中山大學

SUN YAT-SEN UNIVERSITY

C++ 高级程序设计

基于 html 前端与 c++ 后端的学院学生管理系统设计与实现

姓 名 肖伟文

学 号 24305079

学 院 智能工程学院

专 业 智能科学与技术

2026 年 1 月 20 日

摘　要

本报告介绍了一个基于 HTML 前端与 C++ 后端的学院学生管理系统的设计与实现。系统以分层架构为基础，前端负责交互与展示，后端使用 C++ 实现业务逻辑与数据持久化（基于 JSON 文件存储），并通过轻量级 HTTP 库对外提供 RESTful 接口。本文阐述了系统需求、总体设计、关键模块（学生、课程、用户）的数据模型与接口，重点讨论了服务层与 DAO 层的职责划分、路由统一注册及异常处理策略。实现部分包含路由设计示例、数据加载/保存流程与并发控制方案；在功能验证中展示了用户登录、学生信息 CRUD、课程管理与文件上传等主要用例。最后对系统的性能、可扩展性和后续改进（如引入数据库、权限控制与单元测试）进行了总结与展望。

关键词：学生管理系统；C++ 后端；HTML 前端；RESTful；JSON 持久化

目录

1 引言

在小学时期，像是 C++ 程序设计入门这样的书籍就已经广泛出现在书店以及图书馆中。而书中的最后一个章节往往就是让读者们设计一个简单的学生管理系统或是图书管理系统，其实现往往是一个命令行的黑框，将功能与 ui 都一起写在同一个程序里面。ui 上列举着程序实现的每个功能，以 1，2，... 来分隔，让用户输入一个个字符或是字符串来决定程序的走向。

那时，大家常常爱好用这样的文字交互的方式来编写一款款小游戏，来使枯燥的集训生活获得一些心灵上的慰藉，同时也是展示自己简单编程能力的一个途径。那时大家主要学习的内容是算法，大家最多也就搭一个博客网站，也没有什么人会使用 winapi 之类的东西去做图形化页面。大家一般都是一个黑框，一个个分支走向不同的游戏结局，或者积攒金币来打怪兽，诸如此类。

如果刚好有一个机会，让人能学习一些超过黑框的知识，那无疑是非常好的。借用这个机会，我正好了解了一些我从未了解的 web 编程，或是通信框架的知识。感谢这门课，感谢老师和助教们，让我有机会学习到这些知识并学以致用。

2 基本要求的完成

- 代码 750 行 +，给出简单注释

  显然这个项目的代码还是很多的（虽然很多都是重复的，CRUD 之类），在我经常忘记的地方也写上了注释。

  后端 1800 行左右，命令行前端 400 行左右，html+css+js 就很多了，大概 1800 行左右，这是因为 js 文件有辅助的缘故（毕竟网页端功能多一点），不过 html 的代码行数应该是不记的。

- 定义类型 Classxx

  在 Course, Student, User 文件夹中很多了。

- 数据成员包含指针

  在各大 Service 层中均有指针成员。

- 定义构造函数，复制构造函数，运算符重载

  在 BaseEntity 类中均有定义。

- 静态成员，友元，动态内存分配

  静态成员和友元函数在 BaseEntity 类中有定义，动态内存分配姑且认为是 STL 容器的使用了。

- 继承派生，虚基类，虚函数

  Course, Student 类虚继承自 BaseEntity 类，BaseEntity 类中有虚函数。

- 便用 STL. 要求用到容器、函数、算法（查找，排序等），模板

  DAO 层使用了 vector, map 混合存储法,template 在 Result.cpp 和 http_helper.cpp 中大量使用。

- 文件读取与存储

  所有 DAO 层均有文件读取与存储的功能，使用 JSON 文件存储数据。

3 系统版面

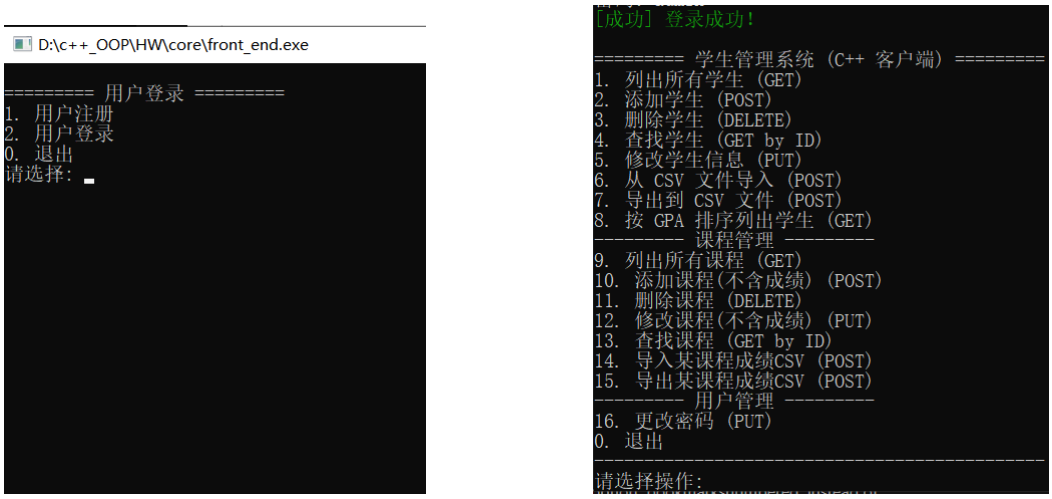

图 1: 命令行前端界面展示

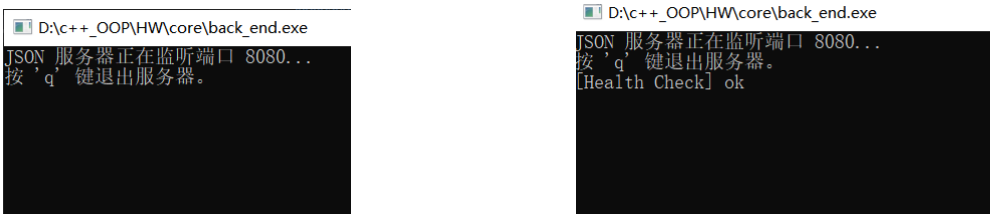

图 2: 命令行前端界面展示

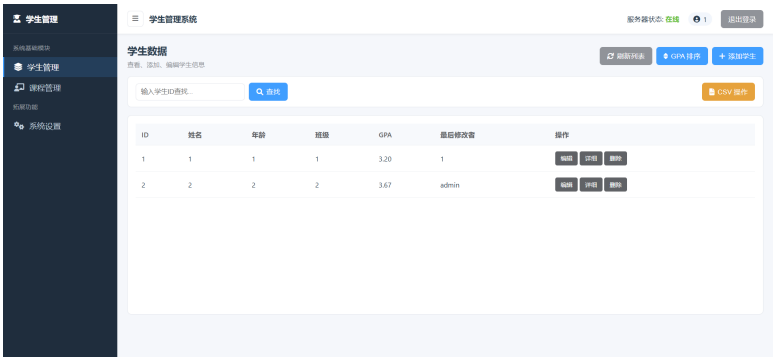

图 3: html 前端展示图

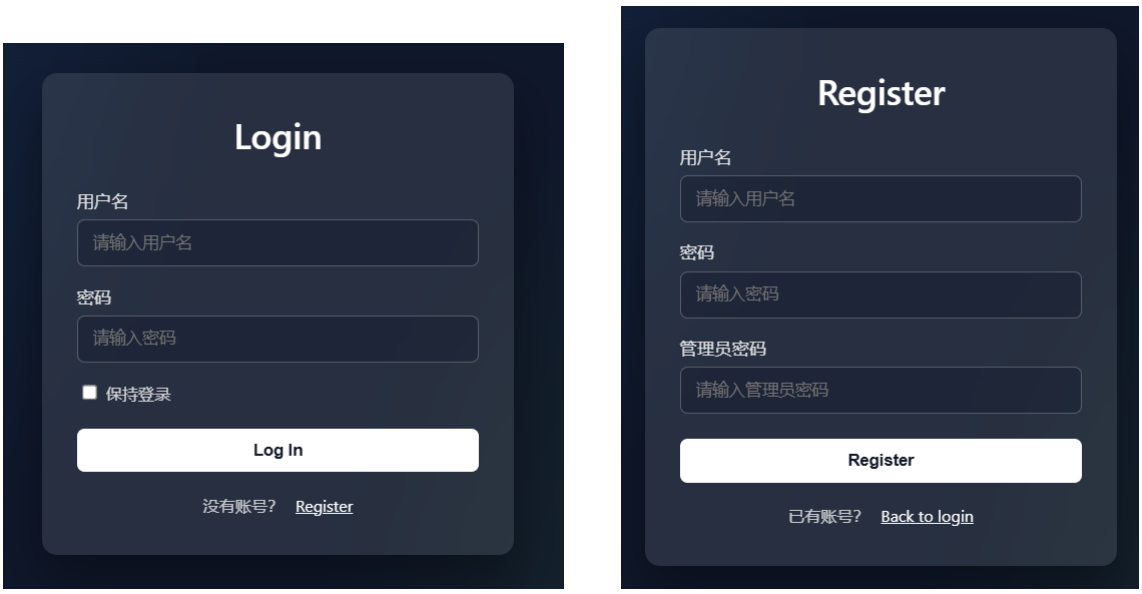

图 4: html 前端登陆注册界面展示

图 5: 学生信息分析图

图 6: 课程信息展示图

4 设计思路

4.1 系统架构

本系统采用前后端分离的架构设计，前端使用 HTML、CSS 和 JavaScript 构建用户界面，后端使用 C++ 实现业务逻辑和数据处理。前端通过 HTTP 请求与后端进行通信，后端提供 RESTful API 接口供前端调用。后端系统采用分层架构设计，主要包括以下几个层次：

- 控制器层（Controller Layer）：负责处理来自前端的 HTTP 请求，调用相应的服务层方法，并将结果返回给前端。
- 服务层（Service Layer）：封装业务逻辑，处理具体的业务需求，如学生信息管理、课程信息管理等。服务层调用数据访问层进行数据操作。
- 数据访问层（DAO Layer）：负责与数据存储进行交互，提供数据的增删改查功能。数据存储采用 JSON 文件进行持久化。
- 实体层（Entity Layer）：定义系统中的主要数据模型，如学生、课程和用户等。

注：控制器层依赖 httplib.h 库实现 HTTP 服务器功能，通信借助 JSON 类型数据，依赖 nlohmann/json 库；数据访问层本来想用 SQLite 数据库，但由于时间关系，最终采用 JSON 文件存储数据。

4.2 UML 图

系统的 UML 图如下所示：

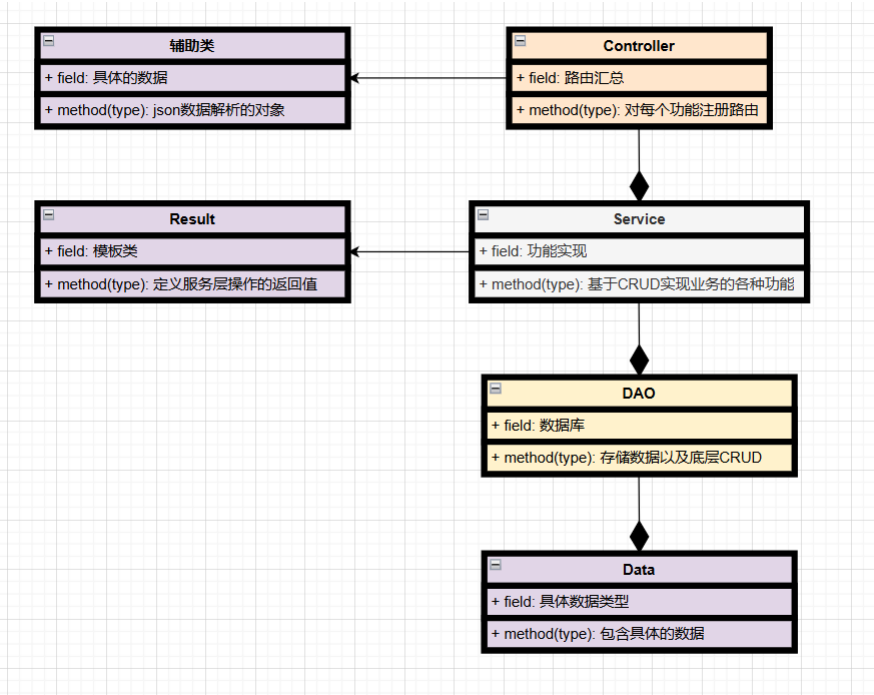
辅助类
+ field: 具体的数据
+ method(type): json数据解析的对象
Controller
+ field: 路由汇总
+ method(type): 对每个功能注册路由
Result
+ field: 模板类
+ method(type): 定义服务层操作的返回值
Service
+ field: 功能实现
+ method(type): 基于CRUD实现业务的各种功能
DAO
+ field: 数据库
+ method(type): 存储数据以及底层CRUD
Data
+ field: 具体数据类型
+ method(type): 包含具体的数据

图 7: 类图

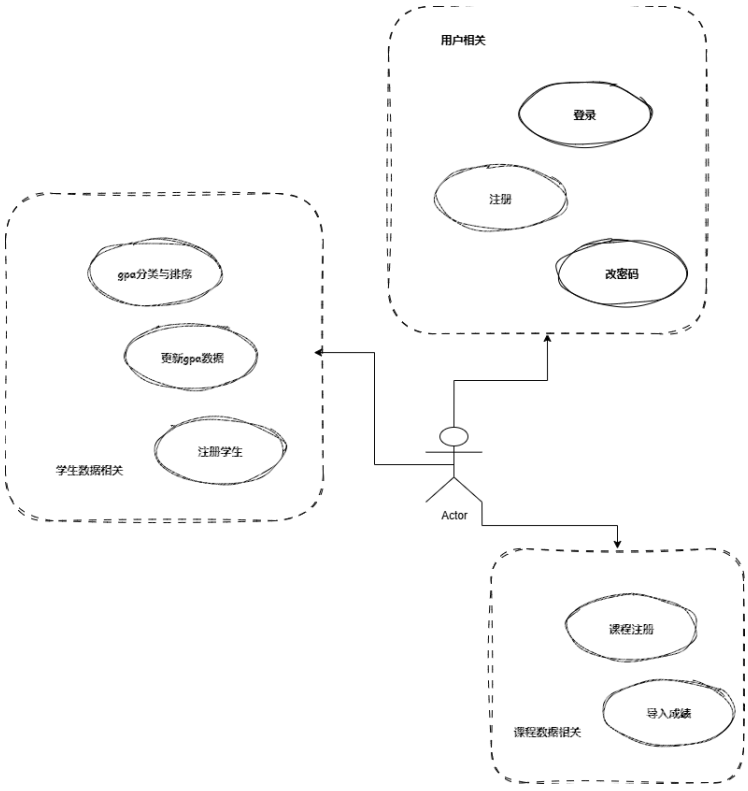
用户相关
登录
注册
改密码
gpa分类与排序
更新gpa数据
注册学生
学生数据相关
Actor
课程注册
导入成绩
课程数据相关

图 8: 实例图

5 结果

代码中成功实现用户注册、登录功能，学生信息的增删改查功能，课程信息的增删改查功能，以及文件上传功能。通过前端页面可以直观地进行各项操作，后端能够正确处理请求并返回相应的数据。

具体使用可见 readme.md 文件，设计思路可见 design.md，api.md 文件。

以下是项目文件夹架构树状图：

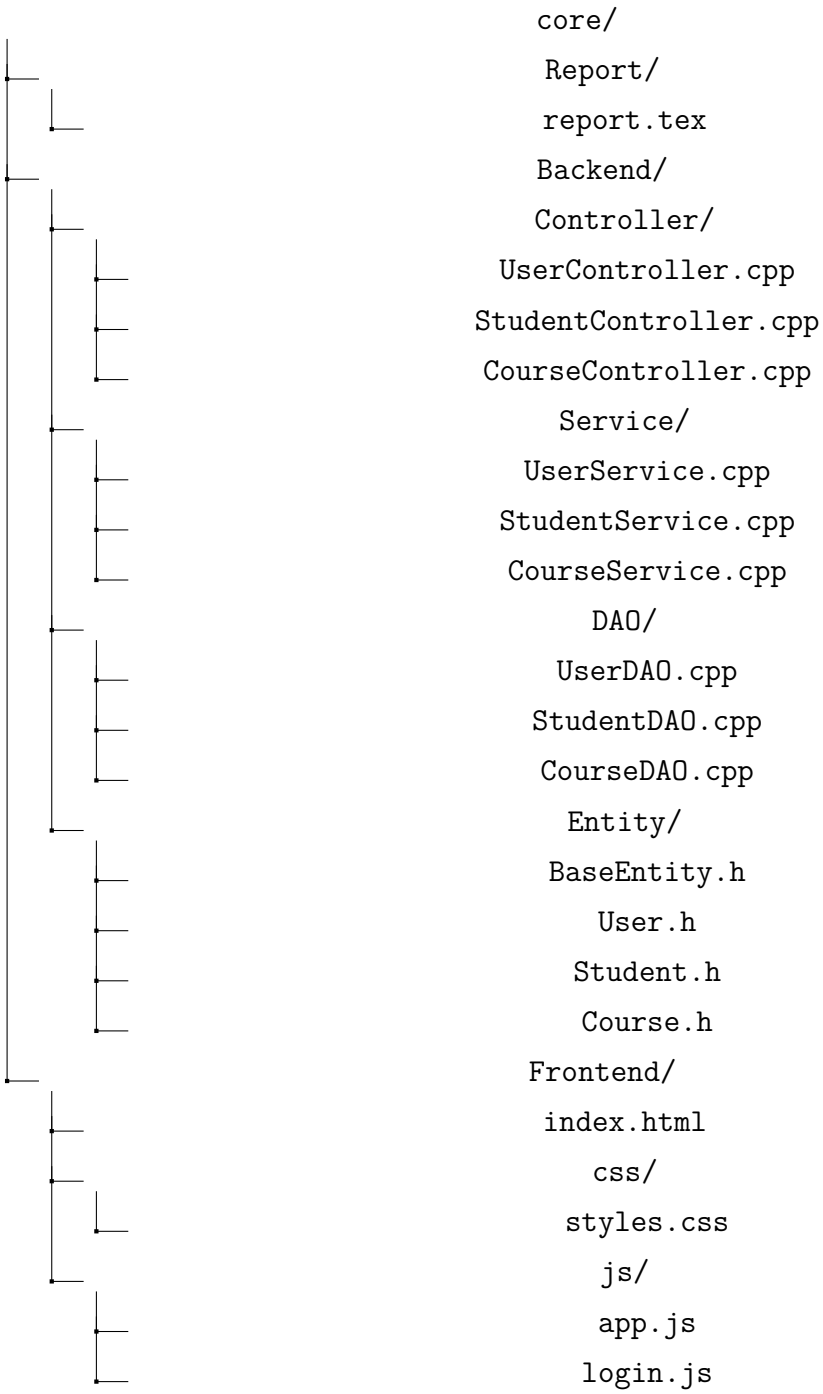

图 9: 项目文件夹架构树状图

6 总结与展望

通过本次课程设计，我深入理解了前后端分离的架构设计理念（B/S, C/S），掌握了 C++ 在后端开发中的应用，熟悉了 HTTP 通信和 RESTful API 的设计方法。同时，我也提升了自己的编程能力和项目管理能力，学会了如何将理论知识应用于实际项目中。

如果有时间改进该学生管理系统，我会增加更多功能，如权限管理、数据统计分析等。同时，我也希望能够学习更多关于数据库设计和优化的知识，将系统的数据存储从 JSON 文件迁移到关系型数据库（MYSQL）中，以提升系统的性能和可扩展性。

通过与同学的交流，我发现我的路由头写的太复杂了，没有充分利用 json 数据的性质。并且模糊查找也并不一定需要使用 LSH 这样的复杂算法，跑一跑 KMP 也是可以的，不过目前的完成度是我能达到的极限了。

本项目已上传至 GitHub，欢迎访问和使用：https://github.com/x-x-gpu/Student-Management-System-with-c-and-html