

实验报告

**学 号： 1804280423**

**姓 名： 帅 小 伟**

**班 级： 物联网1804**

**指导老师： 聂刚**

**报告日期：2020 年 12月20日**

实验题目03：主要实现增删改查四个功能和分页这个功能，以下将这几个功能融入到一个系统中便于突出这几个功能，实训内容如下：

**实训内容**

1. **设计目的**

希望通过项目实践，熟悉巩固所学知识，以使自身达到熟练掌握java、jsp基本结构以及java语言的基本知识和技能，基本掌握面向对象程序开发的基本思路和方法，了解数据库开发基本知识，为今后的进一步深入学习，奠定基础。

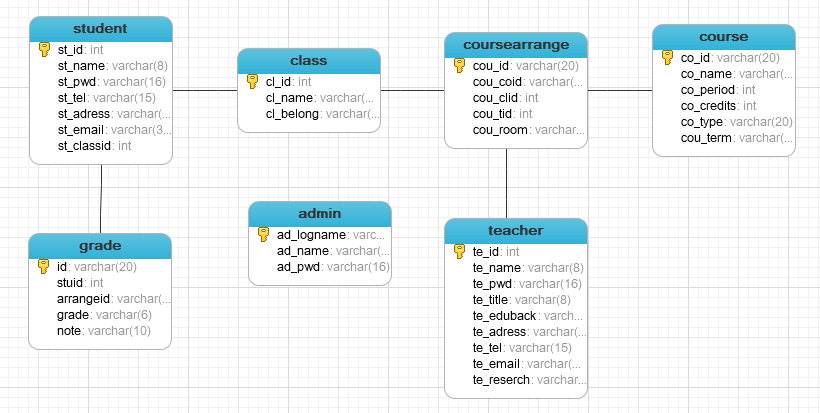
1. **功能分析**

学生信息管理系统的开发目的是为了实现学生个人信息、班级信息、老师个人信息、课程安排、课程信息、成绩管理、班级信息管理。本系统需要完成的功能有:班级信息增删改查、学生信息曾删改查；老师、管理员、学生登录系统执行不同模块访问；老师录入成绩、安排课程；学生查看/修改个人信息、查看成绩；退出登录。

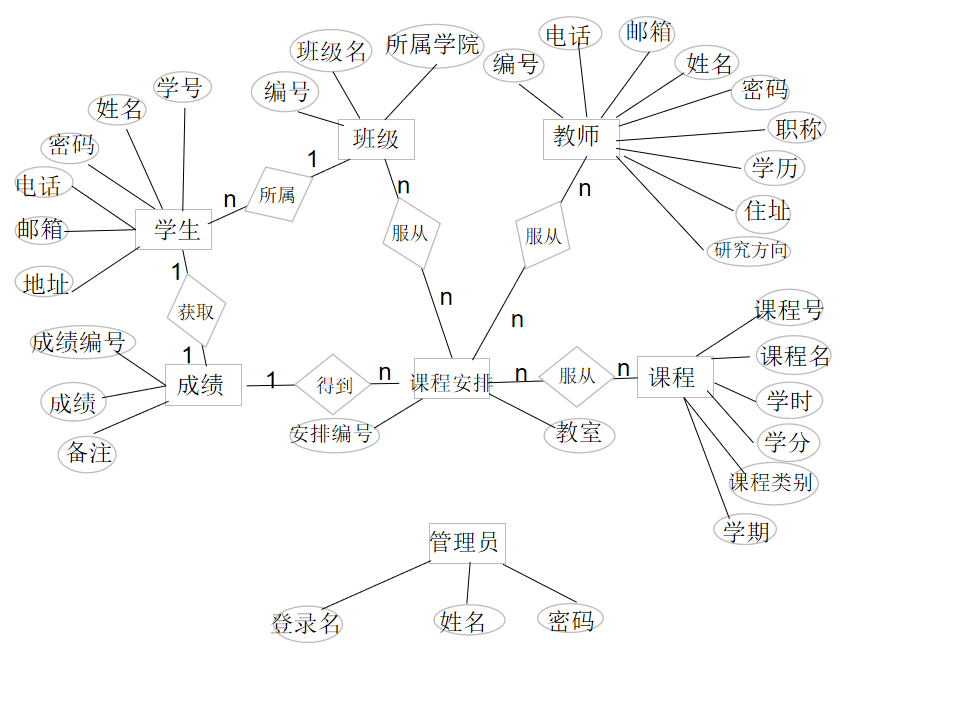
#### 设计

##### 数据表设计（ER图及三线图）

* 1. 数据库设计范式约束数据项



* 1. 数据库ER图：



1.3 数据库三线图：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 管理员表 | | | | |
| 列名 | 数据类型 | 是否为空 | 约束条件 | 中文名称 |
| ad\_logname | varchar | 否 | 主键 | 登录名 |
| ad\_name | varchar | 否 |  | 姓名 |
| ad\_pwd | varchar | 否 |  | 密码 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学生表 | | | | |
| 列名 | 数据类型 | 是否为空 | 约束条件 | 中文名称 |
| st\_id | int | 否 | 主键 | 学号 |
| st\_name | varchar | 否 |  | 姓名 |
| st\_pwd | varchar | 否 |  | 密码 |
| st\_tel | varchar | 是 |  | 电话 |
| st\_adress | varchar | 是 |  | 住址 |
| st\_email | varchar | 是 |  | 邮箱 |
| st\_classid | int | 否 | 外键 | 班级 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 班级表 | | | | |
| 列名 | 数据类型 | 是否为空 | 约束条件 | 中文名称 |
| cl\_id | int | 否 | 主键 | 班级编号 |
| cl\_name | varchar | 否 |  | 班级名 |
| cl\_belong | varchar | 是 |  | 所属学院 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 教师表 | | | | |
| 列名 | 数据类型 | 是否为空 | 约束条件 | 中文名称 |
| te\_id | int | 否 | 主键 | 教师编号 |
| te\_name | varchar | 否 |  | 姓名 |
| te\_pwd | varchar | 否 |  | 密码 |
| te\_title | varchar | 是 |  | 职称 |
| te\_eduback | varchar | 是 |  | 学历 |
| te\_adress | varchar | 是 |  | 住址 |
| te\_tel | varchar | 是 |  | 电话 |
| te\_email | varchar | 是 |  | 邮箱 |
| te\_reserch | varchar | 是 |  | 研究方向 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 课程表 | | | | |
| 列名 | 数据类型 | 是否为空 | 约束条件 | 中文名称 |
| co\_id | varchar | 否 | 主键 | 课程编号 |
| co\_name | varchar | 否 |  | 课程名 |
| co\_period | int | 否 |  | 学时 |
| co\_credits | int | 否 |  | 学分 |
| co\_type | varchar | 否 |  | 课程类别 |
| cou\_term | varchar | 否 |  | 开课学期 |

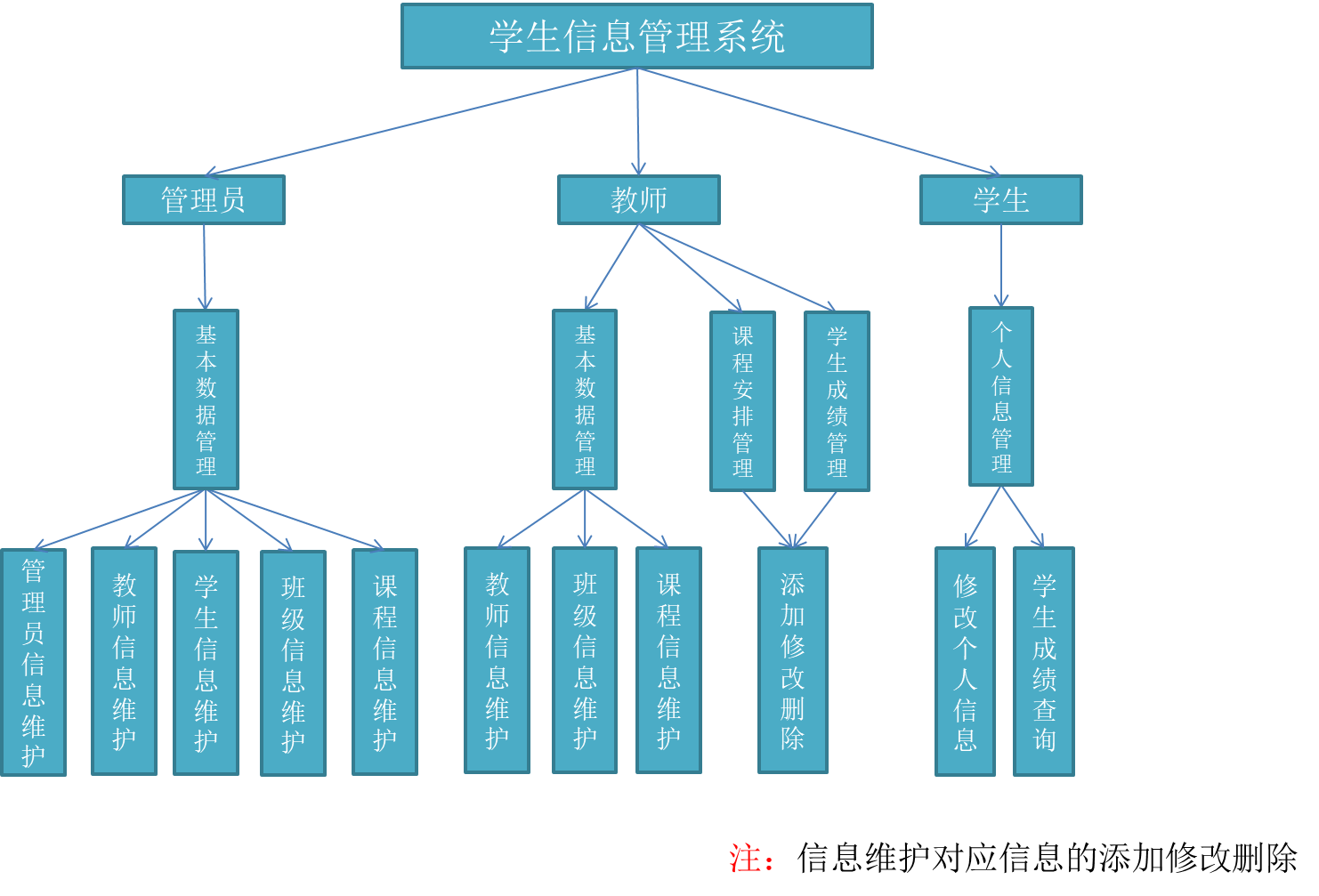
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 课程安排表 | | | | |
| 列名 | 数据类型 | 是否为空 | 约束条件 | 中文名称 |
| cou\_id | varchar | 否 | 主键 | 排课编号 |
| cou\_coid | varchar | 否 | 外键 | 课程编号 |
| cou\_clid | int | 否 | 外键 | 班级编号 |
| cou\_tid | int | 否 | 外键 | 教师编号 |
| cou\_room | varchar | 否 |  | 教室 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 成绩表 | | | | |
| 列名 | 数据类型 | 是否为空 | 约束条件 | 中文名称 |
| id | varchar | 否 | 主键 | 成绩编号 |
| stuid | int | 否 | 外键 | 学号 |
| arrangeid | varchar | 否 | 外键 | 排课编号 |
| grade | varchar | 否 |  | 分数 |
| note | varchar | 否 |  | 备注 |

##### 数据库设计语句

|  |
| --- |
| CREATE DATABASE system;  USE system;  CREATE TABLE `admin` (  `ad\_logname` varchar(20) NOT NULL COMMENT '登录名',  `ad\_name` varchar(8) NOT NULL COMMENT '姓名',  `ad\_pwd` varchar(16) NOT NULL COMMENT '密码',  PRIMARY KEY (`ad\_logname`) USING BTREE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 ROW\_FORMAT=COMPACT;  -- ----------------------------  -- Table structure for `class`  -- ----------------------------  DROP TABLE IF EXISTS `class`;  CREATE TABLE `class` (  `cl\_id` int(11) NOT NULL COMMENT '编号',  `cl\_name` varchar(15) NOT NULL COMMENT '名称',  `cl\_belong` varchar(20) NOT NULL COMMENT '所属学院',  PRIMARY KEY (`cl\_id`) USING BTREE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 ROW\_FORMAT=COMPACT;  -- ----------------------------  -- Table structure for `course`  -- ----------------------------  DROP TABLE IF EXISTS `course`;  CREATE TABLE `course` (  `co\_id` varchar(20) NOT NULL,  `co\_name` varchar(20) NOT NULL,  `co\_period` int(11) NOT NULL,  `co\_credits` int(11) NOT NULL,  `co\_type` varchar(20) NOT NULL,  `cou\_term` varchar(20) NOT NULL,  PRIMARY KEY (`co\_id`) USING BTREE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 ROW\_FORMAT=COMPACT;  -- ----------------------------  -- Table structure for `coursearrange`  -- ----------------------------  DROP TABLE IF EXISTS `coursearrange`;  CREATE TABLE `coursearrange` (  `cou\_id` varchar(20) NOT NULL,  `cou\_coid` varchar(20) DEFAULT NULL,  `cou\_clid` int(20) DEFAULT NULL,  `cou\_tid` int(20) DEFAULT NULL,  `cou\_room` varchar(16) DEFAULT NULL,  PRIMARY KEY (`cou\_id`) USING BTREE,  KEY `cou\_coid` (`cou\_coid`) USING BTREE,  KEY `cou\_clid` (`cou\_clid`) USING BTREE,  KEY `cou\_tid` (`cou\_tid`) USING BTREE,  CONSTRAINT `coursearrange\_ibfk\_1` FOREIGN KEY (`cou\_coid`) REFERENCES `course` (`co\_id`) ON UPDATE CASCADE,  CONSTRAINT `coursearrange\_ibfk\_2` FOREIGN KEY (`cou\_clid`) REFERENCES `class` (`cl\_id`) ON UPDATE CASCADE,  CONSTRAINT `coursearrange\_ibfk\_3` FOREIGN KEY (`cou\_tid`) REFERENCES `teacher` (`te\_id`) ON UPDATE CASCADE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 ROW\_FORMAT=COMPACT;  -- ----------------------------  -- Table structure for `grade`  -- ----------------------------  DROP TABLE IF EXISTS `grade`;  CREATE TABLE `grade` (  `id` varchar(20) NOT NULL,  `stuid` varchar(20) DEFAULT NULL,  `arrangeid` varchar(20) DEFAULT NULL,  `grade` varchar(6) DEFAULT NULL,  `note` varchar(10) DEFAULT NULL,  PRIMARY KEY (`id`) USING BTREE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 ROW\_FORMAT=COMPACT;  -- ----------------------------  -- Table structure for `student`  -- ----------------------------  DROP TABLE IF EXISTS `student`;  CREATE TABLE `student` (  `st\_id` int(11) NOT NULL,  `st\_name` varchar(8) NOT NULL,  `st\_pwd` varchar(16) NOT NULL,  `st\_tel` varchar(15) NOT NULL,  `st\_adress` varchar(50) NOT NULL,  `st\_email` varchar(30) NOT NULL,  `st\_classid` int(11) NOT NULL,  PRIMARY KEY (`st\_id`) USING BTREE,  KEY `st\_classid` (`st\_classid`) USING BTREE,  CONSTRAINT `student\_ibfk\_1` FOREIGN KEY (`st\_classid`) REFERENCES `class` (`cl\_id`) ON UPDATE CASCADE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 ROW\_FORMAT=COMPACT;  -- ----------------------------  -- Table structure for `teacher`  -- ----------------------------  DROP TABLE IF EXISTS `teacher`;  CREATE TABLE `teacher` (  `te\_id` int(10) NOT NULL COMMENT '编号',  `te\_name` varchar(8) NOT NULL COMMENT '姓名',  `te\_pwd` varchar(16) NOT NULL COMMENT '密码',  `te\_title` varchar(8) NOT NULL COMMENT '职称',  `te\_eduback` varchar(12) NOT NULL COMMENT '学历',  `te\_adress` varchar(50) NOT NULL COMMENT '住址',  `te\_tel` varchar(15) NOT NULL COMMENT '电话',  `te\_email` varchar(30) NOT NULL COMMENT '邮箱',  `te\_reserch` varchar(30) NOT NULL COMMENT '研究方向',  PRIMARY KEY (`te\_id`) USING BTREE  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 ROW\_FORMAT=COMPACT; |

**功能流程设计**



##### 基础配置信息

c3p0-config.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <c3p0-config>  <!-- c3p0默认配置，如果没有指定则使用该配置，下面还可以配置多个数据库 -->  <default-config>  <!-- 数据库的url -->  <property name=*"jdbcUrl"*>jdbc:mysql://localhost:3306/system</property>  <!-- 数据库驱动名 -->  <property name=*"driverClass"*>com.mysql.jdbc.Driver</property>  <!--用户名。Default: root -->  <property name=*"user"*>root</property>  <!--密码。Default: 123456 -->  <property name=*"password"*>123456</property>  <!--初始化时获取六个连接，取值应在minPoolSize与maxPoolSize之间。Default: 6 -->  <property name=*"initialPoolSize"*>6</property>  <!--连接池中保留的最大连接数。Default: 50 -->  <property name=*"maxPoolSize"*>50</property>  <!--最大空闲时间,指定的时间内未使用则连接被丢弃。若为0则永不丢弃。Default: 1000 -->  <property name=*"maxIdleTime"*>1000</property>  </default-config>    <!-- 命名的配置,指定时使用 -->  <named-config name=*"named\_ds"*>  <property name=*"driverClass"*>com.mysql.jdbc.Driver</property>  <property name=*"jdbcUrl"*>jdbc:mysql://localhost:3306/system</property>  <property name=*"user"*>root</property>  <property name=*"password"*>123456</property>  <!-- 如果池中数据连接不够时一次增长多少个 -->  <property name=*"acquireIncrement"*>5</property>  <property name=*"initialPoolSize"*>20</property>  <property name=*"minPoolSize"*>10</property>  <property name=*"maxPoolSize"*>40</property>  <property name=*"maxStatements"*>0</property>  <property name=*"maxStatementsPerConnection"*>5</property>  </named-config>    </c3p0-config> |

C3P0Utils.java

|  |
| --- |
| **package** com.test.utils;  **import** java.sql.Connection;  **import** java.sql.SQLException;  **import** javax.sql.DataSource;  **import** com.mchange.v2.c3p0.ComboPooledDataSource;  **public** **class** C3P0Utils {  // 通过c3p0-config文件获取连接池对象，使用name值为named\_ds的配置  **private** **static** ComboPooledDataSource *dataSource* = **new** ComboPooledDataSource(  "named\_ds");  // 提供一个dataSource数据源  // 后面testDBUTils\_Insert类中的queryRunner对象需要该方法来生成一个QueryRunner对象  **public** **static** DataSource getDataSource() {  **return** *dataSource*;  }  // 创建一个ThreadLocal对象，以当前线程作为key  **private** **static** ThreadLocal<Connection> *threadLocal* = **new** ThreadLocal<Connection>();  // 提供当前线程中的connection  **public** **static** Connection getConnection() **throws** SQLException {  Connection conn = *threadLocal*.get();  **if** (**null** == conn) {  conn = *dataSource*.getConnection();  *threadLocal*.set(conn);  }  **return** conn;  }  // 开启事务  **public** **static** **void** startTransaction() {  // 首先获取当前线程的连接  **try** {  Connection conn = *getConnection*();  // 关闭事物自动提交  conn.setAutoCommit(**false**);  } **catch** (SQLException e) {  e.printStackTrace();  }  }  // 提交事务  **public** **static** **void** commit() {  // 获取当前线程的连接  Connection conn = *threadLocal*.get();  **if** (**null** != conn) {  // 提交事务  **try** {  conn.commit();  } **catch** (SQLException e) {  e.printStackTrace();  }  }  }  // 回滚事务  **public** **static** **void** rollback() {  // 首先获取当前线程的连接  Connection conn = *threadLocal*.get();  **if** (**null** != conn) {  // 回滚事务  **try** {  conn.rollback();  } **catch** (SQLException e) {  e.printStackTrace();  }  }  }  // 关闭连接  **public** **static** **void** close() {  Connection conn = *threadLocal*.get();  **if** (**null** != conn) {  **try** {  conn.close();  } **catch** (SQLException e) {  e.printStackTrace();  } **finally** {  *threadLocal*.remove();// 从当前线程移除连接,防止造成内存泄漏  conn = **null**;  }  }  }  } |

#### 程序调试与测试

##### 1、基本数据管理

1-1-1 管理员维护🡪🡪🡪展示管理员信息

CheckAdminServlet.java

|  |
| --- |
| **public** **class** CheckAdminServlet **extends** HttpServlet {  **private** AdminService adminService = **new** AdminServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String ad\_logname = req.getParameter("ad\_logname");  String ad\_name = req.getParameter("ad\_name");  String ad\_pwd = req.getParameter("ad\_pwd");  List<Admin> list = adminService.findAdmin(ad\_logname,ad\_name,ad\_pwd);  req.setAttribute("list", list);  req.getRequestDispatcher("/base/adminList.jsp").forward(req, resp);  }  } |

1-1-2 管理员维护🡪🡪🡪添加管理员信息

addAdminServlet.java

|  |
| --- |
| **public** **class** addAdminServlet **extends** HttpServlet {  **private** AdminService adminService = **new** AdminServiceImpl();    @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String ad\_logname = req.getParameter("ad\_logname");  String ad\_name = req.getParameter("ad\_name");  String ad\_pwd = req.getParameter("ad\_pwd");    Admin admin = **new** Admin();  admin.setAd\_logname(ad\_logname);  admin.setAd\_name(ad\_name);  admin.setAd\_pwd(ad\_pwd);  adminService.addAdmin(admin);  req.setAttribute("admin", admin);  System.***out***.println(admin);  req.getRequestDispatcher("CheckAdminServlet").forward(req, resp);  } |

1-1-3 管理员维护🡪🡪🡪查看管理员信息

ShowAdminServlet.java

|  |
| --- |
| **public** **class** ShowAdminServlet **extends** HttpServlet {  **private** AdminService adminService = **new** AdminServiceImpl();  @Override  **protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {    String ad\_logname = req.getParameter("ad\_logname");  Admin admin = adminService.findAdminByLoginName(ad\_logname);    req.setAttribute("admin", admin);  req.getRequestDispatcher("/base/adminShow.jsp").forward(req, resp);  **return** ;  }  } |

1-1-4 管理员维护🡪🡪🡪修改管理员信息

updateAdminServlet.java

|  |
| --- |
| **public** **class** updateAdminServlet **extends** HttpServlet{  **private** AdminService adminService = **new** AdminServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String ad\_logname = req.getParameter("ad\_logname");  String ad\_name = req.getParameter("ad\_name");  String ad\_pwd = req.getParameter("ad\_pwd");    Admin admin = **new** Admin();  admin.setAd\_logname(ad\_logname);  admin.setAd\_name(ad\_name);  admin.setAd\_pwd(ad\_pwd);  adminService.updateAdmin(admin);  //打印输出admin对象查看修改情况  req.setAttribute("admin", admin);  System.***out***.println(admin);  req.getRequestDispatcher("CheckAdminServlet").forward(req, resp);  }  } |

1-1-5 管理员维护🡪🡪🡪删除管理员信息

deleteAdminServlet.java

|  |
| --- |
| **public** **class** deleteAdminServlet **extends** HttpServlet{  **private** AdminService adminService = **new** AdminServiceImpl();  //a标签里的单击事件都是get提交  @Override  **protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String ad\_logname = req.getParameter("ad\_logname");  Admin admin = **new** Admin();  admin.setAd\_logname(ad\_logname);  adminService.deleteAdminByLoginName(ad\_logname);  req.setAttribute("admin", admin);  System.***out***.println(admin);  req.getRequestDispatcher("listAllAdminServlet").forward(req, resp);  }  } |

1-2-1 学生信息维护🡪🡪🡪展示学生信息

checkStudentServlet.java

|  |
| --- |
| **public** **class** checkStudentServlet **extends** HttpServlet {  **private** StudentService studentSevice = **new** StudentServiceImpl();  **private** ClasstabService classtabService = **new** ClasstabServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  //addStudentServlet转入该方法时以下字段值均为null  String st\_id = req.getParameter("st\_id");  String st\_name = req.getParameter("st\_name");  String st\_classid = req.getParameter("st\_classid");  //点击查询的时候执行该方法返回对应记录  List<Student> list = studentSevice.findStudent(st\_id,st\_name,st\_classid);  List classtab = classtabService.queryAllClasstab();  req.setAttribute("list", list);  req.setAttribute("classtab", classtab);  req.getRequestDispatcher("/base/studentList.jsp").forward(req, resp);  }  } |

1-2-2 学生信息维护🡪🡪🡪添加学生信息

|  |
| --- |
| **public** **class** addStudentServlet **extends** HttpServlet {  **private** StudentService studentService = **new** StudentServiceImpl();  **private** ClasstabService classtabService = **new** ClasstabServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  //获取前台传过来的字段  String st\_id = req.getParameter("st\_id");  String st\_name = req.getParameter("st\_name");  String st\_pwd = req.getParameter("st\_pwd");  String st\_tel = req.getParameter("st\_tel");  String st\_adress = req.getParameter("st\_adress");  String st\_email = req.getParameter("st\_email");  String st\_classid = req.getParameter("st\_classid");  //实例化一个对象并设置这个对象的一些字段  Student student = **new** Student();  student.setSt\_id(st\_id);  student.setSt\_name(st\_name);  student.setSt\_pwd(st\_pwd);  student.setSt\_tel(st\_tel);  student.setSt\_adress(st\_adress);  student.setSt\_email(st\_email);  student.setSt\_classid(st\_classid);  //调用studentService对象的添加学生的方法并传入一个student的对象  studentService.addStudent(student);  //调用classtabService这个对象的queryAllClasstab()方法，将返回到的值放入list中  List classtab = classtabService.queryAllClasstab();  //并把这个两个对象放入request请求域中  req.setAttribute("student", student);  req.setAttribute("classtab", classtab);  System.***out***.println(student);  req.getRequestDispatcher("checkStudentServlet").forward(req, resp);  }  } |

1-2-3 学生信息维护🡪🡪🡪查看学生信息

showStudentServlet.java

|  |
| --- |
| **public** **class** showStudentServlet **extends** HttpServlet {  **private** StudentService studentService = **new** StudentServiceImpl();  **private** ClasstabService classtabService = **new** ClasstabServiceImpl();  @Override  **protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String st\_id = req.getParameter("st\_id");  String st\_name = req.getParameter("st\_name");  String st\_pwd = req.getParameter("st\_pwd");  String st\_tel = req.getParameter("st\_tel");  String st\_adress = req.getParameter("st\_adress");  String st\_email = req.getParameter("st\_email");  String st\_classid = req.getParameter("st\_classid");  //1、当前要修改的课程安排  //CourseArrange courseArrangeObj = courseArrangeService.queryCourseArrange(cou\_id);  Student student = studentService.findStudentBySt\_id(st\_id);  req.setAttribute("student", student);  List classtab = classtabService.queryAllClasstab();  req.setAttribute("classtab", classtab);//classtab  req.getRequestDispatcher("/base/studentShow.jsp").forward(req, resp);  }  } |

1-2-4 学生信息维护🡪🡪🡪修改学生信息

updateStudentServlet.java

|  |
| --- |
| **public** **class** updateStudentServlet **extends** HttpServlet {  StudentService studentService =**new** StudentServiceImpl();  **private** ClasstabService classtabService = **new** ClasstabServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String st\_id = req.getParameter("st\_id");  String st\_name = req.getParameter("st\_name");  String st\_pwd = req.getParameter("st\_pwd");  String st\_tel = req.getParameter("st\_tel");  String st\_adress = req.getParameter("st\_adress");  String st\_email = req.getParameter("st\_email");  String st\_classid = req.getParameter("st\_classid");  Student student = **new** Student();  student.setSt\_id(st\_id);  student.setSt\_name(st\_name);  student.setSt\_pwd(st\_pwd);  student.setSt\_tel(st\_tel);  student.setSt\_adress(st\_adress);  student.setSt\_email(st\_email);  student.setSt\_classid(st\_classid);  studentService.updateStudent(student);  List classtab = classtabService.queryAllClasstab();  req.setAttribute("student", student);  req.setAttribute("classtab", classtab);  System.***out***.println(student);  req.getRequestDispatcher("checkStudentServlet").forward(req, resp);  }  } |

1-2-5 学生信息维护🡪🡪🡪删除学生信息

deleteStudentServlet.java

|  |
| --- |
| **public** **class** deleteStudentServlet **extends** HttpServlet {  StudentService studentService = **new** StudentServiceImpl();  @Override  **protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String st\_id = req.getParameter("st\_id");  Student student = **new** Student();  student.setSt\_id(st\_id);  studentService.deleteStudentBySt\_id(st\_id);    req.setAttribute("student", student);  // System.out.println(student);  req.getRequestDispatcher("listAllStudentServlet").forward(req, resp);  }  } |

1-3-1 教师信息维护🡪🡪🡪展示教师信息

checkTeacherServlet.java

|  |
| --- |
| **public** **class** checkTeacherServlet **extends** HttpServlet {  **private** TeacherService teacherService = **new** TeacherServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  String te\_id = req.getParameter("te\_id");  String te\_name = req.getParameter("te\_name");  String te\_title = req.getParameter("te\_title");  String te\_tel = req.getParameter("te\_tel");  String te\_reserch = req.getParameter("te\_reserch");  List<Teacher> list = teacherService.findTeacher(te\_id, te\_name, te\_title, te\_tel, te\_reserch);  req.setAttribute("list", list);  req.getRequestDispatcher("/base/teacherList.jsp").forward(req, resp);  }  } |

1-3-2 教师信息维护🡪🡪🡪添加教师信息

addTeacherServlet.java

|  |
| --- |
| **public** **class** addTeacherServlet **extends** HttpServlet {  **private** TeacherService teacherService = **new** TeacherServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  String te\_id = req.getParameter("te\_id");  String te\_name = req.getParameter("te\_name");  String te\_pwd = req.getParameter("te\_pwd");  String te\_title = req.getParameter("te\_title");  String te\_eduback = req.getParameter("te\_eduback");  String te\_adress = req.getParameter("te\_adress");  String te\_tel = req.getParameter("te\_tel");  String te\_email = req.getParameter("te\_email");  String te\_reserch = req.getParameter("te\_reserch");  Teacher teacher = **new** Teacher();  teacher.setTe\_id(te\_id);  teacher.setTe\_name(te\_name);  teacher.setTe\_pwd(te\_pwd);  teacher.setTe\_title(te\_title);  teacher.setTe\_eduback(te\_eduback);  teacher.setTe\_adress(te\_adress);  teacher.setTe\_tel(te\_tel);  teacher.setTe\_email(te\_email);  teacher.setTe\_reserch(te\_reserch);  teacherService.addTeacher(teacher);  req.setAttribute("teacher", teacher);  System.***out***.println(teacher);  req.getRequestDispatcher("checkTeacherServlet").forward(req, resp);  }  } |

1-3-3 教师信息维护🡪🡪🡪查看教师信息

showTeacherServlet.java

|  |
| --- |
| **public** **class** showTeacherServlet **extends** HttpServlet {  **private** TeacherService teacherService = **new** TeacherServiceImpl();  @Override  **protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String te\_id = req.getParameter("te\_id");  Teacher teacher = teacherService.findTeacherByTe\_id(te\_id);  req.setAttribute("teacher", teacher);  req.getRequestDispatcher("/base/teacherShow.jsp").forward(req, resp);  **return** ;  }  } |

1-3-4 教师信息维护🡪🡪🡪修改教师信息

updateTeacherServlet.java

|  |
| --- |
| **public** **class** updateTeacherServlet **extends** HttpServlet {  **private** TeacherService teacherService = **new** TeacherServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  String te\_id = req.getParameter("te\_id");  String te\_name = req.getParameter("te\_name");  String te\_pwd = req.getParameter("te\_pwd");  String te\_title = req.getParameter("te\_title");  String te\_eduback = req.getParameter("te\_eduback");  String te\_adress = req.getParameter("te\_adress");  String te\_tel = req.getParameter("te\_tel");  String te\_email = req.getParameter("te\_email");  String te\_reserch = req.getParameter("te\_reserch");  Teacher teacher = **new** Teacher();  teacher.setTe\_id(te\_id);  teacher.setTe\_name(te\_name);  teacher.setTe\_pwd(te\_pwd);  teacher.setTe\_title(te\_title);  teacher.setTe\_eduback(te\_eduback);  teacher.setTe\_adress(te\_adress);  teacher.setTe\_tel(te\_tel);  teacher.setTe\_email(te\_email);  teacher.setTe\_reserch(te\_reserch);  teacherService.updateTeacher(teacher);  req.setAttribute("teacher", teacher);  System.***out***.println(teacher);  req.getRequestDispatcher("checkTeacherServlet").forward(req, resp);  }  } |

1-3-5 教师信息维护🡪🡪🡪删除教师信息

deleteTeacherServlet.java

|  |
| --- |
| **public** **class** deleteTeacherServlet **extends** HttpServlet {  **private** TeacherService teacherService = **new** TeacherServiceImpl();  @Override  **protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  String te\_id = req.getParameter("te\_id");  Teacher teacher = **new** Teacher();  teacher.setTe\_id(te\_id);  teacherService.deleteTeacherByTe\_id(te\_id);  req.setAttribute("teacher", teacher);  System.***out***.println(teacher);  req.getRequestDispatcher("listAllTeacherServlet").forward(req, resp);  }  } |

1-4-1 班级信息维护🡪🡪🡪展示班级信息

BaseServlet.java

|  |
| --- |
| **public** **class** BaseServlet **extends** HttpServlet {  @Override  **public** **void** service(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  String[] methods = req.getParameterValues("method");  String method ;  **if** (**null** != methods && methods.length!=0 && !methods[0].trim().isEmpty()) {  method = methods[0];  } **else** {  method = "execute";  }  System.***out***.println(req.getRequestURI() + " method: " + method);  //为页面请求匹配具体处理的方法  Class clazz = **this**.getClass();  **try** {  Method md = clazz.getMethod(method, HttpServletRequest.**class**, HttpServletResponse.**class**);  **if**(**null**!=md){  //method.invoke()方法，用来执行某个的对象的目标方法  String jspPath = (String) md.invoke(**this**, req, resp);  **if** (**null** != jspPath) {  req.getRequestDispatcher(jspPath).forward(req, resp);  }  }  } **catch** (Exception e) {  e.printStackTrace();  }  }  // 默认方法  **public** String execute(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  **return** **null**;  }  } |

*/studentManager/ClassServlet?method=CheckClass*

1-4-2 班级信息维护🡪🡪🡪添加班级信息

*/studentManager/ClassServlet?method=AddClass*

1-4-3 班级信息维护🡪🡪🡪查看班级信息

*/studentManager/ClassServlet?method=ShowClass*

1-4-4 班级信息维护🡪🡪🡪修改班级信息

*/studentManager/ClassServlet?method=UpdateClass*

1-4-5 班级信息维护🡪🡪🡪删除班级信息

*/studentManager/ClassServlet?method=DeleteClass*

ClassServlet.java

|  |
| --- |
| **public** **class** ClassServlet **extends** BaseServlet{  **private** ClasstabService classtabService = **new** ClasstabServiceImpl();  **private** Classtab classtab = **new** Classtab();  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*点击查看/修改后显示的班级详情页面\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** List ShowClass(HttpServletRequest req,HttpServletResponse resp) **throws** Exception {  String cl\_id = req.getParameter("cl\_id");  // \*\*\*\*\*\*\*根据班级编号查询课程信息\*\*\*\*\*\*\*\*  Classtab classtab = classtabService.queryClasstab(cl\_id);  req.setAttribute("classtab", classtab);  req.getRequestDispatcher("/base/classShow.jsp").forward(req, resp);  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*修改班级详情页面\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** **int** UpdateClass(HttpServletRequest req,HttpServletResponse resp) **throws** Exception {  String cl\_id = req.getParameter("cl\_id");  String cl\_name = req.getParameter("cl\_name");  String cl\_belong = req.getParameter("cl\_belong");  Classtab classtab = **new** Classtab();  classtab.setCl\_id(cl\_id);  classtab.setCl\_name(cl\_name);  classtab.setCl\_belong(cl\_belong);  **int** changes = classtabService.updateclass(classtab);  req.setAttribute("classtab", classtab);  req.getRequestDispatcher("/ClassServlet?method=CheckClass").forward(req, resp);;  **return** changes;  }  /\*用于班级信息列表显示\*/  /\*1.先从前台获取到相应的位置  \*2.从数据库中获取到全部值存入List集合  \*3.存放到request请求中，留着传值  \* \*/  /\*\*\*\*\*\*第一次用到：adminList中的《班级信息维护》\*\*\*\*\*\*/  **public** String CheckClass(HttpServletRequest request,HttpServletResponse response) **throws**  ServletException, IOException {  String cl\_id = request.getParameter("cl\_id");  String cl\_name = request.getParameter("cl\_name");  String cl\_belong = request.getParameter("cl\_belong");  Classtab cls = (Classtab) request.getAttribute("classtab");  List<Classtab> classtabList = **new** ArrayList<Classtab>();  **if** (cls != **null**) {  classtabList.add(cls);  } **else**{  //根据相应字段进行用户查询  classtabList = classtabService.queryAllClasstab(cl\_id,cl\_name,cl\_belong);  }  request.setAttribute("cl\_id", cl\_id);  request.setAttribute("cl\_name", cl\_name);  request.setAttribute("cl\_belong", cl\_belong);  request.setAttribute("classtabList", classtabList);  request.getRequestDispatcher("/base/classList.jsp").forward(request, response);  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*班级添加功能\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** String AddClass(HttpServletRequest req,HttpServletResponse resp) **throws**  ServletException, IOException {  String cl\_id = req.getParameter("cl\_id");  String cl\_name = req.getParameter("cl\_name");  String cl\_belong = req.getParameter("cl\_belong");  classtab.setCl\_id(cl\_id);  classtab.setCl\_name(cl\_name);  classtab.setCl\_belong(cl\_belong);  classtabService.addclass(classtab);  req.setAttribute("classtab", classtab);  System.***out***.println(classtab);  req.getRequestDispatcher("/ClassServlet?method=CheckClass").forward(req, resp);  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*班级删除功能\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** String DeleteClass(HttpServletRequest req,HttpServletResponse resp) **throws**  ServletException, IOException {  String cl\_id = req.getParameter("cl\_id");  classtabService.deleteclass(cl\_id);  req.getRequestDispatcher("/ClassServlet?method=CheckClass").forward(req, resp);  **return** **null**;  }  } |

1-5-1 班级信息维护🡪🡪🡪展示课程信息

*/studentManager/CourseServlet?method=CheckCourse*

1-5-2 班级信息维护🡪🡪🡪添加课程信息

*/studentManager/CourseServlet?method=AddCourse*

1-5-3 班级信息维护🡪🡪🡪查看课程信息

*/studentManager/CourseServlet?method=ShowCourse*

1-5-4 班级信息维护🡪🡪🡪修改课程信息

*/studentManager/CourseServlet?method=UpdateCourse*

1-5-5 班级信息维护🡪🡪🡪删除课程信息

*/studentManager/CourseServlet?method=DeleteCourse*

CourseServlet.java

|  |
| --- |
| **public** **class** CourseServlet **extends** BaseServlet{  **private** Course course = **new** Course();  **private** CourseService courseService = **new** CourseServiceImpl();  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** String CheckCourse(HttpServletRequest req,HttpServletResponse resp) **throws** ServletException, IOException{  String co\_id = req.getParameter("co\_id"); //编号  String co\_name = req.getParameter("co\_name"); //名称  String co\_period = req.getParameter("co\_period"); //学时  String co\_credits = req.getParameter("co\_credits"); //学分  String co\_type = req.getParameter("co\_type"); //课程类别  String co\_term = req.getParameter("co\_term"); //学期  Course co = (Course) req.getAttribute("course");  List<Course> courseList = **new** ArrayList<Course>();  **if**(co != **null**){ //添加课程后跳转到课程列表页面  courseList.add(co);  req.setAttribute("co\_id", co.getCo\_id());  req.setAttribute("co\_name", co.getCo\_name());  req.setAttribute("co\_period", co.getCo\_period());  req.setAttribute("co\_credits", co.getCo\_credits());  req.setAttribute("co\_type", co.getCo\_type());  req.setAttribute("co\_term", co.getCo\_term());  req.setAttribute("courseList", courseList);  }**else**{ //点击查询按钮后，跳转到课程列表页面  courseList = courseService.ListCourseByEachField(co\_id, co\_name, co\_period, co\_credits, co\_type, co\_term);  req.setAttribute("co\_id", co\_id);  req.setAttribute("co\_name", co\_name);  req.setAttribute("co\_period", co\_period);  req.setAttribute("co\_credits", co\_credits);  req.setAttribute("co\_type", co\_type);  req.setAttribute("co\_term", co\_term);  req.setAttribute("courseList", courseList);  }  req.getRequestDispatcher("/base/courseList.jsp").forward(req, resp);  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*课程添加功能\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** String AddCourse(HttpServletRequest req,HttpServletResponse resp) **throws**  ServletException, IOException {  String co\_id = req.getParameter("co\_id"); //编号  String co\_name = req.getParameter("co\_name"); //名称  String co\_period = req.getParameter("co\_period");//学时  String co\_credits = req.getParameter("co\_credits"); //学分  String co\_type = req.getParameter("co\_type"); //课程类别  String co\_term = req.getParameter("co\_term"); //学期    course.setCo\_id(co\_id);  course.setCo\_name(co\_name);  course.setCo\_period(co\_period);  course.setCo\_credits(co\_credits);  course.setCo\_type(co\_type);  course.setCo\_term(co\_term);  courseService.addCourse(course);  req.setAttribute("course", course);  req.getRequestDispatcher("/CourseServlet?method=CheckCourse").forward(req, resp);  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*删除课程功能\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** String DeleteCourse(HttpServletRequest req,HttpServletResponse resp) **throws**  ServletException, IOException {  String co\_id = req.getParameter("co\_id");  courseService.deleteCourse(co\_id);  List courseList = courseService.ListAllCourse();  req.setAttribute("courseList", courseList);  req.getRequestDispatcher("/base/courseList.jsp").forward(req, resp);  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*点击查看/修改后显示的课程详情页面\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** List ShowCourse(HttpServletRequest req,HttpServletResponse resp) **throws** Exception{  String co\_id = req.getParameter("co\_id");  Course course = courseService.ListCourseById(co\_id);  req.setAttribute("course", course);  req.getRequestDispatcher("/base/courseShow.jsp").forward(req, resp);;  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*修改课程详情页面\*\*\*\*\*\*\*\*\*\*\*\*\*/ //UpdateCourse  **public** **int** UpdateCourse(HttpServletRequest req,HttpServletResponse resp) **throws** Exception {  String co\_id = req.getParameter("co\_id"); //编号  String co\_name = req.getParameter("co\_name"); //名称  String co\_period = req.getParameter("co\_period");//学时  String co\_credits = req.getParameter("co\_credits"); //学分  String co\_type = req.getParameter("co\_type"); //课程类别  String co\_term = req.getParameter("co\_term"); //学期  course.setCo\_id(co\_id);  course.setCo\_name(co\_name);  course.setCo\_period(co\_period);  course.setCo\_credits(co\_credits);  course.setCo\_type(co\_type);  course.setCo\_term(co\_term);  **int** changes = courseService.updateCourse(course);  req.setAttribute("course", course);  req.getRequestDispatcher("/CourseServlet?method=CheckCourse").forward(req, resp);  **return** changes;  }  } |

##### 2、课程安排管理

2-1 课程安排管理🡪🡪🡪展示课程安排信息

*/studentManager/CourseArrangeServlet?method=CheckCourse*

2-2 课程安排管理🡪🡪🡪添加课程安排信息

*/studentManager/CourseArrangeServlet?method=*AddArrange

2-3 课程安排管理🡪🡪🡪查看课程安排信息

*/studentManager/CourseArrangeServlet?method=*ShowArrange

2-4 课程安排管理🡪🡪🡪修改课程安排信息

*/studentManager/CourseArrangeServlet?method=*UpdateArrange

2-5 课程安排管理🡪🡪🡪删除课程安排信息

*/studentManager/CourseArrangeServlet?method=*DeleteArrange

CourseArrangeServlet.java

|  |
| --- |
| **public** **class** CourseArrangeServlet **extends** BaseServlet {  **private** CourseArrange courseArrange = **new** CourseArrange();  **private** TeacherService teacherService = **new** TeacherServiceImpl();  **private** ClasstabService classtabService = **new** ClasstabServiceImpl();  **private** CourseService courseService = **new** CourseServiceImpl();  **private** CourseArrangeService courseArrangeService = **new** CourseArrangeServiceImpl();  **public** String CkeckCourse(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  // 直接从首页进入，以下参数或域变量均为null  String cou\_id = req.getParameter("cou\_id"); // 安排编号  String cou\_coid = req.getParameter("cou\_coid"); // 安排课程  String cou\_clid = req.getParameter("cou\_clid"); // 上课班级  String cou\_tid = req.getParameter("cou\_tid"); // 任课老师  String cou\_room = req.getParameter("cou\_room"); // 上课地点  CourseArrange ca = (CourseArrange) req.getAttribute("courseArrange");  List<CourseArrange> arrangeList = **new** ArrayList<CourseArrange>();  **if** (ca != **null**) {// 添加课程安排后跳转到课程列表页面  arrangeList.add(ca);  req.setAttribute("cou\_id", ca.getCou\_id());  req.setAttribute("cou\_coid", ca.getCourse().getCo\_id());  req.setAttribute("cou\_clid", ca.getClassTbl().getCl\_id());  req.setAttribute("cou\_tid", ca.getTeacher().getTe\_id());  req.setAttribute("cou\_room", ca.getCou\_room());  } **else** {// 点击查询按钮后，跳转到课程列表页面  // 从主界面进入ArrangeList界面时，参数全部为空的时候即为全部查询  arrangeList = courseArrangeService.queryAllCourseArrange(cou\_id, cou\_coid, cou\_clid, cou\_tid, cou\_room);  }  req.setAttribute("arrangeList", arrangeList);  List teacherList = teacherService.findAllTeacher();  req.setAttribute("teacherList", teacherList);  List classtabList = classtabService.queryAllClasstab();  req.setAttribute("classtabList", classtabList);  List courseList = courseService.ListAllCourse();  req.setAttribute("courseList", courseList);  req.getRequestDispatcher("/arrange/arrangeList.jsp").forward(req, resp);  **return** **null**;  }  //PreAddArrange 提供预先将课程、班级、老师信息传到前台  **public** String PreAddArrange(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  // 全部课程  List courses = courseService.ListAllCourse();  req.setAttribute("courses", courses);  // 全部班级  List classtbls = classtabService.queryAllClasstab();  req.setAttribute("classtbls", classtbls);  // 全部老师  List teachers = teacherService.findAllTeacher();  req.setAttribute("teachers", teachers);  req.getRequestDispatcher("arrange/addArrange.jsp").forward(req, resp);  **return** **null**;  }  // AddArrange  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 课程安排添加功能 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** String AddArrange(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  String cou\_id = req.getParameter("cou\_id"); // 安排编号  String cou\_coid = req.getParameter("cou\_coid"); // 安排课程  String cou\_clid = req.getParameter("cou\_clid"); // 上课班级  String cou\_tid = req.getParameter("cou\_tid"); // 任课老师  String cou\_room = req.getParameter("cou\_room"); // 上课地点  courseArrange.setCou\_id(cou\_id);  courseArrange.setCou\_coid(cou\_coid);  courseArrange.setCou\_clid(cou\_clid);  courseArrange.setCou\_tid(cou\_tid);  courseArrange.setCou\_room(cou\_room);  courseArrangeService.addCourseArrange(courseArrange);  req.setAttribute("courseArrange", courseArrange);  // 全部课程  List courses = courseService.ListAllCourse();  req.setAttribute("courses", courses);  // 全部班级  List classtbls = classtabService.queryAllClasstab();  req.setAttribute("classtbls", classtbls);  // 全部老师  List teachers = teacherService.findAllTeacher();  req.setAttribute("teachers", teachers);  req.getRequestDispatcher("/CourseArrangeServlet?method=CkeckCourse").forward(req, resp);  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 点击查看/修改后显示的课程安排详情页面 \*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** List ShowArrange(HttpServletRequest req, HttpServletResponse resp) **throws** Exception {  String cou\_id = req.getParameter("cou\_id"); // 安排编号  // 1、当前要修改的课程安排  CourseArrange courseArrangeObj = courseArrangeService.queryCourseArrange(cou\_id);  req.setAttribute("arrange", courseArrangeObj);  // 2 全部课程  List courses = courseService.ListAllCourse();  req.setAttribute("courses", courses);  // 3 全部班级  List classtbls = classtabService.queryAllClasstab();  req.setAttribute("classtbls", classtbls);  // 4 全部老师  List teachers = teacherService.findAllTeacher();  req.setAttribute("teachers", teachers);  req.getRequestDispatcher("/arrange/arrangeShow.jsp").forward(req, resp);  **return** **null**;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 修改课程安排详情页面 \*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** **int** UpdateArrange(HttpServletRequest req, HttpServletResponse resp) **throws** Exception {  String cou\_id = req.getParameter("cou\_id"); // 安排编号  String cou\_coid = req.getParameter("cou\_coid"); // 安排课程  String cou\_clid = req.getParameter("cou\_clid"); // 上课班级  String cou\_tid = req.getParameter("cou\_tid"); // 任课老师  String cou\_room = req.getParameter("cou\_room"); // 上课地点  courseArrange.setCou\_id(cou\_id);  courseArrange.setCou\_coid(cou\_coid);  courseArrange.setCou\_clid(cou\_clid);  courseArrange.setCou\_tid(cou\_tid);  courseArrange.setCou\_room(cou\_room);  **int** changes = courseArrangeService.updateCourseArrange(courseArrange);  req.setAttribute("courseArrange", courseArrange);  req.getRequestDispatcher("/CourseArrangeServlet?method=CkeckCourse").forward(req, resp);  **return** changes;  }  /\* DeleteArrange \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*删除课程安排功能\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  **public** String DeleteArrange(HttpServletRequest req,HttpServletResponse resp) **throws**  ServletException, IOException {  String cou\_id = req.getParameter("cou\_id");  courseArrangeService.deleteCourseArrange(cou\_id);  List arrangeList = courseArrangeService.queryAllCourseArrange();  req.setAttribute("arrangeList", arrangeList);  req.getRequestDispatcher("/arrange/arrangeList.jsp").forward(req, resp);  **return** **null**;  }  } |

##### 3、学生成绩管理

3-1 学生成绩管理🡪🡪🡪展示学生成绩信息

*GradeServlet?method=* CheckGradeServlet

3-2 学生成绩管理🡪🡪🡪添加学生成绩信息

*GradeServlet?method=*addGrade

3-3 学生成绩管理🡪🡪🡪查看学生成绩信息

*GradeServlet?method=*queryGradeby

GradeServlet.java

|  |
| --- |
| **public** **class** GradeServlet **extends** BaseServlet{  ClasstabService clsService = **new** ClasstabServiceImpl();  CourseService courseService = **new** CourseServiceImpl();  GradeService gradeService = **new** GradeServiceImpl();  StudentService studentService = **new** StudentServiceImpl();  CoursearrangeService coursearrangeService = **new** CoursearrangeServiceImpl();  //显示 ok  **public** List CheckGradeServlet(HttpServletRequest request, HttpServletResponse response) **throws** Exception {  List classList = clsService.queryAllClasstab();  request.setAttribute("classList", classList);  List courseList = courseService.findAllCourse();  request.setAttribute("courseList", courseList);  List gradeList = gradeService.findGrades();  request.setAttribute("gradeList", gradeList);  request.getRequestDispatcher("/grade/submitGrade.jsp").forward(request, response);  System.***out***.println(classList);  System.***out***.println(courseList);  System.***out***.println(gradeList);  **return** **null**;  }  //查询  **public** String queryGradeby(HttpServletRequest request,HttpServletResponse response) **throws** Exception{  String courseNo = request.getParameter("courseNo");  String classNo = request.getParameter("classNo");  //通过课程号和班级号获得课程安排编号  //classNo  //List gradeList = cou  List<Coursearrange> arrange = coursearrangeService.findCoursearranges(courseNo, classNo);  List gradeList = gradeService.findGrades(arrange);  request.setAttribute("gradeList", gradeList);  List classList = clsService.queryAllClasstab();  request.setAttribute("classList", classList);  List courseList = courseService.findAllCourse();  request.setAttribute("courseList", courseList);  request.getRequestDispatcher("/grade/submitGrade.jsp").forward(request, response);  System.***out***.println(classList);  System.***out***.println(courseList);  System.***out***.println(gradeList);  **return** **null**;  }  //录入  **public** List addGrade(HttpServletRequest request,HttpServletResponse response) **throws** Exception{  String ids = request.getParameter("ids");  String scores = request.getParameter("scores");  String remarks = request.getParameter("remarks");  String courseNo = request.getParameter("courseNo");  String classNo = request.getParameter("classNo");  String[] arr\_id = ids.split(",");  String[] arr\_socre = scores.split(",");  String[] arr\_remark = remarks.split(",");  **if** (arr\_id.length == arr\_remark.length && arr\_remark.length == arr\_socre.length) {  **for** (**int** i = 0; i < arr\_id.length; i++) {  gradeService.updataGradeByID(arr\_id[i],arr\_socre[i],arr\_remark[i]);  }  }  List classList = clsService.queryAllClasstab();  request.setAttribute("classList", classList);  List courseList = courseService.findAllCourse();  request.setAttribute("courseList", courseList);  List gradeList = gradeService.findGrades();  request.setAttribute("gradeList", gradeList);  //request.getRequestDispatcher("GradeServlet?method=CheckGradeServlet").forward(request, response);  request.getRequestDispatcher("/grade/submitGrade.jsp").forward(request, response);  **return** **null**;  }  } |

##### 4、个人信息管理

4-1 个人信息管理🡪🡪🡪展示个人信息

ShowPerSonServlet.java

|  |
| --- |
| **public** **class** ShowPerSonServlet **extends** HttpServlet{  **private** AdminService adminService = **new** AdminServiceImpl();  **private** StudentService studentService = **new** StudentServiceImpl();  **private** TeacherService teacherService = **new** TeacherServiceImpl();  **private** ClasstabService classtabService = **new** ClasstabServiceImpl();  @Override  **protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  **this**.doPost(req, resp);  }  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  HttpSession session = req.getSession(**false**);  Integer userType = (Integer) session.getAttribute("userType");  String studentNo = (String) session.getAttribute("s1");  String adminName = (String) session.getAttribute("s2");  String teacherid = (String)session.getAttribute("s3");  System.***out***.println(userType);  System.***out***.println(adminName);  **if**(userType==0){//findStudentByStudentNo(studentNo);  Admin admin = adminService.findAdminByLoginName(adminName);  System.***out***.println(admin.toString()+"");  req.setAttribute("admin", admin);  req.getRequestDispatcher("person/adminShow.jsp").forward(req, resp);  }**else** **if**(userType==1){  Teacher teacher = teacherService.findTeacherByid(teacherid);  System.***err***.println(teacherid.toString()+"");  req.setAttribute("teacher", teacher);  req.getRequestDispatcher("person/teacherShow.jsp").forward(req, resp);  }**else** {  Student student = studentService.findStudentByid(studentNo);  List classtab = classtabService.queryAllClasstab();  req.setAttribute("classtab", classtab);//classtab  req.setAttribute("student", student);  System.***out***.println(studentNo);  System.***out***.println(student);  System.***out***.println(classtab);  req.getRequestDispatcher("person/studentShow.jsp").forward(req, resp);  }  **return**;  }  } |

4-2 个人信息管理🡪🡪🡪管理员信息修改

updateAdminServlet.java

|  |
| --- |
| **public** **class** updateAdminServlet **extends** HttpServlet{  **private** AdminService adminService = **new** AdminServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String ad\_logname = req.getParameter("ad\_logname");  String ad\_name = req.getParameter("ad\_name");  String ad\_pwd = req.getParameter("ad\_pwd");  Admin admin = **new** Admin();  admin.setAd\_logname(ad\_logname);  admin.setAd\_name(ad\_name);  admin.setAd\_pwd(ad\_pwd);  adminService.updateAdmin(admin);  //打印输出admin对象查看修改情况  req.setAttribute("admin", admin);  System.***out***.println(admin);  req.getRequestDispatcher("CheckAdminServlet").forward(req, resp);  }  } |

4-3-1 个人信息管理🡪🡪🡪学生成绩查询

*GradeServlet?method=*queryGradeby

|  |
| --- |
| //查询  **public** String queryGradeby(HttpServletRequest request,HttpServletResponse response) **throws** Exception{  String courseNo = request.getParameter("courseNo");  String classNo = request.getParameter("classNo");  //通过课程号和班级号获得课程安排编号  //classNo  //List gradeList = cou  List<Coursearrange> arrange = coursearrangeService.findCoursearranges(courseNo, classNo);  List gradeList = gradeService.findGrades(arrange);  request.setAttribute("gradeList", gradeList);  List classList = clsService.queryAllClasstab();  request.setAttribute("classList", classList);  List courseList = courseService.findAllCourse();  request.setAttribute("courseList", courseList);  request.getRequestDispatcher("/grade/submitGrade.jsp").forward(request, response);  System.***out***.println(classList);  System.***out***.println(courseList);  System.***out***.println(gradeList);  **return** **null**;  } |

4-3-2 个人信息管理🡪🡪🡪学生信息修改

updateStudentServlet.java

|  |
| --- |
| **public** **class** updateStudentServlet **extends** HttpServlet {  StudentService studentService =**new** StudentServiceImpl();  **private** ClasstabService classtabService = **new** ClasstabServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException {  String st\_id = req.getParameter("st\_id");  String st\_name = req.getParameter("st\_name");  String st\_pwd = req.getParameter("st\_pwd");  String st\_tel = req.getParameter("st\_tel");  String st\_adress = req.getParameter("st\_adress");  String st\_email = req.getParameter("st\_email");  String st\_classid = req.getParameter("st\_classid");  Student student = **new** Student();  student.setSt\_id(st\_id);  student.setSt\_name(st\_name);  student.setSt\_pwd(st\_pwd);  student.setSt\_tel(st\_tel);  student.setSt\_adress(st\_adress);  student.setSt\_email(st\_email);  student.setSt\_classid(st\_classid);  studentService.updateStudent(student);  List classtab = classtabService.queryAllClasstab();  req.setAttribute("student", student);  req.setAttribute("classtab", classtab);  System.***out***.println(student);  req.getRequestDispatcher("checkStudentServlet").forward(req, resp);  }  } |

4-4 个人信息管理🡪🡪🡪教师信息修改

updateTeacherServlet.java

|  |
| --- |
| **public** **class** updateTeacherServlet **extends** HttpServlet {  **private** TeacherService teacherService = **new** TeacherServiceImpl();  @Override  **protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {  String te\_id = req.getParameter("te\_id");  String te\_name = req.getParameter("te\_name");  String te\_pwd = req.getParameter("te\_pwd");  String te\_title = req.getParameter("te\_title");  String te\_eduback = req.getParameter("te\_eduback");  String te\_adress = req.getParameter("te\_adress");  String te\_tel = req.getParameter("te\_tel");  String te\_email = req.getParameter("te\_email");  String te\_reserch = req.getParameter("te\_reserch");  Teacher teacher = **new** Teacher();  teacher.setTe\_id(te\_id);  teacher.setTe\_name(te\_name);  teacher.setTe\_pwd(te\_pwd);  teacher.setTe\_title(te\_title);  teacher.setTe\_eduback(te\_eduback);  teacher.setTe\_adress(te\_adress);  teacher.setTe\_tel(te\_tel);  teacher.setTe\_email(te\_email);  teacher.setTe\_reserch(te\_reserch);  teacherService.updateTeacher(teacher);  req.setAttribute("teacher", teacher);  System.***out***.println(teacher);  req.getRequestDispatcher("checkTeacherServlet").forward(req, resp);  }  } |

##### 5、结果分析

1、项目运行截图

1-1登录界面

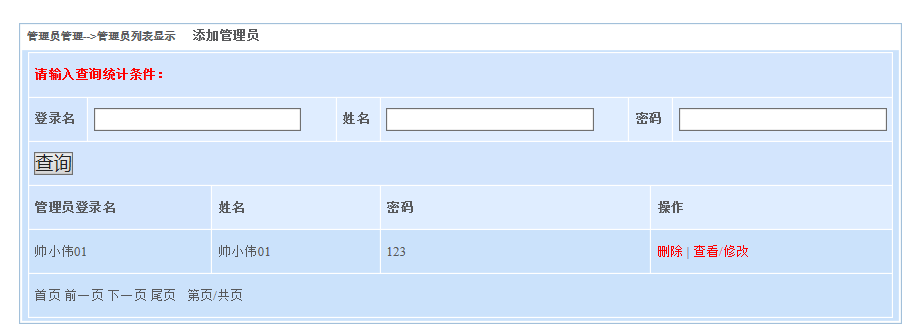


1-2功能模块界面

1-2-1 基本数据管理—管理员维护

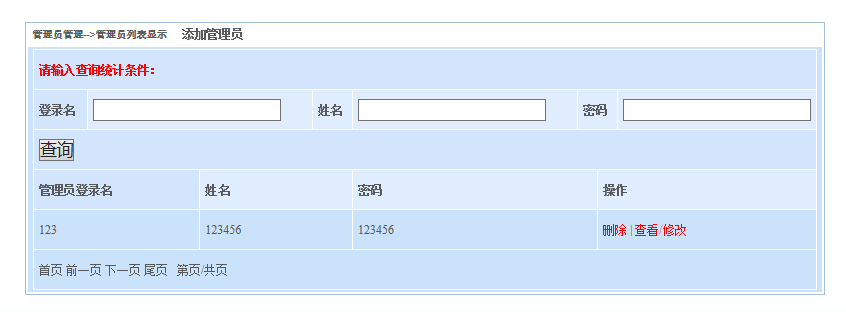






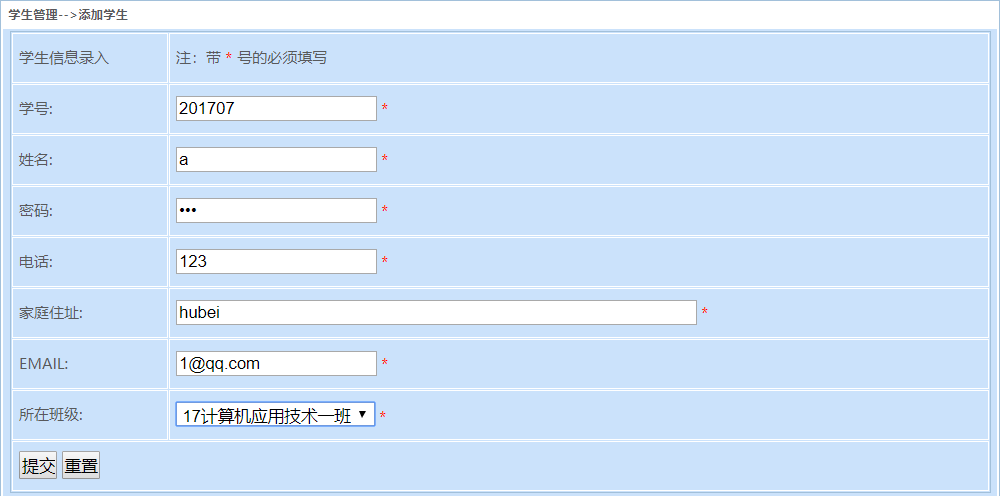






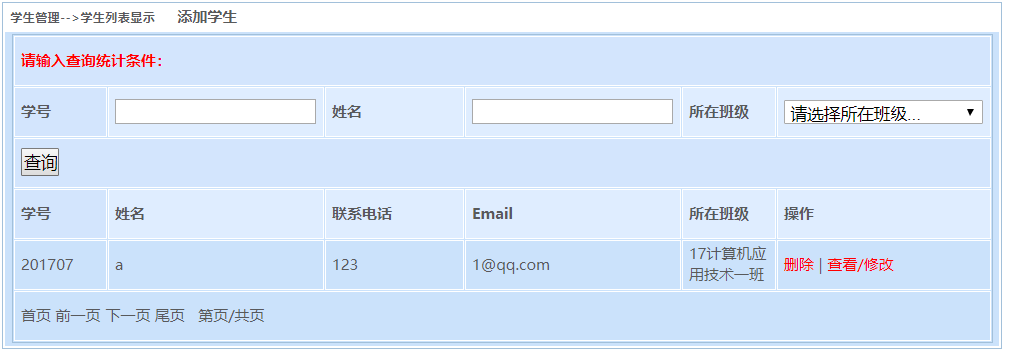
1-3-1 基本数据管理—学生信息维护



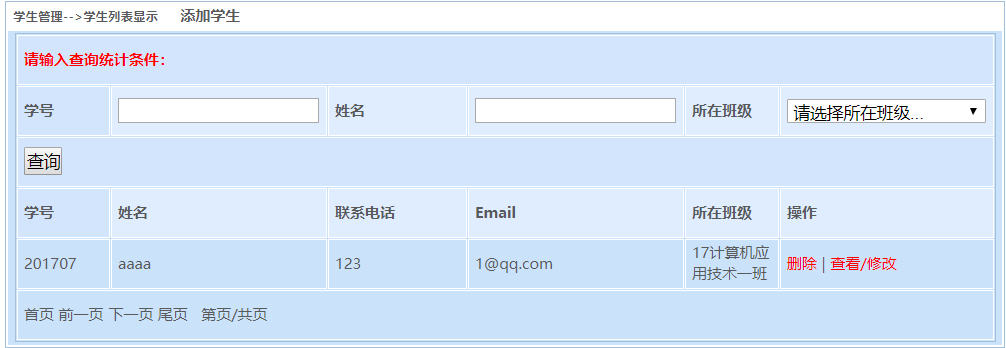












1-4-1 基本数据管理—教师信息维护

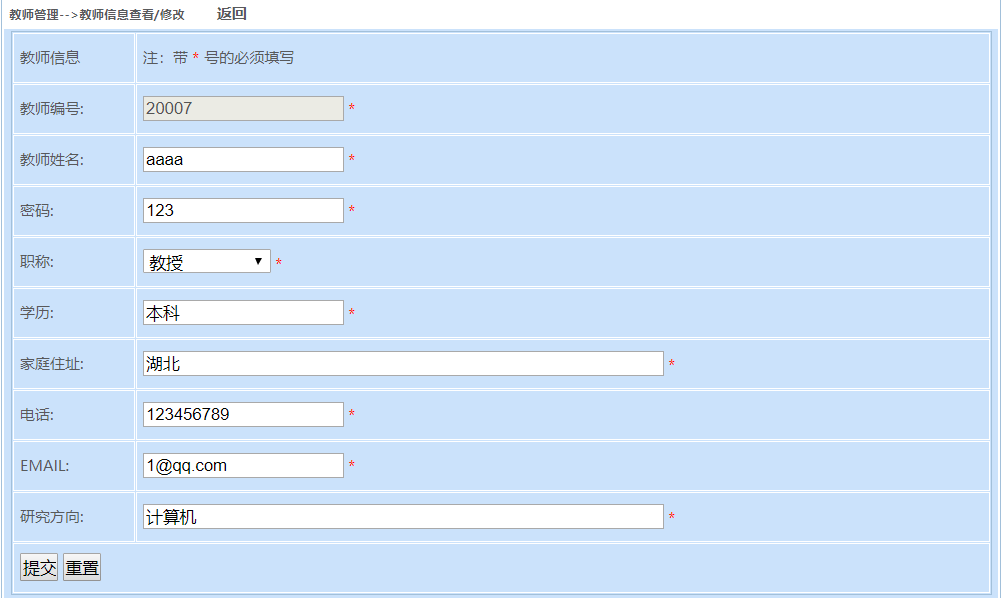














1-5-1 基本数据管理—课程信息维护







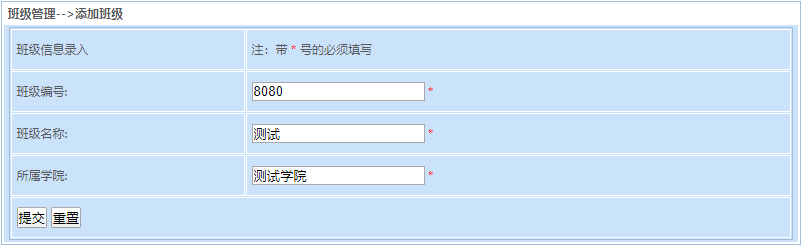






1-6-1 基本数据管理—班级信息维护







1-7-1课程安排管理











1-8-1学生成绩管理





1-9个人信息管理



**实训总结：**

此次实训的项目采用了分层加面向接口的编程思想，在后端采用servlet来进行数据的前后端交互。而在实训过程中由于在最开始的底层设计时后端的相关参数名称与前端jsp页面中的参数名称不同，导致在功能实现的过程中经常出错，调试修改也非常麻烦。前后端的统一真的至关重要。然后在实现的过程中要多debug调试。要养成良好的编程习惯。通过此次实训，我也学到了很多知识，收获良多。