第四五次作业-2020211502-王小龙

1.

问题1：

DispatcherServlet模块：

在整个 Spring MVC 框架中，DispatcherServlet 处于核心位置，它负责协调和组织不同组件完成请求处理并返回响应工作。DispatcherServlet 是 SpringMVC统一的入口，所有的请求都通过它。DispatcherServlet 是前端控制器，配置在web.xml文件中，Servlet依自已定义的具体规则拦截匹配的请求，分发到目标Controller来处理。

HandlerMapping模块：

HandlerMapping是请求映射处理器，通过请求的url找到对应的逻辑处理单元（Controller）

HandlerAdapter模块：

调用具体的方法对用户发来的请求来进行处理。当handlerMapping获取到执行请求的controller时，DispatcherServlte会根据controller对应的controller类型来调用相应的HandlerAdapter来进行处理。

Controller模块：

负责解析用户的请求并将其转换为一个模型

ViewResolver模块：

负责将逻辑视图名解析为具体的视图对象

View模块：

渲染数据模型，返回最终的response给客户端

Model模块：

负责在控制器和展现数据的视图之间传递数据。

Service模块：

是使用一个或多个模型执行操作的方法；封装的具体业务实现方法，来提高业务复用性

Repository模块：

主要是封装数据查询和存储逻辑

问题2：

spring mvc从接收到请求到返回响应的处理过程：

1.用户发起请求到前端控制器。

2.前端控制器通过处理器映射器查找hander。

3.处理器映射器返回执行链。

a)hander对象

b)拦截器（集合）

4.前端控制器通处理器适配器包装，执行hander对象。

5.通过模型hander处理业务逻辑。

6.处理业务完成后，返回ModeAndView对象，其中有视图名称，模型数据。

7.将视图名称和模型数据返回到前端控制器。

8.前端控制器通过视图解释器查找视图对象。

9.视图解释器返回真正的视图。

10.前端控制器通过返回的视图和数据进行渲染。

11.返回渲染完成的视图。

12.将最终的视图返回给用户，产生响应。

问题3：

需要Dispathcher Servlet的原因：

DispatcherServlet是前置控制器，配置在web.xml文件中的。拦截匹配的请求，Servlet拦截匹配规则要自己定义，把拦截下来的请求，依据相应的规则分发到目标Controller来处理，是配置spring MVC的第一步。

DispatcherServlet是前端控制器设计模式的实现，提供Spring Web MVC的集中访问点，而且负责职责的分派，而且与Spring IoC容器无缝集成，从而可以获得Spring的所有好处。

问题4：

@Controller 和 @RestController 二者的区别：

@RestController无法返回指定页面，而@Controller可以；前者可以直接返回数据，后者需要@ResponseBody辅助。

2.

问题1：

cookie和session的区别：

1、数据存储位置：cookie数据存放在客户的浏览器上，session数据放在服务器上。

2、安全性：cookie不是很安全，别人可以分析存放在本地的cookie并进行cookie欺骗，考虑到安全应当使用session。

3、服务器性能：session会在一定时间内保存在服务器上。当访问增多，会比较占用你服务器的性能，考虑到减轻服务器性能方面，应当使用cookie。

4、数据大小：单个cookie保存的数据不能超过4K，很多浏览器都限制一个站点最多保存20个cookie。

5、信息重要程度：可以考虑将登陆信息等重要信息存放为session，其他信息如果需要保留，可以放在cookie中。

问题2：

服务器端如何跟踪到session：

初次访问服务器上的一个jsp，服务器在响应头中设置了临时cookie，并加上了一JSESSIONID，浏览器将存储JSESSIONID的cookie随着请求一起发送到服务器，服务器通过JSESSIONID到内存中找到上次生成的session对象，从而实现客户端(浏览器)共享session。

问题3：

session创建：访问jsp时，服务器通过session = pageContext.getSession();自动生成了session对象

session销毁：

1. 程序调用HttpSession.invalidate()
2. 距离上一次收到客户端发送的session id时间间隔超过了session的最大有效时间
3. 服务器进程被停止

问题4：

关闭浏览器只会使存储在客户端浏览器内存中的session cookie失效，不会使服务器端的session对象失效，同样也不会使已经保存到硬盘上的持久化cookie消失。

问题5：

第一种：将HttpSession作为Spring MVC 的方法参数传入，直接获取

第二种：将HttpServletRequest作为Spring MVC 的方法参数，间接获取

第三种：通过@Autowired HttpServletRequest request 获取

第四种：使用RequestContextHolder类获取request，间接获取到session

第五种： 使用@SessionAttributes

问题6：

缺点：

①当mode="InProc"时，也就是默认设置时，容易丢失数据，为什么?因为网站会因为各种原因重启。

② 当mode="InProc"时，Session保存的东西越多，就越占用服务器内存，对于用户在线人数较多的网站，服务器的内存压力会比较大。

③当mode="InProc"时，程序的扩展性会受到影响，原因很简单：服务器的内存不能在多台服务器间共享。

④虽然Session可以支持扩展性，也就是设置mode="SQLServer"或者mode="StateServer"，但这种方式下，还是有缺点：在每次请求时，也不管你用不用会话数据，都为你准备好，这其实是浪费资源的。

⑤ 如果你没有关闭Session，SessionStateModule就一直在工作中，尤其是全采用默认设置时，会对每个请求执行一系列的调用。浪费资源。

⑥并发问题，前面有解释，也有示例。

⑦当你使用无 Cookie 会话时，为了安全，Session默认会使用 重新生成已过期的会话标识符 的策略，此时，如果通过使用 HTTP POST 方法发起已使用已过期会话 ID 发起的请求，将丢失发送的所有数据。这是因为 ASP.NET 会执行重定向，以确保浏览器在 URL 中具有新的会话标识符。

解决方案：

①如果需要在一个页面的前后调用过程中维持一些简单的数据，可以使用<input type="hidden" />元素来保存这些数据。

②您希望在整个网站都能共享一些会话数据，就像mode="InProc"那样。此时，我们可以使用Cookie与Cache相结合做法，自行控制会话数据的保存与加载。具体做法也简单：为请求分配置一个Key(有就忽略)，然后用这个Key去访问Cache，以完成保存与加载的逻辑。如果要使用的会话数据数量不止一个，可以自定义一个类型或者使用一个诸如Dictionary, HashTable 这样的集合来保存它们。很简单吧，基本上这种方式就是与mode="InProc"差不多了。只是没有锁定问题，因此也就没有并发问题。

③ 如果您想实现mode="StateServer"类似的效果，那么可以考虑使用memcached这类技术，或者自己写个简单的服务，在内部使用一个或者多个Dictionary, HashTable来保存数据即可。这样我们可以更精确的控制读写时机。这种方法也需要使用Cookie保存会话ID。

④如果您想实现mode="SQLServer"类似的效果，那么可以考虑使用mongodb这类技术，同样我们可以更精确的控制读写时机。这种方法也需要使用Cookie保存会话ID。

3.

问题1：

Filter的作用：

- 在执行请求之前执行一段代码
- 是否让客户端访问目标资源
- 调用目标资源以后执行一段代码(通过生命周期函数完成)

SpringMvc—拦截器作用：

拦截器是用来拦截经过dispatcherServlet【请求控制器】的请求。它用来拦截控制器方法的执行。

拦截器通过实现接口HandlerInterceptor并在SpringMvc配置文件中添加配置实现拦截功能。记得为拦截器类加注解把它加到IOC容器中

应用场景：

1、过滤器的应用：字符编码转换，敏感词过滤、登陆权限验证、资源访问权限等；

2、拦截器的应用：AOP、需要有一些业务逻辑（需要注入Bean等）。

问题2：

Spring MVC的拦截器和Servlet中的Filter过滤器的区别：

1、过滤器配置再web.xml中、拦截器配置springmvc的配置文件中（即在DispatcherServlet的contextConfigLocation属性指定文件所在位置，默认加载的是：/WEB-INF/servlet名称-servlet.xml(springmvc-servlet.xml)）；

2、过滤器基于函数回调、拦截器基于反射；

3、过滤器几乎对所有请求起作用，拦截器只对目标执行方法（action方法）起作用；

4、过滤器对请求进行预处理、再交给Servlet处理并且生成响应，最后Filter再对服务器响应进行后处理；

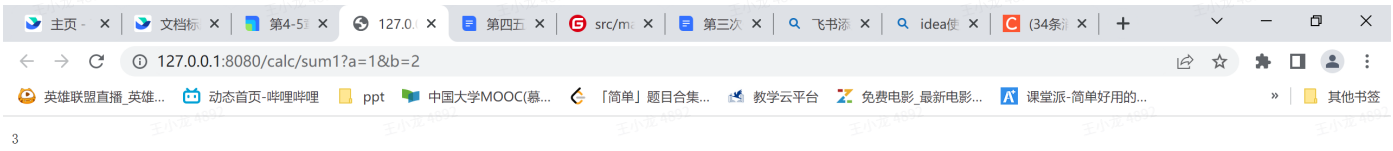
拦截器可以在方法执行前调用（preHandle），方法执行后调用（postHandle），视图页面渲染后调用（afterCompletion）。

4.

gitte仓库链接：https://gitee.com/lhfhlhfl/web_homework4_5.git

a.

i .



ii .


127.0.0.1:8080/calc/sum12/1/2/
英雄联盟直播_英雄...
动态首页-哔哩哔哩
ppt
中国大学MOOC(慕...
「简单」题目合集...
教学云平台
免费电影_最新电影...
课堂派-简单好用的...
其他书签

3

iii.


计算器
127.0.0.1:8080/calc
英雄联盟直播_英雄...
动态首页-哔哩哔哩
ppt
中国大学MOOC(慕...
「简单」题目合集...
教学云平台
免费电影_最新电影...
课堂派-简单好用的...
其他书签
数据1
数据2
提交


主页 -
文档标
第4-5
计算器
第四五
src/m
第三次
飞书添
idea使
(34条
127.0.0.1:8080/calc
英雄联盟直播_英雄...
动态首页-哔哩哔哩
ppt
中国大学MOOC(慕...
「简单」题目合集...
教学云平台
免费电影_最新电影...
课堂派-简单好用的...
其他书签
数据1
数据2
提交

3

b.

第一次：


127.0.0.1:8080/acc?para=10
英雄联盟直播_英雄...
动态首页-哔哩哔哩
ppt
中国大学MOOC(慕...
「简单」题目合集...
教学云平台
免费电影_最新电影...
课堂派-简单好用的...
其他书签

10

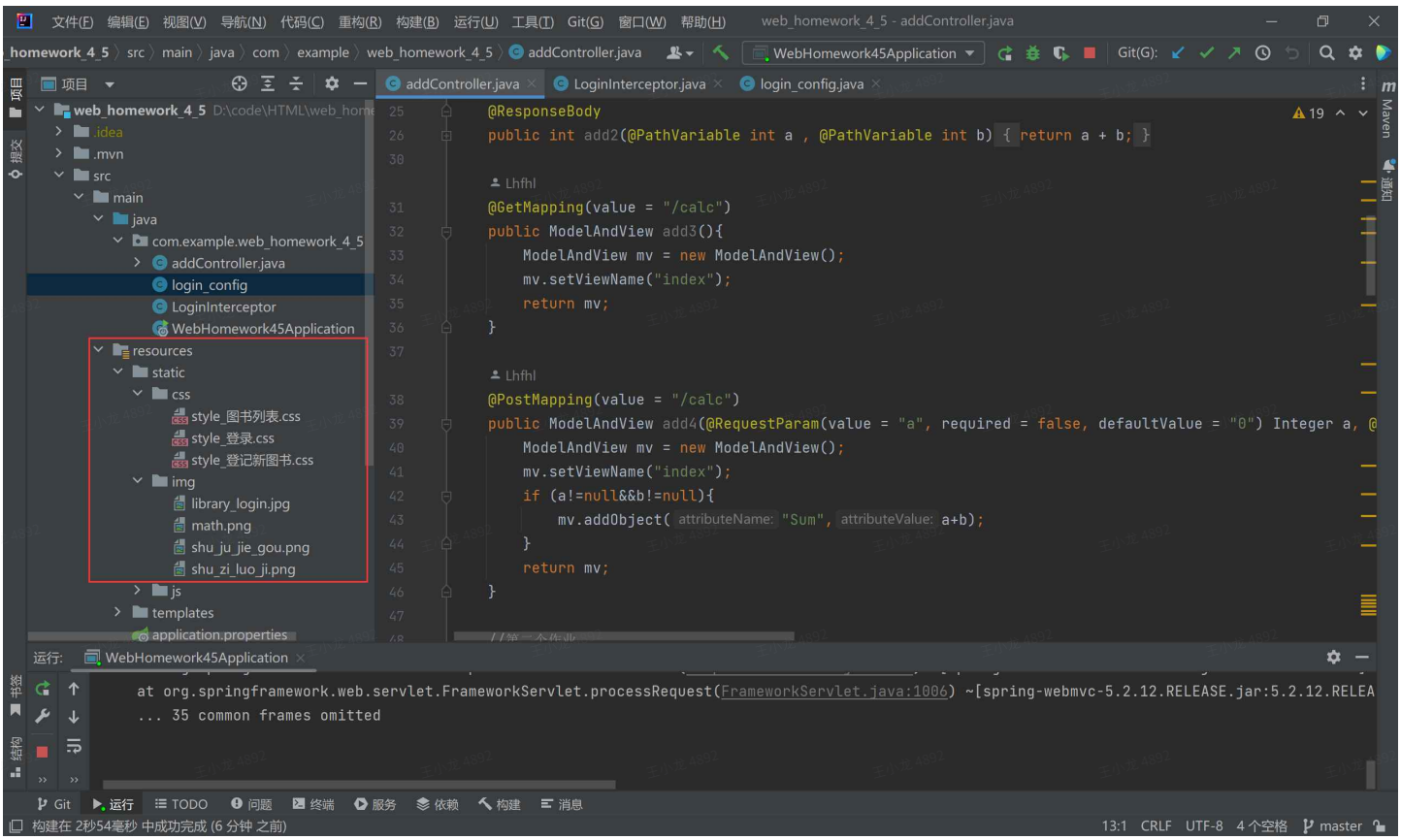
第二次：


127.0.0.1:8080/acc?para=20

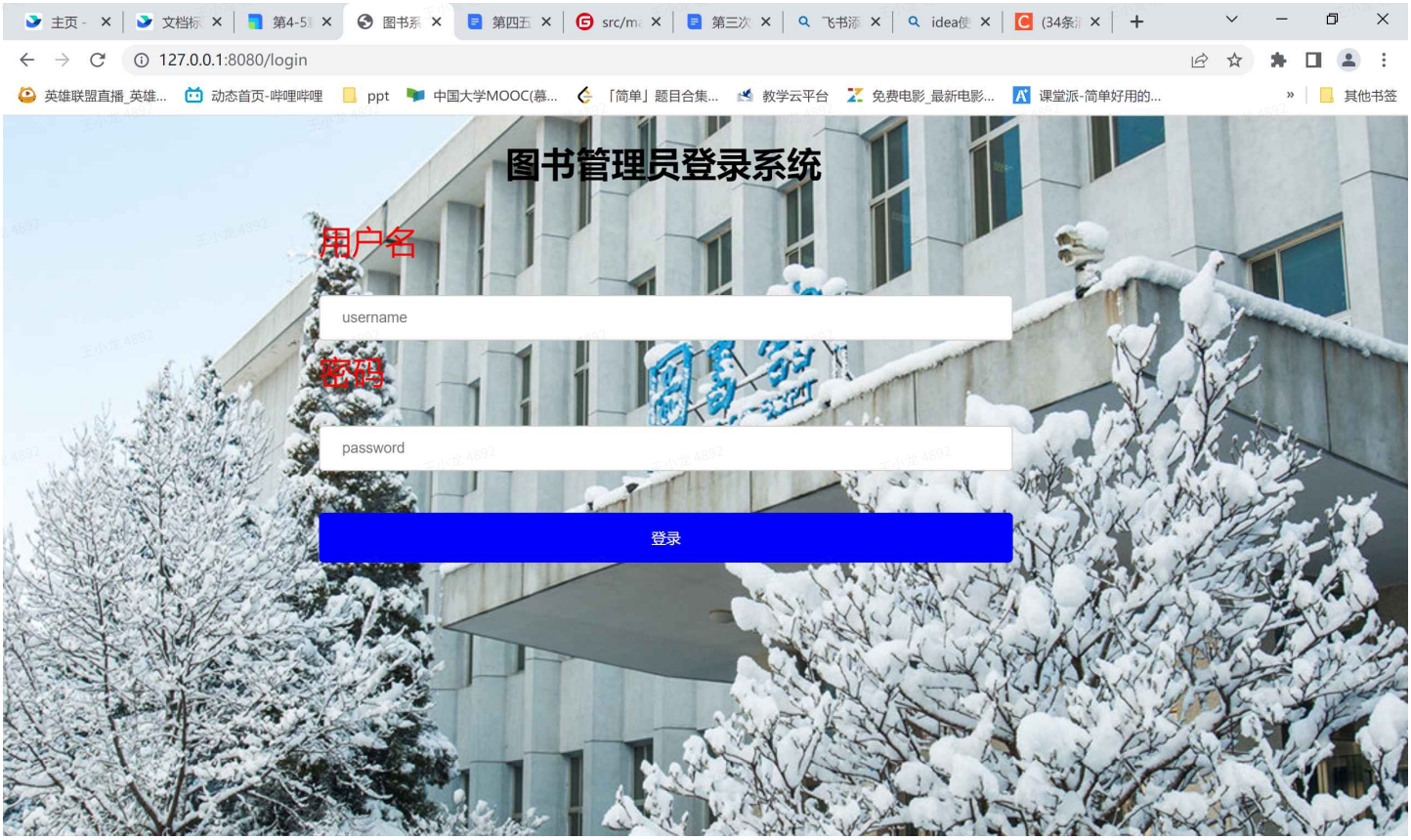
30

c.

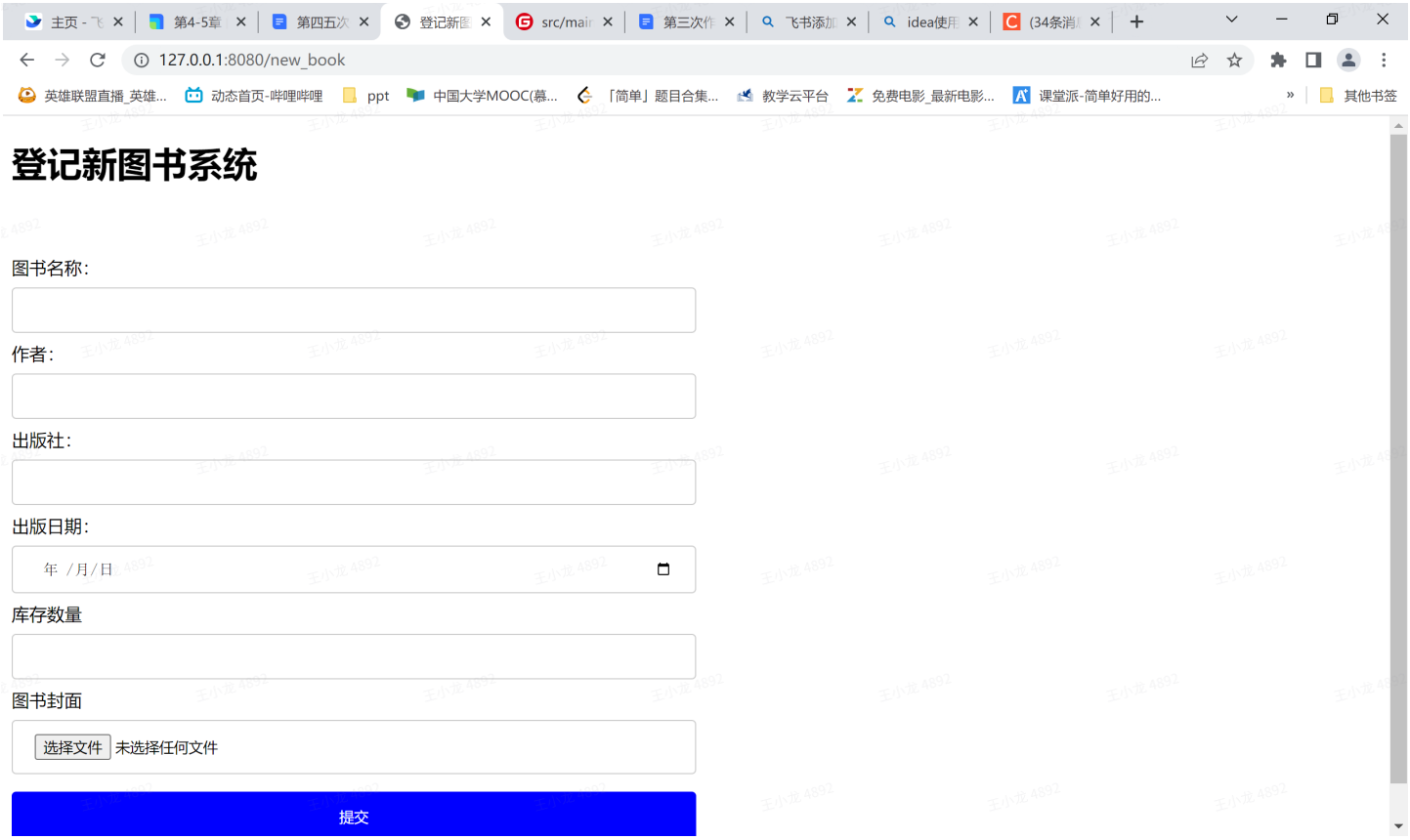
i .


web_homework_4_5 - addController.java
@ResponseBody
public int add2(@PathVariable int a , @PathVariable int b) { return a + b; }
@GetMapping(value = "/calc")
public ModelAndView add3(){
ModelAndView mv = new ModelAndView();
mv.setViewName("index");
return mv;
}
@PostMapping(value = "/calc")
public ModelAndView add4(@RequestParam(value = "a", required = false, defaultValue = "0") Integer a,
ModelAndView mv = new ModelAndView();
mv.setViewName("index");
if (a!=null&&b!=null){
mv.addObject(attributeName: "Sum", attributeValue: a+b);
}
return mv;
}
运行: WebHomework45Application
at org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:1006) ~[spring-webmvc-5.2.12.RELEASE.jar:5.2.12.RELEA
... 35 common frames omitted

ii .

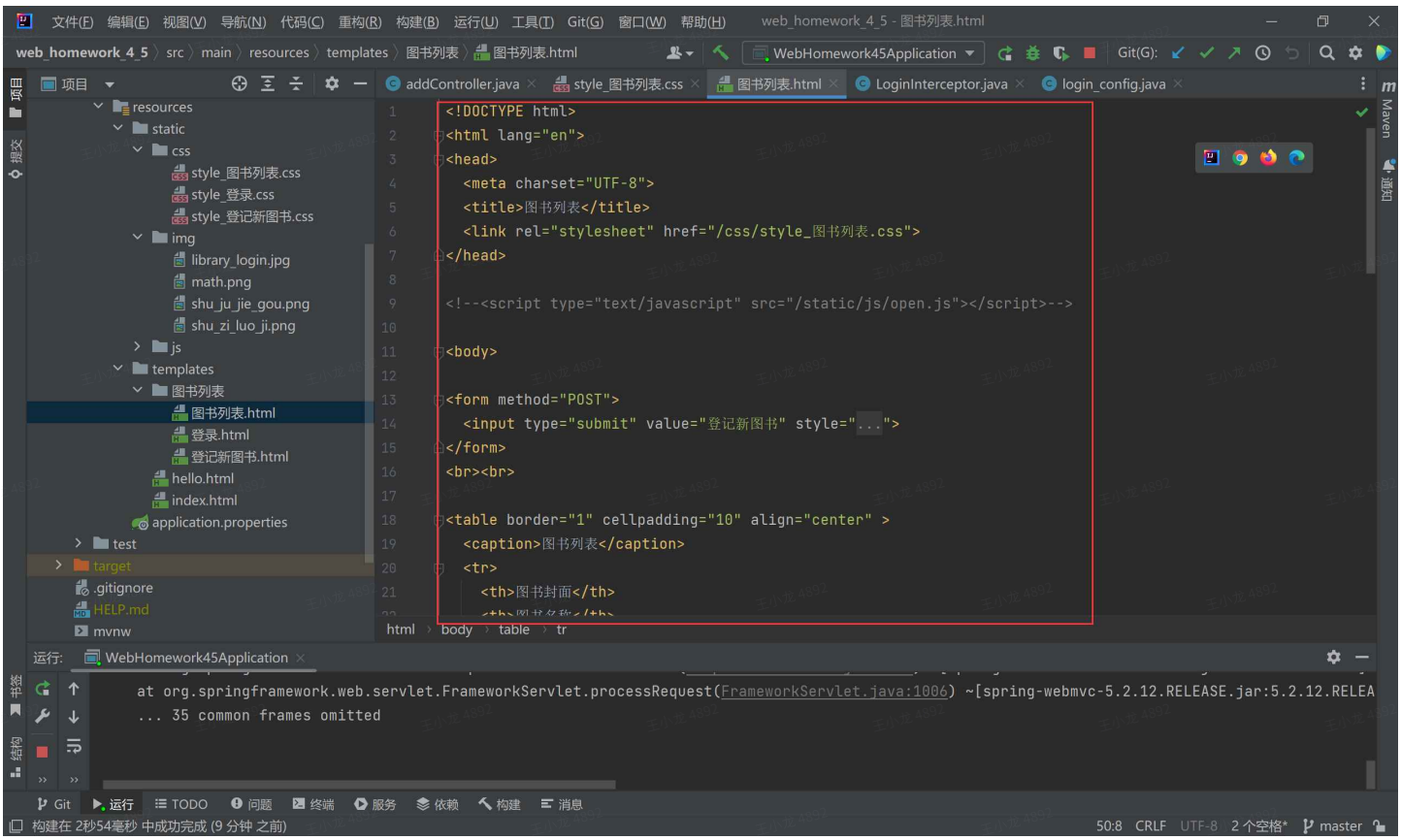


图书封面	图书名称	作者	出版社	出版日期	库存数量(单位/本)
	《高等数学同济第七版》	同济大学数学系	高等教育出版社	上：2014-07-04 下：2014-08-01	100
	《数据结构c语言版》	严蔚敏、吴伟民	清华大学出版社	2012-5	100
	《数字逻辑（第七版·立体化教材）》	白中英、朱正东	科学出版社	2020-12	100

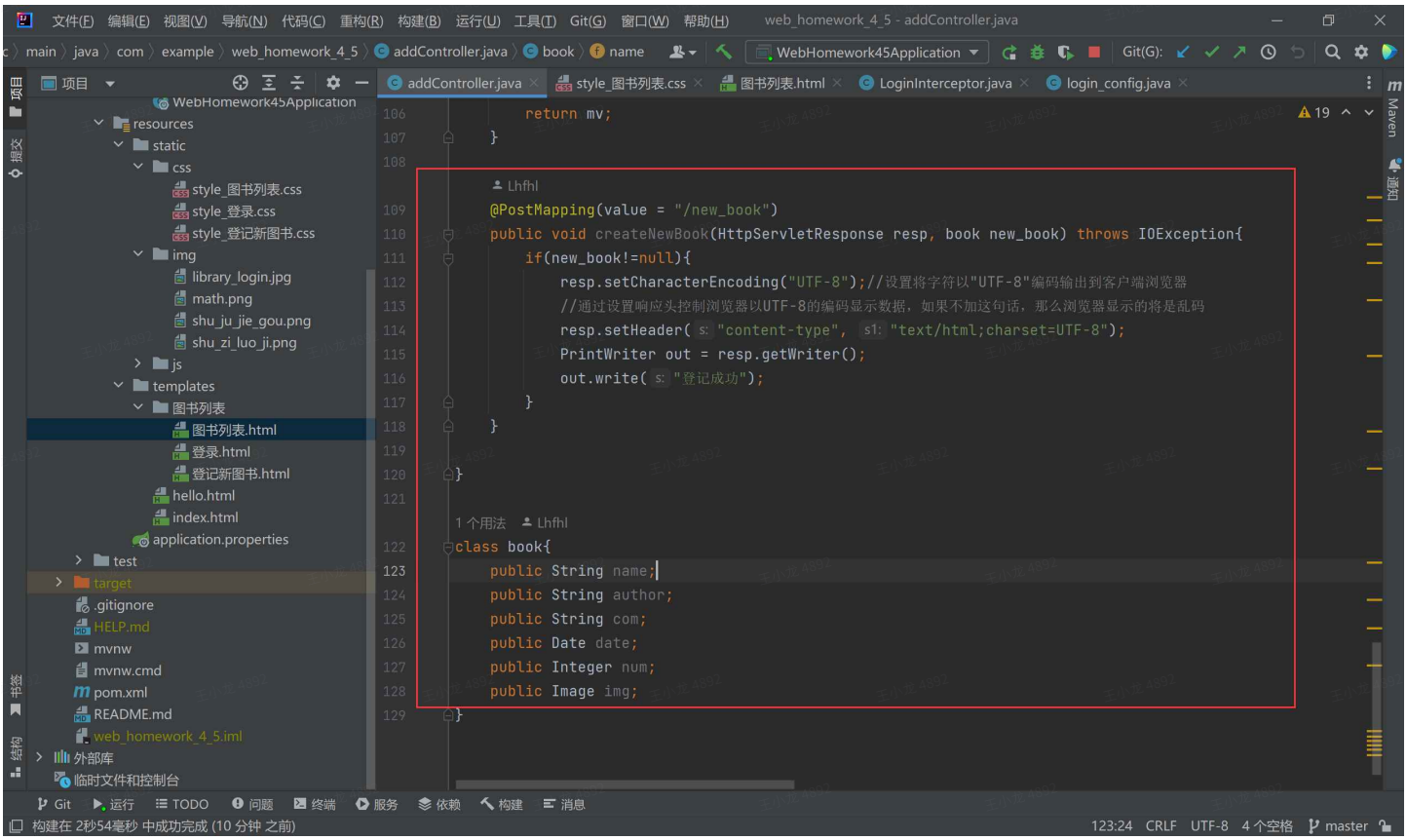


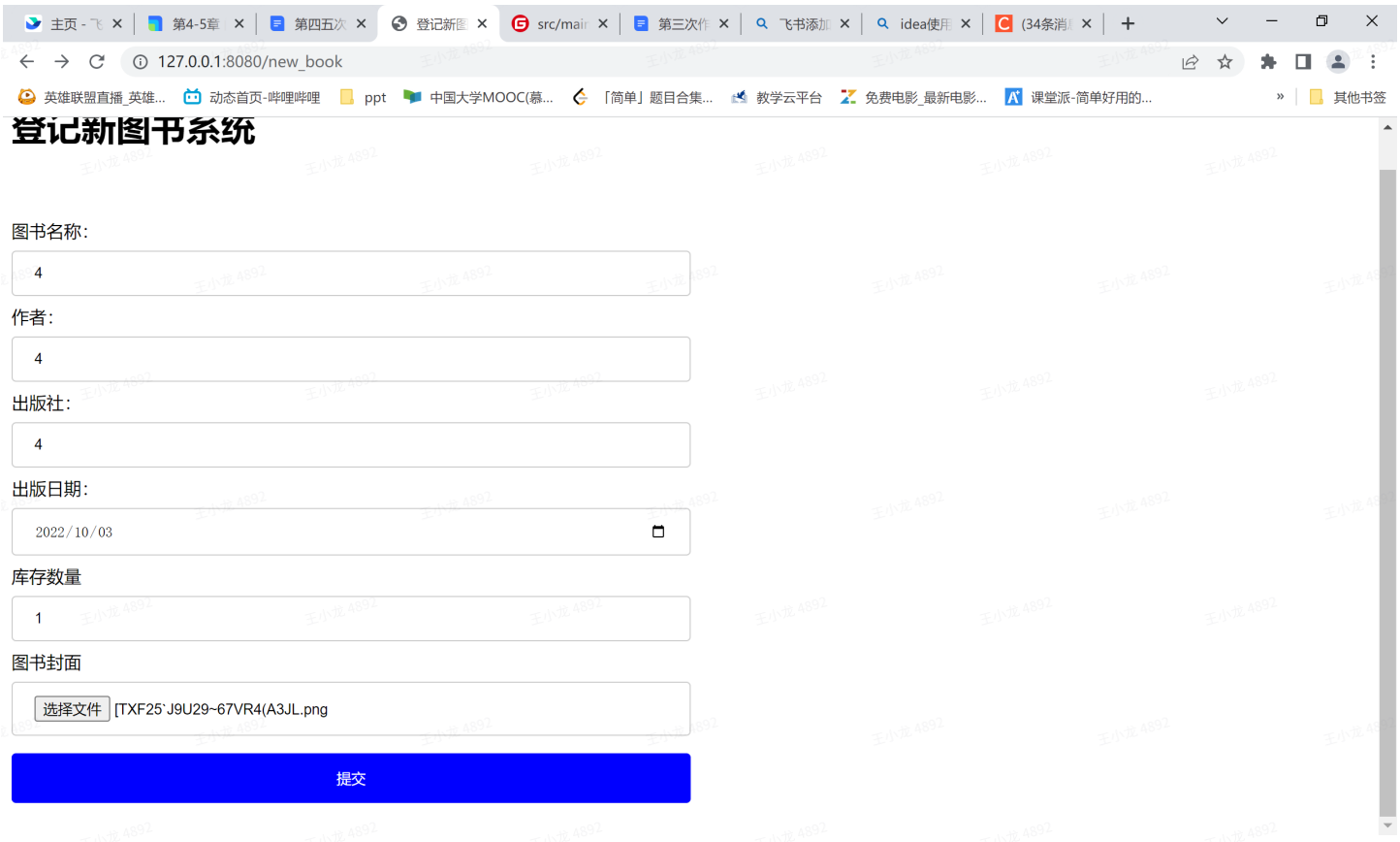
iii.

图书列表静态页面:

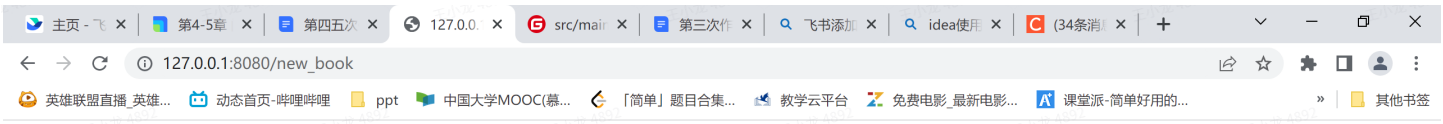


iv.



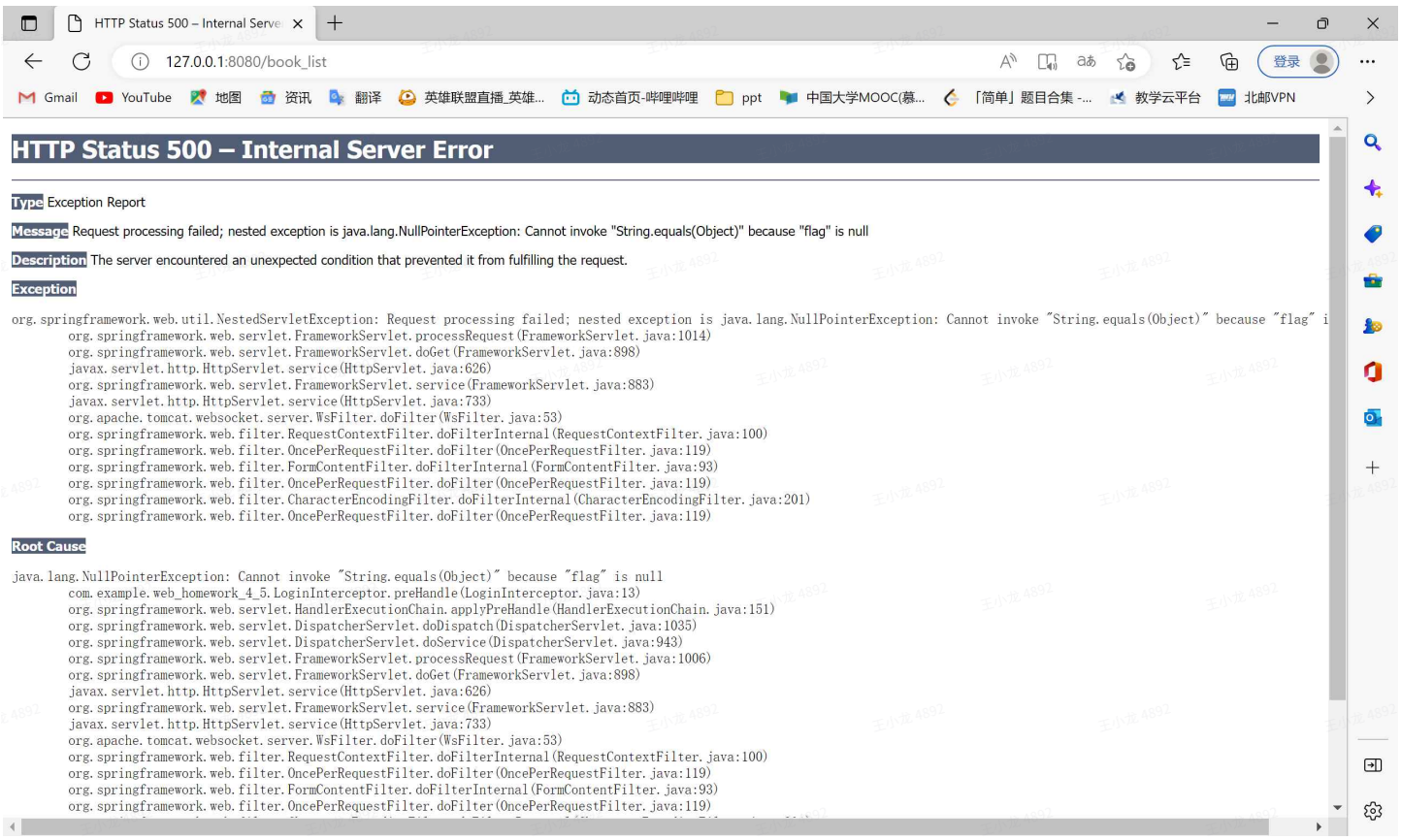


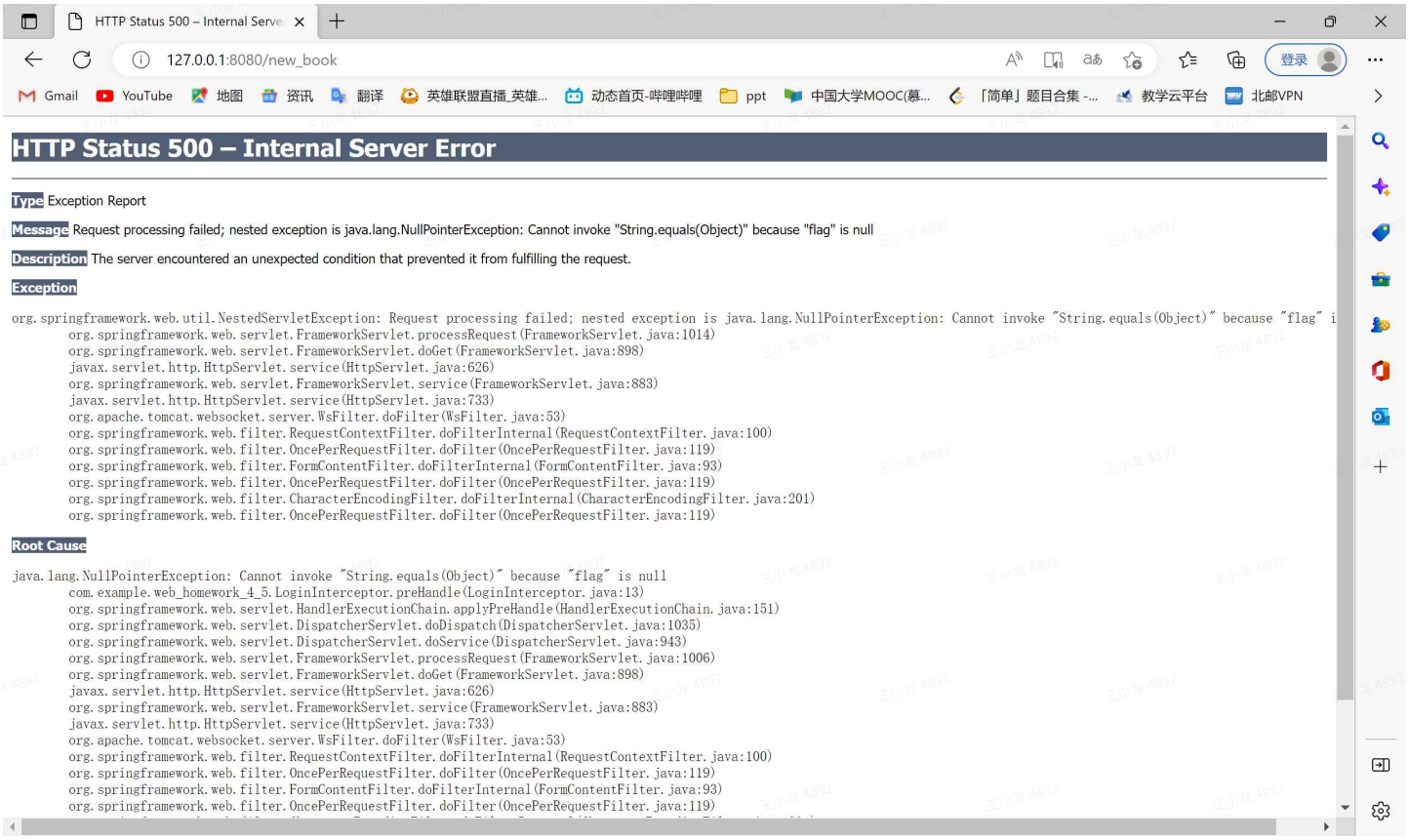
点击提交后，显示登记成功：



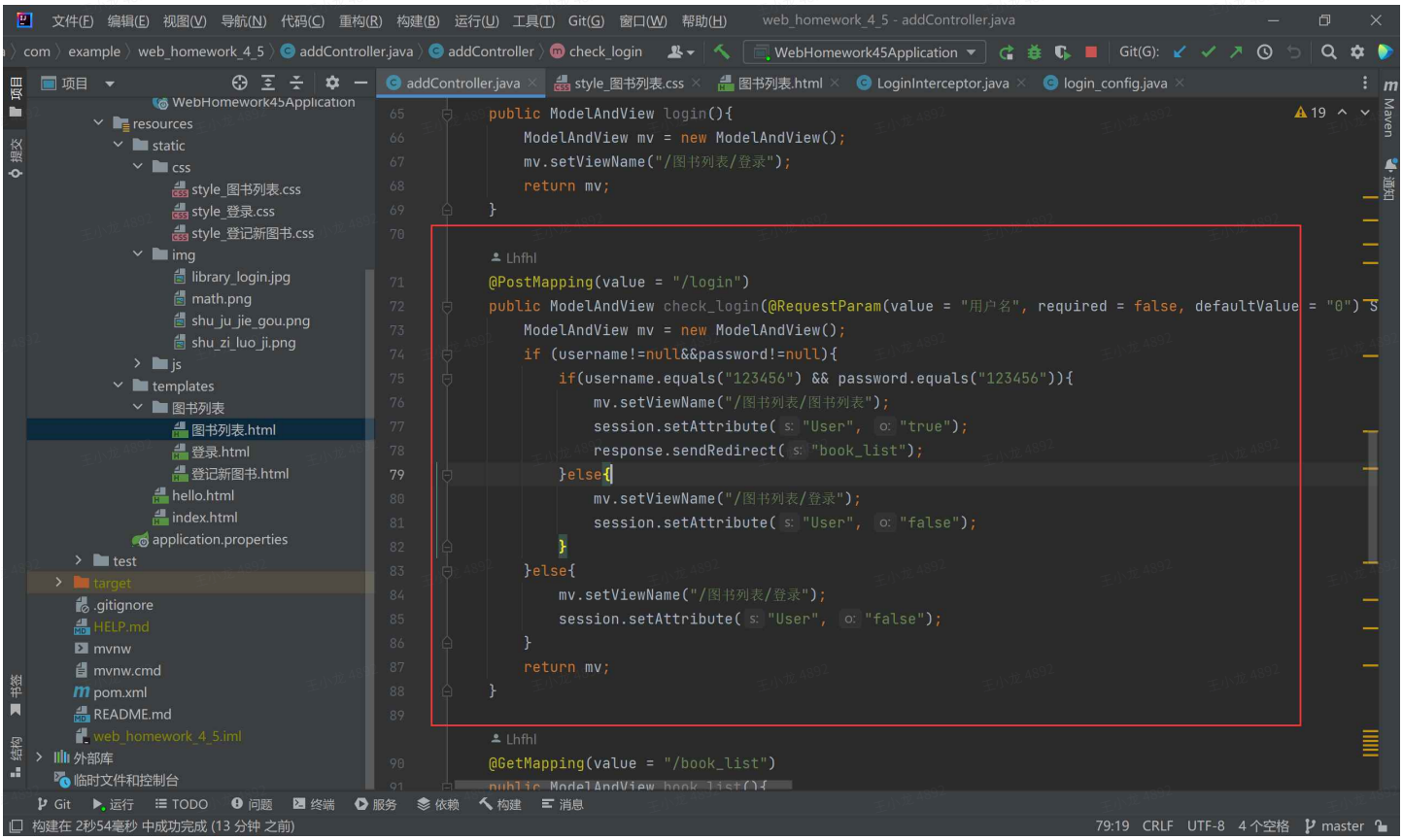
v.

未登录，则无法访问登录后面的页面：




HTTP Status 500 – Internal Serve
127.0.0.1:8080/new_book
登录
Gmail
YouTube
地图
资讯
翻译
英雄联盟直播_英雄...
动态首页-哔哩哔哩
ppt
中国大学MOOC(慕...
「简单」题目合集 -...
教学云平台
北邮VPN
HTTP Status 500 – Internal Server Error
Type Exception Report
Message Request processing failed; nested exception is java.lang.NullPointerException: Cannot invoke "String.equals(Object)" because "flag" is null
Description The server encountered an unexpected condition that prevented it from fulfilling the request.
Exception
org.springframework.web.util.NestedServletException: Request processing failed; nested exception is java.lang.NullPointerException: Cannot invoke "String.equals(Object)" because "flag" i
org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:1014)
org.springframework.web.servlet.FrameworkServlet.doGet(FrameworkServlet.java:898)
javax.servlet.http.HttpServlet.service(HttpServlet.java:626)
org.springframework.web.servlet.FrameworkServlet.service(FrameworkServlet.java:883)
javax.servlet.http.HttpServlet.service(HttpServlet.java:733)
org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:53)
org.springframework.web.filter.RequestContextFilter.doFilterInternal(RequestContextFilter.java:100)
org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:119)
org.springframework.web.filter.FormContentFilter.doFilterInternal(FormContentFilter.java:93)
org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:119)
org.springframework.web.filter.CharacterEncodingFilter.doFilterInternal(CharacterEncodingFilter.java:201)
org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:119)
Root Cause
java.lang.NullPointerException: Cannot invoke "String.equals(Object)" because "flag" is null
com.example.web_homework_4_5.LoginInterceptor.preHandle(LoginInterceptor.java:13)
org.springframework.web.servlet.HandlerExecutionChain.applyPreHandle(HandlerExecutionChain.java:151)
org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:1035)
org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:943)
org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:1006)
org.springframework.web.servlet.FrameworkServlet.doGet(FrameworkServlet.java:898)
javax.servlet.http.HttpServlet.service(HttpServlet.java:626)
org.springframework.web.servlet.FrameworkServlet.service(FrameworkServlet.java:883)
javax.servlet.http.HttpServlet.service(HttpServlet.java:733)
org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:53)
org.springframework.web.filter.RequestContextFilter.doFilterInternal(RequestContextFilter.java:100)
org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:119)
org.springframework.web.filter.FormContentFilter.doFilterInternal(FormContentFilter.java:93)
org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:119)

vi.


web_homework_4_5 - addController.java
WebHomework45Application
addController.java
style_图书列表.css
图书列表.html
LoginInterceptor.java
login_config.java
public ModelAndView login(){
ModelAndView mv = new ModelAndView();
mv.setViewName("/图书列表/登录");
return mv;
}
Lhfhl
@PostMapping(value = "/login")
public ModelAndView check_login(@RequestParam(value = "用户名", required = false, defaultValue = "0") S
ModelAndView mv = new ModelAndView();
if (username!=null&&password!=null){
if(username.equals("123456") && password.equals("123456")){
mv.setViewName("/图书列表/图书列表");
session.setAttribute(s: "User", o: "true");
response.sendRedirect(s: "book_list");
}else{
mv.setViewName("/图书列表/登录");
session.setAttribute(s: "User", o: "false");
}
}else{
mv.setViewName("/图书列表/登录");
session.setAttribute(s: "User", o: "false");
}
return mv;
}
Lhfhl
@GetMapping(value = "/book_list")
resources
static
css
style_图书列表.css
style_登录.css
style_登记新图书.css
img
library_login.jpg
math.png
shu_ju_jie_gou.png
shu_zi_luo_ji.png
js
templates
图书列表
图书列表.html
登录.html
登记新图书.html
hello.html
index.html
application.properties
test
target
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
README.md
web_homework_4_5.iml
外部库
临时文件和控制台
构建在 2秒54毫秒 中成功完成 (13 分钟 之前)
79:19 CRLF UTF-8 4 个空格 master