|  |  |  |
| --- | --- | --- |
|  | | |
|  | | |
| **毕 业 设 计** | | |
| 测试事务管理平台的设计与实现 | | |
|  | | |
| 学生姓名： | 于丽蔷 |  |
| 专业班级： | 软件工程2017级4班 |  |
| 指导教师： | 李莉 副教授 |  |
| 学 院： | 信息与计算机工程学院 |  |
|  | | |
| 2021年6月 | | |

|  |
| --- |
| **测试事务管理平台的设计与实现** |

摘要

随着计算机技术的飞速发展，计算机软件复杂度不断提升，人们对软件质量有了更加严格的要求。软件测试则是软件生命周期中保证软件质量的重要过程，这个过程的核心任务在于严格执行有效的软件测试用例，及时反馈测试问题[1]。而有效的软件测试用例设计工作的难点在于：当需求迭代速度快、需求点繁多时，通过人工构造测试用例往往出现回归无依据、基础用例遗漏、复杂场景考虑不周的情况。目前国内大多数互联网公司已经通过开发管理平台来管理测试用例，方便回归测试时进行复用，但手工录入测试用例一定程度上会加重测试工作量、延缓测试进度，因此软件测试用例管理平台利用效率不高。

本选题针对目前的软件测试用例管理平台进行进一步开发，设计以MySQL作为数据库，基于SpringBoot、Vue.js实现B/S架构。根据实际生产需求，将平台划分为用户管理、测试用例管理、自动化测试框架管理三大模块。测试用例管理模块除测试用例管理功能方便测试用例评审和复用之外，增加两个功能：一是通过链接的形式关联缺陷报告管理功能，其意义在于减少测试人员反馈测试问题时需描述用例场景的工作量，提高测试工作效率，形成测试流程闭环；二是测试用例自动生成功能，可以基于分词和推荐算法实现已有相关测试用例推荐功能，还可以根据边界值分析法等测试理论实现测试用例的自动生成，将部分人工工作量转化为自动化工作量，提高测试工作效率，降低人为因素对测试过程的干扰。

本文针对以上需求，在平台上实现了以上功能，通过对用例图描述每个模块的基本功能，通过E-R图、数据流图、时序图进行了概要设计和详细设计，最终开发出测试事务管理平台。开发结束后对该系统进行了充分测试，测试结果表明本系统满足测试需求。后续相关使用人员调查表明，本平台的测试用例管理关联缺陷报告管理功能大大节约了测试人员和开发人员的沟通成本，减少了测试人员对复现场景描述的工作量，实现了测试流程的快速推进；测试用例推荐和自动生成功能在一定程度上减少了测试人员编写测试用例的时间，弥补了测试人员编写测试用例时由于不熟悉关联业务而导致考虑不周的问题，提高了测试工作质量和效率。

**关键词**　软件质量保障；测试用例管理；B/S架构；推荐算法；边界值分析法

|  |
| --- |
| **Design and implementation of**  **Test Transaction Management platform** |

**Abstract**

With the rapid development of computer technology, the complexity of computer software continues to improve, people have more stringent requirements for software quality. Software testing is an important process to ensure the quality of software in the software life cycle. The core task of this process is to strictly implement effective software test cases and timely feedback test problems. The difficulty of effective software test case design lies in: when the iterative speed of requirements is fast and there are many requirements, the artificial construction of test cases often leads to the situation that there is no basis for regression, the omission of basic cases, and the thoughtlessness of complex scenarios. At present, most of the domestic Internet companies have managed the test cases through the development management platform to facilitate the reuse of regression testing. However, manual entry of test cases will increase the test workload and delay the test progress to a certain extent, so the utilization efficiency of the software test case management platform is not high.

This topic is based on the current software test case management platform for further development, design MySQL as the database, based on SpringBoot, Vue.js to achieve B/S architecture, according to the actual production needs, the platform is divided into user management, test case management, automated test framework management of three modules.Test case management module in addition to the test case management features convenient test case review and reuse, adding two functions: one is through the link in the form of associated defect report management functions, its significance lies in reducing testers feedback test question to describe the use case scenarios workload, improve test efficiency, formation testing process closed loop;2 is the function of automatic generation of test cases, can be based on segmentation and recommendation algorithm existing test cases related recommendations, can also according to the theory of boundary value analysis testing to realize automatic generation of test cases, automation work, part of the artificial workload can be converted to improve test efficiency, decrease the disturbance of human factors on the testing process.

According to the above requirements, this paper realizes the above functions on the platform, describes the basic functions of each module through the use case diagram, and carries out the outline design and detailed design through E-R diagram, data flow diagram and sequence diagram. Finally, the test transaction management platform is developed. After the development, the system is fully tested, and the test results show that the system meets the test requirements. The follow-up survey of relevant users shows that the test case management associated defect report management function of this platform can greatly save the communication cost between testers and developers, reduce the workload of testers to describe the recurrence scenario, and realize the rapid promotion of the test process; The function of test case recommendation and automatic generation reduces the time for testers to write test cases to a certain extent, makes up for the problem that testers are not familiar with the associated business when writing test cases, and improves the quality and efficiency of testing work.

**Keywords** Software quality assurance; Test case management; B/S architecture; Recommendation algorithm; Boundary value analysis method

**目录**

**摘要**

**Abstract**

[1 绪论 1](#_Toc23208)

[1.1 课题研究背景及意义 1](#_Toc28602)

[1.2 国内外研究现状 1](#_Toc4256)

[1.3 课题研究目标与内容 1](#_Toc30797)

[1.4 论文组织结构介绍 2](#_Toc25113)

[1.5 本章小结 2](#_Toc621)

[2 相关理论基础与关键技术 3](#_Toc1975)

[2.1 基础测试理论 3](#_Toc1723)

[2.1.1 软件测试流程 3](#_Toc22521)

[2.1.2 主要测试方法 3](#_Toc18577)

[2.1.3 测试用例设计 3](#_Toc7234)

[2.2 分词算法 3](#_Toc31087)

[2.3 关联规则挖掘 4](#_Toc16150)

[2.3.1 关联规则挖掘基本概念 4](#_Toc25800)

[2.3.2 关联规则挖掘算法 4](#_Toc9518)

[2.4 SpringBoot 框架 4](#_Toc20279)

[2.5 Vue.js 框架 4](#_Toc20571)

[2.6 MySQL 5](#_Toc19560)

[2.7 本章小结 5](#_Toc8778)

[3 需求分析 6](#_Toc10550)

[3.1 平台分析概述 6](#_Toc1708)

[3.2 平台可行性分析 6](#_Toc13262)

[3.2.1 技术可行性 6](#_Toc18251)

[3.2.2 经济可行性 6](#_Toc14809)

[3.2.3 社会因素可行性 6](#_Toc8721)

[3.3 平台需求分析 7](#_Toc5365)

[3.3.1 功能需求 7](#_Toc14355)

[3.3.2 非功能需求 7](#_Toc8204)

[3.4 平台建模 7](#_Toc26135)

[3.4.1 用例建模 7](#_Toc1095)

[3.4.2 数据驱动建模 8](#_Toc23091)

[3.5 本章小结 9](#_Toc685)

[4 概要设计 10](#_Toc6003)

[4.1 平台开发结构设计 10](#_Toc18354)

[4.2 平台总体框架设计 10](#_Toc24494)

[4.3 平台功能模块设计 11](#_Toc64)

[4.3.1 用户管理模块 12](#_Toc31977)

[4.3.2 测试用例管理模块 12](#_Toc25599)

[4.3.3 缺陷报告管理模块 12](#_Toc28416)

[4.3.4 自动化测试框架管理模块 12](#_Toc18149)

[4.3.5 缺陷报告管理模块 13](#_Toc26476)

[4.4 核心功能设计 13](#_Toc5215)

[4.5 平台数据库设计 14](#_Toc3166)

[4.5.1 E-R图设计 14](#_Toc26760)

[4.5.2 数据库表设计 14](#_Toc1415)

[4.6 本章小结 16](#_Toc24647)

[5 详细设计与实现 17](#_Toc10304)

[5.1 平台界面设计与实现 17](#_Toc16216)

[5.1.1 用户登录功能界面设计与实现 17](#_Toc1531)

[5.1.2 用户管理界面设计与实现 17](#_Toc17278)

[5.1.3 测试项目菜单设计与实现 17](#_Toc30493)

[5.1.4 测试用例管理界面设计与实现 18](#_Toc30330)

[5.1.5 缺陷报告管理界面设计与实现 18](#_Toc19882)

[5.1.6 统计报表界面设计与实现 19](#_Toc7285)

[5.2 平台业务逻辑层设计与实现 19](#_Toc30325)

[5.2.1 用户登录 20](#_Toc2814)

[5.2.2 测试用例管理 20](#_Toc17587)

[5.2.3 自动化测试框架管理 23](#_Toc21718)

[5.2.4 测试问题处理 25](#_Toc17173)

[5.3 平台核心算法层设计与实现 25](#_Toc22802)

[5.3.1 分词算法 25](#_Toc32479)

[5.3.2 推荐算法 26](#_Toc30815)

[5.4 本章小结 27](#_Toc9367)

[6 测试 28](#_Toc2421)

[6.1 测试目的及方法 28](#_Toc17212)

[6.2 测试工具及环境 28](#_Toc11761)

[6.3 功能测试 28](#_Toc29224)

[6.3.1 UI 测试 28](#_Toc29487)

[6.3.2 用户管理模块功能测试 28](#_Toc14087)

[6.3.3 测试用例管理模块功能测试 3](#_Toc18148)4

[6.3.4 缺陷报告管理模块功能测试 30](#_Toc13454)

[6.3.5 自动化测试框架管理模块功能测试 31](#_Toc17792)

[6.4 本章小结 32](#_Toc12239)

[结论 33](#_Toc22686)

[参考文献 3](#_Toc10601)4

[致谢 3](#_Toc3606)5

# 

# 绪论

## 课题研究背景及意义

随着计算机软件复杂度不断提升，人们更加看重软件的使用体验及安全等方面。随之而来的问题就是软件从业者如何保证软件质量，保证用户信息和资产安全，提高用户产品使用体验。因此，软件测试行业逐渐趋向成熟[2]。

软件测试是软件生命周期中保证软件质量的重要过程，这个过程的核心任务在于严格执行有效的软件测试用例，及时反馈测试问题。因此，通过开发可视化测试用例管理平台完成测试事务的跟进和合作十分重要。

## 国内外研究现状

在国外一些软件行业发达的国家，如IBM、Microsoft等一些大型软件系统开发公司，软件测试得到了非常多的重视，测试人员占比高，测试技术成熟，测试工具丰富多样，目前国外同类软件主要有Rational公司提供的系列产品，如开放的、可扩展的[测试管理工具](https://baike.baidu.com/item/%E6%B5%8B%E8%AF%95%E7%AE%A1%E7%90%86%E5%B7%A5%E5%85%B7" \t "https://baike.baidu.com/item/Rational/_blank)Test Manager，灵活的缺陷和变更跟踪工具Clear Quest系列等。其中Suite系列产品不仅包括综合的测试计划和管理，还能用面向对象的录制方式快速生成测试脚本。但以上产品，价格昂贵，为英文系统，在功能上没有测试案例具体步骤的管理查询。Test Manager作为 Suite软件包的一部分捆绑出售，用户只需要其中一步分需求缺不可以单独购买，造成不必要的开销[3]。

我国的软件测试技术于上世纪八十年代起源，并伴随着软件工程领域的研究发展而发展[4]。最开始软件规模小、复杂程度低，通常由开发人员自己完成“调试”工作。后来国内软件和IT行业对测试的重视程度也很低，测试并不能贯穿软件生命周期，往往发现问题时需要付出很大的维护代价，且测试单一化，测试流程不够完善，自动化程度不高，软件测试相对开发人员人数极少。近年来，国内软件行业发展也越来越迅速，软件测试也逐渐被重视，测试自动化工具不断出现，测试流程和监督体系不断完善，测试人员也需要了解更多的相关技术。本人在实习过程中，测试相关工作采用的是i-Case和 Bits管理系统，上述系统主要针对测试用例的增删改查、导入导出等功能提供了解决方案，具有一定通用性，但功能单一，且手工录入测试用例工作量大，可能导致测试效率降低；执行状况记录功能没有得到重视，不能及时更新执行结果，会导致测试的随机性和盲目性；需求迭代快，新入职人员在不完全了解业务时，会遗漏很多业务场景，导致由测试遗漏产生的线上问题。

## 课题研究目标与内容

基于以上，本课题预期目标设计并实现测试用例管理及生成系统，针对需求实现测试用例管理和缺陷问题跟进，兼容自动化测试框架管理，实现测试流程闭环。

该系统宏观分为用户管理、测试用例管理、自动化框架管理三个模块，其中用户管理基于企业应用实际，主要进行用户信息记录和根据职位不同进行权限分配；测试用例管理主要包括测试用例的生成、维护、跟踪和统计功能，其中生成功能可以进行测试用例的自动生成并保存至平台；除此之外，可根据实际执行情况修改执行状态和实际结果，并关联缺陷报告；自动化框架管理基于GitHub实现自动化测试代码的管理并统计代码覆盖率情况、生成自动化测试报告，自动化测试报告使用 JaCoCo 插件，在代码中进行插桩统计覆盖率，生成覆盖率报告index.html文件，并统计自动化测试用例执行情况，形成自动化测试报告。

## 论文组织结构介绍

本文论文结构基于软件工程开发模型中的瀑布模型，共分为7章。

第1章：绪论，介绍本平台的研究背景、国内外测试行业的现状和针对当前现状进行测试工具改进的目标。

第2章：相关理论基础与关键技术综述，介绍在本项目开发过程中使用的核心算法相关理论基础与关键开发技术。

第3章：需求分析，主要基于作者实习期间的工作经历，介绍本平台的需求及其可行性。

第4章：概要设计，介绍该平台的总体架构设计方案、功能模块设计方案、数据库设计方案等。

第5章：详细设计与实现，介绍文中涉及到的界面层、业务逻辑层、核心算法层的详细实现方式设计和实现结果展示。

第6章：测试，介绍平台相关运行环境，在此基础上对平台进行功能测试，确保平台符合需求要求。

第7章：结论，对平台设计与实现进行全面总结，包括开发工作、开发结果以及后续优化等内容。

## 本章小结

本章主要描述了课题背景，并详细讨论课题的意义以及相关课题的研究现状，还明确了主要的研究目标和内容，并介绍了论文的组织结构，对整个课题的工作进行全面概述。

# 相关理论基础与关键技术

## 基础测试理论

### 软件测试流程

根据作者实习经历，软件测试工作的主要流程为：需求评审、技术评审、制定测试用例、评审测试用例、冒烟测试、正式测试、回归测试、编写并提交测试报告。

在本课题中，测试用例通过关联需求文档和缺陷报告，形成测试流程闭环，提高测试效率。

### 主要测试方法

从是否关心软件内部结构和具体实现的角度划分，测试方法主要有白盒测试和黑盒测试。白盒测试方法主要有代码检査法、语句覆盖等，黑盒测试方法主要包括等价类划分法、[边界值分析法](https://baike.baidu.com/item/%E8%BE%B9%E7%95%8C%E5%80%BC%E5%88%86%E6%9E%90%E6%B3%95/4137943" \t "/Users/bytedance/Documents\x/_blank)、[判定表驱动法](https://baike.baidu.com/item/%E5%88%A4%E5%AE%9A%E8%A1%A8%E9%A9%B1%E5%8A%A8%E6%B3%95/8643441" \t "/Users/bytedance/Documents\x/_blank)等[5]。除此之外，测试方法按软件特性还可分为功能测试、性能测试、兼容性测试、安全测试等[6]。

在本课题中，根据以上测试方法实现测试用例自动生成功能，对于取值范围使用边界值分析法；除此之外，还可实现功能测试对于字符串输入采用等价类划分法，对于参数组合采用判定树/判定表法，对操作流程判断采用分支覆盖法等；性能测试提取并发数、并发持续时间、业务类型及业务占比、生产环境基础数据量、预期响应时间、系统其他特殊性能值需求（如net I/O不能占用带宽1/2）等；兼容性测试可根据自动爬取网络上主流的浏览器类型、操作系统、手机端机型的统计数据和导入项目组在APP启动时埋点生成的数据分析报表资源，生成推荐测试列表；安全测试关注输入内容的敏感信息加密、批量操作可行性、密码的SQL注入等情况。

### 测试用例设计

1. 完整的测试用例，一般包括用例编号、用例标题、前提条件、操作步骤、预期结果等[7]，需要满足以下原则：
2. 全面性：输入数据要包括合法的、边界内的、常规的数据，也要包括非法的、边界上的、超过边界的、不合理的数据。
3. 代表性：选取其中具有代表性的数据作为输入数据，减少测试用例的冗余性。
4. 可判定性：每个用例必须要有明确的输出结果，用来判断实际结果是否符合预期，进而判定软件质量。
5. 可操作性：需要描述清楚操作步骤，不同步骤对应不同的输出结果，在测试前做好足够的准备工作，以提高测试的效率。
6. 可再现性：在执行测试用例的过程中，相同的测试用例系统所执行的测试结果是相同，以便复现和定位问题[8]。

在本课题中，基于测试用例内容和设计原则的需求，进行了相关数据库设计，确保系统的实用性。

## 分词算法

分词方法有标准分词、NLP分词、索引分词、N-最短路径分词、CRF分词以及极速词典分词等[9]。

在本课题中，通过使用Python中文分词软件中的“结巴”分词，在需求文档分词后，根据关键字匹配测试用例，实现测试用例推荐功能[10]。

## 关联规则挖掘

### 关联规则挖掘基本概念

关联规则挖掘是从大量数据中挖掘出变量之间的相互联系的方法，是数据挖掘中最活跃的研究方法之一**。**关联规则挖掘的对象是事务数据库的记录，一个事务数据库中的关联规则挖掘可以用以下形式表示：

设是一个由m个不同项目组成的集合，是一个由一系列具有唯一标识符的事务组成的事务数据库，每一个事务都对应上的一个子集[11]。

项集：项集是一个包含数据项的集合。

支持度：事务集中的项集的支持度记为，表示包含的事务在事务集中所占的百分比，见公式2.1。

 (2.1)

其中，表示包含项集的事务数量。

若有两个项目集和，，交集为，则关联规则的支持度公式如公式2.2所示。

 (2.2)

置信度：置信度表示包含和的事务数与包含的事务数值比，见公式2.3。置信度体现了关联规则的可靠性，置信度越高说明在包含的事务中出现的可靠性越大。

 (2.3)

频繁项集和最大频繁项集：对于事务数据库中所有满足最小值尺度的项目集称为频繁项目集，否则称为非频繁项目集。若频繁项集的所有超集都是非频繁项集，则称为最大频繁项集。

### 关联规则挖掘算法

**关联规则一般将满足一定阈值的事物通过最小支持度和最小置信度筛选出来，进行关联规则挖掘的首要步骤是进行频繁项挖掘，经典的频繁项挖掘算法有**Apriori**算法**[12]，该算法自底向上逐层搜索迭代，利用项集去搜索项集，直到所有项数的频繁项都被挖掘出来。本文将使用这种算法对测试用例进行推荐生成。

## SpringBoot 框架

Spring Boot减少了配置web.xml、数据库连接、 Spring 事务、加载配置文件的读取、日志文件等工作，使用特定的方式来进行配置，从而较少开发人员工作量[13]。

在本课题中，平台通过使用Spring Boot框架初步搭建，使用逆向工程自动生成SQL文件，提高了开发效率。

## Vue.js 框架

Vue.js专注View层，提供了MVVM风格的双向数据绑定的Javascript库，它被设计为可以自底向上逐层应用，便于与第三方库或既有项目整合[14]。

本课题因采用MVVM架构，所以开发只需关注业务逻辑，View层和Model层的同步则通过双向数据绑定进行连接，大大减少了项目开发量，提高了开发效率。

## MySQL

MySQL是目前最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL是最好的关系数据库管理系统)应用软件之一[15]。

在本课题中，主要用于储存用户信息和测试流程中的相关信息，为用户提供所操作的数据源，以保证平台功能的实现，使用MySQL搭建关系型数据库，关系型数据库将数据保存在不同的表中，可以增加速度并提高灵活性。

## 本章小结

本章重点介绍了在本课题中依赖的相关理论基础、使用工具和相关技术，并阐述了相关算法的基本原理，后文则会基于本章的内容进行引申与应用。

# 需求分析

本课题所需要构造的测试事务管理平台，主要用户是开发人员和测试人员。因此，平台的功能由测试流程的整体需求决定。本文基于对作者实习期间的工作经历，完成了该平台的可行性研究与需求分析。

## 平台分析概述

对本课题测试事务管理平台进行分析，平台功能主要划分为三大部分：用户管理、测试用例管理、自动化框架管理。上述三个部分之间相互独立、相互联系。

从总体框架设计上看，系统采用面向对象的编程范式，本着低耦合、高内聚的设计原则[16]，采用可视化的界面设计，以达到预期效果，预留足够的系统接口，保证了代码的可复用性和系统的可拓展性，为后期业务迭代提供可能。

## 平台可行性分析

可行性分析是软件生命周期中的最初阶段[17]。在软件开发过程中，可行性分析会在需求确定之前进行，在此阶段，将对目标软件开发的可行性进行全方位考虑，可行性分析结果将直接影响项目的成立以及后续开发与维护工作的顺利进行。本文重点围绕技术可行性、经济情况可行性与社会环境可行性等方面进行可行性分析。

### 技术可行性

在本课题中，技术可行性是指功能的实现和质量的保证[18]。

对于本课题来说，使用者主要是软件测试人员和开发人员，用户使用过程中不存在高性能需求，数据存储方面不需要采用大型数据库。对于初步 B/S框架而言，使用SpringBoot和Vue.js进行实现，核心功能通过“结巴”分词软件和推荐算法实现即可，不存在功能实现方面的困难。除此之外，作者实习过程中已积累近一年的测试经验，对于质量保障方面具有一定的理解，可在测试过程中对软件质量进行把关。

综上，可在原定开发时间中完成课题功能需求的实现，并保证软件的质量，本平台在技术实现层面具备较强的可行性。

### 经济可行性

在本课题中，经济可行性主要是包括成本和收益[19]。

对于软件成本而言，本平台采用B/S架构，前端运行在本地浏览器Google Chrome上，后端使用IntelliJ IDEA进行编码开发，使用学校提供的教育邮箱可避开软件收费问题，其他软件则均为开源版本或者免费版本，不存在经济额外支出。对于硬件成本而言，基于作者已有笔记本电脑，软件部署在本地Windows10 系统。

综上，该课题的实现过程中不会产生实际的经济开销，本平台在经济上具有较好的可行性。

### 社会因素可行性

在本课题中，社会因素可行性主要是包括法律可行性、使用可行性[20]。

对于法律因素，所有开发软件都选用正版，开发过程和开发内容严格遵守有关知识产权法和软件行业的相关法律法规，不会侵犯任何个人、集体、国家的利益；使用者主要为团队内部人员，具备专业技术素养，简单介绍即可使用。

综上，该课题开发合法合规，且具有易用性，本平台在社会因素上具有较好的可行性。

## 平台需求分析

本平台主要用户为开发人员和测试人员，因此，下面将根据平台的用户使用需求对平台主要功能进行需求分析。

### 功能需求

基于作者实习经历中的使用需要，得出平台的需求如下：

**（1）用户管理模块：**本平台主要包括三类用户，管理员、开发人员与测试人员。管理员登录后，可以进行用户账号和权限分配；测试人员登录后，可以修改个人信息，进行测试用例和自动化框架维护；开发人员登录后，可以修改个人信息，进行缺陷报告维护和测试用例查看。

**（2）测试用例管理模块：**该模块包括测试用例的新建（包括手动创建和自动创建），导入、导出测试用例，测试用例查询、修改和删除，测试用例执行状态标记并关联缺陷报告等子功能。

**（3）缺陷报告管理模块：**该模块主要实现缺陷报告的新增、修改、删除、查询功能，方便开发人员和测试人员对缺陷问题进行跟进**。**

**（4）自动化测试框架管理模块：**该模块主要实现自动化测试框架的代码管理，并通过JaCoCo插件进行覆盖率统计和测试报告的生成，包括测试用例的统计功能**。**

### 非功能需求

对于非功能需求，一般包括性能需求、硬件需求等除功能需求之外的内容，在本课题中，主要包括以下需求[21]：

（1）界面需求

符合实际测试工作场景，采用统一的配色风格、符合用户操作习惯的功能布局，降低用户操作难度。

（2）性能需求

主要针对于查询等功能，保证平台可至少同时承受请求20次/秒，在进行数据库操作时，较少时间损耗，平均在2秒内返回执行结果。

（3）硬件需求

平台可兼容Windows等主流操作系统、Google Chrome等主流浏览器，具备在主流计算机部署运行的能力，具备较强的容错性，保证系统本身的稳定运行。

## 平台建模

根据上述分析，为具体识别平台需求，并对其进行相应设计与实现，本课题建模主要采用交互模型中的用例建模和行为模型中的数据流建模[22]。

### 用例建模

用例建模主要包括正在开发的系统与其他系统之间的交互，有助于识别用户需求，解决系统间可能产生的交流问题[23]。

在本课题中，平台主要涉及到的用户身份包括管理员、测试人员和开发人员。预计不同职位设计的功能共分为三大类：用户管理功能、测试用例管理功能、缺陷报告管理功能。用户管理模块主要是管理员使用的基础服务，用于管理团队人员信息和统计数据等；测试用例管理模块包括文本用例和自动化测试用例框架管理，主要包括测试用例的生成、维护等基础功能和代码的迭代；缺陷报告管理是基于测试用例管理中的关联缺陷报告功能，提供给开发人员进行缺陷处理的入口，确保测试流程闭环。平台用户用例图详细描述如下图3-1所示。

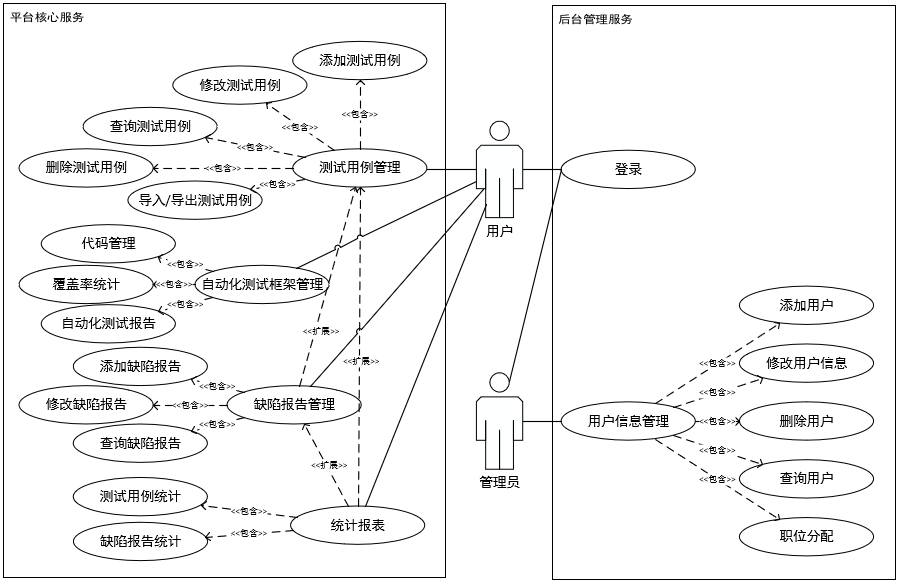


图3-1 平台用户用例图

### 数据驱动建模

数据流建模一般基于数据流图从数据加工者的角度反映了系统必须完成的逻辑功能，以图形方式描绘数据的流动及处理过程[24]。

在实际测试工作流程中，需求评审和技术评审后，由测试人员制定测试用例，再进行测试用例评审，同开发人员一起进行冒烟测试通过后，测试人员进行正式测试。对于缺陷问题编写并提交缺陷报告，待开发人员进行处理后，测试人员进行回归测试。因此，整个平台中主要的数据流走向大致可以分为以下三个数据流图。

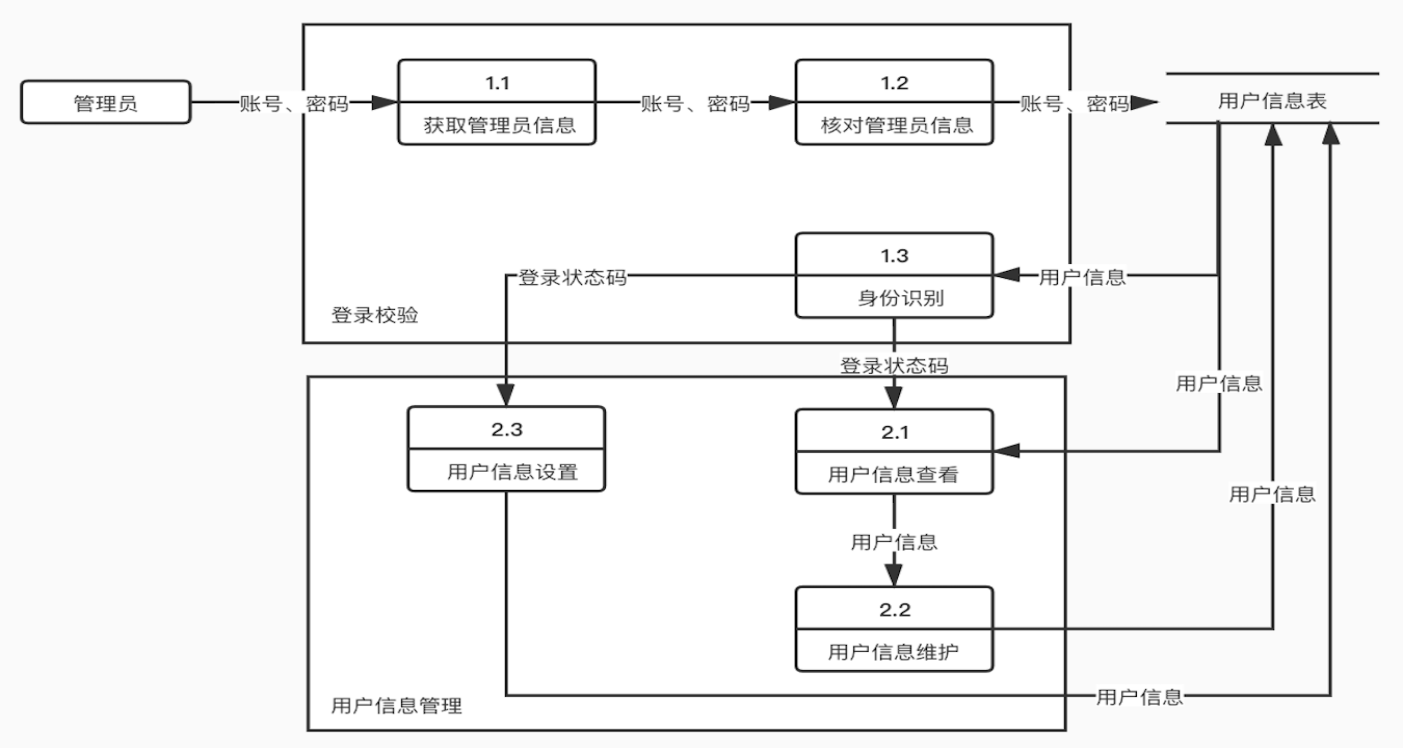


图3-2 管理员业务数据流图

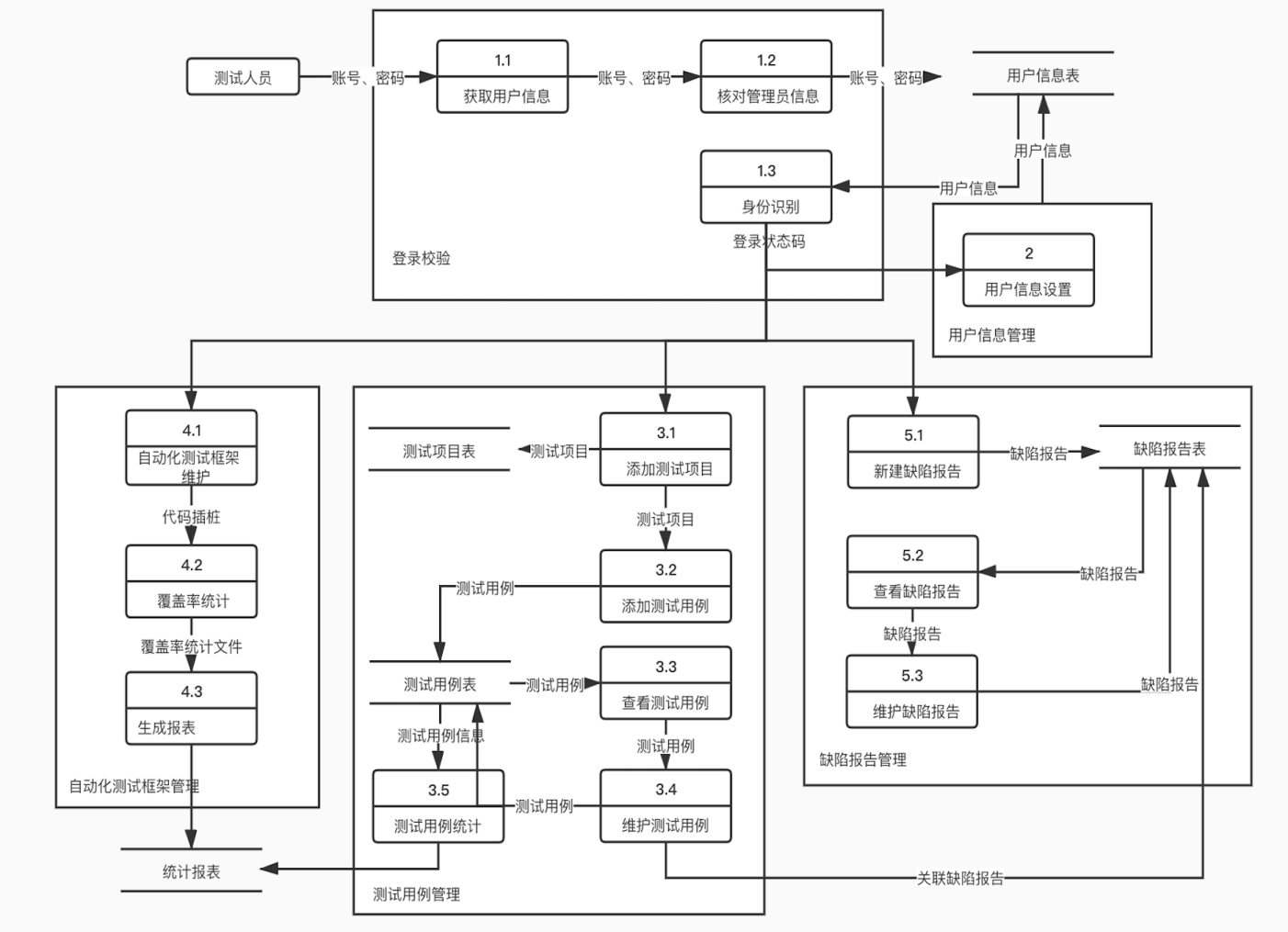


图3-2 测试人员业务数据流图

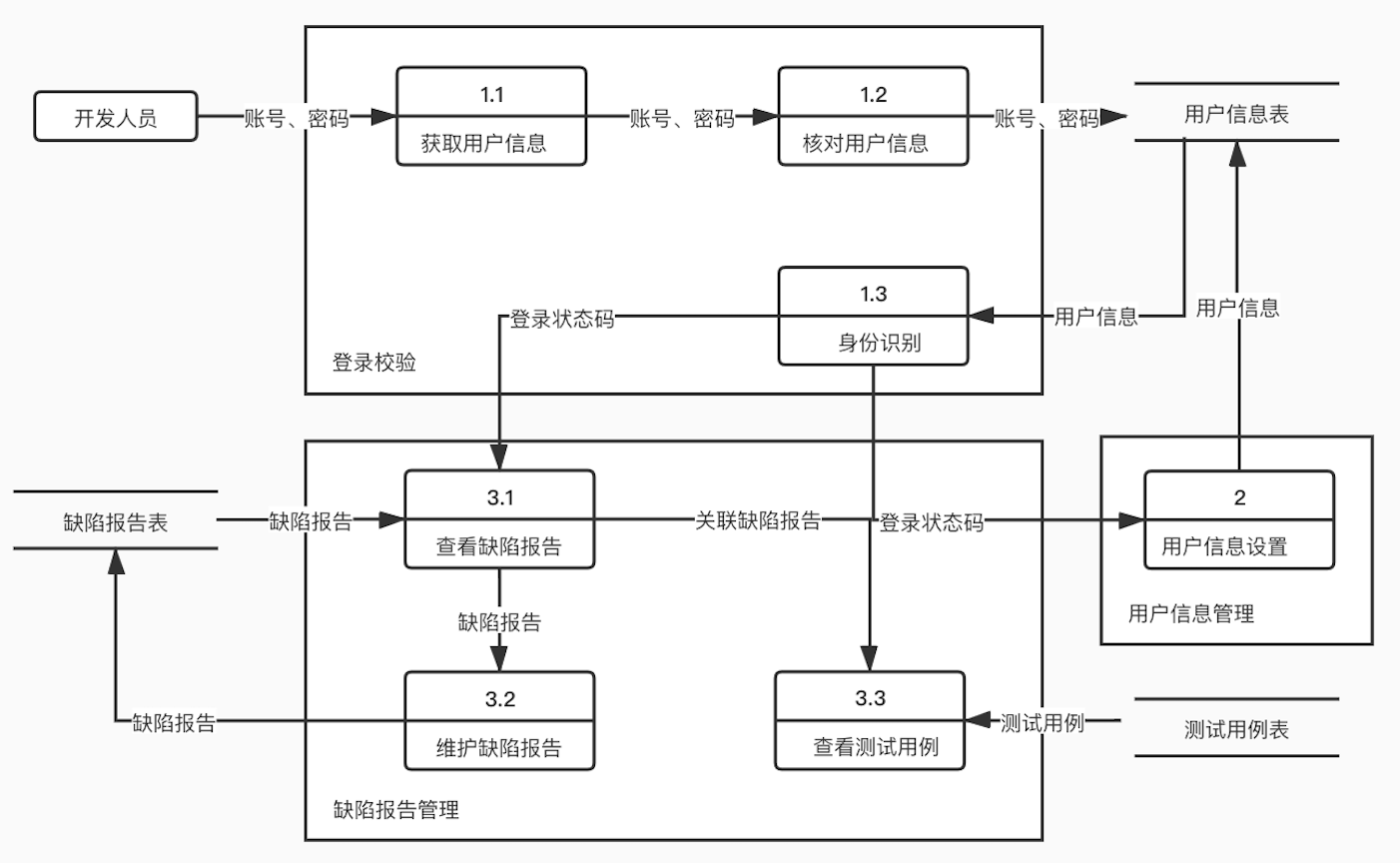


图3-3 开发人员业务数据流图

## 本章小结

本章对平台搭建的可行性进行了分析，同时进行了基于作者实习经历进行了需求分析，并完成了用例建模和数据流建模。下文将基于上述分析进一步阐述平台的构建。

# 概要设计

概要设计，主要是对需求分析之后进行的下一步概要性的工作[25]。其结果决定了平台的易用性和可维护性，概要设计的准确代表着软件风险的大大降低。

在本章中，我们主要对平台的开发架构、功能需求和数据库进行设计，以便保证该平台功能交互顺利。

## 平台开发结构设计

系统开发结构设计的具体流程如图4-1所示。

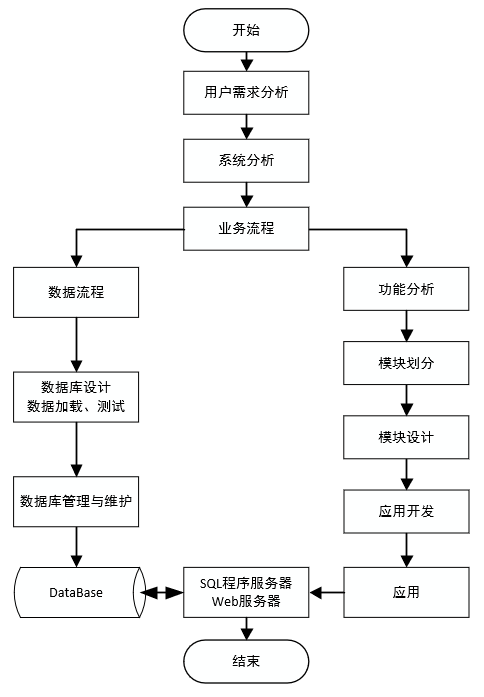


图4-1 系统开发结构设计图

## 平台总体框架设计

本系统采用三层架构模式，将系统划分为表现层、业务逻辑层以及数据访问层，如图 4-2 所示。在表现层中使用 Vue.js 框架，对交互式可视化界面进行描述，实现永华界面和后台代码的分离，最终返回给用户需要的信息。Web Service 层在本系统中作为业务逻辑层出现，在 Web Service 中定义需要的逻辑方法，用户通过表现层将命令发送给逻辑层，调用相关的逻辑方法来更新数据层定义的数据，将操作结果通过表现层返回给用户。数据访问层的作用即是与数据库进行交互，生成对应的数据实体类，完成对数据库的操作[26]。

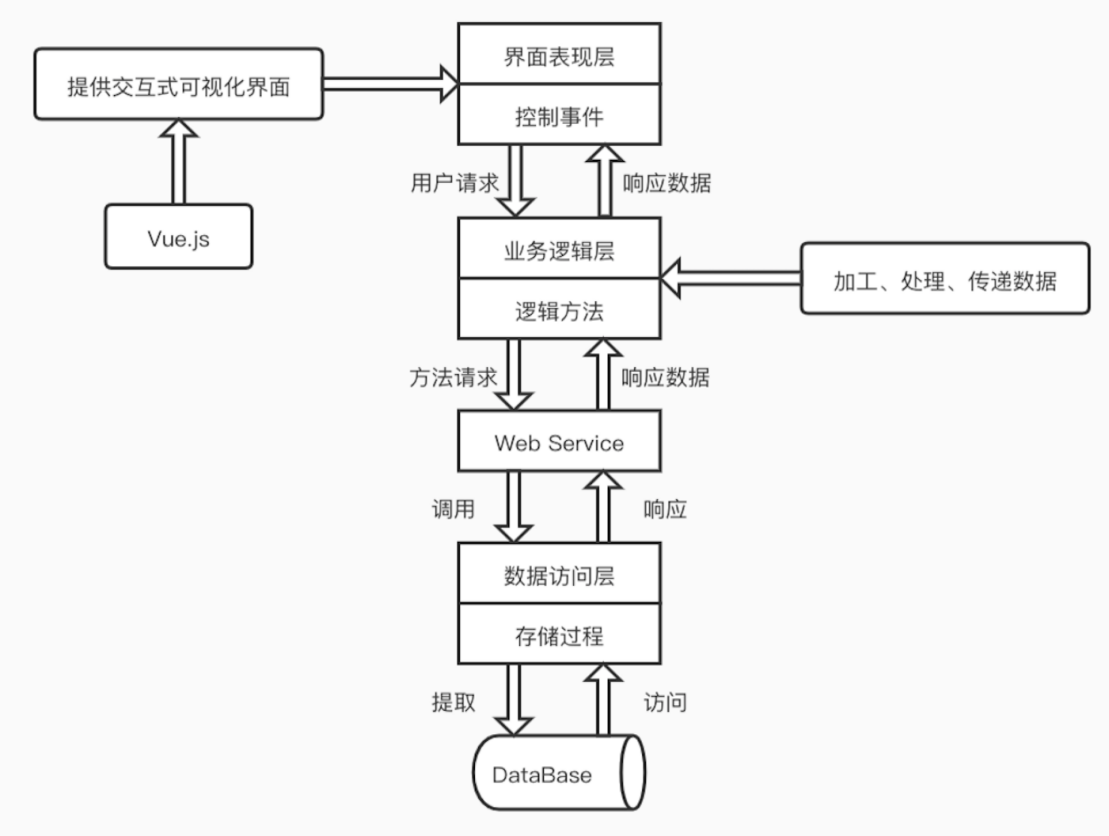


图4-2 系统三层架构图

## 平台功能模块设计

功能结构设计主要用于对软件的功能点和从属关系进行全面的设计[27]。在本课题设计中，采用最简单的树形表现方式来对软件的功能进行全面梳理，主要的功能结构图如图4-3所示。

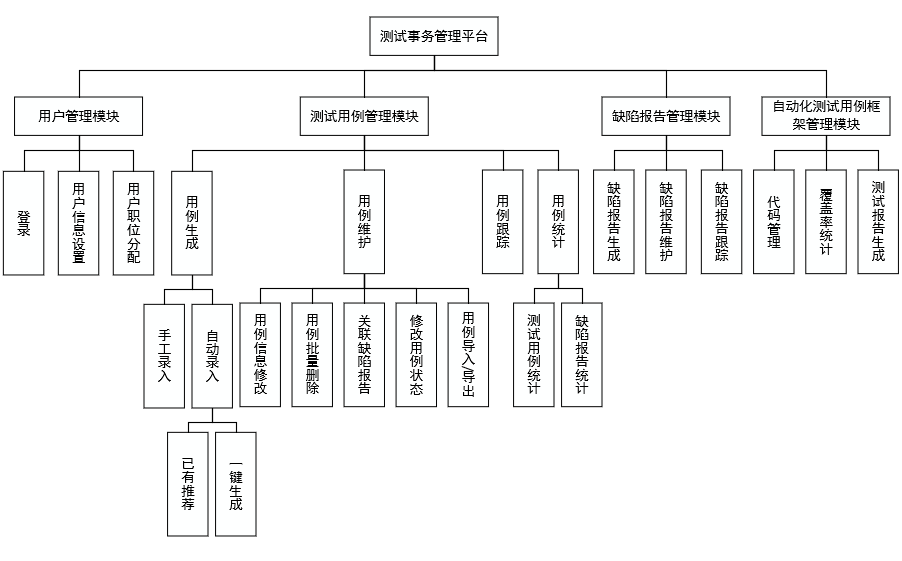


图4-3 系统核心功能结构图

该课题所搭建的核心平台——测试事务管理平台包括三个主要功能模块：用户管理模块、测试用例管理模块和自动化测试框架管理模块，除此之外，根据实际需求单独进行缺陷报告管理。

### 用户管理模块

本平台用户管理模块主要包括登录、用户信息设置，管理员还可进行用户账号分配和职位设置，方便开发人员和测试人员可以根据不同的权限跟进不同项目。

### 测试用例管理模块

用户打开测试用例模块，目录中显示为所有项目文件夹列表，点击项目文件夹，右侧展示相关测试用例，根目录可以显示所有的测试用例。 测试用例管理模块的主要功能点如下：

**（1）用例生成：**

（a）手工录入测试用例：当新建测试用例时，自动生成用例编号和创建时间等信息，用户需要依次填入用例标题、用例描述、测试步骤、预期结果、相关依赖（脚本、数据等）、测试分类等信息，并确认新建，信息完整性校验之后，则新建成功。其中用例信息自动填写创建人和执行人字段，创建人为录入基础信息的用户，执行人为修改执行状态的用户。

（b）自动生成测试用例：根据需求文档分词，匹配已经有的测试用例集推荐生成；或根据测试理论实现自动生成，例如：功能测试对于取值范围使用边界值分析法。

**（2）用例维护：**

（a）修改基本信息：可以进行上述测试用例基本信息的修改；

（b）修改执行状态：测试完成后对应编辑执行状态，方便查看是否测试通过；

（c）关联缺陷报告：如测试不通过，测试人员可提交相应缺陷报告，记录对应链接在测试用例中，方便后入回归；

（d）批量删除用例：测试用例列表采用复选框，如录入有误，可以进行可以进行单选或批量删除；

（e）导入/导出用例：导入测试用例可以下载模板、填写用例信息，再上传至平台；导出可以通过复选框进行测试用例批量导出。

**（3）用例跟踪：**采用字符串匹配算法，根据查询条件进行查询，查询条件包括：根据用例标题查询、根据用户名查询、根据用例状态查询、根据重要程度查询、根据用例类型查询。

**（4）用例统计：**

（a）测试用例统计：采用SQL语句，根据数据库中测试用例表下的用例类型、用例状态、用例等级和用例创建人等字段统计测试用例数量；

（b）缺陷报告统计：根据缺陷报告相关字段，统计项目目录下的缺陷报告数量。

### 缺陷报告管理模块

缺陷报告管理模块主要实现对测试过程中产生的问题以缺陷报告的形式进行更新和管理，包括缺陷报告的生成、维护和跟踪等，与测试用例管理模块类似，在此不做赘述。

### 自动化测试框架管理模块

自动化测试框架管理模块的主要功能点如下：

**（1）代码管理：**实现GitHub功能，可进行代码托管以便实现存储和共享。测试人员通过Git将本地代码提交至同一个项目仓库，进行合入、更新等代码维护工作，实现自动化测试工作合作。

**（2）覆盖率统计：**通过JaCoCo插件，使用插桩的方式记录覆盖率数据。方便收集测试过程中代码覆盖情况，能够很直观展现哪些代码已经测试过，哪些没有被测试，帮助测试人员有针对性地增加用例，提升测试质量。

**（3）测试报告生成：**实现自动化测试用例（如单元测试）的执行，并统计执行结果形成测试报告。

### 缺陷报告管理模块

缺陷报告主要字段包括关联测试用例、问题描述、相关信息（如日志、抓包文件等）、缺陷状态、提交人和跟进人等。缺陷报告管理模块的主要功能点如下：

**（1）缺陷报告生成：**测试人员根据测试用例执行结果新建缺陷报告，并将缺陷报告链接关联至测试用例，方便测试人员后续回归测试，并减少了描述复现场景的工作量，开发人员则可以通过查看缺陷报告关联的测试用例进行问题复现、排查。

**（2）缺陷报告维护：** 除了修改缺陷报告的基本信息外，主要针对缺陷问题的状态进行修改。

**（3）缺陷报告跟踪：** 可以根据查询条件进行缺陷报告查询，查询条件包括：根据缺陷标题查询、根据用户名查询、根据缺陷状态查询、根据优先级查询。

## 核心功能设计

本平台核心功能在于测试用例的自动生成，界面入口为推荐生成和一键生成，需要上传需求文档或测试点文档，推荐生成会根据关键字匹配相关测试用例进行推荐，一键生成可以根据边界值分析生成边界值信息，具体流程如图4-4所示。

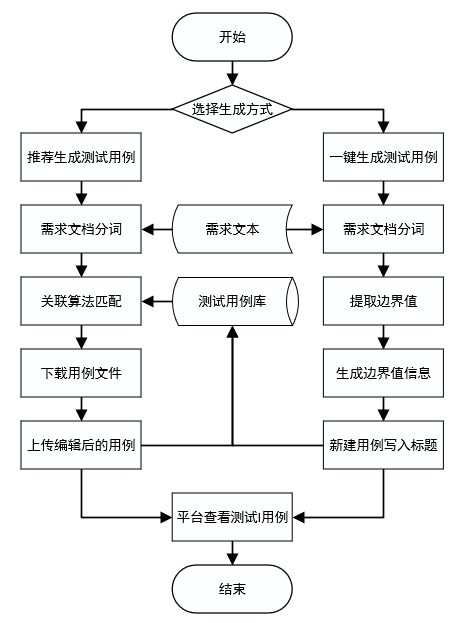


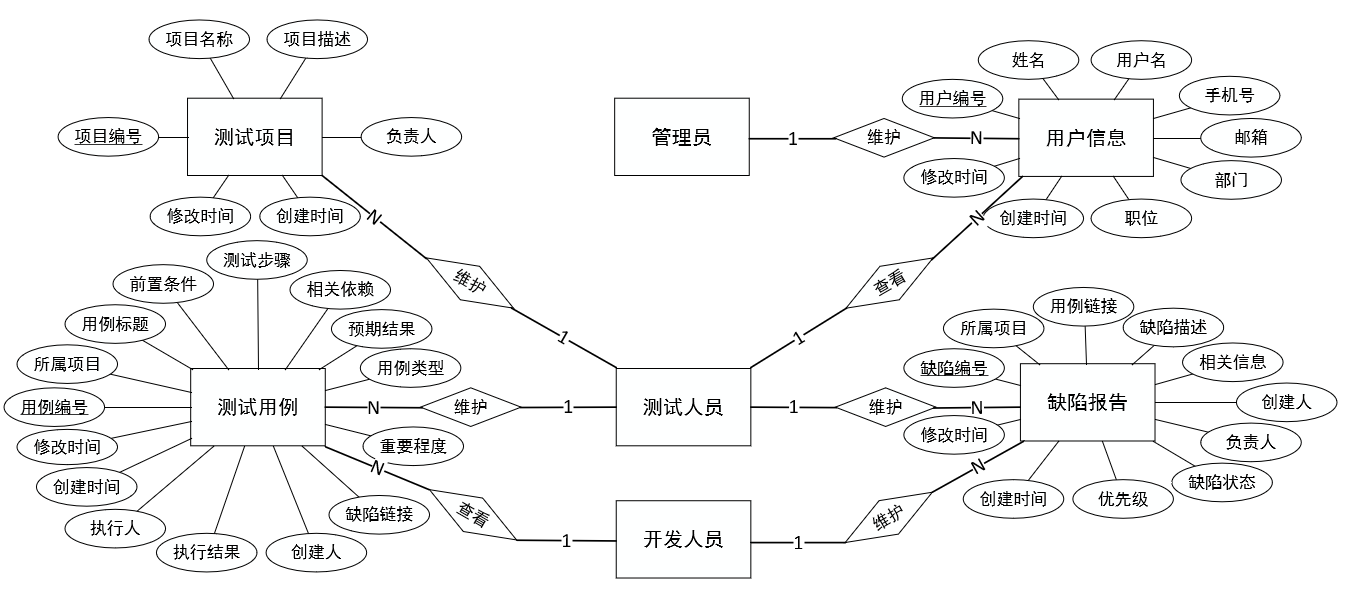
图4-4 核心功能业务流程图

## 平台数据库设计

针对于平台的数据储存需求，本设计引入了关系型数据库来对数据进行储存，根据作者本科期间了解，免费的MySQL性能优越，安装便捷，使用简单。因此在本课题中，使用MySQL数据库来实现对测试流程信息存储和管理，并对数据库进行建模。

### E-R图设计

实体联系图，简称E-R图，主要用来描述实体及其之间的关系，其是数据库概念设计中较为常用的图形设计方法之一[28]。在本课题中，使用E-R图表示各个实体之间的关系，具体设计如图4-5所示：

图4-5 平台E-R图

在本平台中，实体主要有管理员、测试人员、开发人员、用户信息、测试项目、测试用例、缺陷报告等，各个实体均有其相关属性和联系。接下来本文将详细阐述关系型数据库的存储结构。

### 数据库表设计

数据库表设计即为平台的存储结构设计[29]。本平台的所有数据表均来源于上述实体及其联系所产生的数据表，具体表格的设计如表4-1至4-4所示。

表4-1 用户信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 代码 | 名称 | 数据类型 | 是否主键 | 注释 |
| id | 用户编号 | INT | true | 主键，自动增长 |
| name | 姓名 | VARCHAR(255) | false | 非空 |
| user | 用户名 | VARCHAR(255) | false | 非空，唯一索引 |
| email | 邮箱 | VARCHAR(255) | false | 非空，唯一索引 |
| telephone | 手机号 | VARCHAR(255) | false | 非空，唯一索引 |
| department | 部门 | VARCHAR(255) | false | 非空 |
| position | 职位 | INT | false | 0代表管理员，默认值  1代表测试人员  2代表开发人员 |
| create\_time | 创建时间 | DATETIME | false |  |
| update\_time | 修改时间 | TIMESTAMP | false |  |

表4-2 测试项目表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 代码 | 名称 | 数据类型 | 是否主键 | 注释 |
| id | 项目编号 | INT | true | 主键，自动增长 |
| name | 项目名称 | VARCHAR(255) | false | 非空 |
| description | 项目描述 | TEXT | false | 允许为空 |
| owner | 负责人 | VARCHAR(255) | false | 允许为空 |
| create\_time | 创建时间 | DATETIME | false |  |
| update\_time | 修改时间 | TIMESTAMP | false |  |

表4-3 缺陷报告表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 代码 | 名称 | 数据类型 | 是否主键 | 注释 |
| id | 缺陷编号 | INT | true | 主键，自动增长 |
| case | 用例链接 | TEXT | false | 相关测试用例链接 |
| project | 所属项目 | INT | false | 非空,取测试项目id |
| description | 缺陷描述 | TEXT | false | 非空 |
| message | 相关信息 | TEXT | false | 允许为空 |
| level | 优先级 | INT | false | 0代表 P0，默认值  1代表 P1  2代表 P2  3代表 P3 |
| status | 缺陷状态 | INT | false | 0代表新提交，默认值  1代表已修复  2代表已发布  3代表重新打开 |
| founder | 创建人 | VARCHAR(255) | false | 非空 |
| operator | 负责人 | VARCHAR(255) | false | 允许为空 |
| create\_time | 创建时间 | DATETIME | false |  |
| update\_time | 修改时间 | TIMESTAMP | false |  |

表4-4 测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 代码 | 名称 | 数据类型 | 是否主键 | 注释 |
| id | 用例编号 | INT | true | 主键，自动增长 |
| project | 所属项目 | INT | false | 非空,取测试项目id |
| title | 用例标题 | TEXT | false | 非空 |
| condition | 前置条件 | TEXT | false | 允许为空 |
| step | 测试步骤 | TEXT | false | 允许为空 |
| expect | 预期结果 | TEXT | false | 允许为空 |
| dependence | 相关依赖 | TEXT | false | 允许为空 |
| type | 用例类型 | INT | false | 0代表功能用例，默认值  1代表性能用例  2代表兼容性用例  3代表UI用例 |
| level | 重要程度 | INT | false | 0代表P0，默认值  1代表P1  2代表P2  3代表P3  4代表P4 |
| bug | 缺陷链接 | TEXT | false | 相关缺陷报告链接 |

续表4-4 测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 代码 | 名称 | 数据类型 | 是否主键 | 注释 |
| result | 执行结果 | INT | false | 0代表未执行，默认值  1代表通过  2代表不通过  3代表受阻  4代表跳过 |
| status | 用例状态 | INT | false | 0代表草稿，默认值  1代表通过  2代表未通过  3代表废弃 |
| founder | 创建人 | VARCHAR(255) | false | 非空 |
| operator | 执行人 | VARCHAR(255) | false | 允许为空 |
| create\_time | 创建时间 | DATETIME | false |  |
| update\_time | 修改时间 | TIMESTAMP | false |  |

## 本章小结

本章基于此前的需求相关分析，对系统相关开发架构、功能需求与数据库设计进行了相关介绍，后续将根据概要设计结果来驱动平台的详细设计与实现，具体介绍见后文。

# 详细设计与实现

概要设计完成之后，需要对上述提出的各项内容进行详细设计并实现。在该模块中，主要对软件的界面层、业务逻辑层和核心算法层进行详细设计与实现[30]。

## 平台界面设计与实现

平台的界面设计美观、易用，直接影响到用户工作效率，下面将对平台主要模块界面进行设计与实现，平台实现示例图片将根据展示需要，进行字段展示顺序的修改。

### 用户登录功能界面设计与实现

用户登录界面需要用户输入自己的用户名、密码和职位进行登录，具体实现界面如图5-1所示。

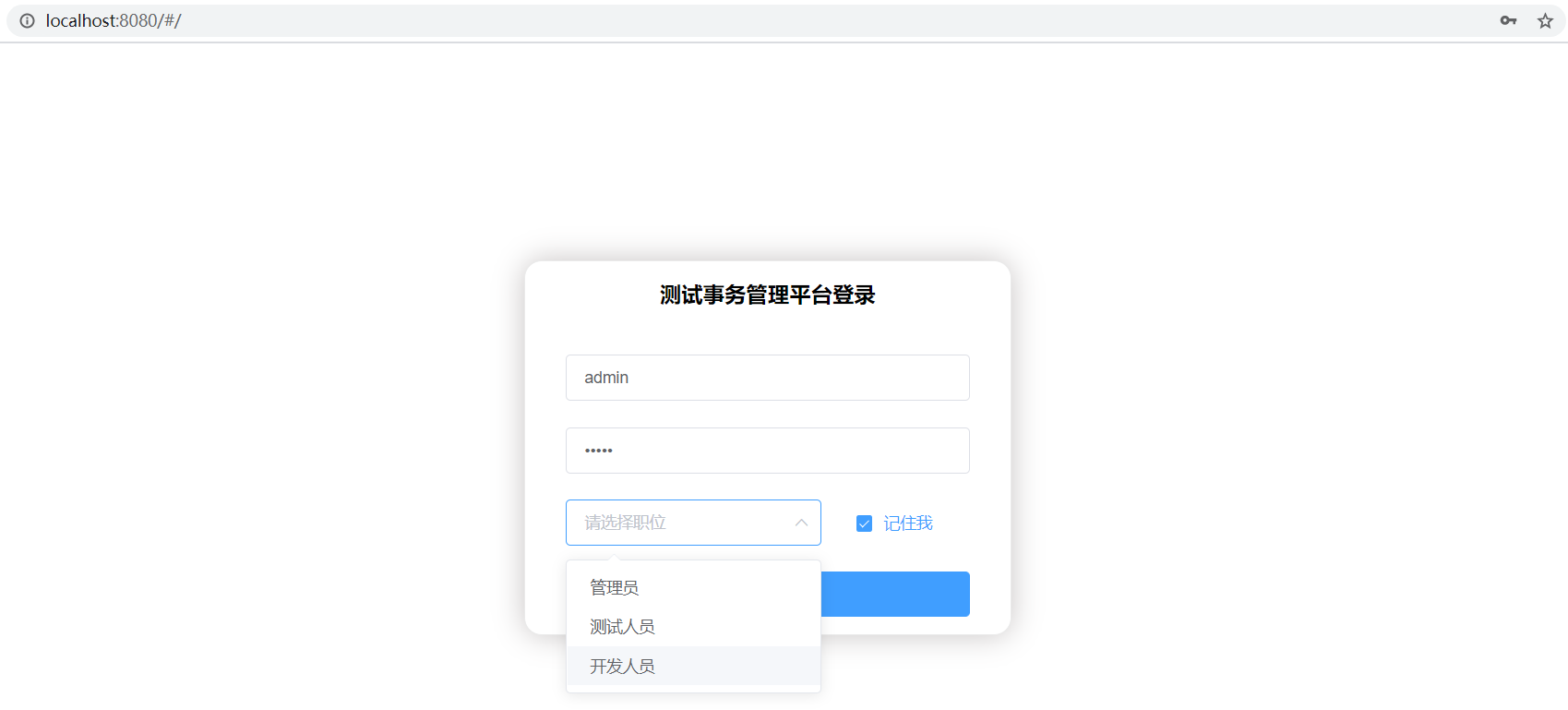


图5-1 用户登录界面

### 用户管理界面设计与实现

用户管理界面主要实现用户信息的显示，包括姓名、用户名、手机号、邮箱、部门、职位等，如图5-2所示。具体功能如下：

1. 添加、查找按钮显示在用户列表上方；
2. 其他功能显示在操作栏中，可以修改用户信息、设置用户密码；
3. 根据实际情况，删除用户采用禁用用户代替。



图5-2 用户管理界面

### 测试项目菜单设计与实现

测试项目在菜单栏中作为文件夹显示，每个文件夹下用子文件夹代表该测试项目的不同模块，如图5-3所示，具体业务规则如下：

1. 鼠标点击根文件，显示所有测试用例；
2. 鼠标点击子文件夹，显示子文件夹中测试用例；
3. 测试项目菜单可以收起至左侧；
4. 鼠标悬浮至文件夹上显示操作按钮，包括新建、编辑和删除。



图5-3 测试项目菜单界面

### 测试用例管理界面设计与实现

测试用例管理界面主要展示测试用例编号、状态、执行结果、标题、用例类型、重要程度、状态、执行人、更新时间、创建时间等信息，其他信息展示在详情处，如图5-4所示，具体业务规则如下：

1. 添加、查找、导入、导出按钮显示在用例列表上方；
2. 可在页面直接修改状态、执行结果、标题、用例类型；
3. 鼠标点击标题，可查看用例详情；
4. 悬停至标题，显示编辑、复制链接、关联缺陷报告按钮，如图5-5所示；





图5-4 测试用例管理界面



图5-5 测试用例操作界面

### 缺陷报告管理界面设计与实现

缺陷报告管理界面主要展示缺陷报告编号、标题、优先级、状态、创建人、负责人、创建时间、更新时间等信息，其他信息展示在详情处，大体如图5-6所示，业务规则与测试用例管理界面相同，在这里不多赘述。



图5-6 缺陷报告管理界面

### 统计报表界面设计与实现

统计报表界面如图5-7和5-8所示，主要实现了简洁直观的用例统计功能，目前可以根据测试用例维护人、测试用例维护人、测试用例状态和测试用例类型进行分布统计，点击报表可以查看具体数据详情。

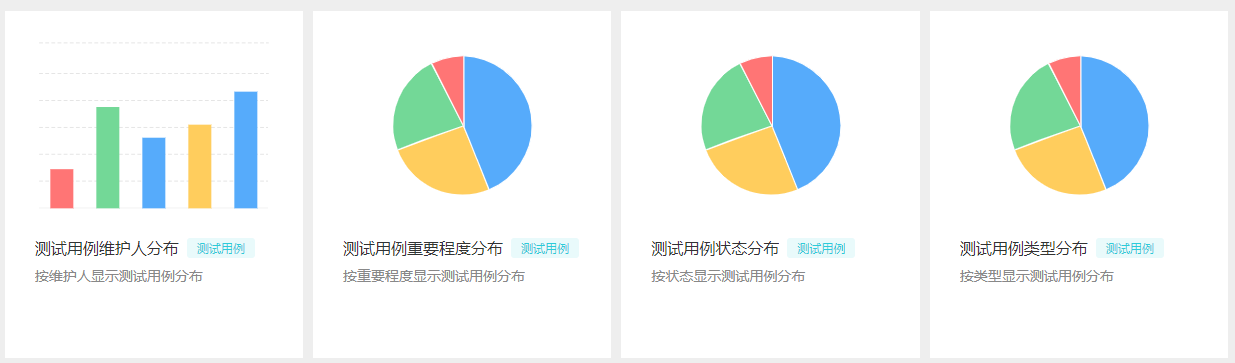


图5-7 统计报表概览图

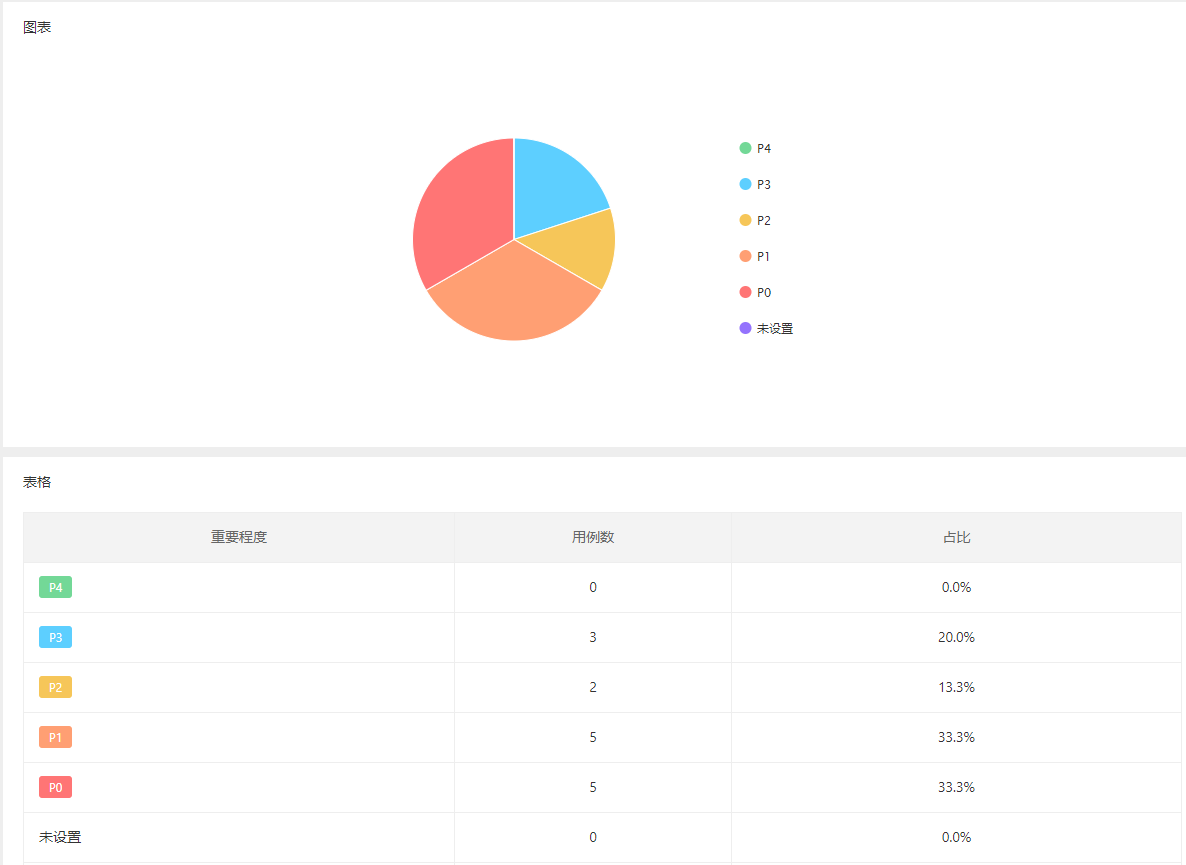


图5-8 统计报表详细图出图

## 平台业务逻辑层设计与实现

本系统中除用户登录外存在较多的业务逻辑，包括添加、修改、查询、删除、导入、导出等。受限于篇幅限制，本文主要通过用户登录、测试用例管理模块、自动化测试框架管理模块以及缺陷问题处理的业务逻辑展开叙述，缺陷问题管理不再详细描述。

### 用户登录

用户登录时序图如图5-9所示：以下简单介绍该系统登录的判断原则：

（1）账号及对应职位不存在，则不允许登录，提示“账号不存在！”

（2）账号及对应职位存在，但密码错误，则不允许登录，提示“账号与密码不匹配！”

（3）账号及对应职位存且密码正确，则跳转到用户职位对应界面。

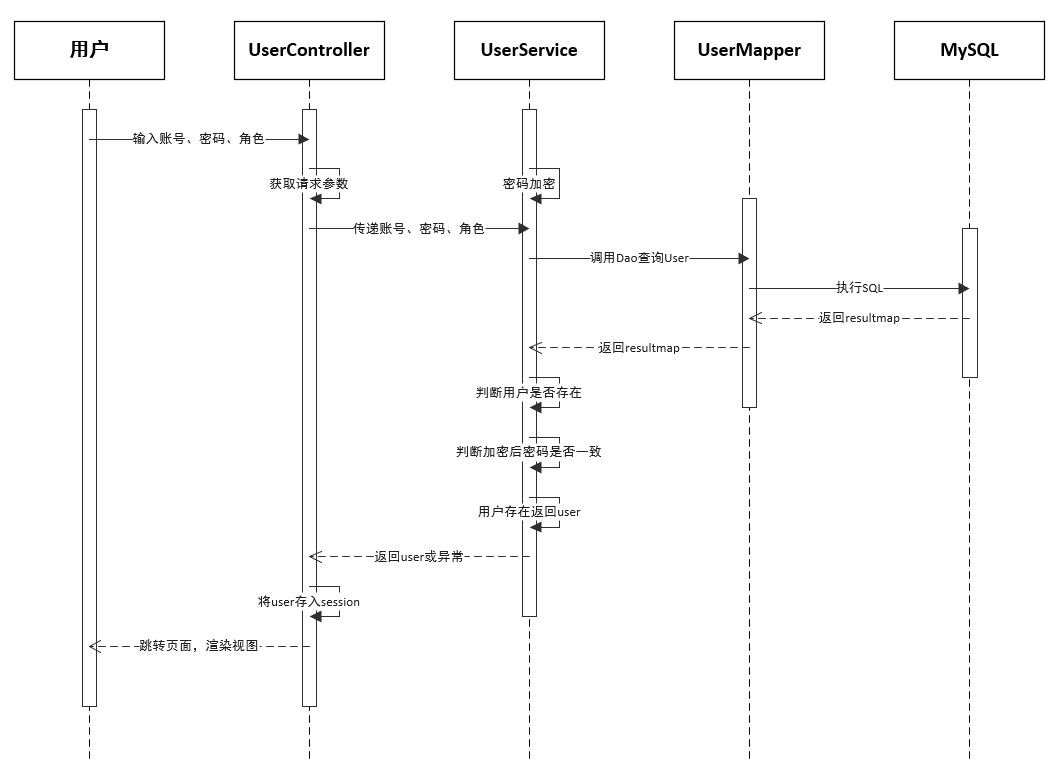


图5-9 用户登录时序图

### 测试用例管理

测试用例管理首先包括添加、修改、删除、查询基本功能，用户发送请求传递参数，后端在controller层接收后调用service访问数据库处理。具体时序图如5-10所示：

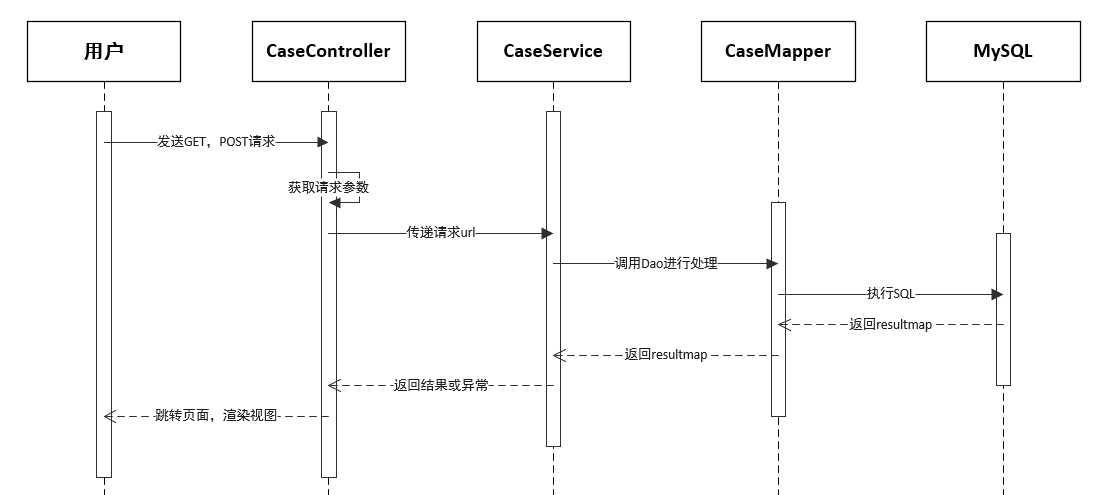


图5-10 测试用例增删改查时序图

测试用例添加界面如图5-11所示：

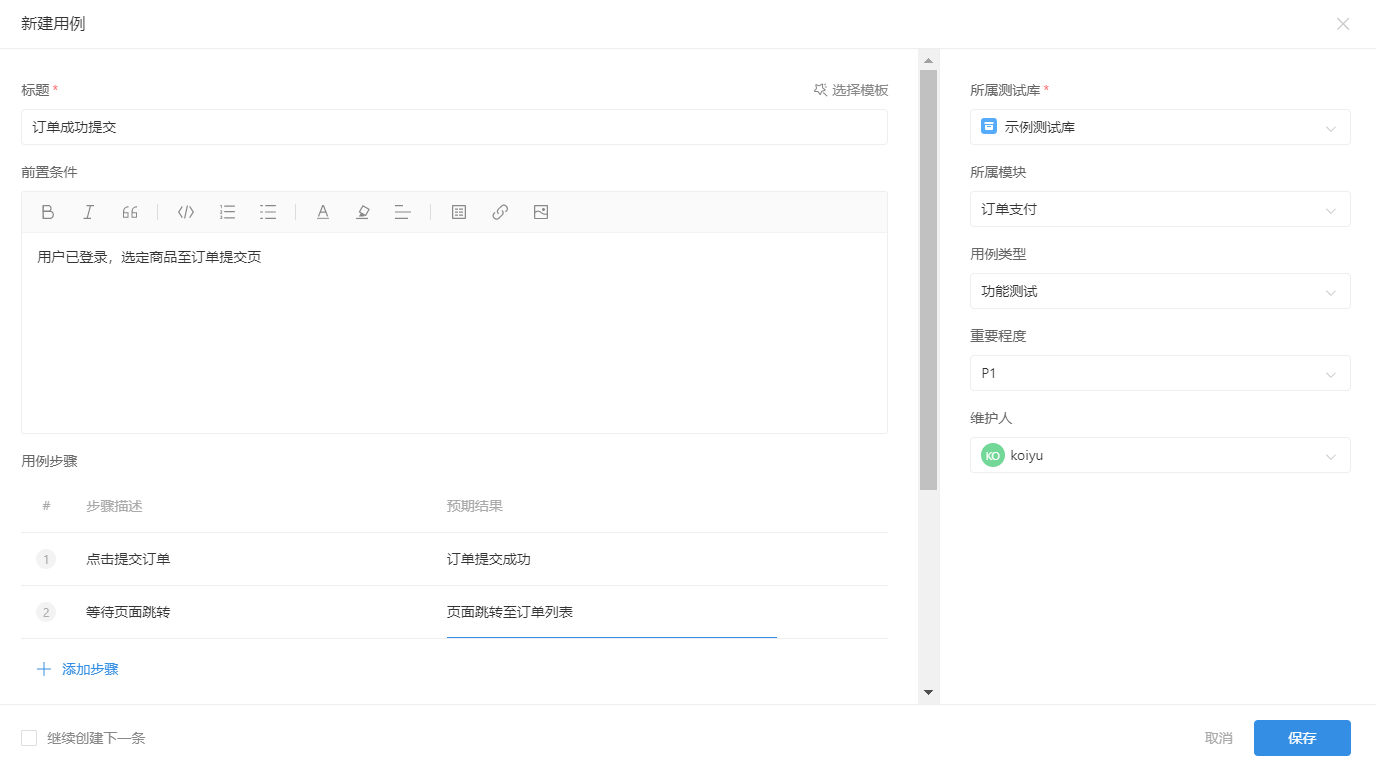


图5-11 测试用例添加图

测试用例修改界面如图5-12所示：

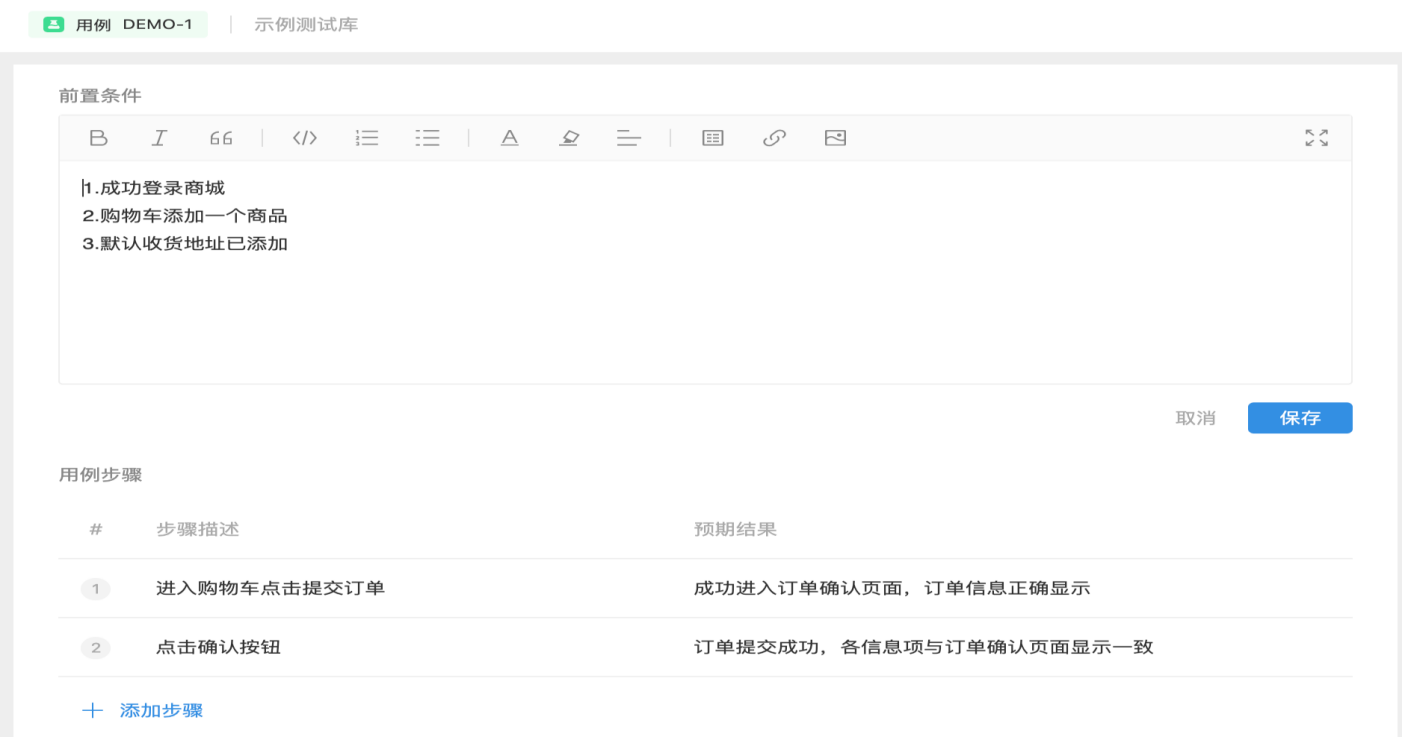


图5-12 测试用例修改图

测试用例删除界面如图5-13所示：

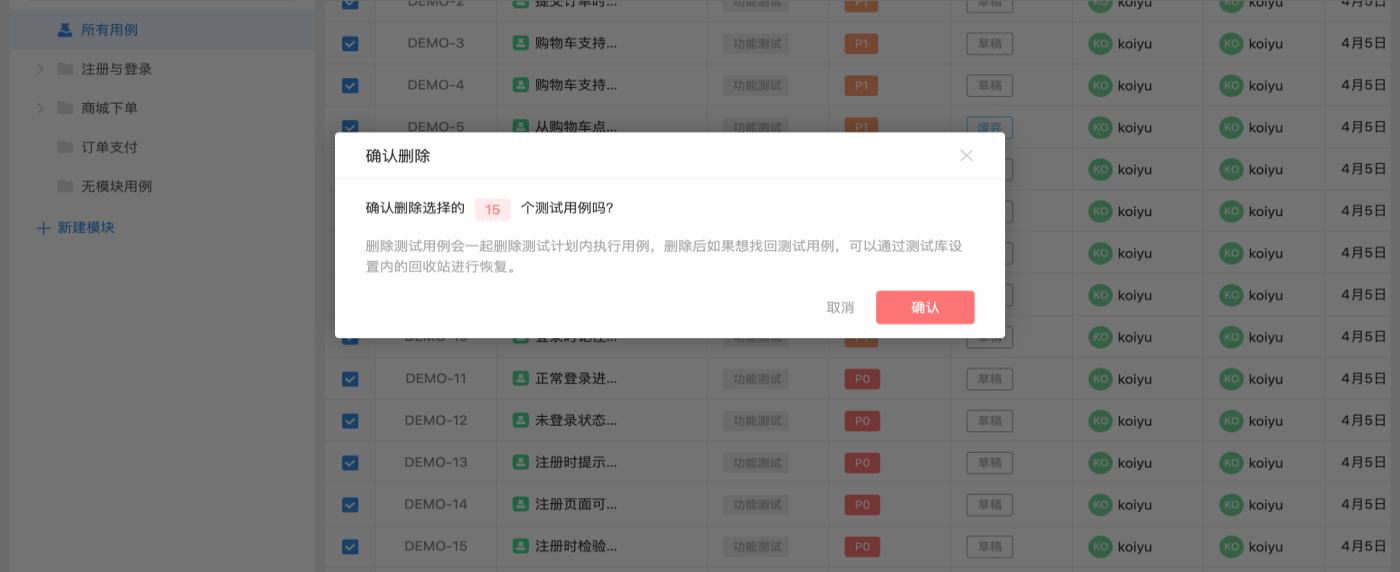


图5-13 测试用例删除图

测试用例查询界面如图5-14所示：



图5-14 测试用例查询图

除以上功能外，测试用例管理还包括导入、导出功能，使用Python封装好的ExcelWriter库进行Excel读写功能，其流程图如图5-15所示：

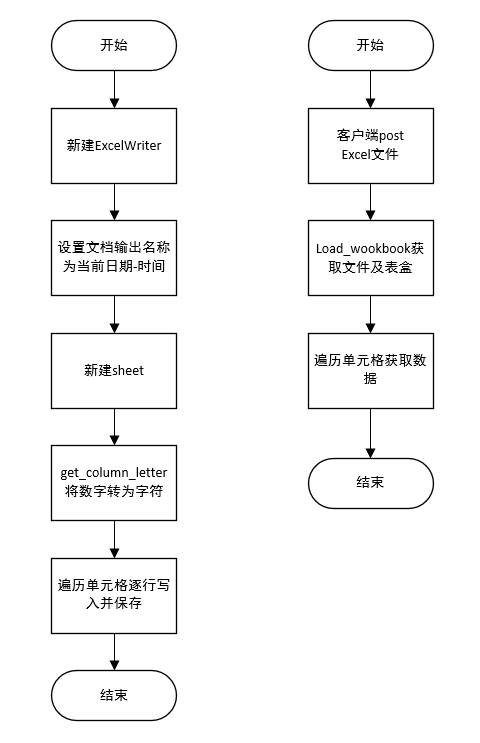


图5-15 测试用例导入（左）、导出（右）流程图

测试用例导入界面如图5-16所示：



图5-16 测试用例导入图

测试用例导出界面如图5-17所示：

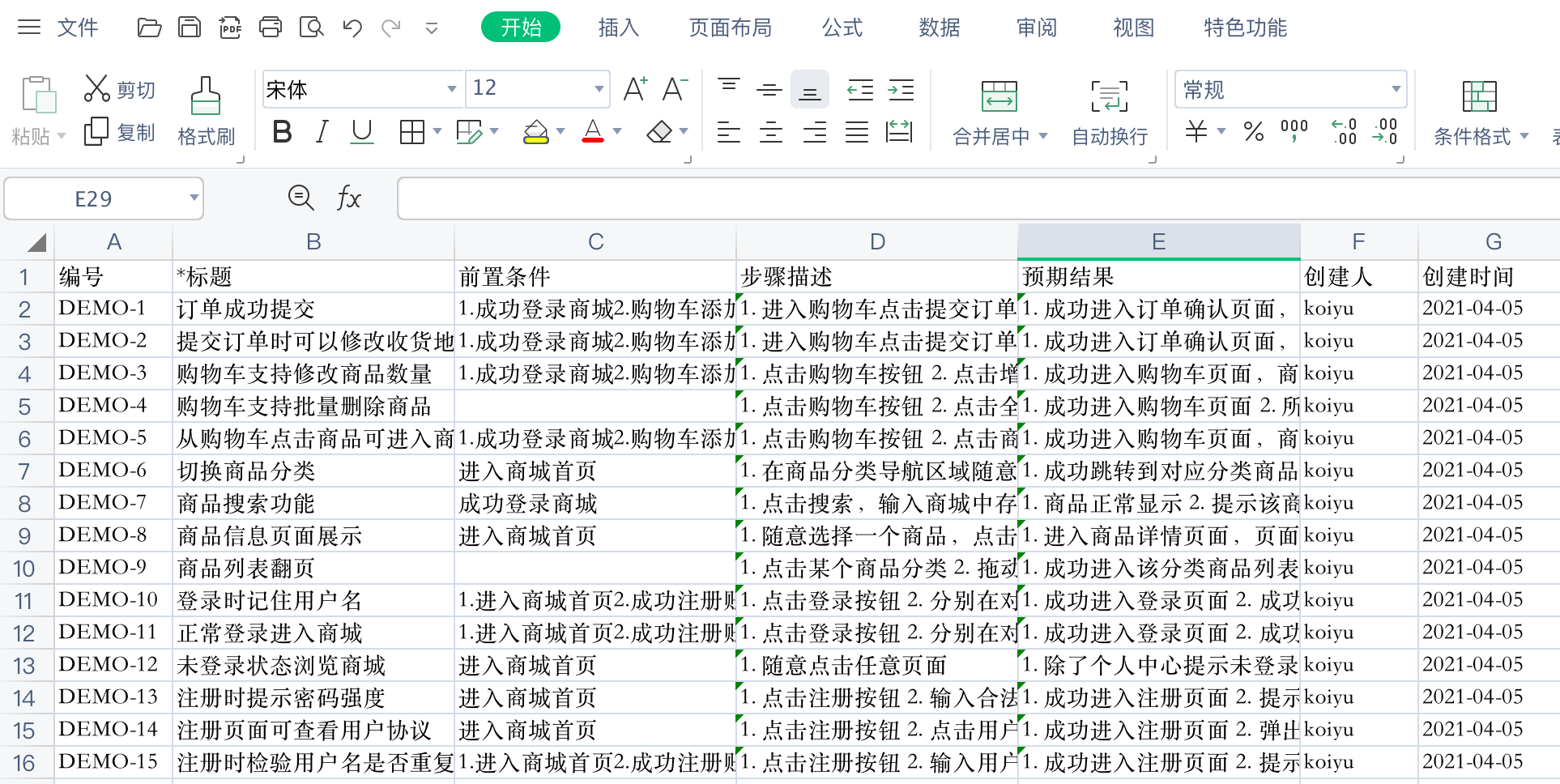


图5-17 测试用例导出图

### 自动化测试框架管理

自动化测试框架管理主要实现代码管理、分支覆盖率统计和自动化测试用例执行报表生成，具体内容如下：

1. 代码管理目前复用GitHub界面，点击代码管理，跳转至GitHub网址，如图5-18所示；
2. 覆盖率统计当前报表为作者代码三个分支插入JaCoCo插件后得出的结果，如图5-19所示；
3. 自动化测试用例执行报告目前主要通过手工执行单元测试用例，数据来源暂时为作者构造，如图5-20所示。

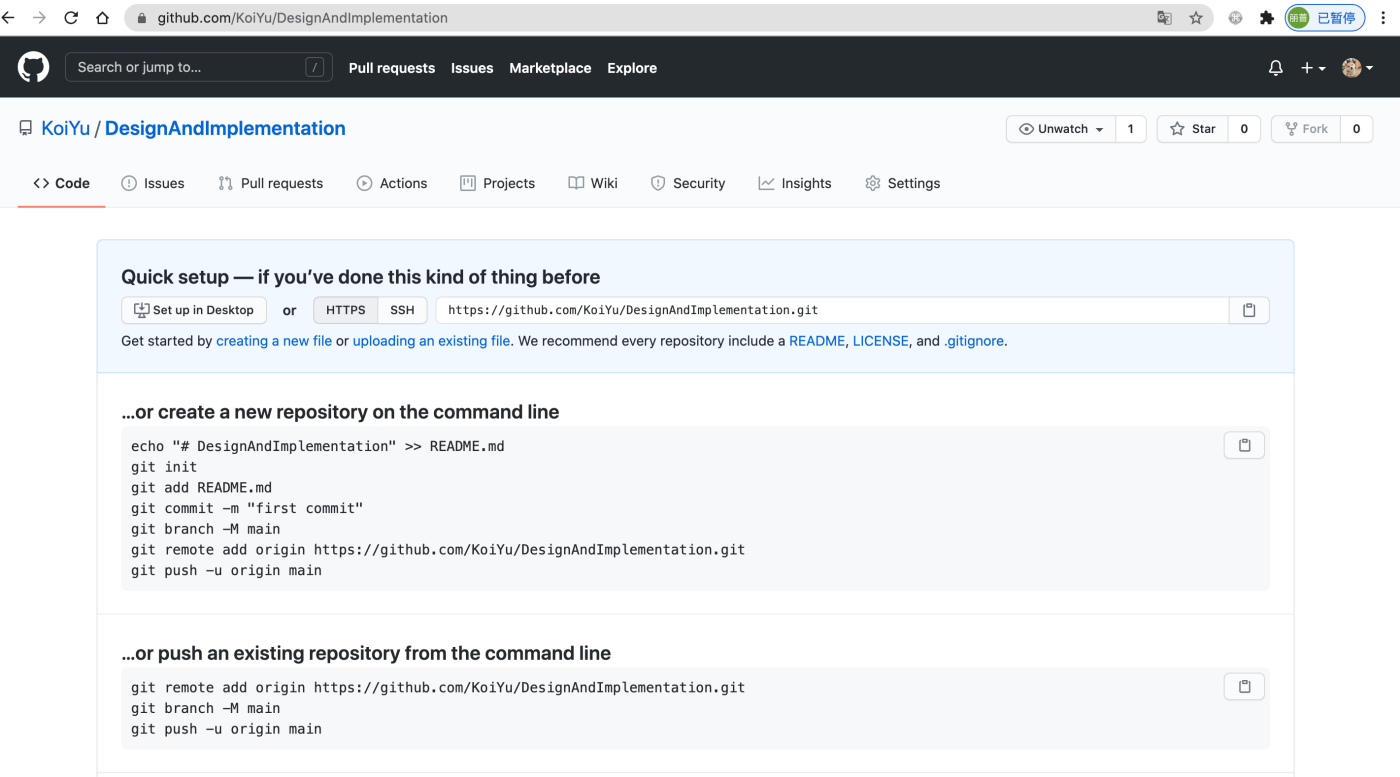


图5-18 代码管理界面图

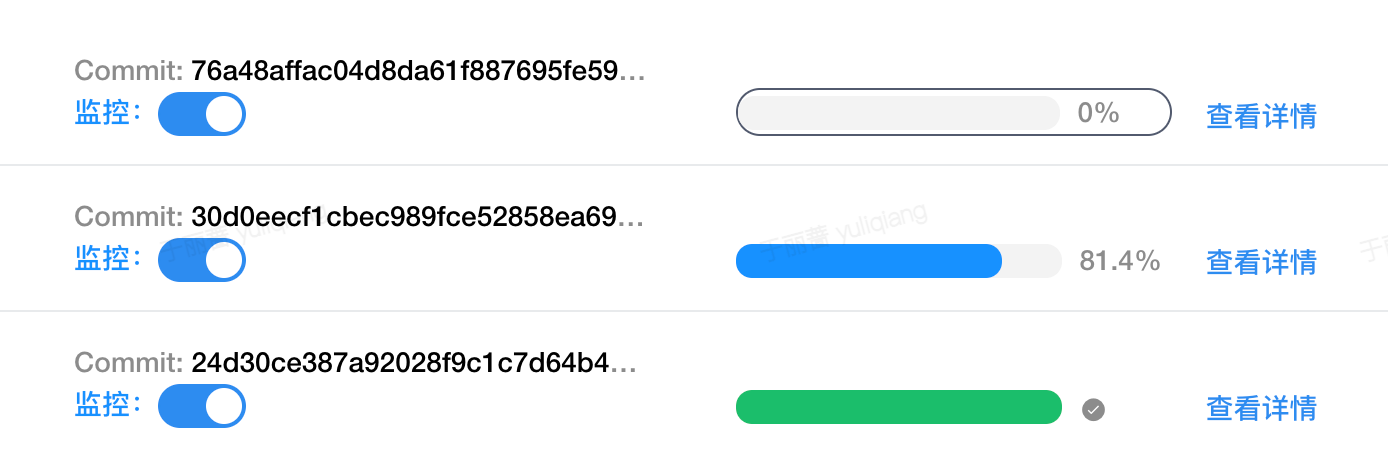


图5-19 覆盖率统计界面图



图5-20 自动化测试报告界面图

### 测试问题处理

平台实现测试用例与缺陷报告的关联，在此介绍一下缺陷问题处理流程，流程图如图5-21：

1. 测试人员执行测试用例时发现问题，新建缺陷报告，并将测试用例链接复制到缺陷报告中，缺陷报告建立后将链接复制到测试用例中，此时缺陷报告状态为新提交；
2. 开发人员查收缺陷报告，并根据测试用例链接查看复现步骤，以便排查、修复问题，确认问题已解决后将缺陷报告状态置为已修复；
3. 测试人员会对已修复缺陷问题进行回归验证，验证通过则将缺陷问题置为已发布，意为已解决，至此缺陷问题处理流程结束；验证不通过需将缺陷状态置为重新打开，待开发人员重复流程（2）。

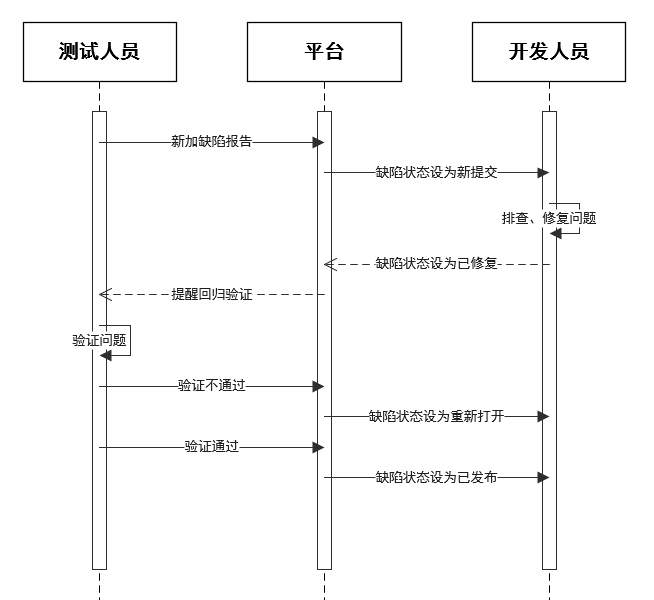


图5-21 缺陷问题处理流程图

## 平台核心算法层设计与实现

本平台核心问题在于测试用例的自动生成，当前可根据对需求文档进行分词后，根据已有测试用例进行推荐生成，后续将实现根据基础测试理论进行自动生成。

### 分词算法

分词算法采用Python中文分词软件中的“结巴”分词，直接导入即可，具体实现如图5-22所示：

import os  
import jieba  
class Segmentation:  
   files = []  
   def \_\_init\_\_(self):  
     self.files = os.listdir("contents")  
   def segment(self):  
     count = 0  
     for file in self.files:  
       count+=1  
       if count == 2:  
          break  
       f = open("contents" + "/" + file, encoding='utf-8')  
       content = f.readline()  
       seg\_list = jieba.cut(content)  # 默认是精确模式  
       print(", ".join(seg\_list))  
if \_\_name\_\_ == '\_\_main\_\_':  
    segmentation = Segmentation()  
    segmentation.segment()

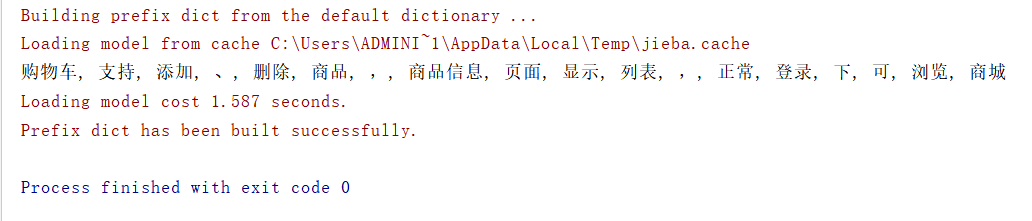


图5-22 分词结果图

### 推荐算法

在进行生成测试用例推荐文本之前，需要使用Apriori算法对需求文本进行数据挖掘，得到关联规则，并存入数据库[31]。实现结果如图5-23所示：

public CommonReturnType apriori(){  
   List<Case> cases = caseMapper.list();  
   List<ProductDoc> prds = productDocMapper.list();  
   Map<Case,Integer> support = getSupport(cases,prds,2);  
   //现在这个Map里存放的即为L1  
   List<Rule>L1 = new ArrayList<>();  
   for (Case case:support.keySet()){  
      Rule rule = new Rule();  
      rule.setAtts(case.getName());  
      int occ=support.get(case);  
      if (occ==0)continue;  
      rule.setOccurence(occ);  
      rule.setSize(1);  
      L1.add(rule);  
      ruleMapper.insert(rule);  
   }  
   dfs(L1,cases,prds,2);  
   return CommonReturnType.create("sucess");  
}

private void dfs(List<Rule>lk,List<Case>cases,List<ProductDoc>prds,int miniSupport){

ArrayList<Rule> lk1 = getLkPlus1(lk,cases,prds,miniSupport);

if (lk1==null)return;

for (Rule rule:lk1){

ruleMapper.insert(rule);

}

dfs(lk1,cases,prds,miniSupport);

}

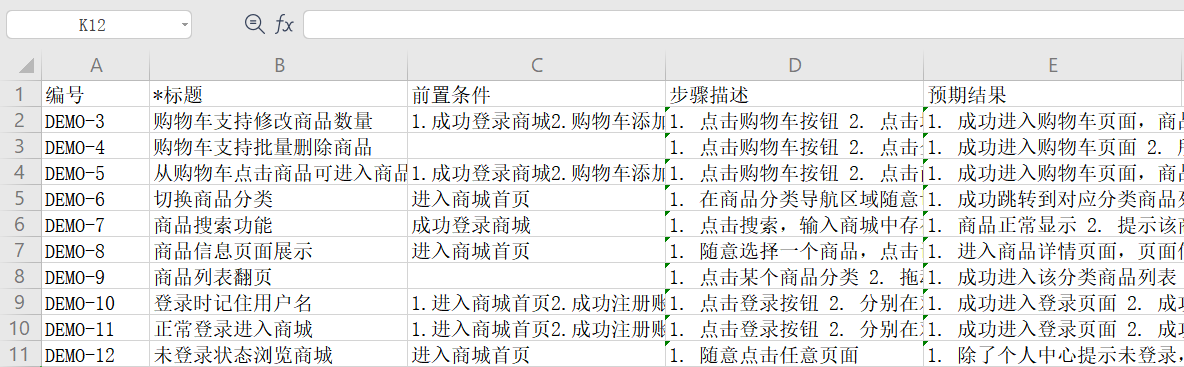


图5-23 测试用例推荐结果图

## 本章小结

本章主要从一些界面展示、功能逻辑、核心算法的详细设计入手，介绍了系统中一些关键功能与核心算法技术的实现过程。到本章为止，平台的全部业务基本实现，接下来，本课题将会对平台开展功能相关测试。

# 测试

## 测试目的及方法

在完成平台的搭建之后，需要验证平台各项功能的可用性等，保证用户征程使用，且应当符合软件工程准则，有良好的使用体验。

鉴于平台性质，测试部分主要采用功能测试。功能测试主要根据产品特性、用户需求来判断平台实现功能是否满足设计需求，主要侧重于业务功能和业务规则。

## 测试工具及环境

本章对平台的各个功能模块进行了功能测试，以验证本文设计实现是否符合用户需求，具体测试环境如下：

(1)硬件环境:

●服务器：本地主机一台

●CPU：[Intel i7 /](https://detail.zol.com.cn/notebook_index/subcate16_list_p28632_1.html) 2.6GHZ

●RAM：16G

●硬盘：512G

(2)软件环境:

●操作系统：Windows10

●测试工具：IntelliJ IDEA，Google Chrome

## 功能测试

本文功能测试主要包括 UI 测试，以及各个功能模块的测试。

### UI 测试

系统通过 UI 界面向用户提供可视化操作，因此需要通过UI 测试来检验用户与系统之间的交互情况。下面主要对菜单、窗口和图形内容进行测试。

（1）菜单

针对于菜单的 UI 测试主要包含菜单文字能够清晰描述对应功能，可以使用基础操作快捷键或热键，如：鼠标右键点击“更多”，将展示操作列表。

1. 窗口

针对于窗口测试，窗口大小合适，不同分辨率设置下可以正常显示；窗口布局合理，放大缩小后内容可以正常显示；窗口界面美观，图标与功能一一对应，文字描述准确且无错别字。

1. 图形

图形测试内容包括图片、边框、字体、颜色、按钮等，确保色调和谐，风格一致，提高用户体验性。

### 用户管理模块功能测试

用户管理模块主要包括登录，用户添加、修改、删除、查询、权限分配等几项功能，本小节主要对于登录功能进行了响应测试。

登录功能根据用户被分配的职位验证登录信息并跳转至相应登录界面，整体测试情况如下表 6-1 所示，测试用例值如下：

测试账号信息：

账号：admin；密码：admin；已分配职位：管理员、测试人员。

表 6-1 用户登录功能测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 测试输入 | 预期输出 | 实际输出 | 是否通过 |
| A01 | 账号：admin  密码：admin  职位：管理员 | 登录成功  显示管理员界面 | 登录成功  显示管理员界面 | 是 |
| A02 | 账号：admin  密码：admin  职位：测试人员 | 登录成功  显示测试人员界面 | 登录成功  显示测试人员界面 | 是 |
| A03 | 账号：admin  密码：admin  职位：开发人员 | 登录失败  提示“账号不存在！” | 登录失败  提示“账号不存在！” | 是 |
| A04 | 账号：admin  密码：admin  职位：开发人员 | 登录失败  提示“账号不存在！” | 登录失败  提示“账号不存在！” | 是 |
| A05 | 账号：admin123  密码：admin  职位：管理员 | 登录失败  提示“账号不存在！” | 登录失败  提示“账号不存在！” | 是 |
| A06 | 账号：admin  密码：admin123  职位：管理员 | 登录失败  提示“账号与密码不匹配！” | 登录失败  提示“账号与密码不匹配！” | 是 |

### 测试用例管理模块功能测试

测试用例管理模块除基础功能外，主要包括测试用例的自动生成功能，其余基础功能与缺陷报告管理模块类似，将在缺陷报告管理模块进行测试。本小节主要对于测试用例的两种自动生成方法进行了响应测试，整体测试情况如下表 6-2所示，测试用例值如下：

1. 需求文本1：购物车支持添加、删除商品，商品信息页面显示列表，正常登陆下可浏览商城。
2. 需求文本2：商品支持购买1~10个。

表 6-2 自动生成测试用例功能测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 测试输入 | 预期输出 | 实际输出 | 是否通过 |
| B01 | 上传需求文本1  点击推荐生成 | 生成成功  提示“生成成功，请下载！” | 生成成功  提示“生成成功，请下载！” | 是 |
| B02 | 点击下载A01生成的测试用例表格表格 | 下载成功  显示相关测试用例  如图5-19所示 | 下载成功  显示相关测试用例  如图5-19所示 | 是 |
| B03 | 点击导入，上传生成的测试用例表格 | 上传成功  平台显示上传文件中的测试用例 | 上传成功  平台显示上传文件中的测试用例 | 是 |
| B04 | 上传需求文本2  点击一键生成 | 生成成功  用例标题：“商品支持购买1~10个：边界值0、1、2、9、10、11” | 生成成功  用例标题：“商品支持购买1~10个：边界值0、1、2、9、10、11” | 是 |
| B05 | 上传需求文本1  点击一键生成 | 生成失败  提示“生成失败，未提取到边界值！” | 生成失败  提示“生成失败，未提取到边界值！” | 是 |

### 缺陷报告管理模块功能测试

缺陷报告管理模块主要包括缺陷报告的添加、修改、删除、查询、导入、导出、关联测试用例等几项功能，用户信息和测试用例管理与此类似，不再进行介绍。

本小节对于缺陷报告每一项功能、测试项目及用例等相关功能都进行了响应测试，整体测试情况如下表 6-3所示，测试用例值如下：

测试项目名称：注册与登录

用例模块名称：注册、登录

测试用例标题：注册页面可查看用户协议（表6-3中测试用例均指此测试用例）

缺陷报告标题：用户免责声明文案有错别字（表6-3中缺陷报告均指此缺陷报告）

表 6-3 缺陷报告管理相关功能测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 测试输入 | 预期输出 | 实际输出 | 是否通过 |
| C01 | 点击新建测试项目  输入项目名称“注册与登录123”  点击回车键 | 新建成功  显示“注册与登录123”文件夹 | 新建成功  显示“注册与登录123”文件夹 | 是 |
| C02 | 点击测试项目“注册与登录123”---“更多”---“编辑”按钮  修改项目名称“注册与登录” | 修改成功  显示“注册与登录”文件夹 | 修改成功  显示“注册与登录”文件夹 | 是 |
| C03 | 点击测试项目“注册与登录123”---“更多”---“新建”按钮  输入用例模块名称“注册”  点击回车键 | 新建成功  “注册与登录”文件夹下显示“注册”文件夹 | 新建成功  “注册与登录”文件夹下显示“注册”文件夹 | 是 |
| C04 | 点击测试项目“注册与登录123”---“更多”---“新建”按钮  输入用例模块名称“登录”  点击回车键 | 新建成功  “注册与登录”文件夹下显示“登录”文件夹 | 新建成功  “注册与登录”文件夹下显示“登录”文件夹 | 是 |
| C05 | 点击用例模块“登录”---“更多”---“删除”按钮  弹窗点击确认删除 | 删除成功  “注册与登录”文件夹下不再显示“登录”文件夹 | 删除成功  “注册与登录”文件夹下不再显示“登录”文件夹 | 是 |
| C06 | 测试用例管理界面进入“注册”文件夹  点击“新建用例”  填写用例标题“注册页面可查看用户协议” | 新建成功  自动生成用例编号和创建时间  创建人为登录用户 | 新建成功  自动生成用例编号和创建时间  创建人为登录用户 | 是 |
| C07 | 缺陷报告管理界面进入“注册”文件夹  点击“新建缺陷”  填写缺陷描述“文案有错别字”和负责人 | 新建成功  自动生成缺陷编号和创建时间  创建人为登录用户 | 新建成功  自动生成缺陷编号和创建时间  创建人为登录用户 | 是 |

续表 6-3 缺陷报告管理相关功能测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 测试输入 | 预期输出 | 实际输出 | 是否通过 |
| C08 | 点击修改缺陷描述“文案有错别字”为“用户免责声明文案有错别字” | 修改成功  自动生成修改时间 | 修改成功  自动生成修改时间 | 是 |
| C09 | 点击复制缺陷报告链接  在浏览器地址栏中查询 | 复制成功  浏览器跳转至缺陷报告页面 | 复制成功  浏览器跳转至缺陷报告页面 | 是 |
| C10 | 点击修改测试用例的缺陷报告字段为复制的缺陷报告链接 | 修改成功  自动生成修改时间和维护人  点击链接可跳转至缺陷报告页面 | 修改成功  自动生成修改时间和维护人  点击链接可跳转至缺陷报告页面 | 是 |
| C11 | 点击复制测试用例链接  在浏览器地址栏中查询 | 复制成功  浏览器跳转至测试用例页面 | 复制成功  浏览器跳转至测试用例页面 | 是 |
| C12 | 点击修改缺陷报告的测试用例字段为复制的测试用例链接 | 修改成功  自动生成修改时间  点击链接可跳转至测试用例页面 | 修改成功  自动生成修改时间  点击链接可跳转至测试用例页面 | 是 |
| C13 | 删除测试用例“用户免责声明文案有错别字”  弹窗点击确认 | 删除成功  不再显示测试用例“用户免责声明文案有错别字” | 删除成功  不再显示测试用例“用户免责声明文案有错别字” | 是 |
| C14 | 删除后点击缺陷报告中的测试用例链接 | 跳转至测试用例界面  提示“此条用例已删除” | 跳转至测试用例界面  提示“此条用例已删除” | 是 |

### 自动化测试框架管理模块功能测试

自动化测试用例管理模块测试主要包括代码管理、代码覆盖率统计功能，除此之外，包括测试用例管理模块中的统计测试用例报表功能，本小节对于三项功能都进行了响应测试，整体测试情况如下表 6-4所示，测试用例值如下：

GitHub项目使用网络开源项目：

代码分支1，未使用JaCoCo插件；代码分支2和3，使用JaCoCo插件

测试用例按重要程度设置数量：

P0：5个；P1：5个；P2：2个；P3:3个；P4：0个

表 6-4 测试用例管理模块功能测试结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例编号 | 测试输入 | 预期输出 | 实际输出 | 是否通过 |
| D01 | 点击代码管理 | 跳转GitHub 页面 | 跳转GitHub 页面 | 是 |
| D02 | 点击覆盖率统计 | 跳转统计页面  显示代码覆盖率  如图5-17 | 跳转统计页面  显示代码覆盖率  如图5-17 | 是 |
| D03 | 点击测试用例重要程度统计报表 | 显示报表统计  数量统计正确  如图5-8 | 显示报表统计  数量统计正确  如图5-8 | 是 |

## 本章小结

本章主要对平台进行了功能测试，执行以上测试用例后，预期输出与实际输出结果相符，因此测试结果均为通过。至此为止整个平台开发完成，可以正常投入使用。最后的结论部分将会对整个系统进行全面分析。

千万不要删除行尾的分节符，此行不会被打印。“结论”以前的所有正文内容都要编写在此行之前。

结论

本平台根据软件工程开发思想完成了全部的开发任务，开发搭建了测试事务管理平台，为测试人员和开发人员的工作流程和项目闭环提供了很好的便利。

开发流程上，平台主要基于软件工程中常用的增量模型进行逐步构建完成整个平台，增量1—用户管理模块开发，增量2—测试用例管理模块基础框架开发，增量3—测试用例关联测试报告功能开发即缺陷报告管理模块开发，增量4—自动化测试框架管理模块基础框架开发，增量5—测试用例自动生成功能开发。

功能上，平台创新性地实现自动生成测试用例，提高测试效率；并设计了基于测试点集的推荐算法，以内容过滤为核心思想，提供可用测试用例供测试人员选择。同时，平台实现了项目管理、测试用例管理、缺陷报告管理全流程跟进，便于测试人员和开发人员敏捷、高效合作。

架构上，平台以 B/S 三层架构作为基础，界面表现层为用户提供了可视化的操作界面；业务逻辑层除了实现各个业务模块的基本功能需求外，根据设计模式六大原则，尽量降低了各业务代码的耦合性；数据层主要为用户操作数据提供服务，包括的读写等更新操作。三者各司其职，完成了平台的底层架构，提高整个平台的灵活性和易用性。

综上，本平台采用了规范的开发方法和标准的设计原则，保证平台开发过程的完整性、正确性和可度量性；经过功能测试验证可得：所开发的平台在业务内容和响应性能上能够满足实际应用需求；所采用的三层架构有利于后续代码维护与软件升级。

平台的后续完善将从以下方面展开：

（1）进一步优化平台界面，提高用户体验。

（2）根据实际使用情况进行已有功能的进一步细化，符合实际开发流程，提高平台易用性。

（3）进一步完善测试用例自动生成的准确性与便捷性，尽可能精准的提供可供选择的推荐内容，且尽可能直接的生成准确可用的测试用例。

参考文献

赫彦文.基于智能优化算法的MC/DC测试用例生成研究[D].北华航天工业学院,2021.

周子闻.自动化测试中GUI目标的模式识别方法研究[D].北华航天工业学院,2021.

陈颖超.[基于Web的测试用例管理及生成系统](https://kns.cnki.net/kcms/detail/detail.aspx?filename=1017302343.nh&dbname=cdmdtotal&dbcode=cdfd&v=" \t "https://pmlc.cnki.net/user/_blank)[D].西安电子科技大学,2017.

傅兵.软件测试技术现状与发展趋势研究[J].电脑编程技巧与维护,2016(02).

张涛.软件技术基础实验教程[M].西北工业大学出版社,2015.01:111.

陈姗姗. 基于SOA的广播综合业务协同管理平台的设计与实现[D].南京邮电大学,2020.

李香菊,孙丽,谢修娟,等.软件工程课程设计教程[M].北京邮电大学出版社,2016:72.

冯灵霞,邵开丽,张亚娟,等.软件测试技术[M].西安电子科技大学出版,2017:14.

王宇成.基于Simhash和层次聚类算法的网页去重技术研究[D].南京邮电大学,2019.

唐聃.自然语言处理[M].电子工业出版社,2018.

余方兴.加权关联规则优化算法研究[D].华中师范大学,2014.

Agrawal R,Srikant R.Fast Algorithms for Mining Association Rules in Large Databases[C].Proceedings of the 20th International Conference on Very Large Data Bases.IEEE,1994.

陈涛,叶荣华.基于Spring Boot和MongoDB的数据持久化框架研究[J].cnki,2016.

Smith,Josh.WPF Apps with the Model-View-ViewModel Design Pattern[J].MSDN Magazine,2009.

Oracle.MySQL参考手册[EB/OL].https://www.mysql.com/,2021-5-5.

刘广峰,黄霞.计算机基础教程[M].华中科技大学出版社,2016:116.

祁薇,杨健,王辉.基于软件生命周期的计算机教学设计创新案例[J].软件工程,2016,19(03):58-60.

邓飞.基于精益创业的新产品开发迭代路径研究[D].重庆邮电大学,2018.

王铮. HBYD公司库存管理改善研究[D].河北工业大学,2016.

梁智欣.广东国税征管辅助信息系统分析与设计[D].华中科技大学,2012.

张峻巍.基于微服务的天文数据检索与学习平台开发[D].东北林业大学,2020.

张玉洁.拖拽式数据可视化工具的设计与实现[D].西安电子科技大学,2020.

张辉,贲可荣,马喆.面向服务的遗留系统再工程过程[J].海军工程大学学报,2009,21(05):57-62.

袁晓平.基于物联网技术的实验室设备管理系统[D].西北师范大学,2013.

谢树仁,邓凯成,喻琨,陈政.基于Python框架的农业信息共享平台研究——以湖南省衡阳市为例[J].经济师,2021(05):37-39.

王洪岩.基于SilverLight和WebService技术的软件测试用例库管理系统的设计与实现[D].东北师范大学,2014.

许崇华.智慧养老信息服务平台的研究与实现[D].广东工业大学,2016.

杨棠勋. BD健身俱乐部会员信息系统设计与实现[D].电子科技大学,2014.

Hallin Peter F, Ussery David W.CBS Genome Atlas Database: a dynamic storage for bioinformatic results and sequence data[J].Bioinformatics,2004(18):18.

霍士伟,刘许刚,张博刚.通用综合评价系统的设计与实现[J].电信快报,2017(09):16-18+23.

王乙超.基于网络游记的旅游景点推荐系统[D].湖南师范大学,2020.

致谢

2017年8月，我收到了东北林业大学的录取通知，自此，带着紧张和不安，我以弱小启程。四年时光转瞬即逝，站在此刻回望，我想告诉四年前的自己：不要怕，这一路繁花似锦，承蒙照顾，满心欢喜，直至此刻，也满怀感激。

首先感谢信息学院的老师们，感谢辅导员张音老师关心照顾，感谢导师李莉老师的尽心指导，感谢专业课老师们的授业解惑，为我提供了解决困难莫大的勇气和帮助。其次想感谢我的家人，感谢他们对我十几年求学生涯的陪伴与关爱、支持与鼓励，替我承担家庭的压力，为我提供了舒适的学习环境，我会用更优异的成绩回报他们。还要感谢我的同学和朋友们，一千多个嬉笑打闹的日夜，是他们学业上的有问必答，生活中的热心帮助，使我更加自信开朗。最后，感谢我实习期间的导师和同事们，此次毕业设计从我的实习经历中诞生，在我的实习日常中汲取他们的建议不断完善，最终于我的实习结束后圆满成功。

从初稿设计到成功运行，回想起独立开发的过程中，陪伴我的是软件工程导论、数据库设计原理、Java 程序设计、Web 开发技术、系统分析与设计、软件质量保证与测试，原来不知不觉中，已满载而归。

我以弱小启程，承蒙照顾，满心欢喜，满怀感激，满载而归。故事从夏天开始，就在夏天告别，也祝一路同行的老师、同学、朋友万事胜意、前程似锦。