|  |  |
| --- | --- |
| 项目编号 | 20191101 |
| 文档编号 | 02 |
| 密 级 | 内部 |

学生管理系统概要设计

**V1.0**

学号：18061615

姓名：李国宇

学院：卓越学院

学校：杭州电子科技大学

目录

[1 导言 4](#_Toc23676153)

[1.1 目的 4](#_Toc23676154)

[1.2 范围 4](#_Toc23676155)

[1.3 缩写说明 4](#_Toc23676156)

[1.4 术语定义 5](#_Toc23676157)

[1.5 引用标准 5](#_Toc23676158)

[1.6 参考资料 5](#_Toc23676159)

[1.7 版本更新信息 5](#_Toc23676160)

[2 系统分析 5](#_Toc23676161)

[3 界面设计 7](#_Toc23676162)

[3.1 管理端界面设计 7](#_Toc23676163)

[3.2 客户端的界面设计 8](#_Toc23676164)

[4 体系结构 9](#_Toc23676165)

[4.1 体系结构 9](#_Toc23676166)

[4.1.1 前后端分离体系结构 10](#_Toc23676167)

[4.1.2 系统体系结构 12](#_Toc23676168)

[4.2 系统运行环境 13](#_Toc23676169)

[4.2.1 网络结构图 13](#_Toc23676170)

[4.2.2 硬件环境 14](#_Toc23676171)

[4.2.3 软件环境 14](#_Toc23676172)

[5 数据模型 15](#_Toc23676173)

[5.1 数据库的概念结构模型设计 15](#_Toc23676174)

[5.2 数据库的逻辑结构模型设计 16](#_Toc23676175)

[5.3 数据库管理物理结构模型设计 18](#_Toc23676176)

[6 模块设计 21](#_Toc23676177)

[6.1 登录管理模块设计 23](#_Toc23676178)

[6.1.1 前端模块设计 23](#_Toc23676179)

[6.1.2 后端API设计 24](#_Toc23676180)

[6.2 功能框架界面设计 25](#_Toc23676181)

[6.2.1 前端模块设计 25](#_Toc23676182)

[6.2.2 后端API设计 26](#_Toc23676183)

[6.3 学生管理界面设计 26](#_Toc23676184)

[6.3.1 前端模块设计 26](#_Toc23676185)

[6.3.2 后端API设计 27](#_Toc23676186)

# 1 导言

## 1.1 目的

该文档的目的是描述网上招聘系统项目的概要设计，其主要内容包括：

* 系统功能简介
* 系统结构设计
* 系统接口设计
* 数据设计
* 模块设计
* 界面设计

本文档的预期读者是：

* 开发人员
* 项目管理人员
* 测试人员

## 1.2 范围

该文档是借助于当前系统的逻辑模型导出目标系统的逻辑模型，解决整个项目系统的 “做什么”的问题。在这里，对于开发技术并没有涉及，而主要是通过建立模型的方式来描 述用户的需求，为客户、用户、开发方等不同参与方提供一个交流的渠道。

## 1.3 缩写说明

* **ADMIN** Administrator(管理员)
* **USER** 用户
* **UML** Unified Modeling Language（统一建模语言）
* **SPA** Singo Page Applications（单页面应用）
* **MVVM** Model-View-ViewModel

## 1.4 术语定义

* **HTML** HyperText Markup Language(超文本标记语言)
* **PHP** PHP: Hypertext Preprocessor(超文本预处理器)
* **AJAX** Asynchronous Javascript And XML(异步 JavaScript 和 XML)
* **JS** JavaScript(直译式脚本语言)
* **SQL** Structured Query Language(结构化查询语言)

## 1.5 引用标准

* [1]缺省

## 1.6 参考资料

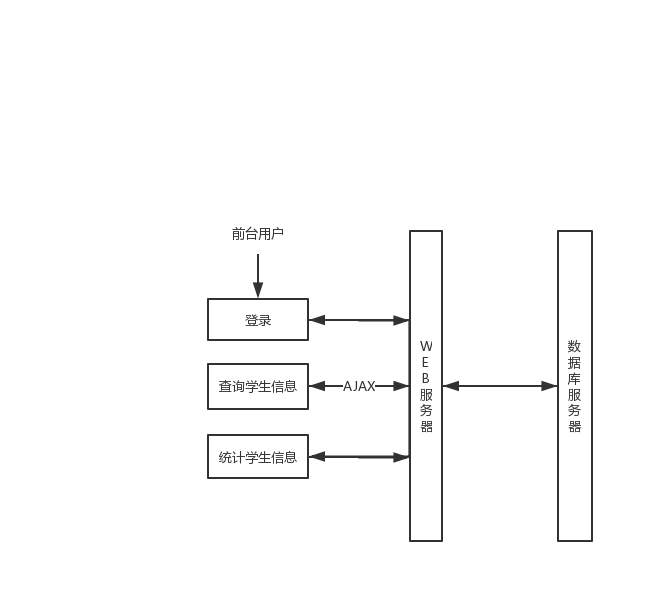
* [1]缺省

## 1.7 版本更新信息

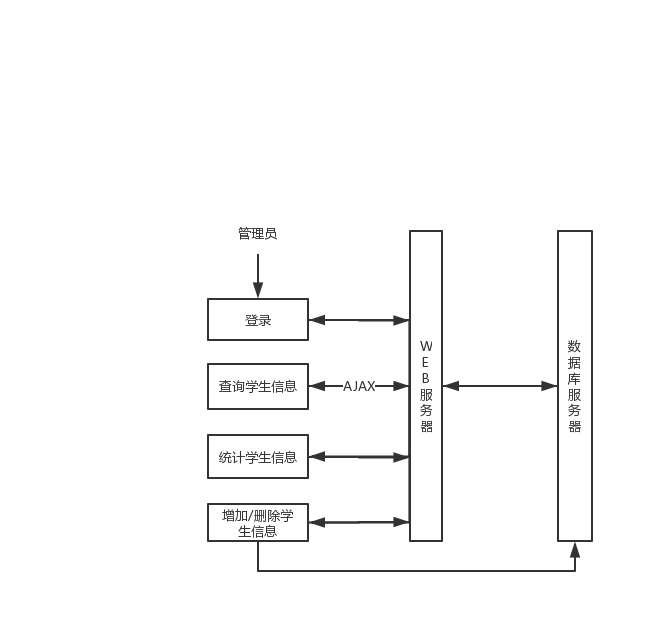
本文档的更新记录如表Ｂ－１所示。

| **修改编号** | **修改日期** | **修改后版本** | **修改位置** | **修改内容概述** |
| --- | --- | --- | --- | --- |
| 000 | 2910.11.12 | 0.1 | 全部 | 初始发布版本 |

# 2 系统分析

本系统可以实现网上的在线学生管理系统，前台用户通过登录系统，可以查询学生的个人信息、统计学生的考试信息。同时管理用户可以增加学生信息、删除学生信息、修改学生信息。极大的方便了学生信息的管理。系统包括管理端子系统和客户端子系统。  
管理端子系统功能包括增加学生信息、删除学生信息、修改学生信息、查询学生信息、统计学生成绩信息。客户端子系统功能包括查询学生信息、统计学生成绩信息等。图B-1和B-2为客户端和管理端的组成结构图  


图B-1：客户端子系统图示

  
图B-2：管理端子系统图示

# 3 界面设计

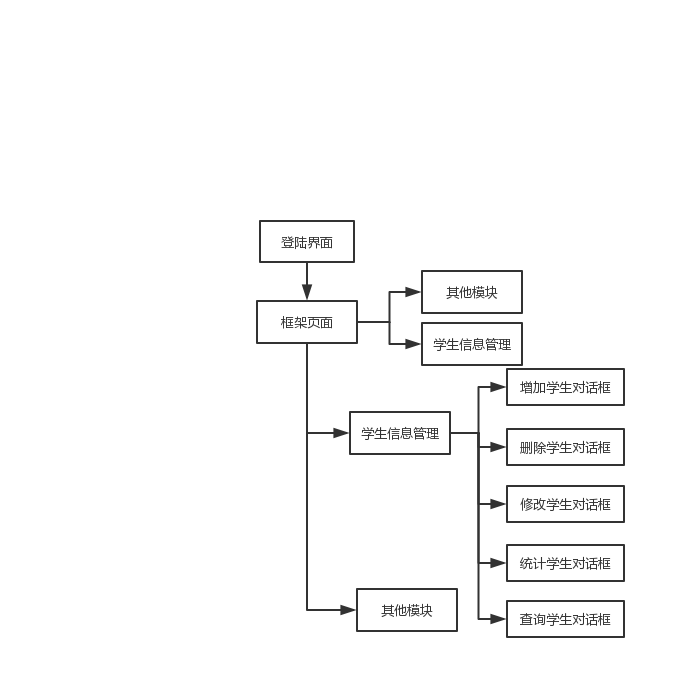
本系统的用户界面按功能分客户端和管理端。

## 3.1 管理端界面设计

管理端主要实现增加学生信息、删除学生信息、修改学生信息、查询学生信息、统计学生成绩信息等功能，主要界面设计如下：

* 登陆界面
  + 通过用户名和密码实现用户登录，并判断用户的权限
* 框架页面
  + 根据用户的权限，进入框架页面，并在框架中展示此用户相应可以操作的权限功能
* 学生管理界面
  + 主界面由一张表格和两个按钮、一个搜索框构成。表格内每个数据项也有对应操作的按钮。表格用于展现筛选到的学生信息。两个按钮分别对应添加、删除学生的功能
* 对话框页面
  + 各类功能的使用为了简洁性和高响应时间，均采取对话框格式进行展现。不必要的页面加载。

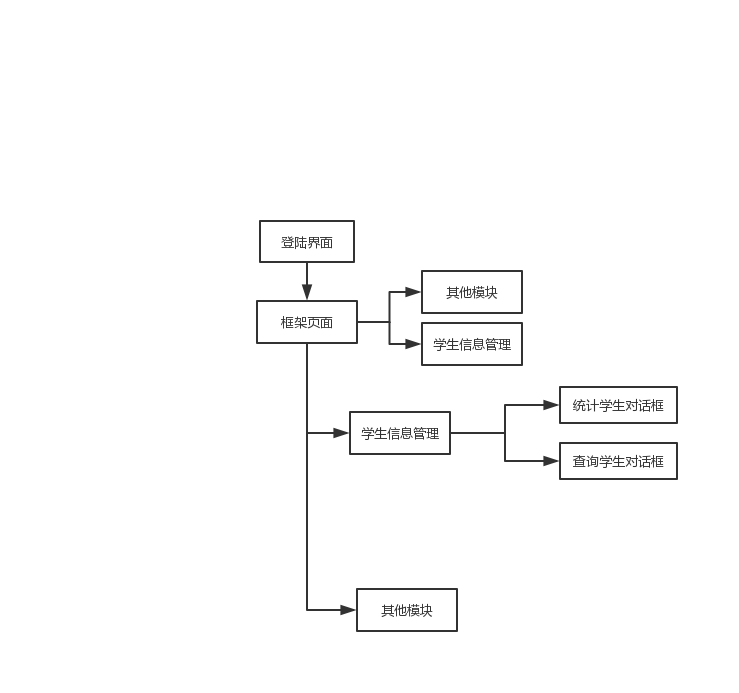
具体页面流如下图B-3所示



图B-3 管理端的页面流程

## 3.2 客户端的界面设计

客户端为前台用户进行学生信息查找和成绩统计的界面。无任何编辑权限。页面结构也与管理端界面类似。但删除了管理相关的功能入口。进入框架页面和管理界面时，不对类似业务逻辑进行展示。  
用户登录后，若为用户权限，同样进入管理界面，选择“学生管理”子菜单后，iframe加载学生管理选择页面。  
具体页面流如下图B-4所示



图B-4 客户端的页面流程

# 4 体系结构

系统的总体结构设计遵循如下原则：  
１）系统应具有良好的适应性：能适应用户对系统的软件环境、管理内容、模式和界面 的要求；  
２）系统应具有可靠性：采用成熟的技术方法和软件开发平台，以保证在以后的实际应 用中安全、可靠；  
３）系统应具有较好的安全性：应提高完善的安全机制和用户权限限制机制，确保数据 的受限访问；  
４）系统应具有良好的可维护性：系统应易于维护、安装；  
５）系统应具有良好的可扩展性：系统应适应未来信息化建设的要求，能方便得进行功 能扩展，以建立完善的信息集成管理体系。

由于系统仅需采取较简单的查询策略，为了方便开发和提高可扩展性，本系统采用前后端分离方法进行开发。

## 4.1 体系结构

随着时代的发展，网络应用的开发MVC光芒渐退，最新的前后端分离解决方案因其分工明确、适配多端、操作简便、易扩展而夺得人们的青睐。

### 4.1.1 前后端分离体系结构

前后端分离已成为互联网项目开发的业界标准使用方式，通过nginx+tomcat的方式（也可以中间加一个nodejs）有效的进行解耦，并且前后端分离会为以后的大型分布式架构、弹性计算架构、微服务架构、多端化服务（多种客户端，例如：浏览器，车载终端，安卓，IOS等等）打下坚实的基础。

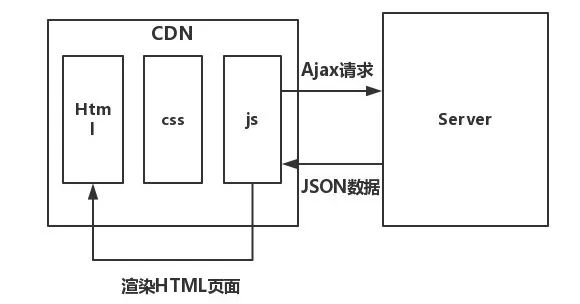
核心思想是前端HTML页面通过AJAX调用后端的RESTFUL API接口并使用JSON数据进行交互。

Web服务器：一般指像Nginx，Apache这类的服务器，他们一般只能解析静态资源；

应用服务器：一般指像Tomcat，Jetty，Resin这类的服务器可以解析动态资源也可以解析静态资源，但解析静态资源的能力没有web服务器好；

一般都是只有web服务器才能被外网访问，应用服务器只能内网访问。

具体信息交互方式如图B-5所示：

图B-5 前后端分离的交互方式

步骤如下：

1、浏览器请求，CDN返回HTML页面；

2、HTML中的JS代码以Ajax方式请求后台的Restful接口；

3、接口返回Json数据，页面解析Json数据，通过Dom操作渲染页面；后端提供的都是以JSON为数据格式的API接口供Native端使用，同样提供给WEB的也是JSON格式的API接口。

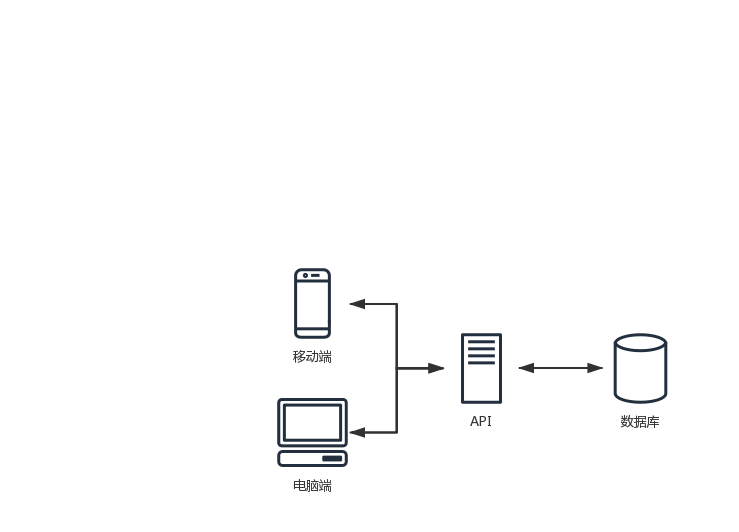
那么意味着WEB工作流程是：  
      1、打开web，加载基本资源，如CSS，JS等；  
      2、发起一个Ajax请求再到服务端请求数据，同时展示loading；  
      3、得到json格式的数据后再根据逻辑选择模板渲染出DOM字符串；  
      4、将DOM字符串插入页面中web view渲染出DOM结构；

这种方式的优点是很明显的。前端不会嵌入任何后台代码，前端专注于HTML、CSS、JS的开发，不依赖于后端。自己还能够模拟Json数据来渲染页面。发现Bug，也能迅速定位出是谁的问题。  
 模块化开发方面，可扩展性较高，仅需前端JS添加后端的API即可实现新功能的上线。

## 4.1.2 系统体系结构

根据系统分析结果，该系统从结构上应满足：

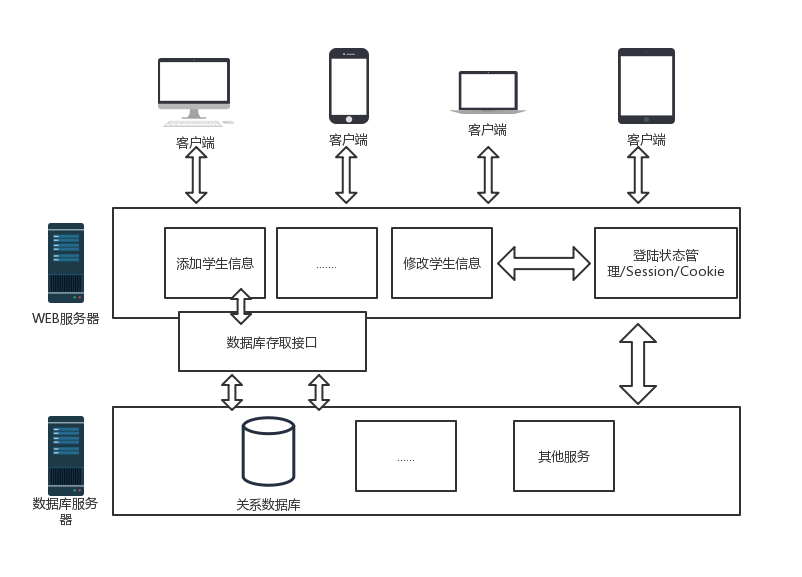
* 基于浏览器进行显示以方便用户使用
* 采用前后端分离的体系结构，分化各个功能组件；
* 采用mysqli技术与数据库通讯以便于数据库的转换；
* 采用 API+AJAX 进行数据的交互渲染  
  本系统的体系结构如图B-6，它基本遵循了前后端分离设计规范。



图B-6 系统的体系结构

其中：  
WEB服务器：用于API的对外服务，给API服务提供运行平台，接受AJAX请求，和数据库动态的交互以JSON返回需要的数据。  
客户端：系统对移动端有优化，可以是任何移动设备接入，接入后通过AJAX向服务端请求数据，动态更新DOM，实现数据交互。

体系结构的具体拓扑图示如图B-7



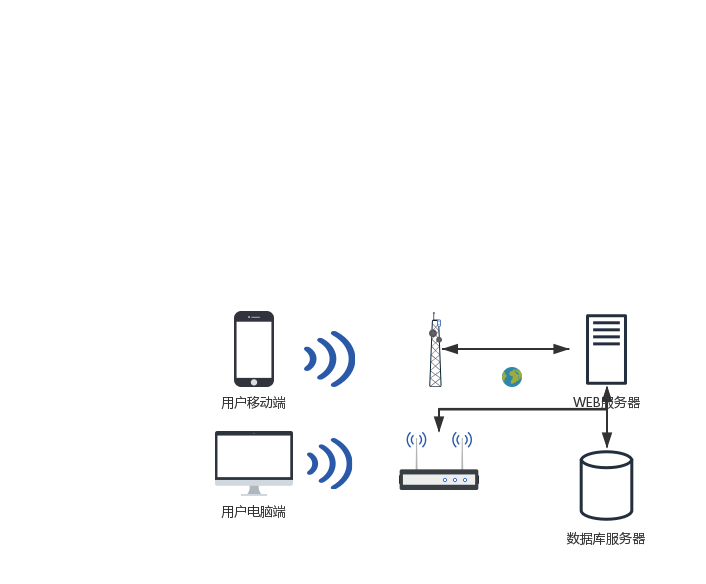
图B-7 结构拓扑图

## 4.2 系统运行环境

系统运行的网络结构图、硬件软件环境图如下。

### 4.2.1 网络结构图

本系统的网络拓构图如图B-8



图B-8 系统的网络拓扑图

### 4.2.2 硬件环境

本系统的硬件环境如下：

* 客户机：普通PC/移动设备
  + CPU：1.8GHz及以上
  + 内存：256MB及以上
  + 分辨率：推荐2.7k及以上
  + 系统：Android/Linux/Mac OS/IOS/Windows
* WEB服务器
  + CPU：1.8GHz以上
  + 内存：256MB以上
  + 系统：Linux/Windows
  + 网卡：千兆
* 数据库服务器
  + CPU：1.8GHz以上
  + 内存：256MB以上
  + 系统：Linux/Windows

### 4.2.3 软件环境

本系统在以下环境测试通过

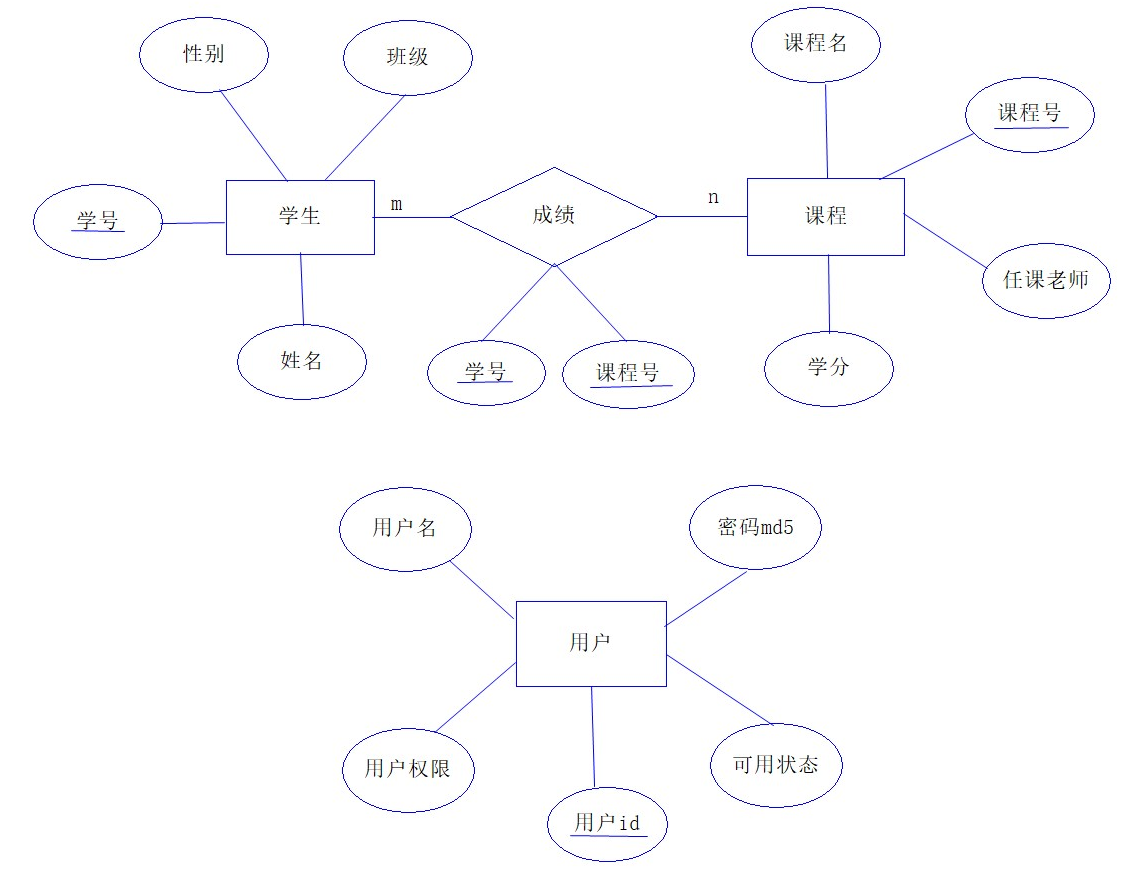
* 操作系统：Linux CentOS 7
* 网页服务器：Apache
* 数据库服务器：MySQL
* PHP版本：7.1
* 浏览器：Chrome/IE6+/FireFox/Opera/Edge  
  （1）数据库及操作系统: 对于核心数据库来说，选择一个合适的数据库系统对我们的系统运行是很重要的，选择 数据库的关键因素是要考虑预计会有多少人同时访问数据库；正常工作时间的级别；用来访 问数据库的应用程序的类型；运行数据库的服务器的硬件和操作系统类型；以及管理人员的 专业技术水平。目前市场上适用于中小型企业的数据库产品有 MySQL 、Microsoft SQL Server 系列、 Oracle 系列。所有这些产品都基于 SQL 语言。同时，它们还拥有精密复杂的安全控制以适应不同的商业需要。服务器操作系统使用 CentOS7 系统。 考虑到价格因素、易用性，我们使用 MySQL 作为系统后台数据库系统，服务器操作系统采用 CentOS7。  
  （2）WEB 服务软件： 目前的 WEB 服务器软件有很多种，成熟而且稳定有 Apache、Nginx、Tomecat和 Microsoft 的 IIS ，它们也是占据着 Web 服务器市场最大的份额。自创建以来，Apache已经成长迅速的普及和接受，成为较流行的Web服务器应用程序。要衡量它的成功，在2010年，它曾接近55％，在世界上的所有网站。它也是先进台Web服务器交叉标记就在一年前的100万个网站。大多数使用的是Apache Web服务器运行在UNIX，Linux或类似的操作系统。作为一个开源软件，它保持，开发和世界各地的许多开发者加入，保持不断的更新和发展的虚拟主机需求的变化。它有许多功能，导致其压倒性的普及。支持的编程语言和接口，包括Python和PHP，Perl，TCL等身份验证模块，如mod\_access，mod\_auth和mod\_digest。它丰富的功能列表包括支持SSL和TLS，代理，URL重写，自定义日志文件的创建，支持过滤和得多。  
  对于虚拟专用主机，它是好的网络托管解决方案becaise只有一个实例支持和承载了许多的网站，使其符合成本效益。它可以让你配置和自定义的错误信息，建立数据库管理系统为基础的身份验证数据库等。因此选择Apache作为 WEB 服务器。

# 5 数据模型

本系统的数据模型主要是进行数据库的设计。

# 5.1 数据库的概念结构模型设计

概念设计以反映现实世界中的实体、属性和它们之间的关系等的原始数据形式，建立数 据库的每一幅用户视图。图B-9是系统 E-R 图。



图B-9 系统的E-R图

## 5.2 数据库的逻辑结构模型设计

数据库的逻辑设计是将各局部的 E-R 图进行分解、合并后重新组织起来形成数据库全 局逻辑结构，包括所确定的关键字和属性、重新确定的记录结构、所建立的各个数据之间的 相互关系。根据本系统需求分析,系统的数据库包括了学生信息、课程信息、成绩信息、用户管理以及需要的基本数据字典等部分。

学生信息包括的库表：

* Sno-学生学号
* Ssex-学生性别
* Clno-学生班级
* Sname-学生姓名
* Sage-年龄
* CreateTime-创建时间
* CreateBy-创建人

课程信息包括的库表：

* Cno-课程号
* Cname-课程名称
* teacher-老师名称
* credir-课程学分
* CreateTime-创建时间
* CreateBy-创建人

成绩信息包括的库表：

* Sno-学生学号
* Cno-课程号
* Score-课程得分

登录信息包括的库表：

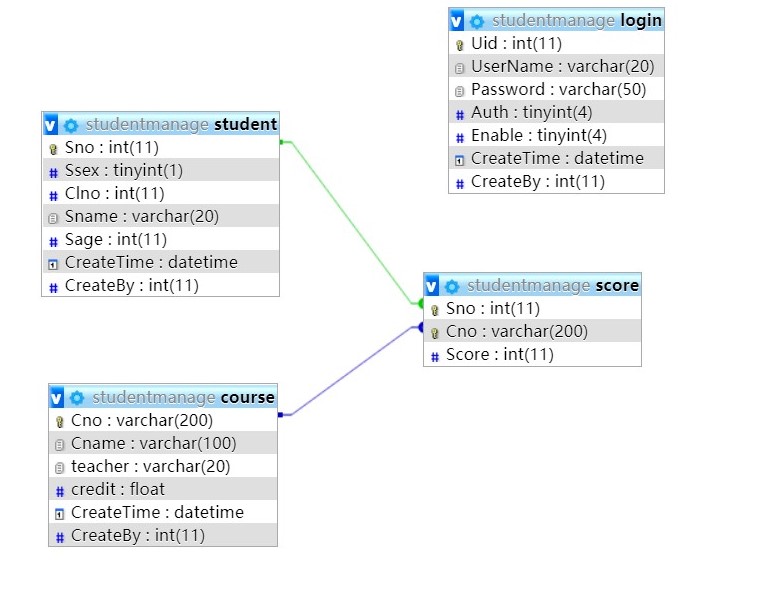
* Uid-用户ID
* UserName-用户名
* Password-密码md5
* Auth-用户权限
* Enable-用户使能状态
* CreateTime-创建时间
* CreateBy-创建人

表B-2是对学生信息表的设计，其他库表的设计详见数据表设计文件

表B-2：学生信息表（student）

| **字段名** | **字段代码** | **字段类型** | **主键** | **可否为空** |
| --- | --- | --- | --- | --- |
| 学生学号 | Sno | int(11) | Y | N |
| 学生性别 | Ssex | tinyint(1) | N | N |
| 所在班级 | Clno | int(11) | N | N |
| 学生姓名 | Sname | varchar(20) | N | N |
| 学生年龄 | Sage | int(11) | N | Y |
| 创建时间 | CreateTime | datetime | N | Y |
| 创建人 | CreateBy | int(11) | N | Y |

在确定了各个表主键字段的基础上,依据表与表相关字段之间的联系建立了各表之间的 关系，如图B-10所示。

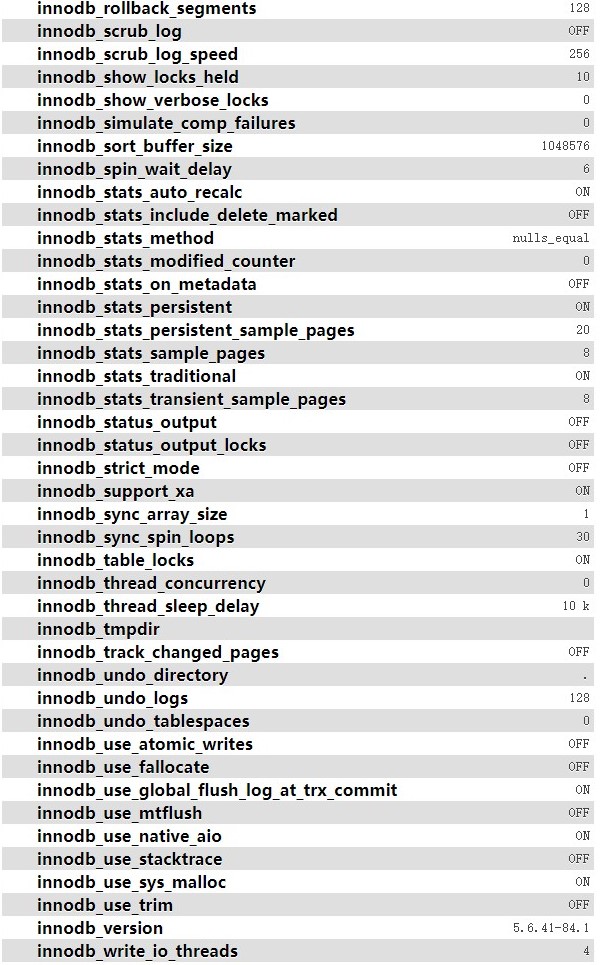
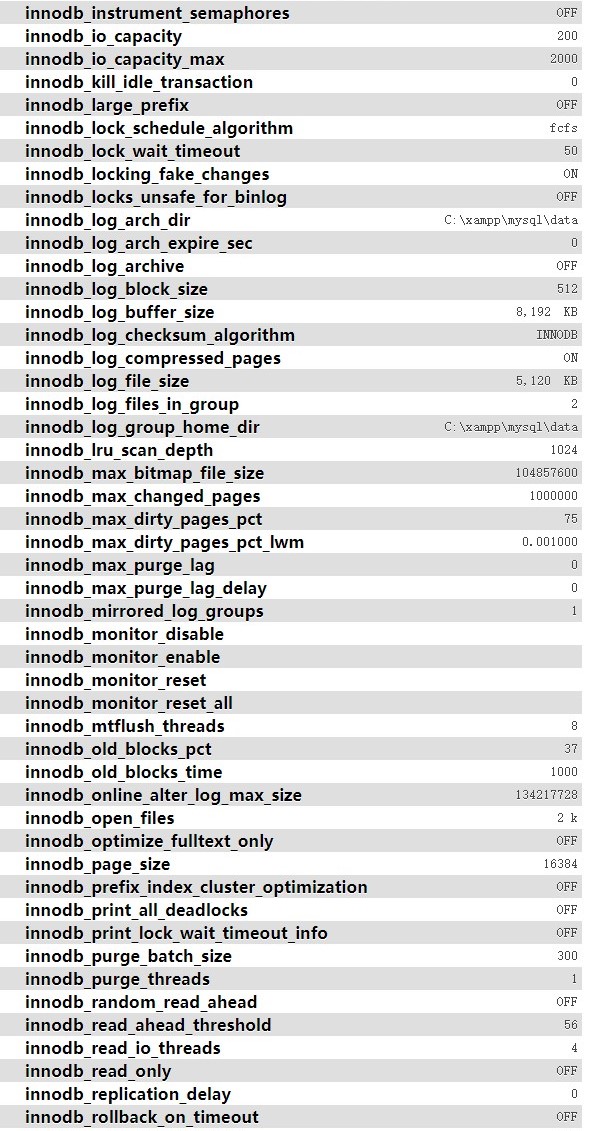
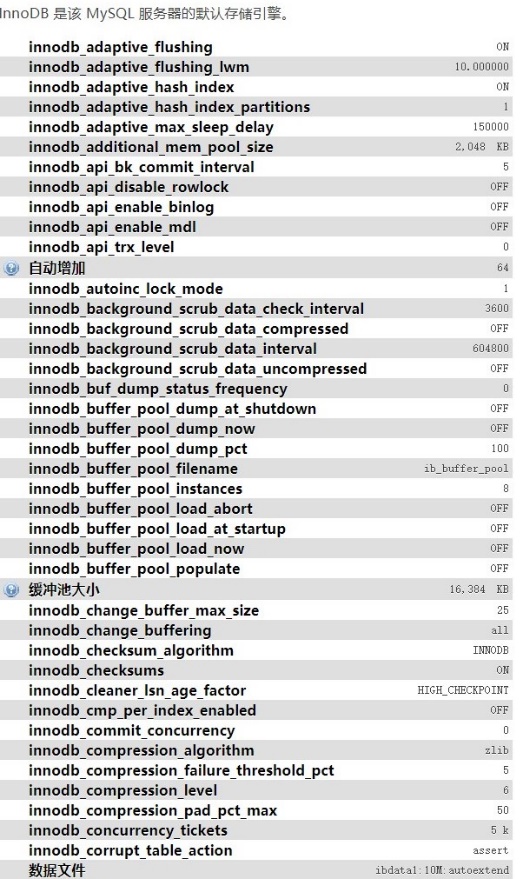
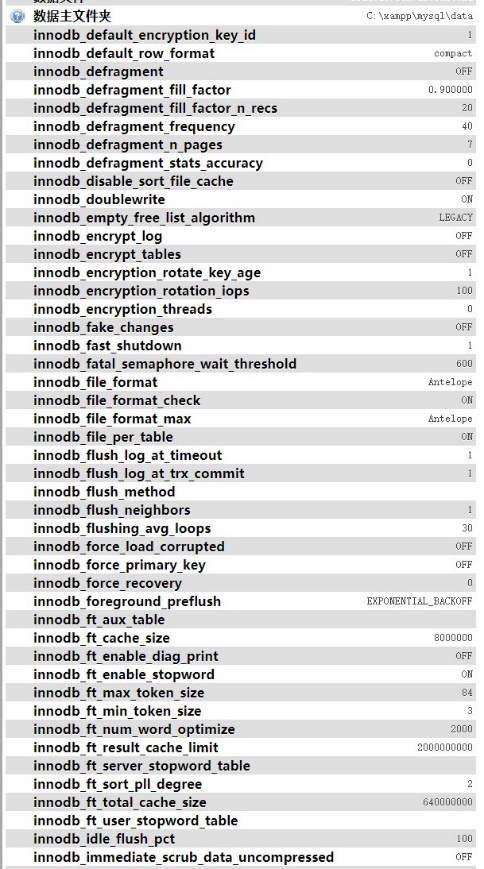


图B-10 系统数据库表关系图

## 5.3 数据库管理物理结构模型设计

信息存储结构的设计在系统的设计中至关重要，要考虑到数据冗余、系统执行效率、信 息控制以及维护等方面的要求。信息的管理离不开数据库的支持，我们采用MySql数据库管理系统。

数据库的物理设计主要是对数据在内存中的安排，包括对索引区、缓冲区的设计；对使用的外存设备及外存空间的组织，包括索引区、数据块的组织与划分；设置访问数据的方式方法。需在非系统卷（操作系统所在卷以外的其他卷）上安装MySQL程序及数据库文件。内存是影响MySQL系统性能的一个重要因素，应在MySQL数 据库安装后进行内存选项(Memory)设置，最大配置值为 2GB。 为了确定MySQL系统最适宜的内存需求，可以从总的物理内存中减去MySQL需要的内存（120M）以及其它一些内存需求后综合确定，理想的情况是给MySQL分配尽可能多的内存，而不产生页面调度。设置服务器的虚拟内存为2G。  
对Mysql变量配置如图B-11所示



图B-11 MySQL变量配置图

1、通过php mysqli扩展建立对数据库的连接

$this->db=new mysqli($this->server\_name,$this->db\_username,$this->db\_pwd,$this->db\_name);

2、打开mysqli创立的数据库对象

function open(){

$this->db=new mysqli($this->server\_name,$this->db\_username,$this->db\_pwd,$this->db\_name);

if (($this->db)->connect\_error) {

return false;

}

return true;

}

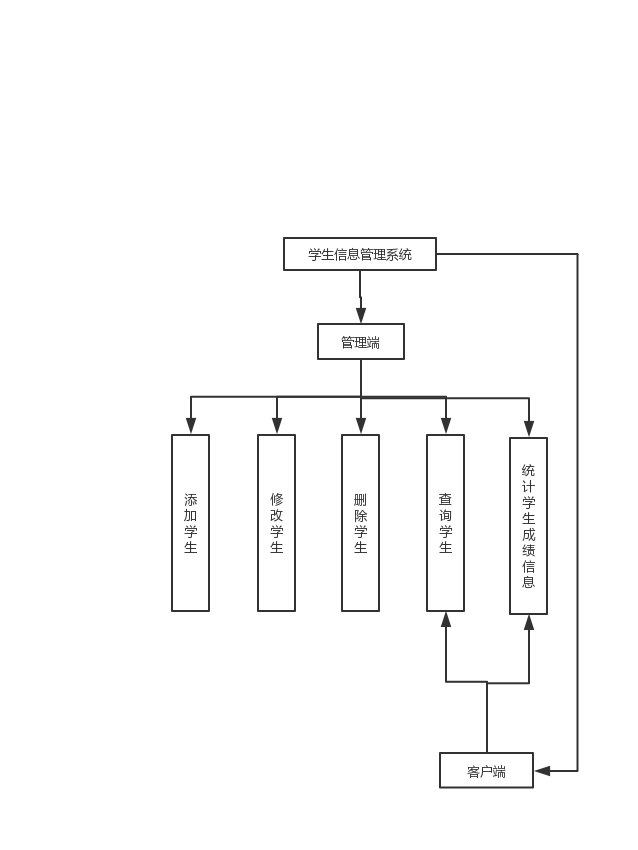
3、用后关闭，释放数据库资源

($this->db)->close();

本项目中后台数据库采用MySQL数据库系统。数据库各库表的脚本代码见数据库脚本文件。

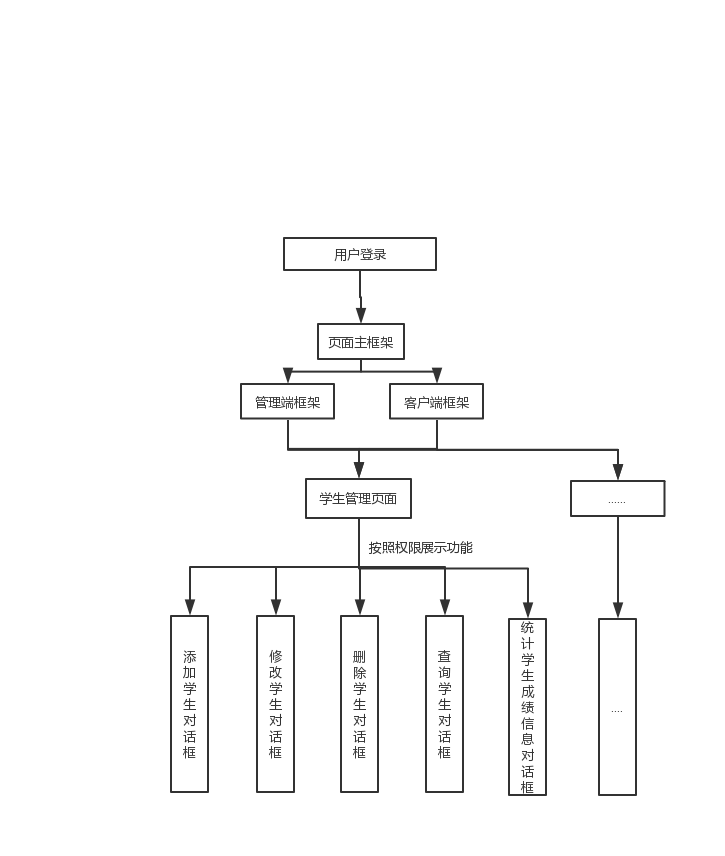
# 6 模块设计

按照功能分解，本系统分为客户端管理系统和管理端系统。根据页面流的设计，管理端系统又分为添加学生、删除学生、修改学生、统计学生、查询学生5个模块 如图B-12。



图B-12 模块设计

各部分关系如图B-13



图B-13 模块各部分关系

不同用户运用相同功能登录接口，接口都以 WEB 页面的形式提供。通统一的页面，登陆后通过判断权限展示不同的功能模块，前台用户和管理员可以从事各自的活动。 以下将分小节对各个部分进行设计。

## 6.1 登录管理模块设计

登录管理模块负责用户的登录。管理员和前台用户都是通过登录界面进入网页主框架的，用户输入用户名和密码进入界面首页，提供了进入功能面板的接口，并根据用户的权限在页中列出相应的操作功能。

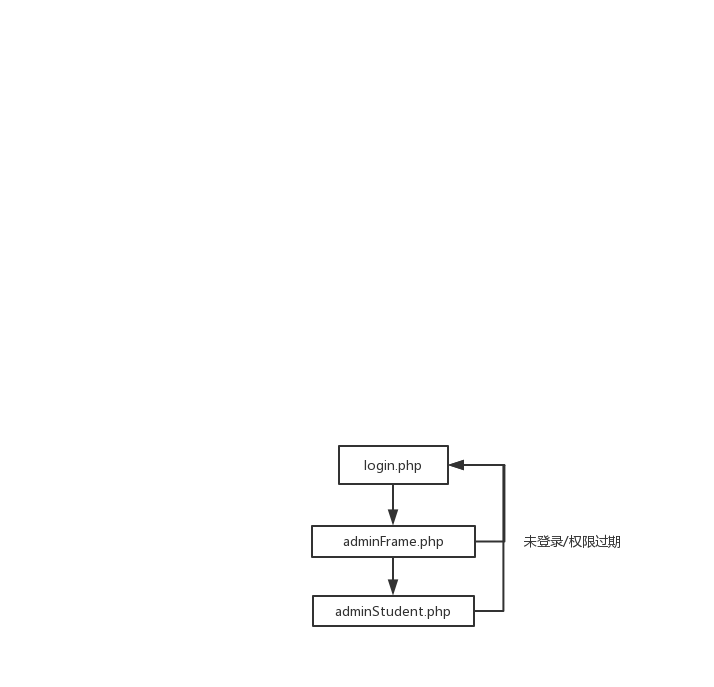
### 6.1.1 前端模块设计

根据上述功能介绍，总结出登陆界面如表B-3所示

表B-3 登录管理的页面设计

| **界面** | **php** | **功能描述** |
| --- | --- | --- |
| 登陆界面 | login.php | 登录的主页面 |

根据界面流的设计可以确定各个界面的访问入口，以及界面之间切换关系，页面的流程图如图B-14所示.



图B-14 登陆管理的页面流程

### 6.1.2 后端API设计

后端API接收前台回传的用户名和密码（经过前端md5加密）后的信息，在数据库中进行信息比对，传回登陆结果，为网页创建Session等权限标识。请求接口如下表B-4所示。

表B-4 请求接口示意

| **操作** | **内容** |
| --- | --- |
| 请求方式 | POST |
| 请求地址 | ./modules/loginAuth.php |
| 请求参数 | UserName(用户名)、Password(密码md5) |
| 返回格式 | JSON |
| 请求示例 | {UserName:“admin”,Password:“21232f297a57a5a743894a0e4a801fc3”} |
| 返回示例 | {“code”:200,“msg”:“登陆成功”,“data”:{“UserName”:admin,“Auth”:0}} |
| 注意事项 | 无 |

## 6.2 功能框架界面设计

登陆框架界面在登陆后展现，登陆后框架会依照用户权限展示相应的功能框架，采取统一的管理。这样的优势是代码复用高，前台展示较为美观，无需单独的管理页面，且切换功能时无需整个页面进行刷新

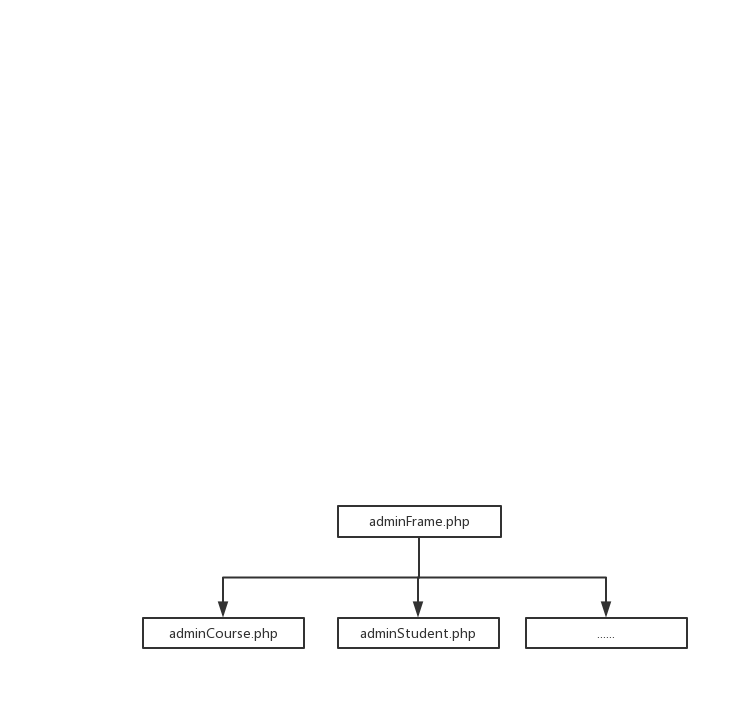
### 6.2.1 前端模块设计

根据上述功能介绍，总结出登陆界面如表B-5所示

表B-5 功能框架的页面设计

| **界面** | **php** | **功能描述** |
| --- | --- | --- |
| 框架界面 | adminFrame.php | 框架主页面 |
| 登陆界面 | login.php | 登录的主页面 |
| 学生信息管理页面 | adminStudent.php | 学生信息管理主页面 |
| 课程信息管理页面 | adminCourse.php | 课程信息管理主页面 |
| 其他功能页面 | … | 后续添加的功能 |

根据界面流的设计可以确定各个界面的访问入口，以及界面之间切换关系，页面的流程图如图B-15所示.



图B-15 框架页面的页面流程

### 6.2.2 后端API设计

后端无API请求。

## 6.3 学生管理界面设计

学生管理界面在功能框架界面内嵌显示，为学生管理的主要功能，运用了单一页面的思想，所有的功能都在一个页面中完成。当所需的功能被点击时，将弹出相应的对话框进行操作。对话框由前台JS负责渲染显示，提交后AJAX请求到服务器处理，回传数据后由JS进行动态更新。

### 6.3.1 前端模块设计

根据上述功能介绍，总结出登陆界面如表B-6所示

表B-6 学生管理页面设计

| **界面** | **php** | **功能描述** |
| --- | --- | --- |
| 学生管理主界面 | adminStudent.php | 学生管理主界面 |

学生管理页面关系图如图B-16所示



图B-16 学生管理页面的页面流程

### 6.3.2 后端API设计

#### 添加学生API设计

表B-7 添加学生API请求接口示意

| **操作** | **内容** |
| --- | --- |
| 请求方式 | POST |
| 请求地址 | ./modules/addStudent.php |
| 请求参数 | Sno(学号)、Ssex(性别)、Clno(班级号)、Sname(姓名) |
| 返回格式 | JSON |
| 请求示例 | {Sno:18992019,Ssex:0,Clno:18994111,Sname:“张三”} |
| 返回示例 | {“code”:200,“msg”:“学生张三插入成功”,“data”:{}} |
| 注意事项 | 学号不应与已有的重复 |

#### 修改学生API设计

表B-8 修改学生API请求接口示意

| **操作** | **内容** |
| --- | --- |
| 请求方式 | POST |
| 请求地址 | ./modules/updateStudent.php |
| 请求参数 | Sno(学号)、Ssex(性别)、Clno(班级号)、Sname(姓名) |
| 返回格式 | JSON |
| 请求示例 | {Sno:18992019,Ssex:0,Clno:18994111,Sname:“李四”} |
| 返回示例 | {“code”:200,“msg”:“学生张三插入成功”,“data”:{}} |
| 注意事项 | 学号不得修改（前台、后台均已限制） |

#### 删除学生API设计

表B-9 删除学生API请求接口示意

| **操作** | **内容** |
| --- | --- |
| 请求方式 | POST |
| 请求地址 | ./modules/deleteStudent.php |
| 请求参数 | Sno(学号) |
| 返回格式 | JSON |
| 请求示例 | {Sno:18992019} |
| 返回示例 | {“code”:200,“msg”:“学生18992019插入成功”,“data”:{}} |
| 注意事项 | 删除的同时会清楚所有该学生的成绩信息 |

#### 统计学生成绩信息API设计

表B-10 统计学生成绩信息API请求接口示意

| **操作** | **内容** |
| --- | --- |
| 请求方式 | POST |
| 请求地址 | ./modules/getScore.php |
| 请求参数 | Sno(学号) |
| 返回格式 | JSON |
| 请求示例 | {Sno:18992019} |
| 返回示例 | {  “code”:200,  “msg”:“成功”,  “data”:{  “student”:{  “Sno”:“18992019”,  “Sname”:“李四”  },  “basic”:{  “ave”:60,  “highest\_C”:“概率论与数理统计”,  “highest\_S”:“90”,  “lowest\_C”:“跨文化交际”,  “lowest\_S”:“30”,  “pass\_rate”:50  },  “content”:[  {“Cname”:“概率论与数理统计”,“Score”:“90”},  {“Cname”:“跨文化交际”,“Score”:“30”}  ]  }  } |
| 注意事项 | student内包含学生个人信息，basic类记录了学生的统计信息，包括平均分ave，通过率pass\_rate，和最高分最低分的课程名称和分数。content类是一个数组，里面返回学生成绩详细信息。 |

#### 查询学生API设计

表B-11 查询学生API请求接口示意

| **操作** | **内容** |
| --- | --- |
| 请求方式 | POST |
| 请求地址 | ./modules/searchStudent.php |
| 请求参数 | keywords(关键字),p(返回个数 默认50) |
| 返回格式 | JSON |
| 请求示例 | {keywords:“920”} |
| 返回示例 | {“code”:200,“msg”:“成功”,“data”:[[“18992019”,“0”,“18994111”,“李四”,20,“2019-10-31 13:53:57”,“1”]]} |
| 注意事项 | data是一个数组，内部存放所有符合结果的学生信息，学生信息也是数组表示，分别对应学号、性别、班级、姓名、年龄、创建时间、创建人。这里仅给出一个学生的示例 |