Meethere 性能测试计划

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [ 、、] 正式发布  [ ] 正在修改 | 文件标识： | 性能测试计划 |
| 当前版本： | 3.0 |
| 作 者： | 杨晓妍 |
| 完成日期： | 2019-12-24 |

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
| 0.1 | 杨晓妍 | 陈帆、甘丽蓉、项逸雯 | 2019/12/20 | 初稿 |
| 0.2 | 杨晓妍 | 陈帆、甘丽蓉、项逸雯 | 2019/12/22 | 独立场景和混合场景 |
| 0.3 | 杨晓妍 | 陈帆、甘丽蓉、项逸雯 | 2019/12/24 | 峰值场景 |

目录

[1 项目概要介绍 4](#_Toc509511835)

[1.1 项目简介 4](#_Toc509511836)

[1.2 项目成员 4](#_Toc509511837)

[1.3 测试范围 4](#_Toc509511838)

[1.3.1 范围内 4](#_Toc509511839)

[1.3.2 范围外 4](#_Toc509511840)

[1.4 前提假设 4](#_Toc509511841)

[1.5 测试目标 5](#_Toc509511842)

[2 性能测试策略 5](#_Toc509511843)

[2.1 性能测试模型 5](#_Toc509511844)

[2.2 性能测试场景 6](#_Toc509511845)

[2.3 重点测试策略 6](#_Toc509511846)

[2.3.1 重点测试原则 6](#_Toc509511847)

[2.3.2 重点测试功能操作 7](#_Toc509511848)

[3 测试案例设计 7](#_Toc509511849)

[3.1 生产压力分析 7](#_Toc509511850)

[3.2 场景通过标准 7](#_Toc509511851)

[3.3 测试场景设计 7](#_Toc509511852)

[3.3.1 独立场景 7](#_Toc509511853)

[具体的独立场景测试用例如下所示： 8](#_Toc509511854)

[3.3.2 混合场景 10](#_Toc509511855)

[3.3.3 峰值场景 11](#_Toc509511856)

[4 测试实施安排 12](#_Toc509511857)

[4.1 测试进度 12](#_Toc509511858)

[4.2 测试流程 12](#_Toc509511859)

[4.3 测试报告需求 12](#_Toc509511860)

[4.4 性能缺陷管理 13](#_Toc509511861)

[5 性能测试标准 13](#_Toc509511862)

[5.1 启动标准 13](#_Toc509511863)

[5.2 中止标准 13](#_Toc509511864)

[5.3 通过标准 13](#_Toc509511865)

[6 测试环境规划 13](#_Toc509511866)

[6.1 部署环境 13](#_Toc509511867)

[6.2 执行环境 14](#_Toc509511868)

[7 测试风险分析 14](#_Toc509511869)

[8 角色与职责 15](#_Toc509511870)

# 项目概要介绍

## 项目简介

共享经济近年来在国内逐步兴起。缺失随时提供预约服务的管理员是共享经济产业的短板。场馆共享作为共享经济中的重要板块，存在相同的问题。因此，基于在线网站的预约系统应运而生，催生了共享经济的信息化建设和电子管理的大规模发展。

MeetHere是一个场馆预约的电子商务网站，功能分为用户和管理员两个模块。对于用户而言，可以不去实体场馆就能方便地预约运动场馆。该软件还提供场馆评论、新闻公告和个人信息维护功能。对于管理员而言，分为区域管理、场馆管理、用户管理、订单管理、新闻管理三大功能板块。管理员可以对五个对象进行增删改查操作，还可以审核预约订单。通过MeetHere在线预约系统，用户和管理员可以实现24小时异步预约场馆。

## 项目成员

陈帆、甘丽蓉、项逸雯、杨晓妍

## 测试范围

### 范围内

MeetHere的前端业务

用户模块：

登录注册、预约场馆、维护个人信息、查看新闻、注销

管理员模块：

登录、管理区域、场馆、用户、订单、留言、新闻

### 范围外

MeetHere的后端业务

包括用户管理、管理员管理、场馆管理、区域管理、订单管理、图片管理、新闻管理、留言管理等业务功能

## 前提假设

被测系统完成功能测试，测试报告显示达到要求。

## 测试目标

1) 系统在单步操作响应时间方面，能满足用户当前及未来1到3年的发展需求；

2) 系统的批量作业运行稳定，处理能力能满足用户当前及未来1到3年的发展需求；

3) 在业务处理能力方面，能满足当前及未来1到3年的业务增长需求；

4) 发现并解决宕机，内存泄漏等严重问题，使系统具备良好的稳定性、健壮性；

5) 系统在资源使用方面比较合理，各项资源平均利用率在30%左右。

# 性能测试策略

与其它测试类型一样，性能测试周期分为：性能测试需求分析、性能测试设计、性能测试实现和性能测试报告四个存在迭代的阶段。其中，性能测试需求和性能测试设计由测试人员手工完成，而性能测试实现需借助性能测试工具达成，通过对性能测试报告的分析判定系统是否存在性能缺陷。

## 性能测试模型

性能测试模型定义了性能测试涉及的各个待测方面，是性能测试设计的指南。

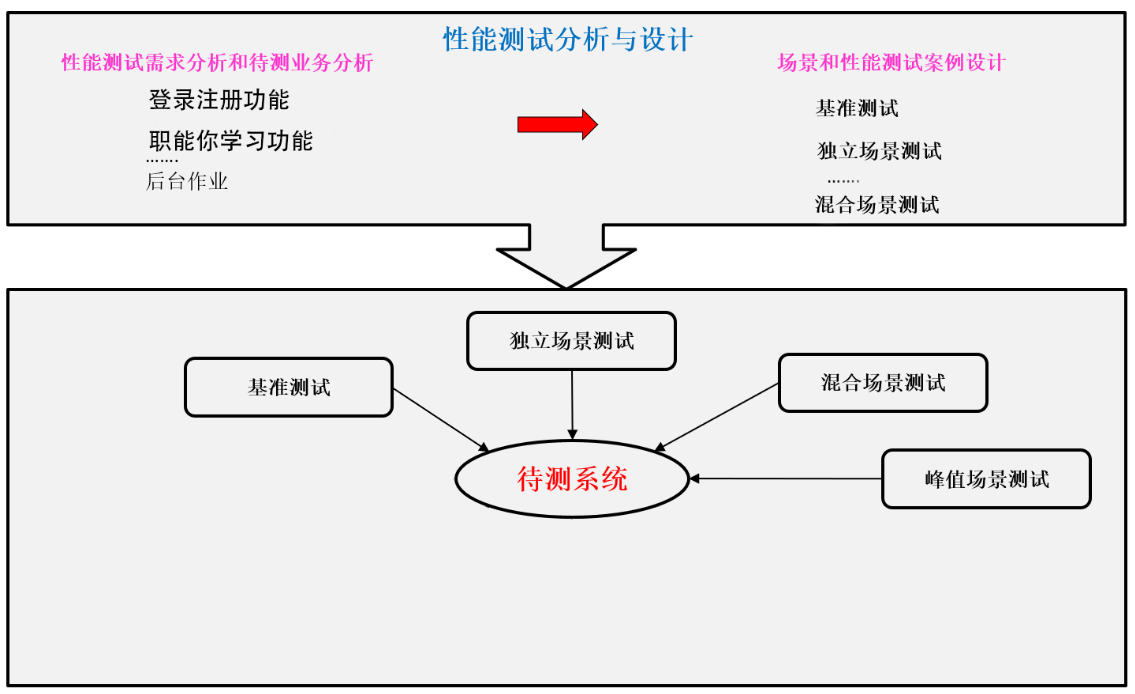


Figure 1 性能测试模型

根据Figure 1所示的性能测试模型，可以设计如Table 1所示的性能测试类型

Table 1性能测试类型

|  |  |  |  |
| --- | --- | --- | --- |
| **测试种类** | **测试方法** | **测试内容** | **场景类型** |
| 性能测试 | 以系统最大并发用户数的1-2倍作为上限对关键功能操作进行梯度加压测试，在压力时间内通过的功能操作量应接近峰值时段的功能操作量，甚至超过系统全天的功能操作量 | 核心业务，需求分析中发现的容易出现性能问题的业务 | 独立场景 |
| 压力测试 | 以系统预期最大并发用户数的1-2倍作为上限对常用/关键功能操作进行混合场景梯度加压测试，在压力时间内通过的功能操作量应接近峰值时段的功能操作量，甚至超过系统全天的功能操作量 | 存在一定关联关系的常用/核心业务 | 混合场景 |
| 峰值测试 | 按照系统峰值时预期最大并发用户数的2-3倍对峰值时段涉及的核心功能操作进行的混合场景梯度加压测试，在测试时间内通过的功能操作数量应接近或超过系统全天的功能操作量 | 生产上峰值场景中的核心业务 | 峰值场景 |

## 性能测试场景

根据项目实际，确定本次性能测试场景，例如可包含以下场景的性能测试：

独立场景

混合场景

峰值场景

## 重点测试策略

### 重点测试原则

为了全面评估系统性能，本次重点测试策略安排如下：

1. 对关键功能操作开展全面的独立场景测试；
2. 对关键功能操作开展全面的峰值测试；
3. 在集成性能测试方面，对功能操作量大、容易出现性能瓶颈的模块，重点进行测试并安排专人在系统发布前一直跟进开发组的需求变更以便进行及时的回归性能测试。
4. 在渠道性能测试方面，对渠道场景按照业务量及重要性来划分优先级，按照优先级先后来执行，保证系统上线后关键业务的稳定性。

### 重点测试功能操作

# 测试案例设计

## 生产压力分析

对于本系统的开发

## 场景通过标准

对于不同的测试场景，我们需要制定不同的通过标准才能顺利完成测试工作。具体要求如下表所示：

Table 2 场景通过标准

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **场景类型** | **单步操作响应时间** | **加压时间** | **操作账户数据** | **处理业务笔数** | **事务成功率** | **各后台利用率** | **并发/在线上限** | **在线思考时间设置** |
| 独立场景 | <1秒 | 30分钟 | >500 | >500 | 100% | <50% | 100 | 无 |
| 混合场景 | <1秒 | 1小时 | >500 | >1000 | 100% | <80% | 200 | 无 |
| 峰值场景 | <2秒 | 1小时 | >500 | >1000 | 100% | <80% | 200 | 无 |

## 测试场景设计

### 独立场景

对于关键场景，将分别对其采用渐进式加压的方式来进行独立场景测试。每个场景测试多组并发、并发数从1逐步增加到200。对于具体的场景，测试几组并发依据案例执行。案例需要按照业务量设计。对于使用频度较低的场景，可以采用独立/混合场景的方式进行测试，即在背景压力下以小并发的方式来进行长时间疲劳测试。

Table 3 独立场景设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **并发数** | **响应时间要求** | **TPS要求** | **加压时间** | **加压方式** |
| 1 | <1秒 | N/A | 3分钟 | 直接加压 |
| 10 | <1秒 | N/A | 10分钟 | 1个用户/5秒 |
| 20 | <1秒 | N/A | 10分钟 | 1个用户/5秒 |
| 50 | <1秒 | >100 次/秒 | 20分钟 | 1个用户/5秒 |
| 100 | <1秒 | >100 次/秒 | 20分钟 | 1个用户/5秒 |
| 200 | <1秒 | >100 次/秒 | 10分钟 | 1个用户/1秒 |

### 具体的独立场景测试用例如下所示：

1. 用户

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **用例编号** | **场景步骤** | **步骤描述** | **输入** | **预期结果** |
| 1、 | 场景1:  用户注册 | 1、进入用户注册页面  2、输入用户名  3、输入密码  4、再次输入密码  5、点击“注册” | 用户名:  register1/  register2/  register3….  密码:123456 | 1. 显示提示：注册成功，请返回登录  2. 跳转至用户登录页面 |
| 2、 | 场景2:用户登录 | 1、进入用户登录页面  2、输入用户名  3、输入密码  4、点击“登录” | 用户名：example  密码：example | 1. 1.成功登录 2. 2.跳转至用户模块首页 |
| 3、 | 场景3:查看全部场馆 | 1、用户登录  2、点击场馆导航 |  | 跳转至场馆页面 |
| 4、 | 场景4:用户查看首页 | 1、用户登录  2、点击首页 |  | 跳转至首页 |
| 5、 | 场景4:查看新闻 | 1、用户登录  2、点击新闻导航  3、点击所有 |  | 跳转至新闻页 |
| 6、 | 场景5:查看订单 | 1、用户登录  2、点击个人中心导航  3、点击我的订单 |  | 跳转至我的订单页面 |

2、管理员

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **用例编号** | **场景步骤** | **步骤描述** | **输入** | **预期结果** |
| 1、 | 场景1:成功登录 | 1、进入管理员登录页面  2、输入用户名  3、输入密码  4、点击“登录” | 用户名:example  密码:example | 1. 1.成功登录 2. 2.跳转至管理员首页 |
| 2、 | 场景2:查找区域 | 1、管理员登录  2、输入区域名  3、点击搜索 | 区域名：区域 | 显示区域名包含输入的区域 |
| 3、 | 场景3:新增区域 | 1. 1. 2. 2.   3、输入区域名  4、点击提交 | 区域名：district1/district2/district3… | 区域新添区域一/区域二/区域三… |
| 4、 | 场景4:新增场馆 | 1、管理员登录  2、点击区域管理  3、点击场馆管理  4、输入场馆名称  5、输入租金  6、输入容量  7、输入营业时间  8、点击提交 | 场馆名称：test  租金：10  容量10  营业时间:7:00~9:00 | 场馆列表增加，ID递增 |
| 5、 | 场景5：查找用户 | 1. 1.管理员登录 2. 2.点击用户管理 3. 3.输入用户名 4. 4.点击搜索 | 用户名：a | 显示用户名包含“a”的用户 |
| 6、 | 场景6:发布新闻 | 1. 1.管理员登录 2. 2.点击新闻管理 3. 3.输入新闻名 4. 4.点击搜索 | 新闻名：新闻 | 显示标题包含“新闻”的新闻 |
| 7、 | 场景7:查看订单 | 1. 1.管理员登录 2. 2.点击订单管理 |  | 进入订单页面 |

### 混合场景

对于相关联的关键功能操作，对其采用渐进式加压的方式来进行混合场景测试，以测试这些关联功能操作是否存在性能问题。每个场景测试多组并发，并发数逐步增加，对于不同的场景，增加的数量与峰值有所不同。对于具体的场景，测试几组并发依据案例来执行（案例主要依据业务量来设计）。对于使用频度较低的场景，可以采用独立/混合场景的方式进行测试，即在背景压力下以小并发的方式来进行长时间疲劳测试。

具体场景内的操作流程包括：

1、用户注册—>用户登录

2、用户登录—>用户点击查看场馆信息->用户查看首页

3、用户登录->用户查看订单—>用户点击查看场馆信息

4、用户登录->用户查看首页—>用户查看新闻信息

5、管理员登录—>管理员新增区域—>管理员新增场馆

6、管理员登录—>管理员查找用户

7、管理员登录—>管理员查看新闻—>管理员发布新闻

对于1，2，6等操作流程，在系统中出现率非常高，采用从20到200逐渐加压的方式。

对于3，7等操作流程，在系统中出现率较高，采用20到100的加压方式。

对于4，5等操作流程，在系统中出现率较低，采用20到50的加压方式。

保证测试的顺利进行并且有效利用资源。

Table 4 混合场景设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **并发数** | **响应时间要求** | **TPS要求** | **加压时间** | **加压方式** |
| 20 | <1秒 | N/A | 10分钟 | 1个用户/5秒 |
| 50 | <1秒 | >100 /秒 | 20分钟 | 1个用户/5秒 |
| 100 | <2秒 | >100/秒 | 20分钟 | 1个用户/5秒 |
| 200 | <2秒 | >100/秒 | 10分钟 | 1个用户/1秒 |

### 峰值场景

通过分析生成数据，多数系统日间各个峰值共有两个时段，上午10:00-11:00的1个小时；下午14:00-15:00点的1个小时，因此，峰值最长持续时间1小时。考虑到系统的扩展性及实际情况，峰值测试共测试二组：第一组测试正常峰值压力的处理能力；第二组测试4倍峰值压力下的系统处理能力，以满足未来的业务发展需求，测试时各组中的交易按照业务比例同时加载。

第一组：先用生成峰值时压力的70%加压1小时作为基准压力，之后采用当前生产上的实际峰值压力并发进行加压。

Table 5 峰值场景设计1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **并发数** | **响应时间要求** | **TPS要求** | **加压时间** | **加压方式** |
| 生产压力的70% | <1秒 | >100/秒 | 1小时 | 1个用户/5秒 |
| 生产压力 | <1秒 | >100/秒 | 4小时 | 1个用户/5秒 |

第二组：分别采用生产压力4倍来进行加压

Table 6 峰值测试设计2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **并发数** | **响应时间要求** | **TPS要求** | **加压时间** | **加压方式** |
| 生产压力的4倍 | <1秒 | >100/秒 | 5小时 | 1个用户/5秒 |

对于性能较好的系统，可以直接加压到最大并发数，持续加压5小时来测试系统的峰值处理能力。

# 测试实施安排

## 测试进度

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编号** | **任务** | **开始日期** | **结束日期** | **参与人员** |
| 1 | 熟悉被测试系统，确定典型事务 | 2019.12.12 | 2018.12.17 | 测试人员  开发人员  项目经理 |
| 2 | 搭建测试环境，  录制典型事务的脚本，增强脚本 | 2019.12.17 | 2019.12.20 | 测试人员 |
| 3 | 执行测试并收集相关数据 | 2019.12.21 | 2019.12.27 | 测试人员 |
| 4 | 数据分析，编写测试报告 | 2019.12.21 | 2019.12.28 | 测试人员 |

## 测试流程

流程分为需求分析、设计、实现和测试报告四个阶段。

性能测试需求分析与设计：

熟悉被测试系统，定义性能测试要求，例如并发用户的数量、典型业务流程和所需响应时间。

性能测试实现：

创建虚拟用户脚本：将最终用户活动捕获到自动脚本中。

定义场景：使用 LoadRunner Controller 设置负载测试环境。

运行场景：通过 LoadRunner Controller 驱动、管理和监控负载测试。

生成性能测试报告：

使用 LoadRunner Analysis 创建图和报告并评估性能。

## 测试报告需求

Table 9 测试报告需求

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **报告类型** | **响应时间要求** | **报告者** | **接受者** | **报告内容** |
| 邮件 | 每日 | 各渠道测试组负责人 | 测试中心，项目经理 | 每日结果总结报告 |
| 邮件 | 每日 | 各渠道测试组负责人 | 测试中心，项目经理 | 每日测试结果记录 |
| 邮件 | 每日 | 性能测试负责人 | 测试中心，项目经理 | 周报 |
| 文档 | 各测试阶段结束 | 各渠道测试组负责人 | 测试中心，项目经理 | 性能测试报告 |

## 性能缺陷管理

测试过程采用缺陷报告进行缺陷管理

# 性能测试标准

## 启动标准

1. 测试环境满足计划需求
2. 基准参数配置完成校验
3. 关键功能操作通过冒烟测试

## 中止标准

1. 测试环境或关键系统不可用
2. 测试环境距生产标准差距太大
3. 缺陷周转周期不符合规定的时间
4. 出现宕机、不响应等严重的性能问题
5. 系统的功能操作成功率低于95%

## 通过标准

系统上线至少满足下面标准：

1. 系统无宕机、不响应类的严重性能问题
2. 系统响应时间80%达到系统的期望值
3. 系统的业务吞吐量达到预期目标，即当前生产需求的3倍以上
4. 柜台类系统要求通过7\*12小时以上的疲劳强度测试
5. 电子渠道类系统要求通过7\*24小时以上的疲劳强度测试

# 测试环境规划

## 部署环境

测试环境要求尽量和真实环境相同，至少要求服务器配置和网络带宽和拓扑结构应该相似。主要内容：服务器数量和配置，操作系统和数据库版本，软硬件部署等。

|  |  |  |  |
| --- | --- | --- | --- |
| **软件环境（相关软件、操作系统等）** | | | |
| **名称** | **版本** | **数量** | **获得途径** |
| MAC OSX | 10.15.1 | 1 | / |
| **硬件环境（网络、设备等）** | | | |
| **名称** | **版本** | **数量** | **获得途径** |
| Web服务器 | Chrome 79.0 | 1 | / |
| 数据库服务器 | Mysql 8.0.18 | 1 | / |
| 数据库服务器 | Redis 5.0.5 | 1 | / |

## 执行环境

|  |  |  |  |
| --- | --- | --- | --- |
| **软件环境（相关软件、操作系统等）** | | | |
| **名称** | **版本** | **数量** | **获得途径** |
| MAC OSX | 10.15.1 | 1 | / |
| Locust | 最新版本 | 1 | https://www.locust.io/ |
| **硬件环境（网络、设备等）** | | | |
| **名称** | **版本** | **数量** | **获得途径** |
| Web服务器 | Chrome 79.0 | 1 | / |
| 数据库服务器 | Mysql 8/0.18 | 1 | / |
| 前台客户机 | Redis 5.0.5 | 1 | / |

# 测试风险分析

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **风险因素** | **可能结果** | **可能发生时间** | **风险**  **级别** | **应对措施** |
| 工具缺陷 | 测试工具和监控工具无法全部支持信贷业务系统的测试和监控 | 随时 | 中 | 评估被测系统，分析所有需求。  通过其它工具实现对需求的支持程度。 |
| 测试数据的准备备份及恢复无法正常完成 | 测试过程中数据用尽或不满足测试需求，将导致测试无法实施。 | 测试执行时 | 高 | 运维方配合完成数据的准备、备份和恢复 |
| 测试环境有其他用户连接进行操作，服务器产生性能缺陷 | * 1. 测试方获得最大负载压力与实际最大负载有差距   2. 服务器出现性能缺陷的现象，运维方定位性能缺陷模块并非真正性能缺陷的模块 | 测试执行时 | 高 | 测试方进行负载测试时，保证测试环境无其他连接和用户操作 |
| 测试服务器访问状态不稳定 | 测试准备和测试执行中断，测试计划时间延后 | 随时 | 高 | 保证测试期间测试环境访问畅通 |

# 角色与职责

|  |  |  |  |
| --- | --- | --- | --- |
| 前端开发/项目经理 | 项逸雯 | 项目管理和协调  前端开发 | 1. 组织协调能力 2. 熟悉JS,CSS,HTML 3. 设计并实现前端页面 |
| 后端开发 | 陈帆 | 后端开发 | 1. 熟悉Springboot开发框架 2. 熟悉应用业务 |
| 后端测试 | 甘丽蓉 | 单元测试  集成测试 | 1. 熟悉jacoco，FingBugs 2. 编写单元测试脚本 3. 编写继承测试脚本 |
| 前端测试 | 杨晓妍 | 系统测试  性能测试 | 1. 熟悉Selenium、postman、locust、webdriver 2. 手工测试系统功能 3. 编写性能测试脚本 |