学生系统管理平台

一,项目概述

一个基于*SSM*的学生管理系统

数据库中默认的管理员身份信息 *：账户名 ：多瑞c*, 密码 `1234`

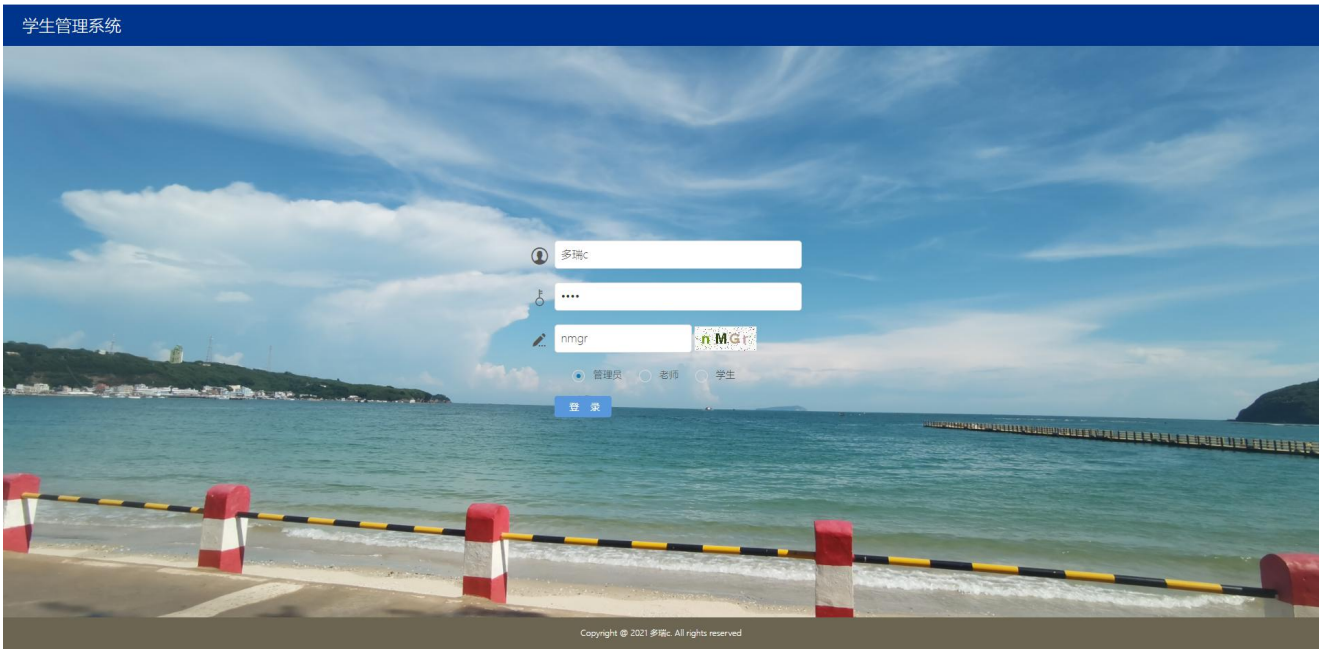
1.1 开发环境

工具	版本或描述
`OS`	Windows 10
`JDK`	1.8
`IDE`	IntelliJ IDEA 2019.3.2
`Maven`	3.5.2
`MySQL`	5.7.19

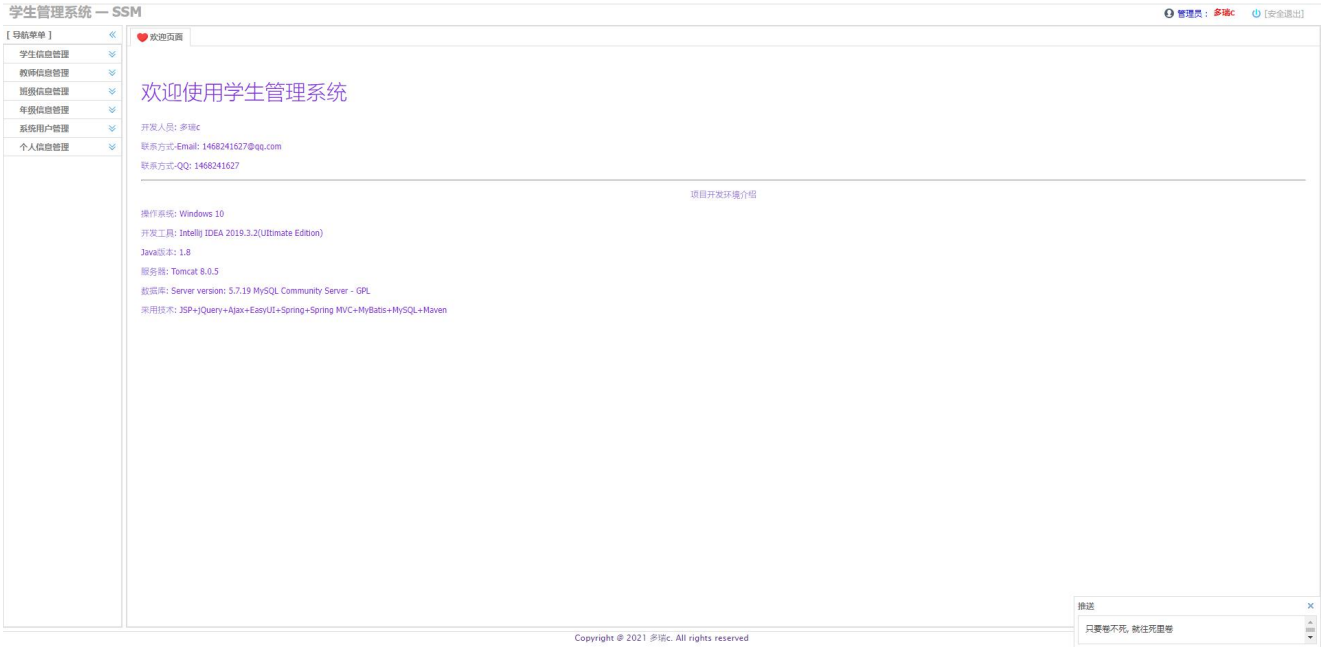
1.2 用户权限介绍

- 管理员 ： 具有所有管理模块的操控权限
- 教师 ： 仅具有学生信息管理模块的所有权限,且在教师信息管理模块中只具有查询及添加信息的权限
- 学生 ： 仅具有学生信息管理模块的查询及添加信息的权限

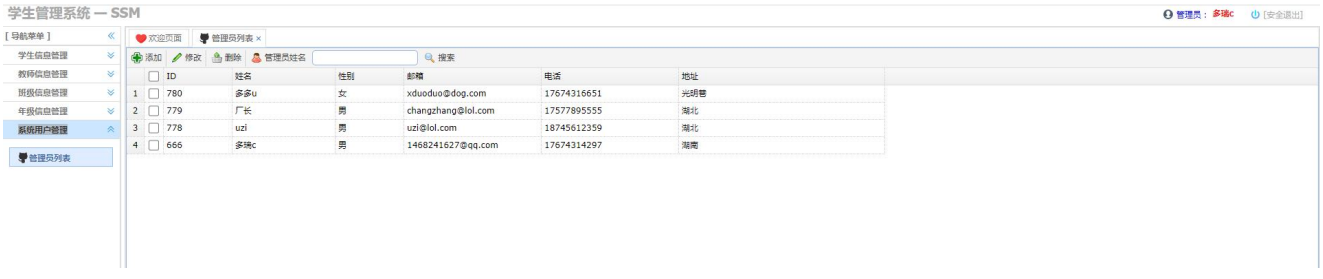
1.3 项目截图 (管理员身份登录)



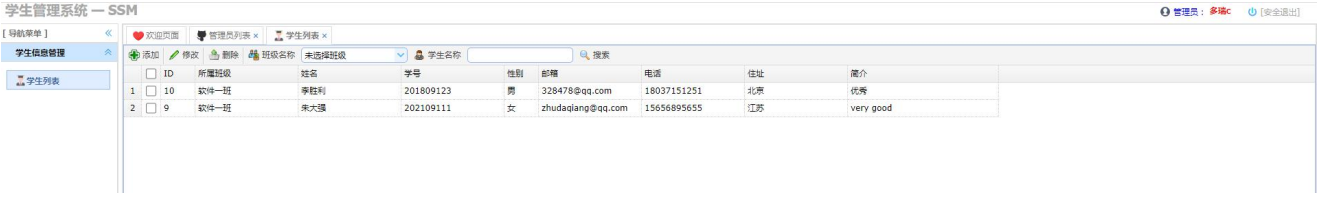
系统主页面



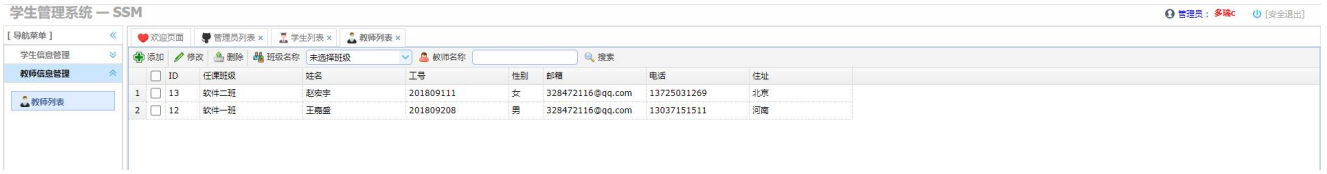
管理员信息管理页面



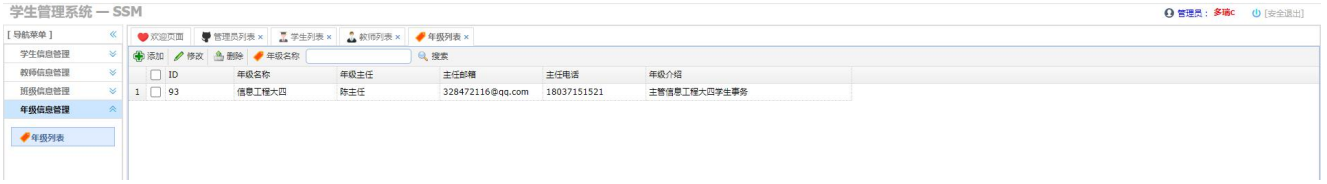
学生信息管理页面



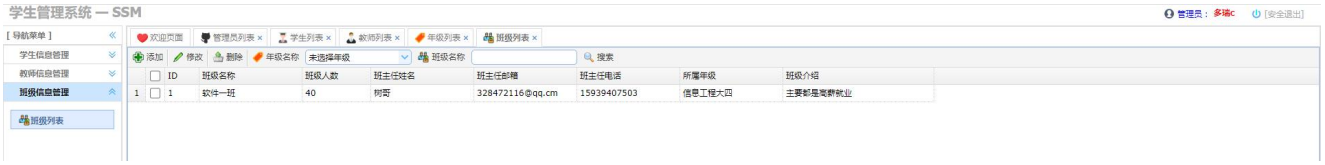
教师信息管理页面



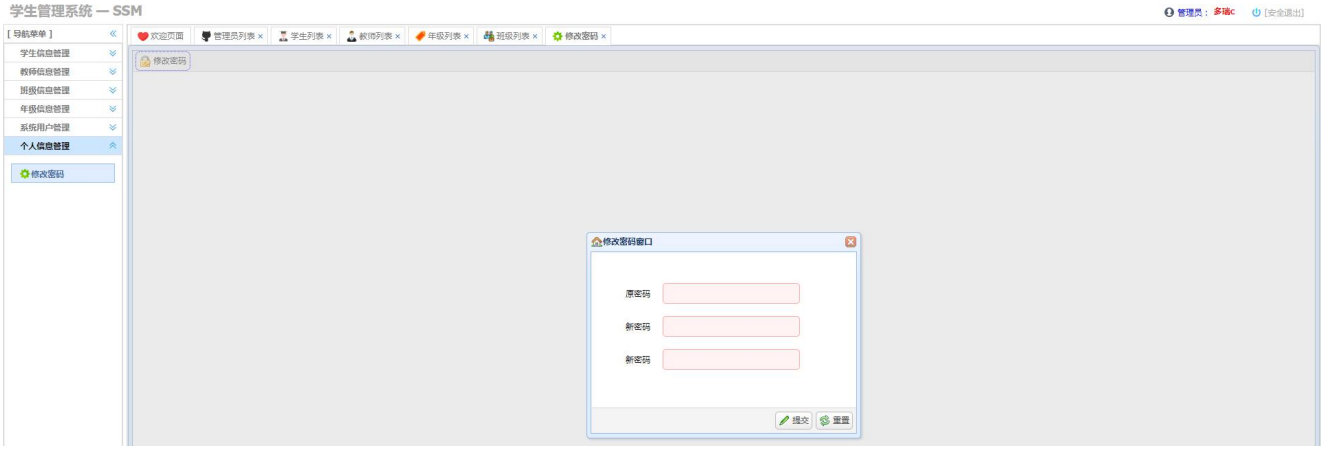
年级信息管理页面



班级信息管理页面

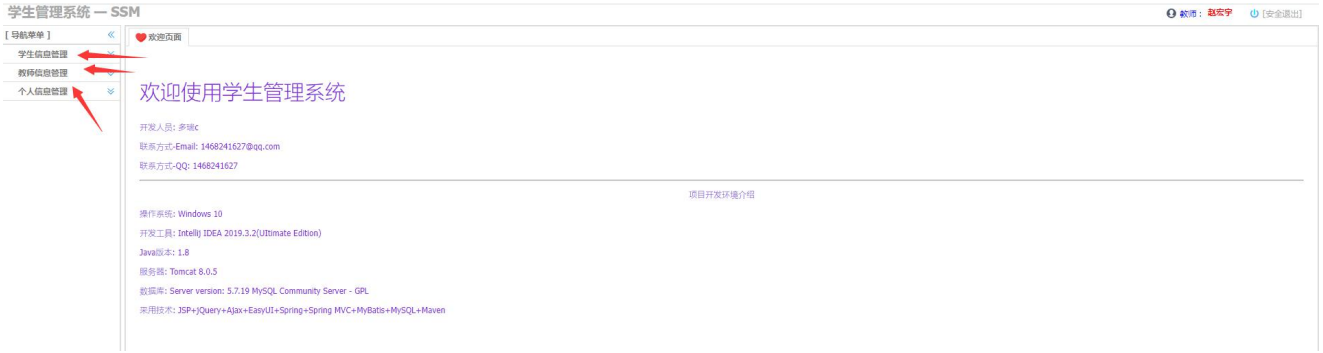


个人信息管理页面



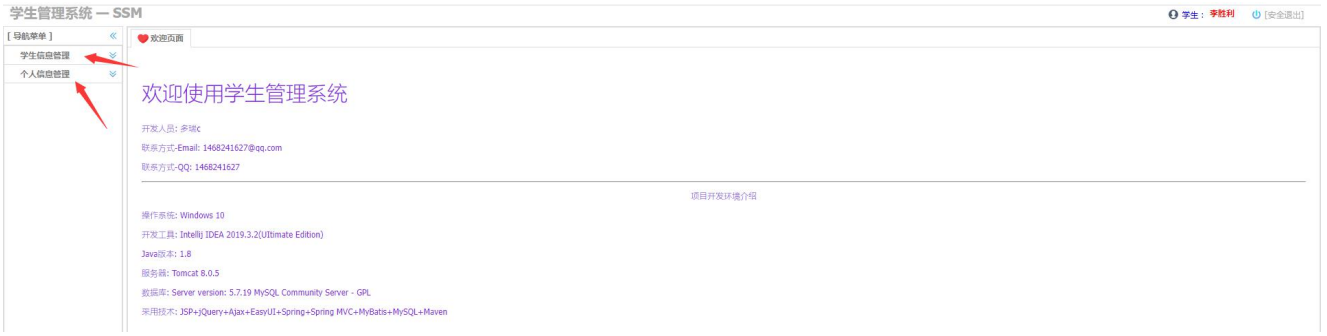
二,项目截图 (教师身份登录)

教师仅具有学生信息管理模块的所有权限,且在教师信息管理模块中只具有查询及添加信息的权限



三,项目截图 (学生身份登录)

学生仅具有学生信息管理模块的查询及添加信息的权限



四,项目结构

```
| .gitattributes
| LICENSE
| README.md
|
├─database file
| ssm_sms.sql
|
├─demonstration_picture
| SMS-ClassInfo-view.PNG
| SMS-GradeInfo-view.PNG
| SMS-Login-view.PNG
| SMS-ModifyPwd-view.PNG
| SMS-Student-permission.PNG
| SMS-StudentInfo-view.PNG
| SMS-Teacher-permission.PNG
| SMS-TeacherInfo-view.PNG
| SSM-AdminInfo-view.PNG
| SSM-Main-view.PNG
|
└─sms
```

```
21  | pom.xml
22  |
23  |
24  └─src
25  └─main
26  ├─java
27  | └─com
28  | └─cc
29  | └─sms
30  | ├─bean
31  | | Admin.java
32  | | Clazz.java
33  | | Grade.java
34  | | LoginForm.java
35  | | Student.java
36  | | Teacher.java
37  | |
38  | ├─controller
39  | | AdminController.java
40  | | ClazzController.java
41  | | CommonController.java
42  | | GradeController.java
43  | | StudentController.java
44  | | SystemController.java
45  | | TeacherController.java
46  | |
47  | ├─dao
48  | | AdminMapper.java
49  | | ClazzMapper.java
50  | | GradeMapper.java
51  | | StudentMapper.java
52  | | TeacherMapper.java
53  | |
54  | ├─interceptor
55  | | LoginInterceptor.java
56  | |
57  | ├─service
58  | | | AdminService.java
59  | | | ClazzService.java
60  | | | GradeService.java
```

```
61  │ │ │ StudentService.java
62  │ │ │ TeacherService.java
63  │ │ │
64  │ │ └─impl
65  │ │ AdminServiceImpl.java
66  │ │ ClazzServiceImpl.java
67  │ │ GradeServiceImpl.java
68  │ │ StudentServiceImpl.java
69  │ │ TeacherServiceImpl.java
70  │ │
71  │ └─util
72  │ CreateVerifiCodeImage.java
73  │ UploadFile.java
74  │
75  ├─resource
76  │ ├─database-conf
77  │ │ c3p0.properties
78  │ │
79  │ ├─mapper
80  │ │ AdminMapper.xml
81  │ │ ClazzMapper.xml
82  │ │ GradeMapper.xml
83  │ │ StudentMapper.xml
84  │ │ TeacherMapper.xml
85  │ │
86  │ ├─mybatis-conf
87  │ │ mybatis-config.xml
88  │ │
89  │ └─spring-conf
90  │ applicationContext.xml
91  │ springmvc-config.xml
92  │
93  └─webapp
94  │ index.jsp
95  │
96  ├─image
97  │ └─portrait
98  │ default_admin_portrait.png
99  │ default_student_portrait.png
100  │ default_teacher_portrait.png
```

```
101 |
102 ├─static
103 | ├─easyui
104 | | |
105 | | ├─css
106 | | |
107 | | ├─js
108 | | |
109 | | └─themes
110 | |
111 | |
112 | └─h-ui
113 | |(略..)
114 |
115 |
116 |
117 └─WEB-INF
118 | web.xml
119 |
120 └─view
121 ├─admin
122 | adminList.jsp
123 |
124 ├─clazz
125 | clazzList.jsp
126 |
127 ├─common
128 | settings.jsp
129 |
130 ├─error
131 | 404.jsp
132 | 500.jsp
133 |
134 ├─grade
135 | gradeList.jsp
136 |
137 ├─student
138 | studentList.jsp
139 |
140 ├─system
```

```
141  | intro.jsp
142  | login.jsp
143  | main.jsp
144  |
145  └─teacher
146  teacherList.jsp
```

五,项目文件说明-

数据库文件

```
1  ssm_sms.sql
```

项目文件说明-数据库配置信息

```
1  c3p0.properties
```

项目文件说明-H-ui 前端框架

```
1  h-ui/
```

项目文件说明-EasyUI 前端框架

```
1  easyui/
```

项目文件说明-Spring 核心配置文件

```
1  applicationContext.xml
```

项目文件说明-Spring MVC 核心配置文件

```
1  springmvc-config.xml
```

项目文件说明-MyBatis 核心配置文件

```
1  mybatis-config.xml
```

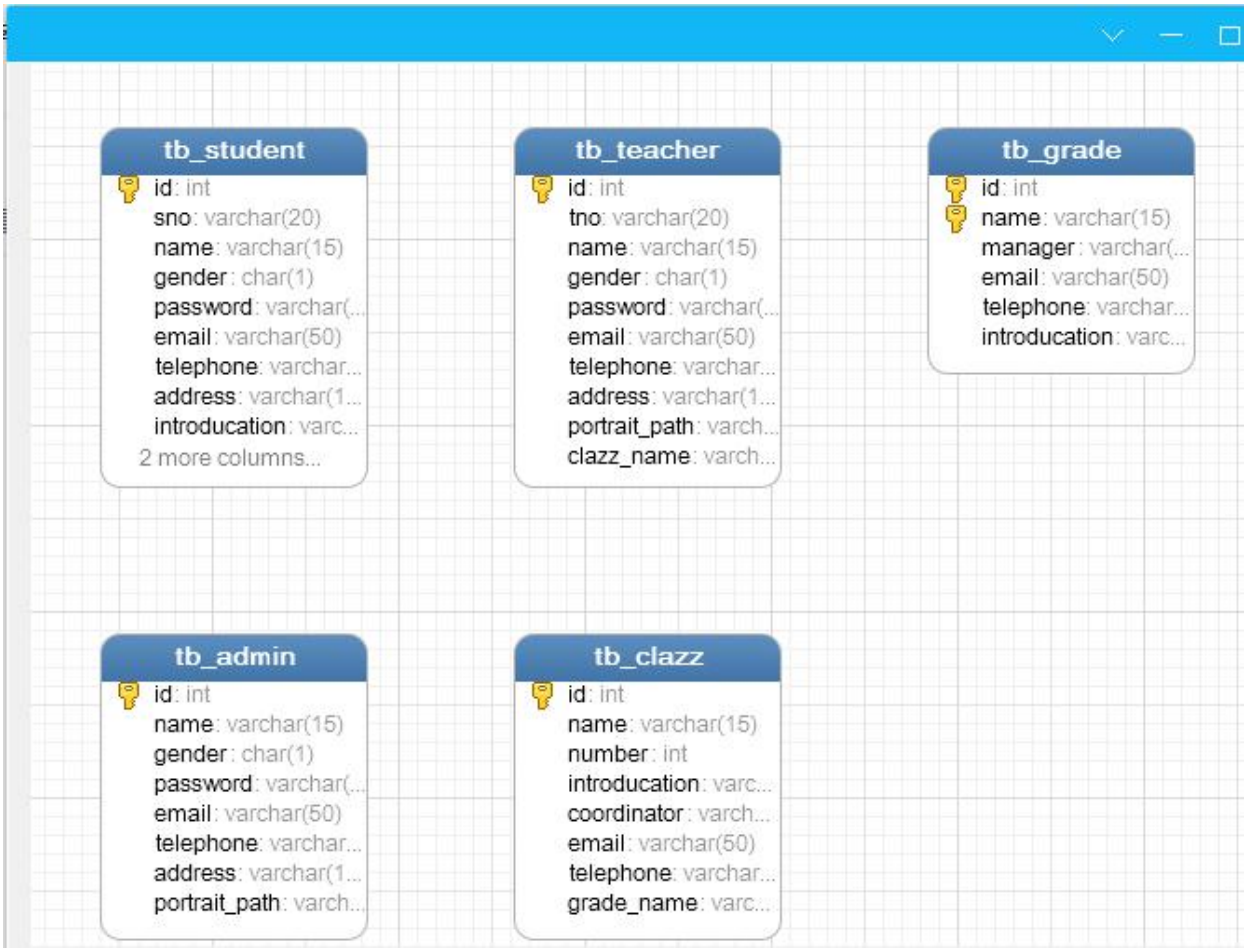
项目文件说明-Mapper 接口映射文件

```
1  mapper/
```

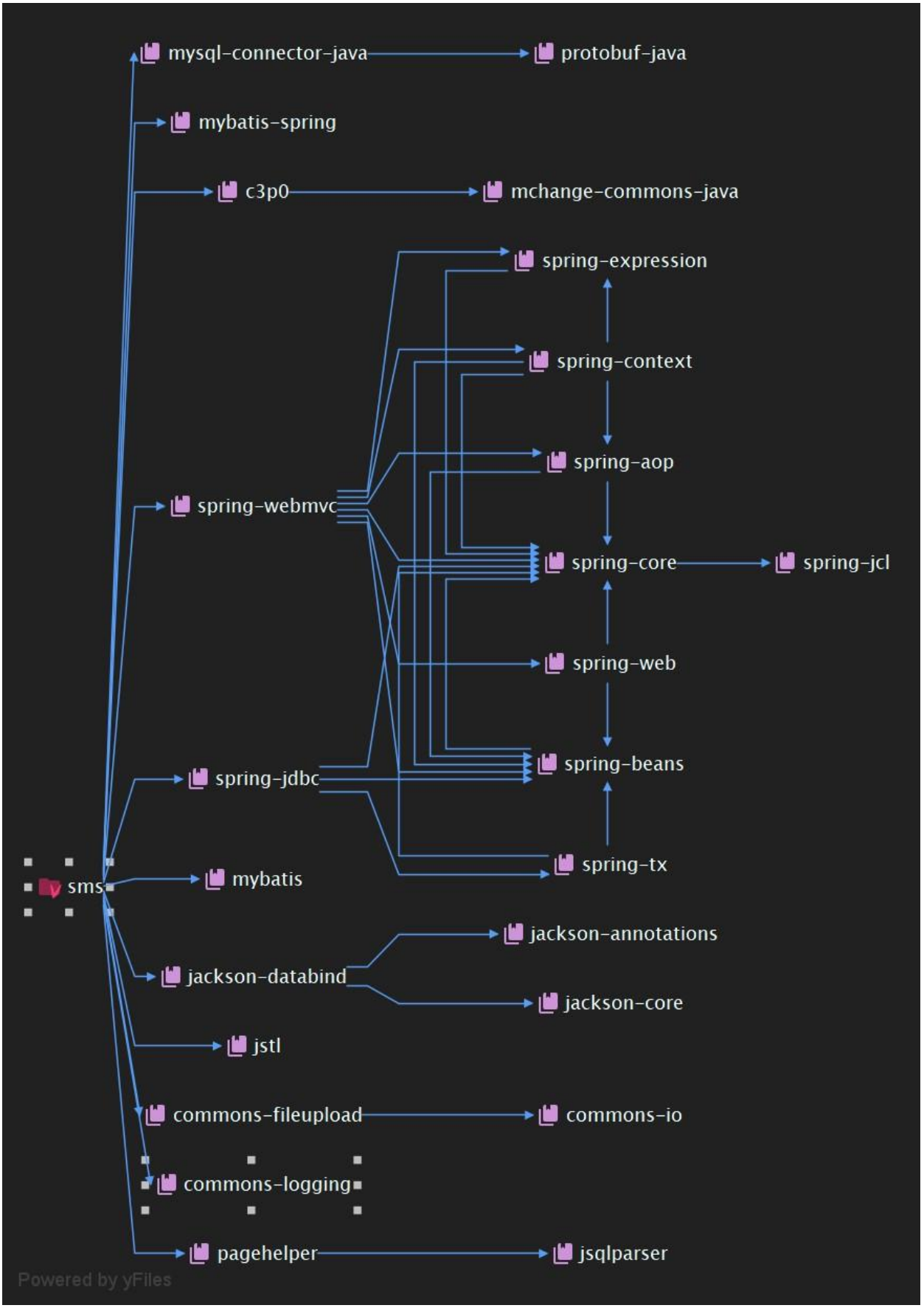
项目文件说明-用户默认头像

```
1  portrait/
```

数据库ER图

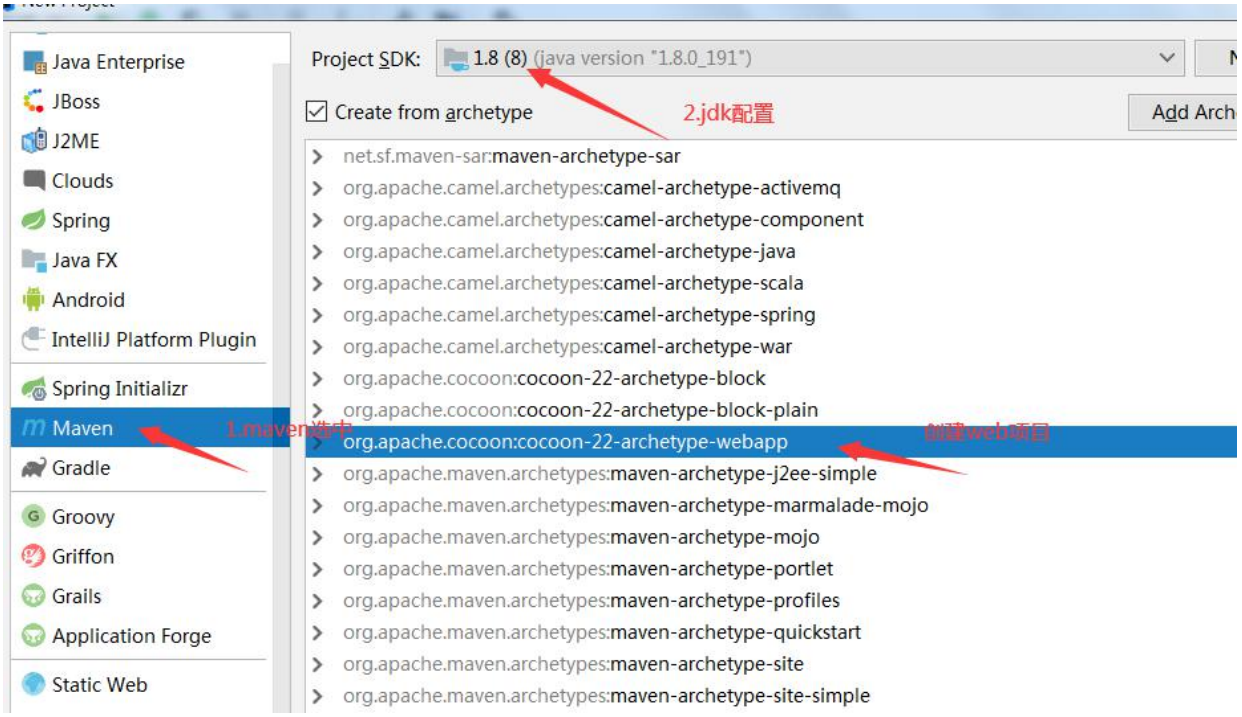

tb_student
id: int
sno: varchar(20)
name: varchar(15)
gender: char(1)
password: varchar(...
email: varchar(50)
telephone: varchar...
address: varchar(1...
introducation: varc...
2 more columns...
tb_teacher
id: int
tno: varchar(20)
name: varchar(15)
gender: char(1)
password: varchar(...
email: varchar(50)
telephone: varchar...
address: varchar(1...
portrait_path: varch...
clazz_name: varch...
tb_grade
id: int
name: varchar(15)
manager: varchar(...
email: varchar(50)
telephone: varchar...
introducation: varc...
tb_admin
id: int
name: varchar(15)
gender: char(1)
password: varchar(...
email: varchar(50)
telephone: varchar...
address: varchar(1...
portrait_path: varch...
tb_clazz
id: int
name: varchar(15)
number: int
introducation: varc...
coordinator: varch...
email: varchar(50)
telephone: varchar...
grade_name: varc...

Jar包依赖关系图

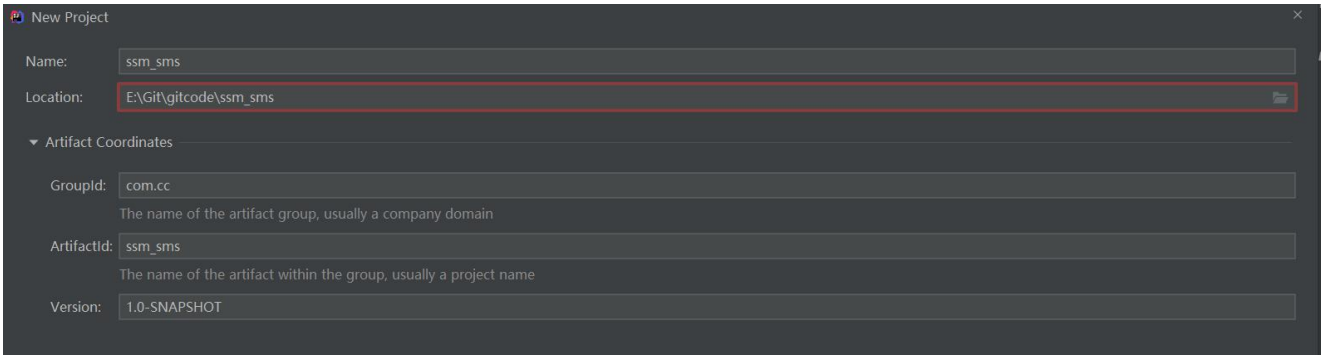

mysql-connector-java
protobuf-java
mybatis-spring
c3p0
mchange-commons-java
spring-expression
spring-context
spring-aop
spring-webmvc
spring-core
spring-jcl
spring-web
spring-beans
spring-jdbc
spring-tx
sms
mybatis
jackson-annotations
jackson-databind
jackson-core
jstl
commons-fileupload
commons-io
commons-logging
pagehelper
jsqlparser
Powered by yFiles

七,创建maven项目

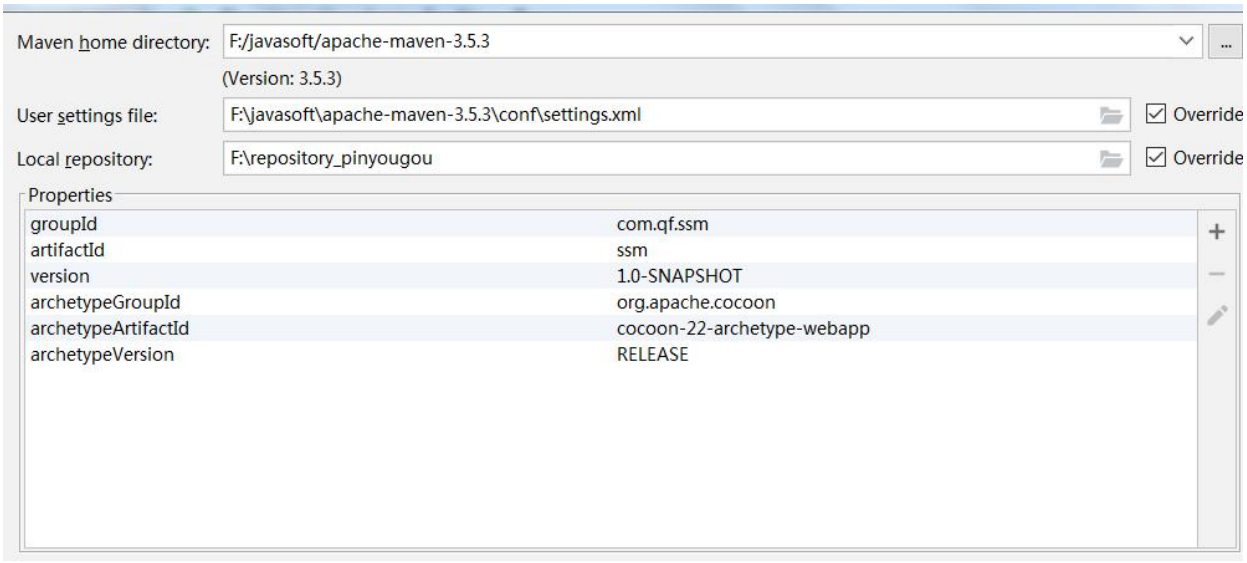
7.1.选择maven选项，选择JDK版本，勾选图示所示的“Create From Archetype”复选框，选择maven模板，点击【Next】



7.2写maven的坐标，“groupId”，“artifactId”，以及“version”，其中groupId是公司域名的反写，而artifactId是项目名或模块名，version就是该项目或模块所对应的版本号，填写完之后，点击【Next】

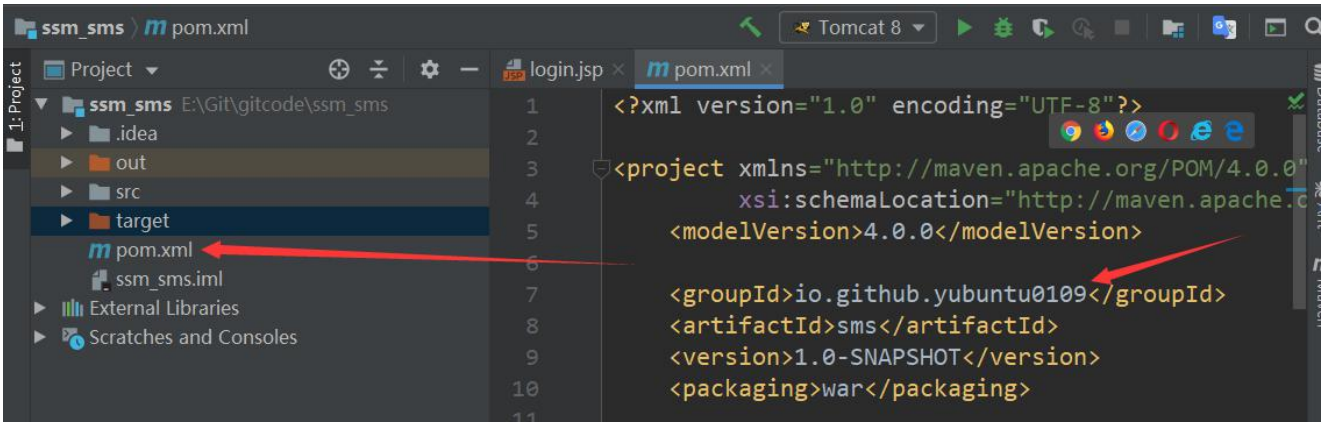


7.3.选择本地的maven，选择maven得配置文件路径及本地maven仓库路径，选择完成后点击【Next】



7.4 填写项目名，后点击【Finish】

7.5 会maven的主页面，maven会自动下载maven依赖，当所有的都自动完成后，创建的maven项目结构如下所示：



7.6导入项目所需要依赖pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <name>ssm</name>
  <groupId>io.github.yubuntu0109</groupId>
  <artifactId>ssm</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>war</packaging>

```

```
<!-- FIXME change it to the project's website -->
<url>http://www.example.com</url>

<properties>
<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
<project.build.spring.webmvc.version>5.1.7.RELEASE</project.build.spring.webmvc.version>
<project.build.spring.webmvc.file.version>1.4</project.build.spring.webmvc.file.version>
<project.build.spring.webmvc.json.version>2.9.10</project.build.spring.webmvc.json.version>
<project.build.spring.jdbc.version>5.1.7.RELEASE</project.build.spring.jdbcersion>
<project.build.mybaits.version>3.5.1</project.build.mybaits.version>
<project.build.myabtis.pagehelper.version>5.1.9</project.build.myabtis.pagehelper.version>
<project.build.mybatis.spring.version>2.0.1</project.build.mybatis.spring.version>
<project.build.servlet.version>4.0.1</project.build.servlet.version>
<project.build.jstl.version>1.2</project.build.jstl.version>
<project.build.mysql.jdbc.version>8.0.16</project.build.mysql.jdbc.version>
<project.build.mysql.c3p0.version>0.9.5.4</project.build.mysql.c3p0.version>
<project.build.logging.version>1.2</project.build.logging.version>
<maven.compiler.source>1.8</maven.compiler.source>
<maven.compiler.target>1.8</maven.compiler.target>
</properties>

<dependencies>
<!-- spring mvc -->
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-webmvc</artifactId>
<version>${project.build.spring.webmvc.version}</version>
</dependency>
<!-- Spring mvc:JSON -->
<dependency>
<groupId>com.fasterxml.jackson.core</groupId>
<artifactId>jackson-databind</artifactId>
```

```
<version>${project.build.spring.webmvc.json.version}</version>
</dependency>
<!-- Spring mvc:file -->
<dependency>
<groupId>commons-fileupload</groupId>
<artifactId>commons-fileupload</artifactId>
<version>${project.build.spring.webmvc.file.version}</version>
</dependency>
<!-- Spring JDBC -->
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-jdbc</artifactId>
<version>${project.build.spring.jdbc.version}</version>
</dependency>
<!-- MyBatis -->
<dependency>
<groupId>org.mybatis</groupId>
<artifactId>mybatis</artifactId>
<version>${project.build.mybaits.version}</version>
</dependency>
<!-- MyBatis 分页插件 -->
<dependency>
<groupId>com.github.pagehelper</groupId>
<artifactId>pagehelper</artifactId>
<version>${project.build.myabtis.pagehelper.version}</version>
</dependency>
<!-- Mybatis与Spring的整合包 -->
<dependency>
<groupId>org.mybatis</groupId>
<artifactId>mybatis-spring</artifactId>
<version>${project.build.mybatis.spring.version}</version>
</dependency>
<!-- Servelt API -->
<dependency>
<groupId>javax.servlet</groupId>
<artifactId>javax.servlet-api</artifactId>
<version>${project.build.servlet.version}</version>
<scope>provided</scope>
</dependency>
<!-- JSTL -->
```

```
81  <dependency>
82  <groupId>javax.servlet</groupId>
83  <artifactId>jstl</artifactId>
84  <version>${project.build.jstl.version}</version>
85  </dependency>
86  <!-- 数据库驱动 -->
87  <dependency>
88  <groupId>mysql</groupId>
89  <artifactId>mysql-connector-java</artifactId>
90  <version>${project.build.mysql.jdbc.version}</version>
91  </dependency>
92  <!-- 数据库连接池 -->
93  <dependency>
94  <groupId>com.mchange</groupId>
95  <artifactId>c3p0</artifactId>
96  <version>${project.build.mysql.c3p0.version}</version>
97  </dependency>
98  <!-- 日志组件 -->
99  <dependency>
100  <groupId>commons-logging</groupId>
101  <artifactId>commons-logging</artifactId>
102  <version>${project.build.logging.version}</version>
103  </dependency>
104  </dependencies>
105
106  <build>
107  <finalName>sms</finalName>
108  <pluginManagement><!-- lock down plugins versions to avoid using Maven
defaults (may be moved to parent pom) -->
109  <plugins>
110  <plugin>
111  <artifactId>maven-clean-plugin</artifactId>
112  <version>3.1.0</version>
113  </plugin>
114  <!-- see http://maven.apache.org/ref/current/maven-core/default-bindin
s.html#Plugin_bindings_for_war_packaging -->
115  <plugin>
116  <artifactId>maven-resources-plugin</artifactId>
117  <version>3.0.2</version>
118  </plugin>
119  <plugin>
```

```
<artifactId>maven-compiler-plugin</artifactId>
<version>3.8.0</version>
</plugin>
<plugin>
<artifactId>maven-surefire-plugin</artifactId>
<version>2.22.1</version>
</plugin>
<plugin>
<artifactId>maven-war-plugin</artifactId>
<version>3.2.2</version>
</plugin>
<plugin>
<artifactId>maven-install-plugin</artifactId>
<version>2.5.2</version>
</plugin>
<plugin>
<artifactId>maven-deploy-plugin</artifactId>
<version>2.8.2</version>
    plugin>
    plugins>
    pluginManagement>
    build>

>roject>
```

配置文件resource

创建deatabase-conf文件夹下面c3p0.properties

```
###### database configuration information ######
c3p0.jdbcUrl=jdbc:mysql://LOCALHOST:3306/ssm_sms?useSSL=false&serverTimezone=UTC&allowPublicKeyRetrieval=true
c3p0.driverClass=com.mysql.cj.jdbc.Driver
c3p0.user=root
c3p0.password=root
###### C3P0 configuration information ######
c3p0.maxPoolSize=100
c3p0.minPoolSize=1
c3p0.initialPoolSize=5
 c3p0.maxIdleTime=1800
```

创建mybatis-conf文件夹下面创建mybatis-conf.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
<!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN" "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
 <!-- 为sql映射文件中的输入/输出参数设置类型别名 -->
 <typeAliases>
 <package name="com.cc.sms.bean"/>
 </typeAliases>
</configuration>
```

创建spring-conf文件夹下面创建applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:context="http://www.springframework.org/schema/context"
xmlns:tx="http://www.springframework.org/schema/tx"
 xsi:schemaLocation="http://www.springframework.org/schema/beans
 http://www.springframework.org/schema/beans/spring-beans.xsd
 http://www.springframework.org/schema/context
 https://www.springframework.org/schema/context/spring-context.xsd
 http://www.springframework.org/schema/tx
  http://www.springframework.org/schema/tx/spring-tx.xsd">

 <description>Spring configuration file</description>

 <!-- 读取c3p0.properties中的数据库配置信息 -->
 <context:property-placeholder location="classpath:database-conf/c3p0.properties"/>

 <!-- 配置数据源 -->
 <bean id="dataSource"
class="com.mchange.v2.c3p0.ComboPooledDataSource">
 <!-- 数据库驱动 -->
 <property name="driverClass" value="${c3p0.driverClass}"/>
 <!-- 连接数据库的url -->
 <property name="jdbcUrl" value="${c3p0.jdbcUrl}"/>
 <!-- 连接数据库的用户名 -->
 <property name="user" value="${c3p0.user}"/>
 <!-- 连接数据库的密码 -->
 <property name="password" value="${c3p0.password}"/>
 <!-- 初始化连接数 -->
 <property name="initialPoolSize" value="${c3p0.initialPoolSize}"/>
 <!-- 最大连接数 -->
```

```
30  <property name="maxPoolSize" value="${c3p0.maxPoolSize}"/>
31  <!-- 最小连接数 -->
32  <property name="minPoolSize" value="${c3p0.minPoolSize}"/>
33  <!-- 连接的生存时间 -->
34  <property name="maxIdleTime" value="${c3p0.maxIdleTime}"/>
35  </bean>
36
37  <!-- 配置Spring事务管理器 -->
38  <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
39  <!-- 原理:控制数据源 -->
40  <property name="dataSource" ref="dataSource"/>
41  </bean>
42
43  <!-- 开启事务注解扫描 -->
44  <tx:annotation-driven transaction-manager="transactionManager"/>
45
46  <!-- MyBatis与Spring整合 -->
47  <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
48  <!-- 注入数据源 -->
49  <property name="dataSource" ref="dataSource"/>
50  <!-- 指定Mapper映射文件位置 -->
51  <property name="mapperLocations" value="classpath:mapper/*.xml"/>
52  <!-- 指定MyBatis核心配置文件位置 -->
53  <property name="configLocation" value="classpath:/mybatis-conf/mybatis-config.xml"/>
54  <!-- 引入插件 -->
55  <property name="plugins">
56  <array>
57  <!-- 引入MyBaits分页插件 -->
58  <bean class="com.github.pagehelper.PageInterceptor">
59  <property name="properties">
60  <!-- 指定数据库类型 -->
61  <value>helperDialct=mysql</value>
62    property>
63    bean>
64    array>
65    property>
66    bean>
67
```

```
<!-- 开启Mapper接口扫描器: 扫描Dao层 -->
<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
<property name="basePackage" value="com.cc.sms.dao"/>
</bean>

<!-- 开启Spring IOC注解扫描器: 扫描Servie层-->
<context:component-scan base-package="com.cc.sms.service"/>



</beans>
```

springmvc-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:context="http://www.springframework.org/schema/context"
 xmlns:mvc="http://www.springframework.org/schema/mvc"
 xsi:schemaLocation="http://www.springframework.org/schema/beans
 http://www.springframework.org/schema/beans/spring-beans.xsd
 http://www.springframework.org/schema/context
 https://www.springframework.org/schema/context/spring-context.xsd
 http://www.springframework.org/schema/mvc
 https://www.springframework.org/schema/mvc/spring-mvc.xsd">

 <description>Spring-MVC configuration file</description>

 <!-- 启用注解扫描器: 扫描被@Controller注解的类 -->
 <context:component-scan base-package="com.cc.sms.controller"/>

 <!-- 加载注解驱动 -->
 <mvc:annotation-driven/>

 <!-- 处理静态资源 -->
 <mvc:default-servlet-handler/>

 <!-- 配置拦截器 -->
 <mvc:interceptors>
 <mvc:interceptor>
 <mvc:mapping path="/**"/>
 <mvc:exclude-mapping path="/static/h-ui/**"/>
```

```
<mvc:exclude-mapping path="/static/easyui/**"/>
<mvc:exclude-mapping path="/system/login"/>
<mvc:exclude-mapping path="/system/getVerifyCodeImage"/>
<bean class="com.cc.sms.interceptor.LoginInterceptor"/>
</mvc:interceptor>
</mvc:interceptors>

<!-- 配置文件上传解析器 -->
<bean id="multipartResolver" class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
<!-- 指定请求编码格式 -->
<property name="defaultEncoding" value="UTF-8"/>
<!-- 指定指定允许上传文件的最大值(20MB) -->
<property name="maxUploadSize" value="20971520"/>
</bean>

<!-- 配置视图解析器 -->
<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
<property name="prefix" value="/WEB-INF/view/"/>
<property name="suffix" value=".jsp"/>
</bean>

</beans>
```

导入webapp下面静态资源

导入WEB-I下面view项目前端页面

配置web.xml

```
<!DOCTYPE web-app PUBLIC
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
 "http://java.sun.com/dtd/web-app_2_3.dtd" >

<web-app>
 <display-name>Archetype Created Web Application</display-name>
 <description>a simple student management system,created by SSM framework~</description>

 <!-- 启动Spring -->
 <context-param>
 <!-- 启动Spring: 加载Spring核心配置 -->
```

```
12  <param-name>contextConfigLocation</param-name>
13  <!-- 注意: 使用classpath:path(防止异常:FileNotFoundException) -->
14  <param-value>classpath:spring-conf/applicationContext.xml</param-value>
15  </context-param>
16
17  <!-- 配置Spring MVC编码过滤器 -->
18  <filter>
19  <filter-name>encoding</filter-name>
20  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
21  <init-param>
22  <param-name>encoding</param-name>
23  <param-value>UTF-8</param-value>
24  </init-param>
25  <init-param>
26  <param-name>forceRequestEncoding</param-name>
27  <param-value>true</param-value>
28  </init-param>
29  <init-param>
30  <param-name>forceResponseEncoding</param-name>
31  <param-value>true</param-value>
32  </init-param>
33  </filter>
34  <filter-mapping>
35  <filter-name>encoding</filter-name>
36  <url-pattern>/*</url-pattern>
37  </filter-mapping>
38
39  <!-- 启动Spring: 配置加载Spring文件的监听器 -->
40  <listener>
41  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
42  </listener>
43
44  <!-- 启动Spring MVC -->
45  <servlet>
46  <!-- 配置Spring MVC的前端核心控制器 -->
47  <servlet-name>spring_mvc</servlet-name>
48  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
```

```
<!-- 加载Spring MVC配置文件 -->
<init-param>
<param-name>contextConfigLocation</param-name>
<param-value>classpath:spring-conf/springmvc-config.xml</param-value>
</init-param>
<!-- 服务器启动后立即加载Spring MVC配置文件 -->
<load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
<servlet-name>spring_mvc</servlet-name>
<url-pattern>/</url-pattern>
</servlet-mapping>

<welcome-file-list>
<welcome-file>index.jsp</welcome-file>
</welcome-file-list>

</web-app>
```

web.xml

```
<!DOCTYPE web-app PUBLIC
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
 "http://java.sun.com/dtd/web-app_2_3.dtd" >

<web-app>
 <display-name>Archetype Created Web Application</display-name>
 <description>a simple student management system,created by SSM framework
~</description>

 <!-- 启动Spring -->
 <context-param>
 <!-- 启动Spring: 加载Spring核心配置 -->
 <param-name>contextConfigLocation</param-name>
 <!-- 注意: 使用classpath:path(防止异常:FileNotFoundException) -->
 <param-value>classpath:spring-conf/applicationContext.xml</param-value>
 </context-param>

 <!-- 配置Spring MVC编码过滤器 -->
 <filter>
 <filter-name>encoding</filter-name>
```

```
<filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
<init-param>
<param-name>encoding</param-name>
<param-value>UTF-8</param-value>
</init-param>
<init-param>
<param-name>forceRequestEncoding</param-name>
<param-value>true</param-value>
</init-param>
<init-param>
<param-name>forceResponseEncoding</param-name>
<param-value>true</param-value>
</init-param>
</filter>
<filter-mapping>
<filter-name>encoding</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>

<!-- 启动Spring: 配置加载Spring文件的监听器 -->
<listener>
<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>

<!-- 启动Spring MVC -->
<servlet>
<!-- 配置Spring MVC的前端核心控制器 -->
<servlet-name>spring_mvc</servlet-name>
<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
<!-- 加载Spring MVC配置文件 -->
<init-param>
<param-name>contextConfigLocation</param-name>
<param-value>classpath:spring-conf/springmvc-config.xml</param-value>
</init-param>
<!-- 服务器启动后立即加载Spring MVC配置文件 -->
<load-on-startup>1</load-on-startup>
</servlet>
```

```
  <servlet-mapping>
  <servlet-name>spring_mvc</servlet-name>
  <url-pattern>/</url-pattern>
  </servlet-mapping>

  <welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>

</web-app>
```

搭建后端架构

```
src/main/java/com/cc/sms/bean 实体类
src/main/java/com/cc/sms/controller 控制器
src/main/java/com/cc/sms/dao 数据库操作
src/main/java/com/cc/sms/interceptor 拦截器
src/main/java/com/cc/sms/service 业务层
src/main/java/com/cc/sms/service/inpl 业务实现
src/main/java/com/cc/sms/util 工具类

```

导入拦截器

LoginInterceptor

导入工具类验证码

验证码 CreateVerifiCodeImage

文件上传 UploadFile

resource下面创建mapper映射文件夹

用户登录模块

查看index.jsp

```
<%-- 重定向到用户登录页面 --%>
<% response.sendRedirect("system/goLogin"); %>
```

创建数据库对应bean实体类

Admin

```
public class Admin {

 private Integer id;
 private String name;
 private char gender;
```

```
 private String password;
 private String email;
 private String telephone;
 private String address;
  private String portrait_path;//存储头像的项目路径
```

Clazz

```
public class Clazz {
 //班级信息
 private Integer id;
 private String name;
 private String number;
 private String introducation;
 //班主任信息
 private String coordinator;
 private String telephone;
  private String email;
  //所属年级
  private String grade_name;

  public Clazz(String name, String grade_name) {
  this.name = name;
  this.grade_name = grade_name;
  }
```

Grade

```
public class Grade {

 //年级信息
 private Integer id;
 private String name;
 private String introducation;
 //年级主任信息
 private String manager;
 private String email;
  private String telephone;
```

LoginForm

```
public class LoginForm {

 private String username;
```

```
4   private String password;
5   private String verifiCode;
6   private Integer userType;
```

Student

```
1  public class Student {
2
3   private Integer id;
4   private String sno;
5   private String name;
6   private char gender = '男';//default
7   private String password;
8   private String email;
9   private String telephone;
10   private String address;
11   private String introducation;
12   private String portrait_path;//存储头像的项目路径
13   private String clazz_name;//班级名称
14
15   public Student(String name, String clazz_name) {
16   this.name = name;
17   this.clazz_name = clazz_name;
18   }
```

Teacher

```
1  public class Teacher {
2
3   private Integer id;
4   private String tno;
5   private String name;
6   private char gender;
7   private String password;
8   private String email;
9   private String telephone;
10   private String address;
11   private String clazz_name;
12   private String portrait_path;//存储头像的项目路径
13
14   public Teacher(String name, String clazz_name) {
```

```
15  this.name = name;
16  this.clazz_name = clazz_name;
17  }
```

查看index.jsp

```
1  <%-- 重定向到用户登录页面 --%>
2  <% response.sendRedirect("system/goLogin"); %>
```

创建用户登录转发到登录页面Controller

```
1  @Controller
2  @RequestMapping("/system")
3  public class SystemController {
4   //注入业务对象
5  @Autowired
6  private AdminService adminService;
7  @Autowired
8  private TeacherService teacherService;
9  @Autowired
10  private StudentService studentService;
11   //存储返回给页面的对象数据
12  private Map<String, Object> result = new HashMap<>();
13   /**
14   * @description: 跳转到用户登录页面
15   *
16   *
17   * @return: java.lang.String
18   */
19  @GetMapping("/goLogin")
20  public String goLogin() {
21   return "system/login";
22  }
23  /**
24   * @description: 获取并显示验证码图片
25   * @param: request
26   * @param: response
27   *
28   * @return: void
29   */
30  @GetMapping("/getVerifyCodeImage")
```

```
31  public void getVerifyCodeImage(HttpServletRequest request, HttpServletRe
sponse response) {
32   // 验证码图片
33   BufferedImage verifyCodeImage = CreateVerifyCodeImage.getVerifyCodeImag
e();
34   // 验证码
35   String verifyCode =
String.valueOf(CreateVerifyCodeImage.getVerifyCode());
36   // 将验证码图片输出到登录页面
37   try {
38   ImageIO.write(verifyCodeImage, "JPEG", response.getOutputStream());
39   } catch (IOException e) {
40   e.printStackTrace();
41   }
42   // 存储验证码Session
43   request.getSession().setAttribute("verifyCode", verifyCode);
44  }
45
```

点击验证码实现

```
1  <!-- 页面事件 -->
2  <script type="text/javascript">
3   $(function () {
4   //点击图片切换验证码
5   $("#vcodeImg").click(function () {
6   this.src = "getVerifyCodeImage?t=" + new Date().getTime();
7   });
8
9   })
10 </script>
```

管理员登录

login.jsp function里面书写

```
1  //登录按钮事件
2  $("#submitBtn").click(function () {
3   //检查登录信息
```

```
 if ($('#username').val() === '') {
 $.messager.alert("提示", "请输入用户名 !", "warning");
 } else if ($('#password').val() === '') {
 $.messager.alert("提示", "请输入密码 !", "warning");
 } else if ($('#verifyCode').val() === '') {
 $.messager.alert("提示", "请输入验证码 !", "warning");
  } else {
  //提交用户的登录表单信息
  var data = $("#form").serialize();
  $.ajax({
  type: "post",
  url: "login",
  data: data,
  dataType: "json",
  success: function (data) {
  if (data.success) {
  window.location.href = "goSystemMainView";//进入系统主页面
  } else {
  $.messager.alert("提示", data.msg, "warning");
  $("#vcodeImg").click();//切换验证码
  $("input[name='vcode']").val("");//清空验证码输入框
  }
  }
  });
  }
});
```

SystemController

```
@Controller
@RequestMapping("/system")
public class SystemController {

 //注入业务对象
 @Autowired
 private AdminService adminService;
 @Autowired
 private TeacherService teacherService;
  @Autowired
  private StudentService studentService;
```

```
12
13  //存储返回给页面的对象数据
14  private Map<String, Object> result = new HashMap<>();
15
16  @PostMapping("login")
17  @ResponseBody
18  public Map<String, Object> login(LoginForm loginForm, HttpServletReques
t request) {
19
20  // 校验验证码信息
21  String vcode = (String)
request.getSession().getAttribute("verifiCode");
22  if ("".equals(vcode)) {
23  result.put("success", false);
24  result.put("msg", "长时间为操作,会话已失效,请刷新页面后重试!");
25  return result;
26  } else if (!loginForm.getVerifiCode().equalsIgnoreCase(vcode)) {
27  result.put("success", false);
28  result.put("msg", "验证码错误!");
29  return result;
30  }
31  request.getSession().removeAttribute("verifyCode");
32
33  //效验用户登录信息
34  switch (loginForm.getUserType()) {
35  //管理员身份
36  case 1:
37  try {
38  Admin admin = adminService.login(loginForm);//验证账户和密码是否存在
39  if (admin != null) {
40  HttpSession session = request.getSession(); //将用户信息存储到Session
41  session.setAttribute("userInfo", admin);
42  session.setAttribute("userType", loginForm.getUserType());
43  result.put("success", true);
44  return result;
45  }
46  } catch (Exception e) {
47  e.printStackTrace();
48  result.put("success", false);
49  result.put("msg", "登录失败! 服务器端发生异常!");
50  return result;
```

```
51  }
52  break;
53  //学生身份
54  case 2:
55  try {
56  Student student = studentService.login(loginForm);
57  if (student != null) {
58  HttpSession session = request.getSession();
59  session.setAttribute("userInfo", student);
60  session.setAttribute("userType", loginForm.getUserType());
61  result.put("success", true);
62  return result;
63  }
64  } catch (Exception e) {
65  e.printStackTrace();
66  result.put("success", false);
67  result.put("msg", "登录失败! 服务器端发生异常!");
68  return result;
69  }
70  break;
71  //教师身份
72  case 3:
73  try {
74  Teacher teacher = teacherService.login(loginForm);
75  if (teacher != null) {
76  HttpSession session = request.getSession();
77  session.setAttribute("userInfo", teacher);
78  session.setAttribute("userType", loginForm.getUserType());
79  result.put("success", true);
80  return result;
81  }
82  } catch (Exception e) {
83  e.printStackTrace();
84  result.put("success", false);
85  result.put("msg", "登录失败! 服务器端发生异常!");
86  return result;
87  }
88  break;
89  }
90  //登录失败
```

```
91   result.put("success", false);
92   result.put("msg", "用户名或密码错误!");
93   return result;
94   }
95
96   /**
97   * @description: 跳转到系统主页面
98   * @param: no
100   * @return: java.lang.String
101   */
102   @GetMapping("/goSystemMainView")
103   public String goSystemMainView() {
104   return "system/main";
105   }
```

AdminService

```
1  public interface AdminService {
2
3   Admin login(LoginForm loginForm);
```

AdminServiceImpl

```
1  @Service
2  @Transactional
3  public class AdminServiceImpl implements AdminService {
4
5   //注入Mapper接口对象
6   @Autowired
7   private AdminMapper adminMapper;
8
9   @Override
10   public Admin login(LoginForm loginForm) { return adminMapper.login(logi
nForm); }
```

AdminMapper

```
1  public interface AdminMapper {
2
3   Admin login(LoginForm loginForm);
```

AdminMapper.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" >
<mapper namespace="com.cc.sms.dao.AdminMapper">

 <!-- 验证登录信息是否正确 -->
 <select id="login" parameterType="loginForm" resultType="com.cc.sms.bean.Admin">
  SELECT id,
  name,
  gender,
   password,
   email,
   telephone,
   address,
   portrait_path
   FROM tb_admin
   WHERE name = #{username}
   AND password = #{password}
   </select>
```

学生

studentService

```
public interface StudentService {
Student login(LoginForm loginForm);
```

studentServiceImpl

```
@Service
@Transactional
public class StudentServiceImpl implements StudentService {

 //注入Mapper接口对象
 @Autowired
 private StudentMapper studentMapper;

 @Override
  public Student login(LoginForm loginForm) {
```

```
11   return studentMapper.login(loginForm);
12   }
```

StudentMapper

```
1  public interface StudentMapper {
2
3   Student login(LoginForm loginForm);
```

StudentMapper.xml

```
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://myba
tis.org/dtd/mybatis-3-mapper.dtd" >
3  <mapper namespace="com.cc.sms.dao.StudentMapper">
4
5   <!-- 验证登录信息是否正确 -->
6   <select id="login" parameterType="loginform" resultType="com.cc.sms.b
ean.Student">
7   SELECT id,
8   sno,
9   name,
10   gender,
11   password,
12   portrait_path,
13   email,
14   telephone,
15   address,
16   introducation,
17   clazz_name
18   FROM tb_student
19   WHERE name = #{username}
20   AND password = #{password}
21   </select>
```

用户注销

查看main.jsp

```
1  <a href="loginOut" id="loginOut" style="color: darkgrey;" class="easyui-l
inkbutton"
2   data-options="iconCls:'icon-exit',plain:true">
```

```
[安全退出]
</a>
```

SystemController

```
@GetMapping("/loginOut")
public void loginOut(HttpServletRequest request, HttpServletResponse response) {
request.getSession().removeAttribute("userInfo");
request.getSession().removeAttribute("userType");

//注销后重定向到登录页面
try {
response.sendRedirect("../index.jsp");
} catch (IOException e) {
 e.printStackTrace();
 }
 }
```

系统用户管理列表查询

查看main.jsp 点击管理员列表请求 "url": "../admin/goAdminListView"

```
{
"menuid": "5", "icon": "", "menuname": "系统用户管理",
"menus": [
{
"menuid": "25",
"menuname": "管理员列表",
"icon": "icon-admin",
"url": "../admin/goAdminListView"
}
 ]
```

AdminController

```
@Controller
@RequestMapping("/admin")
public class AdminController {

//注入业务对象
@Autowired
private AdminService adminService;

```

```
//存储预返回页面的结果对象
 private Map<String, Object> result = new HashMap<>();

 /**
 * @description: 跳转到管理员信息管理页面
 * @param: no
 * @return: java.lang.String
 */
 @GetMapping("/goAdminListView")
 public String getAdminList() {
 return "admin/adminList";
 }
```

转发到adminList,自动异步加载查询列表 查看adminList.jsp 返回列表数据显示在表单

```
$('#dataList').datagrid({
 iconCls: 'icon-more',//图标
 border: true,
 collapsible: false,//是否可折叠
 fit: true,//自动大小
 method: "post",
 url: "getAdminList?t" + new Date().getTime(),
```

AdminController

```
/**
 * @description: 分页查询:根据管理员姓名获取指定/所有管理员信息列表
 * @param: page 当前页码
 * @param: rows 列表行数
 * @param: username 管理员姓名
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/getAdminList")
@ResponseBody
public Map<String, Object> getAdminList(Integer page, Integer rows, Stri
ng username) {

 //获取查询的用户名
 Admin admin = new Admin();
 admin.setName(username);
 //设置每页的记录数
 PageHelper.startPage(page, rows);
```

```
17   //根据姓名获取指定或所有管理员列表信息
18   List<Admin> list = adminService.selectList(admin);
19   //封装查询结果
20   PageInfo<Admin> pageInfo = new PageInfo<>(list);
21   //获取总记录数
22   long total = pageInfo.getTotal();
23   //获取当前页数据列表
24   List<Admin> adminList = pageInfo.getList();
25   //存储对象数据
26   result.put("total", total);
27   result.put("rows", adminList);
28
29   return result;
30  }
```

AdminService

```
1  // TODO: 根据姓名查询指定/所有管理员信息列表
2  List<Admin> selectList(Admin admin);
```

AdminServiceImpl

```
1  @Override
2  public List<Admin> selectList(Admin admin) {
3   return adminMapper.selectList(admin);
4  }
```

AdminMapper

```
1  根据姓名查询指定/所有管理员信息列表
2  List<Admin> selectList(Admin admin);
```

AdminMapper.xml

```
1  <!-- 根据姓名模糊查询指定/所有管理员信息 列表 -->
2  <select id="selectList" parameterType="admin" resultType="com.cc.sms.b
ean.Admin">
3    SELECT id, name, gender, password, email, telephone, address, portrait_p
ath
4    FROM tb_admin
5    <where>
6    <!-- name为Admin中的属性名(在Controller层中已将查询的姓名封装到了Admin中的na
me属性中) -->
7    <if test="name!=null and name!=''">
8    AND name LIKE concat(concat('%',#{name}),'%')
```

```
 </if>
  </where>
</select>
```

系统用户管理添加管理员

查看AdminList

```
text: '添加',
plain: true,
iconCls: 'icon-user_add',
handler: function () {
 var validate = $("#addForm").form("validate");
 if (!validate) {
 $.messager.alert("消息提醒", "请检查你输入的数据哟 !", "warning");
 } else {
 var data = $("#addForm").serialize();//序列化表单信息
  $.ajax({
  type: "post",
  url: "addAdmin",
  data: data,
  dataType: 'json',
```

AdminController

```
/**
 * @description: 添加管理员信息
 * @param: admin
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/addAdmin")
@ResponseBody
public Map<String, Object> addAdmin(Admin admin) {
 //判断用户名是否已存在
  Admin user = adminService.findByName(admin.getName());
  if (user == null) {
  if (adminService.insert(admin) > 0) {
  result.put("success", true);
  } else {
  result.put("success", false);
  result.put("msg", "添加失败! (ಥ_ಥ)服务器端发生异常!");
  }
  } else {
  result.put("success", false);
```

```
 result.put("msg", "该用户名已存在! 请修改后重试!");
 }
 return result;
}
```

AdminService

```
// TODO: 根据用户名查询指定管理员信息
Admin findByName(String name);

// TODO: 添加管理员信息
int insert(Admin admin);
```

AdminServiceImpl

```
@Override
public Admin findByName(String name) {
 return adminMapper.findByName(name);
}

@Override
public int insert(Admin admin) {
 return adminMapper.insert(admin);
}
```

AdminMapper

```
// TODO: 通过姓名查询指定管理员信息
Admin findByName(String name);

// TODO: 添加管理员信息
int insert(Admin admin);
```

AdminMapper.xml

```
<!-- 根据id查询指定管理员信息 -->
<select id="findByName" parameterType="String" resultType="com.cc.sms.bean.Admin">
 SELECT id,
 name,
 gender,
 password,
 email,
```

```
 telephone,
 address,
  portrait_path
  FROM tb_admin
  WHERE name = #{name}
</select>

<!-- 添加管理员信息 -->
<insert id="insert" parameterType="admin">
  INSERT INTO tb_admin(name, gender, password, email, telephone, address, portrait_path)
  VALUES (#{name}, #{gender}, #{password}, #{email}, #{telephone}, #{address}, #{portrait_path})
</insert>
```

根据id修改指定管理员信息

查看AdminLsit.jsp

```
var data = $("#editForm").serialize();//序列化表单信息
$.ajax({
  type: "post",
  url: "editAdmin?t=" + new Date().getTime(),
  data: data,
  dataType: 'json',
```

AdminController

```
/**
  * @description: 根据id修改指定管理员信息
  * @param: admin
  * @return: java.util.Map<java.lang.String, java.lang.Object>
  */
@PostMapping("/editAdmin")
@ResponseBody
public Map<String, Object> edityAdmin(Admin admin) {
  //需排除用户只修改账户名以外的信息
   Admin user = adminService.findByName(admin.getName());
   if (user != null) {
   if (!(admin.getId().equals(user.getId()))) {
```

```
 result.put("success", false);
 result.put("msg", "该用户名已存在! 请修改后重试!");
 return result;
 }
 }
 //修改操作
 if (adminService.update(admin) > 0) {
 result.put("success", true);
 } else {
 result.put("success", false);
 result.put("msg", "修改失败! (ಥ_ಥ)服务器端发生异常!");
 }
 return result;
}
```

AdminService

```
// TODO: 根据id更新指定管理员信息
int update(Admin admin);
```

AdminServiceImpl

```
@Override
public int update(Admin admin) { return adminMapper.update(admin); }
```

AdminMapper

```
// TODO: 根据姓名查询指定/所有管理员信息列表
List<Admin> selectList(Admin admin);
```

AdminMapper.xml

```
<!-- 根据id更新指定管理员信息 -->
<update id="update" parameterType="admin">
 UPDATE tb_admin
 <set>
 <if test="name!=null and name!=''">name=#{name},</if>
 <if test="gender!=null and gender!=''">gender=#{gender},</if>
 <if test="email!=null and email!=''">email=#{email},</if>
 <if test="telephone!=null and telephone!=''">telephone=#{telephone},
</if>
 <if test="address!=null and address!=''">address=#{address},</if>
```

```
  <if test="portrait_path!=null and portrait_path!=''">portrait_path=#{portrait_path},</if>
  </set>
  WHERE id = #{id}
</update>
```

删除指定id的管理员信息

查看AdminList.jsp

```
$.ajax({
 type: "post",
 url: "deleteAdmin?t" + new Date().getTime(),
 data: {ids: ids},
 dataType: 'json',
```

AdminController

```
/**
 * @description: 删除指定id的管理员信息
 * @param: ids 拼接后的id
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/deleteAdmin")
@ResponseBody
public Map<String, Object> deleteAdmin(@RequestParam(value = "ids[]", required = true) Integer[] ids) {

 if (adminService.deleteById(ids) > 0) {
 result.put("success", true);
 } else {
 result.put("success", false);
 }
 return result;
}
```

AdminService

```
// TODO: 根据id删除管理员信息
int deleteById(Integer[] ids);
```

AdminServiceImpl

```
@Override
public int deleteById(Integer[] ids) {
 return adminMapper.deleteById(ids);
}
```

AdminMapper

```
// TODO: 根据id删除指定管理员信息
int deleteById(Integer[] ids);
```

AdminMapper.xml

```
<!-- 根据id批量删除管理员信息 -->
<delete id="deleteById">
 DELETE FROM ssm_sms.tb_admin WHERE id IN
 <foreach collection="array" item="ids" open="(" separator="," close=")">
 #{ids}
 </foreach>
</delete>
```

上传头像-原理:将头像上传到项目发布目录中,通过读取数据库中的头像路径来获取头像

查看AdminList.jsp

```
<!-- 头像信息表单 -->
<form id="add-uploadForm" method="post" enctype="multipart/form-data" action="uploadPhoto"
 target="photo_target">
 <input id="choose-portrait" class="easyui-filebox" name="photo" data-options="prompt:'选择照片'"
 style="width:200px;">
 <input id="add-upload-btn" class="easyui-linkbutton" style="width: 50px; height: 24px;;float:right;"
 type="button" value="上传"/>
</form>
```

AdminController

```
/**
```

```
    @description: 上传头像 原理:将头像上传到项目发布目录中,通过读取数据库中的头像路径来获取头像
    * @param: photo
    * @param: request
    * @return: java.util.Map<java.lang.String, java.lang.Object>
    */
    @PostMapping("/uploadPhoto")
    @ResponseBody
    public Map<String, Object> uploadPhoto(MultipartFile photo, HttpServletRequest request) {
        //存储头像的本地目录
        final String dirPath = request.getServletContext().getRealPath("/upload/admin_portrait/");
        //存储头像的项目发布目录
        final String portraitPath = request.getServletContext().getContextPath() + "/upload/admin_portrait/";
        //返回头像的上传结果
        return UploadFile.getUploadResult(photo, dirPath, portraitPath);
    }
```

修改管理员密码

查看main.jsp

```
menus": [
    {
    "menuid": "26",
    "menuname": "修改密码",
    "icon": "icon-settings",
    "url": "../common/goSettingView"
    }
```

CommonController

```
@Controller
@RequestMapping("/common")
public class CommonController {

    //注入业务对象
    @Autowired
    private AdminService adminService;
    @Autowired
```

```
9   private TeacherService teacherService;
10   @Autowired
11   private StudentService studentService;
12
13   //存储预返回给页面的结果对象
14   private Map<String, Object> result = new HashMap<>();
15
16   /**
17   * @description: 跳转到个人信息管理页面
18   * @param: no
20   * @return: java.lang.String
21   */
22   @GetMapping("/goSettingView")
23   public String getAdminList() {
24   return "common/settings";
25   }
```

查看common下面settings.jsp

```
1  $.ajax({
2   type: "post",
3   url: "editPassword?t=" + new Date().getTime(),
4   data: data,
5   success: function (data) {
```

CommonController

```
1  /**
2   * @description: 修改密码
3   * @param: oldPassword
4   * @param: newPassword
5   * @param: request
6   * @return: java.util.Map<java.lang.String, java.lang.Object>
7   */
8  @PostMapping("/editPassword")
9  @ResponseBody
10  public Map<String, Object> editPassword(String oldPassword, String newPa
ssword, HttpServletRequest request) {
11   //判断当前登录用户的用户类型
12   int userType = Integer.parseInt(request.getSession().getAttribute("user
Type").toString());
```

```
//管理员身份
if (userType == 1) {
Admin admin = (Admin) request.getSession().getAttribute("userInfo");
if (!admin.getPassword().equals(oldPassword)) {
result.put("success", false);
result.put("msg", "原密码错误!");
return result;
}
try {
//修改密码
admin.setPassword(newPassword);//覆盖旧密码值,存储待更新的密码
if (adminService.updatePassowrd(admin) > 0) {
result.put("success", true);
}
} catch (Exception e) {
e.printStackTrace();
result.put("success", false);
result.put("msg", "修改失败! 服务器端发生异常!");
}
}

//学生身份
if (userType == 2) {
Student student = (Student) request.getSession().getAttribute("userInf
o");
if (!student.getPassword().equals(oldPassword)) {
result.put("success", false);
result.put("msg", "原密码错误!");
return result;
}
try {
student.setPassword(newPassword);
if (studentService.updatePassowrd(student) > 0) {
result.put("success", true);
}
} catch (Exception e) {
e.printStackTrace();
result.put("success", false);
result.put("msg", "修改失败! 服务器端发生异常!");
}
```

```
52   }
53
54   //教师身份
55   if (userType == 3) {
56   Teacher teacher = (Teacher) request.getSession().getAttribute("userInf
o");
57   if (!teacher.getPassword().equals(oldPassword)) {
58   result.put("success", false);
59   result.put("msg", "原密码错误!");
60   return result;
61   }
62   try {
63   teacher.setPassword(newPassword);
64   if (teacherService.updatePassowrd(teacher) > 0) {
65   result.put("success", true);
66   }
67   } catch (Exception e) {
68   e.printStackTrace();
69   result.put("success", false);
70   result.put("msg", "修改失败! 服务器端发生异常!");
71   }
72   }
73
74   return result;
75  }
```

AdminService

```
1  // TODO: 根据id修改指定用户密码
2  int updatePassowrd(Admin admin);
```

AdminServiceImpl

```
1  @Override
2  public int updatePassowrd(Admin admin) {
3   return adminMapper.updatePassword(admin);
4  }
```

AdminMapper

```
1  // TODO: 根据id修改指定管理员密码
2  int updatePassword(Admin admin);
```

AdminMapper.xml

```
<!-- 根据id修改指定用户密码 -->
<update id="updatePassword" parameterType="admin">
 UPDATE tb_admin
 SET password = #{password}
 WHERE id = #{id}
</update>
```

修改学生身份密码

StudentService

```
// TODO: 根据id修改指定学生密码
int updatePassowrd(Student student);
```

StudentServiceImpl

```
@Override
public int updatePassowrd(Student student) {
 return studentMapper.updatePassword(student);
}
```

StudentMapper

```
// TODO: 根据id修改指定学生密码
int updatePassword(Student student);
```

StudentMapper.xml

```
<!-- 根据id修改指定学生密码 -->
<update id="updatePassword" parameterType="student">
 UPDATE tb_student
 SET password=#{password}
 WHERE id = #{id}
</update>
```

修改教师密码

TeacherService

```
// TODO: 根据id修改指定教师密码
int updatePassowrd(Teacher teacher);
```

TeacherServiceImpl

```
@Override
public int updatePassowrd(Teacher teacher) {
 return teacherMapper.updatePassword(teacher);
}
```

TeacherMapper

```
// TODO: 根据id修改指定教师密码
int updatePassword(Teacher teacher);
```

TeacherMapper.xml

```
<!-- 根据id修改指定教师密码 -->
<update id="updatePassword" parameterType="teacher">
 UPDATE tb_teacher
 SET password=#{password}
 WHERE id = #{id}
</update>
```

年级信息管理

查看main.jsp

```
"menus": [
 {
 "menuid": "24",
 "menuname": "年级列表",
 "icon": "icon-grade",
 "url": "../grade/goGradeListView"
 }
```

GradeController

```
@Controller
@RequestMapping("/grade")
public class GradeController {

 // 注入业务对象
 @Autowired
 private GradeService gradeService;
```

```
8
9   // 存储预返回页面的结果对象
10   private Map<String, Object> result = new HashMap<>();
11
12   /**
13   * @description: 跳转到年级信息管理页面
14   * @param: no
15   * @return: java.lang.String
16   */
17   @GetMapping("/goGradeListView")
18   public String goGradeListPage() {
19   return "grade/gradeList";
20   }
```

查看gradeList.jsp年级列表查询

```
1  $('#dataList').datagrid({
2   iconCls: 'icon-more',//图标
3   border: true,
4   collapsible: false,//是否可折叠
5   fit: true,//自动大小
6   method: "post",
7   url: "getGradeList?t" + new Date().getTime(),
8   idField: 'id',
```

GradeController

```
1  /**
2   * @description: 分页查询:根据年级名称获取指定/所有年级信息列表
3   * @param: page
4   * @param: rows
5   * @param: gradename
6   * @return: java.util.Map<java.lang.String, java.lang.Object>
7   */
8  @PostMapping("/getGradeList")
9  @ResponseBody
10  public Map<String, Object> getGradeList(Integer page, Integer rows, Stri
ng gradename) {
11
```

```
12   //注意:使用Java Bean传递gradename,防止以下异常 !
13   //org.springframework.web.util.NestedServletException: Request processi
ng failed;
14   // nested exception is org.mybatis.spring.MyBatisSystemException:
15   // nested exception is org.apache.ibatis.reflection.ReflectionExceptio
n:
16   // There is no getter for property named 'name' in 'class java.lang.Str
ing'
17   Grade grade = new Grade();
18   grade.setName(gradename);
19
20   //设置每页的记录数
21   PageHelper.startPage(page, rows);
22   //根据年级名称获取指定或全部年级信息列表
23   List<Grade> list = gradeService.selectList(grade);
24   //封装信息列表
25   PageInfo<Grade> pageInfo = new PageInfo<>(list);
26   //获取总记录数
27   long total = pageInfo.getTotal();
28   //获取当前页数据列表
29   List<Grade> gradeList = pageInfo.getList();
30   //存储数据对象
31   result.put("total", total);
32   result.put("rows", gradeList);
33
34   return result;
35 }
```

GradeService

```
1 // TODO: 根据年级名称查询指定/全部年级列表信息
2 List<Grade> selectList(Grade gradename);
```

GradeServiceImpl

```
1 @Service
2 @Transactional
3 public class GradeServiceImpl implements GradeService {
4
5  //注入Mapper接口对象
6  @Autowired
```

```
7   private GradeMapper gradeMapper;
8
9   @Override
10   public List<Grade> selectList(Grade gradename) {
11   return gradeMapper.selectList(gradename);
12   }
```

GradeMapper

```
1  public interface GradeMapper {
2
3   // TODO: 根据年级名称查询指定/全部年级信息列表
4   List<Grade> selectList(Grade gradename);
```

GradeMapper.xml

```
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://myba
tis.org/dtd/mybatis-3-mapper.dtd" >
3  <mapper namespace="com.cc.sms.dao.GradeMapper">
4
5   <!-- 根据年级名称查询指定/全部年级信息列表 -->
6   <select id="selectList" parameterType="grade" resultType="com.cc.sms.
bean.Grade">
7   SELECT id, name, manager, email, telephone, introducation
8   FROM ssm_sms.tb_grade
9   <where>
10   <if test="name!=null and name!=''">
11   AND name LIKE concat(concat('%',#{name}),'%')
12   </if>
13   </where>
14   </select>
```

年级列表添加查看gadeList.jsp

```
1  var data = $("#addForm").serialize();//序列化表单信息
2  $.ajax({
3   type: "post",
4   url: "addGrade?t" + new Date().getTime(),
5   data: data,
```

GradeController

```
@PostMapping("/addGrade")
@ResponseBody
public Map<String, Object> addGrade(Grade grade) {
 //判断年级名是否已存在
 Grade name = gradeService.findByName(grade.getName());
 if (name == null) {
 if (gradeService.insert(grade) > 0) {
 result.put("success", true);
 } else {
  result.put("success", false);
  result.put("msg", "添加失败! (ಥ_ಥ)服务器端发生异常!");
  }
  } else {
  result.put("success", false);
  result.put("msg", "该年级名称已存在! 请修改后重试!");
  }

  return result;
}
```

GradeService

```
// TODO: 根据年级名称查询指定年级信息
Grade findByName(String gradename);

// TODO: 添加年级信息
int insert(Grade grade);
```

GradeServiceImpl

```
@Override
public Grade findByName(String gradename) {
 return gradeMapper.findByName(gradename);
}

@Override
public int insert(Grade grade) {
 return gradeMapper.insert(grade);
}
```

GradeMapper

```
// TODO: 根据年级名称查询指定的年级信息
Grade findByName(String gradename);

// TODO: 添加年级信息
int insert(Grade grade);
```

GradeMapper.xml

```
<!-- 根据年级名称查询指定年级信息 -->
<select id="findByName" parameterType="String" resultType="com.cc.sms.bean.Grade">
 SELECT id, name, manager, email, telephone, introducation
 FROM ssm_sms.tb_grade
 WHERE name = #{name}
</select>

<!-- 添加年级信息 -->
<insert id="insert" parameterType="grade">
  INSERT INTO ssm_sms.tb_grade(name, manager, email, telephone, introducation)
  VALUES (#{name}, #{manager}, #{email}, #{telephone}, #{introducation})
 </insert>
```

编辑年级信息查看gradeList.jsp

```
$.ajax({
 type: "post",
 url: "editGrade?t=" + new Date().getTime(),
 data: data,
 dataType: 'json',
```

GradeController

```
/**
 * @description: 根据id修改指定的年级信息
 * @param: grade
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/editGrade")
@ResponseBody
```

```
public Map<String, Object> editGrade(Grade grade) {
 //需排除用户只修改年级名以外的信息
  Grade g = gradeService.findByName(grade.getName());
  if (g != null) {
  if (!(grade.getId().equals(g.getId()))) {
  result.put("success", false);
  result.put("msg", "该年级名称已存在! 请修改后重试!");
  return result;
  }
  }
  //添加操作
  if (gradeService.update(grade) > 0) {
  result.put("success", true);
  } else {
  result.put("success", false);
  result.put("msg", "添加失败! (ಥ_ಥ)服务器端发生异常!");
  }
  return result;
 }
```

GradeService

```
// TODO: 根据id修改指定年级信息
int update(Grade grade);
```

GradeServiceImpl

```
@Override
public int update(Grade grade) {
 return gradeMapper.update(grade);
}
```

GradeMapper

```

// TODO: 根据id修改指定年级信息
int update(Grade grade);
```

GradeMapper.xml

```
<!-- 根据id更新指定年级信息 -->
<update id="update" parameterType="grade">
 UPDATE ssm_sms.tb_grade
 SET name=#{name},
```

```
 manager=#{manager},
 email=#{email},
 telephone=#{telephone},
 introducation=#{introducation}
 WHERE id = #{id}
</update>
```

删除年级信息查看grade.jsp

```
$.ajax({
 type: "post",
 url: "deleteGrade?t" + new Date().getTime(),
 data: {ids: ids},
 dataType: 'json',
```

GradeController

```
/**
 * @description: 根据id删除指定的年级信息
 * @param: ids 拼接后的id
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/deleteGrade")
@ResponseBody
public Map<String, Object> deleteGrade(@RequestParam(value = "ids[]", req
uired = true) Integer[] ids) {

 if (gradeService.deleteById(ids) > 0) {
 result.put("success", true);
 } else {
 result.put("success", false);
 }
 return result;
}
```

GradeService

```
// TODO: 根据id删除指定年级信息
int deleteById(Integer[] ids);
```

GradeServiceImpl

```
@Override
public int deleteById(Integer[] ids) {
return gradeMapper.deleteById(ids);
}

```

GradeMapper

```
// TODO: 根据id删除指定年级信息
int deleteById(Integer[] ids);
```

GradeMapper.xml

```
<!-- 根据id批量删除指定的年级信息 -->
<delete id="deleteById">
 DELETE FROM ssm_sms.tb_grade WHERE id IN
 <foreach collection="array" item="ids" open="(" separator="," close=")">
 #{ids}
 </foreach>
</delete>
```

班级信息管理查看main.jsp

```
"menus": [
 {
 "menuid": "23",
 "menuname": "班级列表",
 "icon": "icon-class",
 "url": "../clazz/goClazzListView"
 }
```

ClazzController

```
@Controller
@RequestMapping("/clazz")
public class ClazzController {

 //注入业务对象
 @Autowired
```

```
7   private ClazzService clazzService;
8   @Autowired
9   private GradeService gradeService;
10
11   //存储预返回页面的数据对象
12   private Map<String, Object> result = new HashMap<>();
13
14
15   /**
16   * @description: 跳转到班级信息管理页面
17   * @param: modelAndView
18   * @return: org.springframework.web.servlet.ModelAndView
19   */
20   @GetMapping("/goClazzListView")
21   public ModelAndView goClazzListPage(ModelAndView modelAndView) {
22   //向页面发送一个存储着Grade的List对象
23   modelAndView.addObject("gradeList", gradeService.selectAll());
24   modelAndView.setViewName("clazz/clazzList");
25   return modelAndView;
26   }
```

GradeService

```
1  // TODO: 查询所有年级列表信息(用于在班级管理页面中获取年级信息)
2  List<Grade> selectAll();
```

GradeServiceImpl

```
1  @Override
2  public List<Grade> selectAll() {
3   return gradeMapper.selectAll();
4  }
```

GradeMapper

```
1  // TODO: 查询所有年级信息列表(用于在班级管理页面中获取年级信息)
2  List<Grade> selectAll();
```

gradeMapper.xml

```
1  <!-- 查询所有年级列表信息(用于在班级管理页面中获取年级信息) -->
```

```
<select id="selectAll" resultType="com.cc.sms.bean.Grade">
 SELECT id, name, manager, email, telephone, introducation
 FROM ssm_sms.tb_grade
</select>
```

查看clazzList.jsp

```
//初始化datagrid
$('#dataList').datagrid({
 iconCls: 'icon-more',//图标
 border: true,
 collapsible: false,//是否可折叠
 fit: true,//自动大小
 method: "post",
 url: "getClazzList?t" + new Date().getTime(),
```

ClazzController

```
/**
 * @description: 分页查询班级信息列表:根据班级与年级名查询指定/全部班级信息列表
 * @param: page 当前页码
 * @param: rows 列表行数
 * @param: gradename 年级名称
 * @param: clazzname 班级名称
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/getClazzList")
@ResponseBody
public Map<String, Object> getClazzList(Integer page, Integer rows, Stri
ng clazzname, String gradename) {

 //存储查询的clazzname,gradename信息
 Clazz clazz = new Clazz(clazzname, gradename);
 //设置每页的记录数
 PageHelper.startPage(page, rows);
 //根据班级与年级名获取指定或全部班级信息列表
 List<Clazz> list = clazzService.selectList(clazz);
 //封装列表信息
```

```
20   PageInfo<Clazz> pageInfo = new PageInfo<>(list);
21   //获取总记录数
22   long total = pageInfo.getTotal();
23   //获取当前页数据列表
24   List<Clazz> clazzList = pageInfo.getList();
25   //存储数据对象
26   result.put("total", total);
27   result.put("rows", clazzList);
28
29   return result;
30  }
31
```

ClazzService

```
1  public interface ClazzService {
2
3   // TODO: 根据年级与班级名查询指定/全部班级信息列表
4   List<Clazz> selectList(Clazz clazz);
```

ClazzServiceImpl

```
1  @Service
2  @Transactional
3  public class ClazzServiceImpl implements ClazzService {
4
5   //注入Mapper接口对象
6   @Autowired
7   private ClazzMapper clazzMapper;
8
9   @Override
10   public List<Clazz> selectList(Clazz clazz) { return clazzMapper.selectL
ist(clazz); }
```

ClazzMapper

```
1  public interface ClazzMapper {
2
3   根据班级名称获取指定/全部班级信息列表
4   List<Clazz> selectList(Clazz clazz);
```

ClazzMapper.xml

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" >
<mapper namespace="com.cc.sms.dao.ClazzMapper">

  <!-- 根据年级与班级名查询指定/全部班级信息列表 -->
  <select id="selectList" parameterType="clazz" resultType="com.cc.sms.bean.Clazz">
  SELECT id, name, number, introducation, coordinator, email, telephone, grade_name FROM tb_clazz
  <where>
  <if test="grade_name!=null and grade_name!=''">
   AND grade_name = #{grade_name}
   </if>
   <if test="name!=null and name!=''">
   AND name LIKE concat(concat('%',#{name}),'%')
   </if>
   </where>
   </select


```

班级信息添加 查看clazzLt.jsp

```
var data = $("#addForm").serialize();//序列化表单信息
$.ajax({
 type: "post",
 url: "addClazz?t" + new Date().getTime(),
 data: data,
```

ClazzController

```
/**
 * @description: 添加班级信息
 * @param: clazz
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/addClazz")
@ResponseBody
```

```java
public Map<String, Object> addClazz(Clazz clazz) {
 //判断班级名是否已存在
  Clazz name = clazzService.findByName(clazz.getName());
  if (name == null) {
  if (clazzService.insert(clazz) > 0) {
  result.put("success", true);
  } else {
  result.put("success", false);
  result.put("msg", "添加失败！(ಥ_ಥ)服务器端发生异常!");
  }
  } else {
  result.put("success", false);
  result.put("msg", "该班级名称已存在！请修改后重试!");
  }

  return result;
 }
```

ClazzService

```java
// TODO: 添加班级信息
int insert(Clazz clazz);
// TODO: 根据班级名获取指定班级信息
Clazz findByName(String name);
```

ClazzServiceImpl

```java
@Override
public Clazz findByName(String name) {
 return clazzMapper.findByName(name);
}

@Override
public int insert(Clazz clazz) {
 return clazzMapper.insert(clazz);
}
```

ClazzMapper

```java
// TODO: 获取指定名称的班级信息
Clazz findByName(String name);
```

```
3
4  // TODO: 添加班级信息
5  int insert(Clazz clazz);
```

ClazzMapper.xml

```
1  <!-- 查询指定名称的班级信息 -->
2  <select id="findByName" parameterType="String" resultType="com.cc.sms.
bean.Clazz">
3   SELECT id,
4   name,
5   number,
6   introducation,
7   coordinator,
8   email,
9   telephone,
10   grade_name
11   FROM tb_clazz
12   WHERE name = #{name}
13  </select>
14
15  <!-- 添加班级信息 -->
16  <insert id="insert" parameterType="clazz">
17   INSERT INTO tb_clazz(name, number, introducation, coordinator, email, t
elephone, grade_name)
18   VALUES (#{name}, #{number}, #{introducation}, #{coordinator}, #{email},
#{telephone}, #{grade_name})
19  </insert>
```

编辑班级信息查看clazzList.jsp

```
1  $.ajax({
2   type: "post",
3   url: "editClazz?t=" + new Date().getTime(),
4   data: data,
5   dataType: 'json',
```

ClazzController

```
1  /**
2   * @description: 根据id修改指定的班级信息
3   * @param: clazz
```

```
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/editClazz")
@ResponseBody
public Map<String, Object> editClazz(Clazz clazz) {
 //需排除用户只修改班级名以外的信息
  Clazz c = clazzService.findByName(clazz.getName());
  if (c != null) {
  if (!(clazz.getId().equals(c.getId()))) {
  result.put("success", false);
  result.put("msg", "该班级名称已存在! 请修改后重试!");
  return result;
  }
  }
  //添加操作
  if (clazzService.update(clazz) > 0) {
  result.put("success", true);
  } else {
  result.put("success", false);
  result.put("msg", "添加失败! (ಥ_ಥ)服务器端发生异常!");
  }
  return result;
 }
```

ClazzService

```
// TODO: 根据id修改指定班级信息
int update(Clazz clazz);
```

ClazzServiceImpl

```
@Override
public int update(Clazz clazz) {
 return clazzMapper.update(clazz);
}
```

ClazzMapper

```
// TODO: 根据id修改指定班级信息
int update(Clazz clazz);
```

ClazzMapper.xml

```
<!-- 指定id修改指定班级信息 -->
<update id="update" parameterType="clazz">
 UPDATE tb_clazz
 SET name=#{name},
 number=#{number},
 introducation=#{introducation},
 coordinator=#{coordinator},
 email=#{email},
 telephone=#{telephone},
  grade_name=#{grade_name}
  WHERE id = #{id}
</update>
```

班级信息删除查看clazzList.jsp

```
$.ajax({
 type: "post",
 url: "deleteClazz?t" + new Date().getTime(),
 data: {ids: ids},
 dataType: 'json',
```

ClazzController

```
/**
 * @description: 删除指定id的班级信息
 * @param: ids 拼接后的id
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/deleteClazz")
@ResponseBody
public Map<String, Object> deleteGrade(@RequestParam(value = "ids[]", req
uired = true) Integer[] ids) {

  if (clazzService.deleteById(ids) > 0) {
  result.put("success", true);
  } else {
  result.put("success", false);
  }
```

```
 return result;
}
```

ClazzService

```
// TODO: 根据id删除指定班级信息
int deleteById(Integer[] ids);
```

ClazzServiceImpl

```
@Override
 public int deleteById(Integer[] ids) {
 return clazzMapper.deleteById(ids);
 }

```

ClazzMapper

```
// TODO: 根据id删除指定班级信息
int deleteById(Integer[] ids);
```

ClazzMapper.xml

```
<!-- 根据id批量删除指定的班级信息 -->
<delete id="deleteById">
 DELETE FROM tb_clazz WHERE id IN
 <foreach collection="array" item="ids" open="(" separator="," close=")">
 #{ids}
 </foreach>
</delete>
```

教师信息管 理　查看main.jsp

```
"menus": [
 {
 "menuid": "22",
 "menuname": "教师列表",
 "icon": "icon-teacher",
 "url": "../teacher/goTeacherListView"
 }
```

TeacherController

```java
@Controller
@RequestMapping("/teacher")
public class TeacherController {

 //注入业务对象
 @Autowired
 private ClazzService clazzService;
 @Autowired
 private TeacherService teacherService;

 //存储预返回页面的数据对象
 private Map<String, Object> result = new HashMap<>();

 /**
 * @description: 跳转到教师信息管理页面
 * @param: modelAndView

 * @return: org.springframework.web.servlet.ModelAndView
 */
 @GetMapping("/goTeacherListView")
 public ModelAndView goTeacherListView(ModelAndView modelAndView) {
 //向页面发送一个存储着Clazz的List对象
 modelAndView.addObject("clazzList", clazzService.selectAll());
 modelAndView.setViewName("teacher/teacherList");
 return modelAndView;
 }
```

ClazzService

```java
// TODO: 查询所有班级信息列表(用于在学生管理页面中获取班级信息)
List<Clazz> selectAll();
```

ClazzServiceImpl

```java
@Override
public List<Clazz> selectAll() {
 return clazzMapper.selectAll();
}
```

ClazzMapper

```
// TODO: 查询所有班级列表信息(用于在学生管理页面中获取班级信息)
List<Clazz> selectAll();
```

ClazzMapper.xml

```
<!-- 查询所有班级列表信息(用于在学生管理页面中获取班级信息) -->
<select id="selectAll" resultType="com.cc.sms.bean.Clazz">
 SELECT id,
 name,
 number,
 introducation,
 coordinator,
 email,
 telephone,
  grade_name
 FROM ssm_sms.tb_clazz
</select>
```

查看教师信息列表,查看teacherList.jsp

```
$('#dataList').datagrid({
 iconCls: 'icon-more',//图标
 border: true,
 collapsible: false,//是否可折叠
 fit: true,//自动大小
 method: "post",
 url: "getTeacherList?t" + new Date().getTime(),
```

TeacherController

```
@Controller
@RequestMapping("/teacher")
public class TeacherController {

 //注入业务对象
 @Autowired
 private ClazzService clazzService;
```

```java
@Autowired
private TeacherService teacherService;

//存储预返回页面的数据对象
private Map<String, Object> result = new HashMap<>();

/**
* @description: 跳转到教师信息管理页面
* @param: modelAndView

* @return: org.springframework.web.servlet.ModelAndView
*/
@GetMapping("/goTeacherListView")
public ModelAndView goTeacherListView(ModelAndView modelAndView) {
//向页面发送一个存储着Clazz的List对象
modelAndView.addObject("clazzList", clazzService.selectAll());
modelAndView.setViewName("teacher/teacherList");
return modelAndView;
}
```

ClazzService

```java
// TODO: 查询所有班级信息列表(用于在学生管理页面中获取班级信息)
List<Clazz> selectAll();
```

ClazzServiceImpl

```java
@Override
public List<Clazz> selectAll() {
 return clazzMapper.selectAll();
}
```

ClazzMapper

```java
// TODO: 查询所有班级列表信息(用于在学生管理页面中获取班级信息)
List<Clazz> selectAll();
```

ClazzMapper.xml

```xml
<!-- 查询所有班级列表信息(用于在学生管理页面中获取班级信息) -->
<select id="selectAll" resultType="pers.lige.sms.bean.Clazz">
```

```
 SELECT id,
 name,
 number,
 introducation,
 coordinator,
 email,
 telephone,
  grade_name
  FROM ssm_sms.tb_clazz
</select>
```

查看teacheList.jsp

```
$('#dataList').datagrid({
 iconCls: 'icon-more',//图标
 border: true,
 collapsible: false,//是否可折叠
 fit: true,//自动大小
 method: "post",
 url: "getTeacherList?t" + new Date().getTime(),
```

TeacherController

```
/**
 * @description: 分页查询学生信息列表:根据教师与班级名查询指定/全部教师信息列表
 * @param: page 当前页码
 * @param: rows 列表行数
 * @param: teachername
 * @param: clazzname
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/getTeacherList")
 @ResponseBody
 public Map<String, Object> getTeacherList(Integer page, Integer rows, St
ring teachername, String clazzname) {

  //存储查询的teachername,clazzname信息
  Teacher teacher = new Teacher(teachername, clazzname);
  //设置每页的记录数
  PageHelper.startPage(page, rows);
```

```
17   //根据班级与教师名获取指定或全部教师信息列表
18   List<Teacher> list = teacherService.selectList(teacher);
19   //封装列表信息
20   PageInfo<Teacher> pageInfo = new PageInfo<>(list);
21   //获取总记录数
22   long total = pageInfo.getTotal();
23   //获取当前页数据列表
24   List<Teacher> teacherList = pageInfo.getList();
25   //存储数据对象
26   result.put("total", total);
27   result.put("rows", teacherList);
28
29   return result;
30  }
```

TeacherService

```
1  // TODO: 根据教师与班级名查询指定/全部教师列表信息
2  List<Teacher> selectList(Teacher teacher);
```

TeacherServiceImpl

```
1  @Override
2  public List<Teacher> selectList(Teacher teacher) {
3   return teacherMapper.selectList(teacher);
4  }
```

TeacherMapper

```
1  // TODO: 根据教师与班级名查询指定/全部教师信息列表
2  List<Teacher> selectList(Teacher teacher);
```

TeacherMapper.xml

```
1  <!-- 根据教师与班级名查询指定/全部教师信息列表 -->
2  <select id="selectList" parameterType="teacher" resultType="com.cc.sm
s.bean.Teacher">
3   SELECT id, tno, name, gender, password, email, telephone, address, portr
ait_path, clazz_name
4   FROM tb_teacher
5   <where>
```

```
 <if test="clazz_name!=null and clazz_name!=''">
 AND clazz_name = #{clazz_name}
 </if>
 <if test="name!=null and name!=''">
  AND name LIKE concat(concat('%',#{name}),'%')
  </if>
  </where>
</select>
```

教师信息添加查看teacherList.jsp

```
$.ajax({
 type: "post",
 url: "addTeacher?t" + new Date().getTime(),
 data: data,
 dataType: 'json',
 success: function (data) {
```

TeacherController

```
/**
 * @description: 添加教师信息
 * @param: teacher

 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/addTeacher")
@ResponseBody
public Map<String, Object> addStudent(Teacher teacher) {
 //判断工号是否已存在
 if (teacherService.findByTno(teacher) != null) {
 result.put("success", false);
 result.put("msg", "工号已存在! 请修改后重试!");
 return result;
 }
 if (teacherService.insert(teacher) > 0) {
 result.put("success", true);
 } else {
 result.put("success", false);
 result.put("msg", "添加失败! (ಥ_ಥ)服务器端发生异常!");
```

```
    }
    return result;
}
```

TeacherService

```
// 验证登录信息是否正确
Teacher login(LoginForm loginForm);

// TODO: 根据教师与班级名查询指定/全部教师列表信息
List<Teacher> selectList(Teacher teacher);
```

TeacherServiceImpl

```
@Override
public Teacher findByTno(Teacher teacher) {
    return teacherMapper.findByTno(teacher);
}

@Override
public int insert(Teacher teacher) {
    return teacherMapper.insert(teacher);
}
```

TeacherMapper

```
// TODO: 根据工号查询指定教师信息
Teacher findByTno(Teacher teacher);

// TODO: 添加教师信息
int insert(Teacher teacher);
```

TeacherMapper.xml

```
<!-- 根据工号查询指定教师信息 -->
<select id="findByTno" resultType="teacher" parameterType="com.cc.sms.bean.Teacher">
    SELECT id,
    tno,
    name,
    gender,
    password,
```

```
    email,
    telephone,
     address,
     portrait_path,
     clazz_name
     FROM tb_teacher
     WHERE tno = #{tno}
  </select>

  <!-- 添加教师信息 -->
  <insert id="insert" parameterType="teacher">
     INSERT INTO tb_teacher(tno, name, gender, password, email, telephone, address, portrait_path, clazz_name)
     VALUES (#{tno}, #{name}, #{gender}, #{password}, #{email}, #{telephone}, #{address}, #{portrait_path},
     #{clazz_name})
  </insert>
```

上传头像

```
/**
 * @description: 上传头像-原理:将头像上传到项目发布目录中,通过读取数据库中的头像路径来获取头像
 * @param: photo
 * @param: request
 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/uploadPhoto")
@ResponseBody
public Map<String, Object> uploadPhoto(MultipartFile photo, HttpServletRequest request) {
    //存储头像的本地目录
    final String dirPath = request.getServletContext().getRealPath("/upload/teacher_portrait/");
    //存储头像的项目发布目录
    final String portraitPath = request.getServletContext().getContextPath() + "/upload/teacher_portrait/";
    //返回头像的上传结果
    return UploadFile.getUploadResult(photo, dirPath, portraitPath);
}
```

教师信息编辑查看teacherList.jsp

```
$.ajax({
 type: "post",
 url: "editTeacher?t=" + new Date().getTime(),
 data: data,
 dataType: 'json',
```

TeacherController

```
@PostMapping("/editTeacher")
@ResponseBody
public Map<String, Object> editTeacher(Teacher teacher) {
 if (teacherService.update(teacher) > 0) {
 result.put("success", true);
 } else {
 result.put("success", false);
 result.put("msg", "修改失败! (ಥ_ಥ)服务器端发生异常!");
 }
  return result;
}
```

TeacherService

```
// TODO: 根据id修改指定教师信息
int update(Teacher teacher);
```

TeacherServiceImpl

```
@Override
public int update(Teacher teacher) {
 return teacherMapper.update(teacher);
}
```

TeacherMapper

```
// TODO: 根据id修改指定教师信息
int update(Teacher teacher);
```

TeacherMapper.xml

```
<!-- 根据id修改指定教师信息 -->
<update id="update" parameterType="teacher">
 UPDATE tb_teacher
 <set>
 <if test="name!=null and name!=''">name=#{name},</if>
 <if test="gender!=null and gender!=''">gender=#{gender},</if>
 <if test="password!=null and password!=''">password=#{password},</if>
 <if test="email!=null and email!=''">email=#{email},</if>
 <if test="telephone!=null and telephone!=''">telephone=#{telephone},
</if>
  <if test="address!=null and address!=''">address=#{address},</if>
  <if test="portrait_path!=null and portrait_path!=''">portrait_path=#{po
rtrait_path},</if>
  <if test="clazz_name!=null and clazz_name!=''">clazz_name=#
{clazz_name},</if>
 </set>
 WHERE id = #{id}
</update>
```

删除教师信息查看teacherList.jsp

```
$.ajax({
 type: "post",
 url: "deleteTeacher?t" + new Date().getTime(),
 data: {ids: ids},
 dataType: 'json',
```

TeacherController

```
@PostMapping("/deleteTeacher")
@ResponseBody
public Map<String, Object> deleteTeacher(@RequestParam(value = "ids[]", r
equired = true) Integer[] ids) {

 if (teacherService.deleteById(ids) > 0) {
 result.put("success", true);
 } else {
 result.put("success", false);
```

```
 result.put("msg",  "删除失败! (ಥ_ಥ)服务器端发生异常!");
  }
  return result;
 }
```

TeacherService

```
// TODO: 根据id删除指定教师信息
int deleteById(Integer[] ids);
```

TeacherServiceImpl

```
 @Override
 public int deleteById(Integer[] ids) {
 return teacherMapper.deleteById(ids);
 }

```

TeacherMapper

```
// TODO: 根据id删除指定教师信息
int deleteById(Integer[] ids);
```

TeacherMapper.xml

```
<!-- 根据id批量删除指定教师信息 -->
<delete id="deleteById">
 DELETE FROM tb_teacher WHERE id IN
 <foreach collection="array" item="ids" open="(" separator="," close=")">
 #{ids}
 </foreach>
</delete>
```

学生信息管理查看main.jsp

```
"menuid": "21",
"menuname": "学生列表",
"icon": "icon-student",
"url": "../student/goStudentListView"
```

StudentController

```java
@Controller
@RequestMapping("/student")
public class StudentController {

 // 注入业务对象
 @Autowired
 private ClazzService clazzService;
 @Autowired
 private StudentService studentService;

 //存储预返回页面的数据对象
 private Map<String, Object> result = new HashMap<>();

 /**
 * @description: 跳转到学生信息管理页面
 * @param: modelAndView

 * @return: org.springframework.web.servlet.ModelAndView
 */
 @GetMapping("/goStudentListView")
 public ModelAndView goStudentListView(ModelAndView modelAndView) {
 //向页面发送一个存储着Clazz的List对象
 modelAndView.addObject("clazzList", clazzService.selectAll());
 modelAndView.setViewName("student/studentList");
 return modelAndView;
 }
```

查看studentList.jsp 学生信息列表

```javascript
$('#dataList').datagrid({
 iconCls: 'icon-more',//图标
 border: true,
 collapsible: false,//是否可折叠
 fit: true,//自动大小
 method: "post",
 url: "getStudentList?t" + new Date().getTime(),
 idField: 'id',
```

StudentController

```java
/**
 * @description: 分页查询学生信息列表:根据学生与班级名查询指定/全部学生信息列表
 * @param: page
 * @param: rows
 * @param: studentname
 * @param: clazzname

 * @return: java.util.Map<java.lang.String, java.lang.Object>
 */
@PostMapping("/getStudentList")
@ResponseBody
public Map<String, Object> getStudentList(Integer page, Integer rows, String studentname, String clazzname) {

 //存储查询的studentname,clazzname信息
 Student student = new Student(studentname, clazzname);
 //设置每页的记录数
 PageHelper.startPage(page, rows);
 //根据班级与学生名获取指定或全部学生信息列表
 List<Student> list = studentService.selectList(student);
 //封装信息列表
 PageInfo<Student> pageInfo = new PageInfo<>(list);
 //获取总记录数
 long total = pageInfo.getTotal();
 //获取当前页数据列表
 List<Student> studentList = pageInfo.getList();
 //存储数据对象
 result.put("total", total);
 result.put("rows", studentList);

 return result;
}
```

StudentService

```java
// TODO: 根据班级与学生名获取指定或全部学生信息列表
List<Student> selectList(Student student);
```

StudentServiceImpl

```java
@Override
public List<Student> selectList(Student student) {
 return studentMapper.selectList(student);
```

```
}
```

StudentMapper

```
// TODO: 根据班级与学生名获取指定或全部学生信息列表
List<Student> selectList(Student student);
```

StudentMapper.xml

```
<!-- 根据班级与学生名获取指定/全部学生信息列表 -->
<select id="selectList" parameterType="student" resultType="com.cc.sms.bean.Student">
 SELECT id, sno, name, gender, password, portrait_path, email, telephone, address, introducation, clazz_name
 FROM tb_student
 <where>
 <if test="clazz_name!=null and clazz_name!=''">
 AND clazz_name = #{clazz_name}
 </if>
 <if test="name!=null and name!=''">
  AND name LIKE concat(concat('%',#{name}),'%')
  </if>
  </where>
</select>
```

学生信息管理添加,查看studentList.jsp

```
$.ajax({
 type: "post",
 url: "addStudent?t" + new Date().getTime(),
 data: data,
 dataType: 'json',
```

StudentController

```
@PostMapping("/addStudent")
@ResponseBody
public Map<String, Object> addStudent(Student student) {
 //判断学号是否已存在
 if (studentService.fingBySno(student) != null) {
```

```
 result.put("success", false);
 result.put("msg", "该学号已经存在! 请修改后重试!");
 return result;
 }
  //添加学生信息
  if (studentService.insert(student) > 0) {
  result.put("success", true);
  } else {
  result.put("success", false);
  result.put("msg", "添加失败! (ಥ_ಥ)服务器端发生异常!");
  }

  return result;
}
```

StudentService

```
// TODO: 根据学号获取指定学生信息
Student fingBySno(Student student);

// TODO: 添加班级信息
int insert(Student student);
```

StudentServiceImpl

```

@Override
public Student fingBySno(Student student) {
 return studentMapper.findBySno(student);
}
@Override
public int insert(Student student) {
 return studentMapper.insert(student);
}
```

StudentMapper

```
// TODO: 根据学号查询指定学生信息
Student findBySno(Student student);

```

```
// TODO: 添加班级信息
int insert(Student student);
```

StudentMapper.xml

```
<!-- 根据学号查询指定学生信息 -->
<select id="findBySno" resultType="student" parameterType="com.cc.sms.bean.Student">
 SELECT id,
 sno,
 name,
 gender,
 password,
 portrait_path,
 email,
 telephone,
 address,
 introducation,
 clazz_name
 FROM tb_student
 WHERE sno = #{sno}
</select>

<!-- 添加学生信息 -->
<insert id="insert" parameterType="student">
 INSERT INTO tb_student(sno, name, gender, password, portrait_path, email, telephone, address, introducation,
 clazz_name)
 VALUES (#{sno}, #{name}, #{gender}, #{password}, #{portrait_path}, #{email}, #{telephone}, #{address},
 #{introducation}, #{clazz_name})
</insert>
```

上传头像

```
/**
 * @description: 上传头像-原理:将头像上传到项目发布目录中,通过读取数据库中的头像路径来获取头像
 * @param: photo
 * @param: request
 *
 * @return: java.util.Map<java.lang.String, java.lang.Object>
```

```
 */
@PostMapping("/uploadPhoto")
@ResponseBody
 public Map<String, Object> uploadPhoto(MultipartFile photo, HttpServletRequest request) {
  //存储头像的本地目录
  final String dirPath = request.getServletContext().getRealPath("/upload/student_portrait/");
  //存储头像的项目发布目录
  final String portraitPath = request.getServletContext().getContextPath() + "/upload/student_portrait/";
  //返回头像的上传结果
  return UploadFile.getUploadResult(photo, dirPath, portraitPath);
 }
```

更新学生信息查看studentList.jsp

```
$.ajax({
 type: "post",
 url: "editStudent?t=" + new Date().getTime(),
 data: data,
 dataType: 'json',
```

StudentController

```
@PostMapping("/editStudent")
@ResponseBody
public Map<String, Object> editStudent(Student student) {
 if (studentService.update(student) > 0) {
 result.put("success", true);
 } else {
 result.put("success", false);
 result.put("msg", "添加失败! (ಥ_ಥ)服务器端发生异常!");
 }
  return result;
 }
```

StudentService

```
// TODO: 根据id修改指定学生信息
int update(Student student);
```

StudentServiceImpl

```
@Override
public int update(Student student) {
 return studentMapper.update(student);
}
```

StudentMapper

```
// TODO: 根据id修改指定学生信息
int update(Student student);
```

StudentMapper.xml

```
<!-- 根据id修改指定学生信息 -->
<update id="update" parameterType="student">
 UPDATE tb_student
 <set>
 <if test="name!=null and name!=''">name=#{name},</if>
 <if test="gender!=null and gender!=''">gender=#{gender},</if>
 <if test="password!=null and password!=''">password=#{password},</if>
 <if test="email!=null and email!=''">email=#{email},</if>
 <if test="telephone!=null and telephone!=''">telephone=#{telephone},
</if>
  <if test="address!=null and address!=''">address=#{address},</if>
  <if test="introducation!=null and introducation!=''">introducation=#{in
troducation},</if>
  <if test="portrait_path!=null and portrait_path!=''">portrait_path=#{po
rtrait_path},</if>
  <if test="clazz_name!=null and clazz_name!=''">clazz_name=#
{clazz_name},</if>
 </set>
 WHERE id = #{id}
</update>
```

删除学生信息查看studentL.

```
if (r) {
 $.ajax({
 type: "post",
 url: "deleteStudent?t" + new Date().getTime(),
```

```
5   data: {ids: ids},
6   dataType: 'json',
```

StudentController

```
1  /**
2   * @description: 根据id删除指定的学生信息
3   * @param: ids
4
5   * @return: java.util.Map<java.lang.String, java.lang.Object>
6   */
7  @PostMapping("/deleteStudent")
8  @ResponseBody
9  public Map<String, Object> deleteStudent(@RequestParam(value = "ids[]", r
equired = true) Integer[] ids) {
10
11   if (studentService.deleteById(ids) > 0) {
12   result.put("success", true);
13   } else {
14   result.put("success", false);
15   }
16   return result;
17 }
```

StudentService

```
1  // TODO: 根据id删除指定学生信息
2  int deleteById(Integer[] ids);
```

StudentServiceImpl

```
1   @Override
2   public int deleteById(Integer[] ids) {
3   return studentMapper.deleteById(ids);
4   }
5
```

StudentMapper

```java
// TODO: 根据id删除指定学生信息
int deleteById(Integer[] ids);
```

StudentMapper.xml

```xml
<!-- 根据id批量删除指定学生信息 -->
<delete id="deleteById">
 DELETE FROM tb_student WHERE id IN
 <foreach collection="array" item="ids" open="(" separator="," close=")">
 #{ids}
 </foreach>
</delete>
```

完结