**软 件 测 试 报 告**

|  |  |
| --- | --- |
| 学院院系 | 计算机学院软件工程系 |
| 专业名称 | 软件工程 |
| 学号姓名 | 04191315何翔 |
| 测试题目 | 后台登入功能模块测试 |
| 指导教师 | 郭晓燕 |
| 完成时间 | 2021年12月28日 |

修订历史记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 日期 | AMD | 修订者 | 说明 |
| V1.0 | 2021.12.7 | A | 何翔 | 完成了单元测试和使用Selenium+TestNG+Excel读取数据并测试。功能比较简单，不足之处是没有使用到数据库数据的读取，以及测试失败后提示不全面，也只是简单的判空和页面跳转，没有做断言的处理，没有报告分析。 |
| V2.0 | 2021.12.21 | A、M | 何翔 | 在V1.0的基础上添加了使用Selenium+TestNG+数据库读取数据并测试。代码基本重构了一遍，好的地方是判断测试用例情况更加全面，且增加了断言处理，有生成代码的测试报告等。不足之处是断言的数据的使用不对，没有增加预期值的字段，反而使用了一些不需要的数据作为是预期的判断。 |
| V2.0+ | 2021.12.28 | A、M | 何翔 | 代码在V2.0的基础上进行全面优化，增加了预期值的字段，实际值能够在程序运行过程中获取，与我们的数据库给定的预期值做断言比较，测试用例更加全面完善，且所有测试都成功通过，功能模块没有bug产生，测试报告等也全面完成，分析准确到位。 |

（A-添加，M-修改，D-删除）

目录

[一、测试需求 1](#_Toc91705989)

[1.1 测试模块 1](#_Toc91705990)

[1.2 测试内容 1](#_Toc91705991)

[二、测试设计思想 1](#_Toc91705992)

[2.1 测试用例 1](#_Toc91705993)

[2.2 等价类划分 2](#_Toc91705994)

[三、测试数据 3](#_Toc91705995)

[四、测试代码（核心部分） 4](#_Toc91705996)

[4.1 TestNG 登入测试 4](#_Toc91705997)

[4.2 Junit 单元测试 8](#_Toc91705998)

[五、数据分析 10](#_Toc91705999)

[5.1 测试运行分析 10](#_Toc91706000)

[5.2 测试数据报告 12](#_Toc91706001)

[六、测试总结 12](#_Toc91706002)

[6.1 技术亮点 12](#_Toc91706003)

[6.1.1 ajax实现异步交互 12](#_Toc91706004)

[6.1.2 使用springboot+mybatis-plus框架搭建项目 14](#_Toc91706005)

[6.1.3 使用springsecurity安全框架 14](#_Toc91706006)

[6.1.4 使用EasyExcel工具读取Excel文件数据 15](#_Toc91706007)

[6.1.5 通过IDEA生成TestNG的测试报告 16](#_Toc91706008)

[6.2 问题分析 16](#_Toc91706009)

[6.3 作业总结 18](#_Toc91706010)

[七、附录 18](#_Toc91706011)

# 一、测试需求

## 1.1 测试模块

**后台登入功能模块**

后台指网站或系统用于管理用户数据、网站或系统数据的一部分，一般只允许管理员或特定人员通过后台登录界面进入，对整个网站及系统进行管理，普通用户是没有权限进入的。后台管理主要是用于对网站前台的信息管理，如文字、图片、影音、和其他日常使用文件的发布、更新、删除等操作，同时也包括各种子模块信息的管理。由此可见，后台管理了一个系统相当多至关重要的数据，操控人员一定是需要安全检验认证才可访问的，进而对数据进行管理，因此，设计一个安全严谨的后台的登入功能模块是非常需要的。现对一个设计好的后台登入模块进行登录的测试。

## 1.2 测试内容

* 使用【**Selenium+Java+数据库**】进行数据驱动测试，对自己搭建的Web项目做**登入功能测试**
* 使用【**Selenium+Java+Excel**】进行数据驱动测试，对自己搭建的Web项目做**登入功能测试**
* 使用【**Junit**】对自己开发的web程序进行**单元测试**，**实现简单的增删查改操作**

# 二、测试设计思想

## 2.1 测试用例

|  |  |
| --- | --- |
| 字段名称 | 描 述 |
| 标识符 | UC1 |
| 测试项 | 登入功能 |
| 设计者 | 何翔 |
| 测试环境要求 | 与服务器可以正常连接 ；软件：Chrome浏览器96版本以上 ，jdk1.8+，maven相关依赖以及TestNG相关jar包 |
| 测试方法 | **黑盒测试** |
| 输入说明 | （1） 访问后台（2）填写登入信息，其中所填写的“用户名”、“密码”两个输入框不能为空，且登入的用户信息需要和注册保存在数据库里面的数据一致（3）点击登入按钮 |
| 输出标准 | 界面提示信息： （1）登入成功时有提示，并能够跳转成功的相关页面（2）当输入的信息不符合要求时，要有具体提示（3）登入失败的时，显示登入失败具体失败的具体原因。 |
| 特殊要求 | 进入到后台登入页面 |
| 用例之间的依赖性 | 无 |

## 2.2 等价类划分

我们可以设用户输入的登入用户名为：input\_username，输入的登入密码为：input\_password；正确对应存在的登入用户名为：username，正确对应存在的登入密码为：password。

一个用户想要登入进后台管理系统，需要满足以下条件：

* 登入用户名输入框已填写数据：
* **input\_username ≠ 空**
* 登入密码输入框已填写数据：
* **input\_password ≠ 空**
* 如果表单信息都填写了，还要判断填写的用户名存在：
* **input\_username = username**
* 如果用户名存在，还要判断填写的密码与存在用户的密码一致：
* **input\_password = password**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 输入条件 | 有效等价类编号 | 有效等价类 | 无效等价类编号 | 无效等价类 |
| 是否填写用户名 | (1) | input\_username ≠ 空 | (2) | input\_username = 空 |
| 是否填写密码 | (3) | input\_password ≠ 空 | (4) | input\_password = 空 |
| 是否存在用户 | (5) | input\_username = username | (6) | input\_username ≠ username |
| 是否密码一致 | (7) | input\_password = password | (8) | input\_password ≠ password |

注：以下的XXX表示的是非正确的随机数据

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 输入值（input\_username/input\_password） | 覆盖等价类编号 | 输出 |
| 1 | （“”，XXX） | （2），（3），（6），（8） |  |
| 2 | （“”，""） | （2），（4），（6），（8） | 请输入用户名 |
| 3 | （“”，password） | （2），（3），（6），（7） |  |
| 4 | （XXX，“ ”） | （1），（4），（6），（8） | 请输入密码 |
| 5 | （username，“ ”） | （1），（4），（5），（8） |  |
| 6 | （XXX，XXX） | （1），（3），（6），（8） | 用户不存在 |
| 7 | （XXX，password） | （1），（3），（6），（7） |  |
| 8 | （username，XXX） | （1），（3），（5），（8） | 密码输入错误 |
| 9 | （username，password） | （1），（3），（5），（7） | 登入成功 |

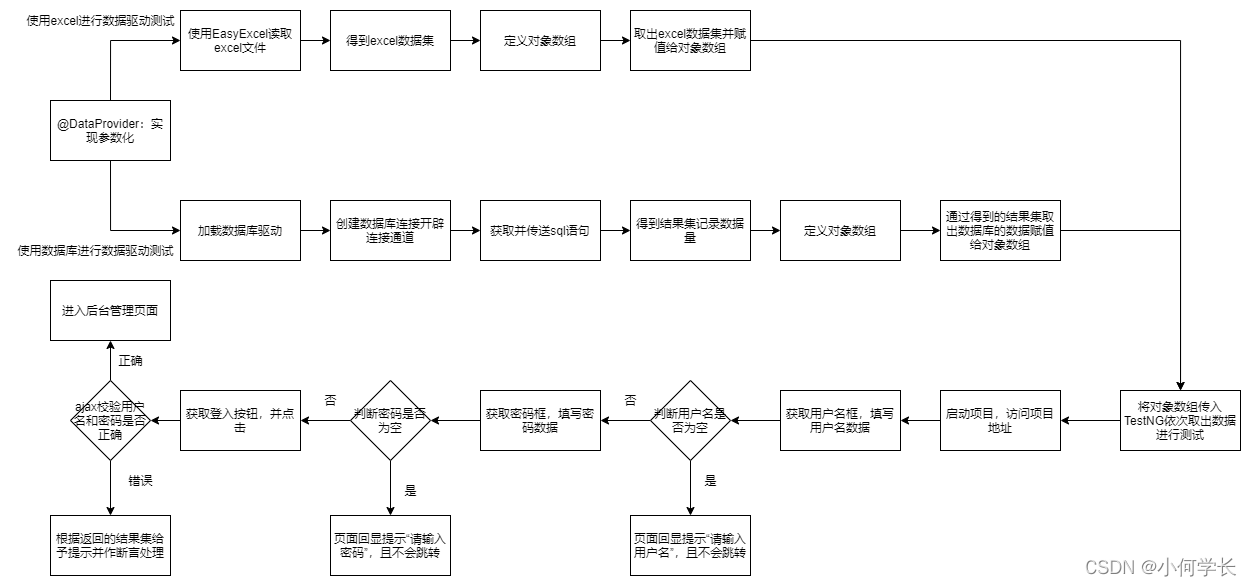
# 三、测试数据

|  |  |  |
| --- | --- | --- |
| 数据库数据 | Excel数据 | 测试用例说明 |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 输入数据 | | 预期值 |
| input\_username | input\_password |
| case1 | 张三 | 123 | 提示“用户不存在” |
| case2 | admin | admin | 提示“密码输入错误 ” |
| case3 | 李四 | error | 提示“用户不存在” |
| case4 |  | 123 | 提示“请输入用户名” |
| case5 | admin |  | 提示“请输入密码” |
| case6 |  |  | 提示“请输入用户名” |
| case7 | 王五 |  | 提示“请输入密码” |
| case8 |  | error | 提示“请输入用户名” |
| case9 | admin | 123 | 登入成功，进入后台主页 |

# 四、测试代码（核心部分）

## 4.1 TestNG 登入测试



**TestNGConfig.java**

package com.study.config;  
/\*  
 \* @ClassName TestNGConfig  
 \* @description: 测试代码相关配置信息和数据文件信息  
 \* @author: 何翔  
 \* @Date 2021/12/27 18:49  
 \*/  
import com.alibaba.excel.EasyExcel;  
  
import java.sql.\*;  
import java.util.List;  
import java.util.Map;  
  
  
public class TestNGConfig {  
  
 static final String url = "jdbc:mysql://localhost/db\_springboot?&useSSL=false&serverTimezone=UTC&rewriteBatchedStatements=true";  
 static final String driver = "com.mysql.cj.jdbc.Driver";  
 static final String sqlUserName ="root";  
 static final String sqlPassword = "1234";  
 static final String ExcelPath = "C:\\StudyProject\\Project\\Java\\SpringBoot\\springboot\\src\\test\\resources" +  
 "\\TestNGData.xlsx";  
 static String username ="admin";  
 static String password ="123";  
 static String expectResult="登录成功";  
   
 public static String getIp(){return "http://localhost:8080/";}  
 public static Object[][] getDatabaseData() {  
 //读取数据库文件信息  
 Object[][] data = null;  
 try {  
 Class.forName(driver);  
 Connection con = DriverManager.getConnection(url,sqlUserName, sqlPassword);  
 Statement s = con.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE, ResultSet.CONCUR\_READ\_ONLY);  
 ResultSet rs = s.executeQuery("select \* from user\_copy ");  
 int total = 0;  
 while (rs.next()) {  
 total++;  
 }  
 data = new Object[total][3];  
 rs.beforeFirst();  
 int a = 0;  
 while (rs.next()) {  
 data[a][0] = rs.getString("user\_name");  
 data[a][1] = rs.getString("user\_pwd");  
 data[a][2] = rs.getString("expect\_result");  
 a++;  
 }  
 } catch (ClassNotFoundException | SQLException e) {  
 e.printStackTrace();  
 }  
 return data;  
 }  
  
 public static Object[][] getExcelData(){  
 //读取Excel文件数据  
 List<Map<Integer, String>> list = EasyExcel  
 .read(ExcelPath)  
 .sheet()  
 .doReadSync();  
 Object[][] data = new Object[list.size()][];  
 int row=0, column;  
 for (Map<Integer, String> map : list) {  
 data[row] = new Object[map.size()];  
 column=0;  
 for (String value : map.values()) {  
 data[row][column] = value;  
 column++;  
 }  
 row++;  
 }  
 return data;  
 }  
   
 public static String getExpectResult() {return expectResult;}  
 public static void setExpectResult(String expectResult) {TestNGConfig.expectResult = expectResult;}  
 public static String getUsername() {return username;}  
 public static void setUsername(String username) {TestNGConfig.username = username;}  
 public static String getPassword() {return password;}  
 public static void setPassword(String password) {TestNGConfig.password = password;}  
  
}

**TestNGWebTest.java**

package com.study.webTest;  
  
/\*  
 \* @ClassName TestNGWebTest  
 \* @description: 使用TestNG对数据库信息或excel文件信息进行登入功能测试  
 \* @author: 何翔  
 \* @Date 2021/11/23 8:47  
 \*/  
import com.study.config.TestNGConfig;  
import org.openqa.selenium.By;  
import org.openqa.selenium.chrome.ChromeDriver;  
import org.testng.annotations.\*;  
  
import static org.testng.AssertJUnit.assertEquals;  
  
  
public class TestNGWebTest{  
  
 static ChromeDriver driver;  
  
 //在当前测试类开始时运行。  
 @BeforeClass  
 public void beforeClass(){  
 System.out.println("-------------------beforeClass");  
 System.setProperty("webdriver.chrome.driver", "src/test/resources/chromedriver.exe");  
 driver = new ChromeDriver();  
 }  
  
 //每个测试方法运行之前运行  
 @BeforeMethod  
 public void before(){  
 System.out.println("=====beforeMethod");  
 }  
  
 @Test(dataProvider="getDatabaseData")  
 public void webTestByUseDatabase(String username , String password, String result){testContent(username, password,result);}  
  
 @DataProvider(name = "getDatabaseData")  
 public Object[][] getDatabaseData() {returnTestNGConfig.getDatabaseData();}  
  
 @Test(dataProvider="getExcelData")  
 public void webTestByUseExcel(String username , String password, String result){testContent(username, password,result);}  
  
 @DataProvider(name = "getExcelData")  
 public Object[][] getExcelData() {  
 return TestNGConfig.getExcelData();  
 }  
  
 //每个测试方法运行之后运行  
 @AfterMethod  
 public void after(){  
 System.out.println("=====afterMethod");  
 try {  
 Thread.sleep(2000);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
  
 //在当前测试类结束时运行。  
 @AfterClass  
 public static void afterClass(){  
 driver.quit();  
 System.out.println("-------------------afterClass");  
 }  
  
 public void testContent(String username , String password , String result){  
 System.out.println("=====testMethod");  
 driver.get(TestNGConfig.getIp());  
 driver.findElement(By.name("username")).sendKeys(username);  
 driver.findElement(By.name("password")).sendKeys(password);  
 driver.findElement(By.id("sign-in-submit")).click();  
 assertEquals( driver.findElement(By.id("expect\_result")).getAttribute("value"), result);  
 }  
  
}

## 4.2 Junit 单元测试



**SpringbootApplicationTests.java**

package com.study;  
  
import com.baomidou.mybatisplus.core.conditions.update.UpdateWrapper;  
import com.baomidou.mybatisplus.core.metadata.IPage;  
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;  
import com.study.springboot.entity.User;  
import com.study.springboot.service.UserService;  
import org.junit.jupiter.api.Test;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.boot.test.context.SpringBootTest;  
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;  
import org.springframework.security.crypto.password.PasswordEncoder;  
  
import java.util.List;  
  
@SpringBootTest  
  
class SpringbootApplicationTests {  
  
/\*  
 \* @author: 何翔  
 \* @date: 2021/10/6 0:56  
 \* @description：security 安全访问测试  
 \*/  
 @Test  
 public void contextLords(){  
 PasswordEncoder pe = new BCryptPasswordEncoder();  
 String encode = pe.encode("123");  
 System.out.println(encode);  
 boolean matches = pe.matches("123",encode);  
 System.out.println(matches);  
 }  
  
/\*  
 \* @author: 何翔  
 \* @date: 2021/10/6 0:55  
 \* @description：mybatis-plus 数据库测试  
 \*/  
  
 @Autowired  
 UserService userService;  
  
 @Test  
 public void query() {  
 //System.out.println(userService.getById(1));  
 System.out.println(userService.list(null));  
 }  
  
 @Test  
 void insert() {  
 User user = new User();  
 user.setUserName("李四");  
 user.setUserPwd("456");  
 System.out.println(userService.save(user));  
 System.out.println(user.getUserId());  
 }  
  
 @Test  
 void delete() {  
 System.out.println(userService.removeById(11));  
 }  
  
 @Test  
 void update() {  
 System.out.println(userService.update(new UpdateWrapper<User>().lambda().set(User::getUserPwd, "223").eq(User::getUserId, 11)));  
 }  
  
 @Test  
 void page() {  
 IPage<User> iPage = new Page<>(1,2);  
 IPage<User> page = userService.page(iPage);  
 List<User> records = page.getRecords();  
 System.out.println(records);  
 System.out.println(page.getPages());  
 }  
  
}

# 五、数据分析

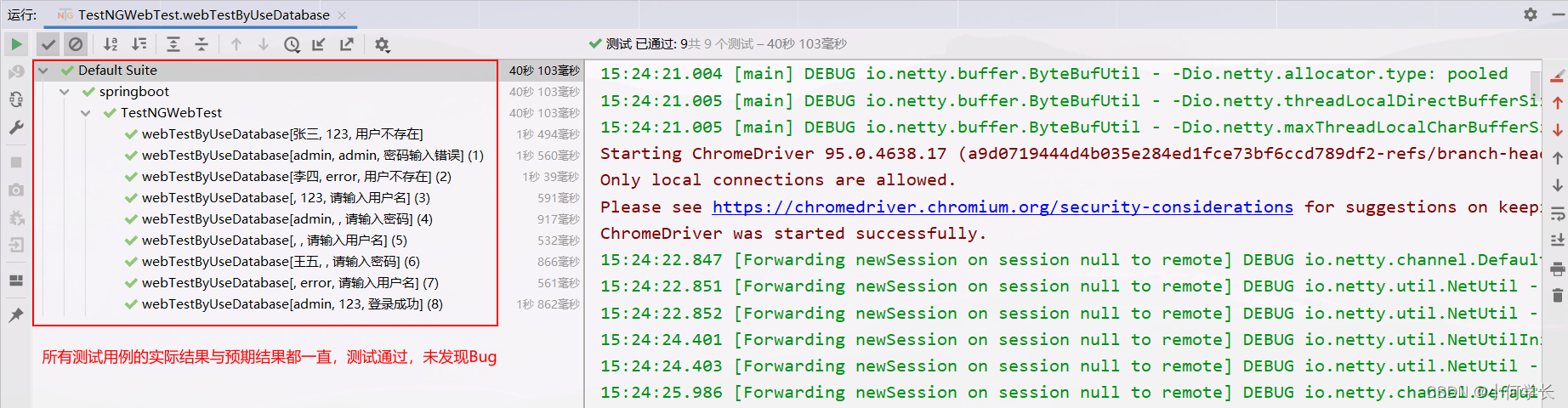
## 5.1 测试运行分析

|  |  |  |
| --- | --- | --- |
|  |  |  |
| case：4，6，8 | case：5，7 | case：6 |

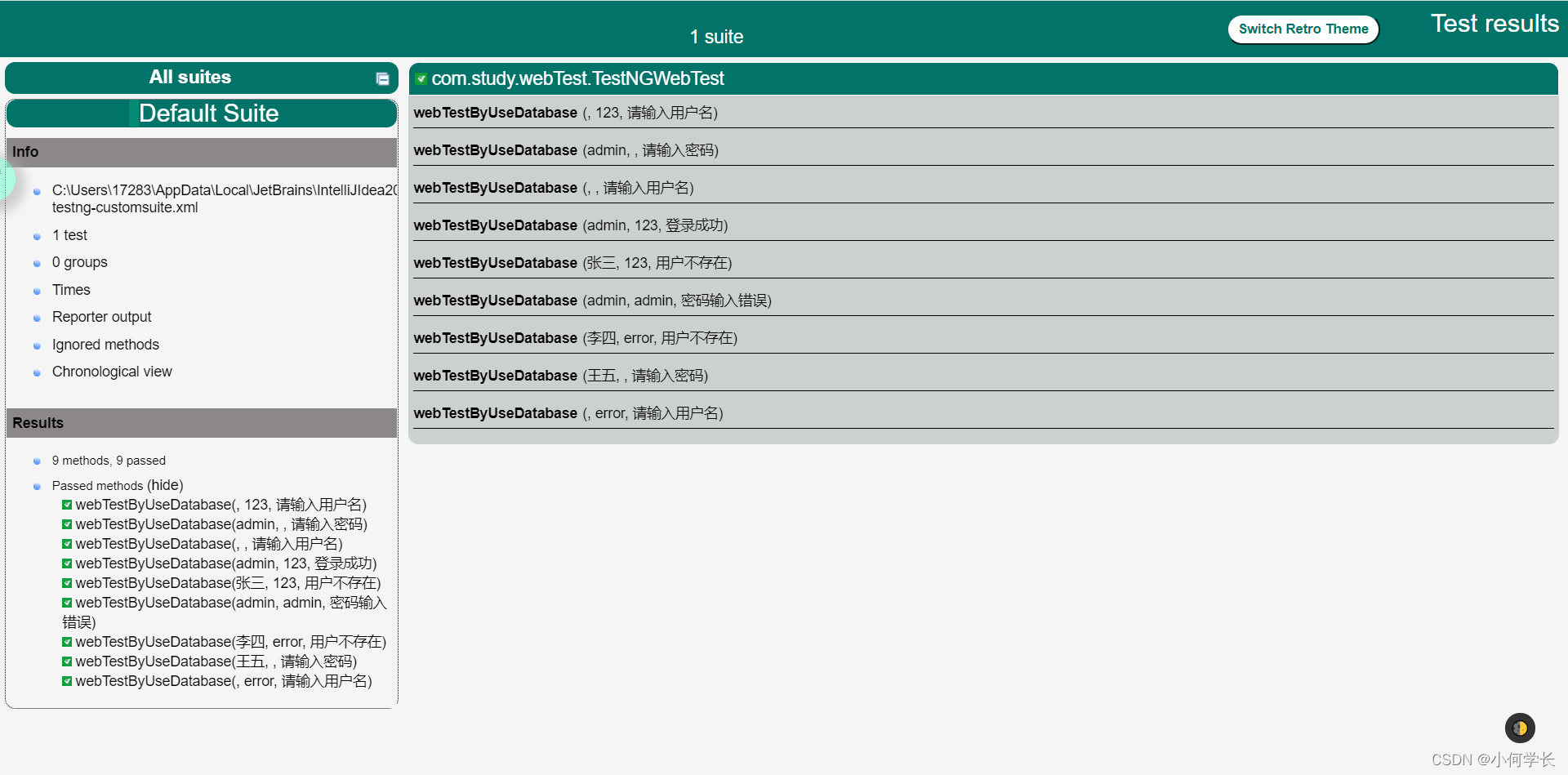
|  |  |  |
| --- | --- | --- |
|  |  |  |
| case：2 | case：1，3 | case：9 |

**登入成功后跳转到相应的后台页面：**



**测试项目的运行如下：**

## 5.2 测试数据报告

**使用TestNG生成测试报告如下：**  


# 六、测试总结

## 6.1 技术亮点

### 6.1.1 ajax实现异步交互

使用ajax，通过在后台与服务器进行少量数据交换，就可以使网页实现异步更新。使得我们可以获得更多的测试用例分析的情况，如判断用户是否存在，存在用户的密码是否正确，细化我们的测试分析情况。

**下面是登入校验的核心代码：**

<script>  
  
 $("#sign-in-submit").click(function () {  
  
 if($("#user").val().length===0){  
 javaex.tip({  
 content : "请输入用户名",  
 type : "error"  
 });  
 $('#expect\_result').val("请输入用户名");  
 return false;  
 }  
  
 if($("#pass").val().length===0){  
 javaex.tip({  
 content : "请输入密码",  
 type : "error"  
 });  
 $('#expect\_result').val("请输入密码");  
 return false;  
 }  
  
 if(!isPermit()){  
 return false;  
 }  
 })  
  
 function isPermit(){  
 var flag = false;  
 $.ajax({  
 type:"post",  
 url : 'springboot/user/login',  
 data:{username:$('#user').val(),password:$('#pass').val()},  
 success: function (res) {  
 $('#expect\_result').val(res["msg"])  
 if(res["type"]==='error'){  
 javaex.tip({  
 mode : "message",  
 content : res["msg"],  
 type : "error"  
 });  
 }  
 else{  
 javaex.tip({  
 mode : "message",  
 content : res["msg"],  
 type : "success"  
 });  
 flag = true;  
 }  
 // 建议延迟加载  
 setTimeout(function() {  
 }, 3000);  
 }  
 });  
 return flag;  
 }  
  
</script>

### 6.1.2 使用springboot+mybatis-plus框架搭建项目

使用springboot框架能够快速搭建项目，对主流的开发框架都提供了⽆配置集成（springboot内置了配置），且项⽬可以独⽴运⾏、⽆需单独配置servlet容器（内置了tomcat），极⼤提⾼了开发、部署效率，此外还提供了运⾏时监控系统（⽇志等）。

使用mybatis-plus能够帮我们逆向生成代码，提高代码编写的效率，同时给我们提供好了CRUD接口，能让我们操作数据库（稍简单的业务）时无需编写sql语句，可以直接调用内置的方法去操作即可。

**项目中的使用（部分示例）：**

|  |  |
| --- | --- |
|  |  |
| springboot工程 | mybatis-plus代码生成器（逆向工程） |

### 6.1.3 使用springsecurity安全框架

SpringSecurity 基于 Spring 框架，提供了一套 Web 应用安全性的完整解决方案，能帮助我们更好地实现登入功能的**认证和授权**。

**项目中的使用（部分示例）：**

|  |  |
| --- | --- |
|  |  |

### 6.1.4 使用EasyExcel工具读取Excel文件数据

EasyExcel是一个基于Java的简单、省内存的读写Excel的开源项目。在尽可能节约内存的情况下支持读写百M的Excel（比POI好）。

github地址：<https://github.com/alibaba/easyexcel>

**项目中的使用：**

public static Object[][] getExcelData(){   
 //读取Excel文件数据   
 List<Map<Integer, String>> list = EasyExcel   
 .read(ExcelPath)   
 .sheet()   
 .doReadSync();   
 Object[][] data = new Object[list.size()][];   
 int row=0, column;   
 for (Map<Integer, String> map : list) {   
 data[row] = new Object[map.size()];   
 column=0;   
 for (String value : map.values()) {   
 data[row][column] = value;   
 column++;   
 }   
 row++;   
 }   
 return data;   
}

### 6.1.5 通过IDEA生成TestNG的测试报告

TestNG 默认自带的有HTML格式的测试报告。这也充分说明拿它来做 UI 自动化测试的优势。

|  |  |
| --- | --- |
|  |  |

## 6.2 问题分析

在读取数据库数据传值给DataProvider的测试过程中，遇到了个别问题，如下分析：

**问题1：**

|  |  |
| --- | --- |
|  |  |
| 使用Mybatis-plus封装的CRUD接口查询数据库 | 测试时报空指针异常错误 |

**问题1分析：**

此处使用了Mybatis-plus框架，并用框架提供的相关方法查询数据库，但是无法将数据查出，报空指针异常，而且框起来那部分代码是没问题的，能在junit单元测试上测试成功，不知道使用了框架后，TestNG是否支持，如果支持是否有哪些地方进行改进。在测试中遇到了此问题，上网查几乎没有相关解答。

因为测试工程是springboot工程，使用TestNG过程中未解决依赖注入等问题，可能在使用时有其他要求或代码添加等问题需要改进。涉及这方面的知识，后续再了解一下，是否真的存在相关问题。

上网查了查一些类似问题：

|  |  |
| --- | --- |
|  |  |
| 类似问题 | 解决方法 |

此问题目前也没有明确问题关键问题与解答，而提供的解决方式就是——将连接操作数据库的代码都直接放在 @DataProvider 中，它就可以正常工作，改后的代码见下面给出的代码示例。



涉及这方面的知识，后续有时间再查阅一下此方面的问题该要如何解决。因此在测试时就改成了用上面能解决数据读取的方法。

除了测试学习过程中遇到的一些问题，测试项目本身也有待完善的地方，但由于期末安排紧张，之后有空自己将会完善改进。

## 6.3 作业总结

通过这次实验，对软件测试有了更进一步的学习了解，在测试中遇到的问题，能够自己寻找解决方案解决，从中也能学习到更多的知识，也能够将学习到的知识运用到实践当中，自己可以自觉学习相关专业知识，相信后续自己也会不断完善相关方面的知识学习，把知识应用的更好。

# 七、附录

文章链接：<https://blog.csdn.net/HXBest/article/details/122083502>

完整代码：<https://github.com/He-Xiang-best/Software-Quality-Assurance-and-Testing>