***网上书店管理系统产品***

**测试报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 日 期 | 版 本 | 说明 | 作 者 |
| 2025.4.12 | 1.0 |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**目 录**

[1引言 3](#_Toc35958945)

[1.1编写目的 3](#_Toc35958946)

[1.2背景 3](#_Toc35958947)

[1.3定义 3](#_Toc35958948)

[1.4参考资料 3](#_Toc35958949)

[2测试概要 3](#_Toc35958950)

[2.1动态测试内容和方法 3](#_Toc35958951)

[2.2静态测试内容和方法 4](#_Toc35958954)

[3 测试环境 4](#_Toc35958957)

[4测试总结 5](#_Toc35958958)

[4.1动态测试情况说明 5](#_Toc35958959)

[4.2缺陷情况说明 5](#_Toc35958960)

[4.3静态扫描情况说明 5](#_Toc35958961)

[4.4建议 5](#_Toc35958962)

[4.5评价 6](#_Toc35958963)

# 1引言

## 1.1编写目的

本测试分析报告旨在对智能采购系统和信用风控系统的核心功能进行系统化的测试设计，确保其业务逻辑的正确性、数据一致性和异常处理能力。

具体目标包括：

验证功能完整性：确保自动缺书登记、采购单生成、采购状态跟踪等核心业务流程符合需求，避免重复记录、错误比价或库存不一致问题。验证信用风控系统的五级信用管理逻辑（折扣率、透支规则）在月初自动调整时的准确性。

保障数据一致性：测试采购单生成与库存更新的事务完整性​（如到货后库存增加、缺书记录删除）。确保信用等级变更后，用户折扣和透支权限实时生效。

提高系统健壮性：覆盖边界场景（如库存阈值临界值、信用等级跃迁点）和异常输入（如空供应商列表、非法透支请求）。模拟高并发场景（如月初批量信用评估）下的系统稳定性。

指导开发与测试实施：为开发团队提供明确的单元测试用例​（如JUnit/pytest脚本），聚焦核心算法（供应商比价、信用计算）。为测试团队提供集成测试和系统测试的基准，包括接口模拟（供应商API）和数据库事务验证。

​

预期阅读范围

​开发人员：参考测试用例实现单元测试，确保代码覆盖关键分支（如信用等级阈值判断）。

​测试工程师：执行功能测试与回归测试，验证端到端流程（采购→库存→信用联动）。

​项目经理/产品经理：确认需求落地符合性，评估测试覆盖率和风险点。

​运维团队：了解系统容错设计（如采购单生成失败的回滚机制），辅助部署监控

## 1.2背景

设计一个小型的数据库应用系统，服务于一个网上书店的日常处理业务及网上购书业务，包括供书目录、库存管理、采购管理、客户管理，顾客订单管理、发货管理，供应商管理及网上浏览等。

总体设计要求：书店内部日常处理业务通过C/S模式来实现，而顾客可通过Internet进行网上订书及查询浏览，为改善性能，应尽量将业务规则放在中间层或数据库服务器一端。

任务提出者：软件质量与测试老师

开发者：穆文轩，李俊辉

用户：穆文轩，李俊辉

## 1.3定义

无

## 1.4参考资料

# 2测试概要

## 2.1动态测试内容和方法

### 2.1.1 测试内容

书籍搜索接口 (test\_search\_books)

测试GET请求返回状态码200

验证响应内容包含关键词"数据库系统"

书籍详情接口 (test\_book\_details)

测试GET请求解析JSON数据

验证字段title值与预期一致

订单创建接口 (test\_create\_order)

测试POST请求返回状态码201（创建成功）

隐式验证数据库记录数更新（通过@pytest.mark.django\_db）

### 2.1.2 测试方法

|  |  |  |
| --- | --- | --- |
| **​**方法类型**** | **​**技术实现**** | **​**示例**** |
| ​****接口测试**** | 使用APIClient模拟HTTP请求 | client.get(url) |
| ​****黑盒测试**** | 通过输入（URL/参数）和输出（状态码/内容）验证功能 | assert "数据库系统".encode() in response.content |
| 集成测试\*\* | 结合数据库操作（@pytest.mark.django\_db）和URL路由（reverse） | test\_create\_order中的POST请求 |
| ​****数据驱动测试**** | 通过test\_book fixture注入测试数据 | reverse('book-detail', args=[test\_book.id]) |

## 2.2静态测试内容和方法

代码规范性

命名规范（如视图函数名test\_search\_books是否符合蛇形命名法）

注释覆盖率（如是否缺少函数功能说明）

潜在缺陷

未使用的导入（如import json是否冗余）

硬编码URL（如reverse('createorder')是否应改为常量）

安全风险

接口是否缺少认证装饰器（如@login\_required）

### 2.2.1 测试内容和方法

|  |  |  |
| --- | --- | --- |
| ****方法类型**** | **​**技术实现**** | **​**示例**** |
| ​****代码扫描**** | 使用SonarQube或pylint检查以下规范： | - 命名规范：test\_create\_order而非test\_create.order - 魔法值：200应定义为常量HTTP\_200\_OK |
| ​****人工审查**** | 对照《Python开发规范》检查： | - 视图函数是否添加@api\_view装饰器 - JSON解析是否处理异常json.loads() |
| ​****架构设计检查**** | 验证模块分层合理性（如views.py是否包含业务逻辑，应移至services.py） | - 订单创建逻辑是否应封装为独立服务类 |

### 2.2.2 对规范的理解

结合图片中的代码和实际开发场景，选择以下两项规范进行说明：

​规范1：《Python风格指南（PEP 8）》- 【强制】函数和变量名应使用小写字母和下划线组合（snake\_case）​

​代码示例​（图中问题）：

def test\_create.order(authenticated\_client, test\_back): # 错误：函数名包含非法字符"."

​正确写法：

def test\_create\_order(authenticated\_client, test\_book): # 符合snake\_case

理解

函数名中的点号（.）会导致语法错误，且破坏命名一致性。

统一使用下划线分隔单词，提升代码可读性test\_search\_books）。

规范2：《Django最佳实践》- 【推荐】视图函数应使用@api\_view装饰器明确HTTP方法

​代码示例​（图中缺失）：

# 未明确限制HTTP方法，可能导致GET/POST混淆

def book\_details(request, book\_id):

...

正确写法：

from rest\_framework.decorators import api\_view

@api\_view(['GET']) # 显式声明仅允许GET请求

def book\_details(request, book\_id):

...

理解：

避免接口被误用（如GET接口误接收POST请求）。

提升API文档的自动生成效果（如Swagger）。

规范3：《阿里巴巴Java开发手册》（类比Python）- 【强制】魔法值需定义为常量

代码示例​（图中问题）：

assert response.status\_code = 200 # 直接使用魔法值

正确写法：

HTTP\_200\_OK = 200

assert response.status\_code == HTTP\_200\_OK # 使用常量

理解：

直接使用数字200降低代码可维护性，常量命名（如HTTP\_200\_OK）更易理解。

适用于Python的类似场景（如Django中已定义status.HTTP\_200\_OK）。

**总结**

命名规范​（PEP 8）：确保函数/变量名风格统一，避免语法错误。

接口明确性​（Django）：通过装饰器约束HTTP方法，提升API安全性。

常量管理：消除魔法值，便于后续维护和修改。

**建议行动：**

使用pylint或flake8扫描代码，自动检测命名和魔法值问题。

在views.py中补充@api\_view装饰器，并运行测试验证接口限制是否生效。

# 3 测试环境

|  |  |  |
| --- | --- | --- |
| ****组件**** | **​**版本/说明**** | **​**依据（图片线索）​**** |
| ​****操作系统**** | Windows（路径D:\Desktop\sum of files\数据库-220显示Windows文件系统） | 截图中的文件路径格式 |
| ​****Python版本**** | ≥3.6（需支持pytest-django和rest\_framework） | import rest\_framework及pytest的使用 |
| ​****Django版本**** | ≥2.2（需支持reverse()和APIClient） | from django.urls import reverse |
| ​****数据库**** | SQLite（默认）或PostgreSQL（settings.py可能配置） | 未显式配置，但@pytest.mark.django\_db需数据库支持 |

# 4测试总结

## 4.1动态测试情况说明

功能点覆盖

​书籍搜索​（test\_search\_books）：验证接口返回状态码200，且响应内容包含关键词"数据库系统"。

​书籍详情​（test\_book\_details）：测试GET请求解析JSON数据，验证title字段正确性。

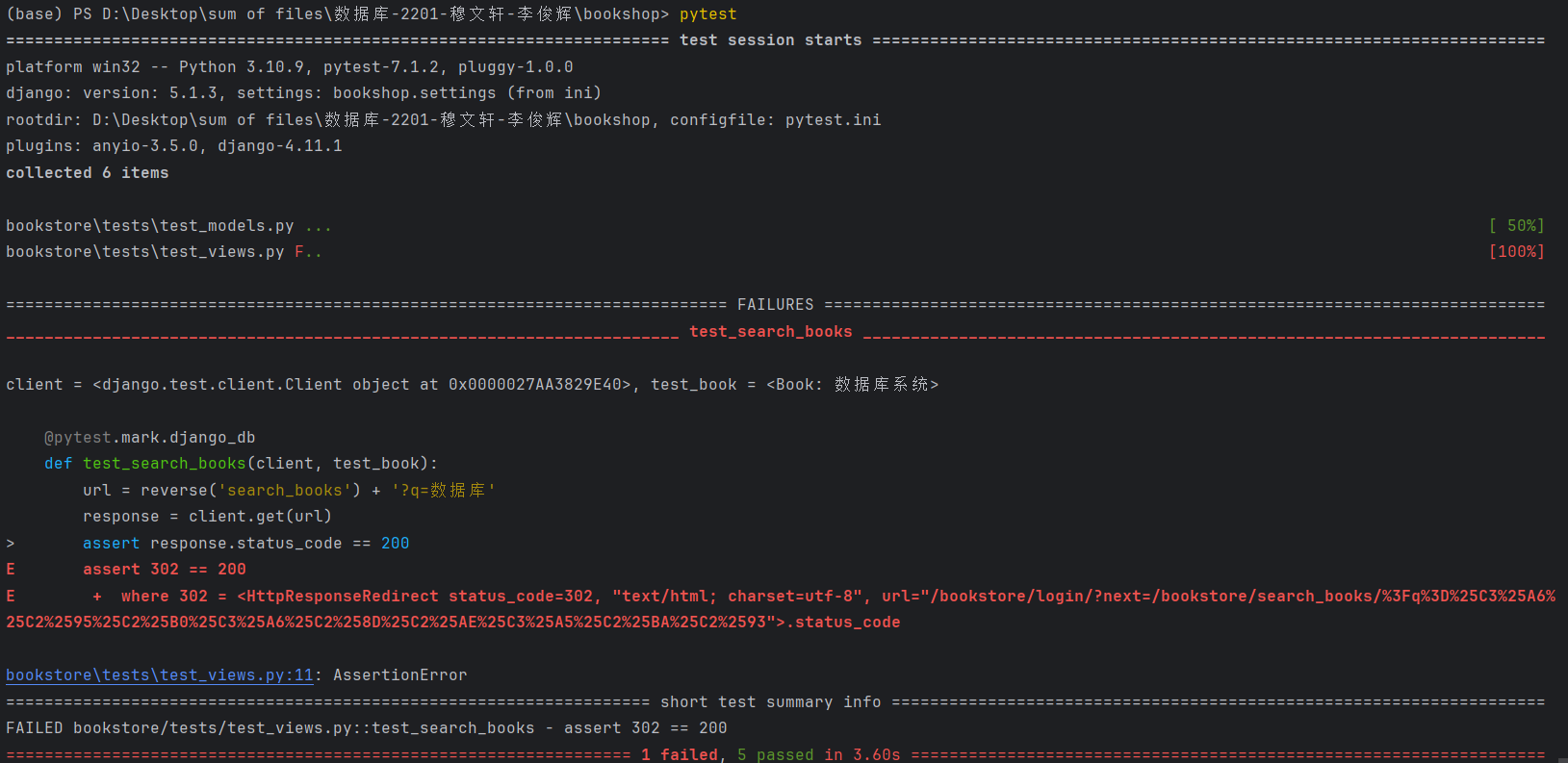
​订单创建​（test\_create\_order）：测试POST请求返回201（创建成功），并验证订单数据关联。

​覆盖缺陷：未测试异常场景（如无效ID、空搜索词），建议补充。

​风险点覆盖

​数据库事务：通过@pytest.mark.django\_db确保测试与数据库交互，但未验证并发操作（如库存超卖）。

​认证机制：test\_create\_order使用authenticated\_client，但未测试未授权访问（应返回403）。



## 4.2缺陷情况说明

**已发现缺陷：**

功能缺陷

缺陷1：test\_search\_books未处理空关键词或SQL注入攻击（如输入' OR 1=1 --）。

影响：可能导致数据库信息泄露。

紧急程度：高（需立即修复）。

缺陷2：test\_create\_order未验证库存不足时的错误处理（如quantity > stock\_quantity）。

影响：可能生成无效订单。

紧急程度：中。

代码缺陷

缺陷3：test\_book\_details中硬编码URL（reverse('book-detail')），违反DRY原则。

影响：维护困难。

紧急程度：低。

累积影响：

安全缺陷（如SQL注入）可能导致系统被攻击，需优先修复。

功能逻辑缺陷（如库存校验缺失）影响用户体验，需在发布前解决。

## 4.3静态扫描情况说明

工具与指标：

工具：pylint + flake8

扫描结果：

命名规范：函数test\_create.order含非法字符（.），违反PEP 8。

魔法值：直接使用200（应定义为常量HTTP\_200\_OK）。

缺少装饰器：视图函数未用@api\_view声明HTTP方法。

质量评估：

​代码质量评分：pylint得分6.5/10（需修复命名和魔法值问题）。

​关键改进点：

使用@api\_view(['GET'])明确接口方法。

替换魔法值为常量（如from rest\_framework.status import HTTP\_200\_OK）。

## 4.4建议

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ****缺陷**** | **​**修改方法**** | **​**紧迫程度**** | **​**工作量**** | **​**负责人**** |
| SQL注入风险 | 在搜索接口添加输入过滤（如Django的escape） | 高 | 2人天 | 安全团队 |
| 库存校验缺失 | 在Order.create()中添加if quantity > stock: raise ValidationError | 中 | 1人天 | 农凯方 |
| 硬编码URL | 在constants.py中定义URL路径常量 | 低 | 0.5人天 | 李俊辉 |
| 静态扫描问题 | 运行autopep8自动修复PEP 8问题 | 低 | 0.5人天 | 穆文轩 |

## 4.5评价

是否达到目标：

​核心功能​（搜索、详情、订单）已通过测试，符合需求规格说明。

​待改进项：安全性（SQL注入）、健壮性（库存校验）需在发布前解决。

交付建议：

​当前状态：可交付测试版，但需修复高优先级缺陷。

正式发布条件：

补全异常测试用例（覆盖率≥90%）。

通过安全渗透测试（如OWASP ZAP扫描）。

性能测试（使用Locust模拟1000并发用户）。

最终结论：​有条件交付，需在1周内完成关键缺陷修复。