武汉纺织大学

Web应用开发课程设计

**个人信息维护系统**

**学 院： 数学与计算机学院**

**班 级： 物联网11804**

**姓 名： 冯迁**

**学 号： 1804280417**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年12月15日**

目 录

[1 需求分析 1](#_Toc59201496)

[1.1用户登陆 1](#_Toc59201497)

[1.2主界面 1](#_Toc59201498)

[1.3各项功能 1](#_Toc59201499)

[2 系统实现 2](#_Toc59201500)

[2.1 概述 2](#_Toc59201501)

[2.2 框架 3](#_Toc59201502)

[3 系统实现 5](#_Toc59201503)

[3.1 项目结构 5](#_Toc59201504)

[3.2 配置文件 6](#_Toc59201505)

[3.2.1 jdbc.properties文件 6](#_Toc59201506)

[3.2.2 application-spring文件 6](#_Toc59201507)

[3.2.3 spring-mvc文件 7](#_Toc59201508)

[3.3 VO类User.java 8](#_Toc59201509)

[3.4 DAO接口类IUserDAO.java 10](#_Toc59201510)

[3.5 接口映射文件UserMapper.xml 10](#_Toc59201511)

[3.6 工具包Util 10](#_Toc59201512)

[3.6.1 LoginLogController.java 10](#_Toc59201513)

[3.6 服务层UserService.java 12](#_Toc59201514)

[3.7 modal.js 13](#_Toc59201515)

[4 系统测试 16](#_Toc59201516)

[5 系统总结 16](#_Toc59201517)

# 1 需求分析

设计一个用户管理界面，实现个人信息的显示与相关信息的修改功能（修改头像，手机，密保邮箱），登录历史信息的表格显示。登录历史信息与个人信息来自于数据库，具体要求如下：

## 1.1用户登陆

当程序运行时，直接进入用户界面，因为没有设计的太精细，所以直接进入用户界面，如果不是设置的用户， 则不会显示信息。

## 1.2主界面

当用户名正确的时候进入系统主界面，如下：

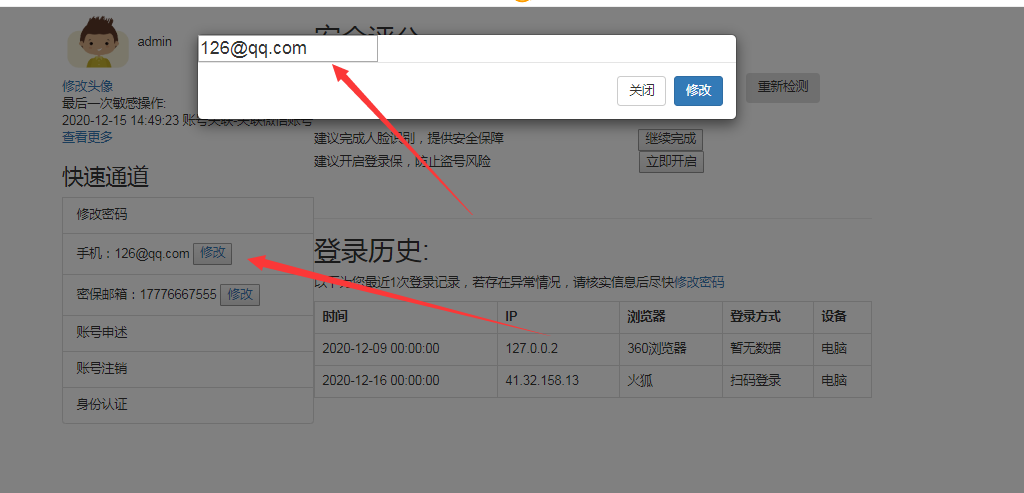


首先显示的是整体的一个界面，左上角是用户头像和名称，下面可以选择修改手机号和密保邮箱，其他的功能没有设计，右边设计了一个安全指数，评分为90分，再下面就是登录历史，通过数据库的数据显示

## 1.3各项功能

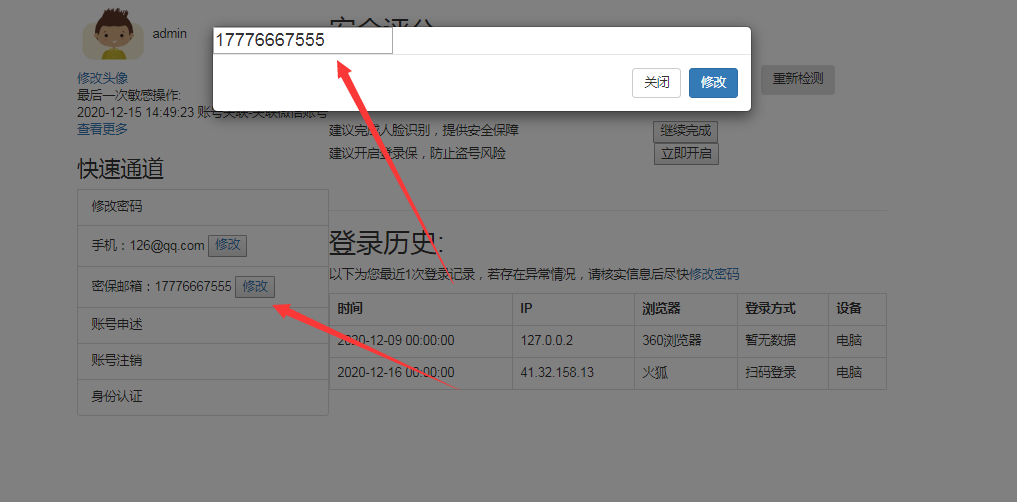
1.3.1 修改手机绑定号码

当用户发现自己账户绑定的手机号和自己的手机号不一致时，可以通过选择修改按钮来修改自己账户绑定的手机号，修改之后页面会显示自己的手机号码



1.3.2 修改用户绑定的邮箱

修改密保邮箱的步骤和修改绑定电话一致，只需点击修改按钮之后输入自己需要的邮箱即可



# 2 系统实现

## 2.1 概述

设计数据库

根据实际业务需要，设计表结构，字段，并插入数据；

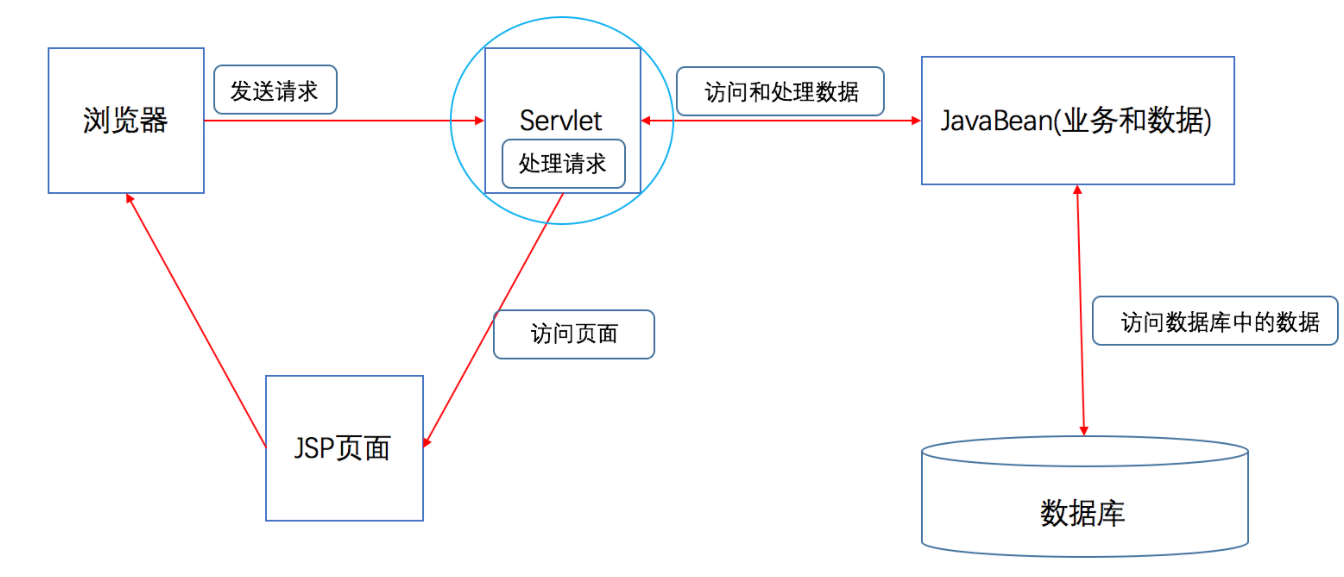
**创建好目录结构**

后台(src)：

* DAO : 数据访问层，创建接口，和接口的实现类
* Service : 业务逻辑层，创建接口，和接口的实现类，用来调用DAO层
* Controller : 工具类的包，比如：分页，读取properties的工具等
* database.properties : 连接数据库的信息

前台(web-root)：

* jsp页面 : 使用EL、JSTL表达式完善页面的表达
* CSS : 样式表
* JS : JS脚本
* IMG : 图片资源
* web.xml : 连接 servlet 和 jsp 页面，以及配置信息



## 2.2 框架

vue是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与[现代化的工具链](https://cn.vuejs.org/v2/guide/single-file-components.html)以及各种[支持类库](https://github.com/vuejs/awesome-vue#libraries--plugins)结合使用时，Vue 也完全能够为复杂的单页应用提供驱动。

AJAX 并不是一种新的编程语言，而仅仅是一种新的技术，它可以创建更好、更快且交互性更强的 web 应用程序。

AJAX 使用 JavaScript 在 web 浏览器与 web 服务器之间来发送和接收数据。

通过在幕后与 web 服务器交换数据，而不是每当用户作出改变时重载整个 web 页面，AJAX 技术可以使网页更迅速地响应。

Bootstrap 是最受欢迎的 HTML、CSS 和 JS 框架，用于开发响应式布局、移动设备优先的 WEB 项目。简单灵活可用于架构流行的用户界面和交互接口的html、css、javascript工具集。  
基于html5、css3的bootstrap，具有大量的诱人特性：友好的学习曲线，卓越的兼容性，响应式设计，12列格网，样式向导文档。  
自定义JQuery插件，完整的类库，基于Less等

Spring是一个JavaEE轻量级的一站式开发框架。

JavaEE： 就是用于开发B/S的程序。（企业级）

轻量级：使用最少代码启动框架，然后根据你的需求选择，选择你喜欢的模块使用。

重量级：早期有的EJB，开发一个HelloWorld程序都需要引入EBJ的全部模块

一站式：提供了，表示层（Struts2/SpringMVC），服务层，持久层的所有支持。

Spring Web MVC是一种基于Java的实现了Web MVC设计模式的请求驱动类型的轻量级Web框架，即使用了MVC架构模式的思想，将web层进行职责解耦，基于请求驱动指的就是使用请求-响应模型，框架的目的就是帮助我们简化开发，Spring Web MVC也是要简化我们日常Web开发的。

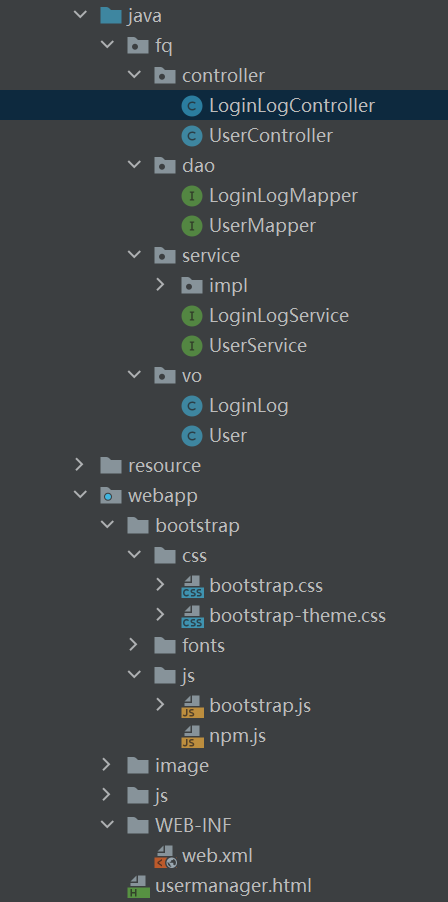
模型（Model ）封装了应用程序的数据和一般他们会组成的POJO。

视图（View）是负责呈现模型数据和一般它生成的HTML输出，客户端的浏览器能够解释。

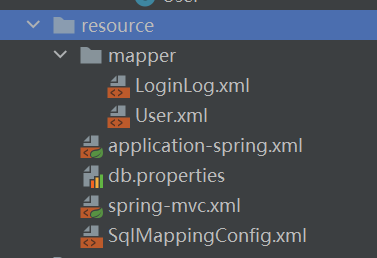
控制器（Controller ）负责处理用户的请求，并建立适当的模型，并把它传递给视图渲染。

# 3 系统实现

## 3.1 项目结构



## 3.2 配置文件



### 3.2.1 jdbc.properties文件

该配置文件主要是以文件形式保存数据库的驱动类名称，连接数据库的URL地址，访问数据库的用户名及对应的密码，程序运行时会读取该文件相关信息，避免硬编码，当相关信息发生变化时，只需修改配置文件而不用修改源代码，增加程序的可扩展型。

jdbc.driver=com.mysql.jdbc.Driver  
jdbc.url=jdbc:mysql://127.0.0.1/fq?characterEncoding=utf-8  
jdbc.username=root  
jdbc.password=123456

### 3.2.2 application-spring文件

描述该文件作用，并对里面的配置进行解释

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:p="http://www.springframework.org/schema/p"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xmlns:aop="http://www.springframework.org/schema/aop"  
 xmlns:mvc="http://www.springframework.org/schema/mvc"  
 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  
 http://www.springframework.org/schema/context/spring-context.xsd  
 http://www.springframework.org/schema/aop  
 http://www.springframework.org/schema/aop/spring-aop.xsd  
 http://www.springframework.org/schema/mvc  
 http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd">  
  
 <!--开启注解扫描-->  
 <context:component-scan base-package="fq">  
 <!--只扫描控制器-->  
 <context:include-filter type="annotation" expression="org.springframework.stereotype.Controller"/>  
 </context:component-scan>  
 <mvc:default-servlet-handler/>  
  
 <mvc:annotation-driven/>  
</beans>

### 3.2.3 spring-mvc文件

<?xml version="1.0" encoding="UTF-8" ?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:p="http://www.springframework.org/schema/p"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xmlns:aop="http://www.springframework.org/schema/aop"  
 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  
 http://www.springframework.org/schema/context/spring-context.xsd  
 http://www.springframework.org/schema/aop  
 http://www.springframework.org/schema/aop/spring-aop.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd">  
  
  
 <!--注解扫描-->  
 <context:component-scan base-package="fq">  
 <context:exclude-filter type="annotation"  
 expression="org.springframework.stereotype.Controller"/>  
 </context:component-scan>  
  
 <context:property-placeholder location="classpath:db.properties"/>  
 <!--连接池-->  
 <bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource">  
 <property name="driverClassName" value="${jdbc.driver}"/>  
 <property name="url" value="${jdbc.url}"/>  
 <property name="username" value="${jdbc.username}"/>  
 <property name="password" value="${jdbc.password}"/>  
 </bean>  
  
 <!--Mybatis工厂-->  
 <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">  
 <property name="dataSource" ref="dataSource"/>  
 <property name="configLocation" value="classpath:SqlMappingConfig.xml"/>  
 <property name="mapperLocations" value="classpath:mapper/\*.xml"/>  
 </bean>  
  
 <!--配置Mapper扫描-->  
 <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">  
 <property name="basePackage" value="fq.dao"/>  
 </bean>  
  
</beans>

## 3.3 VO类User.java

现在我们需要从数据库中查询用户列表t\_user,对应的实体类，但是前端页面需要展示更多个关于用户的消息,如用户的角色Role,而User实体类中的信息不全,  
为了返回更多的信息,有两种做法:1.直接在user类中添加需要的信息属性2.创建一个新类UserVO extends User,只在UserVO中添加更多属性,而且以VO结尾表示用于返回前端的数据.如果采用第一种做法,User类可能会越来越大,也不方便后期维护,而且User类作为对数据库表的映射,添加冗余的属性

package fq.vo;  
  
public class User {  
  
 private Integer id;  
 private String image;  
 private String phone;  
 private String email;  
 private String name;  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getImage() {  
 return image;  
 }  
  
 public void setImage(String image) {  
 this.image = image;  
 }  
  
 public String getPhone() {  
 return phone;  
 }  
  
 public void setPhone(String phone) {  
 this.phone = phone;  
 }  
  
 public String getEmail() {  
 return email;  
 }  
  
 public void setEmail(String email) {  
 this.email = email;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
}

## 3.4 DAO接口类IUserDAO.java

DAO层：DAO层主要是做数据持久层的工作，负责与数据库进行联络的一些任务都封装在此，DAO层的设计首先是设计DAO的接口，然后在Spring的配置文件中定义此接口的实现类，然后就可在模块中调用此接口来进行数据业务的处理，而不用关心此接口的具体实现类是哪个类，显得结构非常清晰，DAO层的数据源配置，以及有关数据库连接的参数都在Spring的配置文件中进行配置。

package fq.dao;  
  
import fq.vo.LoginLog;  
  
import java.util.List;  
  
public interface LoginLogMapper {  
  
 // 查询所有登录记录  
 List<LoginLog> queryAll();  
}

## 3.5 接口映射文件UserMapper.xml

修改用户信息

package fq.dao;  
  
import fq.vo.User;  
  
public interface UserMapper {  
  
 // 修改用户信息  
 int updateUser(User user);  
  
 // 登录  
 User login(User user);  
}

## 3.6 工具包Util

### 3.6.1 LoginLogController.java

控制层负责数据的接收请求和数据的返回

package fq.controller;  
  
import fq.service.LoginLogService;  
import fq.vo.LoginLog;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.ResponseBody;  
  
import java.util.List;  
  
  
@Controller  
public class LoginLogController {  
  
 @Autowired  
 private LoginLogService loginLogService;  
  
 // 查询所有登录记录  
 @RequestMapping("/queryAll")  
 @ResponseBody  
 public List<LoginLog> queryAll(){  
 List<LoginLog> loginLogs = loginLogService.queryAll();  
 return loginLogs;  
 }  
  
  
  
}

控制用户的信息

// 修改用户信息系  
@RequestMapping("/updateUser")  
@ResponseBody  
public int updateUser(User user){  
 int flag = userService.updateUser(user);  
 return flag;  
}  
  
// 用户登录  
@RequestMapping("/login")  
@ResponseBody  
public User login(User user){  
 User queryUser = userService.login(user);  
 return queryUser;

## 3.6 服务层UserService.java

在项目中，控制器完成的工作主要是流程控制，及请求相应，根据对应的结果返回对应的视图，具体的业务代码代码不应该写到控制器里，而是交给service负责实现。比如要验证一个用户名是否存在，应该在控制器里调用Service层的验证的方法，由service去验证用户名是否存在，而service再调用dao层，dao层跟底层的数据库打交道，查询出结果返回service层，service层返回结果给控制器层。控制层根据结果返回对应的资源。

**public** **class** UserService {

**private** IUserDAO userDAO = **null**;

**private** SqlSession sqlSession = **null**;

**public** UserService() {

**this**.sqlSession = MybatisUtils.*getSqlSession*();

**this**.userDAO = sqlSession.getMapper(IUserDAO.**class**);

}

// 登录检查，Map形式返回检查结果

**public** Map<String, Object> checkLogin(User user) **throws** Exception {

Map<String, Object> mapResult = **new** HashMap<String, Object>();

**try** {

User foundUser = **this**.userDAO.getById(user.getUserName());

**if** (foundUser == **null**) {

mapResult.put("code", 1);

mapResult.put("msg", "用户名不存在！");

} **else** {

String md5Password = MD5Util.*md5*(user.getPassword());

**if** (!foundUser.getPassword().equals(md5Password)) {

mapResult.put("code", 1);

mapResult.put("msg", "密码不正确！");

} **else** {

mapResult.put("code", 0);

mapResult.put("msg", "登录成功！");

mapResult.put("user", foundUser);

}

}

} **catch** (Exception e) {

mapResult.put("code", 1);

mapResult.put("msg", e.getMessage());

} **finally** { // 无论是否有异常，都需要关闭数据库会话

sqlSession.close();

}

**return** mapResult;

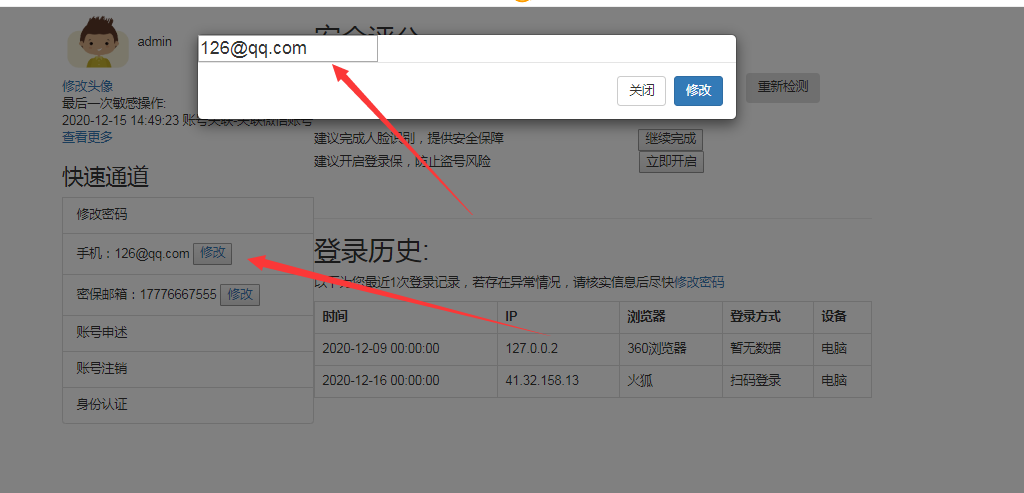
}

}

## 3.7 modal.js

模态框具有最小和最实用的功能集





在此处实现

通过网上的查找实现了列表组最简单的列表组仅仅是一个带有多个列表条目的无序列表，另外还需要设置适当的类。我们提供了一些预定义的样式，你可以根据自身的需求通过 CSS 自己定制。



<ul class="list-group"> <li class="list-group-item">Cras justo odio</li> <li class="list-group-item">Dapibus ac facilisis in</li> <li class="list-group-item">Morbi leo risus</li> <li class="list-group-item">Porta ac consectetur ac</li> <li class="list-group-item">Vestibulum at eros</li> </ul>

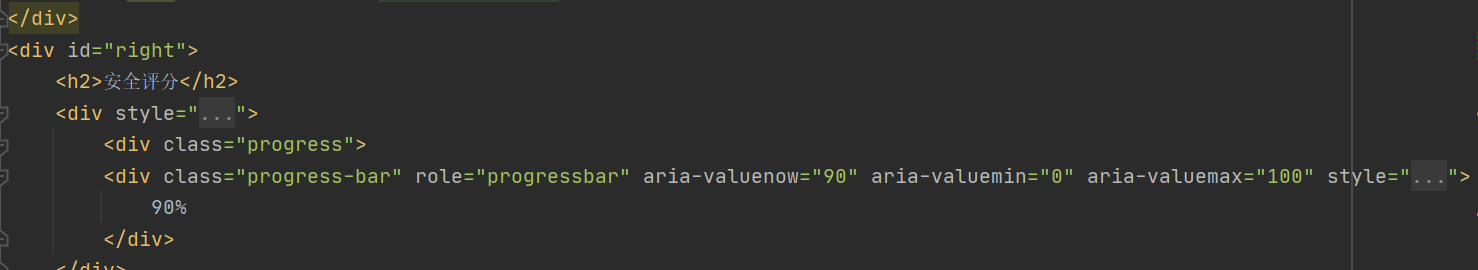


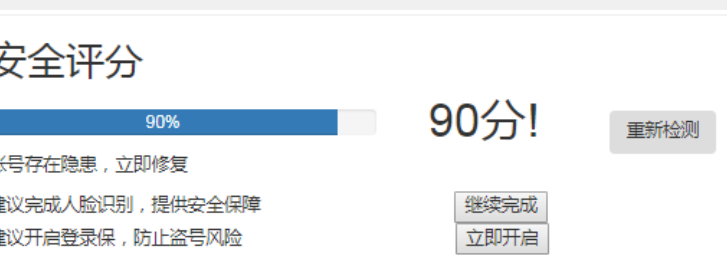


进度条组件使用了 CSS3 的 transition 和 animation 属性来完成一些特效。这些特性在 Internet Explorer 9 或以下版本中、Firefox 的老版本中没有被支持。Opera 12 不支持 animation 属性。



<div class="progress"> <div class="progress-bar" role="progressbar" aria-valuenow="60" aria-valuemin="0" aria-valuemax="100" style="width: 60%;"> 60% </div> </div>

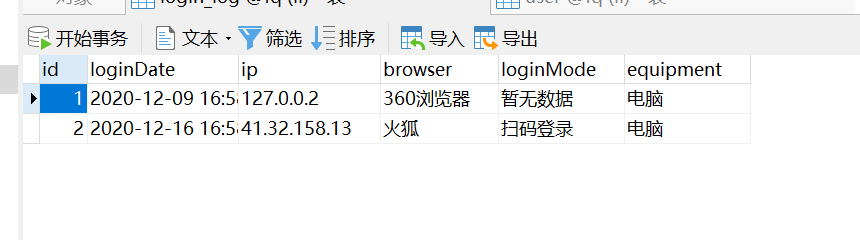
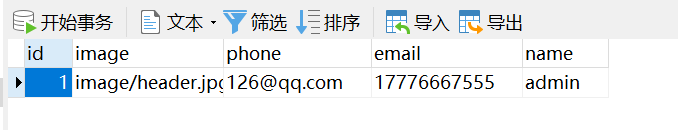




# 4 系统测试

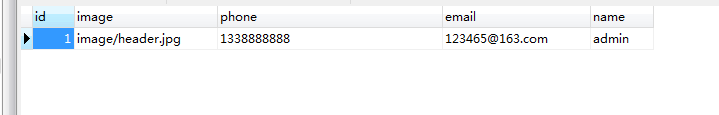
修改前



修改后





# 5 系统总结

我设计的时一个个人信息的界面，这个设计对我来说挑战还是挺大的，因为在没有Java的基础下就学习Javaweb是非常困难的，，幸好在老师的教导下还有同学的帮助下完成了这次的课设。俗话说的好“书山有路勤为径，学海无涯苦作舟”。在这一绝世佳句的鼓动下，我

去不断的收索;同时去不断的充实，去不断的完善自我，在网络的天空下逐渐的美化自

己的人生!在学习网页制作的过程中，我深深体会到一点“有付出，就终有回报”，你的付出

与你的回报终究是成正比的。经过了多少漫漫长路，经过了多少忘我的投入，经过了多

少波折，我的网页与众人见面了。当我第一眼在网络下一览我的个人主页，我是多么的

激动，多么的兴奋，就像多年未见的老同学，一种讲不出道不明的感觉。以后我会更加认真的学习每一门课程。