Student Management Studio V1.0开发文档

# 需求分析

通过网页前端，来实现从网页操作数据库的一个可视化的学生管理系统

**V1.0主要实现的功能如下：**

1. 用户的登录，默认用户名：root，密码：123456
2. 采用拦截器技术防止非法登录
3. 学生基本信息的展示以及增删改查
4. 学生成绩信息的增删改查

**后期V2.0实现的目标：**

1. 用户自定义字典的管理，将学生的部门信息进行字典的保存，在登记学生信息的时候通过下拉选项的输入方式防止用户输入错误，并在数据库中添加触发器以保证数据库信息的准确性和安全性。
2. 用户权限的管理，为用户分配角色，指定系统管理员，为不同的用户呈现不同的数据操作权限，进一步提升数据库的安全和规范性
3. 提供数据库备份方法，可将数据库备份到外接硬盘，保证数据库的安全
4. 根据用户使用习惯进行功能模块和页面跳转逻辑的修改

# 数据库文档

用户表结构(t\_user)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目名 | 字段名 | 数据类型 | 可空 | 说明 |
| 用户id | id | Char(20) | Not null | PK |
| 用户名 | username | Varchar(20) | Not null |  |
| 真实姓名 | realname | Varchar(20) | Not null |  |
| 密码 | password | Char(20) | Not null |  |

学生表结构(t\_stu)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目名 | 字段名 | 数据类型 | 可空 | 说明 |
| 学号 | id | Char(20) | Not null | 由数据库自动递增生成，不可修改，主键，是识别学生的唯一标识符，其格式为：  yyyyddxxx(年级+部门编号+人员编号) |
| 姓名 | name | Varchar(20) | Not null |  |
| 性别 | sex | Char(5) | Not null | “男”“女” |
| 出生时间 | birth | Date | Not null |  |
| 专业 | Major | Char（20） | Not null |  |
| 已修课程数 | course\_num | Int | Default 0 |  |
| 备注 | note | Varchar(500) |  |  |
| 照片 | photo | Varchar(100) |  | 用于存放照片的路径 |

课程表结构(t\_course)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目名 | 字段名 | 数据类型 | 可空 | 说明 |
| 课程号 | course\_id | Char(20) | Not null | PK |
| 课程名 | course\_name | Char(20) | Not null |  |
| 学时 | course\_time | Int | Not null |  |
| 学分 | course\_credit | Int | Not null |  |

课程表结构(t\_score)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 项目名 | 字段名 | 数据类型 | 可空 | 说明 |
| 学号 | id | Char(20) | Not null | 联合FK |
| 课程号 | course\_id | Char(20) | Not null | 联合FK |
| 姓名 | name | Varchar(20) | Not null |  |
| 课程名 | course\_name | Char(20) | Not null |  |
| 成绩 | score | Int(4,1) | Not null | (0~100) |

# 技术介绍

本系统采用当下最流行的**Spring boot**架构设计，同时应用**thymleaf**模版引擎实现前后端分离的效果。

## Spring Boot框架

随着网络技术的不断发展，SSM结构（Spring MVC，Spring，Mybatis）逐步取代了之前流行的SSH结构（Struts-Spring-Hibernate），而Spring boot框架是整合了Spring MVC，Spring，Mybatis三个框架，将之前SSM结构中的复杂的XML配置统统简化，实现了自动配置，用户只需稍作修改就可以将自己的网页项目进行部署。同时在Spring boot中还集成了tomcat，用户无需再手动配置。其将整个网页交互封装为四层：

1、**Service层**：业务层-->控制业务

业务模块的逻辑功能设计，和DAO层一样都是先设计接口，再创建要实现的类，然后在配置文件中进行配置其实现的关联。接下来就可以在service层调用接口进行业务逻辑应用的处理。

2、**Controller层**：控制层-->调用业务层方法来控制业务逻辑

controller层主要调用Service层里面的接口来控制具体的业务流程，控制层的配置也要在配置文件中进行。

Controller和Service的区别是：Controller负责具体的业务模块流程的控制；Service层负责业务模块的逻辑应用设计

3、**Dao层**：持久层，主要与数据库交互

Dao层首先会创建Dao接口，接着就可以在配置文件中定义该接口的实现类；然后就可以在模块中调用Dao的接口进行数据业务的处理，而不用关注此接口的具体实现类是哪一个类，Dao层的数据源和数据库连接的参数都是在配置文件中进行配置的。

4、**Entity层**：实体层-->数据库在项目中的类

主要用于定义与数据库对象应的属性，提供get/set方法,tostring方法,有参无参构造函数。并且将数据库中的数据封装到对象中，以便在其他的类中调用可以按照类的方式去访问和传递数据。

具体的一个web项目中是： Controller层调用了Service层的具体功能方法，Service层调用Dao层的方法，其中调用的参数是使用Entity层进行传递的。

Response

Data

Entity

Request

Entity

SQL

DAO

Service

@Controller

负责接受和处理请求

HTML

Database

## Thymleaf模板引擎

类似于jsp，将@Controller中的数据按照对应的thymleaf语法插入到对应的html标签中。用户无需在html页面中插入大量的Java代码，而仅用th:绑定对应的html属性，在其中放入@{链接，url}，#{取值}，${thymleaf代码}。

## Mybitis

在与数据库交互的阶段，只需要为mybitis提供mapper.xml文件，mybitis会自动解析其中的SQL语句标签，并将其拼接为正确的SQL语句，从而减少了用户在拼接SQL语句中的不必要错误，如下面这段表达式，如果id不存在时，SQL会报错，而<where>标签会识别其中的无效字段，并将其删除，若id不存在，则不会反回where子句，从而减少了SQL语法出错的问题。

<select id=’DAOName’>

Select \* from t\_user

<where>

id=#{id}

</where>

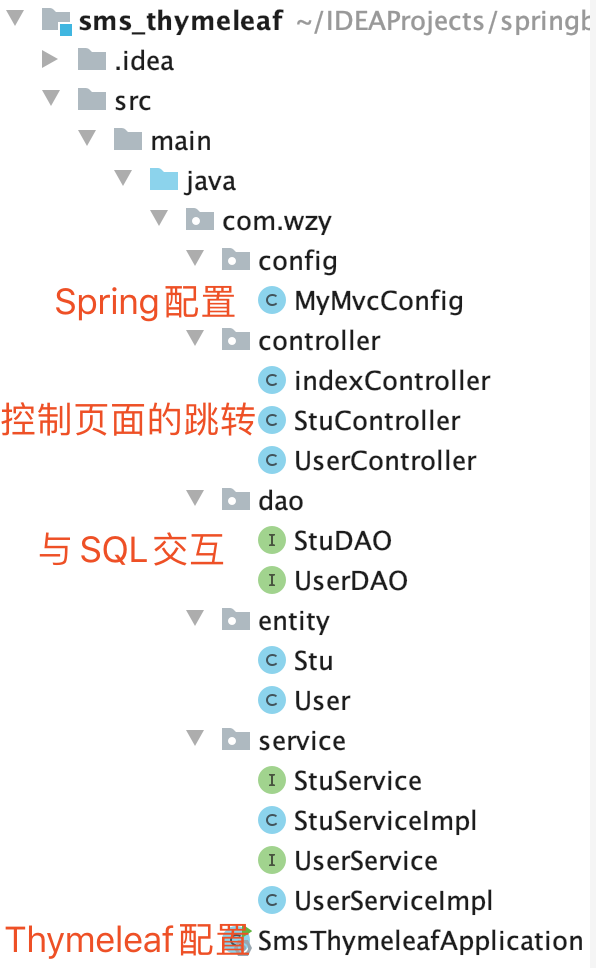
</select>

## Bootstrap

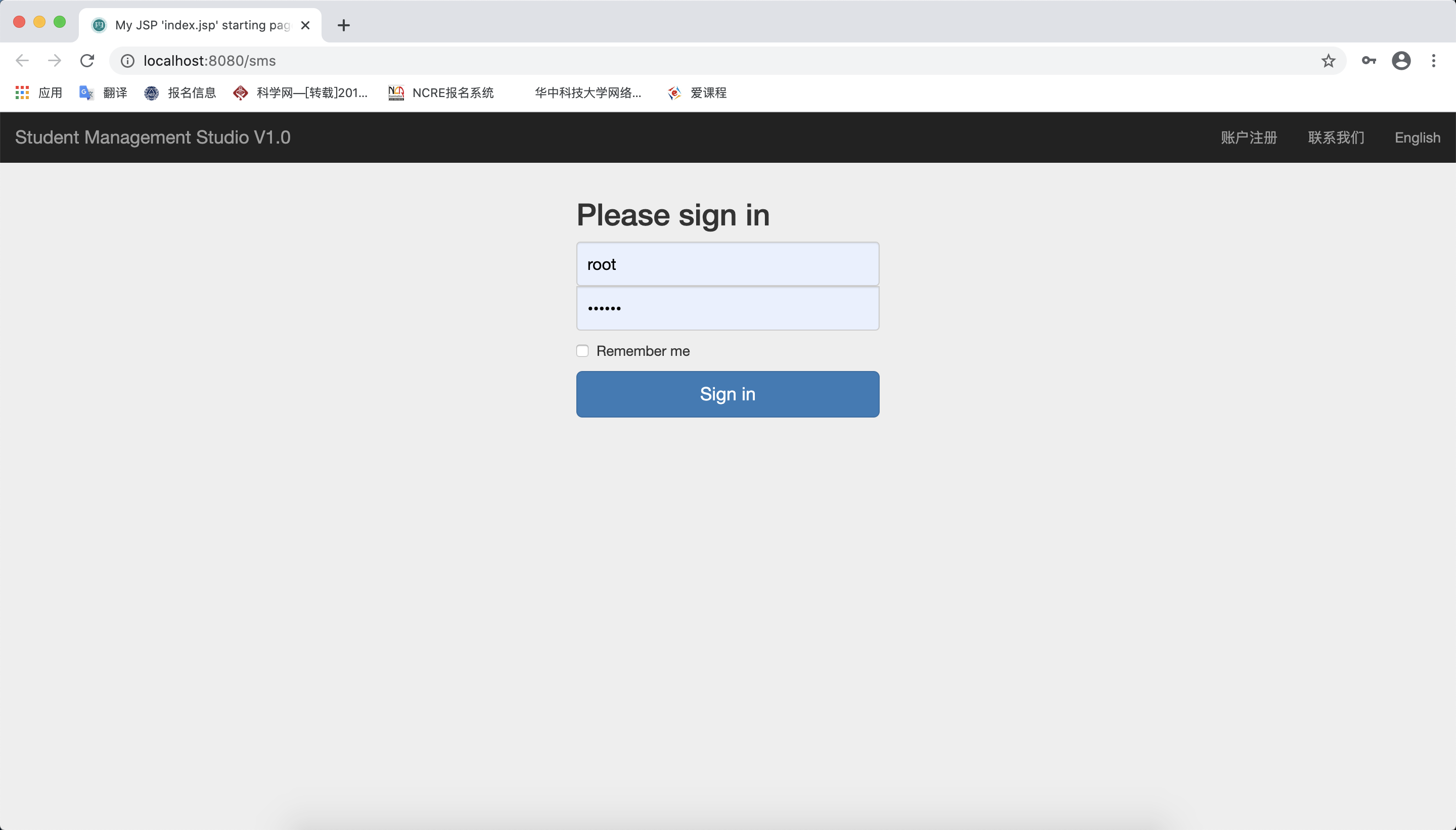
是一种用于快速开发界面的前端框架，其中为用户提供了大量且精美的css样式，同时内置了大量的js功能文件，如表单校验的正则表达式，日期控件等文件，方便用户快速开发网页项目，用于学生管理系统既精美，又不会让程序员投入大量的时间去调试css样式。

# 产品概览

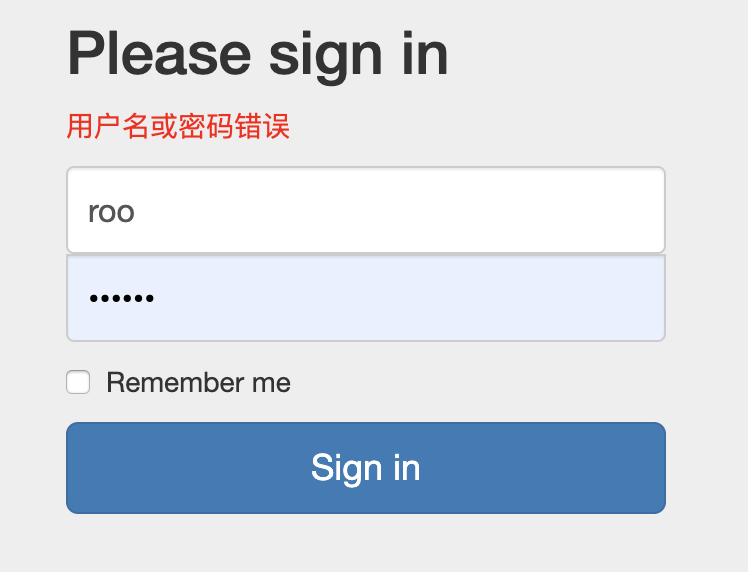
为了说明，项目文件结构如下图所示：

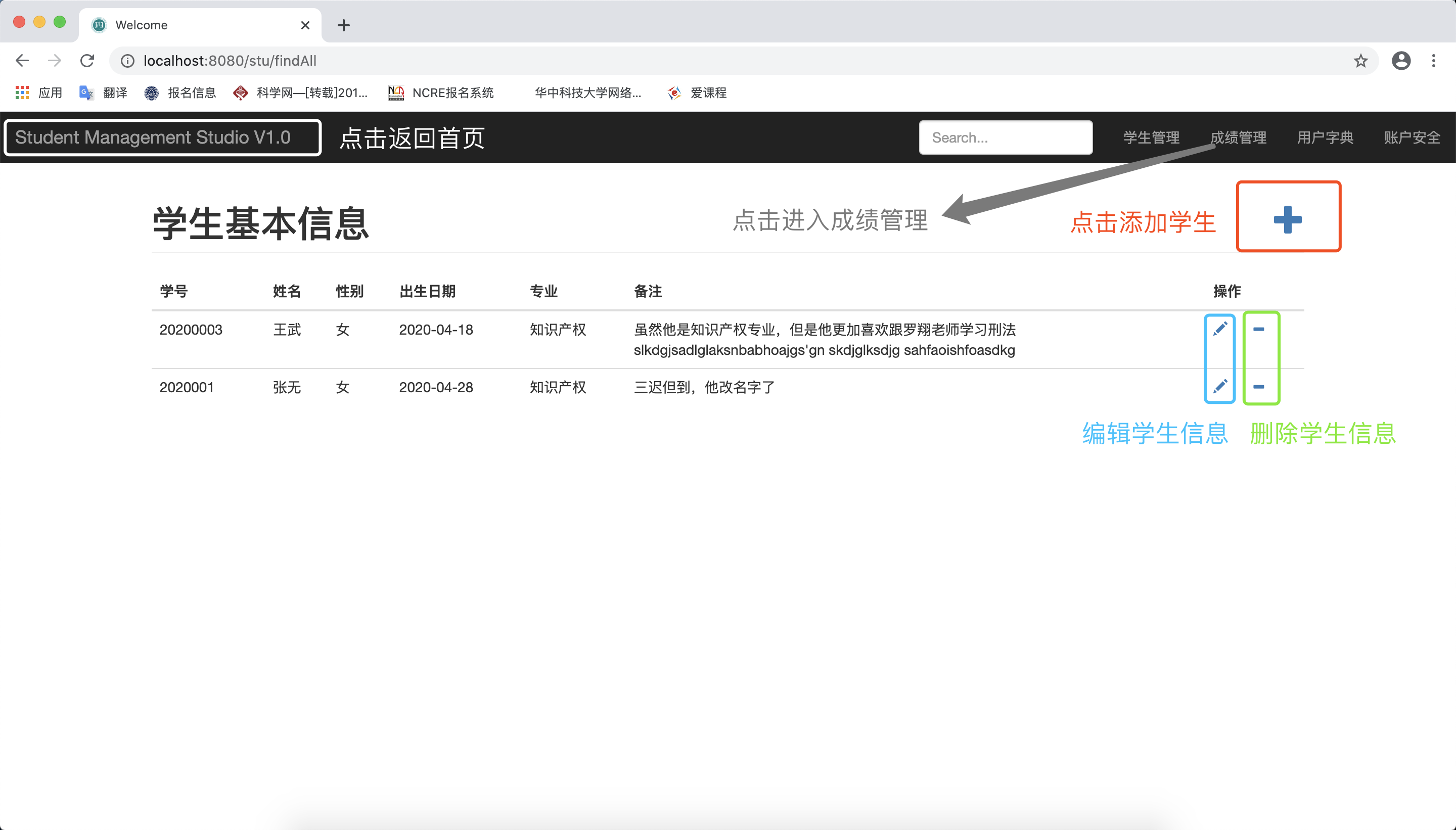
1. 访问项目名localhost：8080/sms ，@Controller跳转到login.html用于用户登录



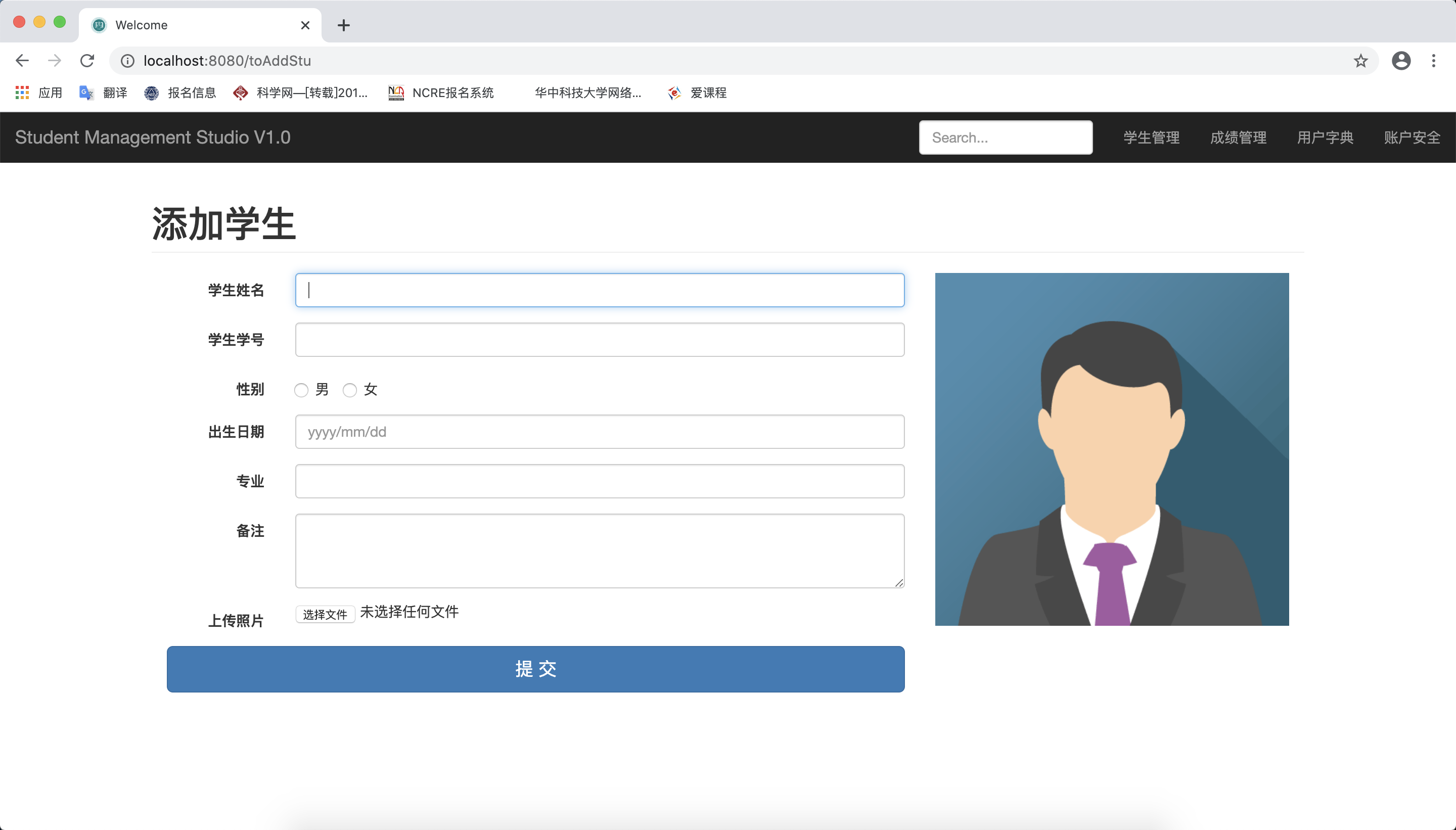
1. 输入正确的用户名或密码登录，否则会提示错误



1. 同时为了防止直接通过域名直接进入后台页面，设置拦截器阻止非法访问
2. 正确登录后进入到stuList.html页面，点击相应的按钮进行操作



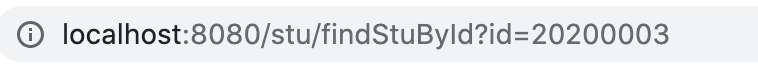
1. 点击“+“添加学生，后台控制器接受跳转请求，跳转页面到addStu.html



同时该界面的表填提交会自动检验用户字段填写是否符合要求，若不符合要求会在前端进行拦截，防止对后台服务器的不必要提交，从而减轻服务器的负担。



1. 点击学生后面的操作进行编辑学生信息，通过控制器跳转到updateStu.html，并且同时将对应学生的信息通过Get请求传递，从服务器取得信息后填写入对应的字段中。同时因为id为学生的唯一标识符，在修改过程中不允许修改





1. 点击“-”删除学生，在删除学生之前会进行弹窗提醒以便确认，防止用户误操作



1. 在进行完增删改三种操作之后，@Controller将重定向到stu/FindAll请求，再次执行对数据库的查询所有列表进行查询。以便更新结果。