卞炜奕 崔宇璇 谢斌 杨潇睿

**华东师范大学 计算机科学与软件工程学院**

摘要

性能测试报告

虚拟宠物医院学习系统

目录

[1 测试结果评价与建议 2](#_Toc512245820)

[1.1 测试结果 2](#_Toc512245821)

[1.2 系统性能评价 2](#_Toc512245822)

[1.2.1 API响应时间 2](#_Toc512245823)

[1.2.2 业务处理能力 3](#_Toc512245824)

[1.2.3 稳定性/健壮性 3](#_Toc512245825)

[1.3 系统性能建议 4](#_Toc512245826)

[2 性能问题解决分析 4](#_Toc512245827)

[2.1 待解决问题 4](#_Toc512245828)

[2.1.1 严重性能问题 4](#_Toc512245829)

[2.1.2 响应时间问题 4](#_Toc512245830)

[2.2 已解决问题 4](#_Toc512245831)

[2.2.1 严重性能问题 4](#_Toc512245832)

[2.2.2 响应时间问题 4](#_Toc512245833)

[3 独立/混合场景结果 4](#_Toc512245834)

[3.1 测试结果描述 4](#_Toc512245835)

[3.2 详细测试结果 5](#_Toc512245836)

[4 测试环境分析 6](#_Toc512245837)

[4.1 系统架构设计 6](#_Toc512245838)

[4.2 测试环境配置 6](#_Toc512245839)

[4.3 环境差异分析 7](#_Toc512245840)

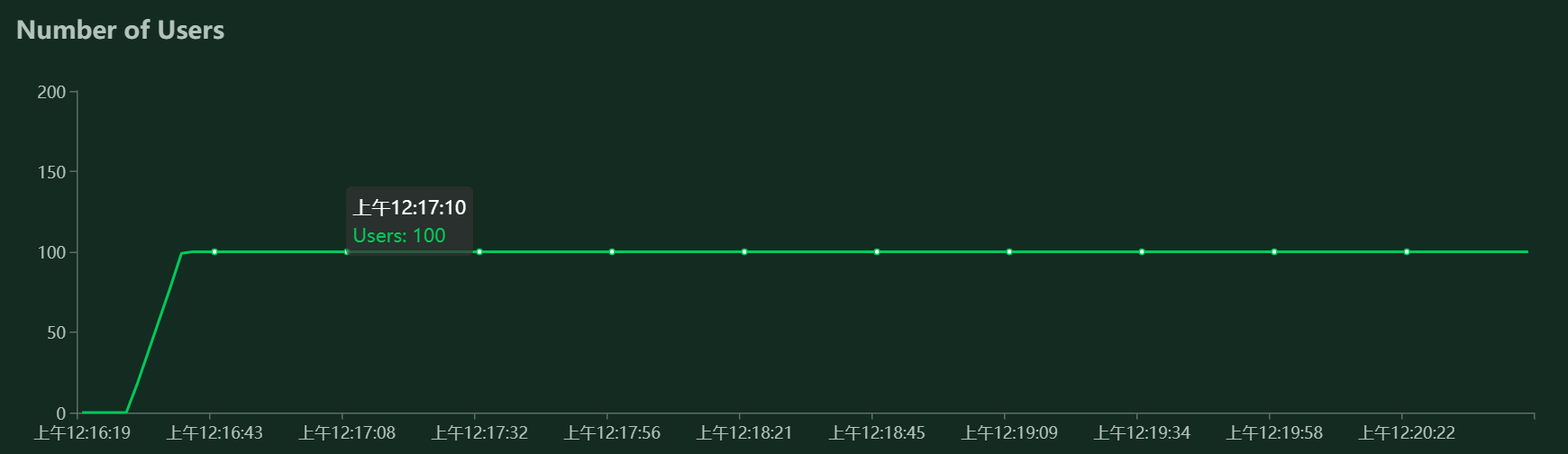
# 测试结果评价与建议

## 测试结果

|  |  |
| --- | --- |
| 测试开始日期 | 2018年4月19日 |
| 测试结束日期 | 2018年4月20日 |
| 提交缺陷数 | 5 |
| 延期/未解决缺陷数 | 3 |
| 拒绝缺陷数 | 2 |
| 总测试场景 | 12个独立场景+16个混合场景 |
| 执行场景数 | 12个独立场景+16个混合场景 |
| 场景执行率 | 100% |
| 缺陷率（%） | 2个独立场景失败，2个混合场景失败 |
| 投入人数 | 100 |
| 测试生产率 | 85.7% |
| 系统上线建议 | 部分页面与API响应时间过长，需 |

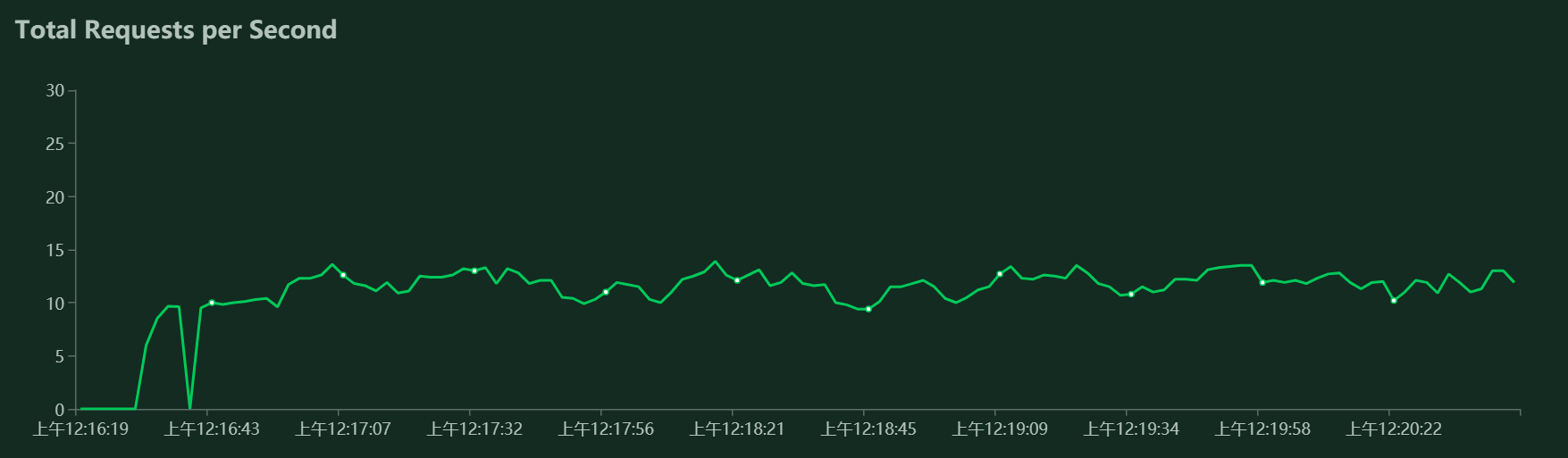
## 系统性能评价

测试用户数量：

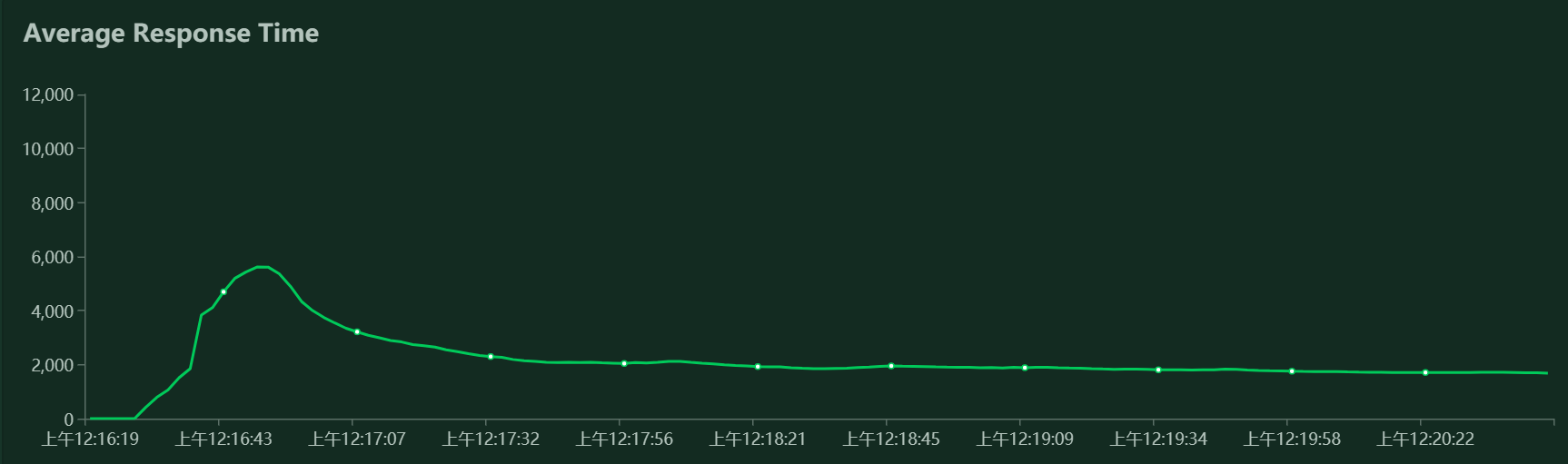


### API响应时间

API请求次数如下：



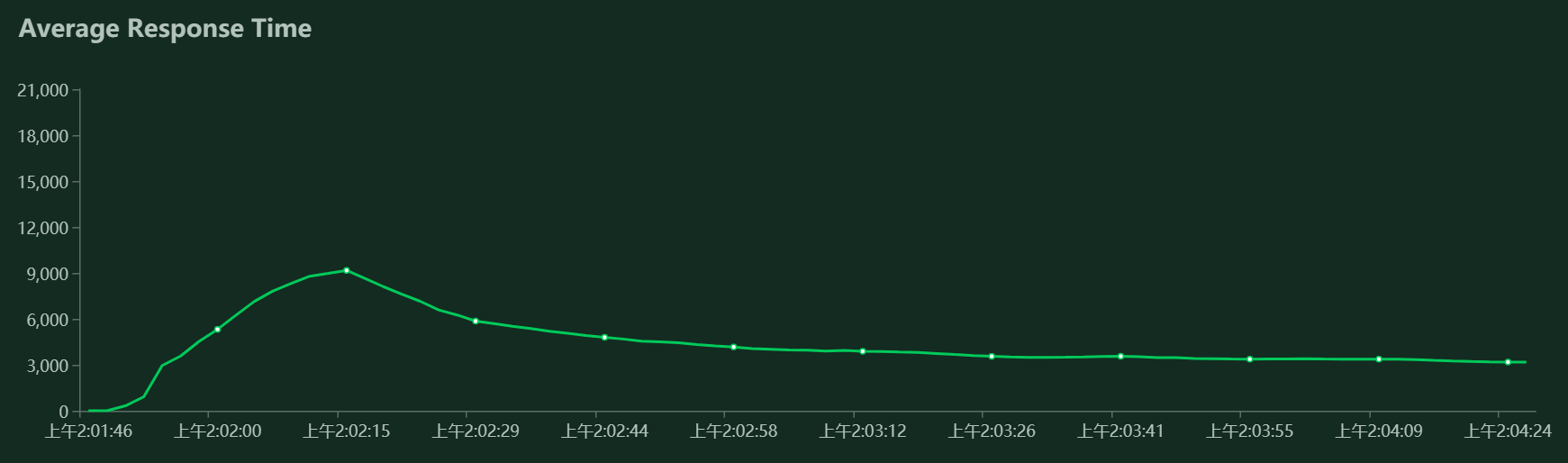
响应时间如下：



由图可知，系统每秒发出的request的数量平均在10到15个之间，系统的平均响应时间基本稳定于0.2秒以下。

### 业务处理能力

业务处理时还需要进行页面加载时间的测试，用户数量不变，其并发访问的响应时间如下：



由图可知，随着测试进行，页面访问时间基本稳定在0.3秒左右。

结合之前API测试的结果，可以得出进行业务处理时，系统所花费的时间基本稳定在1秒以下，可以保证用户使用顺畅。

### 稳定性/健壮性

由业务处理部分测试结果，所有页面平均访问时间基本稳定在0.3秒左右，未发现其出现大规模波动。可以得出系统的稳定性良好，保证用户体验良好。

## 系统性能建议

虽然系统页面平均访问时间较为令人满意，但是经过测试发现以下页面的访问时间波动较大：3D导览页面，测试部分页面首页，错题本部分。可能造成波动的原因：

1、网络原因，测试环境较为不稳定。

2、页面加载资源过多。

建议对这些界面进行检查，尝试改进，以此稳定其访问时间。

# 性能问题解决分析

## 待解决问题

### 严重性能问题

暂无

### 响应时间问题

暂无

## 已解决问题

### 严重性能问题

暂无

### 响应时间问题

暂无

# 独立/混合场景结果

## 测试结果描述

根据性能测试报告中所制定的独立与混合测试场景，我们采取逐渐加压的方式。通过设置相应权重，来控制其访问的并发数量。对于高等级的测试场景需要进行更多次的测试。结果测试，所有独立测试场景的运行结果均达到要求。当用户数从0逐渐增加到100时，所有的响应时间均小于1秒，平均响应时间为0.5秒。

对于混合场景，平均响应时间均达到测试计划中的要求。当用户数量从0逐渐增加到100时，除去刚开始测试时受网络环境波动影响，其余的响应时间均小于2秒，达到要求，测试通过。

## 详细测试结果

注册部分独立测试结果：



登录部分独立测试结果：



测试部分独立测试结果：

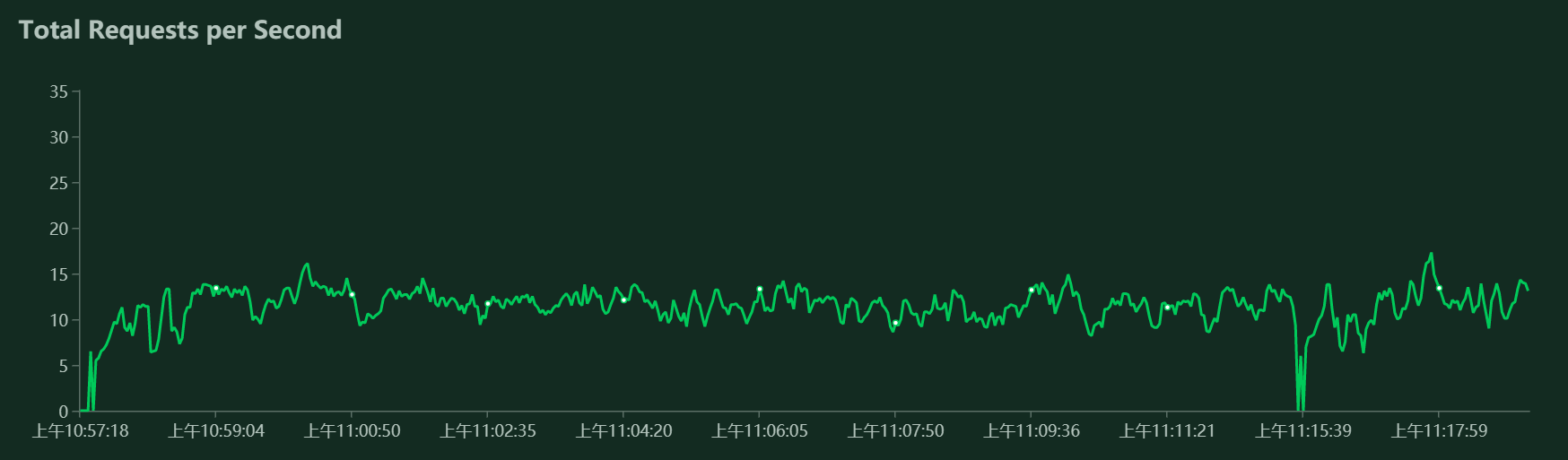


3D导览部分独立测试结果：

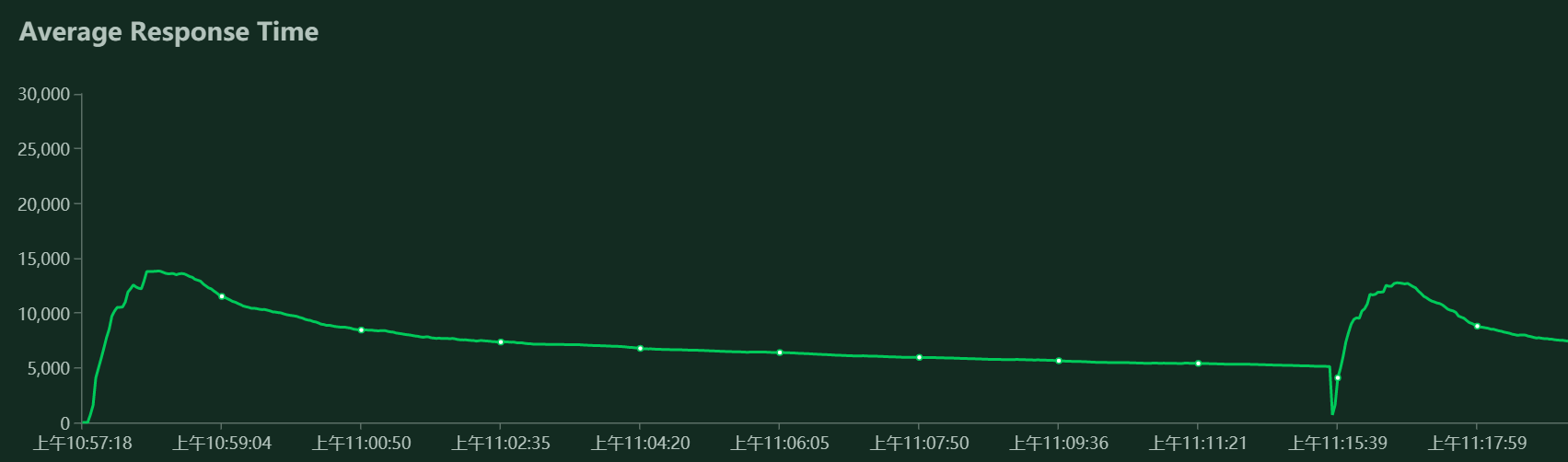


混合场景测试结果：

请求数量：



响应时间：



# 测试环境分析

## 系统架构设计

本次测试采用Locust测试框架，这是一个开源的性能测试工具。在Locust测试框架中，测试场景是采用纯Python脚本进行描述的。对于最常见的HTTP(S)协议的系统，Locust采用Python的requests库作为客户端，使得脚本编写大大简化，富有表现力的同时且极具美感。而对于其它协议类型的系统，Locust也提供了接口，只要我们能采用Python编写对应的请求客户端，就能方便地采用Locust实现压力测试。从这个角度来说，Locust可以用于压测任意类型的系统。

## 测试环境配置

在确保正确安装python3后，locust下载指令为：

<pip install locustio

之后便可以使用python编写测试脚本，运行时需给出总用户数量以及每秒并发数量。

## 环境差异分析

此次测试的环境为：windows 10系统，网速为20兆。

## 测试脚本

from locust import HttpLocust, TaskSet, task  
  
  
class UserBehavior(TaskSet):  
 token = None  
 def on\_start(self):self.login()  
  
  
 def login(self):  
 global token  
 response = self.client.post("/user/login", {"username": "admin", "password": "qwertyuiop"})  
 token = response.json()['token']  
  
 @task(1)  
 def profile(self):  
 self.client.get("/user/profile", headers={'Authorization': 'Token {}'.format(token)})  
  
 @task(1)  
 def doc(self):  
 self.client.get("/docs/")  
  
 @task(2)  
 def leaderboard(self):  
 self.client.get("/test/leaderboard")  
  
 @task(1)  
 def disease\_group(self):  
 self.client.get('/disease/group')  
  
 @task(2)  
 def disease\_image(self):  
 self.client.get('/disease/images')  
  
 @task(2)  
 def disease\_video(self):  
 self.client.get('/disease/video')  
  
 @task(5)  
 def exam(self):  
 self.client.get("/test/exam/1")  
  
 @task(5)  
 def exam(self):  
 self.client.get("/test/exam/2")  
  
 @task(5)  
 def exam\_list(self):  
 self.client.get("/test/exam-list")  
  
 @task(5)  
 def exercise(self):  
 self.client.get("/test/exercise/low")  
  
 @task(4)  
 def exercise(self):  
 self.client.get("/test/exercise/high")  
  
  
 @task(5)  
 def error(self):  
 self.client.get('/test/error', headers={'Authorization': 'Token {}'.format(token)})  
  
 @task(5)  
 def history(self):  
 self.client.get('/test/history', headers={'Authorization': 'Token {}'.format(token)})  
  
  
class WebsiteUser(HttpLocust):  
 host = 'http://localhost:4200'  
  
 task\_set = UserBehavior  
 min\_wait = 5000  
 max\_wait = 9000