

一、实验步骤

4.1、创建项目以及配置环境

创建项目的步骤适应 IntelliJ IDEA 2020.2 版本。

4.1.1、先创建一个普通项目，选择 New Project > Java，点击 Next，之后设置项目的名称和路径。

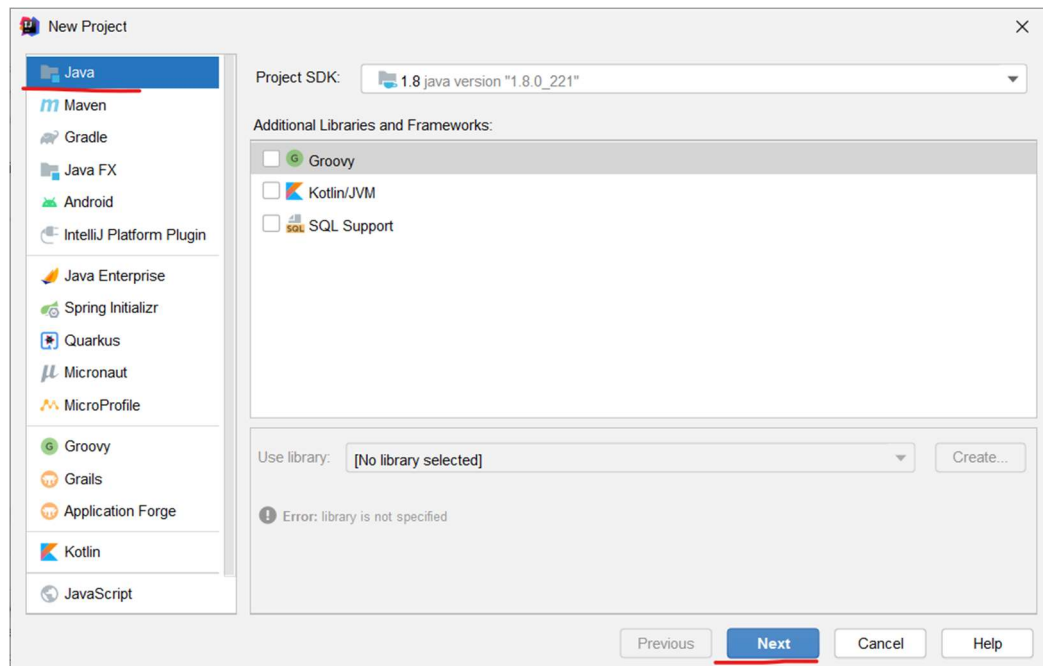


图 1 创建项目

4.2.2、创建项目结构，右键项目，点击 Add Framework Support..., 勾选 JavaEE 下的 Web Application。

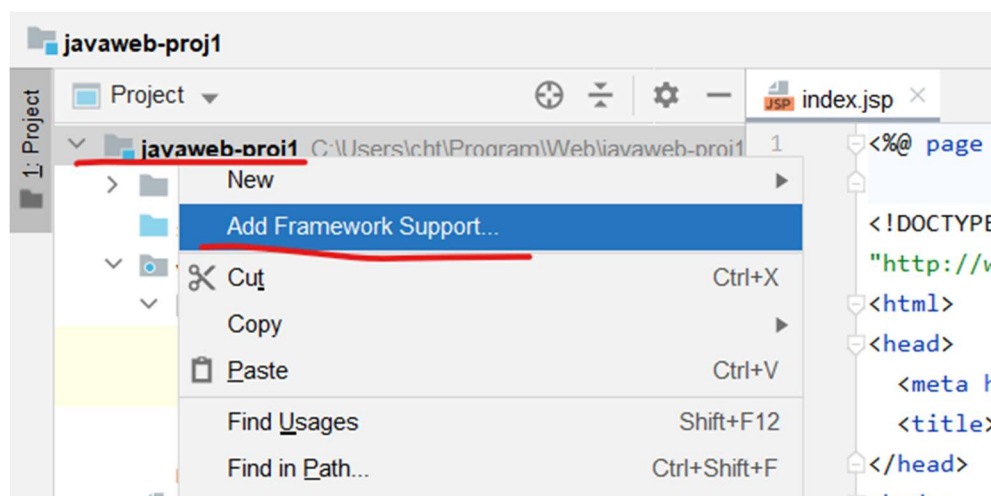


图 2 创建项目

4.2.3、创建网页文件夹，再 web>WEB-INF 下创建 classes 文件夹和 lib 文件夹。

4.2.4、修改项目的 Project Struture, 再 Module> Sources 下设置 src 为 Source Folders, lib 为 Excluded, 修改 Path 和 Dependencies

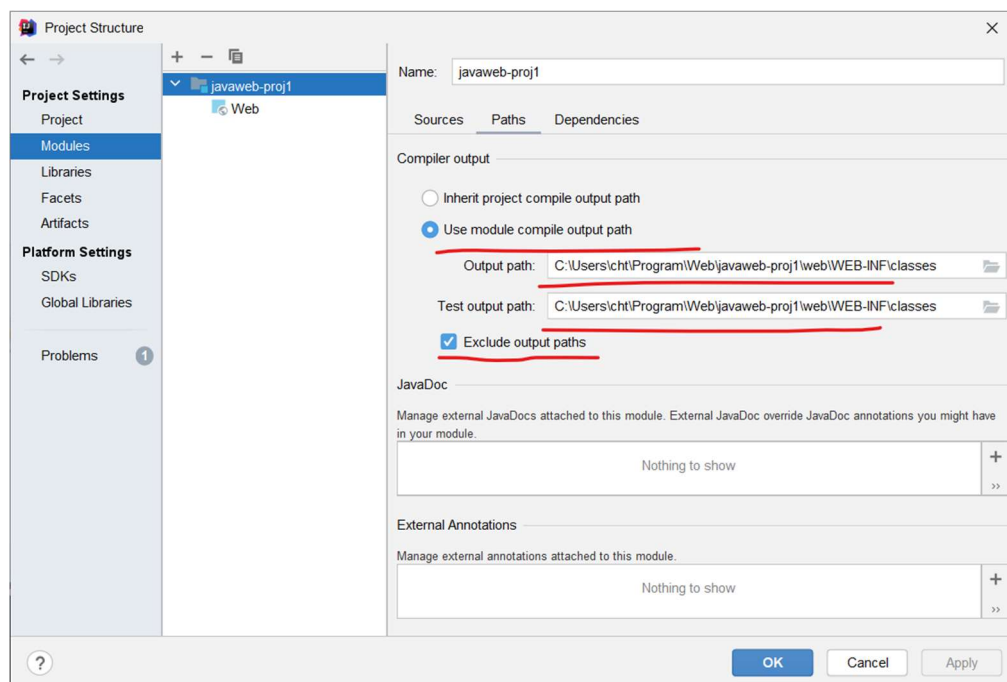


图 3 创建项目

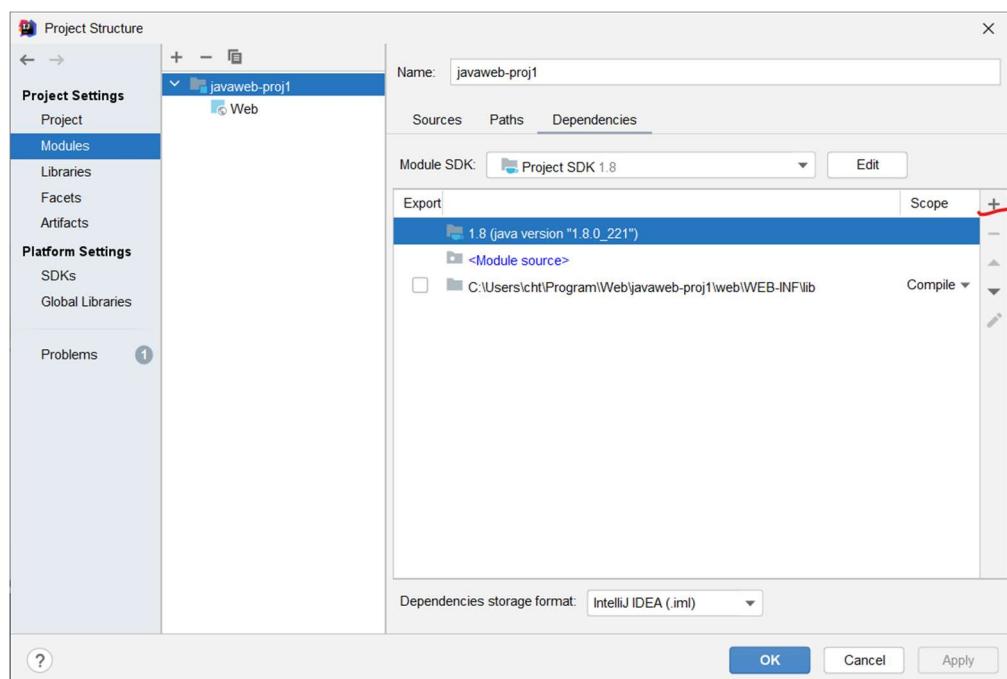


图 4 创建项目

4.2.5、配置 lib，找到 tomcat 下的 lib 文件夹，将 jsp-api.jar 和 servlet-api.jar

复制到 lib 文件夹下，然后再 Project Structure > Library 中导入此文件夹。

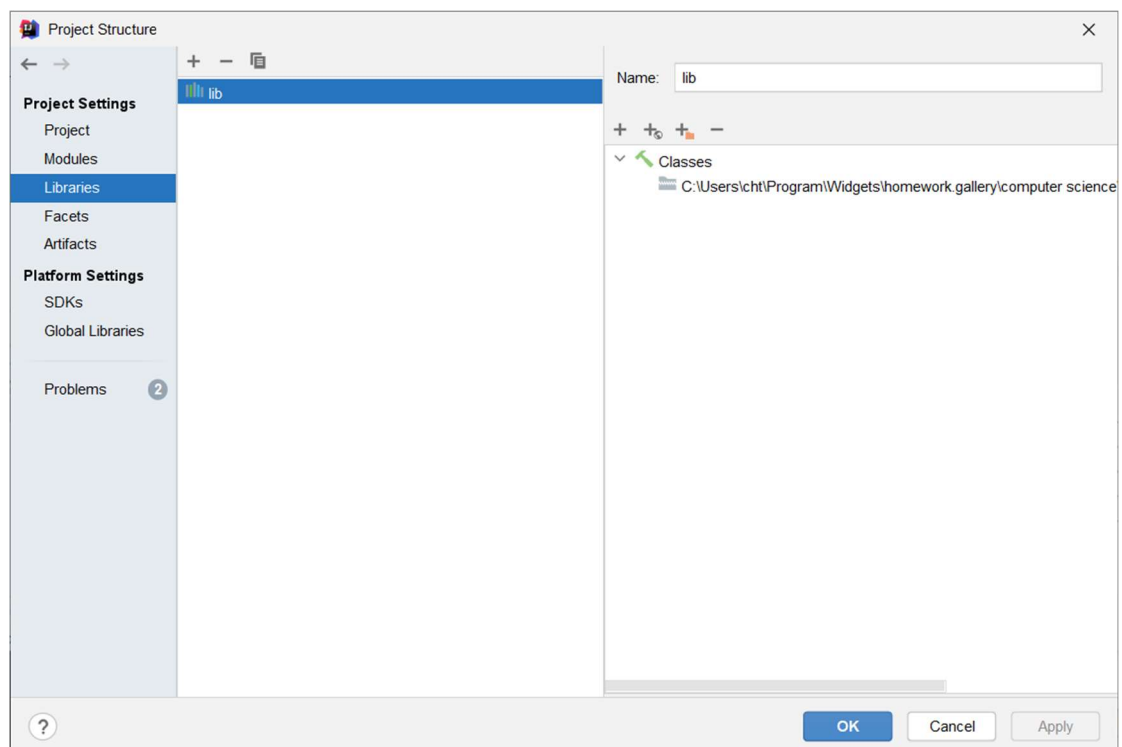


图 5 创建项目

4.2、实验一

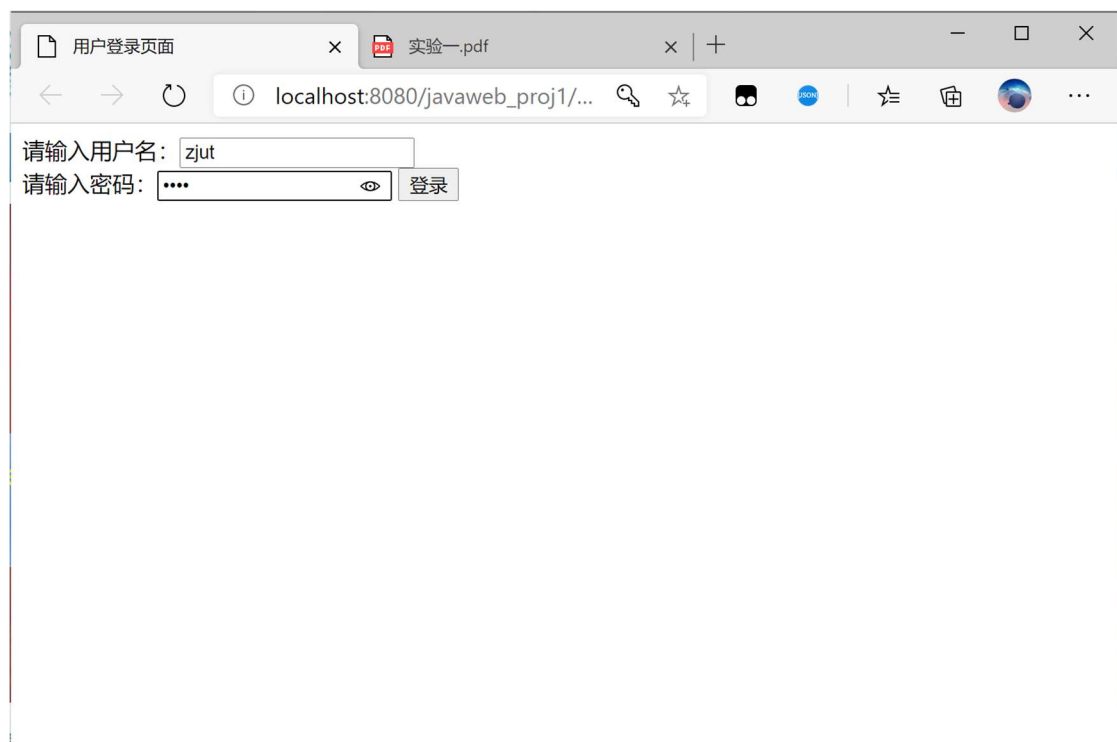


图 6 实验 1 执行结果

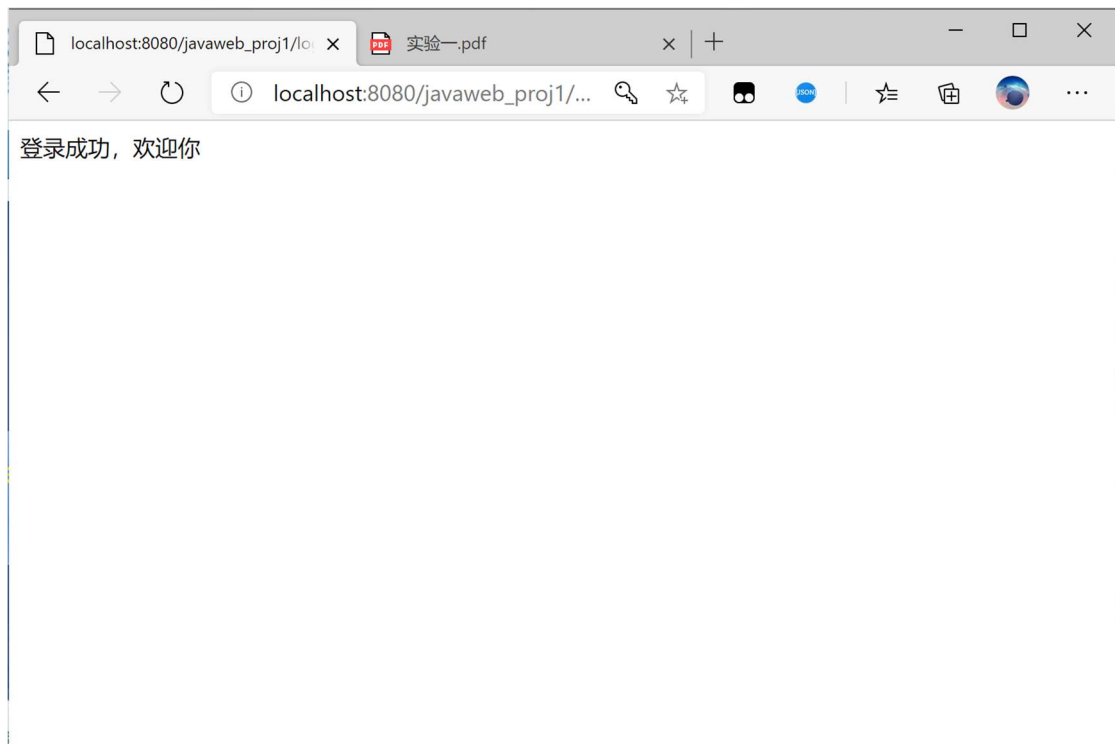


图 7 实验 1 执行结果

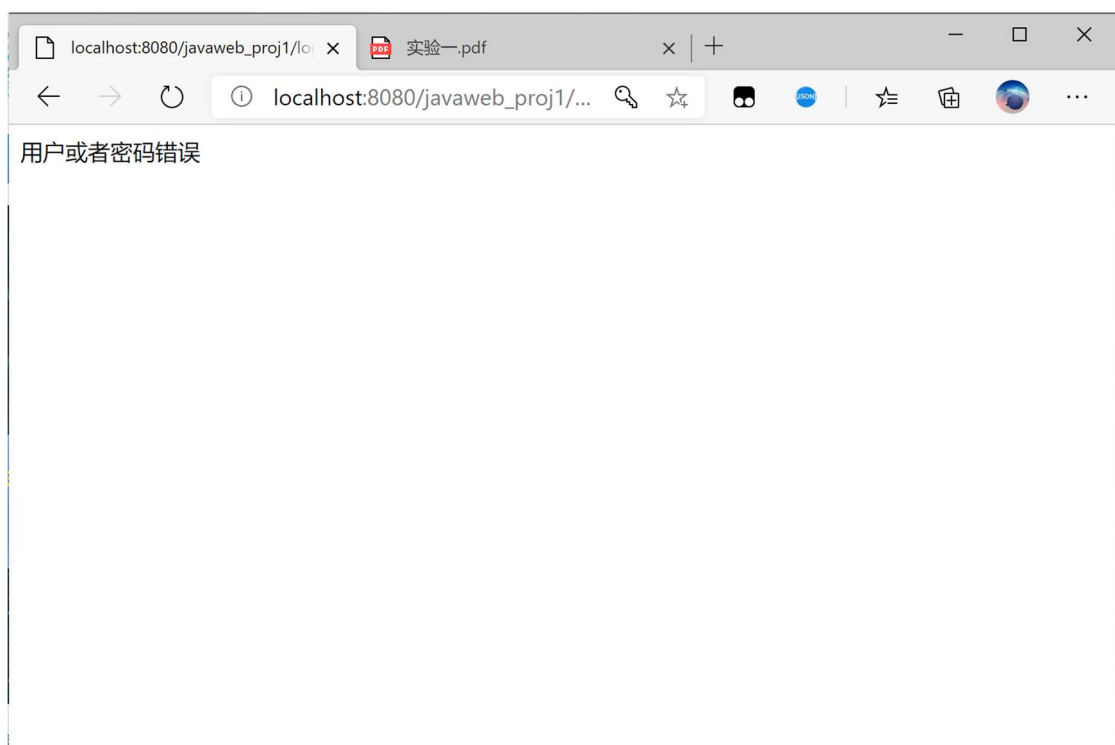


图 8 实验 1 执行结果

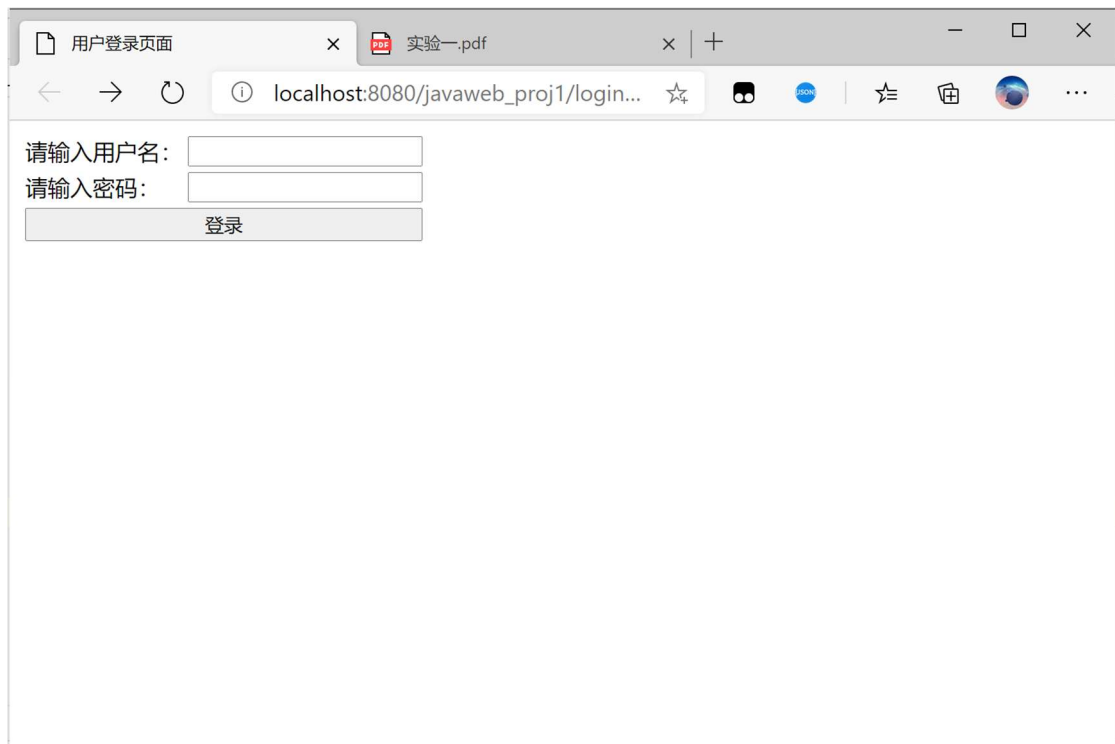


图 9 实验 1 表格对齐实验结果

关键代码：

```
<table>
<tbody>
  <tr>
    <td>用户类型:</td>
    <td>
      <select name="type">
        <option value="2">普通用户</option>
        <option value="1">管理员</option>
      </select>
    </td>
  </tr>
  <tr>
    <td>请输入用户名: </td>
    <td><input name="username" type="text"></td>
  </tr>
  <tr>
    <td>请输入密码: </td>
    <td><input name="password" type="password"></td>
  </tr>
  <tr>
    <td colspan="2"><input type="submit" value="登录" style="width:
100%"></td>
  </tr>
```

```
</tbody>  
</table>
```

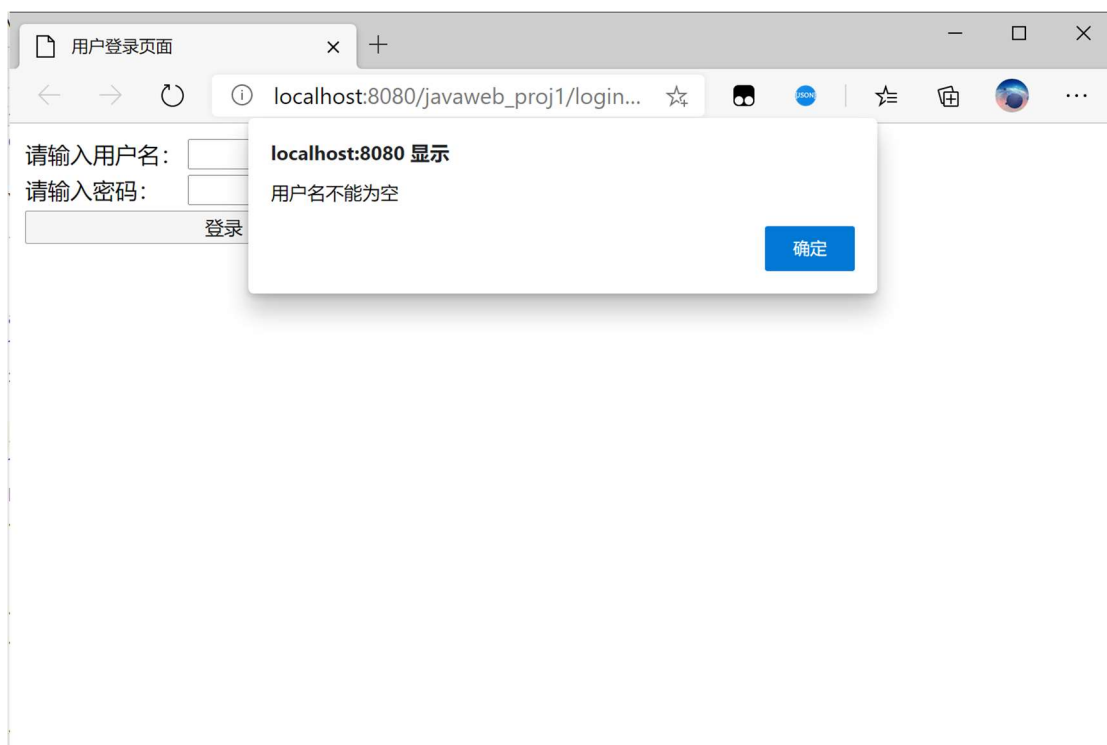


图 10 实验 1 前端 js 验证实验结果

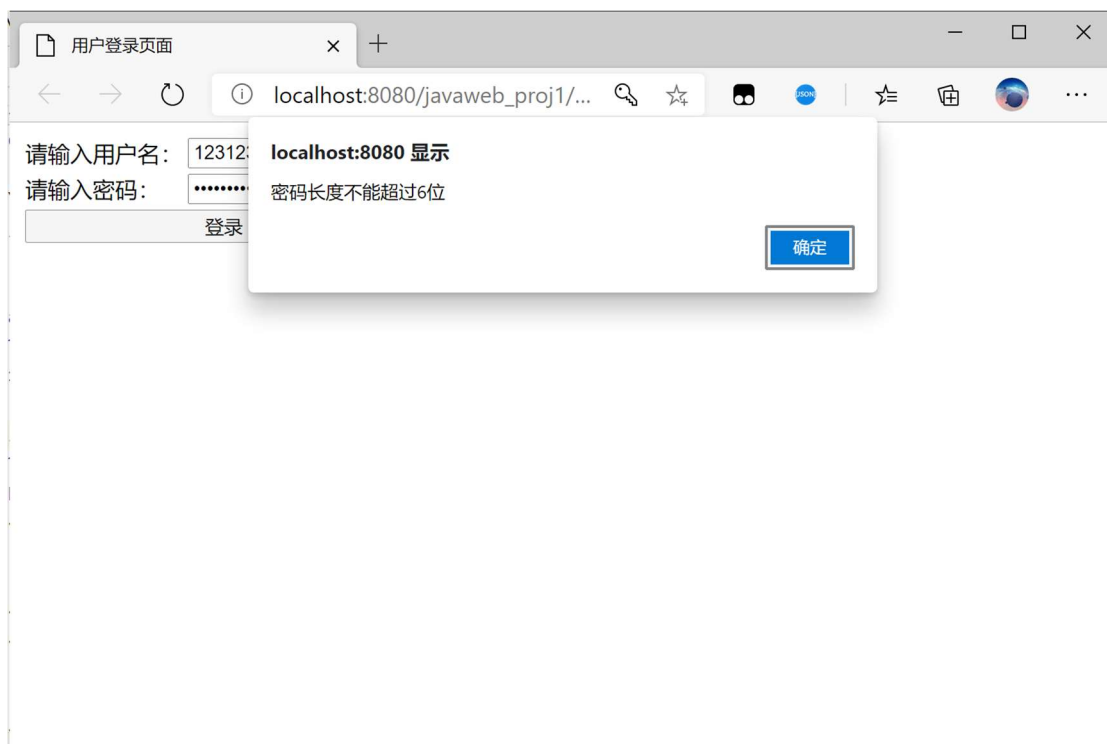


图 11 实验 1 前端 js 验证实验结果

关键代码：

```
/* functions */
function validateUser(username, password){
/*
    if (!(username instanceof String && password instanceof
String)){
        throw new TypeError("username and password should be String
instance.")
    }
*/

    if (username.length === 0) {
        return "用户名不能为空"
    } else if (username.length > 6){
        return "用户名长度不能超过 6 位"
    }

    if (password.length === 0) {
        return "密码不能为空"
    } else if (password.length > 6){
        return "密码长度不能超过 6 位"
    }

    return true /* pass validation */
}
```

4.3、实验二

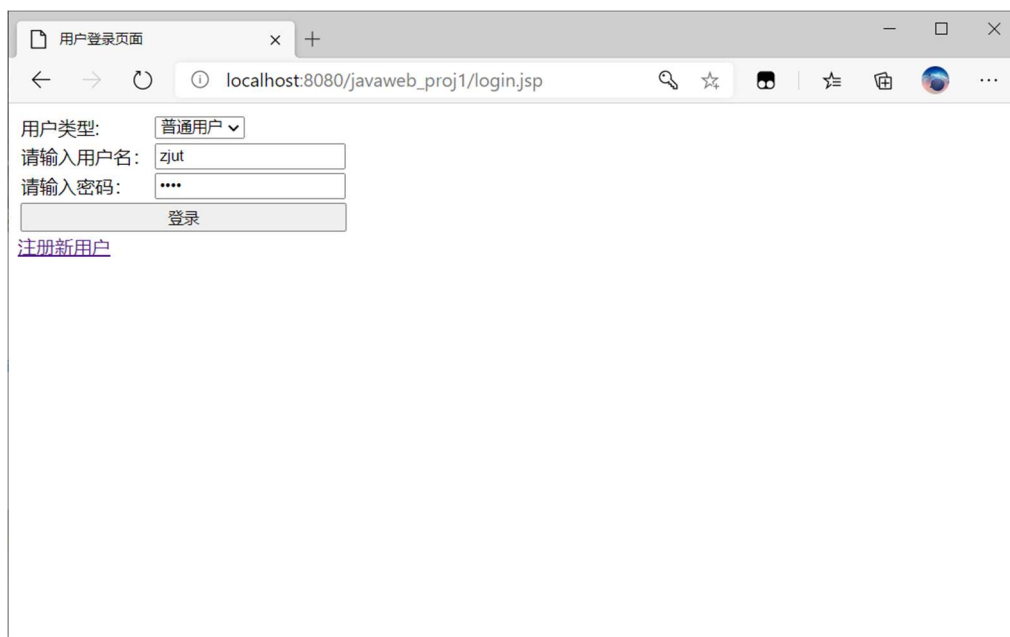


图 12 实验 2 实验结果

关键代码：

```
// 属性声明
private int type;
private String username;
private String password;
```

```
boolean checkUser(UserBean user){
    if(user.getType() == UserBean.ADMIN &&
        "zjut".equals(user.getUsername()) &&
        "zjut".equals(user.getPassword())) {
        return true;
    } else {
        return false;
    }
}
```

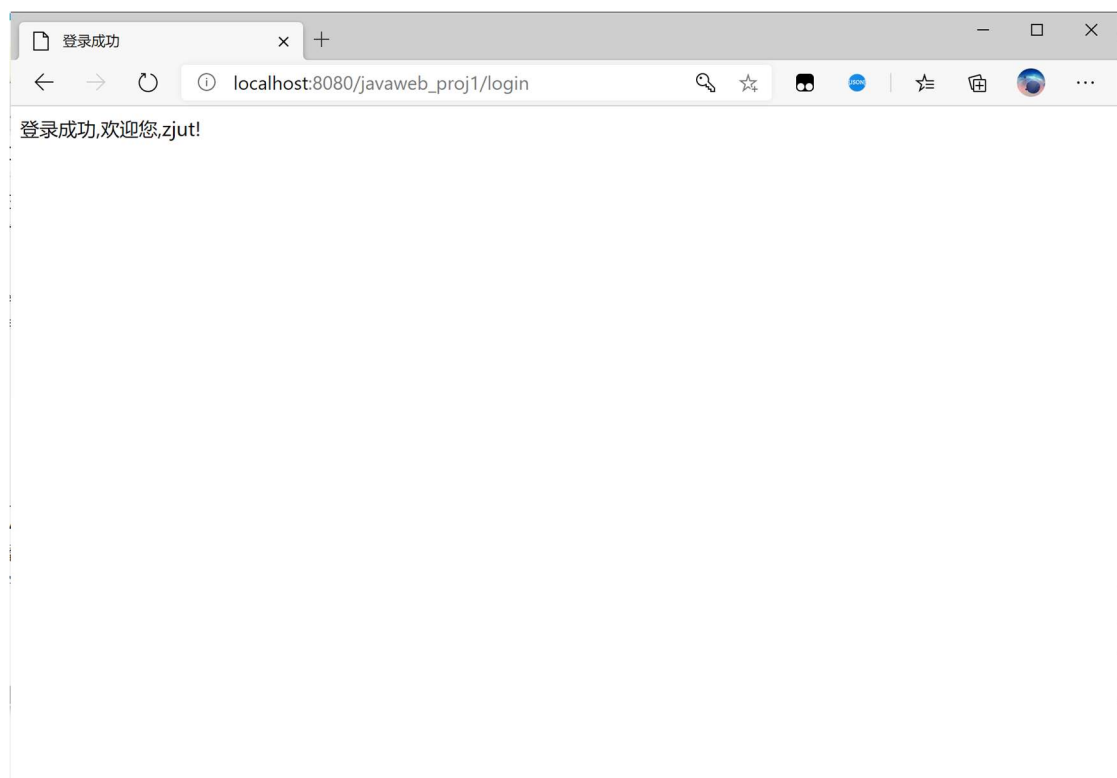


图 13 实验 2 实验结果

4.4、实验三

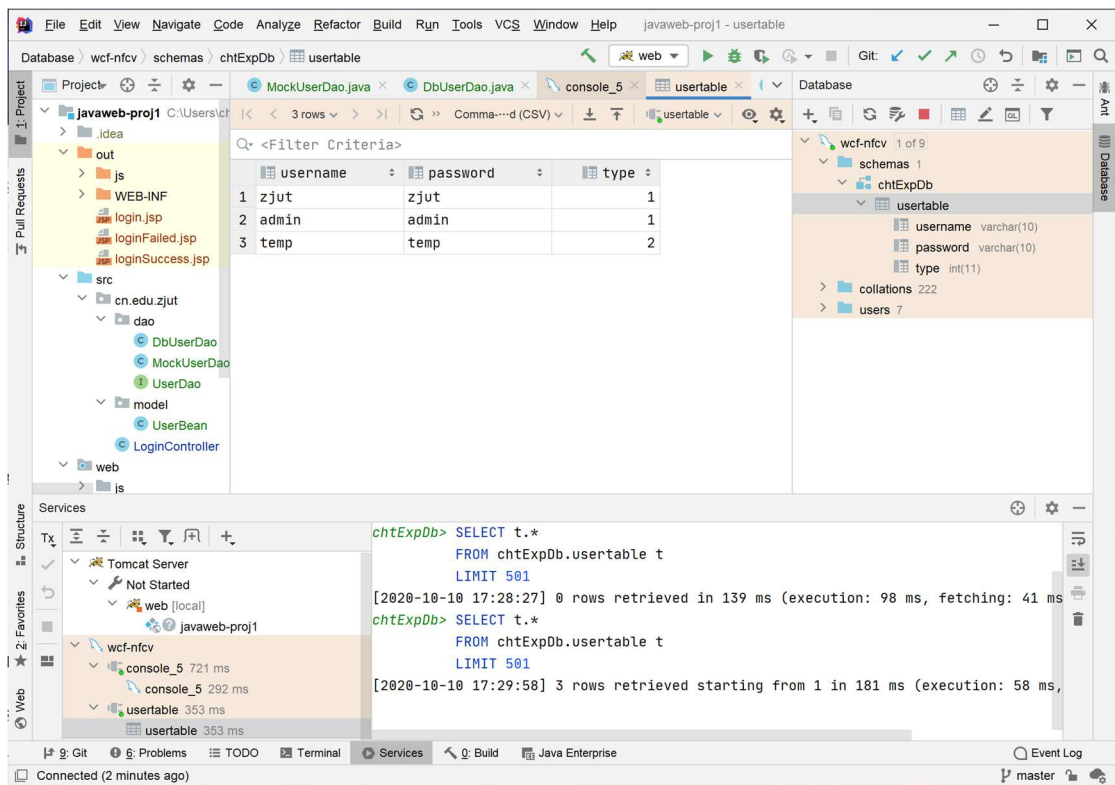


图 14 实验 3 实验结果

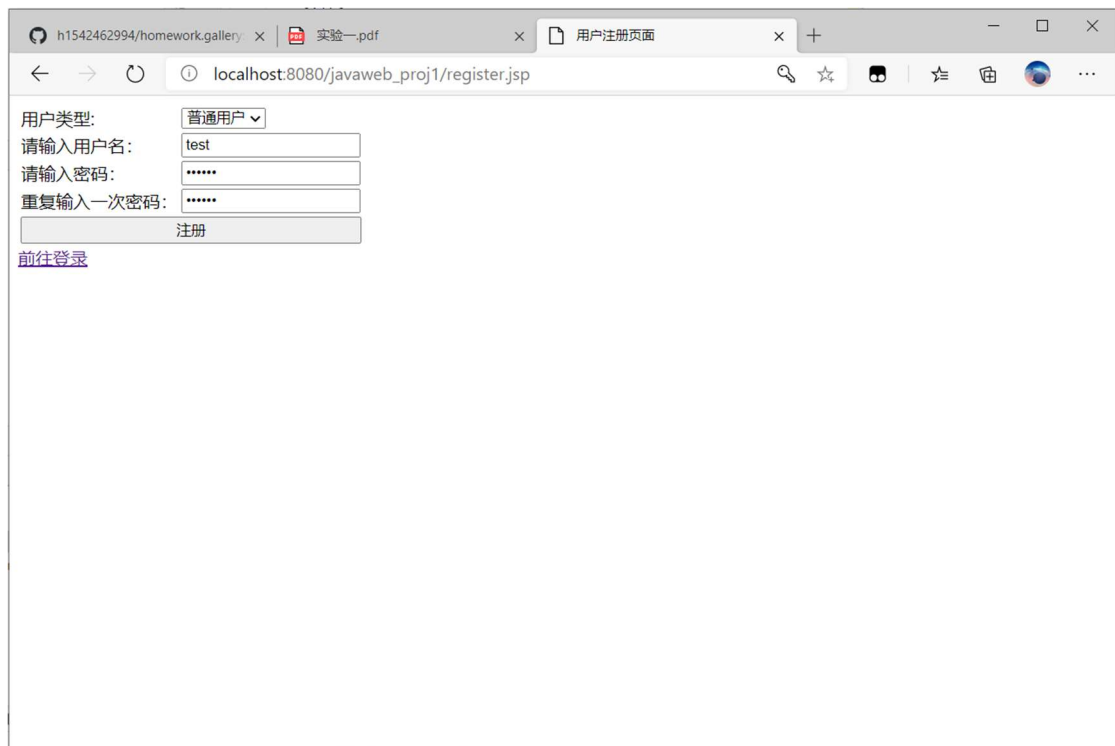


图 15 实验 3 注册部分实验结果

```

6      <servlet>
7          <servlet-name>LoginController</servlet-name>
8          <servlet-class>cn.edu.zjut.LoginController</servlet-class>
9      </servlet>
10     <servlet>
11         <servlet-name>RegisterController</servlet-name>
12         <servlet-class>cn.edu.zjut.RegisterController</servlet-class>
13     </servlet>
14     <servlet-mapping>
15         <servlet-name>LoginController</servlet-name>
16         <url-pattern>/login</url-pattern>
17     </servlet-mapping>
18     <servlet-mapping>
19         <servlet-name>RegisterController</servlet-name>
20         <url-pattern>/register</url-pattern>
21     </servlet-mapping>
22 </web-app>

```

图 16 实验 3 注册部分实验结果

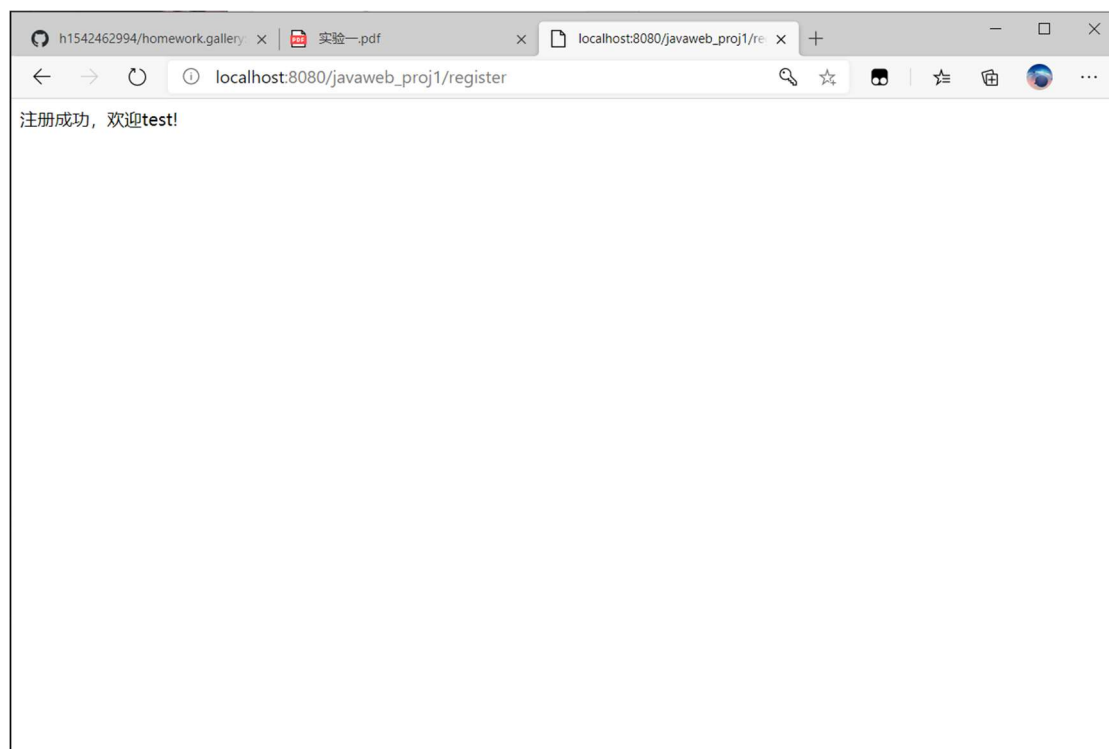
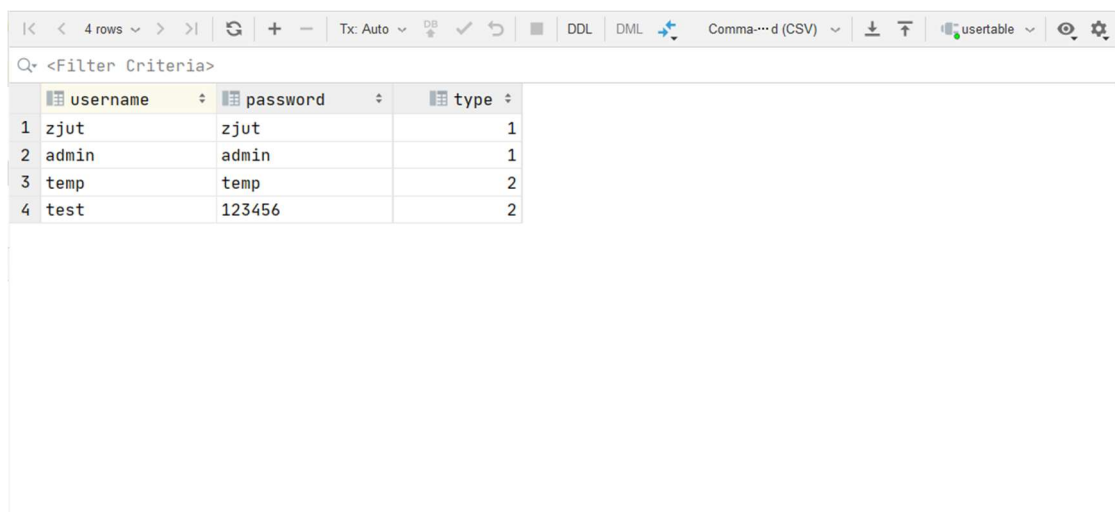


图 17 实验 3 注册部分实验结果



	username	password	type
1	zjut	zjut	1
2	admin	admin	1
3	temp	temp	2
4	test	123456	2

图 18 实验 3 注册部分实验结果

二、实验结果及分析（16 分）

2.1、实验一

这部分的主要部分是开发环境的配置，由于这里用的是 IntelliJ IDEA 2020.2 版本，配置的方式和老版本的 IntelliJ IDEA 和 Eclipse 有不同的地方。但是原理是相通的，需要了解的是 web 文件夹的结构，WEB-INF 中的 classes 和 lib 等的位置。

其他需要注意的问题是非常常见也非常麻烦的乱码问题，首先 html 中很多文件的声明都是可以设置 charset 的，例如引入 js 文件时应当设置 charset，在 mysql 连接字符串中也应当声明 charset，防止出现由于乱码导致的查询错误。

2.2、实验二

MVC 设计模式的优缺点：

一切现代框架的出现和逐步完善都是为了解决工程逐渐复杂导致的模块之间关联度非常大的问题，MVC 设计模式可以“解耦”模块，让各个模块分成相互独立的部分，从而实现逻辑上相互独立，然后功能上相互依赖。或者说，就像零件一样可以“插拔”。当然，框架的考虑和使用也是需要考虑到实际需求的，成熟的框架很可能隐藏了很多的技术细节，导致初学者难以理解，甚至产生过度设计的问题，这些问题都是需要和实际项目结合来理解的。

实验二通过 HttpServlet 充当 Controller, UserBean 充当 Model, jsp 充当 View,

已经有了 MVC 的初步雏形。当然，为了剔除 Model 层中业务逻辑无关的部分，我们常常使用拦截器。

2.3、实验三

Dao 设计模式实际上是一种工厂设计模式，标准的 Dao 设计模式应当由一组预定义的存储逻辑和相关的实现来实现，在之后的 Ioc 中应当会详细介绍这一个设计模式，例如在此项目中 UserDao 表示与 UserBean 相关的存储逻辑，MockUserDao 是一种实现方式（使用 Hashset 做存储对象），DbUserDao 是另一种实现方式。

Dao 设计模式将数据存储相关的逻辑抽象出来，形成一个接口，也是使用到了“解耦”的概念，这种方式有利于扩展和修改应用，如果一种实现不行，只要把接口的实现换成另外一个就好了。

备注：

三个实验的相关代码放在 javaweb-proj1-snapshot 中，存放了三个版本，分别对应了三个实验。