“学生毕业管理系统”

设计规格说明书

版 本 号：V1.0

编 写 者：肖梦杰

审 核 者：严轶轩

批 准 者：赵鹏程

1 引言

1.1 编写目的

本设计规格说明书的撰写目的是为对“学生毕业管理系统”做出设计分析，文档面向小组内部成员，用以指导开发各个阶段的流程，明确软件设计、安排项目规划与进度、组织软件的开发与测试，以及日后对系统进行改进，为开发人员、维护人员及用户之间提供共同的协议以保证开发任务能够顺利进行。是项目开发的基础，对小组日后工作具有总领和指导的意义。

1.2 背景

项目名称：学生毕业管理系统
项目委托单位：西北大学软件学院付丽娜老师
项目开发单位：赵鹏程、严轶轩和肖梦杰开发小组
项目简介：本系统主要学生毕业要求达成度的计算；

（1）学生查询学业信息，收到预警通知；
（2）任课教师导入学生课程评价值；
（3）课程负责人审核学生课程评价值；
（4）专业负责人格式化培养方案，分析成绩数据；
（5）辅导员查看学生成绩数据和预警名单。

1.3 定义

Nodejs：是一个基于 Chrome V8 引擎的 JavaScript 运行环境。

Vue：是一套用于构建用户界面的渐进式框架；只关注视图层，采用自底向上增量开发的设计；通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。

Element：一套为开发者、设计师和产品经理准备的基于 Vue 2.0 的桌面端组件库。

Mock：让接口开发更简单高效，让接口的管理更具可读性、可维护性，让团队协作更合理。Mock 模板规则可轻松编写接口，这将大大提高定义接口的效率，并且无需为编写 Mock 数据烦恼，所有的数据都可以实时随机生成。生成的 Mock 数据可以直接用 ajax 请求使用，也可以通过服务器代理使用（不需要修改项目一行代码）。

Mysql：一个免费的功能较强的关系型数据库管理系统。

Python：一种跨平台的计算机程序设计语言，是一种面向对象的动态类型语言。

Django：一个开放源代码的 Web 应用框架，由 Python 写成。可以用于快速搭建高性能，优雅的网站！采用了 MVC 的框架模式。

Github： 一个面向开源及私有软件项目的托管平台.

1.4 参考资料

《管理信息系统分析与设计》 高等教育出版社 蔡淑琴著
《附件 1-“学生毕业管理系统”需求规格说明书》
《软件设计文档国家标准 GB8567》

2 总体设计

2.1 需求规定

详细需求说明请见《附件 1-“学生毕业管理系统”需求规格说明书》

2.2 运行环境

（1）软件环境：

操作系统：Windows 7/8/10

网络协议：TCP/IP

浏览器：Chrome

数据库：My SQL 5.7

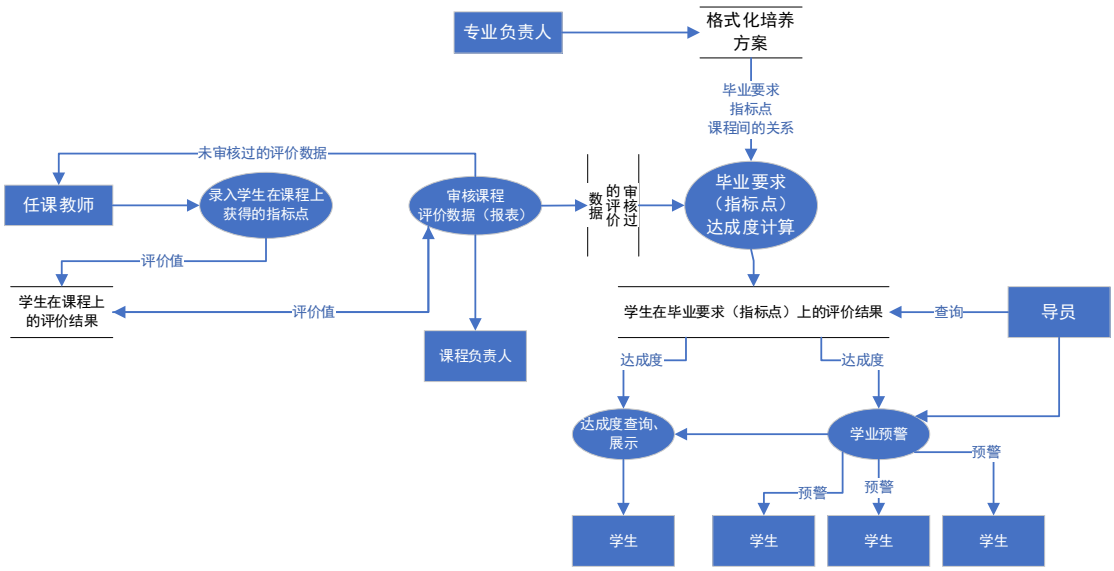
（2）硬件环境：

服务器 CPU：p42.0G 以上，内存：256M 以上

客户机 CPU：p42.0G 以上，内存：256M 以上

2.3 基本设计概念和处理流程

系统 DFD 模型：
描述：
图中矩形图表示图的数据源点，主要由用户实体构成；
图中椭圆表示数据的加工处理；
图中双横线之间组成部分是系统的数据存储模型。



2.4 结构

2.5 功能需求与程序的关系

2.6 人工处理过程

使用前后端分离技术，前端必须模拟假数据以满足前端开发需求。

2.7 尚未解决的问题

3 接口设计

3.1 用户接口

详细用户接口请见《附件 6-“学生毕业管理系统”接口文档》

3.2 外部接口

（1）软件接口

服务器程序可使用 Django 提供的对 MySQL 的接口，进行对数据库的所有访问；

服务器程序上可使用 MySQL 的对数据库的备份命令，以做到对数据的保存；文件的下载与上传。

（2）硬件接口

在输入方面，对于键盘，鼠标的输入。可用 Python 的标准输入/输出，对输入进行处理。

3.3 内部接口

前端验证；

封装 API。内部接口方面，各模块之间采用函数调用，参数传递，返回值的方式进行信息传递，具体参数的结构将在数据结构设计的内容中说明。接口传递的信息将是以数据结构封装了的数据，以参数传递或返回值的形式在各模块间传输。

4 运行设计

4.1 运行模块组合

（1）“学生毕业管理系统”的所有模块在服务器启动的时候完成所有模块的加载工作，随时等候用户的调用；

（2）不同的角色根据权限的不同调用不同的模块：

用户	模块
学生	查询学业信息、查看预警通知、修改个人信息
任课教师	所授课程信息、管理学生成绩、管理个人信息
课程负责人	负责课程信息、审核课程成绩、管理个人信息
专业负责人	格式化培养方案、成绩数据分析、管理个人信息
辅导员	分析学生数据、修改个人信息

用户有输入时，通过各模块的调用，读入并对输入进行格式化，服务器得到数据后返回信息，对信息进行处理后，产生相应的输出。

4.2 运行控制

运行控制将严格执照各模块间函数调用关系来实现。在各事务中心模块中，需对运行控制进行正确的判断，选择正确的运行控制路径。

4.3 运行时间

在需求分析中，对运行时间的要求为必须对做出的操作有较快的反应。网络硬件对运行的时间有较大的影响，所以将采用高速网络。其次是服务器的性能，这将影响对数据库的访问时间即操作时间的长短。硬件对本系统的速度的影响将大于软件的影响。

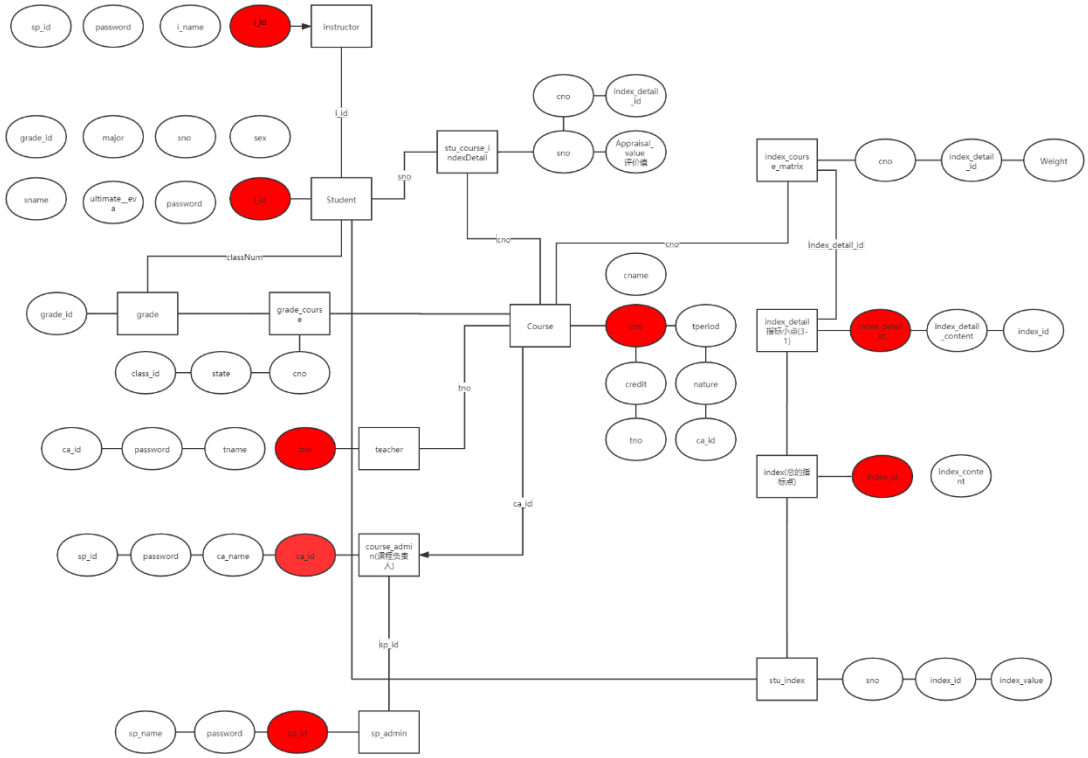
5 系统数据结构设计

5.1 逻辑结构设计要点

后台数据库的详细设计，包括每张表的列名、数据类型、关系模式即说明等。详细用户接口请见《附件 4-“学生毕业管理系统”数据字典》

5.2 物理结构设计要点

ER 图：



5.3 数据结构与程序的关系

系统的数据结构由标准数据库语言 SQL 生成。

服务器程序在对用户访问进行操作时需对数据库数据结构，也就是数据表进行查询和修改；格式化培养方案、审核课程成绩等过程中都需要对数据库中的所有表，进行联合查询，修改。

物理数据结构主要用于各模块之间函数的信息传递。接口传递的信息将是以数据结构封装了的数据，以参数传递或返回值的形式在各模块间传输。出错信息将送入显示模块中。

6 设计约束

从应用软件开发的角度考虑，应选择开发平台功能强，共享软件资源丰富，硬件驱动支持多的操作系统；

系统的设计约束：

（1）尽量减少 IO 以及网络的访问，将多次的调用整合在一次操作中完成，尽量减少 IO 资源的浪费；

（2）对于系统的配置文件，数据库字段修改，或者其他显示复杂逻辑修改；尽量采用增加的操作；而少采用 update 的操作；update 永远比 insert 成本大的很多；

（3）系统之间交互，pull 的效果往往比 push 来的稳定性高；选择只读 API，而不是读写 API，“写”部分尽量采用事件驱动或者消息驱动；

（5）往往内存中的复杂数据结构组装要优先于数据库的链接。

数据库设计约束：

（1）尽可能在数据模型上控制业务对象的约束关系，如果通过程序逻辑去保证完整性与一致性，会存在一定的风险；

（2）数据模型总的唯一性约束，比如学号，一定要在数据库层面得到控制；

（3）尽量少用存储过程，将复杂的业务逻辑抽离到上层应用中，也就是时候尽量使用程序中的数据结构完成复杂的关系运算，避免用存储过程或者复杂的 sql 语句，因为应用服务器的扩展以及优化的成本往往比 DB 服务器的成本小的多；

（4）sql 语句尽量不要依据业务逻辑以及动态拼接的 sql 字符串，而是采用预编译的方式，否则有 sql 注入的风险；

（5）如果主表与子表是一对一的关系，主键尽量相同。

外部交互设计规范：

（1）最好是 pull 对方的数据比较好，比对方 push 过来稳定性好；

（2）异步消息处理的时候，最好先落地到本地库再进行处理；这样避免消息的丢失，以及消息队列的堆积，导致消息系统挂掉；

（3）系统中只能有一种异常：处理中状态等待超时或者重试次数达到最大值。

7 系统出错处理设计

7.1 出错信息

能够对用户录入的各种数据和各种文件进行校验。

用户操作成功或失败后及时给出相应提示。

能够及时捕捉系统在运行时的错误信息，并给出相应的提示，系统应有一定的容错能力。

7.2 补救措施

所有的客户机及服务器都必须安装不间断电源以防止停电或电压不稳造成的数据丢失的损失。在断电情况下，客户机上将不会有太大的影响，主要是服务器上；在断电后恢复过程可采用 MySQL 的日志文件，对其进行 ROLLBACK 处理，对数据进行恢复。

在网络传输方面，可考虑建立一条成本较低的后备网络，以保证当主网络断路时数据的通信。在硬件方面要选择较可靠、稳定的服务器机种，保证系统运行时的可靠性。

7.3 系统维护设计

由于系统没有外加维护模块，因为维护工作比较简单，仅靠数据库的一些基本维护，维护方面主要是对数据库进行维护。可使用 MySQL 的数据库维护功能机制。要定期的为数据库进行备份，维护管理工作数据库死锁问题和维护数据库内数据的一致性。