

**《软件测试》**

**期末课程设计**

**同济基金交易系统**

**系统测试文档**

小组成员（按学号排列）

2252635 王 玥

2252700 李一鑫

2253206 韩明洋

2254198 段子涛

目录

[一、引言 2](#_Toc201353273)

[1.1 目的 2](#_Toc201353274)

[1.2测试环境 2](#_Toc201353275)

[1.3术语表 2](#_Toc201353276)

[二、项目概述 3](#_Toc201353277)

[2.1项目目的 3](#_Toc201353278)

[2.2总用例图 3](#_Toc201353279)

[2.3系统体系架构设计 4](#_Toc201353280)

[三、功能测试 5](#_Toc201353281)

[3.1测试用例总表 5](#_Toc201353282)

[3.1.1产品管理子系统测试用例 5](#_Toc201353283)

[3.1.2基金账户管理子系统测试用例 6](#_Toc201353284)

[3.1.3申购赎回子系统测试用例 7](#_Toc201353285)

[3.1.4清算子系统测试用例 7](#_Toc201353286)

[3.1.5业务查询子系统测试用例 8](#_Toc201353287)

[3.1.6登录子系统测试用例 9](#_Toc201353288)

[3.2测试 9](#_Toc201353289)

[四、非功能测试 10](#_Toc201353290)

[4.1测试目的 10](#_Toc201353291)

[4.2 测试工具 10](#_Toc201353292)

[4.3测试需求分析 10](#_Toc201353293)

[4.4测试方案设计 11](#_Toc201353294)

[4.4.1 50线程压力测试 11](#_Toc201353295)

[4.4.2 100线程压力测试 13](#_Toc201353296)

[4.4.3 120线程压力测试： 15](#_Toc201353297)

[4.5测试结论 17](#_Toc201353298)

# 一、引言

## 1.1 目的

本文是描述同济基金交易系统的系统测试的大纲，主要描述如何进行系统测试活动，如何控制系统测试活动，系统测试活动的流程以及系统测试活动的工作安排等。保证程序部署完成后，能够正常完成系统功能以及基本业务流能够正常实现。

## 1.2测试环境

测试环境：独立测试服务器

数据库：MySQL 8.0

浏览器：Chrome/Firefox

Java环境：JDK 11

操作系统环境： Windows10

输入习惯： 中文 & 英文

## 1.3术语表

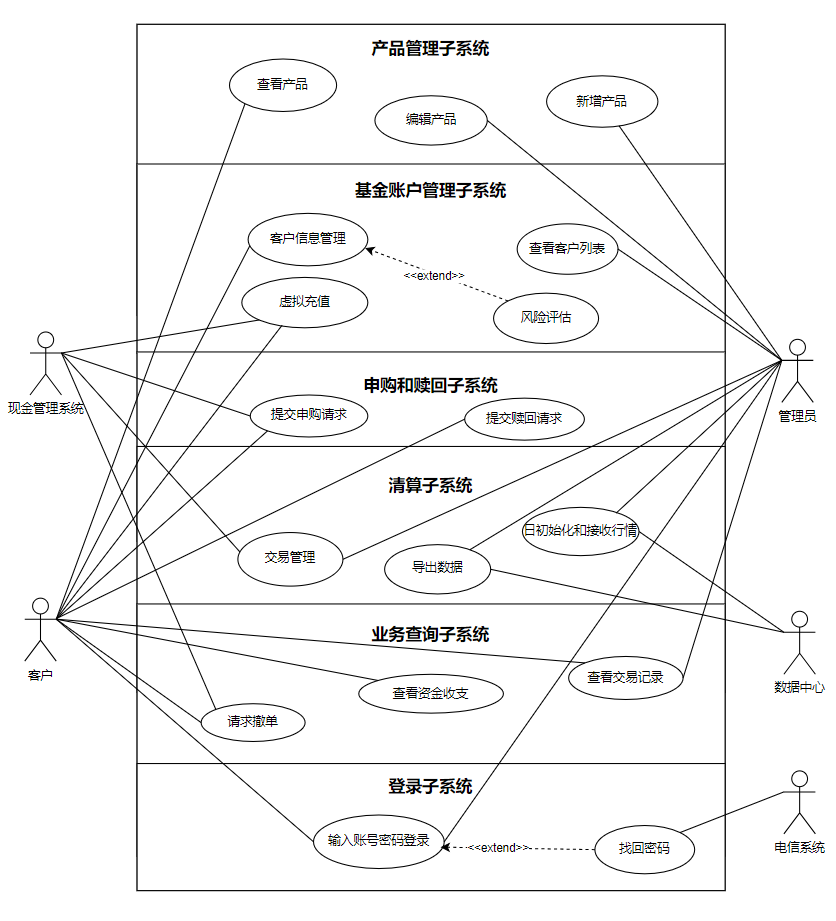
|  |  |
| --- | --- |
| **术语** | **定义** |
| **基金** | 集合投资工具，汇集投资者资金进行多元化证券投资 |
| **申购** | 投资者购买基金份额的行为 |
| **赎回** | 投资者卖出基金份额的行为 |
| **基金净值** | 每份基金份额对应的资产净值，计算公式：基金资产总值/基金总份额 |
| **清算** | 基金公司对当日交易进行确认、资金结算和份额登记的过程 |
| T**日** | 交易日（Trade Day），投资者提交交易申请的日期 |
| T+1 | 交易日的下一个工作日，基金公司完成交易确认的日期 |
| **风险等级** | 评估投资者风险承受能力的分级体系（保守型、稳健型、进取型等） |
| **持仓** | 投资者当前持有的基金份额 |

# 二、项目概述

## 2.1项目目的

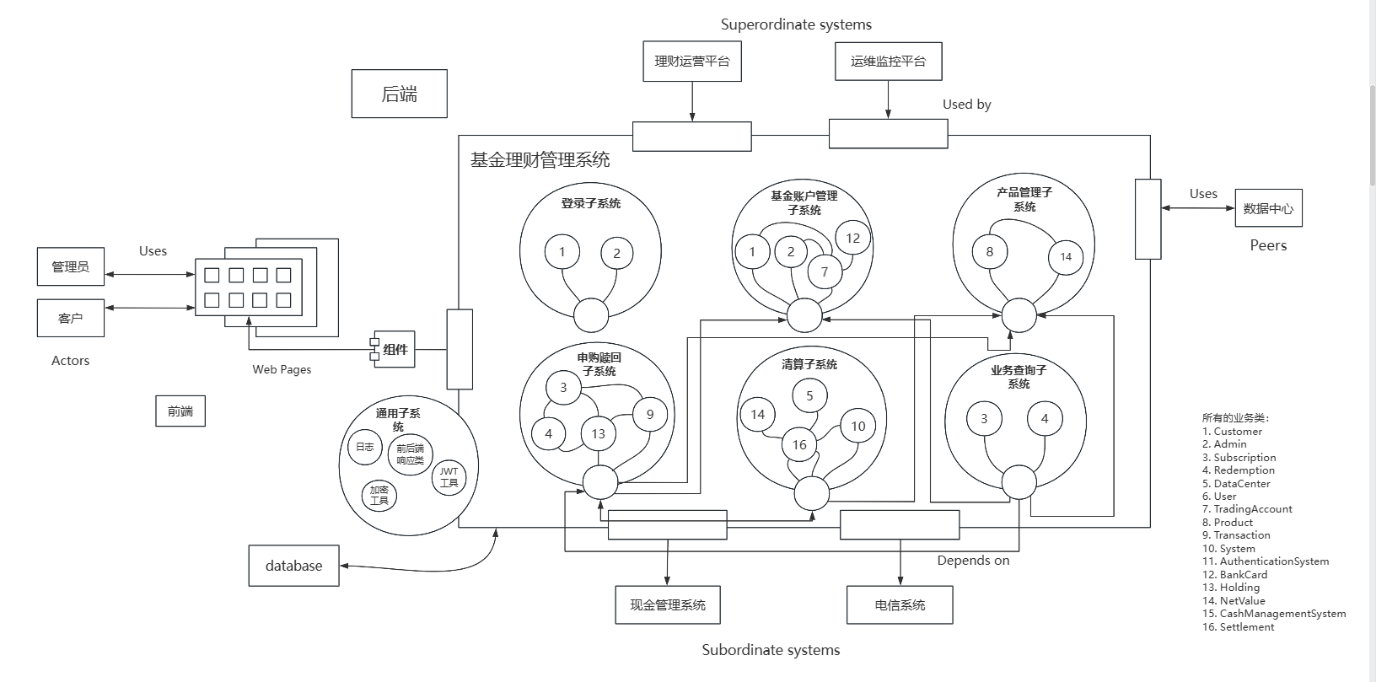
基金交易系统旨在为基金产品的持有者和管理员提供一个高效、便捷的操作平台，实现基金产品管理、开户、申购、赎回、清算及查询等功能。通过该系统，客户可以方便地完成基金交易操作，管理员则能确保交易的安全和准确，提升管理效率。

## 2.2总用例图

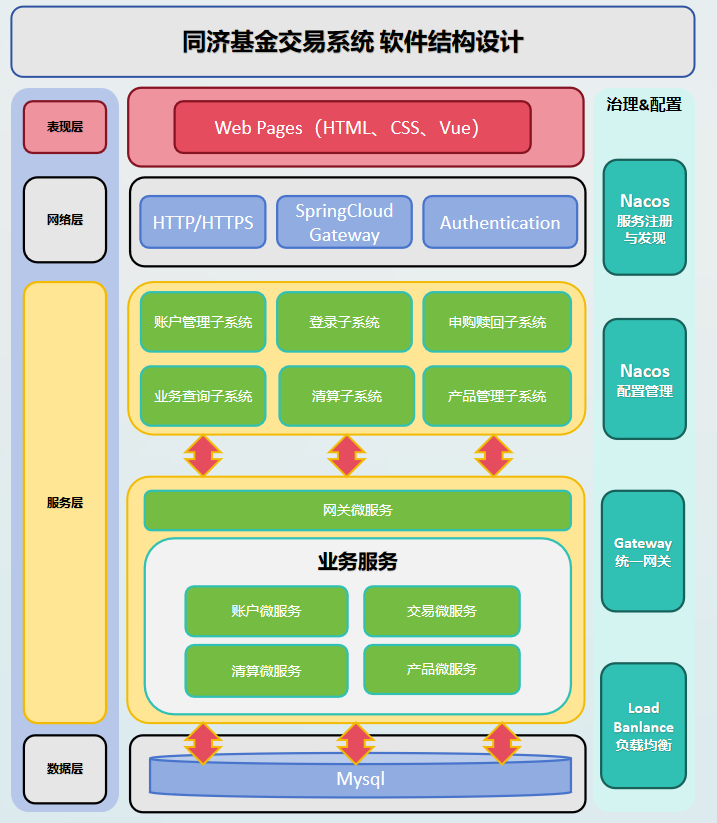


我们将同济基金交易系统分成了六个子系统：产品管理子系统、基金账户管理子系统、申购和赎回子系统、清算子系统、业务查询子系统、登录子系统。并采用微服务架构对应不同的子系统细分成不同的微服务。我们在先不同子系统进行系统功能测试，然后再对整个系统进行非功能测试。

## 2.3系统体系架构设计



这是同济基金交易系统的体系架构，包括了上游系统、下游系统、同级系统，系统中的类及其中联系。系统间和类间的接口在集成测试已经做过，此处我们不进行过多赘述，现在我们对于途中长方形中的主要系统进行功能测试和非功能测试。



表现层由 HTML、CSS 和 Vue 构建的 Web 页面组成，负责与用户进行交互。用户通过浏览器访问这些页面，进行各种操作，如查看账户信息、进行交易等。

网络层通过 HTTP/HTTPS 协议进行通信，确保数据在网络上传输的安全性。Spring Cloud Gateway 用于请求路由和认证，确保请求被正确地转发到相应的服务，并进行必要的身份验证。

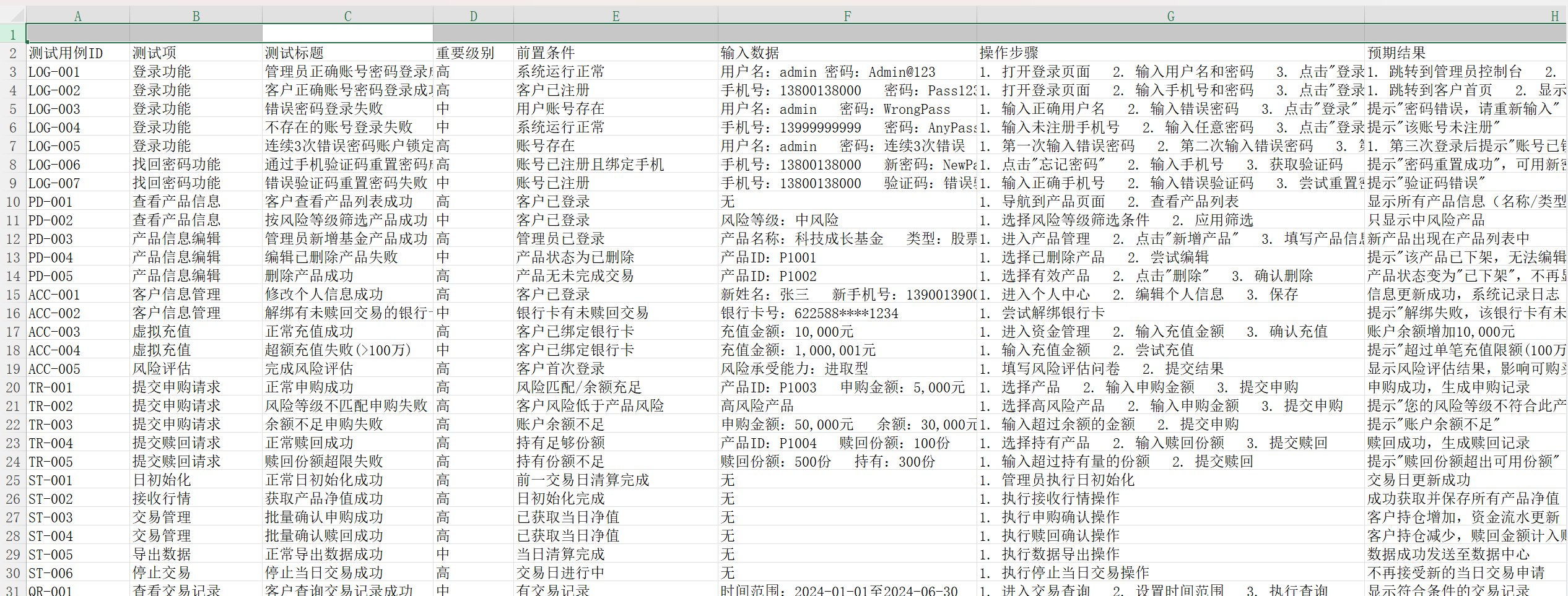
服务层包含多个子系统，每个子系统作为独立的微服务运行，处理特定的业务逻辑。这些子系统包括账户管理、登录、申购赎回、业务查询、清算和产品管理等。微服务架构使得各个子系统可以独立开发、部署和扩展，提高了系统的灵活性和可维护性。

数据层使用 MySQL 数据库进行数据存储和管理，确保数据的持久性和一致性。数据库为各个微服务提供数据支持，满足系统的业务需求。

# 三、功能测试

针对六个子系统，我们设计了全面的测试用例集，覆盖所有子系统的功能点和非功能需求。用例共38个，以下是测试用例的Excel格式汇总：（以下所有表格在附件中，截图作为展示）

## 3.1测试用例总表



### 3.1.1产品管理子系统测试用例

**功能需求：**

产品管理子系统允许客户和管理员共同参与产品的生命周期管理。客户可以浏览产品列表，了解每种产品的基本信息及其收益情况。对于感兴趣的产品，客户可进一步查阅详细资料，如规格参数、产品价格等。而作为后台管理者，除了拥有上述查看产品权限外，还可执行增加新产品、修改已有条目属性值及删除过时产品等功能。如此一来，不仅满足了消费者需求变化快的特点，也便于商家及时调整库存结构以适应市场竞争态势。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 测试用例ID | 测试项 | 测试标题 | 重要级别 | 前置条件 | 输入数据 | 操作步骤 | 预期结果 |
| PD-001 | 查看产品信息 | 客户查看产品列表成功 | 高 | 客户已登录 | 无 | 1. 导航到产品页面 2. 查看产品列表 | 显示所有产品信息（名称/类型/收益率） |
| PD-002 | 查看产品信息 | 按风险等级筛选产品成功 | 中 | 客户已登录 | 风险等级：中风险 | 1. 选择风险等级筛选条件 2. 应用筛选 | 只显示中风险产品 |
| PD-003 | 产品信息编辑 | 管理员新增基金产品成功 | 高 | 管理员已登录 | 产品名称：科技成长基金 类型：股票型 风险等级：高 | 1. 进入产品管理 2. 点击"新增产品" 3. 填写产品信息 4. 保存 | 新产品出现在产品列表中 |
| PD-004 | 产品信息编辑 | 编辑已删除产品失败 | 中 | 产品状态为已删除 | 产品ID：P1001 | 1. 选择已删除产品 2. 尝试编辑 | 提示"该产品已下架，无法编辑" |
| PD-005 | 产品信息编辑 | 删除产品成功 | 高 | 产品无未完成交易 | 产品ID：P1002 | 1. 选择有效产品 2. 点击"删除" 3. 确认删除 | 产品状态变为"已下架"，不再显示在客户视图 |

### 3.1.2基金账户管理子系统测试用例

**功能需求：**

基金账户管理子系统是一个综合性的财务管理平台，旨在帮助客户高效地管理他们的基金账户。客户可以通过客户信息管理模块查看和管理自己的个人信息，包括姓名、联系方式、地址等基本资料。在银行卡管理模块下，客户可以绑定多张银行卡并与之关联，方便日后进行资金转入转出操作。同时，系统还会定期提醒客户关注银行卡的状态变化，避免因未及时处理而导致的问题出现。管理员有权访问查看客户列表功能，用于监视所有客户的活动情况，确保每一位客户都能够得到适当的服务和支持。虚拟充值模块是一个模拟货币充值的过程，目的是测试系统性能并收集反馈意见。客户可以在个人信息界面选择对自己的账户进行风险评估。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 测试用例ID | 测试项 | 测试标题 | 重要级别 | 前置条件 | 输入数据 | 操作步骤 | 预期结果 |
| ACC-001 | 客户信息管理 | 修改个人信息成功 | 高 | 客户已登录 | 新姓名：张三 新手机号：13900139000 | 1. 进入个人中心 2. 编辑个人信息 3. 保存 | 信息更新成功，系统记录日志 |
| ACC-002 | 客户信息管理 | 解绑有未赎回交易的银行卡失败 | 中 | 银行卡有未赎回交易 | 银行卡号：622588\*\*\*\*1234 | 1. 尝试解绑银行卡 | 提示"解绑失败，该银行卡有未赎回交易" |
| ACC-003 | 虚拟充值 | 正常充值成功 | 高 | 客户已绑定银行卡 | 充值金额：10,000元 | 1. 进入资金管理 2. 输入充值金额 3. 确认充值 | 账户余额增加10,000元 |
| ACC-004 | 虚拟充值 | 超额充值失败(>100万) | 中 | 客户已绑定银行卡 | 充值金额：1,000,001元 | 1. 输入充值金额 2. 尝试充值 | 提示"超过单笔充值限额(100万元)" |
| ACC-005 | 风险评估 | 完成风险评估 | 高 | 客户首次登录 | 风险承受能力：进取型 | 1. 填写风险评估问卷 2. 提交结果 | 显示风险评估结果，影响可购买产品范围 |

### 3.1.3申购赎回子系统测试用例

**功能需求：**

支持客户选择产品输入申购金额并提交申购请求，系统判断客户风险、用户余额并记录申购相关信息并显示申购请求成功信息。

支持客户选择产品并输入赎回份额提交赎回请求，系统确保客户赎回份额合法后记录相关信息并显示赎回请求成功信息。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 测试用例ID | 测试项 | 测试标题 | 重要级别 | 前置条件 | 输入数据 | 操作步骤 | 预期结果 |
| TR-001 | 提交申购请求 | 正常申购成功 | 高 | 风险匹配/余额充足 | 产品ID：P1003 申购金额：5,000元 | 1. 选择产品 2. 输入申购金额 3. 提交申购 | 申购成功，生成申购记录 |
| TR-002 | 提交申购请求 | 风险等级不匹配申购失败 | 高 | 客户风险低于产品风险 | 高风险产品 | 1. 选择高风险产品 2. 输入申购金额 3. 提交申购 | 提示"您的风险等级不符合此产品要求" |
| TR-003 | 提交申购请求 | 余额不足申购失败 | 高 | 账户余额不足 | 申购金额：50,000元 余额：30,000元 | 1. 输入超过余额的金额 2. 提交申购 | 提示"账户余额不足" |
| TR-004 | 提交赎回请求 | 正常赎回成功 | 高 | 持有足够份额 | 产品ID：P1004 赎回份额：100份 | 1. 选择持有产品 2. 输入赎回份额 3. 提交赎回 | 赎回成功，生成赎回记录 |
| TR-005 | 提交赎回请求 | 赎回份额超限失败 | 高 | 持有份额不足 | 赎回份额：500份 持有：300份 | 1. 输入超过持有量的份额 2. 提交赎回 | 提示"赎回份额超出可用份额" |

### 3.1.4清算子系统测试用例

**功能需求：**

1.在新的工作日，管理员获取基金产品的新净值；

2.管理员对前一交易日的申购申请和赎回申请进行确认，完成相应的份额转换和金额入账；

3.管理员关闭当日交易的申请通道，开放下一交易日的申请通道；

4.管理员将所有交易处理数据导出到数据中心。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 测试用例ID | 测试项 | 测试标题 | 重要级别 | 前置条件 | 输入数据 | 操作步骤 | 预期结果 |
| ST-001 | 日初始化 | 正常日初始化成功 | 高 | 前一交易日清算完成 | 无 | 1. 管理员执行日初始化 | 交易日更新成功 |
| ST-002 | 接收行情 | 获取产品净值成功 | 高 | 日初始化完成 | 无 | 1. 执行接收行情操作 | 成功获取并保存所有产品净值 |
| ST-003 | 交易管理 | 批量确认申购成功 | 高 | 已获取当日净值 | 无 | 1. 执行申购确认操作 | 客户持仓增加，资金流水更新 |
| ST-004 | 交易管理 | 批量确认赎回成功 | 高 | 已获取当日净值 | 无 | 1. 执行赎回确认操作 | 客户持仓减少，赎回金额计入账户 |
| ST-005 | 导出数据 | 正常导出数据成功 | 中 | 当日清算完成 | 无 | 1. 执行数据导出操作 | 数据成功发送至数据中心 |
| ST-006 | 停止交易 | 停止当日交易成功 | 高 | 交易日进行中 | 无 | 1. 执行停止当日交易操作 | 不再接受新的当日交易申请 |

### 3.1.5业务查询子系统测试用例

**功能需求：**

1.管理员或客户查看基金产品的申购、赎回申请记录和撤单记录；

2.客户请求撤销上一交易日提交的申购或赎回申请；

3.客户查看所购买产品的资金收支情况

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 测试用例ID | 测试项 | 测试标题 | 重要级别 | 前置条件 | 输入数据 | 操作步骤 | 预期结果 |
| QR-001 | 查看交易记录 | 客户查询交易记录成功 | 中 | 有交易记录 | 时间范围：2024-01-01至2024-06-30 | 1. 进入交易查询 2. 设置时间范围 3. 执行查询 | 显示符合条件的交易记录 |
| QR-002 | 请求撤单 | 在可撤单时间内撤单成功 | 高 | 申购提交未清算 | 申购单号：SUB20240620001 | 1. 找到申购记录 2. 点击"撤单" | 订单状态变为"已撤销" |
| QR-003 | 请求撤单 | 超过撤单时限失败 | 中 | 数据已导出到数据中心 | 申购单号：SUB20240619001 | 1. 尝试撤单 | 提示"已超过撤单时间" |
| QR-004 | 查看基金收支 | 查询资金流水成功 | 中 | 有资金变动记录 | 时间范围：最近一个月 | 1. 进入资金收支查询 2. 设置查询条件 3. 执行查询 | 显示所有资金流水明细 |
| QR-005 | 查看基金收益 | 查看产品收益成功 | 中 | 持有该产品 | 产品ID：P1005 | 1. 选择产品 2. 查看收益 | 显示产品收益情况（含图表） |

### 3.1.6登录子系统测试用例

**功能需求：**

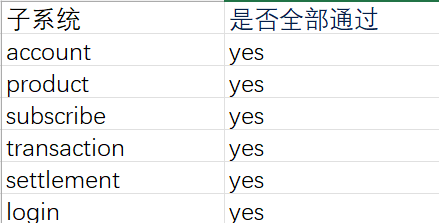
客户或管理员通过输入账号和密码进行身份验证，验证账号和密码的有效性通过后根据角色赋予相应的系统访问权限。

用户在电信系统验证身份后，可以设置新密码恢复账户访问。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| LOG-001 | 登录功能 | 管理员正确账号密码登录成功 | 高 | 系统运行正常 | 用户名：admin 密码：Admin@123 | 1. 打开登录页面 2. 输入用户名和密码 3. 点击"登录"按钮 | 1. 跳转到管理员控制台 2. 显示欢迎信息 |
| LOG-002 | 登录功能 | 客户正确账号密码登录成功 | 高 | 客户已注册 | 手机号：13800138000 密码：Pass123! | 1. 打开登录页面 2. 输入手机号和密码 3. 点击"登录"按钮 | 1. 跳转到客户首页 2. 显示账户概览 |
| LOG-003 | 登录功能 | 错误密码登录失败 | 中 | 用户账号存在 | 用户名：admin 密码：WrongPass | 1. 输入正确用户名 2. 输入错误密码 3. 点击"登录" | 提示"密码错误，请重新输入" |
| LOG-004 | 登录功能 | 不存在的账号登录失败 | 中 | 系统运行正常 | 手机号：13999999999 密码：AnyPass | 1. 输入未注册手机号 2. 输入任意密码 3. 点击"登录" | 提示"该账号未注册" |
| LOG-005 | 登录功能 | 连续3次错误密码账户锁定 | 高 | 账号存在 | 用户名：admin 密码：连续3次错误 | 1. 第一次输入错误密码 2. 第二次输入错误密码 3. 第三次输入错误密码 | 1. 第三次登录后提示"账号已锁定，请10分钟后重试" 2. 无法继续登录 |
| LOG-006 | 找回密码功能 | 通过手机验证码重置密码成功 | 高 | 账号已注册且绑定手机 | 手机号：13800138000 新密码：NewPass123! | 1. 点击"忘记密码" 2. 输入手机号 3. 获取验证码 4. 输入验证码 5. 设置新密码 | 提示"密码重置成功"，可用新密码登录 |
| LOG-007 | 找回密码功能 | 错误验证码重置密码失败 | 中 | 账号已注册 | 手机号：13800138000 验证码：错误验证码 | 1. 输入正确手机号 2. 输入错误验证码 3. 尝试重置密码 | 提示"验证码错误" |

## 3.2测试

使用Selenium包来对我们的系统进⾏测试，并分别在几个子系统内对各个测试用例进行手动。



经过测试，六个子系统的全部测试用例均通过。

# 四、非功能测试

## 4.1测试目的

压力测试在性能测试中是一个重要的环节，其主要目的是模拟系统在正常或峰值负载下的表现，以评估系统的稳定性、可靠性和性能指标。通过在系统承受高负载的情况下进行压力测试，可以识别系统的性能瓶颈，即系统在特定负载下的性能瓶颈点。这有助于开发团队针对性地优化和改进系统，提升性能。

## 4.2 测试工具

JMeter 是一款开源的性能测试工具，用于模拟多种协议的负载，并收集性能指标和测试结果。它能够模拟大量用户并发送请求到目标服务器，以评估系统的性能表现。本次测试将使用 JMeter 进行压力负载测试。

## 4.3测试需求分析

一般来说，需要进行压力测试的通常满足以下条件：核心业务功能、用户量以及与外部接口的对接。查看产品是几乎每个用户进入系统的第一件事，我们的一系列业务也都是围绕产品产生的，所有产品列表的展示是我们系统的核心功能之一，所有我们计划对产品查看进行压力测试。同时获取交易账户这一功能的数据量较大，需要显示所有用户的交易账户，数据量非常大的功能也是压力测试的核心目标之一，我们也将其作为压力测试的指标之一。同时，获取交易时间这一功能是许多功能所需要用到的，使用次数非常多，我们也需要知道其在高压环境下的性能要求。

经过服务器等因素的综合考虑，我们认为这些业务需要达到以下要求：

1.90%的请求响应时间应该小于200ms

2.50%的请求响应时间应该小于100ms

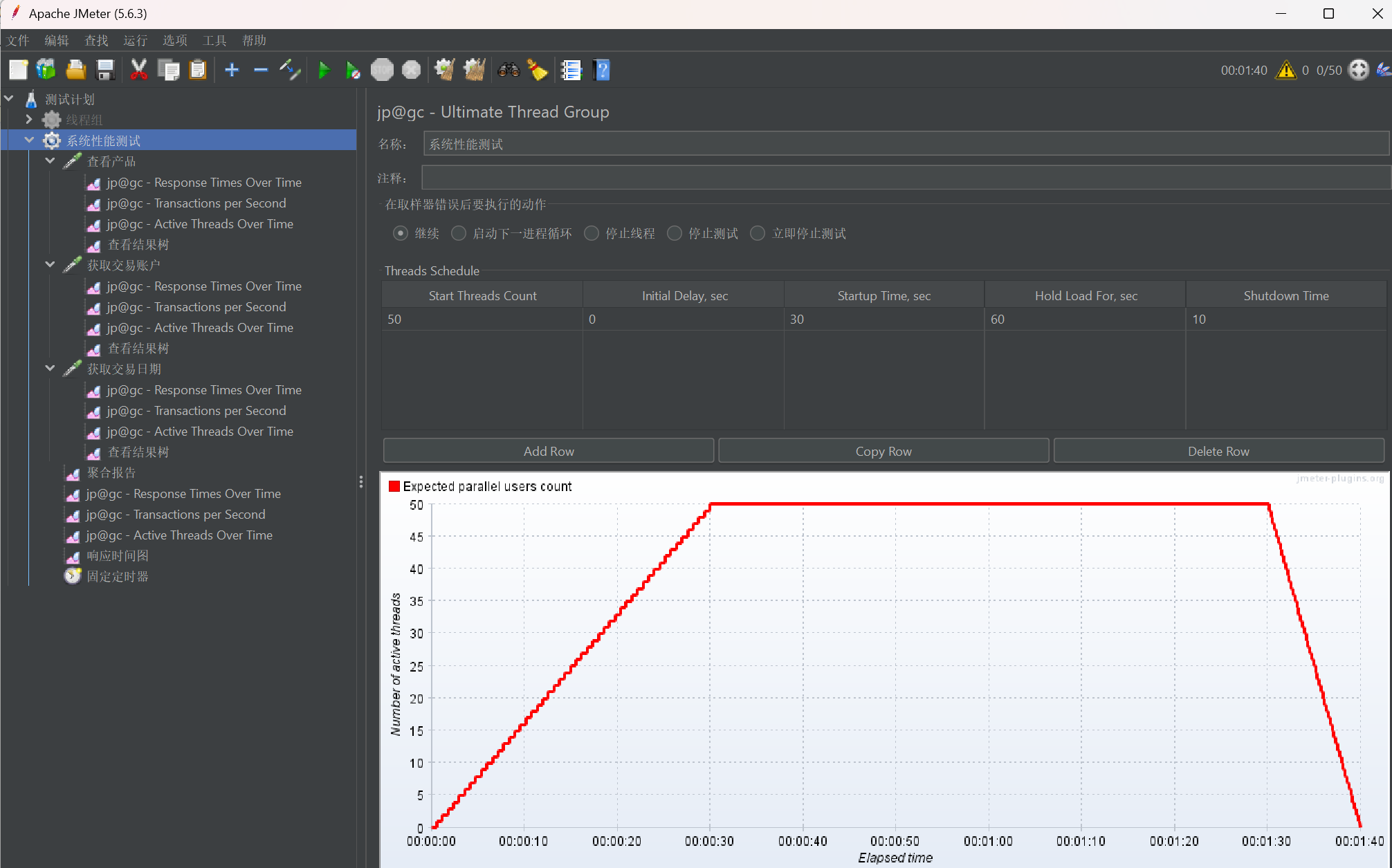
3.最少支持50个用户同时查询

4.事务成功率达到100%

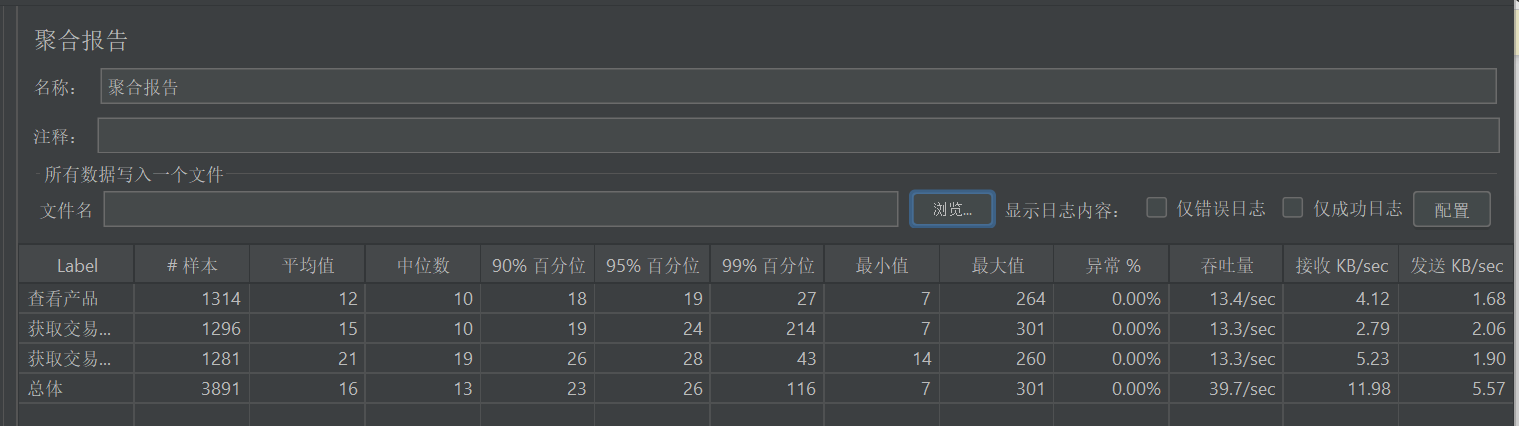
## 4.4测试方案设计

### 4.4.1 50线程压力测试

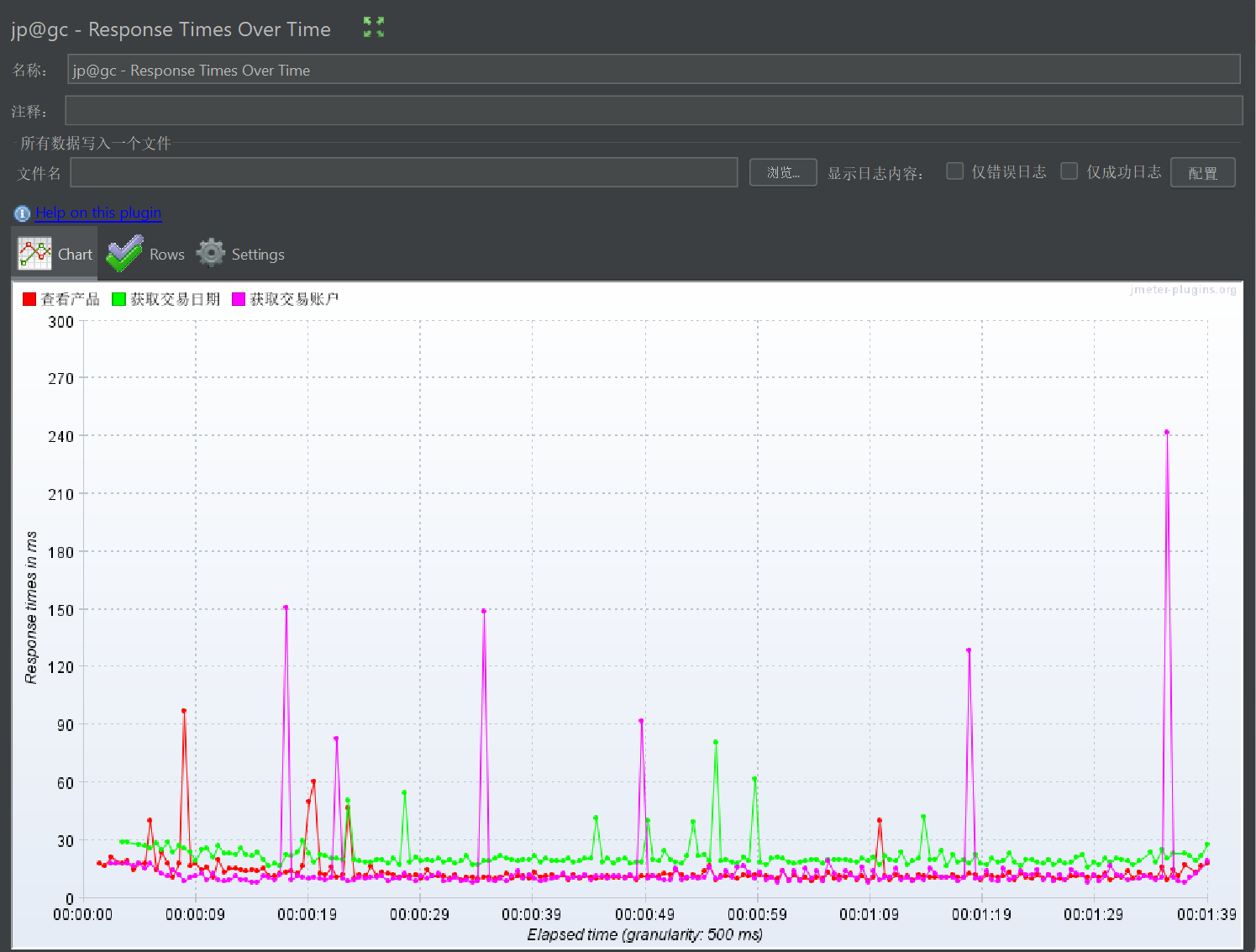
我们使用Jmeter的Ultimate Thread Group插件，对三个用例进行压力测试，并设置监听器。如图所示：

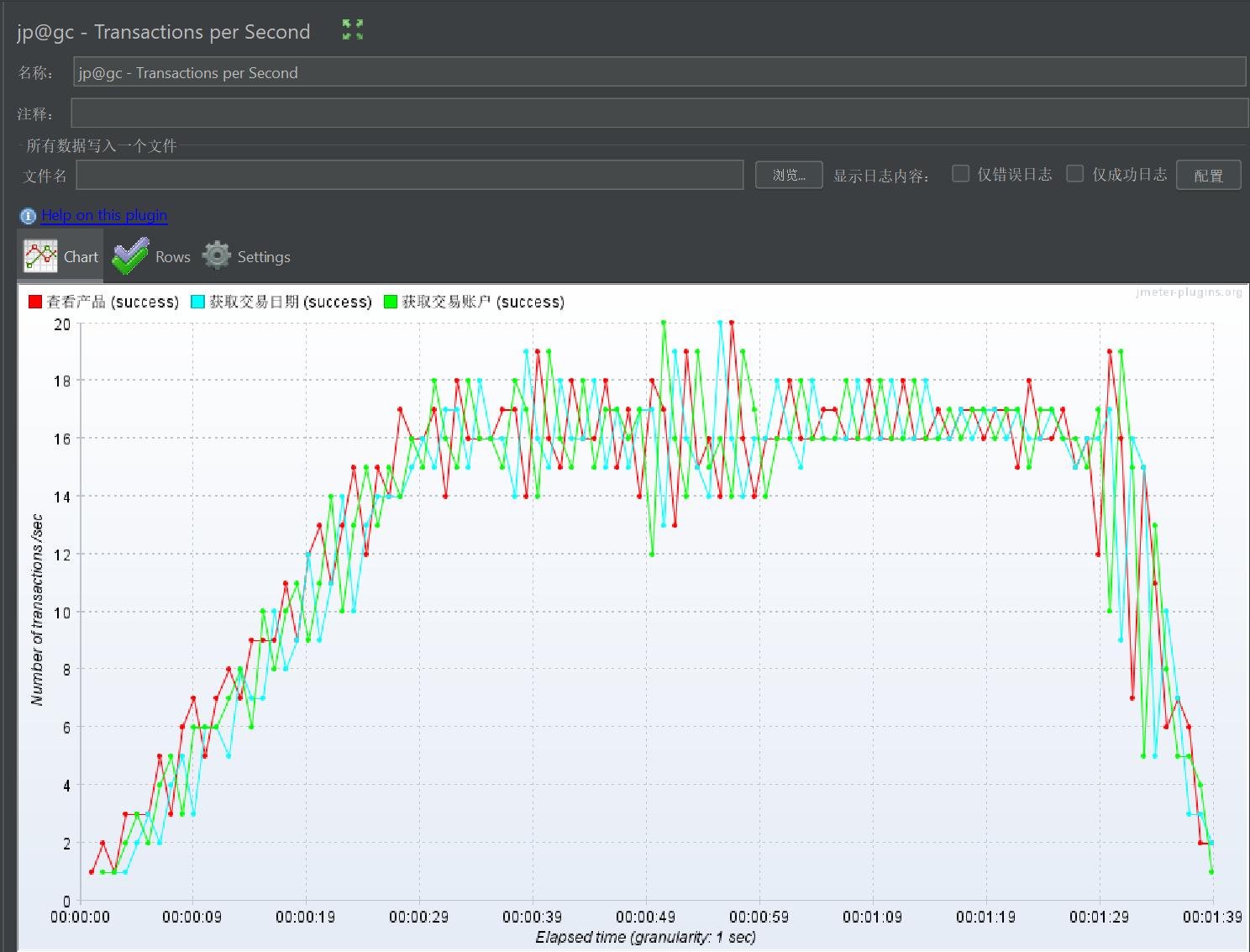


设置线程数为50（即模拟50个用户同时使用系统），从0秒开始累加，经过30s到达最高值50个并持续一分钟，最后在10s内关闭所有线程。



通过50个线程的压力测试得到聚合报告，我们发现样本的反应时间99百分位普遍在200ms以下，中位数甚至在20ms以下，异常率为0。已经大大超出了我们的预期性能。

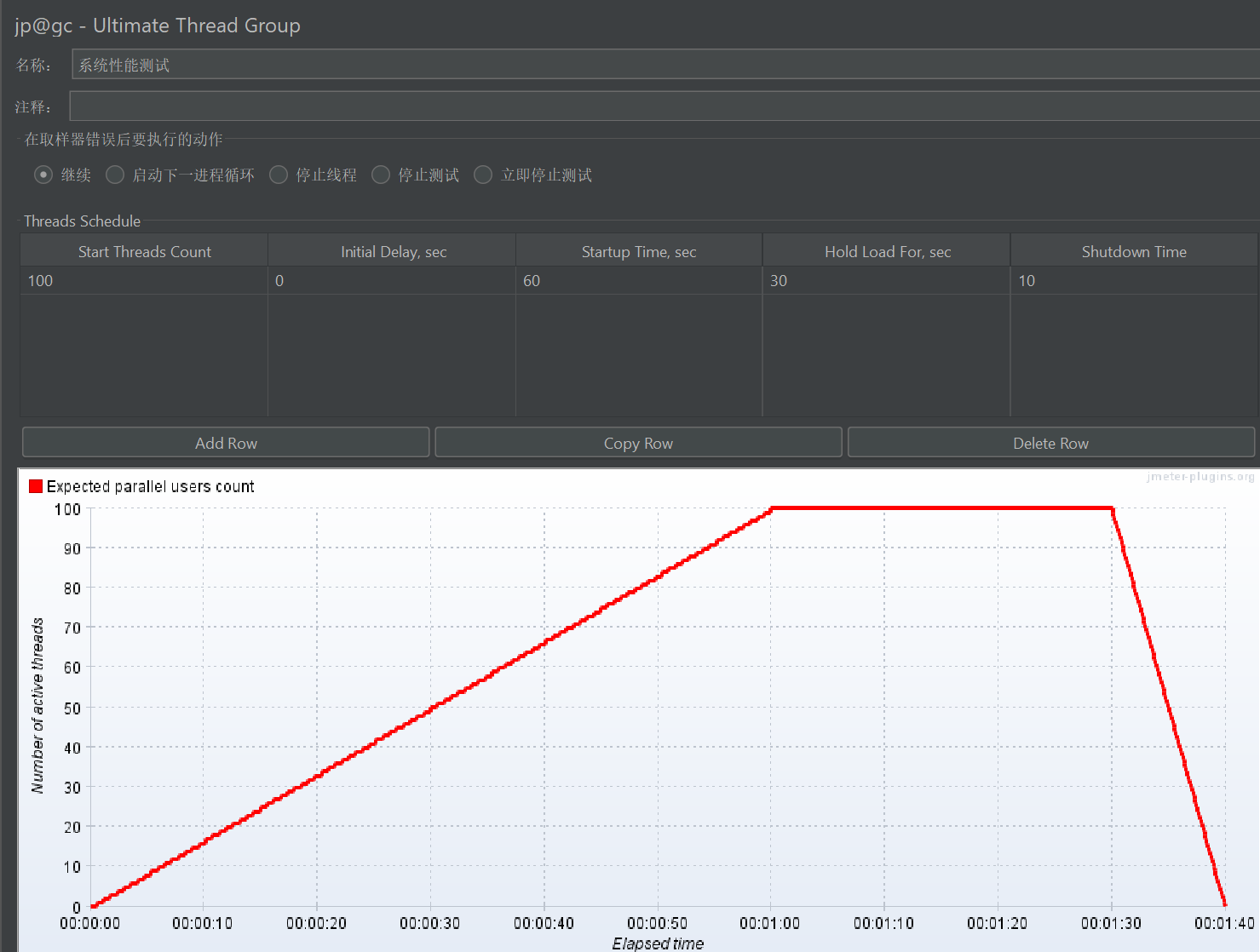
Response Times Over Time 主要用于监听整个事务运行期间的响应时间。显示除了个别查询以外其他查询时间还是趋于稳定，并且查看产品和获取交易账户查询时间相近并普遍低于获取交易日期。



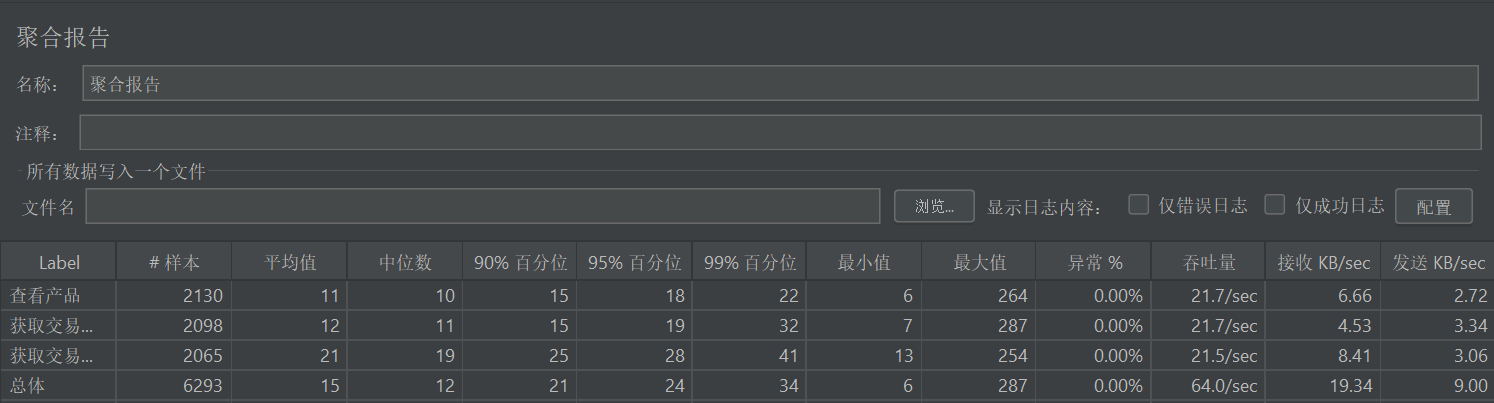
Transactions per Second（TPS）监听器是一个用于分析系统吞吐量的重要工具。TPS，即每秒处理的事务数，表示一个客户机向服务器发送请求后，服务器做出反应的过程。这个指标反映了系统在同一时间内处理业务的最大能力。TPS值越高，说明系统的处理能力越强。由图可见在到达50线程时几个用例TPS基本稳定与16-18之间

### 4.4.2 100线程压力测试

由于我们系统在50线程时发挥较好，我们就对性能**提高了要求**，测试系统在100个线程同时工作时的性能。

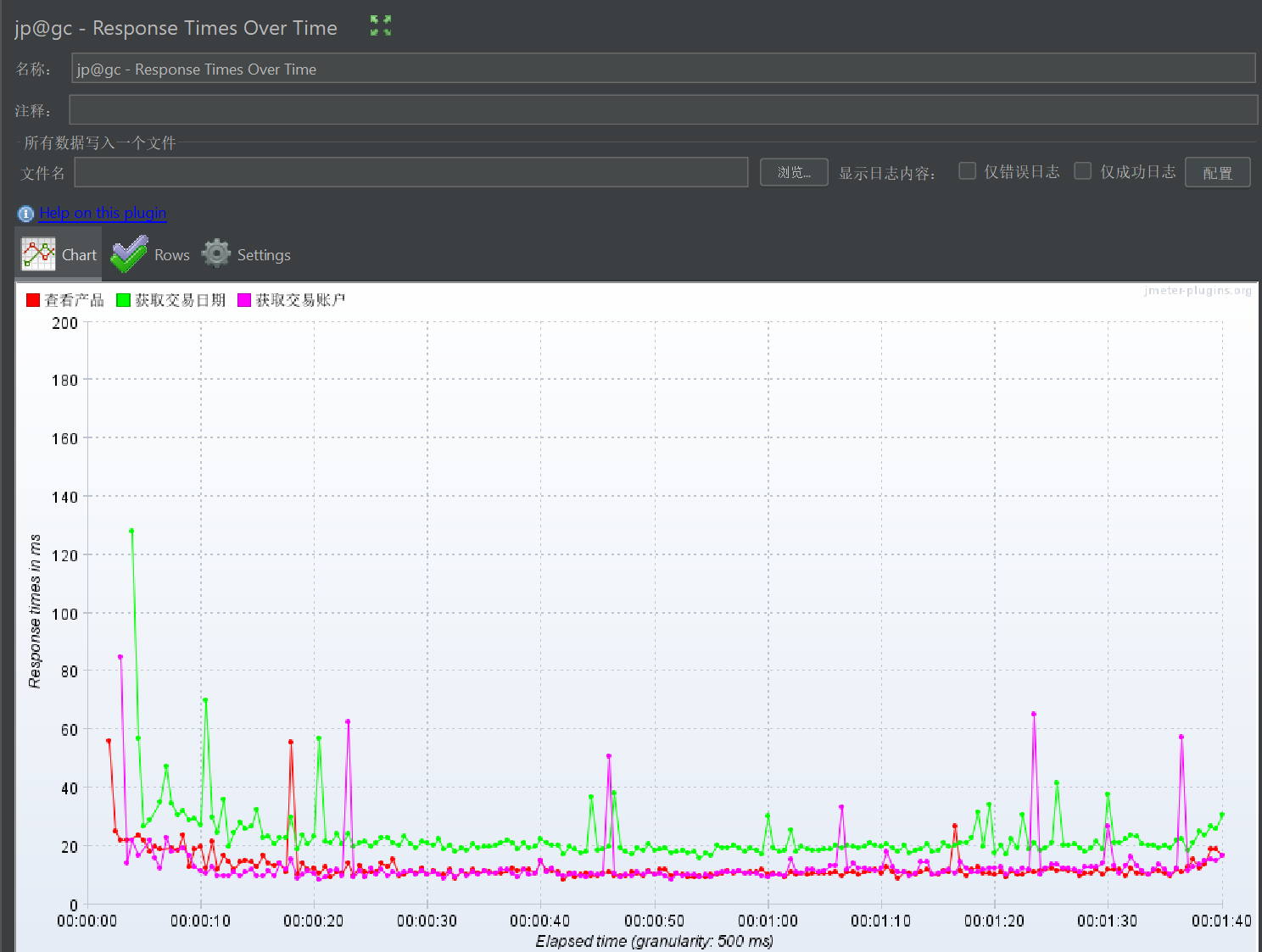


设置线程数为100（即模拟100个用户同时使用系统），从0秒开始累加，经过60s到达最高值100个并持续班分钟，最后在10s内关闭所有线程。

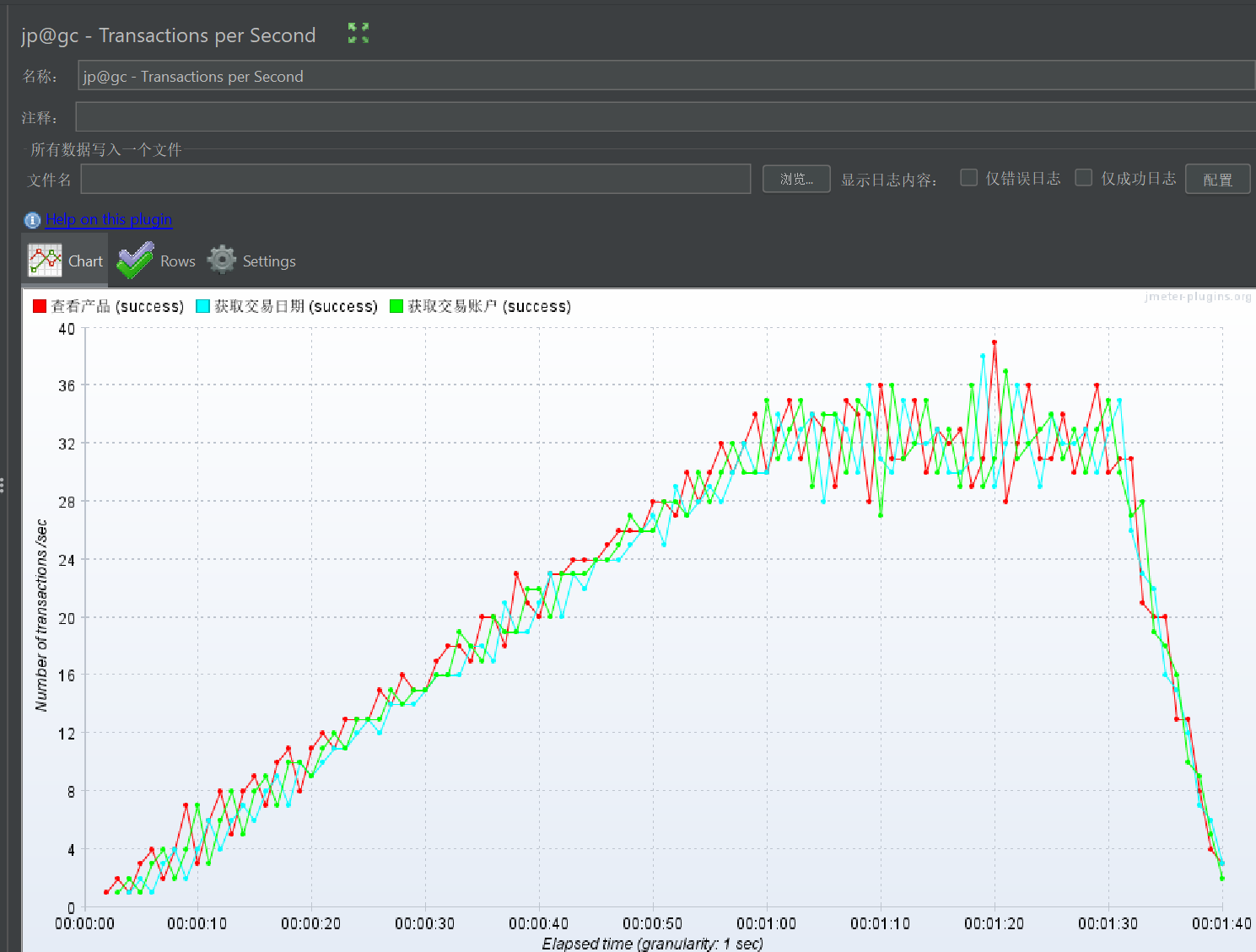


通过100个线程的压力测试得到聚合报告，我们发现样本的最大反应时间普遍在300ms以下，中位数仍在20ms以下，异常率为0。系统在100个用户使用时仍有较好性能。

这是Response Times Over Time：



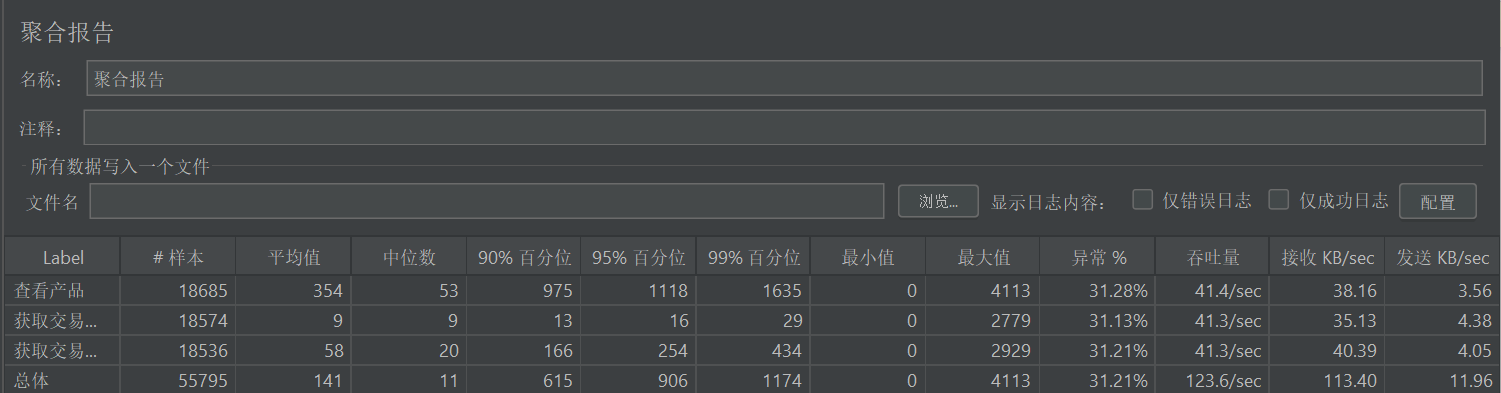
Transactions per Second在100线程时升高了，基本稳定在32-36之间：



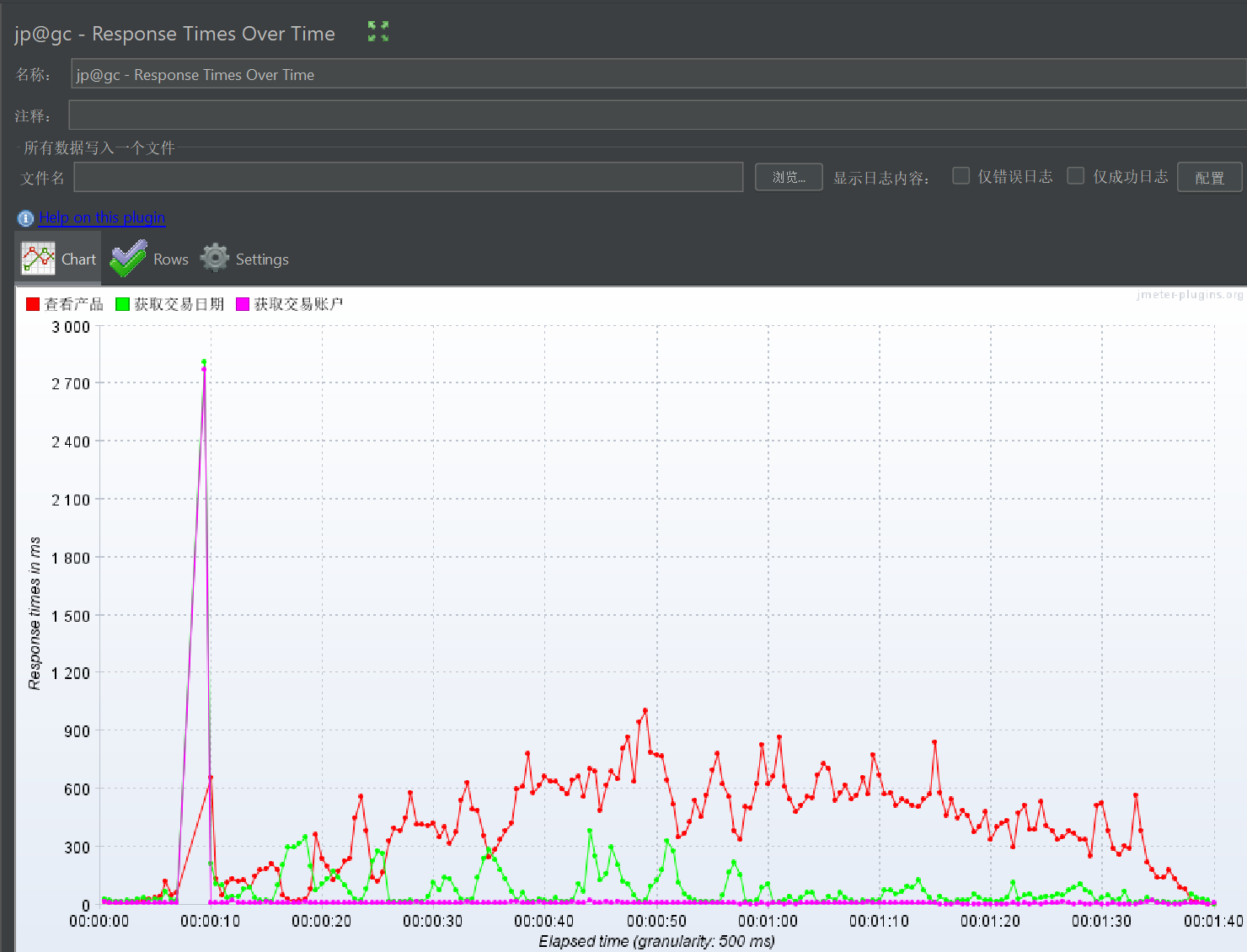
为追寻我们系统的**极限性能**，我们继续增大压力：

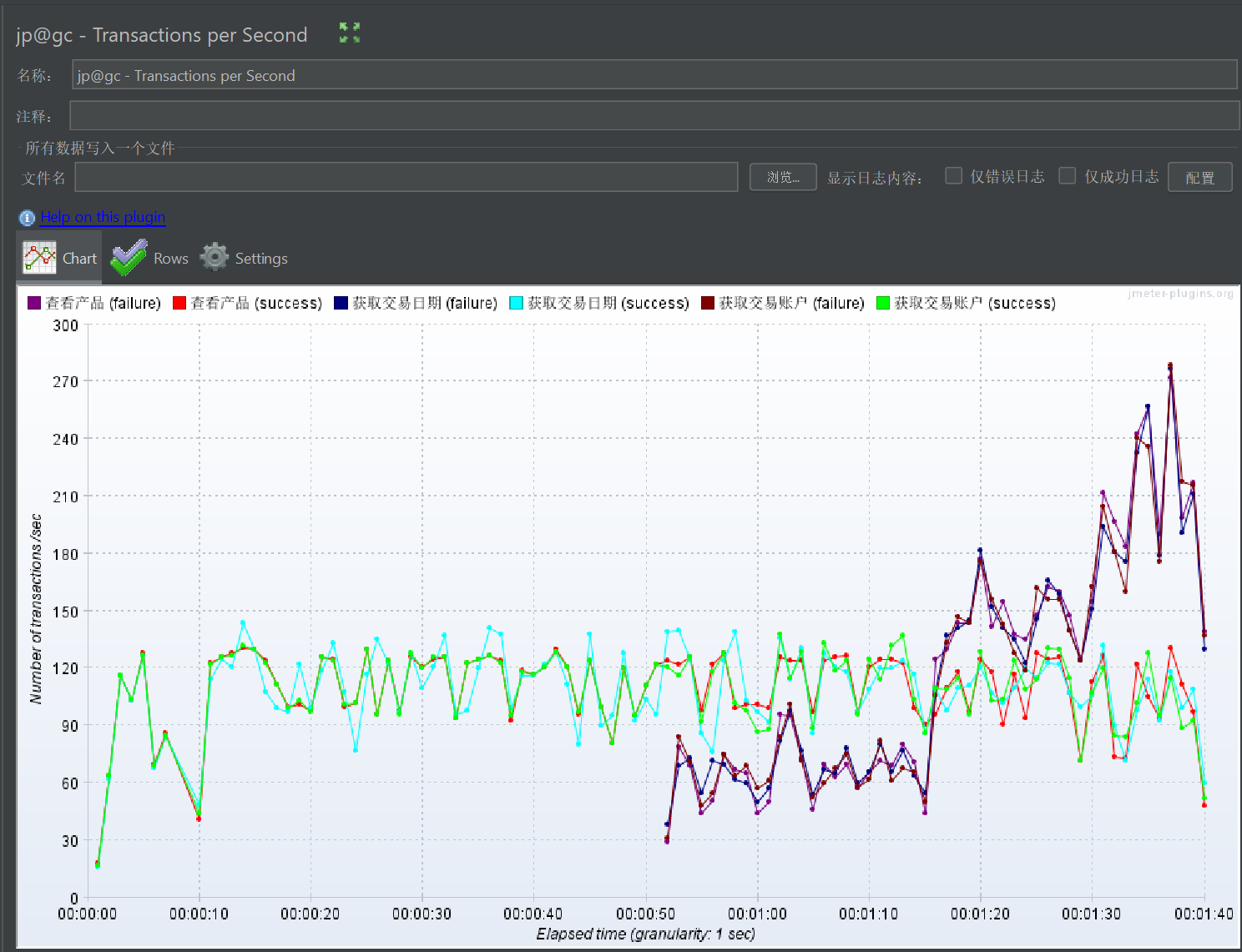
### 4.4.3 120线程压力测试：

当把线程增加到120个，我们得到聚合报告：

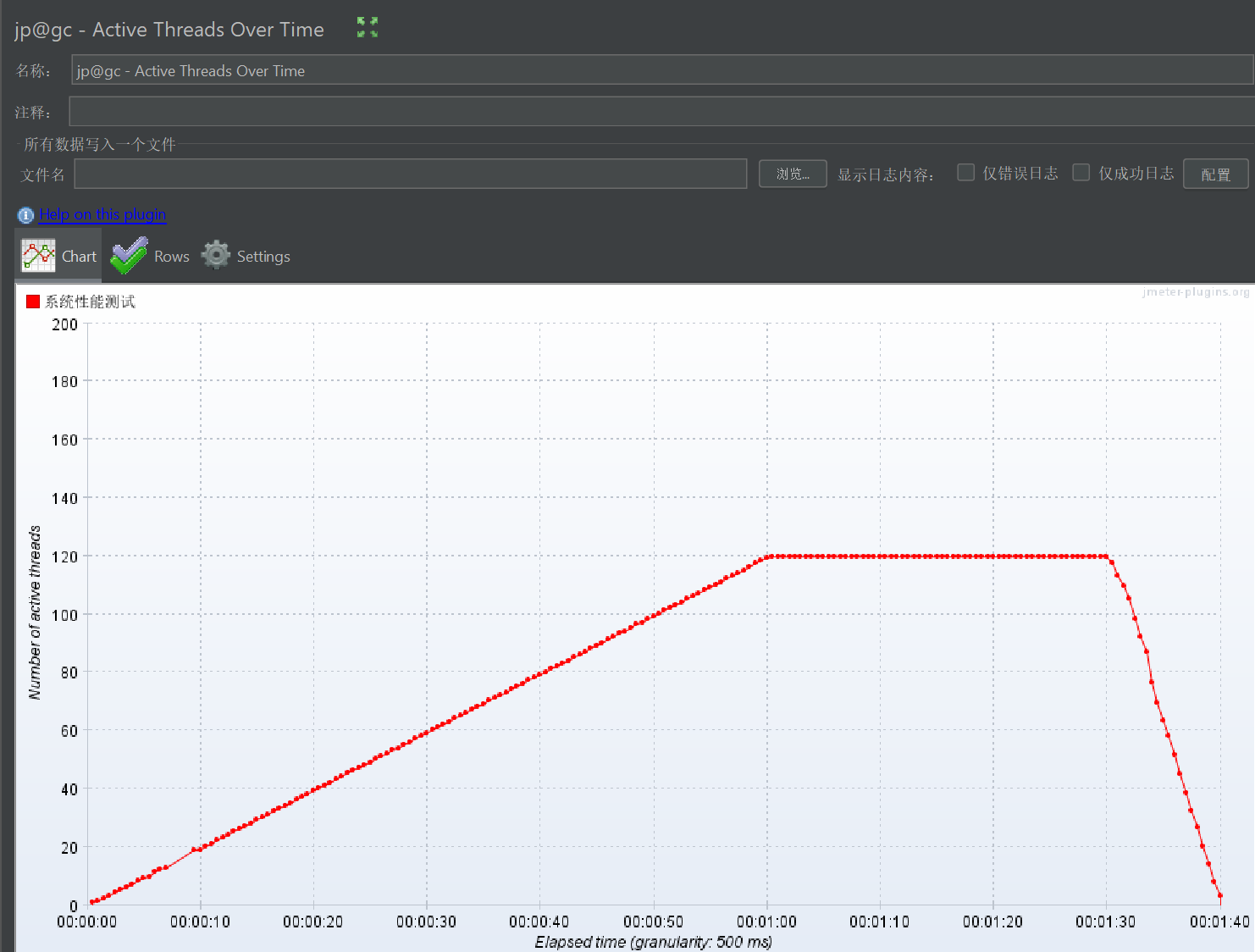


发现异常率已经到了约31%，而且最大响应时间已经来到4000多ms，说明系统在这样的高压环境中出现了故障。





根据TPS我们发现系统在50s时开始出现failure，而且几乎是不同用例同时出现，经过分析我们发现这些failure是因为服务器中数据连接过多导致的。在对比下图我们发现在50s左右线程数正好达到100多个，所以我们认为同济基金交易系统的极限人数应该在100人以下。



## 4.5测试结论

服务器瓶颈：100个线程

1. CPU配置过低：根据测试结果，CPU的性能无法满足当前的负载需求。CPU使用率过高可能导致系统响应变慢，影响性能。建议考虑升级服务器的CPU配置，以提升系统的计算能力和响应速度。
2. CPU使用率过高，mysqld服务占比太高：在测试过程中，mysqld服务占用了过高的CPU使用率。这可能是由于数据库查询和处理操作较为复杂，导致mysqld进程负载过重。建议对数据库进行优化，例如优化查询语句、添加索引等措施，以减轻数据库的负载压力，并提升性能。

针对以上问题，可以采取以下措施：

1. 升级服务器的CPU配置：通过增加或更换更高性能的CPU，提升服务器的计算能力和处理能力，以满足系统在高负载情况下的性能需求。
2. 数据库性能优化：针对mysqld服务占用过高的CPU使用率，进行数据库性能优化。可以优化数据库查询语句、调整数据库索引，以提高数据库查询和处理的效率，减少CPU负载。
3. 系统资源监控：实时监控服务器的系统资源使用情况，包括CPU、内存、磁盘等。这有助于及时发现并解决性能瓶颈，确保系统的稳定性和可靠性。
4. 反复测试和验证：根据优化措施，进行反复的性能测试和验证。通过持续的测试和优化迭代，逐步改善系统的性能表现，确保系统能够在预期的负载条件下正常运行。