信息发送到 Message 中
传入参数：无

方法：
getAdminTransactionInfoOKState ()
功能：获得流水信息成功的信息状态
传入参数：无

方法：
getAdminTransactionInfo()
功能：获得流水信息
传入参数：无

方法：
updateAdminQueryTransactionInfoOKState (Message mes)
功能：更新在银行管理员点击 UI 上的的确认查询交易信息后从数据库中特定用户流水信息取出有没有成功存入的状态
传入参数：Socket 中传输的信息

方法：
updateAdminQueryTransactionInfo (Message mes)
功能：更新在银行管理员点击 UI 上的确认查询交易信息后从数据库中将特定用户流水信息取出
传入参数：Socket 中传输的信息

方法：
adminQueryTransactionInfoOK ()
功能：将获取流水信息的信息发送到 Message 中
传入参数：无

方法：
getAdminQueryTransactionInfoOKState ()
功能：获得流水信息成功的信息状态
传入参数：无

方法：
getAdminQueryTransactionInfo ()
功能：获得流水信息
传入参数：无

方法：
updateAdminUserInfoOKState (Message mes)
功能：更新在银行管理员点击 UI 上的用户信息 Label 后从数据库中将所有的用户信息取出成功与否的状态
传入参数：Socket 中传输的信息

方法：
updateAdminUserInfo (Message mes)
功能：更新在银行管理员点击 UI 上的用户信息 Label 后从数据库中将所有的用户信息取出
传入参数：Socket 中传输的信息

方法：
adminUserInfoOK ()
功能：将获取用户信息的信息发送到 Message 中
传入参数：无

方法：
getAdminUserInfoOKState ()
功能：获得用户信息成功的信息状态

传入参数：无

方法：

getAdminUserInfo ()

功能：获得用户信息

传入参数：无

方法：

updateAdminQueryUserInfoOKState (Message mes)

功能：更新在银行管理员点击 UI 上的查询用户信息按钮后从数据库中将所有的用户信息取出成功与否的状态

传入参数：Socket 中传输的信息

方法：

updateAdminQueryUserInfo (Message mes)

功能：更新在银行管理员点击 UI 上的用户信息按钮后从数据库中将特定的用户信息取出

传入参数：Socket 中传输的信息

方法：

adminQueryUserInfoOK ()

功能：将获取用户信息的信息发送到 Message 中

传入参数：无

方法：

getAdminQueryUserInfoOKState ()

功能：获得用户信息成功的信息状态

传入参数：无

方法：

getAdminQueryUserInfo ()

功能：获得用户信息

传入参数：无

方法：

updateAdminDeleteOKState (Message mes)

功能：更新在银行管理员点击 UI 上的用户信息删除后从数据库中将特定的用户信息更新

传入参数：Socket 中传输的信息

方法：

adminDeleteOK(String bID)

功能：更新用户信息

方法：

getAdminDeleteOKState ()

功能：更新用户信息成功的信息状态

传入参数：无

方法：

updateAdminInvokeOKState (Message mes)

功能：更新在银行管理员点击 UI 上的用户信息激活后从数据库中将特定的用户信息更新

传入参数：Socket 中传输的信息

方法：

adminInvokeOK (String bID)

功能：更新用户信息

传入参数：无

方法：

getAdminInvokeOKState ()

功能：更新用户信息成功的信息状态

传入参数：无

ShopService 类：

方法：

public void updateService(Message mes) // 接受 Server 端传回的信息对状态更新。
public boolean goodsIsEmpty() // 判断购物车中是否有货物。
public void emptyGoods() // 清空货物。
public void buyGoods(String name,int num,double price) // 购买 name 物品 num 件，总价为 price。
public boolean returnGoods(String name,int num) // 退回 name 物品 num 件。
public int sendOrder(Client client,String id) // 向服务器发送用户账号为 id 的发送订单请求。
public void queryOrder(Client client,String userID,String time) // 向服务器发送查询 id 用户在 time 时间的订单。
public void sendChangedGoods(Client client) // 向服务器发送更改货物数量的信息。
public void queryInventory(Client client) // 向服务器发送查询商店库存的信息。
public void getAccount(Client client,String uid) //向服务器发送查询用户余额的信息

(3) view 包中的程序设计

AdminEducationalCtrl 类：

方法：

public void setClient(Client c) {} // client 变量 setter 函数
public void setID(String c) {} // ID 变量 setter 函数
void cancel(ActionEvent event) {} // 进入时选择退出界面跳转函数
void clearTextField(TextField x) {} // 清空文本框内容的函数
void logout(MouseEvent event) throws Exception {} // 退出系统函数
void lblUserManageClick(MouseEvent event) {} // 跳转用户管理模块函数
void lblStudManageClick(MouseEvent event){} // 跳转虚拟学生管理模块函数
void lblLibManageClick(MouseEvent event){} // 跳转虚拟图书馆模块函数
void lblShopManageClick(MouseEvent event) {} // 跳转虚拟商店模块函数
void lblBankManageClick(MouseEvent event) {} // 跳转虚拟银行模块函数
public void initialize(URL location, ResourceBundle resources) {} // 时间设置函数
void lblCourseAdminClick(MouseEvent event) {} // 课程管理模块切入函数
void btnAddCourseClick(ActionEvent event) {} // 点击开始执行添加课程操作函数
void cbClassificationClick(ActionEvent event) {} // 初始化课程类别的下拉框内容
void btnAddCheck(ActionEvent event) {} // 添加课程的确认按钮事件函数
void btnAddDelete(ActionEvent event) {} // 添加课程的取消按钮事件函数
void btnClearCourseClick(ActionEvent e) // 清空表中呈现的已添加的课表考务信息
void lblSchedulingClick(MouseEvent event) {} // 排课管理模块切入函数
void AutoScheduling(ActionEvent event) {} // 自动排课按钮事件函数
void gradeCourse(ActionEvent event) {} // 显示班级课表函数
void teacherCourse(ActionEvent event) {} // 显示老师课表函数
内部类：public class tbCourseForm {} // 课表 tableView 的数据模型
public void ShowAll(TableView<?> ,String[] , ObservableList <?>) {}//显示课表函数
void lblTestAdminClick(MouseEvent event) {}// 考务管理模块切入函数
void btnTestManageClick(ActionEvent event) {}// 实现考务管理模块切入函数
void btnAddTestCheckClick(ActionEvent event) {}// 确认增加考务函数
void btnAddTestDeleteClick(ActionEvent event) {} // 便于清空已添加的考务信息
void lblCheckCourseClick(MouseEvent event) {} // 课程查询模块切入函数
void btnCheckCourseClick(ActionEvent event) {}// 课程查询确认函数
void btnClearCourseClick1(ActionEvent event) {} // 便于清空已查询到的课表信息
void lblCheckTestClick(MouseEvent event) {} // 考务查询模块切入函数

void btnCheckCourseTestClick(ActionEvent event) {} // 查询指定课程考务 void btnCheckStuTestClick(ActionEvent event) {} // 查询指定学生考务
void btnClearTestClick(ActionEvent event) {} // 便于清空已查询到的课表考务信息
void lblGradeAdminClick(MouseEvent event) {} // 课程班成绩查询模块切入函数
void btnGradePerformanceClick(ActionEvent event) {} // 初始化课程名称 comboBox
内部类：public class tbGradeForm {} // 成绩 tableView 的数据模型
void btnGradePerformanceCheckClick(ActionEvent event) {} // 确认查询课程班成绩
void btnGradePerformanceDeleteClick(ActionEvent event) {} // 取消查询课程班成绩
void lblCheckGradeClick(MouseEvent event) {} // 学生成绩查询功能函数
void btnStudGradeClick(ActionEvent event) {} // 初始化学生编号 comboBox 函数
void btnStudGradeCheckClick(ActionEvent event) {} // 确认查询学生成绩函数
void btnStudGradeDeleteClick(ActionEvent event) {} // 取消查询学生成绩函数
double calculateGPA(double grade) {} // 4.8 分制绩点计算函数

StudEducationalCtrl 类：

方法：

前 11 个方法函数同 AdminEducationalCtrl 类
void invisible(ActionEvent event) {} //错误界面显示函数
void lblChooseCourseClick(MouseEvent event) {} // 选课模块切入函数
void btnChooseCourseClick(ActionEvent event) {} // 选课函数
void lblMyCourseClick(MouseEvent event) {} // 我的课表模块切入函数
内部类：public class tbCourseForm {} // 课表 tableView 的数据模型
void btnMyCourseClick(ActionEvent event) {} // 点击查看我的课表
void lblCheckCourseClick(MouseEvent event) {} // 查询课程模块切入函数
void btnCheckCourseClick(ActionEvent e) {} //点击按照课程编号或课程名称去查询
void btnClearCheckCourseClick(ActionEvent e) {} // 便于清空已查询到的课程信息
void lblCheckTestClick(MouseEvent event) {} // 查询考务模块切入函数
void btnCheckCourseTestClick(ActionEvent event) {} // 点击查询课程考务信息
void btnClearTestClick(ActionEvent event) {} // 便于清空已查询到的考务信息
void btnCheckStuTestClick(ActionEvent event) {} // 点击查询自己的考务信息
void lblCheckGradeClick(MouseEvent event) {} // 查询成绩信息模块切入函数
double calculateGPA(double grade) {} // 4.8 分制绩点计算函数
内部类：public class tbGradeForm {} // 成绩 tableView 的数据模型
void btnMyGradeClick(ActionEvent event) {} // 点击查看我的课程成绩

TeachEducationalCtrl 类：

方法：

前 11 个方法函数同 AdminEducationalCtrl 类
void lblCheckCourseClick(MouseEvent event) {} // 查询课程信息模块
void btnClearCourseClick(ActionEvent event) {} // 便于清空已查询到的课程信息
void btnCheckCourseClick(ActionEvent e) {} //点击按照课程编号或课程名称去查询
void lblMyCourseClick(MouseEvent event) {} // 我的课表信息模块切入函数
内部类：public class tbGradeForm {} // 成绩 tableView 的数据模型
void btnMyCourseClick(ActionEvent event) {} // 点击查看我的课表
void lblCheckTestClick(MouseEvent event) {}// 查询考务模块切入函数
void btnClearTestClick(ActionEvent event) {} // 便于清空已查询到的考务信息
void btnCheckCourseTestClick(ActionEvent)//点击按照课程编号或课程名称去查询
void btnCheckStuTestClick(ActionEvent event) {}// 查看自己的课程班考务
void lblUpdateGradeClick(MouseEvent event) {} // 录入成绩模块
void btnStudGradeClick(ActionEvent e) {} // 初始化下拉框里学生编号、学生姓名
void cbStudIDClick(ActionEvent e) {} // 初始化下拉框里课程名称、绑定学生姓名
void cbStudNameClick(ActionEvent event) {}// 学生编号绑定学生姓名
void btnGradeCheckClick(ActionEvent event) {} // 录入学生成绩

void btnGradeDeleteClick(ActionEvent event) {} // 取消录入学生成绩
void btnCourseGradeClick(ActionEvent event) {} // 录入课程成绩
void cbCourseName1Click(ActionEvent e) {} //初始化下拉框里学生编号、学生姓名
void cbStudIDClick1(ActionEvent event) {}// 学生编号绑定学生姓名
void cbStudNameClick1(ActionEvent event) {} // 学生编号绑定学生姓名
内部类：public class tbGradeForm {} // 成绩 tableView 的数据模型
void btnGradeCheckClick1(ActionEvent event) {} // 确认录入课程成绩
void btnGradeDeleteClick1(ActionEvent event) {}// 取消录入
void lblCheckGradeClick(MouseEvent event) {} // 查询成绩信息模块
void btnGradePerformanceClick(ActionEvent e) {}// 初始化下拉框里教的课程名称
void btnGradePerformanceCheckClick(ActionEvent) {} //确认查询自己的课程班成绩
void btnGradePerformanceDeleteClick(ActionEvent event) {} // 取消

AdminInformationCtrl 类：

方法：

void btnAddStuClick(ActionEvent event)//添加学生按钮事件响应
void btnDeleteStuClick(ActionEvent event)//删除学生事件响应
btnModStuInfoClick(ActionEvent event) // 编辑学生信息事件响应
private void initialize() // 初始化，设置时间和表格属性
private void initModPanel() // 初始化编辑界面各文本框为当前学生信息
public void setClient(Client c) //初始化 Client 变量，初始化查询表格信息
void btnCheckStuTestClick(ActionEvent event)// 查询学生按钮事件响应

UserSpaceViewCtrl 类：实现 Initializable 接口

内部类：public class chatMessageListofsomeone // 保存 ID 和消息记录

方法：

void ctchat(MouseEvent event) // 显示在线聊天界面，查询并动态显示聊天列表，设定聊天列表双击事件
public void initchatPane(int i) // 初始化聊天框界面，新建线程监听聊天消息
void btnQuerychatMember(ActionEvent event) // 查询特定人员并动态显示聊天列表，设定聊天列表双击事件
public void restoreMessage(Message mes) // 从服务端消息保存聊天信息
public synchronized void putchatMessage(String name,String str,int leftOrRight) // 将聊天消息显示在聊天框上
void btnSendChatMessage(ActionEvent event) // 发送聊天消息按钮事件响应

LoginViewCtrl 类：

Client client;//客户类实体
int flag = 0;//判断帐号是否符合规范
private TextField username;//登录帐号
private PasswordField password;//登陆密码
private Label wronglabel;//提示错误信息
private Button stu;//学生登录按钮
private Button teach;//教师登录按钮
private Button admin;//管理员登录按钮

方法：

void valuechange(KeyEvent event) {}//监听用户 ID 框文本，实现按钮亮起
void stuClick(ActionEvent event) throws IOException {}//学生登录点击按钮跳转
void teachClick(ActionEvent event) {}//教师登陆跳转
void adminClick(ActionEvent event) {}//管理员登陆跳转
以及 client 的 set 函数

LoginInViewCtrl 类：

Client client;//用户实体
String ID;//用户 ID

int left=1;//窗口切换顺序
int middle=3;
int right=2;
TranslateTransition t=new TranslateTransition();//动画初始化
Pane pane21=new Pane();//动画根节点
DisplacementMap dismap1=new DisplacementMap();//位移动画实体
DisplacementMap dismap2=new DisplacementMap();
DisplacementMap dismap3=new DisplacementMap();
private AnchorPane apane;
private GridPane grid2;//最前面的容器
private StackPane pane12;
private HBox pane1;//pane123 为放切换图片的容器
private HBox pane2;//在最前面
private HBox pane3;
private Button bt1;//左滑图片按钮
private Button bt2;//右滑图片按钮
private GridPane grid;
private AnchorPane pane;
private Button user;//用户模块按钮
private Button jwc;//教务模块按钮
private Button bank;//银行模块按钮
private Button shop;//商店模块按钮
private Button stud;//学生模块按钮
private Button library;//图书馆模块按钮

方法：

void bankdo(ActionEvent event) throws Exception {}//点击银行按钮跳转
void jwcdo(ActionEvent event) throws IOException {}//点击教务按钮跳转
void librarydo(ActionEvent event) throws IOException {}//点击图书馆按钮跳转
void shopdo(ActionEvent event) throws IOException {}//点击商店页面跳转
void studo(ActionEvent event) throws IOException {}//点击学生管理按钮跳转
void userdo(ActionEvent event) throws IOException {}//点击用户按钮跳转
void bt1do(ActionEvent event) {}//图片左滑
void bt2do(ActionEvent event) {}//图片右滑
void btview(MouseEvent event) {}//鼠标进入时显示按钮 1 和按钮 2
void btnoview(MouseEvent event) {}//鼠标离开时隐藏按钮 1 和按钮 2
public void initialize(URL location, ResourceBundle resources) {}//界面初始化
public void button1do() {}//左滑函数具体实现
public void button2do() {}//右滑函数具体实现
以及 client 和 ID 的 set 函数

UserSpaceViewCtrl 类：

Client client;//用户实体
String ID;//用户 ID
String vcodetext;//验证码文本
String Path;//验证码路径
String headpath;//头像路径
String newheadpath;//更新后头像路径
String headencoded;//头像图片编码
int flag=0;//是否修改头像，默认为 0 无修改
private Label logoutlabel;//登出标签
private Circle touxiang;//用户头像显示圆
private Label user;
private Label flash;

private Text selfinfo;
private Label selfcenterlabel;//切换模块标签*3
private Label changepwlabel;
private Label changeinfolabel;
private Label time;//显示当前时间
private Label id;
private GridPane changeinfo;//修改个人信息模块容器
private TextArea changeself;//修改个人信息文本输入框
private TextField changename;
private TextArea changesig;
private Pane picture1;//用户背景图片 1-5
private Pane picture2;
private Pane picture3;
private Pane picture4;
private Pane picture5;
private GridPane selfcenter;//个人主页模块容器
private Label name;
private Label signature;
private GridPane changepw;//修改密码模块容器
private TextField vcode;//修改密码的文本输入框
private TextField revealpw;
private PasswordField newpw;
private PasswordField cfmnewpw;
private PasswordField currentpw;
private GridPane gridpane;//初始化模块容器
private Label menu;
private Label getname;//显示用户信息的标签
private Label getsig;
private Label label1;//右侧弹窗模块对应标签
private Label label2;
private Label label3;
private Label label4;
private Label label5;
private Label wrong1;//wrong 和 right 都是提示信息
private Label wrong2;
private Label wrong21;
private Label wrong3;
private Pane right1;
private Pane right2;
private Rectangle vcodepic;//验证码矩形图片
private Rectangle eye;//密码可视化
private Rectangle headpic;//用户上传头像

方法：

void cancel(ActionEvent event) {}//进入时选择退出界面跳转
void changeveal(MouseEvent event) {}//鼠标进入时使密码可见
void changeconceal(MouseEvent event) {}//鼠标离开时使密码不可见
void choosehead(ActionEvent event) throws IOException {}//用户选择头像上传头像并在当前界面显示
private static byte[] InputStreamToByte(InputStream is) throws IOException {}//InputStream 流转 byte[]数组
private static void configureFileChooser(final FileChooser fileChooser) {}//图片过滤器，文件打开时可选择，包括 all、jpg、png
void ctchangeinfo(MouseEvent event) {}//切换到修改个人信息页面
void ctchangepw(MouseEvent event) throws Exception {}//切换到修改密码界面

void ctselfcenter(MouseEvent event) throws IOException, Exception {}//切换到个人中心页面
void movein(MouseEvent event) {}//鼠标离开时隐藏模块切换菜单
void moveout(MouseEvent event) {}//鼠标移入时弹出模块切换菜单
void logout(MouseEvent event) throws Exception {}//退出登录，回到登陆界面切换
void module1(MouseEvent event) throws IOException {}//切换到学生管理模块
void module2(MouseEvent event) throws IOException {}//切换到虚拟教务模块
void module3(MouseEvent event) throws IOException {}//切换到虚拟图书馆模块
void module4(MouseEvent event) throws IOException {}//切换到虚拟商店模块
void module5(MouseEvent event) throws IOException {}//切换到虚拟银行模块
void changepw(ActionEvent event) throws Exception {}//修改密码点击按钮事件
void changeinfo(ActionEvent event) {}//修改个人信息按钮点击事件
void invisible(ActionEvent event) {}//个人信息修改完毕通知信息关闭按钮点击事件
void ctpic1(MouseEvent event) {}//修改个人主页背景为图片 1
void ctpic2/ctpic3/ctpic4/ctpic5{}同理
public void initialize(URL location, ResourceBundle resources) {}//页面初始化
void flashvcode(MouseEvent event) throws Exception {}//刷新验证码
public void getvcode() throws Exception {}//选择随机数，根据随机数读取相应图片
public void returntologin() {}//返回登陆界面函数封装
以及 client 和 ID 的 set 函数

BookCtrl 类：

方法：

void showMyBorrow(MouseEvent event) // 展示个人借阅记录响应函数
void showBorrow(MouseEvent event) // 展示面向用户的图书信息响应函数
void showInfo(MouseEvent event) // 展示面向管理员的图书信息响应函数
void addBook(MouseEvent event) // 向图书馆添加图书响应函数
void queryBook(MouseEvent event) // 管理员查询图书信息响应函数
void borrowQueryBook(MouseEvent event) // 用户查询图书响应函数
void infoQueryBook(MouseEvent event) // 查看个人借阅信息功能

BankViewCrtl 类：

方法：
depositLabelClicked ()
功能：存款 Label 点击事件
传入参数：无

方法：
loadLabelClicked ()
功能：贷款 Label 点击事件
传入参数：无

方法：
repayLabelCliced ()
功能：还款 Label 点击事件
传入参数：无

方法：
selfInfoLabelClicked ()
功能：个人信息 Label 点击事件
传入参数：无

方法：
deposit_amountCancelButtonClicked ()
功能：存款信息 ButtonCancel 点击事件
传入参数：无

方法：

deposit_amountOKButtonClicked ()
功能：存款信息 ButtonOK 点击事件
传入参数：无

方法：
load_amountCancelButtonClicked()
功能：贷款信息 ButtonCancel 点击事件
传入参数：无

方法：
load_amountOKButtonClicked ()
功能：贷款信息 ButtonOK 点击事件
传入参数：无

方法：
repay_amountCancelButtonClicked ()
功能：还款信息 ButtonCancel 点击事件
传入参数：无

方法：
repay_amountOKButtonClicked ()
功能：还款信息 ButtonOK 点击事件
传入参数：无

方法：
transactioInfoQueryLabelClicked ()
功能：查询流水信息 ButtonCancel 点击事件
传入参数：无

方法：
transactioInfoQueryLabelClicked ()
功能：流水信息 Label 点击事件
传入参数：无

方法：
transactionInfoQueryIDCancelButtonClicked ()
功能：查询流水信息 ButtonCancel 点击事件
传入参数：无

方法：
transactionInfoQueryIDOkButtonClicked()
功能：查询流水信息 ButtonOK 点击事件
传入参数：无

方法：
userInfoDeletedLabelClicked ()
功能：删除信息 Label 点击事件
传入参数：无

方法：
userInfoDeleteCancelClicked ()
功能：删除信息 ButtonCancel 点击事件
传入参数：无

方法：
userInfoDeleteOkClicked ()
功能：删除信息 ButtonOK 点击事件
传入参数：无

ShopAdminCtrl 类：

方法：
void PurchingButton(MouseEvent event) // 显示管理员进退货管理页面。

void RecordButton(MouseEvent event) // 显示管理员查询任意顾客任意时间订单页面。
void changeGoods() // 管理员对商店中货物进行增添或下架。
void checkOrder(ActionEvent event) // 管理员查询任意人的购物订单。

ShopCustomerCtrl 类：

方法：

void PurchingButton(MouseEvent event) // 显示顾客购买商品页面。
void RecordButton(MouseEvent event) // 显示顾客查询自己订单记录页面。
void AccountButton(MouseEvent event) // 显示顾客查询个人账户信息页面。
void BuyXXX() // 顾客点击加一按钮后向购物车中加入一件对应商品（XXX 如 Cola,Bis.etc）
void ReturnXXX(ActionEvent event) // 顾客点击减一按钮后再购物车中减少一件相应商品（XXX 如 Cola,Bis,etc.）
void sendOrder() // 顾客选择好商品后向服务器发送生成订单请求。
void checkOrder(ActionEvent event) // 顾客查询个人订单记录。

(4) vo 包中的程序设计

Class 类：

public ArrayList<String> observed = new ArrayList<String>(); //排的课
private int grade; //年级
private String name; //班级名称
private int sum; //班级人数
private Sequence sq; //班级空闲时间
public Class() {}//缺省构造函数
public Class(int grade, String name, int sum) {}// 构造函数
以及所有变量的 setter、getter 函数

Classroom 类：

private int id;//教室编号
private String name;//教室名
private int capacity;//容量
private Sequence sq;//教室空闲时间
public Classroom() {}//缺省构造函数
public Classroom(int id, String name, int capacity) {}// 构造函数
以及所有变量的 setter、getter 函数

Course 类：

private int id;//课程编号
private String name;//课程名称
private int timesPerWeek;//周次
private String preCourse;//先修课程
private String classification;
private double credit;//学分
private String roomId;//上课地点
private String courseDate;//上课时间
private String teacher;//任课老师
private String testAddress;//考试地址
private String testDate;//考试时间
private int choosed;//选课人数
public Course() {} // 缺省构造函数
public Course(int id, String name, int timesWeek) {} // 构造函数
以及所有变量的 setter、getter 函数

Grade 类：

private ArrayList<Integer> id=new ArrayList<Integer>();//编号
private ArrayList<String> studentName= new ArrayList<String>();//学生姓名
private ArrayList<String> courseName= new ArrayList<String>();//课程名称
private ArrayList<String> teacherName= new ArrayList<String>();//老师姓名
private ArrayList<Double> grade= new ArrayList<Double>();//成绩分数
public Grade() {} // 缺省构造函数
public Grade(int id, String student,String course,String teacher,double g) {}// 构造函数
以及所有变量的 setter、getter 函数

Sequence 类：

private LinkedList<String> list = new LinkedList<>();
public Sequence() {} // 缺省构造函数
以及 list 变量的 setter、getter 函数

Student 类：

private int id;//编号
private String studID;// 一卡通号
private String sno;// 学号
private String name;// 姓名
private String sex;// 性别
private String grade;// 年级
private String className;// 班级
private String contact;// 在线状态
private Sequence sq;// 空闲时间
public ArrayList<String> choosedCourse=new ArrayList<String>();// 选的课
以及四种构造函数和所有变量的 setter、getter 函数

Teacher 类：

public String[] observed = new String[20];//课程安排
public ArrayList<String> teachCourse = new ArrayList<String>();//教的课程
private int id;//教师工号
private String name;//教师姓名
private String busyTime;//教师忙碌时间（非空闲）
private Sequence sq;//空闲时间
private String inCharge;//管理的班级
以及三种构造函数和所有变量的 setter、getter 函数

Vcode 类：

private int AID;//验证码序号
private String vcodetext;//验证码对应文本
private Blob vcodepic;//验证码二进制文件
以及对应 set、get 函数

User 类：

private String id;
private String password;
private String name;//昵称
private String signature;//签名
private int background;//名片背景图片设置
private String selfinfo;//个人简介
private Blob headpic;
private int status;//区分管理员、老师、学生，0-管理员，1-老师，2-学生
以及对应 set、get 函数

Book 类：

private SimpleStringProperty bookId;//索书号
private SimpleStringProperty bookName;//书名

private SimpleStringProperty state;// 书籍状态
private SimpleStringProperty price;// 价格
private SimpleStringProperty writer;//作者
private SimpleStringProperty address;//书籍存放位置
private SimpleStringProperty borrower;//借阅学生编号
private SimpleStringProperty borrowDate;//借阅时间
private SimpleStringProperty returnDate;//归还时间
public Book(String bookId, String bookName, String state, String price,
String writer, String address, String borrower, String borrowDate,
String returnDate) // 构造函数构造 Book 类对象
以及所有变量的 set、get 函数

Bank 类：

private String bID;
private double bDeposit;
private double bLoad;
private double bDebit;
private int bState;
以及对应 set、get 函数

BankRecord 类：

private String originatorID;
private String recipientID;
private String transactionmode;
private String transactionamount;
private String transactiontime;
以及对应 set、get 函数

Server 端:

Server 类：

方法：

public void start() // start 函数：调用 ServerConnection 类构造函数以及
ConnectionFrame 类构造函数
public ServerConnectionService getConnectionService() // 返回类对象 connection

(1) dao 包中的程序设计

ClassDAO 类：

方法：

public Class query(int ID) throws Exception {}// 按班级 id 查询
public Class queryName(String name) throws Exception {} // 按班级名称查询
public ArrayList<String> totalClass() {} // 显示所有班级名称
public void updateAdd(String newString, String field, int id) {}// 增加班级某字段内容
public void update(String newString,String field, int id) {}// 修改班级某字段信息

ClassroomDAO 类：

方法：

public Classroom query(int ID) throws Exception {}// 按教室 id 查询

CourseDAO 类：

方法：

public Course query(int ID) throws Exception {}// 按课程 id 查询
public Course queryName(String name) throws Exception {}// 按课程名称查询
public void add(int id,String name,int timesPerWeek,String preCourse,String
classification, double credit) {}// 增加课程信息
public void delete(int id){}// 删除课程信息

public void update(int id,String name,int timesPerWeek,String preCourse,String classification,double credit,String address,String date,String teacher) {}//更新课程基本信息

public void update(int id,String name,String classification,double credit,String testAddress,String testDate,String teacher) {}//修改课程考务信息

public void updateScheduling(String name,String name1,String date,String teach) {}//更新课程排课信息

public Course updateTest(int id,String testDate,String testAddress) {}// 增加考务信息

GradeDAO 类：

方法：

public Grade queryStudent(String name) throws Exception {}//按照学生名查成绩

public Grade queryStudID(String ID) throws Exception {}//按照学生 ID 查成绩

public Grade queryCourse(String name) {}//按照课程名查成绩

public void updateGrade(String studentName,String courseName,double grade){}//更新成绩

public void addGrade(int id,String studentName,String courseName,String teacherName,double grade) {}//录入成绩

public ArrayList<String> queryTeachCourse(String name) {}//查询指定教师所教课程

public ArrayList<String> queryTeachStudent(String name) {}//查询指定教师所教学生

StudentDAO 类：

方法：

public Student query(int id){}//按学生 id 查询信息

public Student queryStudID(String id){}//按一卡通查询学生信息

public Student queryName(String name) {}//按学生姓名查询学生信息

public int chooseCourse(String studentID, String courseName) {}//选课函数

public ArrayList<String> queryCourse(String name) {}//查询学生是否选择某课程

public boolean deleteCourse(String studentID, String courseName) {}//退课函数

TeacherDAO 类：

方法：

public Teacher query(int ID) {}//按 ID 查询教师信息

public Teacher queryName(String name) {}//按教师名称查询教师信息

public void update(String newString,String field,int id) {}//更新教师信息

public void updateAdd(String newString,String field,int id) {}//增加教师某字段信息

public Teacher queryTeachID(int ID) {}//按教师工号查询教师信息

public ArrayList<String> totalTeacher() {}//查询所有教师

InfomationDAO 类：

方法：

public ArrayList<Student> quaryALL() // 从数据库中查询所有学生

public ArrayList<Student> quaryInClass(String className) // 根据班级查询学生

public ArrayList<Student> quaryInID(String id) // 根据 ID 查询学生

public ArrayList<Student> quaryInIDAndClass(String id,String className) //根据班级和 ID 查询学生

public boolean modStu(Student stu)// 更新学生信息

public void addStu(Student stu) // 添加学生信息

public void deleteStu(String Uid) // 删除学生信息

public Teacher quaryTeacher(String Uid)// 根据 ID 查询老师

BookDAO 类：

private ArrayList<Book> bookList; // 存放图书信息

private ArrayList<Book> borrowList; // 存放图书借阅记录信息

方法：

public void addBook(Book book) // 将 book 信息添加进 bookinfo 表

public void deleteBook(String id) // 将 book 信息从 bookinfo 表删除
public void upDateBookInfo(String id, String name, String state, String price,String writer, String address,String borrower,String borrowDate,String returnDate) // 更新表 bookinfo 中 book 信息
public void upDateBorrowInfo(String id,String borrower, String name,String state,String borrowDate,String returnDate) // 更新表 tb_borrow 中 book 信息
public Book query(String ID) // 根据书号取得对应图书信息
public Book queryName(String name) // 根据书名取得对应图书信息
public ArrayList<Book> AllListBook () // 将 bookinfo 表中所有信息传入 bookList 中
public ArrayList<Book> AllListBorrow (String borrowerID) // 将 tb_borrow 表中所有信息传入 list 中
public void addBorrowBook(String id,String borrower, String name, String state,String borrowDate,String returnDate) // 向 tb_borrow 表中添加借书信息
public Book queryBorrow(String name, String borrower) // 查询个人借阅信息

OrderDAO 类：

方法：

public void createOrder(String orderID,String buyerID,String goodsName,int goodsNum,double price,String date) // 向数据库中添加一条订单记录。
public Order queryOrder(String userID,String time)//从数据库中查询对应时间与用户 id 的订单记录。
public int queryInventory(String goodsName) // 从数据库中查询商品库存。
public void changeGoods(String goodsName,int goodsNum) // 更改数据库中对应商品的库存。

(2) net 包中的程序设计

ReceiveListener 类：

方法：

public void run() // 重载，监听并接受输入流中的消息，根据消息类型调用相应的处理方法

ServerConnectionService 类：

方法：

public ServerConnectionService(Server s) // 构造函数，初始化线程列表
public void netConnection() // ServerSocket 是等待客户端的请求，一旦获得一个连接请求，就创建一个 Socket 示例来与客户端进行通信，利用对象流传输 Message 类

SocketThread 类：实现 Runnable 接口

方法：

public SocketThread(Server s,int rTid,InputStream rinputStream,OutputStream routStream,ObjectInputStream rin,ObjectOutputStream rout) // 构造方法，初始化变量和输入输出流
public void run() // 重载，新建监听线程监听来自客户端的信息
public synchronized Message receive() // 监听线程监听到客户端发送消息后，同步接受消息
public synchronized void send(Message mes) // 同步发送消息
public void close() // 中止该线程
public void findClientThread(Message mes) // 找到聊天消息的接收方客户端
public void sendChatMes(Message mes) // 发送聊天信息

(3) srv 包中的程序设计

EducationalAdminService 类：

方法：

public void scheduling(Message mes) {}//排课函数

public boolean Arranging() {}//排课函数
public LinkedList<String> randList(LinkedList<String> list, int timesPerWeek) {}//从LinkedList<String>中随机取出 timesPerWeek 个元素组成的 LinkedList<String>
public boolean order(Teacher te, Class cl, Course co) {}//排课函数
public void showClass(Message mes) {}//显示班级课程
public void showTeacher(Message mes) {}//显示老师课程
public void checkCourse(Message mes) {}//查询课程函数
public void deleteCourse(Message mes) {}//删除课程函数
public void updateCourse(Message mes) {}//修改课程信息
public void addCourse(Message mes) {}//添加课程信息
public void showTest(Message mes) {}// 查询考务函数
public void addTest(Message mes) {} //增加考务信息
public void deleteTest(Message mes) {}//删除考务信息
public void showStudentGrade(Message mes) {}//显示学生成绩函数
public void showCourseGrade(Message mes) {}//显示课程班成绩函数
public void gradePerformance(Message mes) {}//显示成绩表里的班级
public void showStudent(Message mes) {}//显示成绩表里的学生
public void updateTest(Message mes) {}//更新考务信息
public void showTotalTeacher(Message mes) {}//显示所有老师函数
public void showTotalGrade(Message mes) {}//显示所有班级函数

EducationalStudService 类：

方法：

public void chooseCourse(Message mes) {}//选课函数
public void showCourse(Message mes) {}//显示学生选的课程
public void matchCourse(Message mes) {}//显示选某门课程的学生
public void showCourseForm(Message mes) {}//显示课表函数
public void showCourseList(Message mes) {}//显示所有学生课程
public void deleteCourse(Message mes) {}//退课函数

EducationalTeachService 类：

方法：

public void updateGrade(Message mes) {}//录入成绩函数
public void showCourse(Message mes) {}//显示教的课程
public void showStudent(Message mes) {}//显示教的学生

InformationService 类：

方法：

public InformationService(Server s,SocketThread so) // 初始化变量
public void quaryStudent(Message mes) // 查询学生信息，判断是根据 ID 还是 class 查询
public void setListMessage(Message mes,ArrayList<Student> stuList) // 将查询到的学生列表转化为消息
public void modStudent(Message mes) // 修改学生信息
public void addStudent(Message mes) // 增加学生
public void deleteStudent(Message mes) // 删除学生
public void quaryTeacher(Message mes) // 查询老师
public void quaryThisStudent(Message mes) // 查询当前学生
public void queryChatMember(Message mes) // 查询聊天列表
public void checkChatMember(Message mes) // 检查是否在线
public void logout(Message mes) // 退出客户端登录，更新在线状态

BookService 类：

方法:

public void addBook(Message mes) // 读取 mes 信息向数据库中添加图书信息

public void updateBook(Message mes) // 向数据库中更新图书信息
public void deleteBook(Message mes) // 向数据库中删除图书信息
public void queryBook(Message mes) // 查询图书信息，将得到的信息打包为 message 类进行通信
public void queryBorrowBook(Message mes) // 查询图书借阅记录，将得到的信息打包为 message 类进行通信
public void BorrowBook(Message mes) // 借书，在数据库中更新相应信息
public void queryMyBorrow(Message mes) // 查询用户借阅记录，将得到的信息打包为 message 类进行通信
public void returnBook(Message mes) // 还书，在数据库更新相应信息
public void renewBook(Message mes) // 续借，在数据库更新相应信息
public void showBook(Message mes) // 将数据库中所有图书信息打包为 message 类进行通信
public void showBorrow(Message mes) // 将数据库中所有指定用户的图书借阅记录打包为 message 类进行通信

BankService 类：

方法：
answerSelfInfoOK(Message mes)
功能：后端与数据库交互获得银行用户信息
传入参数：Socket 传入后端的 Message

方法：
answerDeposit_amountOK (Message mes)
功能：后端与数据库交互存入银行用户新的存款信息
传入参数：Socket 传入后端的 Message

方法：
answerLoad_amountOK (Message mes)
功能：后端与数据库交互存入银行用户新的贷款信息
传入参数：Socket 传入后端的 Message

方法：
answerRepay_amountOK (Message mes)
功能：后端与数据库交互存入银行用户新的还款信息
传入参数：Socket 传入后端的 Message

方法：
adminTransactionInfoOK (Message mes)
功能：后端与数据库交互获得银行所有用户新的流水信息
传入参数：Socket 传入后端的 Message

方法：
adminQueryTransactionInfoOK (Message mes)
功能：后端与数据库交互获得银行某个用户新的流水信息
传入参数：Socket 传入后端的 Message

方法：
adminUserInfoOK (Message mes)
功能：后端与数据库交互获得银行全部用户信息
传入参数：Socket 传入后端的 Message

方法：
adminQueryUserInfoOK (Message mes)
功能：后端与数据库交互获得银行某个用户信息
传入参数：Socket 传入后端的 Message

方法：
adminDeleteOk (Message mes)
功能：后端与数据库交互挂失银行某个用户信息
传入参数：Socket 传入后端的 Message

方法：

adminInvokeOk (Message mes)

功能：后端与数据库交互激活银行某个用户信息

传入参数：Socket 传入后端的 Message

OrderService 类：

方法：

public void createOrder(Message mes) // 向数据库中增加来自消息 Message 的订单。

public void queryOrder(Message mes) // 从数据库中查询来自消息 Message 要求查询的订单。

public void queryInvento() // 从数据库中查询所需全部库存。

public void changeGoods(Message mes) // 在数据库中更改 Message 中要求更改商品的库存。

public void getAccount(Message mes) // 从数据库里查询 Message 中要求账户的余额。

(4) util 包中的程序设计

DbUtil 类：

方法：

public DbUtil() {}//DbUtil 构造函数

public Connection getCon() {}//获取数据库连接

public void closeCon(Connection con) {}//关闭数据库连接

public boolean update(String sql) {}//更新

public boolean add(String sql) {}//添加

public boolean delete(String sql){}//删除

public ResultSet query(String sql) {}//查询

public int totalCount(String sql) throws SQLException{}//返回数据库的表的总行数

(5) vo 包中的程序设计

同 Client 端

四．项目关键技术

我们这款虚拟校园系统软件项目使用基于 C/S 架构的 Windows 应用程序，服务器端与客户端均采用 Java 作为开发语言，使用 MySQL 管理数据库数据，使数据的准确性与安全性得到了很大的提高，且在用户的并行操作与用户管理方面也有了极大地改善。服务器是后台支持程序，按需进行 GUI 界面设计；客户端程序面向实际用户，设计友好的界面；实现方面，主要采用 Socket 网络编程、多线程、输入输出流等技术来实现 Swing 界面编程。系统的性质为管理软件，因而数据库的设计与操纵是系统设计的核心。这里从两方面分析技术上的可行性。

首先，对于 java 编程部分，基于团队成员对 C++编程语言的一定掌握，学习与使用 java 编程语言不是难事，基于团队成员 QT、MFC 等图形用户界面应用程序开发框架的一定掌握，因此在程序系统设计中，对 java 图形 GUI 编程方面的问题都能迎刃而解。

其次，对于数据库的有关知识，团队成员运用一定的时间对相关书籍、资料进行阅读以及上机学习实验，以基本掌握运用 MySQL 数据库技术，具备一定的系统分析与设计能力，熟悉数据库的设计与操纵；因而该系统的实现在技术上是可行的。

最后，在 UI 设计方面，从 swing 和 awt 转为 JavaFX 可视化设计，scenebuilder 使用是 UI 设计很好的工具，再结合 css，fxml 文件整体美化。

五．项目经验及存在问题

（一）经验：

1. 学习使用 JavaFX 框架；
2. javafx 的熟练使用和可视化编程 scenebuilder 的使用；
3. 连接数据库操作时，Blob 的消息类传递需要序列化，可用编码解码的方法解决；
4. 实现管理系统；
5. 多设计方法可以有效地将工程的架构变的更加易于扩展；
6. 在操作系统中学习关于线程的操作问题。

（二）存在问题：

1. 各模块没有分离，给团队编程带来很大问题；
2. Blob 类型只能接收 65KB 以下的图片，上传头像时需要注意，如换成 MediumBlob 或 LongBlob 会更好；
3. 功能模块还有待完善；
4. 可以将界面设计的更加人性化；
5. 学习 java 及 javaFXd 的使用中有遇到理解与操作上问题；小组任务中有需要与其余成员任务交互的需求，协同沟通中遇到对方已完成任务的使用的问题。

（三）团队沟通和工作方式：

1. 小组成员做各自的分工板块，最后一起整合；
2. 先统一学习 java 和 socket 和数据库相关内容，先用登录模块一起学习，再分开做彼此的模块；
3. 线上与线下讨论结合，通过 gitee、qq 文件传输项目文件；
4. 团队主要通过 QQ 群聊进行线上沟通，关于新软件的使用方面线下进行了交流学习。工作方式上将各自任务的完成与更新代码发布在 Gitee 上，组长最终将所有功能进行整合。

（四）技术理解：

1. 多线程处理一定要注意同步问题，非必要不使用自旋锁尽量，防止 while 长时间循环；
2. UI 的许多效果是靠监听来做的，包括内在逻辑的安排，需要手动写；
3. 初步掌握了 Java 语言，学习了基于 Javafx 的 GUI 开发，MySQL 数据库操作以及 socket 相关知识；
4. MySQL 极大的简化了我们存储数据的逻辑；
5. 为增加软件页面的整体美观性，降低设计难度，采用了 javaFx 进行界面设计；利用 java 网络编程功能实现客户端与服务端通过网络进行沟通，实现多个客户端对同一服务端的操作。

（五）工作方式的解决：

1. 部分成员先构建好公共类以及初步 UI，接下来分工进行各个功能模块的实现；
2. 由于 Gitee 的限制性，同时利用 QQ 群文件辅助传递工程文件；
3. 先靠网上教程入门，再自己尝试学习，遇到问题时再去搜索，经常一试一下午；
4. 组长承担责任，熬夜赶工整合模块。

教师评语【优　良　中　及格　不及格 】