<在线论坛>

测试计划

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <13/09/2024> | <1.0> | <初步完成此次项目的测试计划> | <苏沛泽> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[1. 简介](#1.     Introduction)

[1.1 目的](#1.1     Purpose)

[1.2 背景](#1.2     Background)

[1.3 范围](#1.3     Scope)

[1.4 项目核实](#1.4               Project Identification)

[2. 测试需求](#2.     Requirements for Test)

[3. 测试策略](#3.     Test Strategy)

[3.1 测试类型](#3.1               Testing Types)

[3.1.1 数据和数据库完整性测试](#3.1.1     Data and Database Integrity Testing)

[3.1.2 功能测试](#3.1.2     Function Testing)

[3.1.3 业务周期测试](#3.1.3     Business Cycle Testing)

[3.1.4 用户界面测试](#3.1.4     User Interface Testing)

[3.1.5 性能评测](#3.1.5     Performance Profiling)

[3.1.6 负载测试](#3.1.6     Load Testing)

[3.1.7 强度测试](#3.1.7     Stress Testing)

[3.1.8 容量测试](#3.1.8     Volume Testing)

[3.1.9 安全性和访问控制测试](#3.1.9     Security and Access Control Testing)

[3.1.10 故障转移和恢复测试](#3.1.10     Failover and Recovery Testing)

[3.1.11 配置测试](#3.1.11     Configuration Testing)

[3.1.12 安装测试](#3.1.12     Installation Testing)

[3.2 工具](#3.2     Tools)

[4. 资源](#4.     Resources)

[4.1 角色](#4.1     Workers)

[4.2 系统](#4.2     System)

[5. 项目里程碑](#5.     Project Milestones)

[6. 可交付工件](#6.     Deliverables)

[6.1 测试模型](#6.1     Test Model)

[6.2 测试记录](#6.2      Test Logs)

[6.3 缺陷报告](#6.3     Defect Reports)

[7. 附录 A： 项目任务](#7.     Appendix A: Project Tasks)

测试计划

# 1.     简介

## 1.1     目的

<在线论坛> 的这一“测试计划”文档有助于实现以下目标：

1. 确定现有项目的信息和应测试的软件构件。
2. 列出推荐的测试需求（高级需求）。
3. 推荐可采用的测试策略，并对这些策略加以说明。
4. 确定所需的资源，并对测试的工作量进行估计。
5. 列出测试项目的可交付元素。

## 1.2     背景

本次测试的对象是一个在线论坛系统，旨在为用户提供发布帖子、评论互动等功能。系统的主要功能包括用户注册与登录、文章的发布、修改与删除，以及用户评论等。该系统支持普通用户和管理员登录，不同身份的用户有相应的权限设置。普通用户可以发帖、评论、修改个人信息等，而管理员则有权对普通用户的操作进行管理，例如封禁用户或处理违规内容。

系统的架构采用了前后端分离的方式，前端使用HTML进行界面展示，后端则基于Java实现业务逻辑。数据库部分使用了RDBMS来存储用户数据、帖子信息等。系统支持第三方登录（如GitHub），并实现了增删改查、文章标签推荐等功能，能够在不同用户之间创建一个高效的社区互动平台。为了确保性能和可靠性，系统被设计为能支持最多2000名用户同时在线，保证80%的事务能在2分钟内完成。

该项目的开发始于作为软件工程课程的实践项目，目的是通过实际开发过程，提高团队成员的软件工程技能。项目历经了系统设计、前后端开发、数据库架构搭建等多个阶段，并最终计划在第八周完成交付。项目开发过程中还特别关注了系统的安全性和可用性，确保用户数据的安全及系统的稳定运行。

## 1.3     范围

**一、本测试计划涵盖以下几个主要阶段：**

1. 单元测试：单元测试将专注于系统中最小的可测试单元，如用户登录、发帖、修改帖子等基本功能。每个模块的输入、输出、边界条件将单独测试，以确保其功能正常。

2. 集成测试：在单元测试完成后，将进行集成测试。集成测试的重点是验证前后端的交互是否正常，包括用户登录后能否顺利发帖、修改个人信息，以及后台管理功能是否能够正常使用。

3. 系统测试：系统测试阶段将对整个系统进行全面测试，涵盖功能性测试和非功能性测试。测试内容包括所有主要和次要的业务流程，确保系统在实际运行时能够满足设计需求。

4. 功能测试：该阶段主要测试用户操作的功能是否按预期工作，如用户注册、登录、发帖、评论、修改帖子等核心功能。

5. 性能测试：性能测试将评估系统的响应时间、并发用户数、资源消耗等，确保系统在高负载条件下能够正常运行，并达到性能标准。

**二、接受测试的功能：**

- 用户注册和登录：验证用户可以正常注册、登录和注销，包括GitHub第三方登录功能。

- 文章发布和管理：测试用户发帖、修改和删除文章的功能，以及管理员对文章的管理权限。

- 评论功能：测试用户可以发表评论、删除评论及二级评论的功能。

- 文章推荐：测试系统的文章标签和相似标签推荐功能。

- 用户权限管理：验证管理员的权限是否能够有效管理普通用户，确保封禁等操作的执行准确性。

**三、接受测试的性能：**

- 系统稳定性：验证系统在高并发条件下的稳定性，确保系统能同时处理2000名用户在线，且宕机时间低于10%。

- 事务处理效率：确保80%的事务能在2分钟内完成，满足性能要求。

**四、不接受测试的功能和性能：**

- 遗留系统的兼容性：由于遗留系统已经存在较长时间，本测试计划不覆盖课程目录系统的功能测试，但会确保系统与遗留系统的接口正常运行。

- 用户体验设计：UI的细节设计和交互体验不会在本次测试中进行细致测试，主要关注功能性实现。

**五、在编写本测试计划时，作出以下假设：**

1. 所有测试数据均为有效且可用的，并且已经准备好用于测试的执行。

2. 开发环境、测试环境和生产环境之间不存在重大差异，确保测试结果具有良好的代表性。

3. 测试环境中有足够的硬件资源支持模拟高并发的用户行为。

4. 所有系统接口都已正确实现，并且在开发和测试阶段不会发生显著变化。

**六、风险和意外事件：**

1. 高并发条件下的数据库性能问题：在并发用户数接近系统最大负载时，可能出现数据库响应时间过长或数据丢失的情况。

2. 第三方服务不稳定：GitHub第三方登录依赖外部服务，若该服务中断或响应时间过长，将影响系统登录功能的测试结果。

3. 开发进度延迟：若开发团队无法按时完成功能开发，将导致测试阶段延期。

**七、 约束：**

1. 时间约束：项目需在第八周之前完成交付，测试时间较为紧迫，可能无法进行全面的回归测试。

2. 资源约束：测试过程中，硬件和人员资源有限，无法在所有环境下模拟极端负载情况。

3. 技术约束：遗留系统的架构设计较为老旧，与新系统集成时可能会遇到兼容性问题。

## 1.4 项目核实

下表列出了制定*测试计划*时所使用的文档，并标明了各文档的可用性：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **文档 （版本/日期）** | **已创建或可用** | **已被接收或已经过复审** | **作者或来源** | **备注** |
| 需求规约1.1 | ☑ 是  否 | ☑ 是  否 | 张睿超 |  |
| 功能性规约1.1 | ☑ 是  否 | ☑ 是  否 | 张睿超 |  |
| 项目计划1.1 | ☑ 是  否 | ☑ 是  否 | 张星驰 |  |
| 设计规约 |  是 ☑ 否 |  是 ☑ 否 |  |  |
| 原型1.1 | ☑ 是  否 | ☑ 是  否 | 胡子昂 |  |
| 用户手册 |  是 ☑ 否 |  是 ☑ 否 |  |  |
| 业务模型或业务流程 |  是 ☑ 否 |  是 ☑ 否 |  |  |
| 数据模型或数据流 |  是 ☑ 否 |  是 ☑ 否 |  |  |
| 业务功能和业务规则1.1 | ☑ 是  否 | ☑ 是  否 | 张睿超 |  |
| 项目或业务风险评估 |  是 ☑ 否 |  是 ☑ 否 |  |  |

# 2.     测试需求

下表用于确定被当作测试对象的各项需求（例如用例、功能性需求和非功能性需求）。表中列出了将要测试的对象。

|  |  |  |  |
| --- | --- | --- | --- |
| **测试需求** | **测试内容** | **优先级** | **备注** |
| 用户登录功能 | 测试用户注册、登录、注销功能是否正常，包括GitHub第三方登录 | 高 | 验证所有用户身份验证流程 |
| 发帖功能 | 测试用户能否成功发布帖子，包括填写主题、内容及附件 | 高 | 包括正常发布和编辑功能 |
| 修改帖子功能 | 测试用户能否修改已发布的帖子，验证修改后的帖子是否正确保存 | 高 | 包括帖子内容和标题的编辑 |
| 删除帖子功能 | 测试用户能否删除已发布的帖子，确保删除操作能正确执行 | 高 | 验证帖子删除后的系统状态 |
| 评论功能 | 测试用户能否成功评论帖子，添加、修改和删除评论 | 高 | 包括对单一评论和二级评论的操作 |
| 文章推荐功能 | 测试系统根据文章标签提供的推荐功能是否正常 | 中 | 验证推荐算法的有效性和准确性 |
| 用户权限管理 | 测试管理员是否能够有效管理用户权限，包括封禁和权限设置 | 高 | 验证管理员操作的准确性和效果 |
| 系统稳定性 | 测试系统在高并发条件下的稳定性，确保系统能够处理最大负载 | 高 | 包括并发用户数达到2000的场景 |
| 事务处理效率 | 测试系统能否在2分钟内完成80%的事务 | 高 | 包括登录、发帖、评论等操作的响应时间 |
| 系统安全性 | 测试系统是否存在安全漏洞，包括SQL注入、XSS攻击等 | 高 | 确保用户数据和系统操作的安全性 |
| UI界面 | 验证前端页面的用户界面是否符合设计要求，功能是否正常 | 中 | 包括用户体验的基本验证 |
| 兼容性 | 测试系统在不同浏览器和设备上的兼容性 | 中 | 确保系统在主流浏览器和设备上的一致性 |

# 3.     测试策略

**1. 单元测试**

说明：单元测试将验证系统中每个模块或功能的正确性，如登录功能、发帖功能、评论功能等。每个模块的输入和输出将逐个测试，确保其在独立环境下能够正常工作。

方法：使用JUnit测试框架，编写单元测试用例。对每个模块进行输入输出的验证，包括正常场景和边界条件测试。

执行原因：确保系统的基础功能模块在开发阶段能够正确工作，减少后续集成中的问题。

完成标准：当所有单元测试用例都通过，且每个模块在所有测试数据下返回期望的结果时，单元测试完成。

**2. 集成测试**

说明：集成测试将验证前端和后端系统、以及各个模块之间的交互是否正常，确保数据流动和功能实现的完整性。

方法：模拟用户登录、发帖、修改帖子、评论等操作，测试前后端的请求与响应的正确性。重点测试数据流在不同模块之间的传递，确保模块间协同工作无误。

执行原因：验证系统各个模块在集成环境中的功能完整性，确保模块之间无兼容性问题。

完成标准：所有集成测试用例成功通过，前后端通信、数据库交互以及模块间的依赖均无错误。

**3. 系统测试**

说明：系统测试将覆盖整个系统的所有功能，包括功能性和非功能性需求，确保系统在预期的真实场景中能正常运行。

方法：基于用例场景执行功能性操作，包括用户注册、登录、发帖、评论等全流程测试。针对每个功能进行输入验证、结果验证和用户权限管理验证。同时，评估系统在不同并发用户数下的表现，尤其是事务处理效率和系统稳定性。

执行原因：全面验证系统功能和性能，确保交付的产品符合预期需求和标准。

完成标准：所有系统功能在真实场景下运行正常，且系统性能和可靠性符合验收标准。

**4. 功能测试**

说明：功能测试的主要目标是验证系统的核心功能是否按预期工作，尤其是用户交互功能和管理权限。

方法：基于实际业务场景设计测试用例，涵盖登录、发帖、评论、修改、删除、推荐等主要功能。每个功能的输入输出结果将被验证，确保功能性需求被满足。

执行原因：验证系统的功能性需求，确保用户能够按照预期进行操作，管理功能有效。

完成标准：所有功能测试用例通过，且系统的功能性需求被正确实现。

**5. 性能测试**

说明：性能测试用于评估系统在高并发条件下的表现，以及响应时间、事务处理效率等是否达到预期标准。

方法：使用工具（如JMeter或LoadRunner）模拟高并发场景，评估系统在2000名用户同时在线时的稳定性。重点测试响应时间、事务处理时间和资源利用率。

执行原因：确保系统能够在高负载环境下保持正常工作，并满足性能要求。

完成标准：系统在80%的事务能够在2分钟内完成，且宕机时间低于10%，满足设计要求。

**6. 安全测试**

说明：安全测试的目的是验证系统是否存在安全漏洞，如SQL注入、跨站脚本攻击（XSS）、用户权限越界等。

方法：使用自动化安全测试工具（如OWASP ZAP）和手动渗透测试来查找潜在的漏洞。同时，验证用户身份验证机制和权限控制是否严格有效。

执行原因：保障系统的安全性，防止用户数据泄露和未经授权的操作。

完成标准：系统在所有已知的安全测试中通过，未检测到重大安全漏洞。

**7. 兼容性测试**

说明：兼容性测试用于验证系统在不同设备、操作系统和浏览器上的表现是否一致，确保系统在不同环境下能正常运行。

方法：在多种操作系统（Windows 10、macOS等）、浏览器（Chrome、Firefox、Edge等）和设备（桌面、移动设备）上运行系统，确保UI一致性和功能性正常。

执行原因：确保系统能适应多种设备和平台，用户体验一致且不受设备限制。

完成标准：在所有测试的操作系统、浏览器和设备上，系统能够正常运行且无重大兼容性问题。

**8. 不实施的测试**

用户体验测试：将不实施和执行该测试。原因是用户体验设计不是本次测试计划的重点，主要关注的是功能和性能的实现。

**测试将在以下条件下结束：**

所有计划的测试类型和用例已执行，且大部分用例均通过。

所有高优先级的功能和性能缺陷已修复，且验证通过。

关键的安全性、性能和可靠性测试通过，达到预期标准。

## 3.1               测试类型

### 3.1.1     数据和数据库完整性测试

|  |  |
| --- | --- |
| 测试目标： | 确保数据库访问方法和进程正常运行，数据不会遭到损坏。 |
| 技术： | **数据库访问方法调用**：直接调用数据库中的各种数据访问方法和进程，不通过前端界面，而是通过SQL语句、数据库API等接口，直接操作数据库。测试中将填充有效和无效的数据，以模拟真实操作场景和异常情况。  **数据完整性检查**：验证插入、更新、删除等数据库操作是否准确执行，确保数据在数据库中的存储符合预期。对于查询操作，将确保返回的结果正确无误。  **日志和事件验证**：检查数据库中的所有相关事件日志，确保事务、锁机制、索引等都按设计正确运行。 |
| 完成标准： | 所有的数据库访问方法和进程都按照设计的方式运行，数据没有遭到损坏。 |
| 需考虑的特殊事项： | 测试可能需要 DBMS 开发环境或驱动程序 在数据库中直接输入或修改数据。  进程应该以手工方式调用。  应使用小型或最小的数据库（记录的数量有限） 来使所有无法接受的事件具有更大的可视度。 |

### 3.1.2     功能测试

|  |  |
| --- | --- |
| 测试目标： | 确保系统的核心功能运行正常，包括用户导航、数据输入、数据处理、数据检索等操作。  验证业务规则是否正确应用，并确保所有操作在符合业务需求的前提下正常工作。  核实系统在处理有效和无效数据时的响应是否符合预期。 |
| 技术： | 用例执行：根据已定义的用例及业务功能，利用有效和无效的数据执行每个用例和功能流，核实其输出结果。  业务规则验证：确保系统对所有业务规则的执行符合业务逻辑的要求，尤其是在复杂操作（如多步骤流程、数据处理规则等）中正确应用规则。  黑盒测试：通过黑盒技术，仅关注输入和输出的验证，不涉及系统内部代码的实现，确保所有可见的功能运行正常。 |
| 完成标准： | 所有计划中的功能测试用例均已执行，并且覆盖了系统的全部核心功能。  所有发现的缺陷都已记录、修复，并通过重新测试验证。  系统能够在不同数据输入条件下按预期返回正确的结果，或适当处理错误输入。 |
| 需考虑的特殊事项： | 外部依赖因素：某些业务流程可能依赖于外部系统或服务，如第三方API或数据库接口。此类依赖可能会影响功能测试的顺利执行，需要在测试前确保相关外部系统的可用性。  测试环境：测试应在接近实际生产环境的测试环境中进行，确保所有系统配置和集成点都按实际场景执行，避免测试结果偏离真实情况。  数据隐私与安全：在功能测试过程中，需确保测试数据的合法性和安全性，避免使用生产数据或泄露敏感信息。 |

### 3.1.3     业务周期测试

|  |  |
| --- | --- |
| 测试目标 | 确保系统和后台进程能够按照设计的业务模型和时间表正确运行，包括周期性事务的处理。  核实系统在执行周期性操作时能够正确处理时间相关的事件，如月度报表生成、每日任务提醒等。 |
| 技术： | 1. 功能测试改进：在业务周期测试中，功能测试将被修改和增强，以增加每个功能的执行次数。通过模拟不同用户在不同时间段内的操作，测试在系统负载变化时的行为。例如，增加每日登录、发帖、评论等操作的次数来模拟一年内用户活动的情况。 2. 时间模拟：使用有效和无效的时间段来测试与时间相关的功能。例如，测试系统在不同时间点（如过期时间、未来时间）执行任务的情况，确保日期相关的业务逻辑得到正确应用。 3. 周期性功能验证：对于系统中需要定期触发的功能，如定时报告生成、任务提醒等，测试将确保这些功能能够按计划在预定的时间正确执行。 4. 有效与无效数据验证：使用有效和无效的数据及时间段进行测试，验证系统能够在正确的数据条件下返回预期结果，并在输入无效数据或错误时间时显示相应的错误消息或警告。 5. 业务规则核查：通过反复执行周期性操作，验证业务规则在长期的业务周期中能否持续正确实施，确保系统的稳定性。 |
| 完成标准： | 所有计划的业务周期测试用例均已执行，并覆盖了系统的所有周期性操作。  系统在不同时间段和数据条件下表现正常，所有与时间相关的业务功能都正确执行。  所有发现的缺陷都已解决，并通过重新测试验证其修复的正确性。 |
| 需考虑的特殊事项： | 1. 系统日期和事件支持：在模拟不同业务周期时，系统日期可能需要手动调整或通过特殊脚本支持，以确保在测试中能够触发未来或过去的事件。应特别关注与日期有关的边界条件（如年终、季末等）。 2. 业务模型驱动的测试：测试需求和测试过程应根据具体的业务模型来确定。业务模型将指导测试设计，确保所有关键的业务操作（如每月数据报表生成、每日任务提醒等）都能得到充分的测试。 |

### 3.1.4     用户界面测试

|  |  |
| --- | --- |
| 测试目标： | 交互验证：确保用户能够通过界面顺利完成系统功能的操作，如窗口导航、数据输入和提交等。验证各个界面元素（如按钮、文本框、菜单、图标）是否能够正常响应用户操作。  一致性检查：确保所有窗口的布局、大小、位置、状态（如启用/禁用）符合设计规范和用户期望，并在不同分辨率和设备上显示一致。  标准合规：确保用户界面设计符合公司或行业的UI标准，用户体验符合最佳实践，且界面元素位置、颜色、字体等符合易用性规范。 |
| 技术： | 导航测试：针对每个窗口，测试不同的导航方式，包括通过Tab键、鼠标点击、快捷键等方式进行窗口之间、字段之间的切换，确保浏览顺畅且直观。  UI对象测试：为每个窗口内的对象（按钮、输入框、下拉菜单等）创建测试案例，确保每个UI元素都能正确响应用户的输入操作。重点核查按钮点击、输入验证、菜单展开等交互是否按预期运行。  窗口状态测试：测试窗口在不同状态下的表现，例如最大化、最小化、关闭、启用/禁用等，确保窗口在这些状态下的表现符合预期。 |
| 完成标准： | 交互功能正常：所有窗口与窗口之间、字段与字段之间的导航和交互能够按照预期顺利进行。用户能够通过界面正确访问系统功能。  UI一致性：所有测试窗口的UI元素与基准版本或设计规范保持一致，界面布局、功能及响应符合可接受的标准，用户体验流畅无障碍。  缺陷修复：发现的UI缺陷已得到修复，并通过回归测试验证。 |
| 需考虑的特殊事项： | 第三方UI控件限制：一些定制的或来自第三方库的UI控件可能无法完全访问其所有属性和状态，因此需要针对这些对象进行特殊处理或限制范围的测试。  响应式设计：如果系统需要在移动设备上使用，需考虑响应式设计测试，确保界面在不同尺寸的设备上都能正常显示和操作。 |

### 3.1.5     性能评测

|  |  |
| --- | --- |
| 测试目标： | 正常工作量：验证系统在正常业务操作条件下的性能是否符合预期，包括响应时间、事务处理时间等关键指标。  最繁重工作量：测试系统在最高负载情况下的性能，评估系统能否在极端情况下正常处理高并发事务。 |
| 技术： | 功能或业务周期测试的扩展：基于功能测试或业务周期测试的已有测试场景，通过增加事务数量或提高测试的迭代次数，模拟不同的工作负载条件，测试系统在高负载情况下的性能表现。  数据文件和脚本调整：  增加数据文件中事务的数量，或通过测试脚本的调整增加每个事务的迭代次数，以模拟更多用户或事务的并发处理。  测试脚本可以从单个用户、单个事务为基准逐步扩展，模拟多个并发用户或事务。  多用户/事务测试：  测试可以在单个机器上执行，并逐渐增加虚拟客户机数量，模拟多用户并发环境。  使用虚拟客户机或远程终端仿真技术模拟数百个并发用户，增加服务器和网络的负载。 |
| 完成标准： |  单个事务或单个用户：在每个事务的执行时间范围内，所有测试脚本应无故障完成，响应时间符合预期。   多个事务或多个用户：系统在高并发负载下，所有事务应在可接受的时间范围内完成，且无故障，系统资源利用率和响应时间保持在合理水平。 |
| 需考虑的特殊事项： | 服务器负载管理：综合性能测试可能需要在服务器上添加额外的后台工作负载，以模拟真实操作场景。虚拟用户模拟：使用远程终端仿真工具模拟大量（数百个）虚拟客户机，同时进行操作，测试系统在多用户环境下的响应能力。  实际客户机测试：在高并发测试中，使用多台实际客户机运行测试脚本，以更真实地模拟系统在现实环境中的表现。  环境控制：性能测试应在专用的计算机或专用的测试时间段内进行，以保证测试环境的可控性和测试结果的准确性。  数据库规模：性能测试使用的数据库应与实际系统中的数据库大小一致，或者与实际数据库规模成比例缩放，确保测试结果与实际情况具有一致性和可参考性。 |

### 3.1.6     负载测试

|  |  |
| --- | --- |
| 测试目标： | 核实系统在不同负载条件下的性能：验证系统在从正常工作负载到极端高负载情况下，是否能够维持稳定运行，确保关键事务或业务功能能够在可接受的时间范围内完成。  评估响应时间和事务处理能力：测试系统在不同负载水平下的响应时间和事务处理速率，确保这些性能特征在高负载情况下仍符合要求。 |
| 技术： | 基于功能或业务周期测试的扩展：将功能或业务周期测试中的已有测试场景进行扩展，通过增加事务数量或提高事务的迭代次数，模拟不同的工作负载场景。  修改数据文件和测试脚本：通过增加数据文件中的事务数量，或通过修改测试脚本以增加每项事务的执行次数，创建高负载环境，以观察系统的性能表现。 |
| 完成标准： | 多个事务或多个用户：系统应在可接受的时间范围内完成所有预定的测试脚本，没有任何系统崩溃或故障，确保负载下的事务处理时间仍然符合预期。 |
| 需考虑的特殊事项： | 专用的计算资源和环境：负载测试应在专用的计算机上或在特定的机时内进行，以确保测试环境的可控性和结果的精确性，避免外部因素影响测试结果。  实际大小或相同缩放比例的数据库：负载测试所使用的数据库应与生产环境中的数据库大小一致，或根据实际情况按比例缩放，以保证测试的真实有效性。 |

### 3.1.7     强度测试

|  |  |
| --- | --- |
| 测试目标： | 核实系统在资源极其有限或最大负荷下的性能表现，确保在以下情况下系统能够正常运行或正确处理错误：  服务器几乎没有可用的内存（RAM 和 DASD）。  连接或模拟的客户机数量达到最大实际允许值。  多个用户同时对相同的数据或账户执行相同的事务。  系统面临最繁重的事务量或最差的事务组合。  确定系统无法继续正常运行的边界条件，并记录导致系统失效的情况。 |
| 技术： | 基于性能评测或负载测试的技术：使用为性能评测或负载测试制定的测试脚本，结合特定的强度条件来测试系统。  减少或限制服务器资源：在单台计算机上运行测试，通过减少服务器上的可用内存（RAM）和磁盘空间（DASD）等资源，模拟资源匮乏的情境。  增加事务负荷：通过多台客户机同时执行相同或互补的测试，生成最大事务量或最复杂的事务组合，从而测试系统在繁重负载下的表现。 |
| 完成标准： | 无软件故障：系统在达到或超出指定系统限制时，仍能继续正常运行或正确处理错误，没有出现任何软件故障。  故障条件不在规定范围内：如果系统在测试过程中发生故障，则故障条件必须在测试目标规定的范围之外。 |
| 需考虑的特殊事项： | 增加网络负荷：如果需要增加网络工作强度，可使用网络工具为网络加载消息或信息包，以模拟高网络负载场景。  限制数据库增长：暂时减少系统可用的磁盘空间（DASD），限制数据库可用空间的增长，从而模拟磁盘空间不足的场景。  同步访问冲突：使多个客户机对相同记录或数据账户进行同步访问，以测试资源争用情况。 使多个客户机对相同的记录或数据账户 同时进行的访问达到同步。 |

### 3.1.8     容量测试

|  |  |
| --- | --- |
| 测试目标： | 验证系统在高容量条件下的稳定性和性能，包括：  连接或模拟了最大数量的客户机（实际或允许的最大值），所有客户机在长时间内执行最坏情况下的业务功能。  达到最大数据库大小（实际的或按比例缩放的），同时执行多个查询或报表事务。 |
| 技术： | 基于性能评测或负载测试的技术：使用先前为性能评测或负载测试创建的测试方案，针对容量测试的需求进行适当调整。  多台客户机模拟并发：通过多台客户机运行相同或互补的测试，在长时间内产生最繁重的事务量或最差的事务组合，模拟高负载下的系统运行情况。  创建最大数据库：构建一个实际大小、按比例缩放的、或填充了代表性数据的最大数据库，在多台客户机同时运行查询和报表事务的情况下，测试系统的容量极限。 |
| 完成标准： | 无软件故障：测试成功的标准是系统能够在达到或超出指定的系统限制时正常运行，没有发生任何软件故障。  持续长时间运行：在高容量条件下，系统需要能够持续长时间运行，而不会出现性能大幅下降、数据丢失或系统崩溃等问题。 |
| 需考虑的特殊事项： | 时间段的可接受性：需要明确在上述高容量条件下，哪个时间段是系统可接受的运行时间。例如，是否允许系统在高负荷运行时响应变慢，或者系统应当在一定时间内完成处理任务。  数据库容量的实际可用性：确定测试时所使用的数据库大小是否足够代表实际生产环境的数据规模，以确保测试结果的有效性和代表性。 |

### 3.1.9     安全性和访问控制测试

|  |  |
| --- | --- |
| 测试目标： | 应用程序级别的安全性： 核实不同用户角色（Actor）只能访问其被授权的功能和数据。  系统级别的安全性： 确认只有具有相应系统和应用权限的用户才能成功登录并访问系统。 |
| 技术： | 应用程序级别的安全性：   1. 列出每个用户角色及其授权的功能和数据。 2. 为每个用户角色设计测试，验证其权限的正确性。 3. 修改用户角色权限，重新运行相同的测试，以确保权限被正确授予或限制。   系统级别的安全性：   1. 测试系统登录和远程访问的安全性，确保只有授权用户能够访问系统资源。 |
| 完成标准： | 每个已知用户角色仅能访问其授权的功能或数据，所有事务按照预期运行，无安全漏洞。 |
| 需考虑的特殊事项： | 与网络或系统管理员合作，确认系统访问权的配置是否正确。这部分测试可能会被分配给网络管理团队。 |

### 3.1.10     故障转移和恢复测试

|  |  |
| --- | --- |
| 测试目标： | 确保恢复进程（手工或自动）将数据库、应用程序和系统正确地恢复到预期的已知状态。测试应包括以下各种情况：  客户机断电  服务器断电  通过网络服务器产生的通信中断  DASD 和/或 DASD 控制器被中断、断电或与 DASD 和/或 DASD 控制器的通信中断  周期未完成（如数据过滤进程或数据同步进程被中断）  数据库指针或关键字无效  数据库中的数据元素无效或遭到破坏 |
| 技术： | 使用为功能和业务周期测试创建的测试来创建一系列的事务。  模拟或实际执行以下操作：  客户机断电：关闭 PC 机的电源。  服务器断电：模拟或启动服务器的断电过程。  网络通信中断：模拟或启动网络的通信中断（如实际断开通信线路的连接或关闭网络服务器或路由器的电源）。  DASD 和 DASD 控制器中断：模拟与一个或多个 DASD 控制器或设备的通信，或实际取消这种通信。  执行其他事务后，调用恢复过程。  在测试不完整的周期时，异常终止或提前终止数据库进程。  对于破坏数据库字段、指针和关键字的测试，使用数据库工具手动进行破坏。其他事务通过应用程序功能测试和业务周期测试执行。 |
| 完成标准： | 在所有上述情况中，应用程序、数据库和系统应该在恢复过程完成时立即返回到一个已知的预期状态，包括：  仅限于已知损坏的字段、指针或关键字范围内的数据损坏  表明进程或事务因中断而未被完成的报表 |
| 需考虑的特殊事项： | 恢复测试可能会给其他操作带来麻烦。断开缆线连接的方式（如模拟断电或通信中断）可能不可取或不可行，可能需要使用其他方法（例如诊断性软件工具）。  需要系统（或计算机操作）、数据库和网络组中的资源。  测试应在工作时间之外或在一台独立的计算机上运行。 |

### 3.1.11     配置测试

|  |  |
| --- | --- |
| 测试目标： | 核实在线论坛系统在多种硬件和软件配置中正常运行，确保其在不同用户环境中具备稳定性与兼容性，尤其是在用户可能使用的多种浏览器、操作系统和设备上。 |
| 技术： | 使用功能测试脚本 使用为在线论坛开发的功能测试脚本，验证系统的核心功能，包括用户注册、发帖、回复、搜索和文件上传等是否能在各种硬件与软件环境中正常运行。  打开非测试对象相关的软件 在进行配置测试时，打开各种与在线论坛无关的常用软件（如浏览器扩展程序、聊天工具、文档编辑工具等），模拟用户实际的工作场景。观察这些非相关软件是否对论坛功能产生影响。  执行事务并模拟资源消耗 在进行测试时，通过执行一系列的用户事务（如发帖、点赞、回复、上传图片等），模拟用户在实际使用中与论坛系统的交互。同时通过运行多个其他应用程序来减少工作站可用内存，观察在内存不足的情况下，系统是否依然稳定。  浏览器与设备兼容性测试 针对不同浏览器（如Chrome、Firefox、Edge、Safari等）和设备类型（PC、平板、手机）进行兼容性测试，确保在线论坛在多种配置下的响应速度和界面显示正确。 |
| 完成标准： | 对于在线论坛系统和非测试对象软件的各种组合，所有用户事务（发帖、注册、搜索、文件上传等）都能成功完成且没有出现任何功能性故障或崩溃。 |
| 需考虑的特殊事项： | 用户可能使用的非测试对象软件：需要了解论坛用户可能会在其设备上运行哪些软件（如文档编辑器、邮箱、社交软件等）并结合这些软件进行测试。  设备资源占用：当用户设备运行大型应用程序（如在浏览器中打开多个标签页，或运行资源密集型软件如Excel或Photoshop）时，论坛系统是否仍能保持良好的响应速度和功能稳定性。  网络环境差异：考虑到用户可能会使用不同的网络环境（如Wi-Fi、4G、5G），测试应包括在这些不同条件下的论坛性能表现。 |

### 3.1.12     安装测试

|  |  |
| --- | --- |
| 测试目标： | 核实在线论坛系统在以下情况下能够正确安装到各种所需的硬件配置中，并确保在安装后系统能够正常运行：  首次安装：在以前从未安装过在线论坛系统的计算机上进行首次安装。  版本更新：在已安装相同版本在线论坛系统的计算机上进行更新。  升级安装：在已安装早期版本在线论坛系统的计算机上进行升级安装。 |
| 技术： | 手工或自动脚本验证目标计算机状态 开发手工脚本或自动化脚本，验证目标计算机的初始状态。确保系统处于以下状态：  从未安装过在线论坛系统。  已安装相同版本的在线论坛系统。  已安装早期版本的在线论坛系统。  启动或执行安装程序 根据不同的安装条件（首次安装、版本更新或升级安装），手动或自动执行在线论坛的安装过程。特别注意以下内容：  检查磁盘空间是否足够。  检查安装目录的创建权限。  处理可能的异常情况，如磁盘空间不足或权限缺失。  功能测试 使用功能测试脚本的子集，模拟用户操作，验证系统安装后是否能够正常执行论坛的核心功能，例如：  用户注册和登录。  发帖与回复。  文件上传与下载。  搜索功能。 |
| 完成标准： | 在线论坛系统能够成功执行所有选定的核心事务，且没有出现任何安装或运行故障。 |
| 需考虑的特殊事项： |  应根据安装测试的目标，选择哪些核心事务最能体现在线论坛系统已成功安装且主要软件构件未遗漏。   验证安装过程中的日志，确保所有必要的组件和依赖项均已成功安装。 |

## 3.2     工具

此项目将使用以下工具：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 工具 | 产商/自产 | 版本 |
| 测试管理 | Jira | Atlassian | Jira 9.x |
| 缺陷跟踪 | Bugzilla | Mozilla Foundation | Bugzilla 6.x |
| 用于功能性测试的 ASQ 工具 | Selenium | 开源社区 | Selenium 4.x |
| 用于性能测试的 ASQ 工具 | JMeter | Apache | JMeter 5.6 |
| 测试覆盖监测器或评测器 | SonarQube | SonarSource | SonarQube 9.x |
| 项目管理 | Microsoft Project | Microsoft | MS Project 2024 |
| DBMS 工具 | MySQL | Oracle | MySQL 8.1 |

# 4.     资源

## 4.1     角色

下表列出了在此项目的人员配备方面所作的各种假定。

|  |  |  |
| --- | --- | --- |
| 人力资源 | | |
| 角色 | 所推荐的最少资源  （所分配的专职角色数量） | 具体职责或注释 |
| 测试经理，  测试项目经理 | 2 | 进行管理监督。  职责：           提供技术指导           获取适当的资源           提供管理报告 |
| 测试设计员 | 1 | 确定测试用例、确定测试用例的优先级并实施测试用例。  职责：           生成测试计划           生成测试模型           评估测试工作的有效性 |
| 测试员 | 2 | 执行测试。  职责：           执行测试           记录结果           从错误中恢复           记录变更请求 |
| 测试系统管理员 | 1 | 确保测试环境和资产得到管理和维护。  职责：           管理测试系统           分配和管理角色对测试系统的访问权 |
| 数据库管理员 | 1 | 确保测试数据（数据库）环境和资产得到管理和维护。  职责：           管理测试数据（数据库） |
| 设计员 | 1 | 确定并定义测试类的操作、属性和关联。  职责：           确定并定义测试类           确定并定义测试包 |
| 实施员 | 2 | 实施测试类和测试包，并对它们进行单元测试。  职责：           创建在测试模型中实施的测试类和测试包 |

## 4.2     系统

下表列出了测试项目所需的系统资源。

|  |  |
| --- | --- |
| 系统资源 | |
| 资源 | 名称/类型 |
| 数据库服务器 | MySQL Server |
| -网络或子网 | TBD |
| -服务器名称 | TBD |
| -数据库名称 | TBD |
| 客户端测试 PC | Windows 11, 16GB, Intel i7 |
| -包括特殊的配置需求 | TBD |
| 测试存储库 | GitHub |
| -网络或子网 | TBD |
| -服务器名称 | TBD |
| 测试开发 PC | TBD |

# 5.     项目里程碑

|  |  |  |  |
| --- | --- | --- | --- |
| **里程碑任务** | **工作** | **开始日期** | **结束日期** |
| 制定测试计划 | 确定测试目标、策略、资源需求和时间表，编写测试计划文档。 | 2024-09-16 | 2024-09-19 |
| 设计测试 | 设计详细的测试用例、测试脚本和测试数据，定义测试环境和配置。 | 2024-09-20 | 2024-09-25 |
| 实施测试 | 设置测试环境，准备测试数据，部署测试工具。 | 2024-09-26 | 2024-09-29 |
| 执行测试 | 执行测试用例，记录测试结果，处理和报告发现的问题。 | 2024-09-30 | 2024-10-05 |
| 对测试进行评估 | 分析测试结果，总结测试报告，评估测试覆盖率和效果，提出改进建议。 | 2024-10-06 | 2024-10-07 |

# 6.     可交付工件

## 6.1     测试模型

 测试计划文档：描述测试的范围、目标、策略和资源需求。

创建人员：测试经理

交付对象：项目团队和利益相关者

交付时间：2024-09-19

 测试用例文档：包含详细的测试用例、测试脚本和预期结果。

创建人员：测试设计人员

交付对象：测试团队

交付时间：2024-09-25

 测试数据集：用于测试的实际数据，包括数据生成脚本。

创建人员：测试开发人员

交付对象：测试团队

交付时间：2024-09-25

## 6.2     测试记录

 测试执行记录：记录每次测试执行的详细结果，包括通过/失败状态和实际结果。

工具：JIRA

创建人员：测试工程师

交付对象：测试项目团队

交付时间：持续更新

 测试状态报告：定期生成的报告，概述测试的进展、覆盖率和关键问题。

工具：JIRA

创建人员：测试经理

交付对象：项目经理和利益相关者

交付时间：2024-10-05

## 6.3     缺陷报告

 缺陷报告文档：详细描述发现的缺陷，包括缺陷的严重性、影响和解决方案建议。

工具： JIRA

创建人员：测试工程师

交付对象：开发团队和项目经理

交付时间：2024-10-07

 缺陷跟踪报告：跟踪缺陷的修复状态，记录缺陷的处理进展。

工具：JIRA

创建人员：测试经理

交付对象：项目经理和利益相关者

交付时间：持续更新

# 7.     附录 A：项目任务

以下是一些与测试有关的任务：

 制定测试计划

-          确定测试需求

-          评估风险

-          制定测试策略

-          确定测试资源

-          创建时间表

-          生成测试计划

 设计测试

- 准备工作量分析文档

- 确定并说明测试用例

- 确定测试过程，并建立测试过程的结构

- 复审和评估测试覆盖

 实施测试

- 记录或通过编程创建测试脚本

- 确定设计与实施模型中的测试专用功能

- 建立外部数据集

 执行测试

- 执行测试过程

- 评估测试的执行情况

- 恢复暂停的测试

- 核实结果

- 调查意外结果

- 记录缺陷

 对测试进行评估

- 评估测试用例覆盖

- 评估代码覆盖

- 分析缺陷

- 确定是否达到了测试完成标准与成功标准