

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
1.1 测试模块	1
1.2 测试内容	1
二、测试设计思想	1
2.1 测试用例	1
2.2 等价类划分	2
三、测试数据	3
四、测试代码（核心部分）	4
4.1 TestNG 登入测试	4
4.2 Junit 单元测试	8
五、数据分析	10
5.1 测试运行分析	10
5.2 测试数据报告	12
六、测试总结	12
6.1 技术亮点	12
6.1.1 ajax 实现异步交互	12
6.1.2 使用 springboot+mybatis-plus 框架搭建项目	14
6.1.3 使用 springsecurity 安全框架	14
6.1.4 使用 EasyExcel 工具读取 Excel 文件数据	15
6.1.5 通过 IDEA 生成 TestNG 的测试报告	16
6.2 问题分析	16
6.3 作业总结	18
七、附录	18

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)	请输入用户名
2	("", "")	(2), (4), (6), (8)	
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

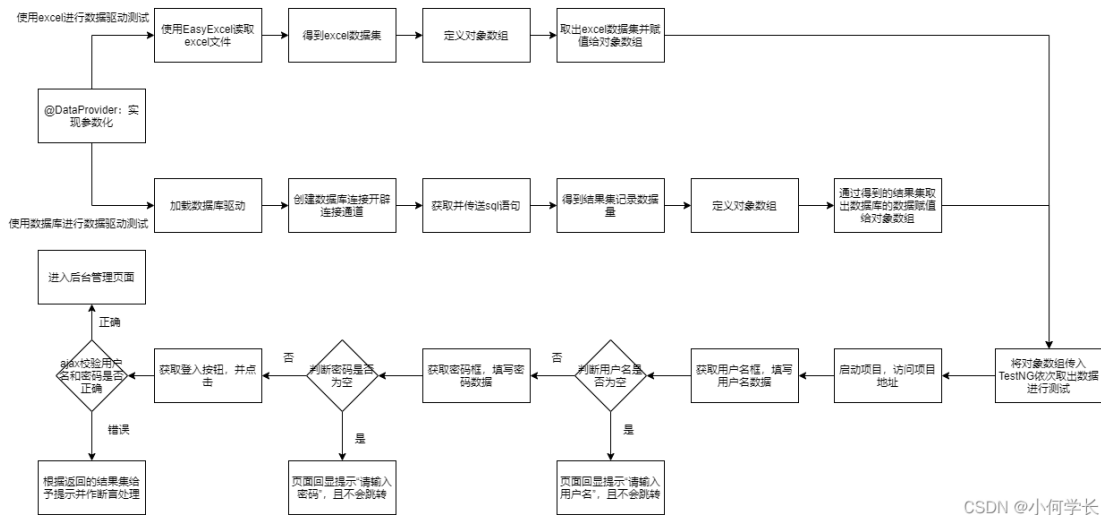
三、测试数据

数据库数据			Excel 数据			测试用例说明	
user_name	user_pwd	expect_result		A	B	C	
张三	123	用户不存在	1	用户名	密码	预期结果	<div>case1: 用户名错误, 密码正确</div> <div>case2: 用户名正确, 密码错误</div> <div>case3: 用户名错误, 密码错误</div> <div>case4: 用户名为空, 密码正确</div> <div>case5: 用户名正确, 密码为空</div> <div>case6: 用户名为空, 密码为空</div> <div>case7: 用户名错误, 密码为空</div> <div>case8: 用户名为空, 密码错误</div> <div>case9: 用户名正确, 密码正确</div>
admin	admin	密码输入错误	2	张三	123	用户不存在	
李四	error	用户不存在	3	admin	admin	密码输入错误	
	123	请输入用户名	4	李四	error	用户不存在	
admin		请输入密码	5		123	请输入用户名	
		请输入密码	6	admin		请输入密码	
		请输入用户名	7			请输入用户名	
王五		请输入密码	8	王五		请输入密码	
	error	请输入用户名	9		error	请输入用户名	
admin	123	登陆成功	10	admin	123	登陆成功	

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



CSDN @小何学长

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) {TestNGConf
ig.expectResult = expectResult;}
    public static String getUsername() {return username;}
    public static void setUsername(String username) {TestNGConfig.usern
ame = username;}
    public static String getPassword() {return password;}
    public static void setPassword(String password) {TestNGConfig.passw
ord = 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() {return TestNGConfig.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) {

```

```

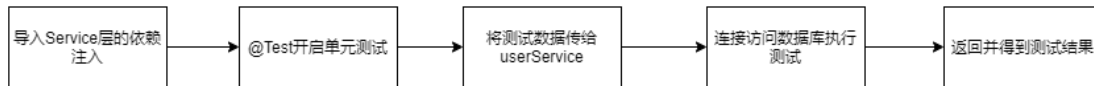
        e.printStackTrace();
    }
}

//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 contextLoads(){

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.getId());

}

@Test

void delete() {

System.out.println(userService.removeById(11));

}

```

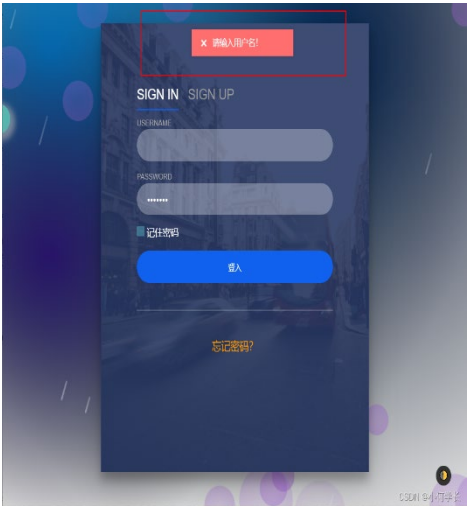
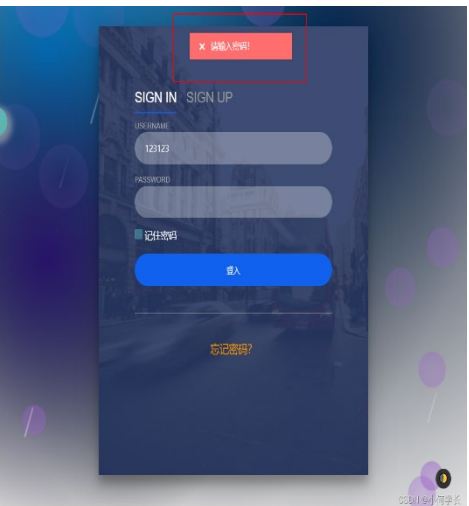
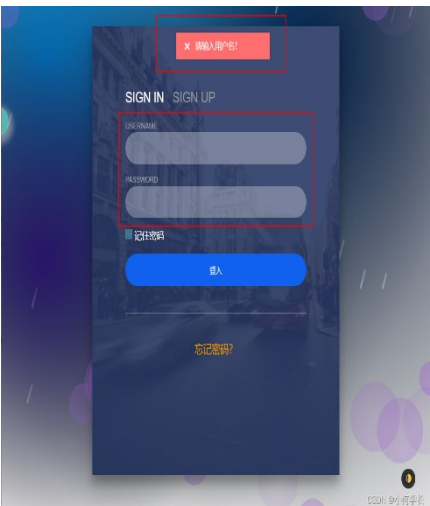
@Test
void update() {
    System.out.println(userService.update(new UpdateWrapper<User>().lambda
bda().set(User::getUserPwd, "223").eq(User::getUserId, 11)));
}

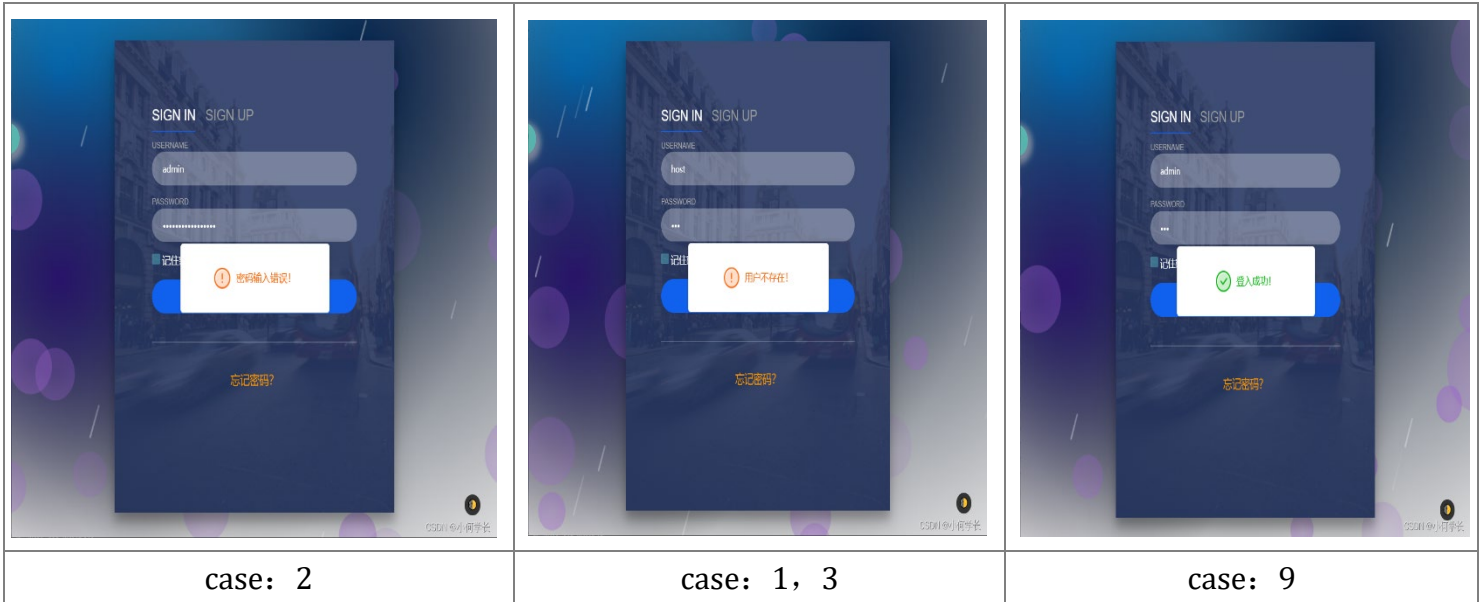
@Test
void page() {
    IPage<User> iPage = new Page<>(1,2);
    IPage<User> page = userService.page(iPage);
    List<User> records = page.getRecords();
    System.out.println(records);
    System.out.println(page.getPages());
}
}

```

五、数据分析

5.1 测试运行分析

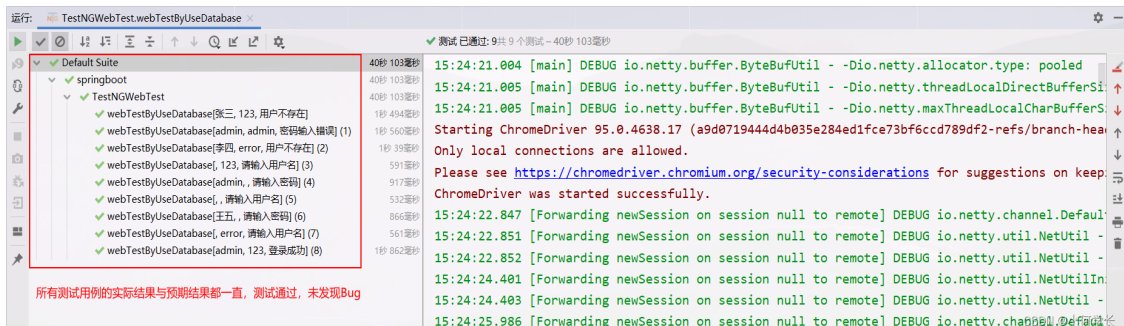
		
case: 4, 6, 8	case: 5, 7	case: 6



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：

The screenshot displays the TestNG Test results report for a suite named '1 suite'. The report is organized into two main sections: 'Info' and 'Results'.

Info Section:

- Location: C:\Users\17283\AppData\Local\JetBrains\IntelliJ\idea2\testing-customsuite.xml
- 1 test
- 0 groups
- Times
- Reporter output
- Ignored methods
- Chronological view

Results Section:

- 9 methods, 9 passed
- Passed methods (hide)
- webTestByUseDatabase(, 123, 请输入用户名)
- webTestByUseDatabase(admin, , 请输入密码)
- webTestByUseDatabase(, , 请输入用户名)
- webTestByUseDatabase(admin, 123, 登录成功)
- webTestByUseDatabase(张三, 123, 用户不存在)
- webTestByUseDatabase(admin, admin, 密码输入错误)
- webTestByUseDatabase(李四, error, 用户不存在)
- webTestByUseDatabase(王五, , 请输入密码)
- webTestByUseDatabase(, error, 请输入用户名)

The 'Test results' tab on the right shows a list of test cases with their status (passed, failed, or error) and the corresponding test data.

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>

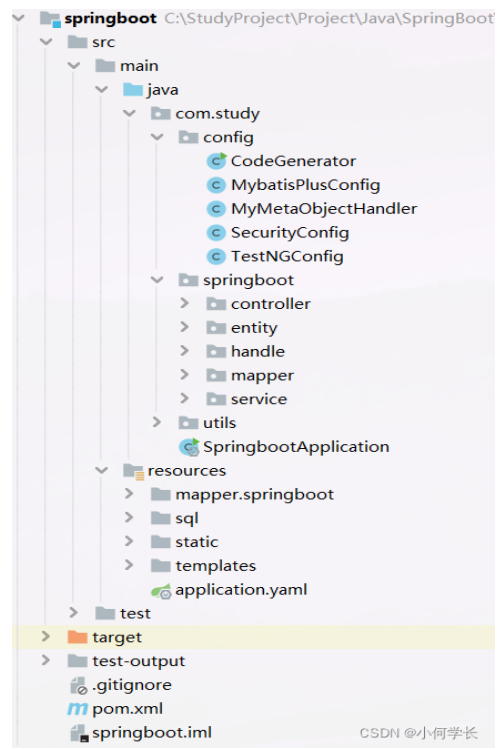
```

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）：

| | |
|---|--|
|  <p>springboot 工程</p> | <pre>public static void main(String[] args) {
 // 代码生成器
 AutoGenerator mpg = new AutoGenerator();
 // 全局配置
 GlobalConfig gc = new GlobalConfig();
 // 获得当前项目的路径
 String projectPath = System.getProperty("user.dir");
 System.out.println(projectPath);
 // 设置生成路径
 gc.setOutputDir(projectPath + "/src/main/java");
 // 作者
 gc.setAuthor("何翔");
 // 代码生成后是否打开文件夹
 gc.setOpen(false);
 // 生成Swagger2注解
 gc.setSwagger2(true);
 // mapper映射文件中自动添加字段映射
 gc.setBaseResultMap(true);
 // 启动生成时覆盖之前的文件
 gc.setFileOverride(true);
 gc.setDateType(DateType.ONLY_DATE);
 // 名字命名要求
 gc.setEntityName("%s");
 gc.setMapperName("%sMapper");
 gc.setXmlName("%sMapper");
 gc.setServiceName("%sService");
 gc.setServiceImplName("%sServiceImpl");
 // 全局配置放入生成器中
 mpg.setGlobalConfig(gc);
}</pre> <p>mybatis-plus 代码生成器（逆向工程）</p> |
|---|--|

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）：

| | |
|--|--|
| <pre> @Override protected void configure(HttpSecurity http) throws Exception { //表单提交 http.formLogin() //当发现以下地址请求时认为是登入, 必须和login.html表单提交的地址一样, 去执行UserDetailsServiceImpl .loginProcessingUrl("/toMain") //自定义登入页面 //也可以用跳转实现 .loginPage("/login.html") //..loginPage("/showLogin") //登入成功跳转页面(通过controller请求跳转) //..successForwardUrl("/toMain") .successHandler(new MyAuthenticationSuccessHandler(url: "/toMain")) //登入失败后跳转页面, post请求 //..failureForwardUrl("/toError404"); .failureHandler(new MyAuthenticationFailureHandler(url: "/404.html")); //授权认证 http.authorizeRequests() // 资源、认证放行(也可以通过跳转实现) .antMatchers(...antPatterns: "/login.html").permitAll() //..antMatchers("/showLogin").permitAll() .antMatchers(...antPatterns: "/springboot/user/login").permitAll() //写法2: //..antMatchers("/login.html").access("permitAll()") .antMatchers(...antPatterns: "/404.html").permitAll() .antMatchers(...antPatterns: "/css/**", "/js/**", "/images/**", "/assets/**", "/javaex/**").permitAll() } </pre> | <pre> @Override public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException { /* * @author: 何翔 * @param: [username] * @return: org.springframework.security.core.userdetails.UserDetails * @date: 2021/10/6 1:16 * @description: 自定义Security访问用户名和密码 */ System.out.println("执行了 loadUserByUsername方法"); //1. 查询数据库, 判断用户名是否存在, 如果不存在就抛出UsernameNotFoundException异常 if(!"admin".equals(username)){ throw new UsernameNotFoundException("用户名不存在!"); } //2. 把查询出来的密码(注册时已经加密过)进行解析, 或者直接把密码放入构造方法里 String password = pw.encode(rawPassword: "123"); return new User(username, password, AuthorityUtils. commaSeparatedStringToAuthorityList(authorityString: "admin,normal,ROLE_myRole"); } </pre> |
|--|--|

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:

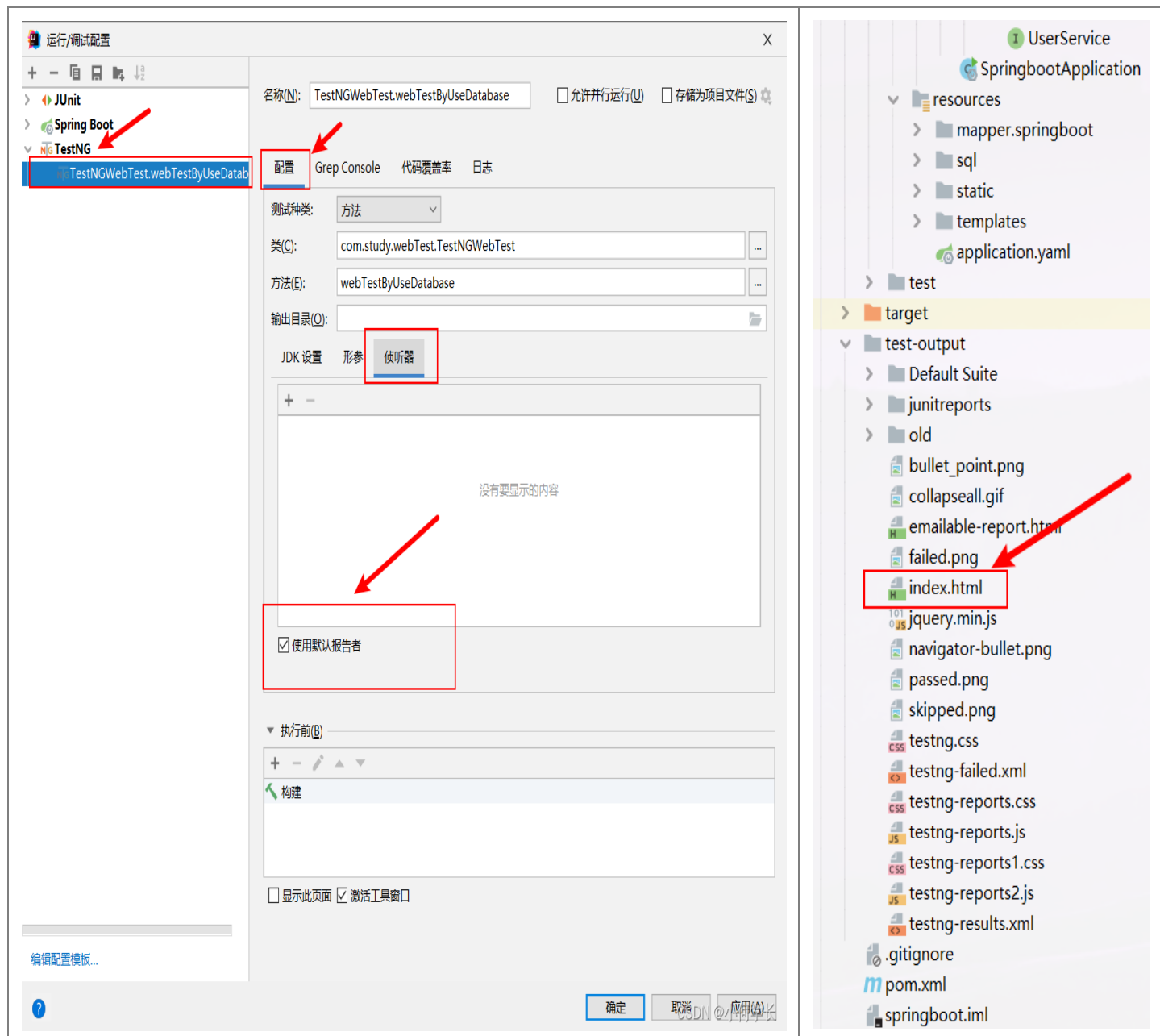
```

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：

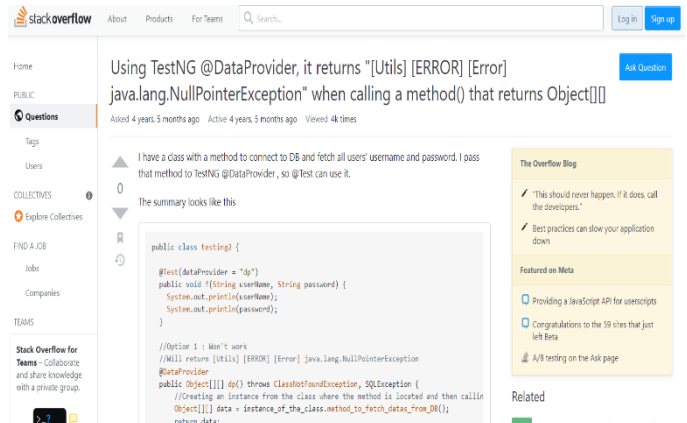
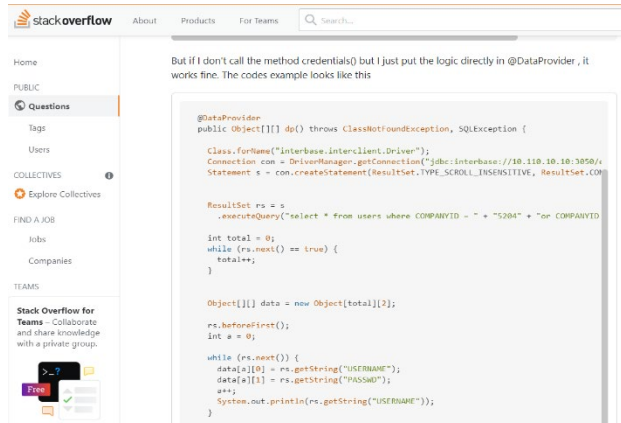
| | |
|---|--|
| <pre> /* * @author: 何翔 * @param: null * @return: Object[][] * @date: 2021/12/23 16:43 * @description: 此方法不可被执行，使用了框架无法得到数据，原因待检查 */ @DataProvider(name = "getDatabaseDataByMybatis") public Object[][] getDatabaseDataByMybatis() { //读取数据库文件信息 List<User> users = userService.list(queryWrapper: null); Object[][] data = new Object[users.size()][2]; int row = 0; for (User user : users) { data[row] = new Object[2]; data[row][0] = user.getUserName(); data[row][1] = user.getUserPwd(); row++; } return data; } </pre> |  |
| 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。

```
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( sql: "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( columnName: "user_name");
            data[a][1] = rs.getString( columnName: "user_pwd");
            data[a][2] = rs.getString( columnName: "expect_result");
            a++;
        }
    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
    }
    return data;
}
```

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>