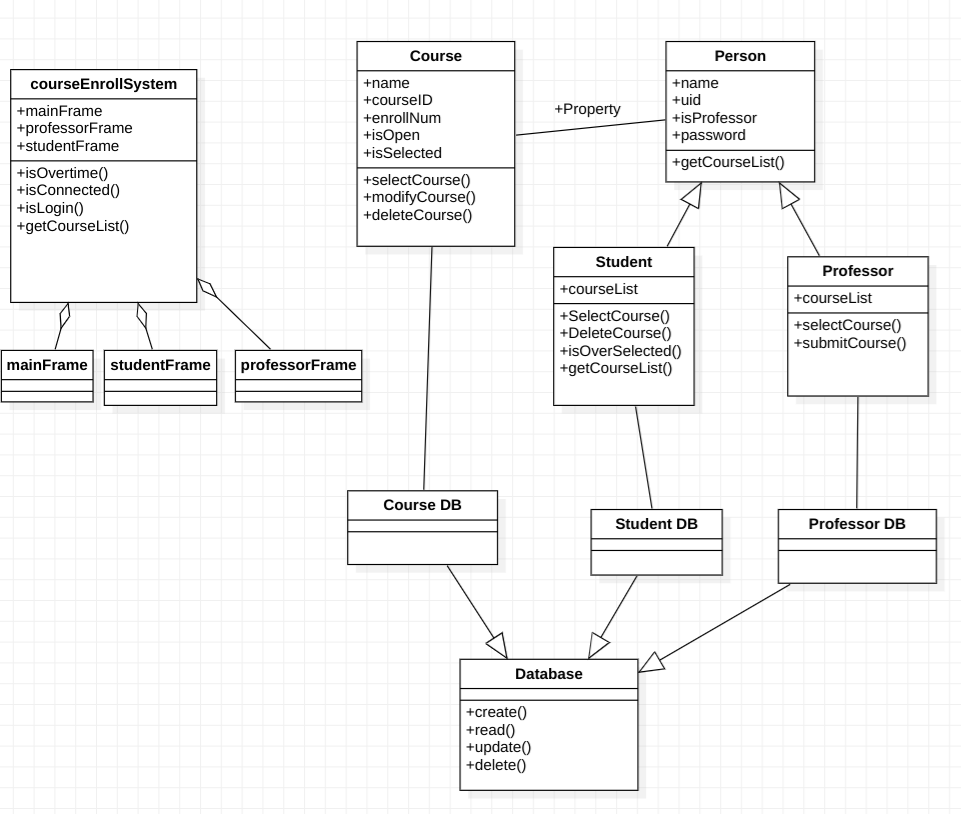
**面向对象分析与设计作业**

2020年11月

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 班级 | 学号 | 负责完成任务 | 参加完成任务 | 备注 |
| ttzztztz |  |  | UML设计、报告撰写 | UML设计、报告撰写 |  |
| Macsed |  |  | UML设计、报告撰写 | UML设计、报告撰写 |  |

1. **系统类图**
   1. **三层结构系统类图**



**1.2实体类属性设计**

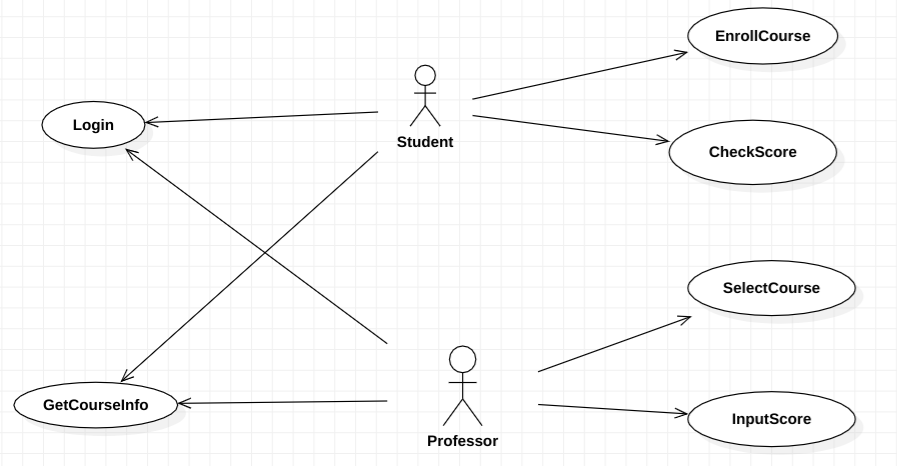
|  |  |  |
| --- | --- | --- |
| 类名 | 属性 | 说明 |
| Student | name | 学生姓名，数据类型：string |
| userID | 用户账号，数据类型：int |
| password | 密码，数据类型：int |
| isProfessor | 人物身份，数据类型：boolean |
| mainCourseNum | 已选主要课程数量，数据类型：int |
| optCourseNum | 已选备选课程数量，数据类型：int |
| enrollSchedule | 学生的学习计划，数据类型：arraylist |

|  |  |  |
| --- | --- | --- |
| 类名 | 属性 | 说明 |
| Professor | name | 学生姓名，数据类型：string |
| userID | 用户账号，数据类型：int |
| password | 密码，数据类型：int |
| isProfessor | 人物身份，数据类型：boolean |
| selectCourseList | 老师教授的课程列表，数据类型：arraylist |

|  |  |  |
| --- | --- | --- |
| 类名 | 属性 | 说明 |
| Course | name | 课程名称，数据类型：string |
| courseID | 课程编号，数据类型：int |
| enrollNum | 选择该课的人数，数据类型：int |
| isOpen | 本学期是否开了这门课，数据类型：boolean |
| isSelected | 是否被当前学生选中，数据类型：boolean |

**2 系统用例图**

**2.1 系统用例图**



**2.2 参与者说明**

**学生**：

每个学期初，学生可以登录该系统（Login），获取这个学期所开设的所有课程的目录（GetCourseInfo），并且可以选取注册相应课程（EnrollCourse）。学期结束，学生可以查询成绩（CheckScore）。

教授：新学期学生选课前，教授可以登录该系统（Login），获取这个学期所开设的所有课程的目录（GetCourseInfo），选取自己决定要教授的课程，并获取所教课程的信息（SelectCourse）。学期结束后，教授可以对选取自己课程学生的成绩进行登记（InputScore）。

**2.3 用例简要说明**

**用例**：登录（login）

**简要描述**：学生和教授完成选课系统登录的整个过程。

**参与者**：学生和教授

**前置条件**：系统运行正常。

**基本流程**：学生和教授输入用户ID和密码，点击提交，系统对ID和密码进行有效检验，

检查通过，则返回至用户界面。

**异常**：密码输入错误，系统弹出密码输入异常的警告，学生和教授重新输入密码。

**后置条件**:如果学生和教授登陆成功，可获取这个学期所开设的所有课程的目录，进行

选取。

**用例：**获取课程目录（GetCourseInfo）

**简要描述：**在学期开始阶段，学生和教授可以获取本学期开设的所有课程列表。

**参与者：**学生和教授

**前置条件：**开始这个用例之前，学生和教授必须已登录（login）到系统。

**基本流程：**学生和教授进入选课系统主界面，界面显示本学期开设的所有课程的详细信息。

**异常：**无

**后置条件：**如果用例成功结束，学生可对课程进行注册，教授可对所教课程进行选取。

**用例：**注册课程（EnrollCourse）

**简要描述：**这个用例允许学生注册本学期需要学习的课程。在学期开始阶段，学生可以修改或者删除所选择的课程。课程目录系统提供了本学期开设的所有课程列表。

**参与者：**学生

**前置条件：**开始这个用例之前，学生必需已登录到系统。

**事件流程：**当学生希望注册课程，或想改变他的课程计划（schedule）时，用例开始执行。

1）系统要求学生选择要执行的操作（创建计划，修改计划，或删除计划）。

2）一旦学生提供了系统要求的信息，以下子流程中的某一个将被执行。

(1)如果学生选择的是“create a schedule”，则“创建计划”子流程将被执行。

(2)如果学生选择的是“update a schedule”，则“修改计划”子流程将被执行。

(3)如果学生选择的是“delete a schedule”，则“删除计划”子流程将被执行。

1.1 创建计划

1）系统从课程目录系统中检索出有效的课程列表并显示。

2）学生从有效课程列表中选择4门主选课和2门备选课。

3）当学生完成选择后，系统将为这个学生创建一个“课程计划”，这个课程计划包含了学生所选的课程。

4）执行“提交计划”子流程。

1.2 修改计划

1）系统检索并显示学生当前的课程计划。

2）系统从课程目录系统中检索出有效的课程列表并显示。

3）学生从有效的课程列表中选择要增加的课程，也可以从当前的课程计划中选择任何想要删除的课程。

4）当学生完成选择，系统将修改这个学生的课程计划。

5）执行“提交计划”子流程

1.3 删除计划

1）系统检索并显示学生当前的课程计划

2）系统提示学生确认这次删除

3）学生确认这次删除

4) 系统删除课程计划。

如果这个课程计划中包含“已注册”（enrolled in）的course offering，则在 course offering 中删除关于这个学生的信息。

1.4 提交计划

1）对于课程计划中所选的每门课程，如果还没有标记为“已注册”，则系统将验证学生是否满足选修条件、课程是否处于“open”状态以及课程计划中是否没有冲突。如果验证通过，则系统将把学生加到所选的course offering中，课程计划中所选的课程标记为“已注册”。

2）课程计划被保存在系统中。

2 可选流程

2.1 保存计划

在任何情况下，学生可以选择保存课程计划而不提交课程计划。在这种情况下，“提交计划”这一步被下面的步骤所代替：

1）课程计划中没有被标记为“已注册”的课程应标记为“选择”（selected）。

2）课程计划被保存在系统中。

2.2 选修条件不满足或课程满员或课程计划冲突。

如果在“提交计划”子流程中，系统检测出学生没有满足选修条件，或学生所选的课程已满，或课程计划存在冲突，则系统显示错误消息。学生可以选择其他课程（用例继续），或保存课程计划（和“保存计划”子流程一样） ，或取消此操作，如果是取消操作，则用例基本流程重 新开始。

2.3没有找到计划

如果在“修改计划”或“删除计划”子流程中，系统不能检索到学生的课程计划，则系统显示错误信息。学生确认该错误，用例基本流程重新开始。

2.4 课程目录系统不可用

如果系统不能和课程目录系统通讯，则系统将向学生显示 错误信息，学生确认该错误，用例终止。

2.5 课程注册结束

如果在用例开始的时候，系统检测到已经过了本学期的课程注册时间，则系统将向学生显示信息，用例终止。学生在本学期的课程注册结束后就不能再注册课程了。

2.6 取消删除

如果再“删除计划”子流程中，学生决定不删除课程计划了，则删除被取消，用例基本流程重新开始。

**异常：**无

**后置条件：**如果用例成功结束，则会创建、修改或删除学生的课程计划，否则系统的状态不变。

**用例：**查看课程分数（CheckScore）

**简要描述：**学生选择查看分数，系统输出成绩单。

**参与者：**学生

**前置条件：**学生已登录。

**基本流程：**学生登录系统，选择查看分数。系统把学生的各科成绩输出在屏幕上。

**异常：**

1. 学生没有选课，没有学习计划。系统提示，你目前还没有选课，无法输出成绩。

2. 教授目前没有输入成绩。系统提示，课程成绩尚未公布。

**后置条件：**学生选择查看成绩。

**用例：**选择课程（SelectCourse）

**简要描述：**这个用例允许教授选择本学期决定要教的课程。

**参与者：**教授

**前置条件：**开始这个用例之前，教授必须已登录到系统。

**基本流程：**当教授希望选取课程时，用例开始执行。教授选取将要教的课程，完成后进行提交。

**异常：**若教授所选课程存在日程冲突，则系统发出警告，返回选课界面。

**后置条件：**如果用例成功结束，则会创建教授所选课程的列表。

**用例：**教授输入课程分数。（InputScore）

**简要描述：**教授选择输入分数，系统给出课程列表，教授将学生成绩输入系统。

**参与者：**教授

**前置条件：**教授已登录。

**基本流程：**教授登录系统，选择课堂，选择输入分数。系统给出课程学生名单，教授将

每个学生的成绩输入到系统中。

**异常：**

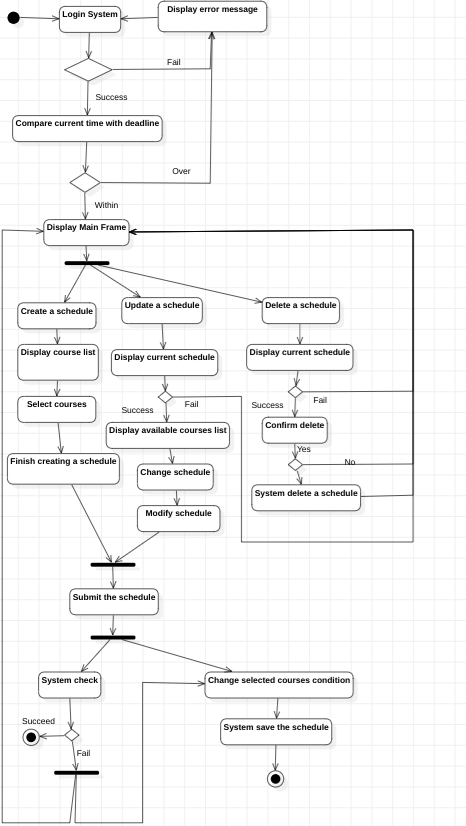
1.教授选择的课程不是自己的课堂。系统提示：请选择你本学期本人教授的课堂。

2.未到学期末。系统提示，还没有进行考试，无法输入成绩。

3.输入的成绩不在范围内。系统提示，请仔细核查，该成绩无效。

**后置条件：**教授选择输入成绩。

**3 注册课程用例的活动图**

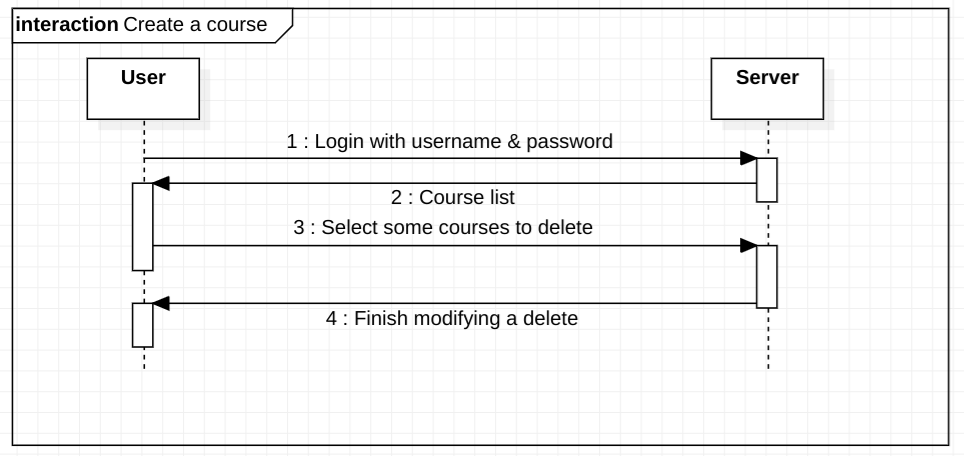


**4 注册课程用例的时序图：**

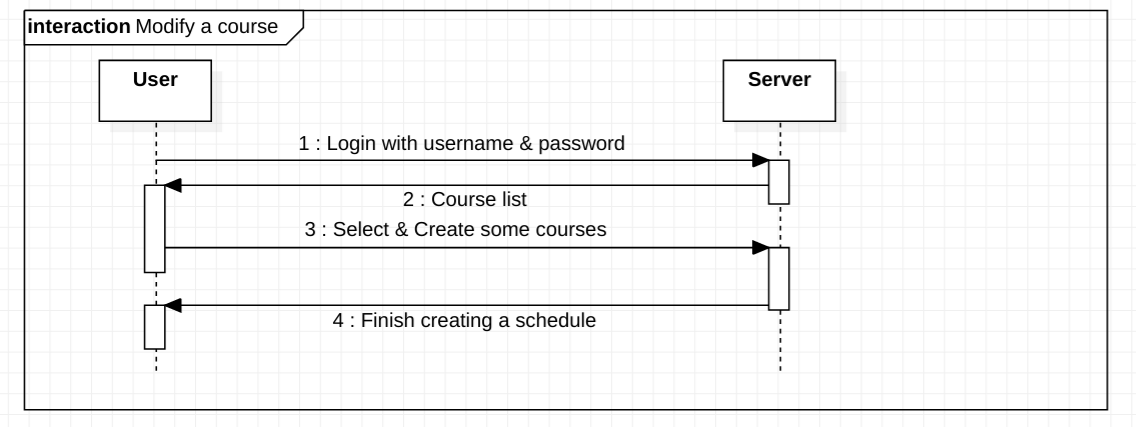
分别为创建课程、修改课程、删除课程等操作的时序图

**4.1 创建课程时序图**

用户使用账号密码登陆系统，服务端返回课程列表，用户选中部分课程删除，服务端返回删除成功

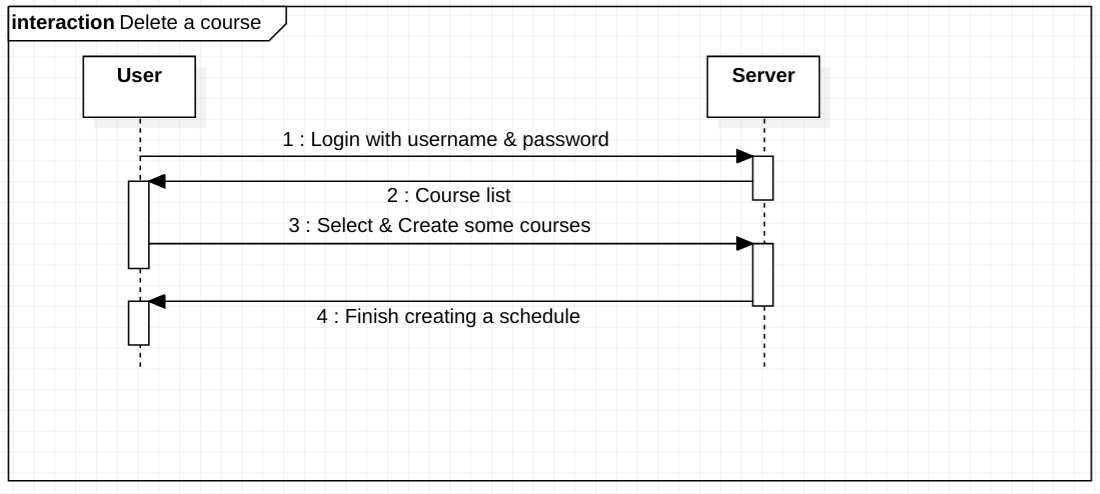
**4.2 修改课程时序图**

用户使用账号密码登陆系统，服务端返回课程列表，用户选中部分课程并执行创建，服务端返回创建成功



**4.3 删除课程时序图**

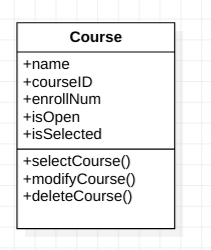
用户使用账号密码登陆系统，服务端返回课程列表，用户选中部分课程并执行修改，服务端返回修改成功

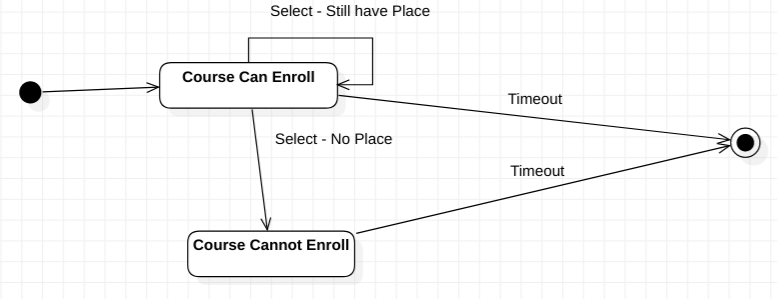


**5 2个核心类的设计类图、状态图：**

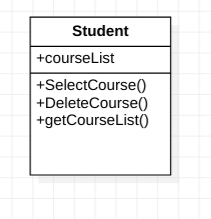
**5.1 课程类**

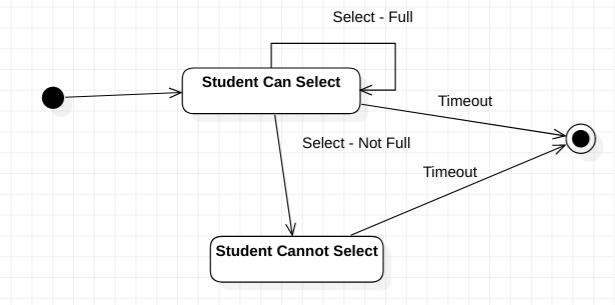
课程类为所有课程的父类，其存储了课程名称、课程唯一标识符、人数、开课状态、是否被选中等信息。同时提供课程操作相关接口



**5.2 课程类的状态图****5.3 学生类**

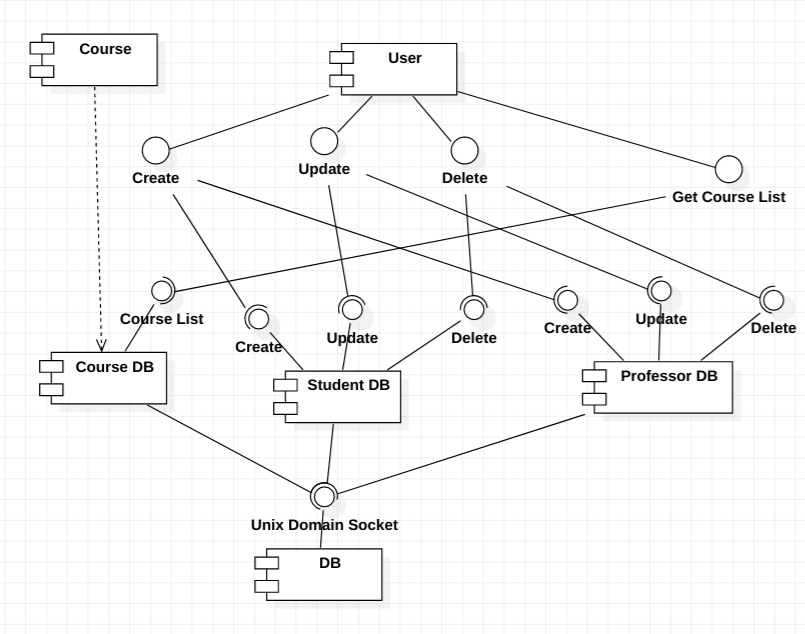
学生类为核心类之一，其属性中提供了已选课程列表，并暴露获取列表、选择课程、删除课程等接口



**5.4 学生类的状态图**

**6 系统组件图**

下图为系统的组件图。

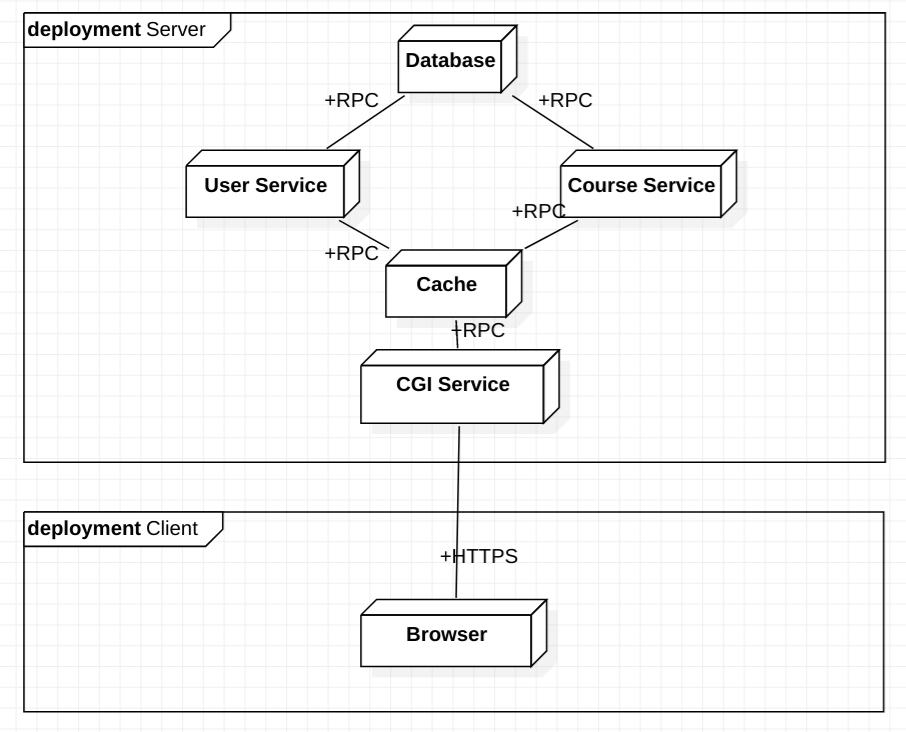


**7 系统部署图**

下图为系统的部署图。

其后端部分借由RPC完成各组件通信，使用DB持久化数据，最后借由CGI向外暴露HTTPS 接口。

其客户端借由HTTPS实现安全可靠的消息传输。直接使用前端页面来实现人机交互。



**8 学生类的生成代码**

此处生成的代码为Course类的C++ 代码的头文件声明。

class Course {

public:

std::string name;

std::string courseID;

int enrollNum;

bool isOpen;

bool isSelected;

void selectCourse();

void modifyCourse();

void deleteCourse();

};