学生选课系统 — 设计报告书

学号：

姓名：

2022年07月03日

目 录

[1 简介 3](#_Toc107753663)

[1.1 作品创意/项目背景 3](#_Toc107753664)

[1.2 项目实施计划 3](#_Toc107753665)

[2 总体设计 4](#_Toc107753666)

[2.1 系统功能 4](#_Toc107753667)

[2.1.1 功能概述 4](#_Toc107753668)

[2.1.2 功能说明 4](#_Toc107753669)

[2.2 系统软硬件平台 5](#_Toc107753670)

[2.2.1 系统开发平台（含开源/第三方工具） 5](#_Toc107753671)

[2.2.2 系统运行平台 5](#_Toc107753672)

[2.3 关键技术和作品特色 5](#_Toc107753673)

[3 详细设计 6](#_Toc107753674)

[3.1 系统结构设计 6](#_Toc107753675)

[3.1.1 技术架构 6](#_Toc107753676)

[3.1.2 关键功能/算法设计 6](#_Toc107753677)

[3.2 数据结构设计 13](#_Toc107753678)

[3.2.1 存储数据 13](#_Toc107753679)

[3.2.2 接口（模块接口、系统间接口） 15](#_Toc107753680)

[3.2.3 关键数据结构 15](#_Toc107753681)

[3.3 系统界面设计 18](#_Toc107753682)

[3.3.1 界面设计风格 18](#_Toc107753683)

[3.3.2 主要功能页面 18](#_Toc107753684)

[4 系统安装及使用说明 30](#_Toc107753685)

[5 总结 31](#_Toc107753686)

[6 附录 32](#_Toc107753687)

[6.1 名词定义 32](#_Toc107753688)

[6.2 参考资料 32](#_Toc107753689)

[6.3 源代码清单 32](#_Toc107753690)

# 简介

## 作品创意/项目背景

随着我国高等教育的不断发展，每一所普通高校的在校大学生人数都已高达成千上万，与此同时，大学生自主选课的需求日益增长，传统的人工管理和纸质文档管理的选课管理方式，已无法适应。

本项目开发的学生选课系统旨在顺应高校智慧校园建设的历史潮流，满足师生相关需求，有效提高高校学生选课管理的工作效率和信息化水平。

学生选课系统（以下简称：本系统）具有以下功能：学生课程选择；教师课程开设、课程信息管理；管理员用户管理、课程与选课信息管理等。

本系统分为服务端和客户端两部分。其中，服务端包含Mysql和Redis数据库。Redis的原子性保证在高并发请求下不会出现“超选”的情况。客户端使用C语言实现Socket客户端，可保证软件的性能。

本系统使用Socket通信；采用自定义数据传输格式；数据请求包含签名，从而实现请求防伪造。

## 项目实施计划

2022年6月23日至2022年6月24日：Socket通信数据格式制定；Python端Socket服务器初步实现；Python编写Socket客户端测试通过。

2022年6月25日至2022年6月26日：C语言客户端编写，实现Socket请求、用户登录、自动登录、密码修改、异常处理等操作；针对GBK编码与UTF8编码不兼容的问题进行转换与适配。

2022年6月27日至2022年7月2日：实现学生角色、教师角色、管理员角色的必需功能。

2022年7月3日至2022年7月4日：完善程序，优化代码与内存回收，撰写项目说明书。

2022年7月4日至2022年7月5日（预计）：交付项目。

# 总体设计

## 系统功能

### 功能概述

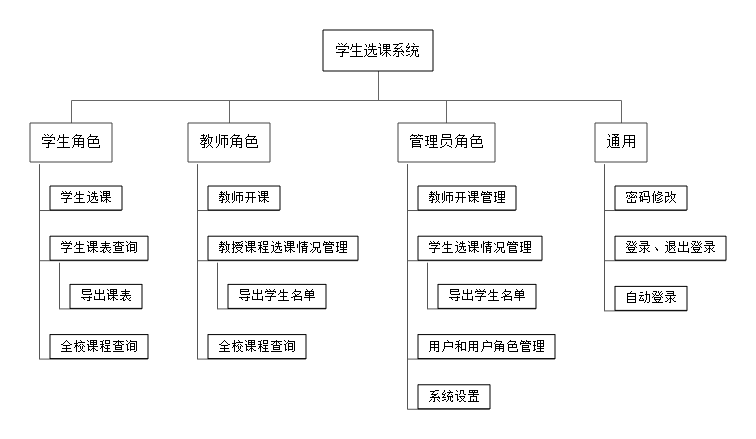


图1-1 系统功能框架图

系统功能框架图如图1-1所示。本系统的目标用户群体可分三类：学生、教师和管理员。不同用户群体分别对应不同的功能模块——学生：“线上选课”模块；教师：“课程管理”模块；管理员：“用户管理”模块。这些功能模块通过用户权限系统进行区分和限制。本系统在每学期开学时供选课所使用。对于学生用户而言，应保证在高并发状态下不出现“超选”等问题，同时，应尽可能提升服务器的相应速度，优化用户体验；对于教师和管理员而言，系统应提供对于大量数据处理的支持。

### 功能说明

对于学生用户，可使用本系统进行课程选择、查询、退选等操作。在选课结束后，可查看生成的学生课表，并可将其导出为txt文件中。

对于教师用户，可使用本系统开设、删除、编辑课程，并可实时查看选课情况，导出学生名单。

对于管理员用户，可使用本系统开设、删除、编辑课程，同时可对每个课程的选课情况进行查看、编辑、退选、批量导入名单等操作；可对系统用户进行新增、删除、修改、查询操作；还可对本系统的信息进行自定义设置——例如：设置某一学期的选课限制；修改当前系统学期；修改学校名称等。

所有用户，均可对自己的账户进行密码修改，同时，拥有浏览全校课程信息的权限。

为提升用户的使用体验，在用户登录后，系统可自动保存用户的登录状态，在此后的24小时内，可免密自动登录。为保证用户账户信息安全，登录状态文件（session）由通过用户硬盘序列号生成的唯一机器码AES加密后存储于本地，实现用户登录状态与机器绑定；同时，为保证请求内容的完整性和可靠性，客户端发出的请求将会附加签名。

## 系统软硬件平台

### 系统开发平台（含开源/第三方工具）

本系统后端使用Pycharm开发；C语言客户端使用Clion开发。

后端采用的技术栈：Python+MySQL+Redis。

前端采用纯C编写，引用cJSON开源库用于生成JSON请求和解析JSON响应数据；引用AES用于对本地session加密保存，确保数据安全；引用pcre2（正则表达式库）实现前端基础验证功能；引用hmac\_sha256库，用于生成请求签名，以确保请求防伪造。

### 系统运行平台

后端需在配置Python3.8版本以上的服务器中运行，同时，该服务器需安装MySQL5.7（或以上）版本和Redis缓存数据库。

前端需在Windows7或更高版本的操作系统上运行；要求内存不低于128M；硬盘容量高于10G。

## 关键技术和作品特色

本系统采用C/S技术架构，即包含服务端和客户端。在服务端与客户端之间通信过程中，采用TCP协议通信。为保证数据安全，数据传输规范未使用HTTP等通用规范，而是采用完全自定义的数据传输规范，同时，为保证数据不被更改，在每次请求时，都会包含签名。

本系统面临的场景是高并发下的学生选课请求，因此，在Python服务端和Mysql数据层之间，额外增加了Redis缓存数据库，从而确保高并发下不出现“超选”的情况。

本系统为保证用户登录状态数据安全，将本地Session文件以AES加密的密文形式保存，且密钥为相对唯一的机器码。

本系统在批量导入学生名单时，会自动去除重复的数据，在面临大数据量的导入场景时，本系统采用了哈希表的数据结构保证查重的低时间复杂度（最低可达到O（1）），以保证用户的使用体验。

# 详细设计

## 系统结构设计

### 技术架构

本系统采用C/S技术架构，具体系统架构设计如图3-1所示。

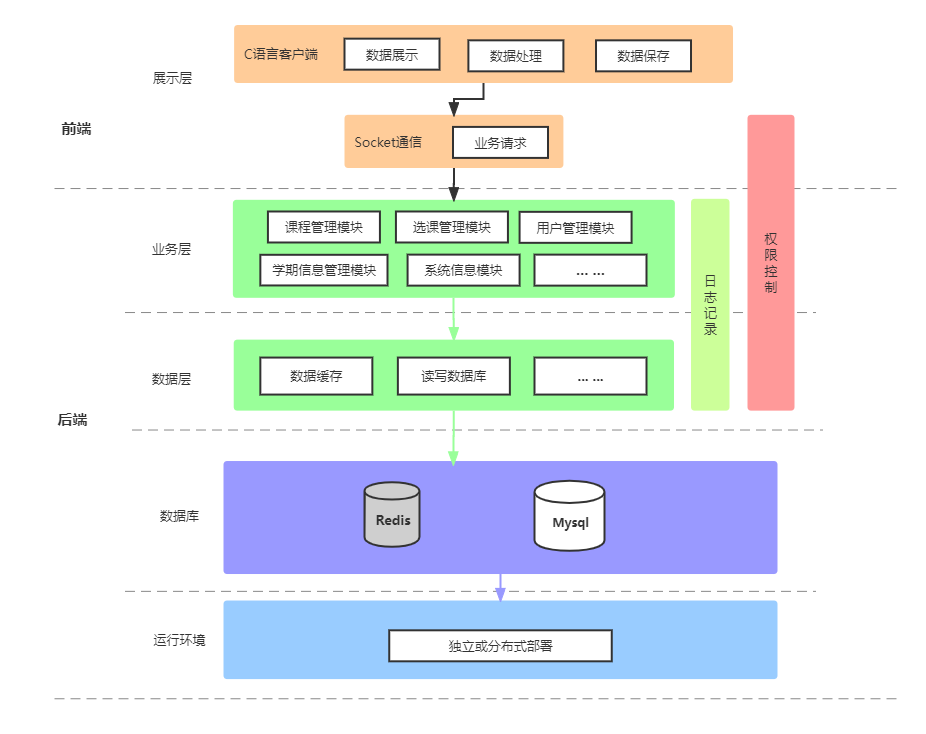


图3-1 系统架构设计图

### 关键功能/算法设计

#### 3.1.2.1 客户端与服务端数据交互功能实现

为实现数据交互，本系统制定了前、后端数据传输规范。

请求数据格式规范如下：

socket 0.1\n

此处socket为签名，版本号为识别不同版本协议的基础。

<client\_name> <os> <timestamp> <version> <sign>\n

第二行为客户端信息，用空格分割。

<client\_token|null>\n

第三行为客户端鉴权信息，使用jwt，如无则传入null。

<action>\n

第四行为传入的操作信息（服务器路由模块判断转发函数）。

<content>\n

JSON传入信息。

响应数据格式规范如下：

socket 0.1<length>

此处socket为签名，版本号为识别不同版本协议的基础，没有换行符，紧接着是长度为10的整数，表示当前数据长度，不足10位的部分补零。

<server\_name>,<resp\_start\_timestamp>,<resp\_end\_timestamp>,<status\_code>,<message>

接下来的数据为服务端信息，包括服务器名称、收到请求的时间戳、返回数据的时间戳、状态码与响应数据，数据间均用英文字符逗号分隔。状态码，在请求无异常时为0；若状态码非零，在额外信息部分一定包含message字段，用于描述错误信息，额外信息为JSON格式。

注：<sign>签名算法：hmac\_sha256(data=content+timestamp, key=client\_uuid)，其中，client\_uuid为客户端读取用户硬盘序列号后计算出的机器码，具有唯一性，在登录时作为参数提交，并存放在JWT中。Sign签名可以防止请求数据被篡改。

C客户端部分，可通过调用callBackend函数发送请求并解析服务器返回数据，callBackend函数流程如图3-2所示。首先，客户端会获取当前时间戳、机器码、操作系统环境、登录状态和请求数据等信息，并将其按照制定的规范组合字符串，通过winsock库发送数据。在接收到数据后，将会按照制定规范解析数据，若数据不符合规范，则视作无效数据，函数返回状态码-500（网络错误）；成功解析的数据将被存在ResponseData结构体中，作为函数返回值返回。

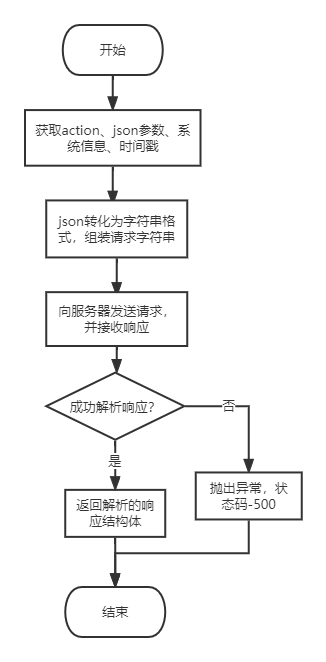


图3-2 callBackend函数流程图

Python服务端部分，使用socketserver库的ThreadingTCPServer开设多线程Socket服务器，保证单进程下可处理并发请求。接收请求后，服务器处理流程如图3-3所示，首先，服务器将解析请求，判断请求是否符合规范，并校验请求签名，判断请求是否过期或被伪造；接着，解析完毕的请求将通过路由模块判断并找到相应的业务流程，一个完整的业务流程通常包括中间件和主业务函数，在中间件函数和主业务函数间，使用context变量传递：context变量包含请求信息，响应信息和日志对象，函数可通过修改context变量中的信息实现信息传递的功能；业务流程结束后，会返回context变量，服务端可对响应信息进行处理，并通过Socket传递给客户端。

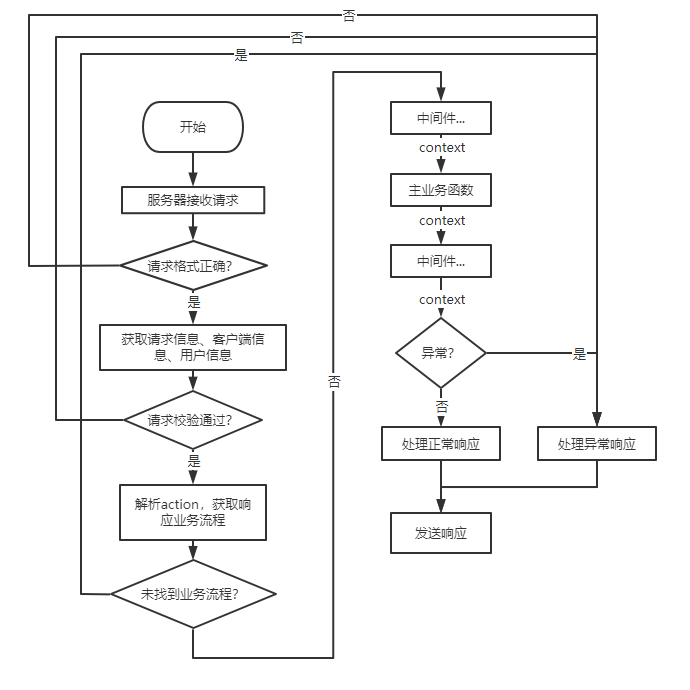


图3-3 服务器处理请求流程

#### 3.1.2.2 课程防超选功能实现

在接收选课请求前，Python会定时执行脚本，对选课状态预处理并存入Redis缓存数据库中，缓存时效为24小时，过期后将立即更新。

在Python服务端接收到选课请求后，会首先查询Redis缓存数据库中该课程对应的剩余人数，若仍有空余位置，则会在Redis的lua脚本中减少对应剩余人数，并进行选课业务流程；若没有空余位置，则抛出错误。

在接收到学生的退选请求时，Redis缓存数据库对应的剩余人数也会及时增加，在收到课程最大选课人数信息修改时，Redis数据库也会作出相应的更改，从而保证数据同步。

通过Redis数据库的原子操作，可保证在高并发场景下，对数据变动操作加锁，保证数据不越界。

选课流程如图3-4所示。

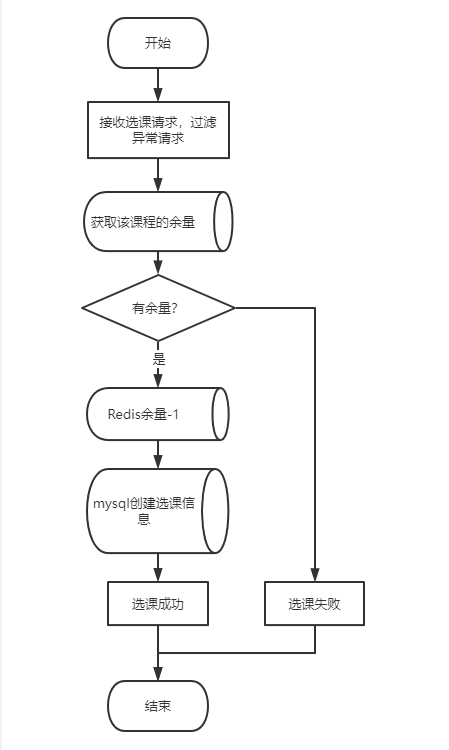


图3-4 选课流程

#### 3.1.2.3 客户端人机交互功能实现

通过使用Windows.h、conio.h、stdlib.h库，可实现对控制台的大小、显示颜色等的操作，同时，也可实现读取用户按键的功能，实现例如密码输入（如图3-5所示）、可交互数据列表（如图3-6所示）、可视化排课（如图3-7所示）等操作。

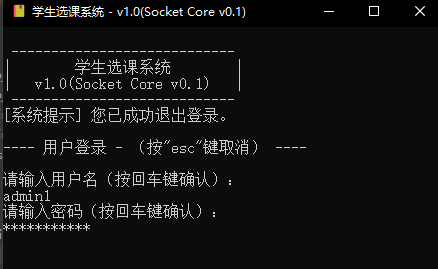


图3-5 密码输入不明文显示



图3-6 可交互的数据列表



图3-7 可视化排课

由于数据传输过程中，使用UTF-8编码格式，而Windows操作系统默认采用GBK编码格式，为保证兼容性，本系统还实现了UTF-8与GBK编码互相转化的功能，保证数据显示和数据传输正常。

在数据显示功能的实现过程中，本系统采用了链表数据结构，实现灵活的数据表长度，同时，为提升相应速度、保证用户体验，引入了分页、模糊搜索、排序等功能。

在新建课程、新建用户等操作时，本系统会对用户输入的数据进行正则表达式校验，保证数据的合法性。

#### 3.1.2.4 学生名单导入功能实现

为减轻用户工作量，在名单导入过程中，支持同时导入多个课程的数据。通过提供用户导入模板，并以可视化对话框选择文件导入的方式，保证了系统的用户友好度。同时，对于导入的数据，本系统将依次进行正则表达式校验，舍弃无效数据；面对可能出现重复数据的情况，本系统使用建立简易字符哈希表的方式实现大数据量的快速查重功能。为应对哈希冲突的情况，本系统采用哈希表+单链表的数据结构（如图3-8所示），即哈希表中存放链表节点，若冲突时，直接扩充链表即可。基于本系统中哈希表的使用场景皆为字符串，使用的哈希算法为：字符串中每个字符对应ASCII码值\*权重%哈希表大小并求和。在释放哈希表时，为保证释放效率，将预先再开辟一个变长数组，存放不同哈希值对应链表的头指针。

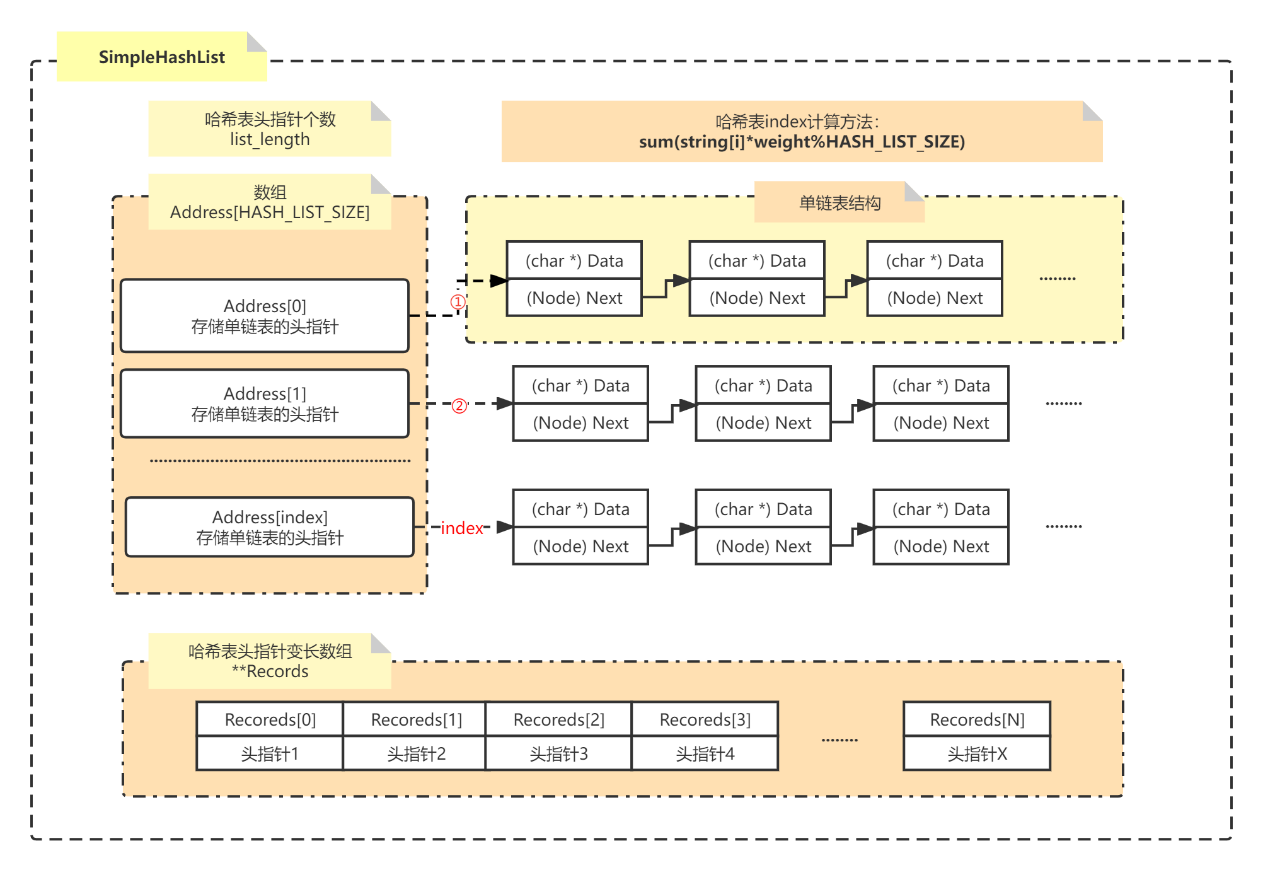


图3-8 Simple Hash List数据结构示意图

#### 3.1.2.5 自动登录功能实现

在用户登录时，将向服务端提供用户名、SHA256加密后的密码与设备码三个参数，服务端在校验通过后，将返回写到设备码的JWT和相关用户信息。客户端在接收后，以二进制的形式将登录状态数据保存至login.session文件中，为保证数据安全，登录状态数据将被AES加密。当用户再次打开程序时，程序会自动读取并解密登录状态数据，判断登录状态是否过期。若登录状态仍然有效，则向服务端发送校验请求，请求通过后即可自动登录客户端；若登录状态无效，则跳转至登录界面。

## 数据结构设计

### 存储数据

#### 3.2.1.1 数据库

表1 课程信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据表名: Courses 中文描述：课程信息表 | | | | | | |
| 字段名称 | 字段描述 | 数据类型 | 长度 | 允许空 | 缺省 | 备注 |
| Course\_id | 课程ID | varchar | 32 | N | / | 主键、索引 |
| title | 课程标题 | varchar | 100 | N | / | 索引 |
| description | 课程描述 | varchar | 500 | N | / | 索引 |
| type | 课程性质 | Int | 11 | N | / | 索引，0-必修、1-选修、2-公选、3-辅修 |
| Semester | 开课学期 | varchar | 10 | N | / | 索引 |
| Schedule | 课程安排 | Text | / | N | / |  |
| Week\_start | 开课周 | Int | 11 | N | / |  |
| Week\_end | 结课周 | Int | 11 | N | / | 结课周须大于开课周 |
| Points | 学分 | Float | / | N | / | 最多精确至2位小数 |
| Teacher | 开课教师UID | Varchar | 20 | N | / | 索引、外键 |
| Max\_members | 最大可选人数 | Int | 11 | N | / |  |

表2 选课信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据表名: Selections 中文描述：选课信息表 | | | | | | |
| 字段名称 | 字段描述 | 数据类型 | 长度 | 允许空 | 缺省 | 备注 |
| id | 选课ID | Int | 10 | N | / | 主键 |
| Uid | 选课用户UID | varchar | 15 | N | / | 索引、外键 |
| Course\_id | 课程ID | varchar | 32 | N | / | 索引、外键 |
| Select\_date | 选课时间 | Datetime | / | N | / | 索引 |

表3 用户信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据表名: Users 中文描述：用户信息表 | | | | | | |
| 字段名称 | 字段描述 | 数据类型 | 长度 | 允许空 | 缺省 | 备注 |
| Uid | 用户UID | varchar | 15 | N | / | 主键、索引 |
| Passwd | 用户密码 | varchar | 256 | N | / | 使用哈希算法加密（HMAC SHA 256），始终不会作为返回值返回至客户端 |
| Name | 用户姓名 | varchar | 20 | N | / | 索引 |
| Role | 用户角色 | Int | 11 | N | / | 索引，0-学生、1-教师、2-管理员 |
| Last\_login\_time | 最后登录时间 | Datetime | / | Y | NULL |  |
| Last\_login\_ip | 最后登录IP | Varchar | 200 | Y | NULL |  |
| Status | 用户状态 | Int | 11 | N | / | 索引，0-启用、1-停用 |

表4 学期信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据表名: Semester\_limits 中文描述：学期信息表 | | | | | | |
| 字段名称 | 字段描述 | 数据类型 | 长度 | 允许空 | 缺省 | 备注 |
| Semester | 学期 | Varchar | 10 | N | / | 主键、索引 |
| Max\_score | 最高选课学分 | Float | / | N | / |  |

表5 系统环境设置表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据表名:settings 中文描述：系统环境信息表 | | | | | | |
| 字段名称 | 字段描述 | 数据类型 | 长度 | 允许空 | 缺省 | 备注 |
| Key | 键 | Varchar | 255 | N | / | 主键、索引 |
| Value | 值 | Varchar | 255 | Y | NULL |  |

#### 3.2.1.2 文件存储

客户端在启动时将生成server.conf文件，该文件为服务器IP和端口配置文件，包含两行：第一行为说明内容，程序默认不读取；第二行为IP和端口信息，以空格分隔，默认为127.0.0.1和2333。

为实现自动登录功能，客户端在登录成功后将生成login.session文件，该文件为加密后的二进制文件，具体见3.1.2.5（自动登录功能实现）部分。

人机交互过程中，学生可选择导出课表，教师和管理员可选择导出/导入学生名单。

### 接口（模块接口、系统间接口）

客户端与服务端之间使用Socket通信。客户端内部，包含多个功能接口。多数接口在传递时使用指针变量实现数据传递。客户端的部分关键接口列表见表6。完整接口见代码。

表6 关键接口列表

|  |  |
| --- | --- |
| 接口声明 | 接口用途 |
| void loadBasicInformation(); | 程序初始化，加载基本数据 |
| char \*findExceptionReason(ResponseData \*resp); | 通过服务器相应数据，分析错误原因 |
| char \*UTF8ToGBK(const char \*strUTF8); | UTF8编码转换为GBK编码 |
| char \*GBKToUTF8(const char \*strGBK); | GBK编码转换为UTF8编码 |
| LinkList\_Object \*linkListObject\_Init(); | 链表结构初始化 |
| LinkList\_Node \*linkListObject\_Append(LinkList\_Object \*list, void \*data); | 链表尾部增加节点 |
| void linkListObject\_Delete(LinkList\_Object \*list, char deleteData); | 删除链表 |
| SimpleHashList \*hashList\_init(); | 哈希表初始化 |
| int hashList\_appendString(SimpleHashList \*list, char \*string); | 哈希表增加String |
| char hashList\_findString(SimpleHashList \*list, char \*string); | 哈希表搜索 |
| void hashList\_delList(SimpleHashList \*list); | 哈希表释放 |
| void printStudentCourseSelection(); | 打印学生选课界面 |
| void printSemesterData(); | 打印学期信息管理界面 |
| ui\_printMainMenu(char wrong\_command); | 打印主菜单 |
| int Serv\_login(char status); | 执行登录操作 |
| int Serv\_getLoginSession(); | 执行自动登录 |
| void printAllUsers(); | 打印用户管理界面 |
| ResponseData callBackend(const char \*action, cJSON \*json); | 向后端发送请求 |
| char regexMatch(char \*pattern, char \*string); | 正则表达式匹配 |

### 关键数据结构

// 链表节点定义

typedef struct linkListNode\_\_ {

struct linkListNode\_\_ \*prev, \*next; // 前后指针

void \*data; // 数据存放处指针

} LinkList\_Node;

// 链表结构体定义

typedef struct linkListObject\_\_ {

LinkList\_Node \*head, \*foot; // 头尾指针

int length; // 链表长度

} LinkList\_Object;

// 哈希表节点定义

typedef struct hashNode\_\_ {

struct hashNode\_\_ \*next; // 后指针

char \*data; // 字符串存储

} SimpleHashList\_Node;

// 哈希表定义

typedef struct hash\_list\_\_ {

SimpleHashList\_Node \*address[HASH\_LIST\_MAX\_LENGTH]; // 哈希主表，用于快速定位

SimpleHashList\_Node \*\*records; // 变长数组，顺序记录节点内存地址，便于释放

unsigned int list\_length; // 哈希列表长度

} SimpleHashList;

// 系统当前用户登录状态存储结构

typedef struct userInfo {

char jwt[500], // JWT为判断用户身份信息的Token

userid[30], // 用户UID

name[20]; // 用户姓名

long long expired; // JWT过期时间，为时间戳格式

int role; // 用户角色

} UserInfo;

// 在可视化排课操作时，用于传递一个完整的数组，获取后再使用memcpy传递给course的schedule

typedef struct schedule\_ {

int schedule[7][13];

} Schedule;

// 课程信息结构体

typedef struct course {

char course\_id[33], // 课程ID

title[101], // 课程标题

description[501], // 课程描述

semester[11]; // 开课学期

User teacher; // 开课老师

int type, // 课程类型：0-必修 1-选修 2-公选 3-辅修

week\_start, // 开课周

week\_end, // 结课周

current\_members, // 当前报名人数

max\_members; // 最大可报名人数

double points; // 学分

int schedule[7][13]; // 每星期课程安排

} Course;

// 学生选课结构体

struct studentCourseSelection {

int status; // 是否可选该门课程

char locked\_reason[100]; // 该门课不能选择的原因

long long selection\_time; // 选择该课程的时间

Course course; // 课程信息

};

// 学生选课管理结构体

struct teacherCourseSelection {

User student; // 学生信息

Course course; // 课程信息

long long selection\_time; // 选择该课程的时间

};

// 学期信息结构体

typedef struct semester\_limit\_\_ {

char semester[11]; // 学期

double max\_points; // 学期最大学分

} SemesterLimit;

// 用户信息结构体

typedef struct user {

char uid[21], // 用户ID

name[21],// 用户姓名

passwd[65], // 密码

last\_login\_ip[201]; // 用户最后一次登录IP

int role,// 用户角色

status; // 用户状态 0-启用 1-停用

long long last\_login\_time; // 最后一次登录时间戳

} User;

// 响应数据结构体

typedef struct responseData {

int start\_resp\_ts, // 响应开始时间戳

finish\_resp\_ts, // 响应结束时间戳

status\_code; // 状态码

char \*raw\_string; // 响应数据（字符串格式）

cJSON \*json; // 响应数据（JSON格式）

} ResponseData;

## 系统界面设计

### 界面设计风格

界面为控制台界面，采用键盘按键实现人机交互。参见3.3.2（主要功能页面）各图所示。

### 主要功能页面

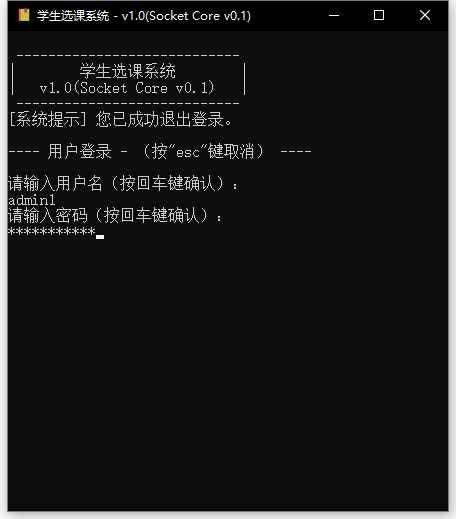


图3-9 用户登录界面

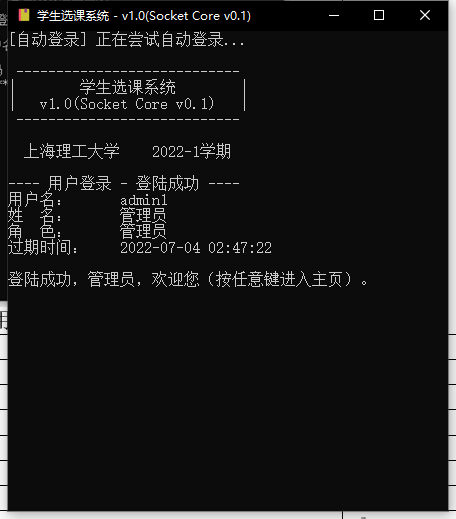


图3-10 用户自动登录界面

图3-11 学生主菜单界面



图3-12 学生选课界面

图3-13 学生课表界面



图3-14 学生/教师-全校课表一览界面



图3-15 教师/管理员课程信息管理界面



图3-16 教师/管理员开课界面

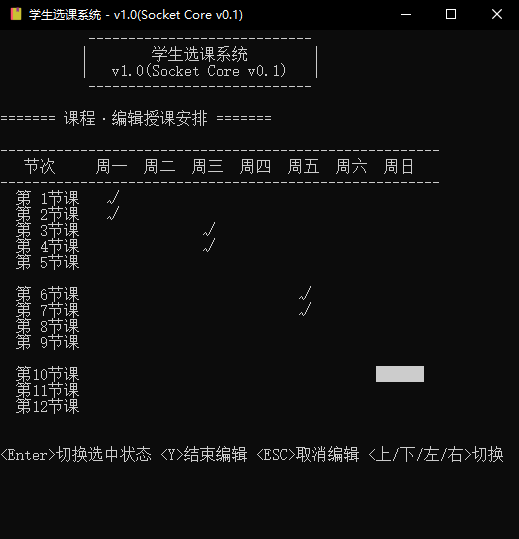


图3-17 教师/管理员排课界面



图3-18 教师/管理员编辑课程界面

图3-19 教师学生名单查看页面

图3-20 管理员学生名单查看页面

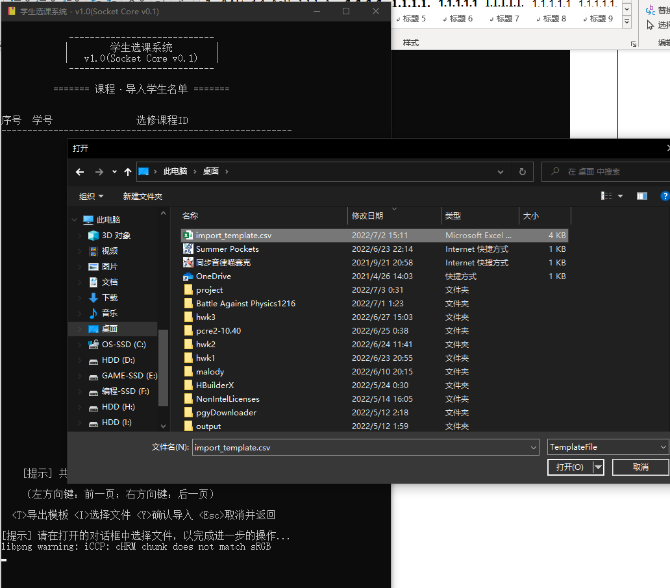


图3-21 管理员导入学生名单界面1

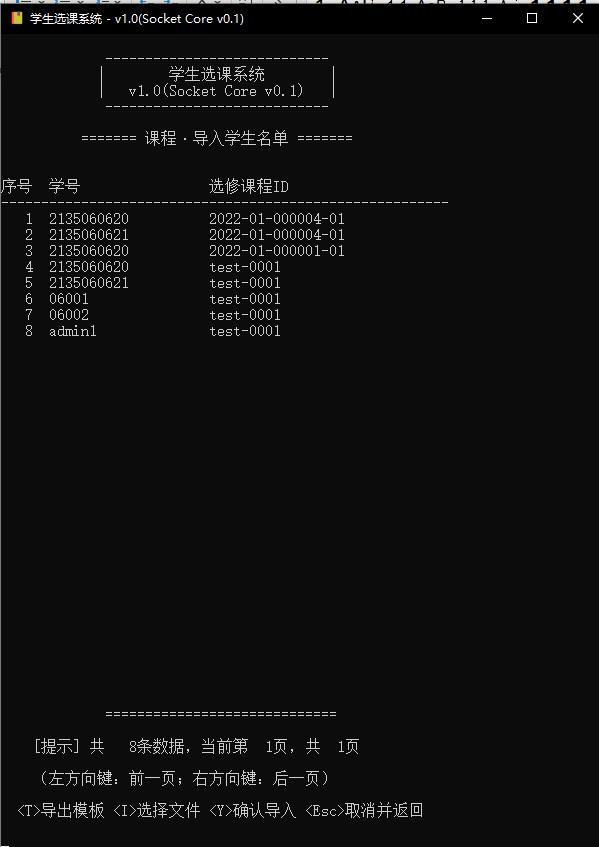


图3-22 管理员导入学生名单界面2

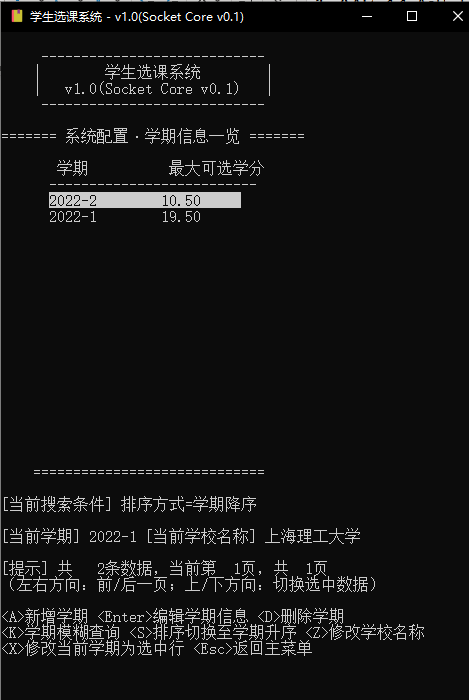


图3-23 管理员系统配置管理界面

图3-24 管理员用户管理界面

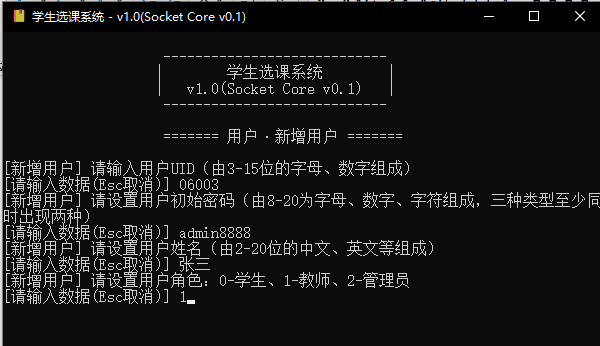


图3-25 管理员新增用户界面



图3-26 管理员新增/编辑用户界面

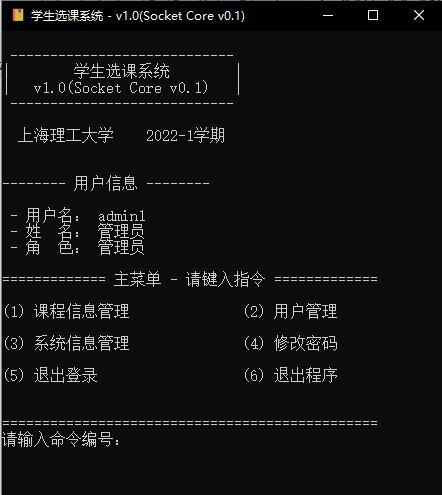


图3-27 管理员主菜单界面

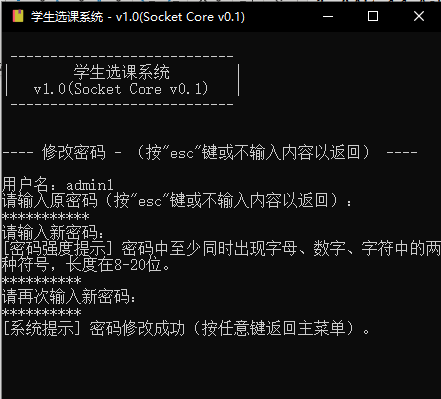


图3-28 用户修改密码界面

# 系统安装及使用说明

本系统之C语言客户端仅支持在Windows环境下编译运行。编译时，请注意编译器的存储位置不能包含空格、中文或特殊符号，以免在编译rc文件时编译失败。建议安装clion后，打开项目编译（安装时，请注意安装路径不能包含空格）。

请在编译前仔细阅读CMakeLists.txt的注释。由于CPU架构的多样性，若因pcre2库不支持，导致编译失败，可至项目官网：<https://pcre2project.github.io/pcre2/> 下载源代码后，自行编译。

本系统之服务端可在Linux/MacOS/Windows等多个操作系统环境下运行。请确保在运行前已安装Python3.8或更高版本，并安装requirements.txt中的所有支持库、Mysql 5.7或更高版本的数据库和Redis缓存数据库。为保证程序正常运行，请在开启服务器之前，运行源代码文件夹下mysql/course\_system.sql文件，对数据库进行初始化，同时，在mysql/db\_config.py文件中配置mysql数据库连接账户，并在config/redis\_config.py文件中配置Redis缓存数据库连接信息。

服务器端运行过程中将产生日志文件，默认保存至/log/文件夹下。请确保服务器空间充足。

# 总结

为了顺应高校智慧校园建设的历史潮流，满足师生相关需求，有效提高高校学生选课管理的工作效率和信息化水平，开发了一个学生选课系统（简称：本系统）。

本系统具有以下功能：学生课程选择；教师课程开设、课程信息管理；管理员用户管理、课程与选课信息管理等。

本系统采用C/S技术架构。在服务端综合使用Python、Mysql、Redis，实现高并发场景下不出现“超选”的情况，同时，提供流畅的用户体验。在服务端与客户端之间通信过程中，为保证数据安全，采用完全自定义的数据传输规范，并在请求中包含签名以防止信息篡改。为保证用户登录状态数据安全，将本地Session文件以AES加密的密文形式保存，且密钥为相对唯一的机器码。在批量导入学生名单时，会自动去除重复的数据，在面临大数据量的导入场景时，本系统采用了哈希表的数据结构保证查重的低时间复杂度（最低可达到O（1）），以保证用户的使用体验。

本系统结合数组、结构体、链表、哈希表等多种数据结构，并综合使用Windows.h、conio.h等库，从而确保较为用户友好的人机交互体验。

由于时间仓促，本系统仍然存在若干有待进一步完善之处，例如：C语言客户端的内存占用较大，错误处理尚有提升空间。在通信过程中，虽已加入签名，但仍然是明文传输，数据易被窃听，后续可通过对数据RSA加密等方式防止抓包。python服务器实现功能简单，中间件有待开发。在课程信息管理方面，多位老师教授同一门课、分年级授课等场景暂未实现；批量编辑、和批量删除功能暂未实现。在学期信息管理方面，仅实现了限制最高选课学分，尚未做更多配置。

# 附录

## 名词定义

|  |  |
| --- | --- |
| 名词/缩写 | 说明 |
| JWT | Java Web Token，为了在网络应用环境间传递声明而执行的一种基于JSON的开放标准。 |

## 参考资料

[1] DaveGamble. cJSON[EB/OL]. [2022-06-26]. https://github.com/DaveGamble/cJSON.

[2] Clnulijiayou. HMAC-SHA256签名加密[EB/OL]. [2021-05-28]. <https://blog.csdn.net/Clnulijiayou/article/details/117374825>.

[3] 码农小明. AES算法描述及C语言实现[EB/OL]. [2018-05-06]. <https://blog.csdn.net/shaosunrise/article/details/80219950>.

[4] 洌冰. 数据结构 Hash表（哈希表）[EB/OL]. [2018-05-20]. <https://blog.csdn.net/u011109881/article/details/80379505>.

[5] Microsoft. MultiByteToWideChar function (stringapiset.h)[EB/OL]. [2021-10-13]. <https://docs.microsoft.com/en-us/windows/win32/api/stringapiset/nf-stringapiset-multibytetowidechar>.

[6] Microsoft. WideCharToMultiByte function (stringapiset.h)[EB/OL]. [2021-10-13]. <https://docs.microsoft.com/en-us/windows/win32/api/stringapiset/nf-stringapiset-widechartomultibyte>.

[7] Microsoft. GetVersion function (sysinfoapi.h)[EB/OL]. [2021-06-30]. <https://docs.microsoft.com/en-us/windows/win32/api/sysinfoapi/nf-sysinfoapi-getversion>.

[8] CMake. CMake Documentation[EB/OL]. [2022-07-03]. <https://cmake.org/documentation/>.

## 源代码清单

本系统的源代码包括：Python服务端源代码、数据库初始化用sql代码和C语言客户端代码。