IEEE829 软件测试计划

软件测试计划

1 目的

本文档旨在规定 `com.demo` 项目后端（Service 层和 Controller 层）的测试范围、测试方法、测试所需的资源和测试活动的时间表。主要目的是确保 Service 层业务逻辑的正确性和 Controller 层 API 接口功能的稳定性与可靠性。本文档将确定具体的测试项、要测试的功能特性、要执行的测试任务（单元测试和集成测试）、每个任务的责任人以及与本计划相关的潜在风险与应对措施，为后续的测试设计、执行和评估提供依据。

2 测试计划标识

DEMO-STP-V1.0-20250325

3 介绍

本项目（以下简称 `Demo App`）是一个基于 Java 开发的 Web 应用程序，其后端主要负责处理用户和管理员对消息(Message)、新闻(News)、订单(Order)、用户(User)和场地(Venue)等核心实体的管理操作。项目结构显示，业务逻辑在 `service` 包中实现，并通过 `controller` 包暴露 RESTful API 接口，区分了普通用户 (`user`) 和管理员 (`admin`) 的操作权限。本次测试活动将聚焦于 `service` 层的单元测试和 `controller` 层的集成测试，以验证其功能的正确性、健壮性和接口的可用性。

4 测试项

- **源代码**:
  - `com.demo.service` 包下的所有接口 (`*.java`) 及其实现类 (`impl/*.java`)。
  - `com.demo.controller` 包下的 `admin` 和 `user` 子包中的所有 Controller 类 (`*.java`)。
- **版本号**: v1.0.0
- **先决条件**:

- 项目代码已成功编译打包。
- 测试环境（包括 JDK、Maven、数据库等）已准备就绪。
- 所有项目依赖库已正确解析和下载。
- 测试：无需部署，直接在开发环境运行。

5 要测试的特性

根据项目结构和通用功能推断，将测试以下主要特性：

- 单元测试 (Service Layer - com.demo.service.impl.*):

UserService

- 用户登录功能：验证用户 ID 和密码的正确性校验、用户不存在、密码错误、空值、空字符串等异常情况，以及管理员用户的登录。

OrderService

- 订单创建功能：验证正常订单提交，以及场地名称、开始时间、预订小时数、用户 ID 为空/null/无效值，场地不存在等异常情况。
- 订单查询功能（ID）：验证订单存在、不存在以及订单 ID 为无效值时的查询结果。
- 订单状态变更：验证确认、完成、拒绝订单功能的正确性，包括订单存在与不存在的情况，以及对已完成/已拒绝订单进行状态变更的限制。
- 用户订单查询（分页）：验证用户订单分页查询功能，包括用户存在/不存在订单、分页参数为空/无效等情况。
- 订单删除：验证删除存在和不存在的订单，以及删除被其他数据引用的订单。
- 订单更新：验证更新存在和不存在的订单，以及更新时部分参数为空/无效、场地名称不存在、预订时间冲突等情况。

MessageService

- 留言查询（ID）：验证根据 ID 查找存在和不存在的留言，以及留言 ID 为无效值的情况。
- 状态变更：验证通过和拒绝留言审核的功能，包括留言存在与不存在的情况，以及对已通过/已拒绝的留言进行状态变更的限制。
- 用户留言查询（分页）：验证用户留言分页查询功能，包括用户存在/不存在留言、分页参数为空/无效等情况。
- 待审核查询（分页）：验证待审核留言分页查询功能，包括有无待审核留言、分页参数为空/无效等情况。

- 已通过查询（分页）：验证已通过留言分页查询功能，包括有无已通过留言、分页参数为空/无效等情况。
- 创建留言：验证正常情况下的留言创建，以及用户 ID、内容为空/null 的情况。

VenueService

- 场地信息查询：验证通过 ID、名称查找场馆，以及查询所有场馆（分页和不分页）的功能，并测试数据不存在和参数无效的情况。
- 创建：验证正常创建场馆，以及场馆名称为空/null、关键信息缺失等情况。
- 更新：验证更新存在和不存在的场馆，以及场馆名称已存在、部分信息为空/无效等情况。
- 删除：验证删除存在和不存在的场馆，以及删除被其他数据引用的场馆。
- 名称统计：验证统计场馆名称数量的功能，包括名称存在、不存在、为空/null 的情况。

NewsService

- 新闻查询：验证查询所有新闻（分页）以及根据 ID 查找新闻的功能，并测试数据不存在和参数无效的情况。
- 创建：验证正常创建新闻，以及标题或内容为空/null 的情况。
- 删除：验证删除存在和不存在的新闻。
- 更新：验证更新存在和不存在的新闻，以及标题或内容为空/null 的情况。

VoService

- OrderVoService：验证根据订单 ID 返回 OrderVo，以及订单列表转换为 OrderVo 列表的功能，并处理订单不存在、为 null、列表为空或包含 null 元素的情况。
- MessageVoService：验证根据留言 ID 返回 MessageVo，以及留言列表转换为 MessageVo 列表的功能，并处理留言不存在、为 null、列表为空或包含 null 元素的情况。

集成测试 (Controller Layer - com.demo.Controller.*)

- **Admin Controllers:**
  - **AdminMessageController:** 测试获取消息管理页面、待审核消息列表、通过/拒绝/删除消息功能，并验证权限控制、参数校验、接口成功响应和异常处理。
  - **AdminVenueController:** 测试获取场馆管理/编辑/添加页面、获取场馆列表、添加/修改/删除场馆、检查场馆名称功能，并验证权限控制、参数校验、接口成功响应和异常处理。
  - **AdminNewsController:** 测试获取新闻管理/添加/编辑页面、获取新闻列

表、删除/修改/添加新闻功能，并验证权限控制、参数校验、接口成功响应和异常处理。

- **AdminOrderController:** 测试获取预约管理页面、未审核订单列表、通过/拒绝订单功能，并验证权限控制、参数校验、接口成功响应和异常处理。
- **AdminUserController:** 测试获取用户管理/添加/编辑页面、获取用户列表、修改/添加/删除用户、检查用户 ID 功能，并验证权限控制、参数校验、接口成功响应和异常处理。

- **User Controllers:**
  - **MessageController:** 测试获取消息列表/已通过/用户消息列表、发送/修改/删除消息功能，并重点验证用户认证与授权、参数校验、接口成功响应、异常处理以及安全性校验（拒绝空内容、未登录访问限制、操作他人留言限制、SQL/JavaScript 注入防御）。
  - **VenueController:** 测试获取场馆详情/列表页面、获取场馆列表功能，并验证用户认证与授权、参数校验、接口成功响应、异常处理以及 SQL 注入防御。
  - **NewsController:** 测试获取新闻详情/列表页面、获取新闻列表功能，并验证用户认证与授权、参数校验、接口成功响应、异常处理。
  - **UserController:** 测试获取注册/登录/用户信息页面、普通/管理员登录、登录失败、用户注册/登出、检查密码、更新用户信息功能，并重点验证用户认证与授权、参数校验、接口成功响应、异常处理以及用户 ID 校验、密码存储安全、弱密码校验。
  - **OrderController:** 测试获取订单管理/下单/编辑页面、获取用户/场馆订单列表、添加/完成/修改/删除订单功能，并重点验证用户认证与授权、参数校验、接口成功响应、异常处理以及整数溢出/数值范围/空场馆/操作权限/营业时间/历史时间校验。
- **IndexController:** 测试主页和管理员首页的访问，验证返回状态码、视图名称以及模型数据的正确性，并验证用户认证与授权和接口成功响应。

测试设计规格说明

详细的测试用例见《测试用例设计文档》。

6 不会被测试的特性

- **前端 UI**: 本次测试不涉及任何前端页面或用户界面的测试。
- **数据库本身**: 不直接测试数据库的性能、备份恢复等，仅关注通过 Service 层对其操作的正确性。
- **第三方依赖库**: 假设项目依赖的外部库（如 Spring Framework, Lombok 等）本身

是可靠的，不专门对其进行测试。

- **基础设施**: 如服务器操作系统、网络配置、中间件（如消息队列、缓存，除非其是核心业务逻辑的一部分且包含在 Service 层）的配置和性能。
- **非功能性测试**: 如性能测试、压力测试、安全性渗透测试等，这些需要专门的计划和工具，不在本次测试范围内。
- **原因**: 主要基于本次测试的目标（验证 Service 和 Controller 层的功能逻辑）和资源限制，将范围聚焦于核心后端代码。

7 方法

- **总体方法**: 采用白盒测试和灰盒/黑盒测试相结合的方法。利用 JUnit 5 测试框架执行测试。
- **单元测试**:
  - **目标**: `com.demo.service.impl` 包下的所有公共方法。
  - **技术**: 使用 JUnit 5 进行测试用例编写和执行。针对方法逻辑、分支、边界值、异常情况设计测试用例。
  - **工具**: JUnit 5, IDE (IntelliJ IDEA)。
  - **广度**: 目标是达到较高的代码覆盖率，特别是核心业务逻辑。
  - **完成准则**: 所有单元测试用例通过，代码覆盖率达到预定目标，无严重逻辑缺陷。
- **集成测试**:
  - **目标**: `com.demo.controller` 包下的所有 Controller 类及其暴露的 API 端点。
  - **技术**: 使用 JUnit 5 和 Spring Boot Test (如果项目是基于 Spring Boot)。利用 `@SpringBootTest` 加载应用上下文，使用模拟 HTTP 请求并验证响应（状态码、响应头、响应体）。
  - **工具**: JUnit 5, Spring Boot Test, IDE。
  - **广度**: 覆盖所有定义的 API 端点，包括不同的 HTTP 方法 (GET, POST, PUT, DELETE 等)、正常场景、异常场景（无效输入、权限不足、资源不存在等）。
  - **完成准则**: 所有集成测试用例通过，API 行为符合设计预期，无接口层面的严重缺陷。
- **测试用例设计**: 将采用等价类划分、边界值分析等方法设计测试用例。

8 测试项通过/失败准则

- **通过准则**:
  - 所有计划执行的单元测试和集成测试用例均已执行。
  - 所有测试用例执行结果为 "Pass"。
  - 发现的所有严重 (Critical) 和高 (High) 优先级的缺陷都已修复并经验证关闭。
  - 没有影响核心功能或导致系统崩溃的未解决缺陷。
- **失败准则**:
  - 存在一个或多个未解决的严重 (Critical) 或高 (High) 优先级缺陷。
  - 核心功能模块的测试用例失败率高于 10%。
  - 单元测试覆盖率远低于预定目标且无合理解释。
  - 测试过程中发现重大设计缺陷或需求偏差。
  - 关键 API 接口无法按预期工作。

9 暂停准则和继续准则

- **暂停准则**:
  - 发现阻塞性缺陷 (Blocker Bug)，导致大部分后续测试无法进行。
  - 测试环境不稳定或不可用，严重影响测试执行效率和结果准确性。
  - 测试版本构建失败或存在严重问题，无法部署或运行基本功能。
  - 需求发生重大变更，导致当前测试用例和计划失效。
- **继续准则**:
  - 导致暂停的问题（如阻塞性缺陷、环境问题、构建失败）已得到解决，并经验证。
  - 若因需求变更暂停，则测试计划和用例已根据新需求更新并评审。
  - 开发团队确认已修复相关问题并提供新的测试版本。
  - **重新进行的测试活动**: 重新执行之前失败或被阻塞的测试用例；执行受变更影响区域的回归测试；根据风险评估决定是否需要执行更大范围的回归测试。

10 测试交付物

- **软件测试计划** (本文档)

- **测试用例设计文档**
- **单元测试脚本** (JUnit Java 代码)
- **集成测试脚本** (JUnit Java 代码)
- **测试总结报告** (包含测试执行情况、结果分析、覆盖率、缺陷统计、风险评估和通过/失败结论)

11 测试任务

1. **测试计划制定**: 编写和评审本文档。 (责任人: 郭光灿)
2. **测试环境准备**: 配置和验证单元测试及集成测试所需的环境和工具。(责任人: 王驭风、傅文杰、韩创意、侯一夫)
3. **测试设计**:
    - 设计单元测试用例。(责任人: 韩创意、侯一夫)
    - 设计集成测试用例。(责任人: 王驭风、傅文杰)
4. **测试脚本开发**:
    - 编写单元测试代码。(责任人: 韩创意、侯一夫)
    - 编写集成测试代码。(责任人: 王驭风、傅文杰)
5. **测试执行**:
    - 执行单元测试。(责任人: 韩创意、侯一夫)
    - 执行集成测试。(责任人: 王驭风、傅文杰)
6. **测试总结**: 生成并提交测试报告。(责任人: 郭光灿)

- **所需技能**: Java 编程, JUnit 5, Spring Boot Test , REST API 理解, 测试用例设计方法, 数据库基础。

12 环境需求

- **硬件**:
    - 开发人员/测试人员工作站: 至少 8GB RAM, 256GB SSD, i5 处理器
- **软件**:
    - 操作系统: Windows
    - JDK: Java 17
    - IDE: IntelliJ IDEA or VS Code
    - 测试框架: JUnit 5, Spring Boot Test

- **其他**:
  - 项目源代码访问权限。
  - 需求文档、设计文档访问权限。

13 责任

- **测试工程师**: 负责测试设计、测试用例编写、测试脚本开发、测试执行、缺陷报告和初步验证、测试报告编写。

14 人手和培训的需要

- **人手**: 5 名测试工程师共同负责。
- **技能需求**:
  - 熟练掌握 Java 语言。
  - 精通 JUnit 5 测试框架。
  - 熟悉 Spring Boot Test 进行集成测试。
  - 理解 RESTful API 原理和测试方法。
  - 掌握等价类、边界值等测试用例设计技术。

15 时间表

- **测试计划完成**: 2025-03-25
- **测试环境准备完成**: 2025-03-26
- **单元测试设计与脚本开发**: 2025-04-01 至 2025-04-07
- **集成测试设计与脚本开发**: 2025-04-01 至 2025-04-08
- **测试执行 (单元+集成)**: 2025-04-09
- **测试总结报告完成**: 2025-04-08
- **关键里程碑**:
  - 测试计划评审通过: 2025-03-25
  - 单元测试脚本完成: 2025-04-07
  - 集成测试脚本完成: 2025-04-08
  - 最终测试通过: 2025-04-09

16 风险以及应急措施

1. **风险**: 开发交付延期或代码质量低下，影响测试进度。
应急措施: 加强沟通，尽早介入代码评审；调整测试优先级，先测核心稳定模块。

2. **风险**: 需求理解不充分或需求频繁变更，导致测试用例失效。
应急措施: 加强需求评审参与度；与产品、开发人员保持密切沟通，及时澄清疑问；建立需求变更管理流程，评估变更对测试的影响并调整计划。

3. **风险**: 测试人员技能不足或资源短缺。
应急措施: 提供针对性培训或知识分享；实施结对测试；调整测试范围或优先级；申请外部资源支持。

4. **风险**: 测试覆盖率不足，可能遗漏缺陷。
应急措施: 定期监控代码覆盖率报告；评审测试用例，确保关键路径和逻辑被覆盖；采用多种测试设计方法；进行探索性测试补充。

5. **风险**: 难以获取或构造有效的测试数据。
应急措施: 开发测试数据生成工具/脚本。

17 授权

姓名 (Name)	职务 (Title)	日期 (Date)
郭光灿	测试报告编写	2025/3/25
傅文杰	测试用例编写	2025/3/25
王驭风	测试脚本开发	2025/3/25
韩创意	测试用例编写	2025/3/25
侯一夫	测试脚本开发	2025/3/25