武汉纺织大学

Web应用开发课程设计

实验三

**学 院： 数学与计算机学院**

**班 级： 物联网11802**

**姓 名： 沈晓晗**

**学 号： 1804241003**

**指导老师： 聂刚**

**成 绩：**

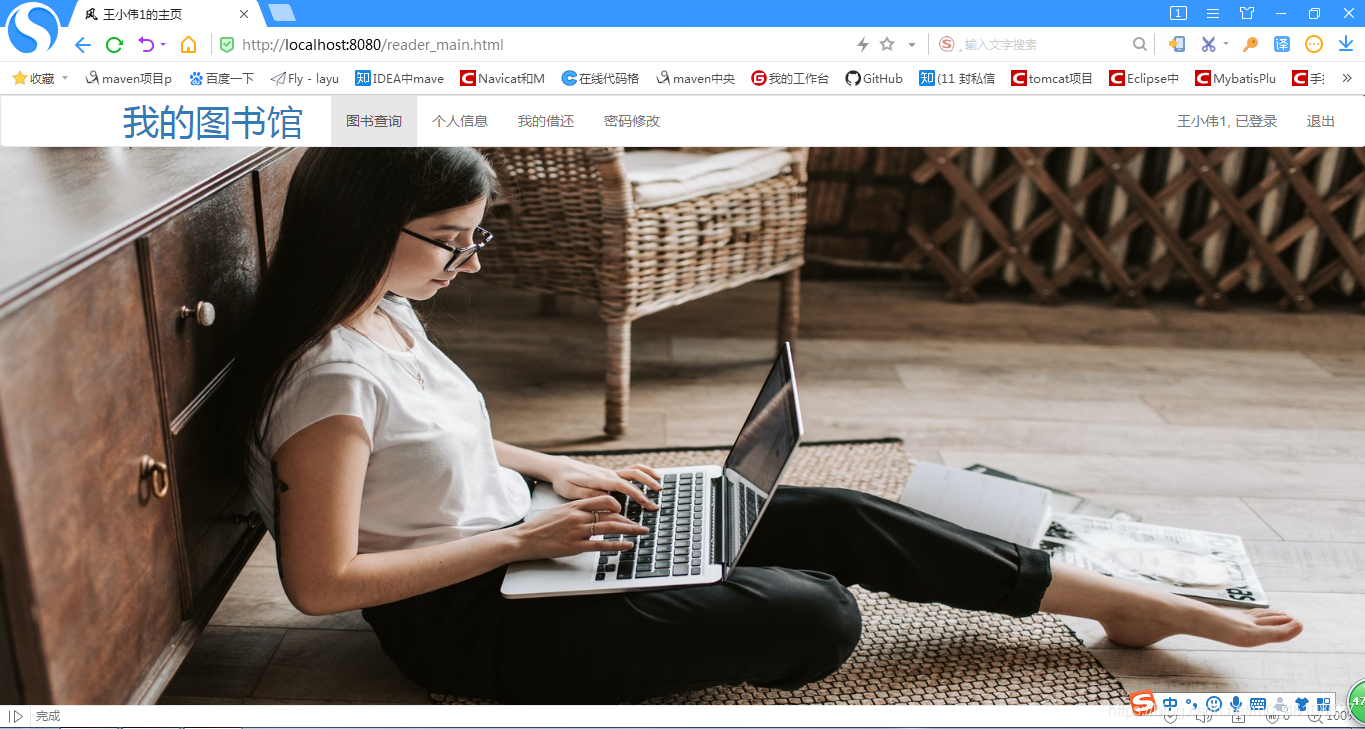
**完成日期： 2020年12月15日**

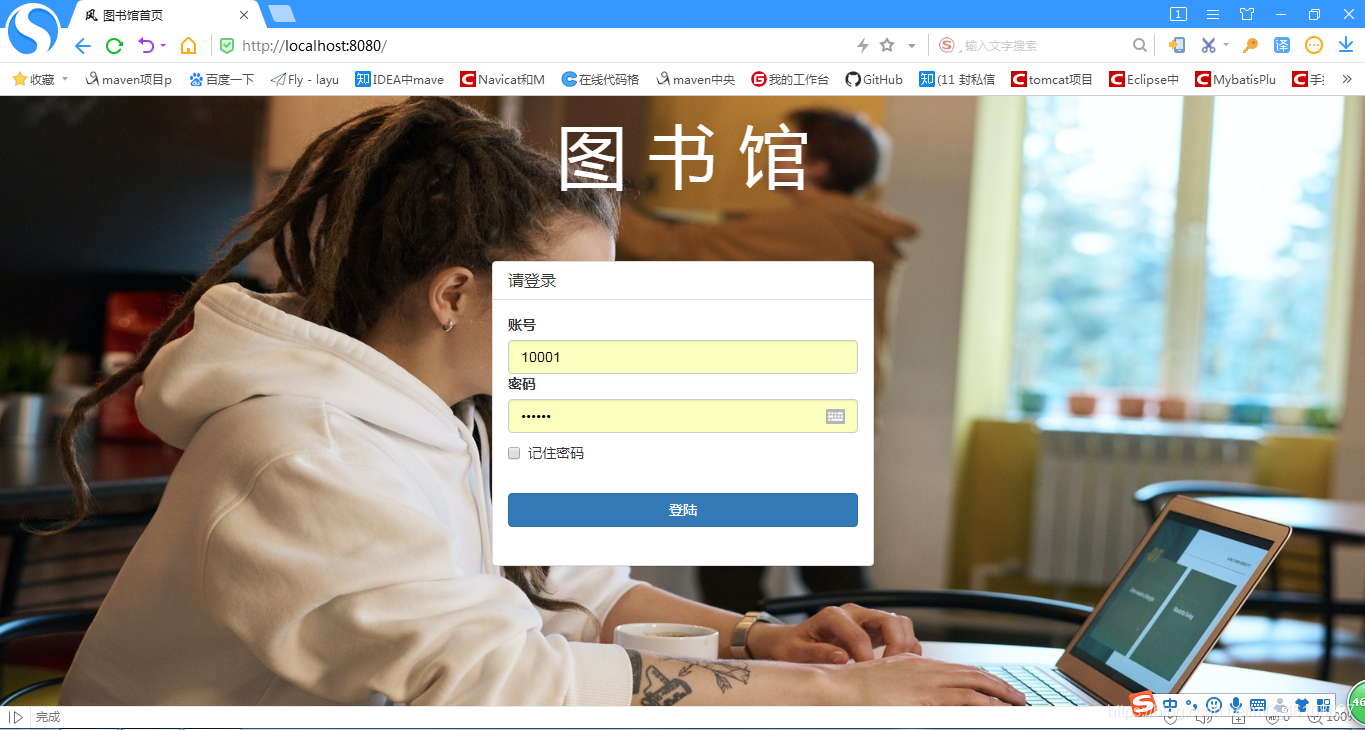
# 1 需求分析

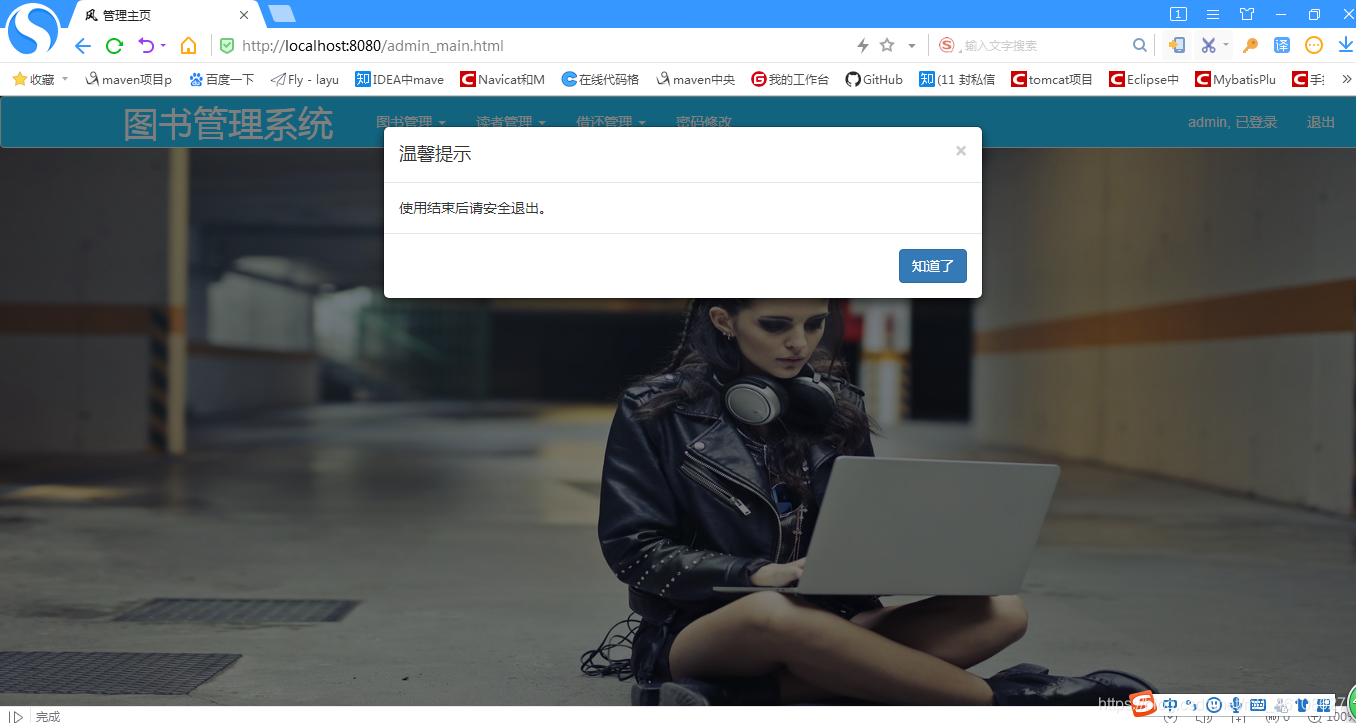
设计一个图书管理系统，实现“登陆”、“借书”、“查询统计”、“图书维护”、“密码修改”、“数据导出”等功能，具体要求如下：

## 1.1用户登陆

当程序运行时，显示“图书馆”，提示用户依次输入用户名及其密码，当用户名及密码不正确时，显示提示信息“用户名或密码不正确，请重新输入”，**用户最多可以尝试3次，若输入3次均不正确，则给出提示信息“最多只能尝试3次”**，退出系统；

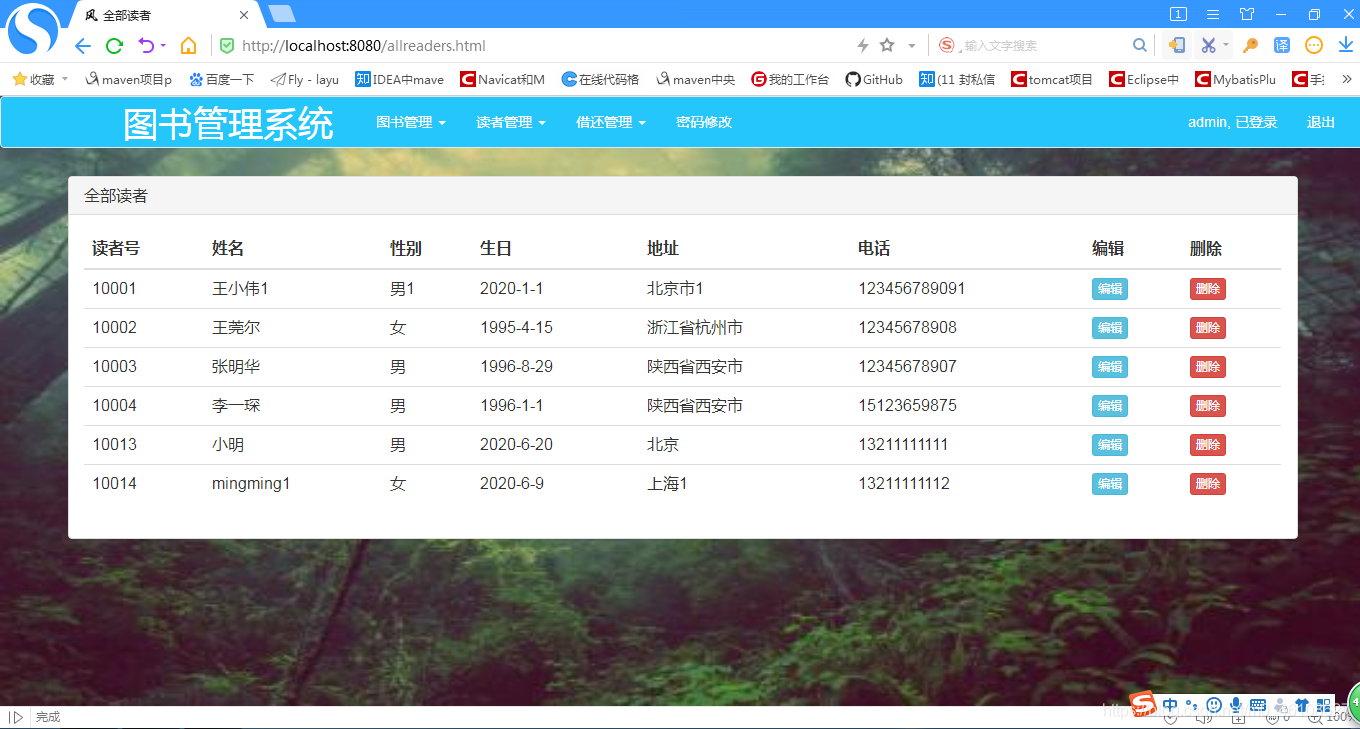






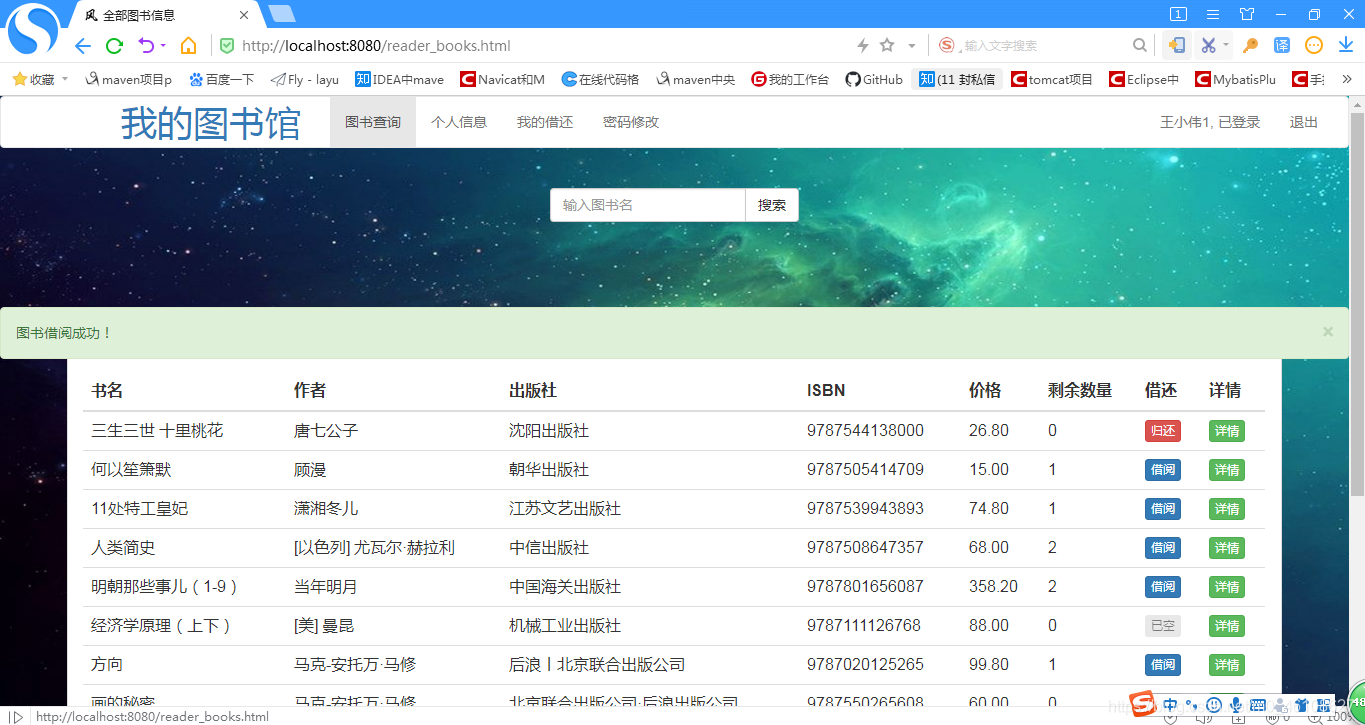
1.2读者管理

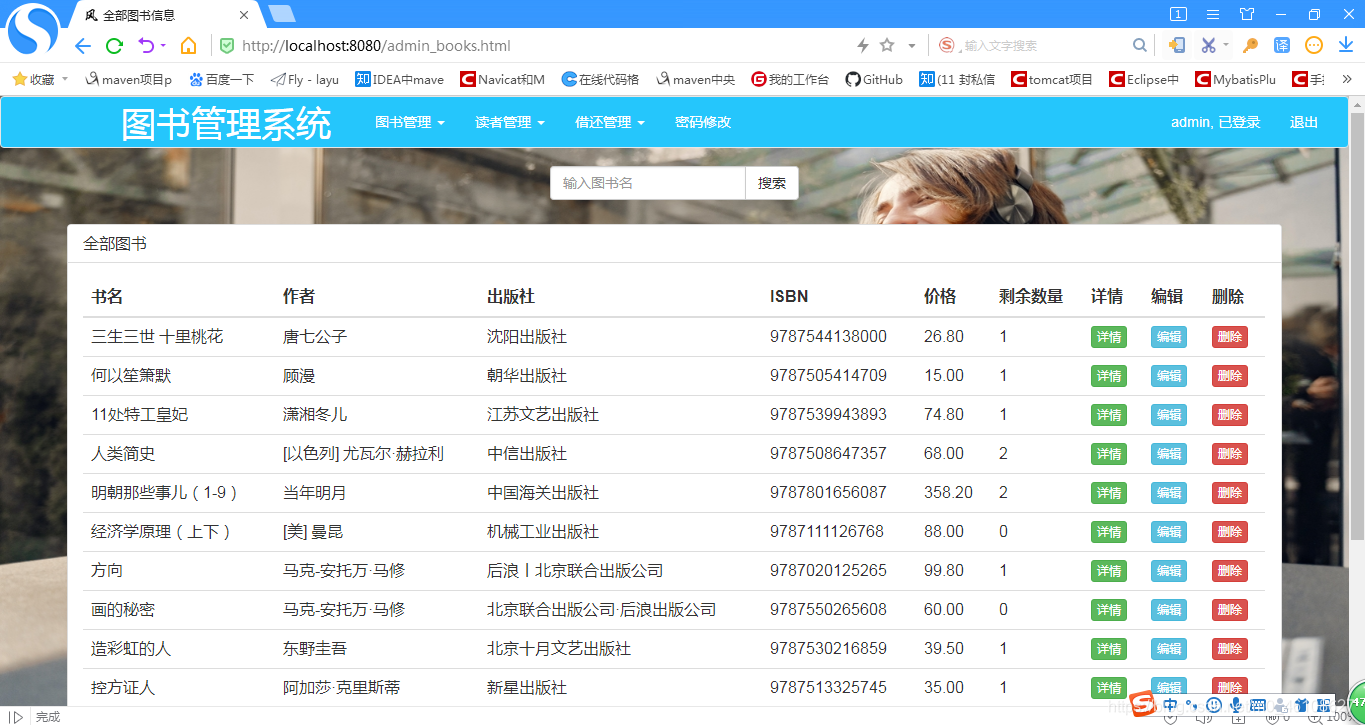
程序运行时，会有以下列表，记录读者的信息，可以添加，删除



1.3图书管理

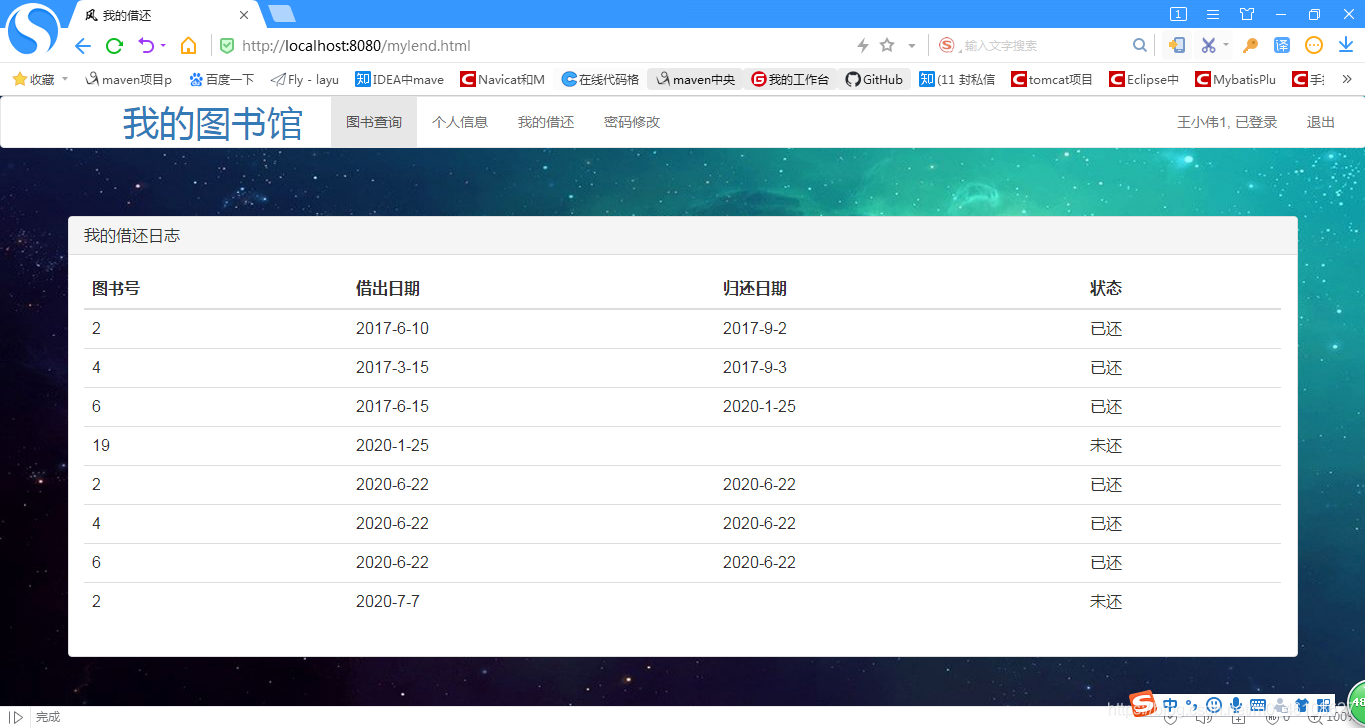
程序运行时，可以看到图书的信息，可以增添删除，查找





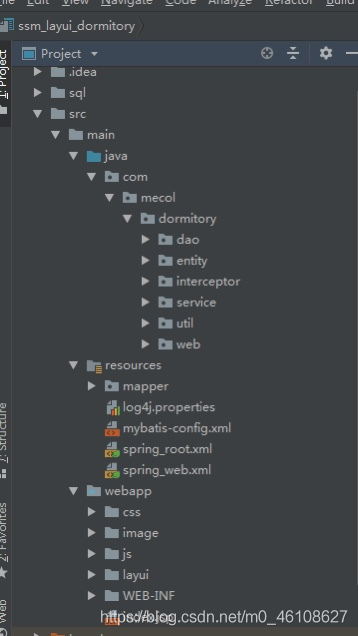
1.4借还记录

程序运行时，可以看到借还记录





2.完整的目录结构



3.mysql用到的表

分别是管理员表，图书信息表，图书分类表，借阅表，读者信息表，读者借阅表



4.完整的pom文件

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

<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/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.mecol.demo</groupId>

<artifactId>ssm\_mysql\_hotel</artifactId>

<version>1.0-SNAPSHOT</version>

<packaging>war</packaging>

<name>SSM\_demo Maven Webapp</name>

<!-- FIXME change it to the project's website -->

<url>http://www.example.com</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.7</maven.compiler.source>

<maven.compiler.target>1.7</maven.compiler.target>

</properties>

<dependencies>

<!-- spring 依赖包 -->

<!-- spring web jar包 -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>4.3.8.RELEASE</version>

</dependency>

<!-- spring aop jar包 -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>4.3.8.RELEASE</version>

</dependency>

<!-- spring 驱动jar包-->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>4.3.8.RELEASE</version>

</dependency>

<!-- spring mvc 依赖包 -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>4.3.8.RELEASE</version>

</dependency>

<!-- mybatis 依赖包 -->

<!--mybatis jar包 -->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis</artifactId>

<version>3.4.6</version>

</dependency>

<!-- mybatis spring-->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis-spring</artifactId>

<version>1.3.2</version>

</dependency>

<!-- 连接数据库驱动的jar包 我这里使用的是 jtds链接sqlserver数据库的驱动包 -->

<dependency>

<groupId>net.sourceforge.jtds</groupId>

<artifactId>jtds</artifactId>

<version>1.2.4</version>

</dependency>

<!-- MySql -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.11</version>

</dependency>

<!-- 连接接池 c3p0 -->

<dependency>

<groupId>com.mchange</groupId>

<artifactId>c3p0</artifactId>

<version>0.9.5.2</version>

</dependency>

<!--调试输出的log4j-->

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.17</version>

</dependency>

<!-- jstl jar包 -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<!-- jackson begin -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>2.8.8</version>

</dependency>

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-lang3</artifactId>

<version>3.7</version>

</dependency>

<dependency>

<groupId>org.codehaus.groovy</groupId>

<artifactId>groovy-all</artifactId>

<version>2.2.2</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi</artifactId>

<version>3.7</version>

</dependency>

<!--下面两个包 解决采用@ResponseBody 前台ajax请求 报错406-->

<dependency>

<groupId>org.codehaus.jackson</groupId>

<artifactId>jackson-core-asl</artifactId>

<version>1.8.5</version>

</dependency>

<dependency>

<groupId>org.codehaus.jackson</groupId>

<artifactId>jackson-mapper-asl</artifactId>

<version>1.8.5</version>

</dependency>

<!--分页查询的工具-->

<dependency>

<groupId>com.github.pagehelper</groupId>

<artifactId>pagehelper</artifactId>

<version>4.1.4</version>

</dependency>

<!-- json处理工具包 -->

<dependency>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

<version>2.8.5</version>

</dependency>

<dependency>

<groupId>commons-io</groupId>

<artifactId>commons-io</artifactId>

<version>2.6</version>

</dependency>

<dependency>

<groupId>commons-fileupload</groupId>

<artifactId>commons-fileupload</artifactId>

<version>1.4</version>

</dependency>

<!-- 单元测试 -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<finalName>SSM\_demo</finalName>

<pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->

<plugins>

<plugin>

<artifactId>maven-clean-plugin</artifactId>

<version>3.1.0</version>

</plugin>

<!-- see http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_war\_packaging -->

<plugin>

<artifactId>maven-resources-plugin</artifactId>

<version>3.0.2</version>

</plugin>

<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>

</project>

5.spring的配置文件

web. xml

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

<web-app version="2.4"

xmlns="http://java.sun.com/xml/ns/j2ee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee

http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd">

<display-name>Archetype Created Web Application</display-name>

<!--配置 Spring 容器-->

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:spring\_root.xml</param-value>

</context-param>

<!--中文乱码处理过滤-->

<!-- 编码过滤器 -->

<filter>

<filter-name>characterEncodingFilter</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>

</filter>

<filter-mapping>

<filter-name>characterEncodingFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

<!-- Jasperreport Config Begin -->

<servlet>

<servlet-name>JasperImageServlet</servlet-name>

<servlet-class>net.sf.jasperreports.j2ee.servlets.ImageServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>JasperImageServlet</servlet-name>

<url-pattern>/servlets/image</url-pattern>

</servlet-mapping>

<!-- Jasperreport Config End -->

<!--配置 Spring mvc 容器-->

<!--将所有请求，都交给Spring MVC 处理-->

<servlet>

<servlet-name>webs</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:spring\_web.xml</param-value>

</init-param>

</servlet>

<!--将 请求 截取 转交给 webs -->

<servlet-mapping>

<servlet-name>webs</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

</web-app>

<?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:contxt="http://www.springframework.org/schema/context" xmlns:p="http://www.springframework.org/schema/p"

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/tx http://www.springframework.org/schema/tx/spring-tx.xsd">

<!-- 扫描 service 包下面使用@Component/@Service/@Repository/@Controller 注解的类 将其实例化，放入容器中 -->

<contxt:component-scan base-package="com.mecol.hotel.service" />

<!--创建数据源-->

<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">

<!-- 数据库驱动 -->

<property name="driverClass" value="com.mysql.jdbc.Driver" />

<!-- 数据库连接 -->

<property name="jdbcUrl" value="jdbc:mysql://127.0.0.1:3306/hotel?characterEncoding=utf8&amp;useSSL=false&amp;serverTimezone=CTT"/>

<!-- 数据库用户名 -->

<property name="user" value="root"/>

<!-- 数据库密码 -->

<property name="password" value="123489"/>

</bean>

<!-- 配置mybatis -->

<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">

<!-- 数据源的实例化 -->

<property name="dataSource" ref="dataSource"/>

<!-- 映射 DML 语法 只要文件夹变蓝色 就是classpath 所以要有个mapper文件夹 下面所有的.xml都会映射-->

<property name="mapperLocations" value="classpath:mapper/\*.xml"/>

<!--加载mybatis自身的配置文件-->

<property name="configLocation" value="classpath:mybatis-config.xml"/>

</bean>

<!-- 映射 数据访问层 ( dao ) 中的接口 -->

<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">

<property name="sqlSessionFactoryBeanName" value="sqlSessionFactory"/>

<property name="basePackage" value="com.mecol.hotel.dao"/>

</bean>

<!-- 文件上传 -->

<bean id="multipartResolver" class="org.springframework.web.multipart.commons.CommonsMultipartResolver">

<!-- 设置上传文件的最大尺寸为10MB -->

<property name="maxUploadSize">

<value>10485760</value>

</property>

<!--设置编码，与jsp页面编码一致-->

<property name="defaultEncoding">

<value>UTF-8</value>

</property>

</bean>

<!-- JDBC事务管理器 -->

<bean id="transactionManager"

class="org.springframework.jdbc.datasource.DataSourceTransactionManager"

p:dataSource-ref="dataSource"/>

<!-- 启用支持annotation注解方式事务管理 -->

<tx:annotation-driven transaction-manager="transactionManager"/>

</beans>

<?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:contxt="http://www.springframework.org/schema/context" xmlns:p="http://www.springframework.org/schema/p"

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/tx http://www.springframework.org/schema/tx/spring-tx.xsd">

<!-- 扫描 service 包下面使用@Component/@Service/@Repository/@Controller 注解的类 将其实例化，放入容器中 -->

<contxt:component-scan base-package="com.mecol.hotel.service" />

<!--创建数据源-->

<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">

<!-- 数据库驱动 -->

<property name="driverClass" value="com.mysql.jdbc.Driver" />

<!-- 数据库连接 -->

<property name="jdbcUrl" value="jdbc:mysql://127.0.0.1:3306/hotel?characterEncoding=utf8&amp;useSSL=false&amp;serverTimezone=CTT"/>

<!-- 数据库用户名 -->

<property name="user" value="root"/>

<!-- 数据库密码 -->

<property name="password" value="123489"/>

</bean>

<!-- 配置mybatis -->

<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">

<!-- 数据源的实例化 -->

<property name="dataSource" ref="dataSource"/>

<!-- 映射 DML 语法 只要文件夹变蓝色 就是classpath 所以要有个mapper文件夹 下面所有的.xml都会映射-->

<property name="mapperLocations" value="classpath:mapper/\*.xml"/>

<!--加载mybatis自身的配置文件-->

<property name="configLocation" value="classpath:mybatis-config.xml"/>

</bean>

<!-- 映射 数据访问层 ( dao ) 中的接口 -->

<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">

<property name="sqlSessionFactoryBeanName" value="sqlSessionFactory"/>

<property name="basePackage" value="com.mecol.hotel.dao"/>

</bean>

<!-- 文件上传 -->

<bean id="multipartResolver" class="org.springframework.web.multipart.commons.CommonsMultipartResolver">

<!-- 设置上传文件的最大尺寸为10MB -->

<property name="maxUploadSize">

<value>10485760</value>

</property>

<!--设置编码，与jsp页面编码一致-->

<property name="defaultEncoding">

<value>UTF-8</value>

</property>

</bean>

<!-- JDBC事务管理器 -->

<bean id="transactionManager"

class="org.springframework.jdbc.datasource.DataSourceTransactionManager"

p:dataSource-ref="dataSource"/>

<!-- 启用支持annotation注解方式事务管理 -->

<tx:annotation-driven transaction-manager="transactionManager"/>

</beans>

5总结

自己开发过程中遇到的各种失误总结一下：

把用户资料的修改和修改密码写在了同一个页面，这样修改用户资料的同时发现密码全都被重置了，原理很简单就是提交表单的同时，如果密码框中是空就把值为空的密码传递到了数据库，密码当然被修改了。犯的最低等的错误就是把’= =’写成了赋值的=，并且还不止一次，这个真是害死人啊，检查了一遍又一遍还找不错误的所在，等发现的时候自己都乐了，这么低级的错误真是太可怕了。

Web程序的开发我认为大约分4个层次:

1.表现层 (represent layer)

2.控制层 (logic control layer)

3.业务逻辑层 (service layer)

4.数据存储层 (persistent layer)

一个标准的系统大致就是做3件事,I(Input)P(Process)O(Output),也就是输入,处理,输出.

由于开发,部署,移植,性能和代码可重用性的考虑,Web开发将IPO分为了若干层次.:

1.表现层:

此层的主要作用是:向用户展示信息,并且得到用户输入数据和向用户展示处理后的反馈.

2.控制层:

此层的主要作用是:为了让开发人员和维护人员方便控制Web页面的流向,一目了然的对其走向进行控制.同时此层也可以进行一些简单的预处理,使业务逻辑避开本不该它们触碰的外部检测.此层的大部分任务是程序走向的控制,小部分任务是一般预处理和检测功能.

3.业务逻辑层:

此层的主要作用是:进行用户所要关心的业务逻辑,进行整个程序的核心业务处理,此层一般会使用从表现层传入的数据并调用数据存储层的接口来进行相应的查询和更新删除保存功能.并将最终处理结果反馈给控制层,由控制层根据处理结果去寻找表现给用户的路径.

4.数据存储层:

此层的主要作用是:进行数据的查询和持久化过程.