logo：

徽标, 公司名称

描述已自动生成

### 测

### 试

### 计

### 划

**G02小组**

**二手书交易系统**

目录

[1. 引言 3](#_Toc184585370)

[2. 任务概述 3](#_Toc184585371)

[3. 计划 4](#_Toc184585372)

[4. 测试项目说明 5](#_Toc184585373)

[5. 评价 6](#_Toc184585374)

### 1. 引言

#### 1.1 编写目的

在开发大型软件的漫长过程中，面对极其错综复杂的问题，人的主观认识不可能完全符合客观现实，与工程密切相关的各类人员之间的通信和配合也不可能完美无缺。因此，在软件生命周期的每个阶段都不可避免地会产生差错。尤其对于机票预订系统这类会影响人们生活．财产的工程软件，必须尽量减少差错，以免造成严重的损失。测试是“为了发现程序中的错误而执行程序的过程”。测试的目的就是在软件投入生产性运行之前，尽可能多的发现软件中的错误。目前软件测试仍然是保证软件质量的关键步骤，它是对软件规格说明．设计和编码的最后复审，也是必不可少的关键步骤。

#### 1.2 项目背景

（1）项目名称

（2）任务提出者：杨枨

（3）开发者：G02软件工程小组

（4）用户：学生与老师

（5）实施单位：G02软件工程小组

#### 1.3 参考资料

《软件工程》 张海藩 清华大学出版社

### 2. 任务概述

#### 2.1 目标

* 测试是“为了发现程序中的错误而执行程序的过程”, 测试的目的就是在软件投入生产性运行之前，尽可能多的发现软件中的错误。

#### 2.2 运行环境

* **软件环境**：windows 10系统、MacOS系统
* **开发工具**：**前端框架与开发工具**：HTML，JavaScript，element-ui,Vue

**后端框架与开发工具：**Springboot、MybatisPlus、IDEA

**数据库技术与开发工具：**MySQL、Navicat

* **测试工具**：

**单元测试：Junit 5**

**集成测试：Postman**

**系统测试：JMeter**

#### 2.3 需求概述

该二手书交易网站需要界面简洁美观，功能友好。系统采用了操作简单的页面方便用户通过专业查找书籍，分为用户、卖家、管理员三个方式登陆，方便管理个人信息和交易信息，增强用户体验。

### 3. 计划

#### 3.1 测试方案

测试方案是测试阶段的关键技术问题。为了提高测试效率降低测试成本，本测试方案采用黑盒法设计基本的测试方案，再用白盒法补充一些方案。在黑盒法测试案中，采用等价划分技术，把所有可能的输入数据（有效的和无效的）划分成几等价类，其划分类在以下的输入中再详述。

#### 3.2 测试项目

* 单元测试

单元测试通过对软件中的最小可测试单元进行检查和验证，一般来说，要根据实际情况去判定其具体含义，如Java里单元指一个类，图形化的软件中可以指一个窗口或一个菜单等。

这里的话我们是对后端编写的各个接口类进行测试，查看其各种覆盖率，看其是否能够达到标准。

* 集成测试

集成测试（也叫组装测试，联合测试）是单元测试的逻辑扩展。它最简单的形式是：把两个已经测试过的单元组合成一个组件，测试它们之间的接口。

实践表明，一些模块虽然能够单独地工作，但并不能保证连接起来也能正常的工作。一些局部反映不出来的问题，在全局上很可能暴露出来。

这里的话我们将各个模块进行连接后，进行相应的集成测试。

* 系统测试

系统测试是对整个系统的测试，将硬件、软件、操作人员看作一个整体，检验它是否有不符合系统说明书的地方。这种测试可以发现系统分析和设计中的错误。如安全测试是测试安全措施是否完善，能不能保证系统不受非法侵入。再例如，压力测试是测试系统在正常数据量以及超负荷量(如多个用户同时存取) 等情况下是否还能正常地工作。

* 需求测试

我们在测试活动中，首先需要明确测试需求(What)，才能决定怎么测(How)，测试时间(When)，需要多少人(Who)，测试的环境是什么(Where)，测试中需要的技能、工具以及相应的背景知识，测试中可能遇到的风险等等，以上所有的内容结合起来就构成了测试计划的基本要素。

这里我们主要测试是否满足了用户的需求。

#### 3.3 测试工具

* Postman：前后端接口测试
* Junit5: 后端单元测试以及集成测试
* JMeter：系统性能测试

#### 3.4 测试人员

G02小组全体成员

### 4. 测试项目说明

在测试过程中，首先需要对各子单元过程进行测试。在各子单元过程测试完毕后，再对各模块（包括各子单元过程之间的接口）进行测试，处理好各模块之间的接口，最后对系统进行测试和维护。

#### 4.1 测试过程

本次的测试计划的顺序，是根据软件的开放过程来设计的，测试人员在需求和设计阶段参与需求评审和设计评审、在开发完成前实施测试案例设计和测试开发，在系统开发完成之后正式执行测试

#### 4.2 前端测试用例

包括UI测试、功能测试和交互测试

例：登陆功能测试

// login.test.js

describe('登录功能测试', () => {

test('正常登录测试', async () => {

// 准备测试数据

const loginData = {

username: 'testUser',

password: '123456'

};

// 模拟用户输入

await page.type('#username', loginData.username);

await page.type('#password', loginData.password);

// 点击登录按钮

await page.click('#loginButton');

// 验证结果

expect(await page.url()).toBe('/dashboard');

expect(await page.$('.user-info')).toBeTruthy();

});

test('登录失败测试', async () => {

const loginData = {

username: 'wrongUser',

password: 'wrongPass'

};

await page.type('#username', loginData.username);

await page.type('#password', loginData.password);

await page.click('#loginButton');

// 验证错误提示

const errorMessage = await page.$('.error-message');

expect(errorMessage).toContainText('用户名或密码错误');

});

});

#### 4.3 后端测试用例

1.单元测试用例：

测试用例编号: UT001

测试用例标题: 二手图书服务查询测试

预置条件:

- 数据库中存在测试数据

- 已注入Mock对象

步骤名称: 执行查询方法

步骤描述: 调用ErshoutushuService.queryPage方法

预期输出: 返回包含测试数据的分页对象

实际输出: 返回PageUtils对象,包含预期数据

优先级: 高

编写人: 开发人员

执行结果: Pass

对应代码:



2.集成测试用例：

测试用例编号: IT001

测试用例标题: 二手图书完整流程测试

预置条件:

- 应用已启动

- 数据库已初始化

步骤名称: 执行完整CRUD流程

步骤描述:

1. 添加图书

2. 查询图书

3. 修改图书

4. 删除图书

预期输出: 所有操作返回成功状态码

实际输出: 所有接口返回200状态码

优先级: 高

编写人: 测试人员

执行结果: Pass

对应代码: 

3.系统测试用例：

测试用例编号: ST001

测试用例标题: 用户购买二手图书流程

预置条件:

- 系统完整部署

- 测试账号已创建

- 测试数据已准备

步骤名称: 完整业务流程测试

步骤描述:

1. 用户登录

2. 浏览图书列表

3. 添加购物车

4. 下单支付

5. 确认收货

预期输出:

- 所有步骤正常完成

- 订单状态正确更新

实际输出: 流程完整执行,状态正确

优先级: 高

编写人: 测试人员

执行结果: Pass

对应代码:

java@Testvoid testFullPurchaseFlow() { // 1. 登录测试 String token = userLogin("testUser", "password"); // 2. 浏览图书 mockMvc.perform(get("/ershoutushu/list") .header("token", token)) .andExpect(status().isOk()); // 3. 购物车操作 mockMvc.perform(post("/cart/add") .header("token", token) .content(cartJson)) .andExpect(status().isOk()); // 4. 下单操作 mockMvc.perform(post("/order/create") .header("token", token) .content(orderJson)) .andExpect(status().isOk());}

#### 4.4 测试进度

|  |  |  |  |
| --- | --- | --- | --- |
| 任务名称 | 工期 | 开始时间 | 完成时间 |
| 单元测试 | 2个工作日 | 2024年12月1日 | 2024年12月2日 |
| 集成测试 | 2个工作日 | 2024年12月3日 | 2024年12月4日 |
| 系统测试 | 2个工作日 | 2024年12月5日 | 2024年12月6日 |
| 需求测试 | 2个工作日 | 2024年12月7日 | 2024年12月8日 |

### 5. 评价

#### 5.1 准册

* **可执行软件与需求规格说明书、设计说明书是一致的**
* **测试覆盖率：单元测试：行覆盖率>80%**

**集成测试：接口覆盖率>90%**

**系统测试：功能覆盖率>95%**

**必须覆盖：核心业务逻辑、数据计算处理、异常处理、边界条件**

**可以较低覆盖：简单的getter和setter、日志记录、框架生成的代码**

* **测试用例通过率达到95%**
* **系统页面风格符合规范化要求，程序代码编写以及各种命名符合规范化要求**
* **各模块正常衔接**
* **对异常数据应有相应的提示信息，并能安全终止异常操作**
* **测试优先级：P0（最高优先级）、P1（高优先级）、P2（中优先级）、P3（低优先级）**

#### 5.2 数据处理

测试结果分为通过和未通过。测试达到通过准则的要求称为"通过"，测试结果没有达到测试通过准则的称为"未通过"。