

南京航空航天大学

**软件设计模式与体系结构**

**课程报告**

**姓名：刘浩**

**班级：1620304**

**学号：162030227**

**专业：软件工程**

1. 实验目的

理解框架的概念，理解配置在企业级开发中的重要性，了解Spring框架的基本结构，特点，并以Spring框架为基础，构件一个简单的web应用系统，包含基本的用户管理，界面展示功能，体现MVC特征，领会多种设计模式的复合使用，重点是观察者模式在MVC中的使用。

下载jdk1.8，IntelliJ IDEA，VSCode等相关软件及数据库

1. Spring框架架构及MVC特点
2. Spring框架架构、

springmvc是spring[框架](https://so.csdn.net/so/search?q=%E6%A1%86%E6%9E%B6&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/Philanderers/article/details/_blank)的一个模块，springmvc和spring无需通过中间整合层进行整合。springmvc是一个基于[mvc](https://so.csdn.net/so/search?q=mvc&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/Philanderers/article/details/_blank)的web框架，而mvc是一个设计模式。图1为springmvc框架架构的原理分析：

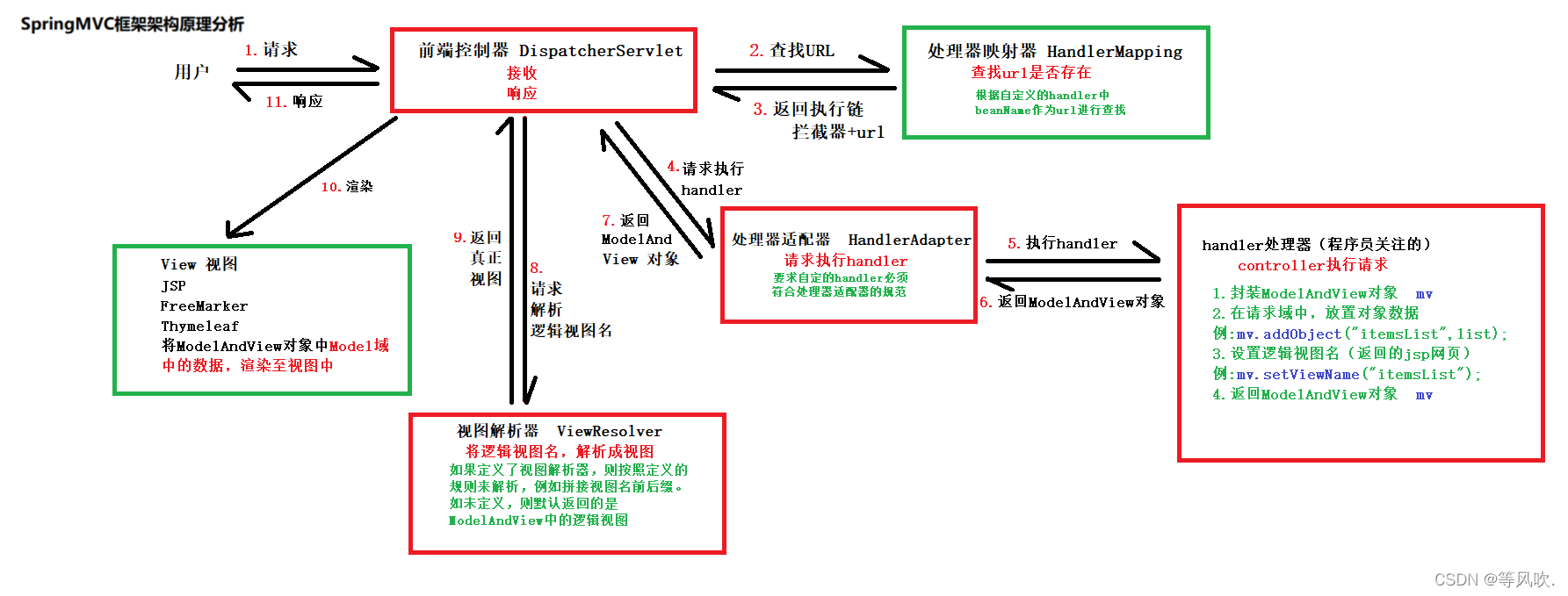


图1 springmvc框架架构原理分析

第一步：发起请求到前端控制器（DispatcherServlet）

第二步：前端控制器请求HandlerMapping查找Handler，可以通过xml配置和注解进行查找

第三步：处理器映射器HandlerMapping向前端控制器返回Handler

第四步：前端控制器调用处理器适配器去执行Handler

第五步：处理器适配器去执行Handler

第六步：Handler执行完成给适配器返回ModelAndView

第七步：处理器适配器向前端控制器返回ModelAndView。ModelAndView是springmvc提供的一个底层对象，包括Model和View

第八步：前端控制器请求视图解析器去进行视图解析。根据逻辑视图名解析成真正的视图

第九步：视图解析器向前端控制器返回View

第十步：前端控制器进行视图渲染。视图渲染将模型数据填充到request域

第十一步：前端控制器向用户响应结果

1. Mvc及其特点

java SpringMVC的工程结构一般来说分为三层，自下而上是Modle层（模型，数据访问层）、Cotroller层（控制，逻辑控制层）、View层（视图，页面显示层），其中Modle层分为两层：dao层、service层，MVC架构分层的主要作用是解耦。采用分层架构的好处，普遍接受的是系统分层有利于系统的维护，系统的扩展。就是增强系统的可维护性和可扩展性。

对于Spring这样的框架，（View\Web）表示层调用控制层（Controller），控制层调用业务层（Service），业务层调用数据访问层（Dao）。图2即为三层架构的具体实现过程：

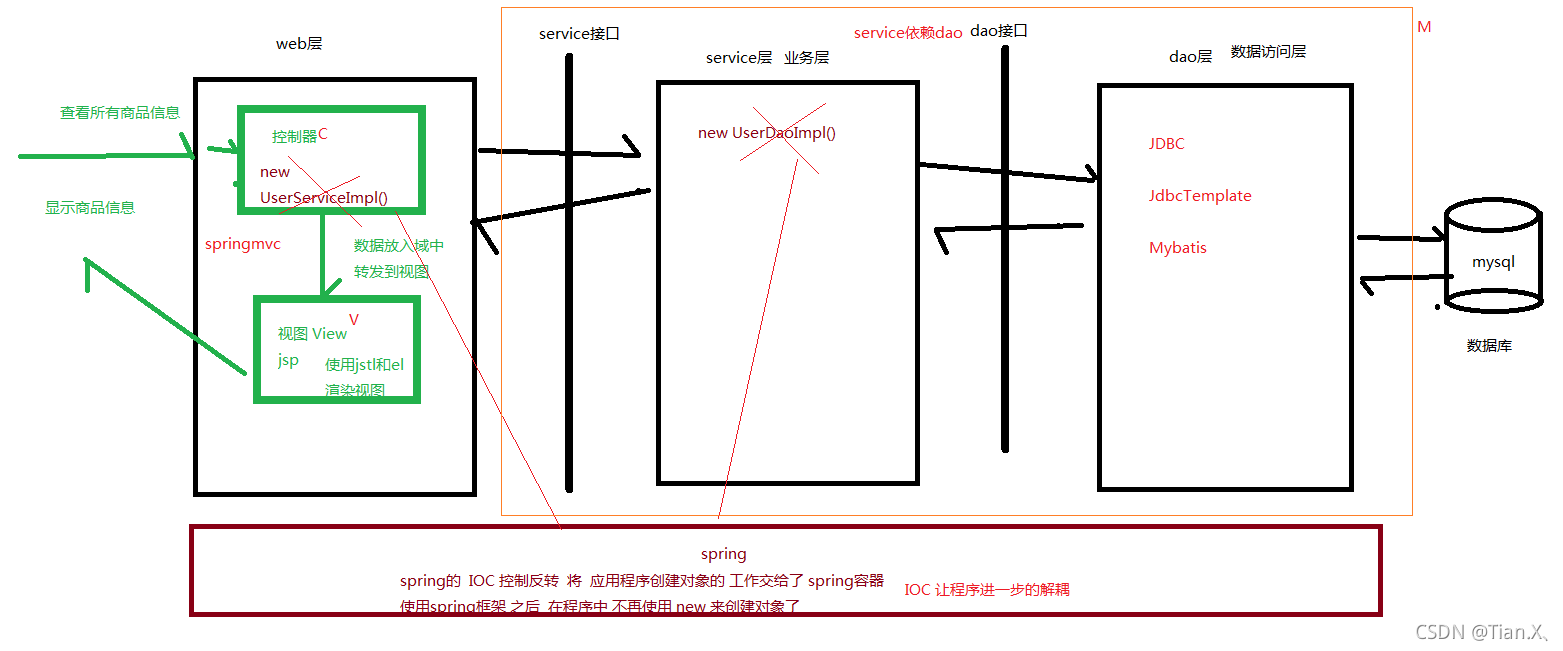


图 2 三层架构图

service层：业务层，用来实现业务逻辑。能调用dao层或者service层，返回数据对象DO或者业务对象BO，BO通常由DO转化、整合而来，可以包含多个DO的属性，也可以是只包含一个DO的部分属性。通常为了简便，如果无需转化，service也可以直接返回DO。外部调用（HTTP、RPC）方法也在这一层，对于外部调用来说，service一般会将外部调用返回的DTO转化为BO。是专注业务逻辑，对于其中需要的数据库操作，都通过Dao去实现。主要去负责一些业务处理，比如取得连接、关闭数据库连接、事务回滚,一些复杂的逻辑业务处理就放到service层。

DAO层：负责访问数据库进行数据的操作，取得结果集，之后将结果集中的数据取出封装到VO类对象之后返回给service层。数据层，直接进行数据库的读写操作，返回数据对象DO，DO与数据库表一一对应。Dao的作用是封装对数据库的访问：增删改查，不涉及业务逻辑，只是达到按某个条件获得指定数据的要求。

Cotroller层：叫做控制层，主要的功能是处理用户发送的请求。主要处理外部请求。调用service层，将service层返回的BO/DO转化为DTO/VO并封装成统一返回对象返回给调用方。如果返回数据用于前端模版渲染则返回VO，否则一般返回DTO。不论是DTO还是VO，一般都会对BO/DO中的数据进行一些转化和整合。

View层：叫做显示层，主要是负责现实数据。

在实际开发中dao层要先定义出自己的操作标准即标准接口，就是为了解耦合。

1. SpringMvc的优点
   1. 基于 MVC 架构，功能分工明确。解耦合。
   2. 容易理解，上手快；使用简单。就可以开发一个注解的 SpringMVC 项目，SpringMVC 也是轻量级的，jar 很小。不依赖的特定的接口和类。
   3. 作为 Spring 框架一部分，能够使用 Spring 的 IoC 和 Aop。方便整合Strtus，MyBatis，Hiberate, JPA 等其他框架。
   4. SpringMVC 强化注解的使用，在控制器，Service，Dao 都可以使用注解。方便灵活。使用@Controller 创建处理器对象,@Service 创建业务对象，@Autowired 或者@Resource 在控制器类中注入 Service, Service 类中注入Dao。
2. 简单实例应用开发介绍
3. 需求分析：

随着高校的扩招，需要处理的学生信息日趋加大，不仅花费大量的教师资源，处理的效率还十分低下。为提高学生管理的管理水平，优化资源，尽可能降低管理成本成为学生管理的新课题，学生管理系统是从学生管理现状触发，根据学生管理的新要求进行开发设计的，它解决了学生管理数据信息量大，修改不方便，对一系列数据进行分析时花费时间长等问题，帮助学生管理人员有效管理学生信息，成为管理高校中必不可少的管理工具。

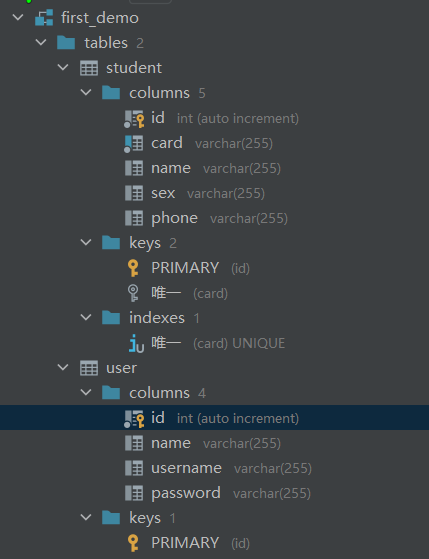
1. 功能：
2. 注册：用户注册信息，输入包括用户名，账号和密码
3. 登录：注册后可直接登陆，通过用户名和密码进行登陆
4. 重新登陆：在进入管理界面后，点击右上角的重新登陆可回到登陆界面
5. 学生管理：可以实现新增学生，查看全部学生，修改学生信息，删除学生，以及搜索学生的功能
6. 显示时间：在网页上显示当时的系统时间，年月日形式
7. 项目设计
8. 框架

使用了springboot框架，前端代码使用freemarker+jQuery+css实现，后端则采用java实现，数据库采用mysql。

1. 业务流程

首先管理员进行登录输入个人的账号信息，系统将用户或系统管理员输入的信息和后台数据库作比对，如果不合法提示用户名或密码错误，需要重新登录，若合法进入系统界面。同时如果没有账号，可以在当前页面进行注册，注册时需输入账号，用户名，密码，同时和后台数据库进行比对，若没用相同的账号即创建成功，回到刚才界面。进入主界面后，可以通过左边的标题进入不同的页面设置，比如首页，删除学生，查询学生等。

1. 数据库设计



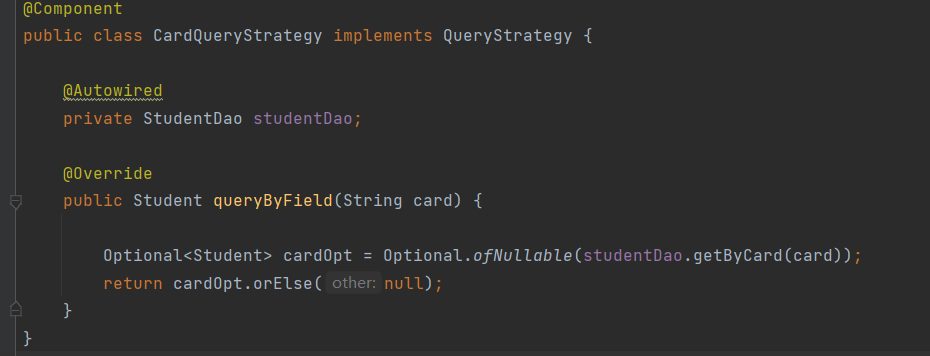
**表1 管理员表**

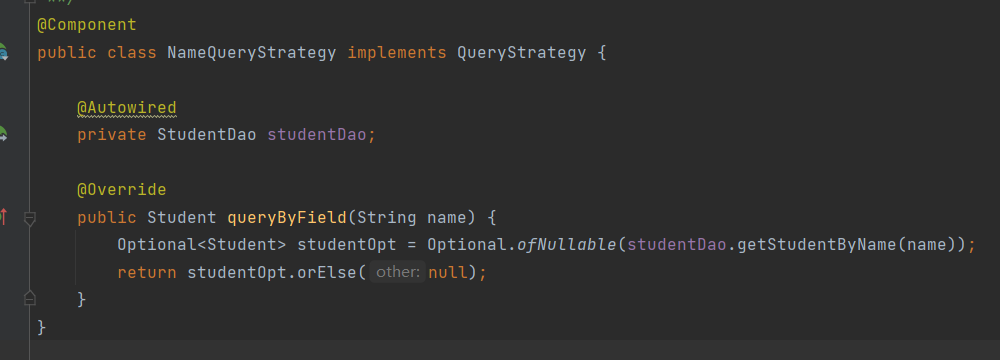
| **列名** | **类型** | **KEY** | **可否为空** | **注释** |
| --- | --- | --- | --- | --- |
| id | int | PRI | 否 | 管理员id |
| admin | varchar(50) |  | 否 | 管理员账号 |
| password | varchar(50) |  | 否 | 管理员密码 |
| realname | varchar(50) |  | 否 | 管理员性别 |
| phone | varchar(50) |  | 否 | 管理员电话 |

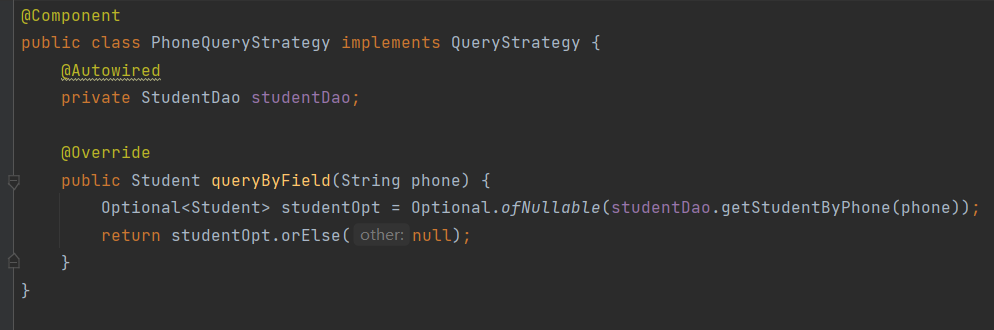
**表2 学生用户表**

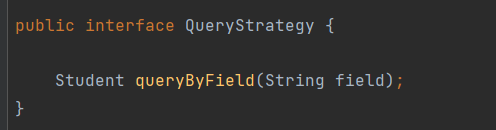
| **列名** | **类型** | **KEY** | **可否为空** | **注释** |
| --- | --- | --- | --- | --- |
| id | int | PRI | 否 | 学生id |
| user | varchar(50) |  | 否 | 学生学号 |
| password | varchar(50) |  | 否 | 学生密码 |
| phone | varchar(50) |  | 否 | 学生电话 |

1. 设计模式:
2. 策略模式的代码分析





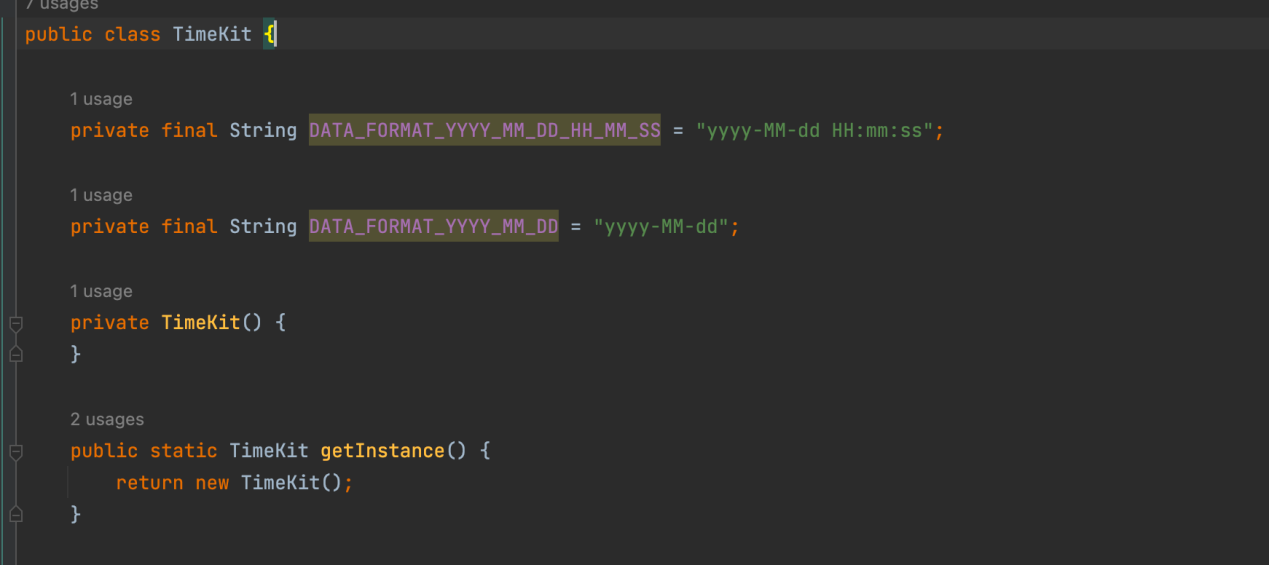


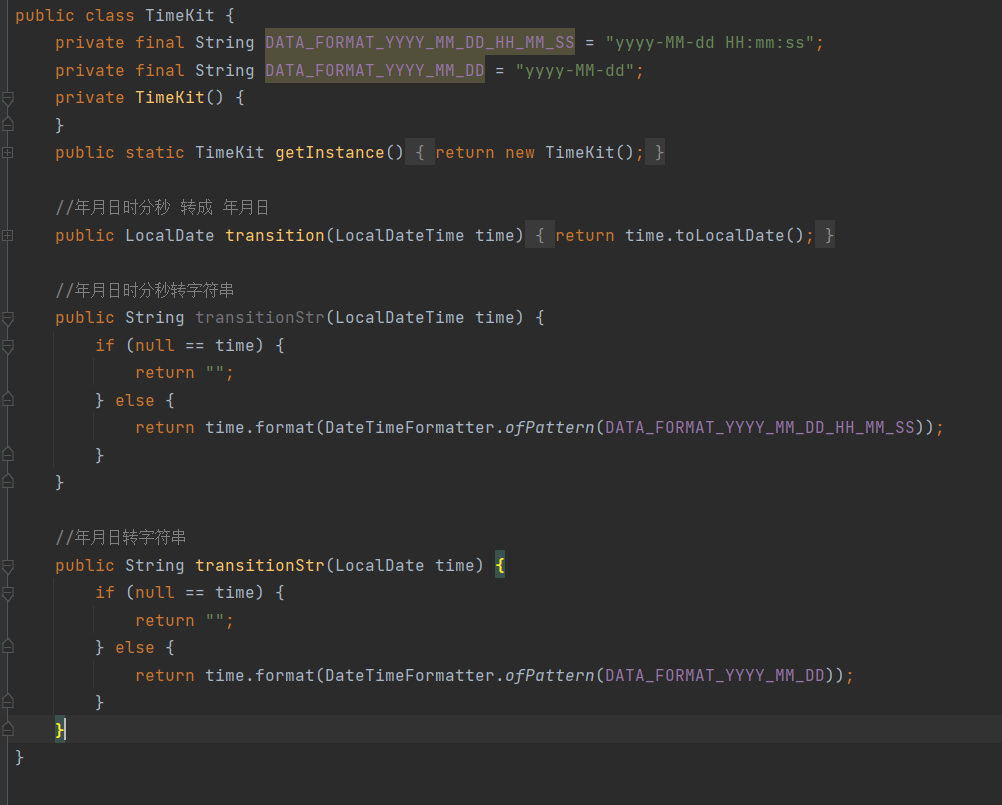


使用场景：此项目在查询学生信息详情处，根据用户选择不同选项(学号,姓名,手机号)查询对应的学生详情信息，出于后期扩展出发点，可能还会有别的查询项，采用策略模式是比较好的出发点。

代码实现方式：此处定义一个QueryStrategy的策略查询接口，里面定义了一个queryByField()方法，方式实现QueryStrategy接口的类去实现具体的查询业务方法，列如CardQueryStrategy策略类根据学号查询的学生信息，NameQueryStrategy策略类根据姓名查询学生信息，PhoneQueryStrategy策略类根据手机号查询学习信息。此处的使用的策略模式管理是借助spring的@Autowired Map<String, QueryStrategy> 该Map集合key值是对应的策略列如(phoneQueryStrategy,nameQueryStrategy,cardQueryStrategy)根据接口的传入的对应策略，获取Map对应的策略方法去查询学生信息

1. 单例模式代码分析



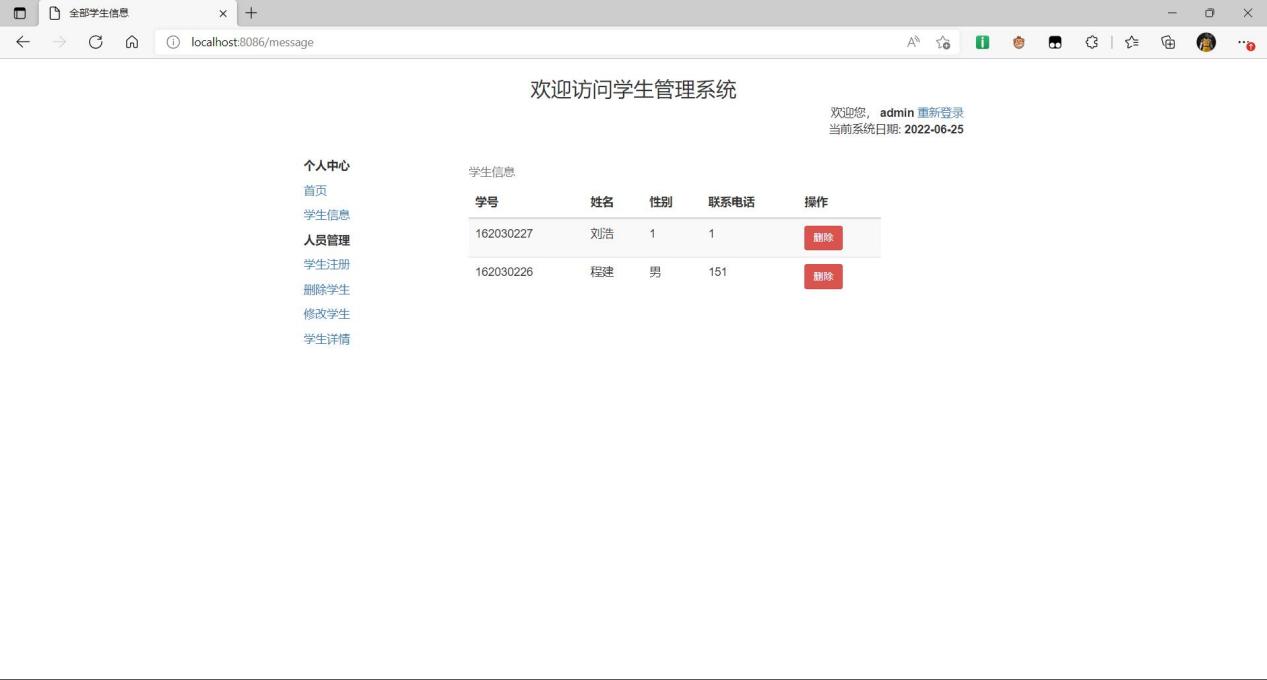
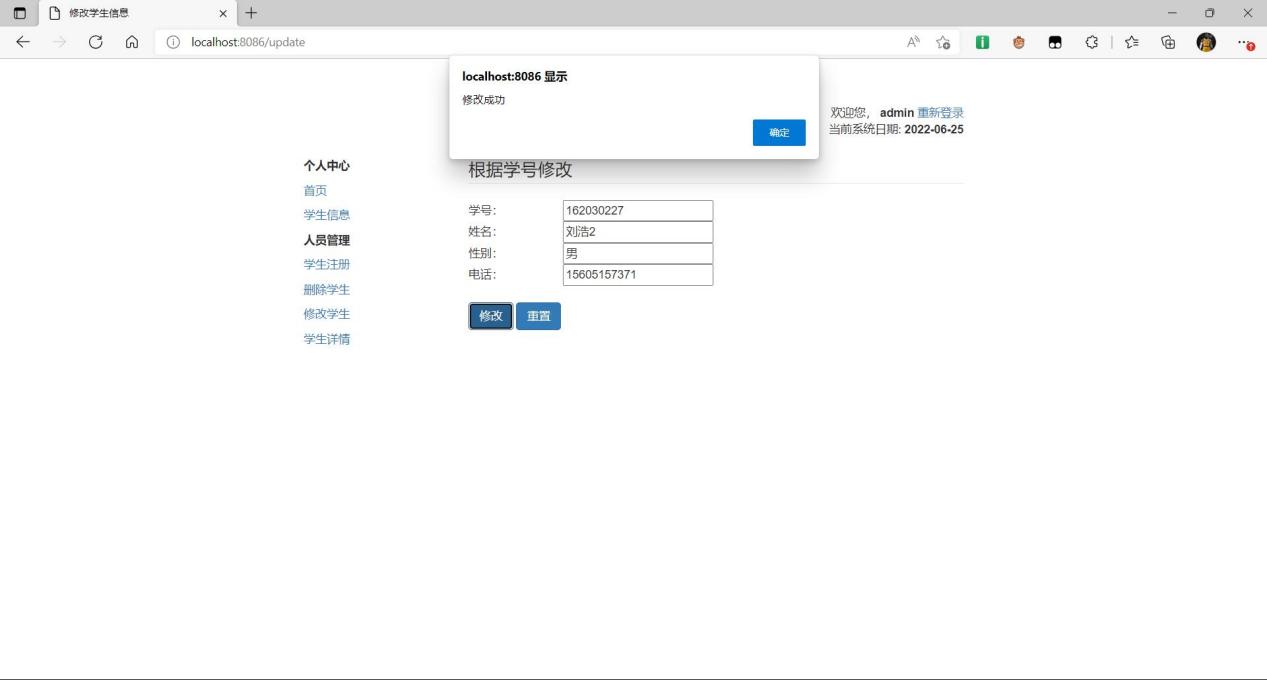


项目里面时间处理类这边采用了单例设计模式，保障了项目里面使用的时间处理对象TimeKit获取的是同一份对象。代码实现方式，将TimeKit的构造器私有化，对外提供一个静态的方法getInstance方法，获取到真实的TimeKit对象。

1. mvc模式

这是由springmvc框架自身体现的特点。

1. 功能实现即截图:



1. 心得体会

在做这一次的java设计模式课程设计之前，我既没有使用过java进行编程，也没有学习过前端的相关知识，整个人就是一个小白。得知这个题目之后，根本不知道从哪开始，先干什么，要干什么，怎么干。不得已，只能去b站上面找一些视频进行学习，开始想先看springmvc的相关内容，发现没有java的基础学这个和看天书一样，所以买了本java编程相关的书籍想着能够速成一下。学了几个星期学会了使用java之后再去学习springmvc相关的知识，发现没有web的经验还是不会写程序，而且没有学过前端，更是前进路上的绊脚石。只能在b站上面先找一些具体项目实践的视频同时网上搜索一些项目的源码，研究一下那些博主是怎样搭建这样的一个项目的，一步步跟着他们实现一个小项目，再去类比到自己要实现的功能上面。最后也是经过好久的努力完成了本次java设计模式，在这次课程设计中，我真的付出了我自己好多血和泪，多少个深夜在电脑前还在学习相关知识。也算是从小白加上了一些斑点吧，了解了web服务基本方法，还有前后端的一些用法收获真的很多。