BÁO CÁO KIỂM THỬ HIỆU SUẤT DỰ ÁN "BOOKSTORE"

**Ngày thực hiện: 24/03/2025**

**Người thực hiện: Nguyễn Hữu Hoàng Kiệt**

1. MỤC TIÊU

Báo cáo này trình bày kết quả kiểm thử hiệu suất của hệ thống "Bookstore" (Quản lý nhà sách) thông qua hai phương pháp: **Load Testing (Kiểm thử tải)** và **Stress Testing (Kiểm thử căng thẳng)**. Mục tiêu chính bao gồm:

1. **Load Testing**: Đánh giá khả năng xử lý của hệ thống dưới tải người dùng bình thường và tải cao trong giới hạn thiết kế (ví dụ: 100-1000 người dùng).
2. **Stress Testing**: Xác định điểm phá vỡ (breaking point) của hệ thống khi vượt quá giới hạn, từ đó đánh giá độ ổn định và khả năng phục hồi (ví dụ: 2000-5000 người dùng).

**Khác với Load Testing**: Load testing kiểm tra trong giới hạn thiết kế, còn stress testing cố ý vượt quá giới hạn để tìm điểm yếu.

Các chức năng được kiểm tra bao gồm đăng nhập, tìm kiếm sách, thêm vào giỏ hàng, thanh toán, và xem báo cáo thống kê.

2. PHƯƠNG PHÁP KIỂM THỬ

**2.1. Công cụ sử dụng: Locust**

**Locust** là một công cụ kiểm thử tải mã nguồn mở, sử dụng Python để mô phỏng hàng ngàn người dùng đồng thời gửi yêu cầu đến hệ thống của bạn nhằm đánh giá hiệu suất.

**Locust test** có thể sử dụng bởi cả công cụ dòng lệnh hoặc giao diện Web thân thiện, dễ tiếp cận. Các thông số như thông lượng (*throughput*), thời gian phản hồi (*response times*) và các lỗi (*errors*) có thể được xem trực tiếp theo thời gian thực, hoặc/và được export ra để phân tích sau.

**2.2. Môi trường kiểm thử**

* **RPS (Requests per Second)**: Số yêu cầu mỗi giây hệ thống xử lý được.
* **Response Time**: Thời gian phản hồi trung bình (ms).
* **Failure Rate**: Tỷ lệ lỗi (% yêu cầu thất bại, như timeout hoặc mã lỗi 500).

**2.3. Cài đặt locust**

* Nếu bạn chưa cài Locust, thực hiện như sau: pip install locust
* Kiểm tra phiên bản: locust --version

3. KIỂM THỬ TẢI (LOAD TESTING)

**3.1. Kịch bản kiểm thử**

Kịch bản mô phỏng hành vi người dùng bình thường trong hệ thống "Bookstore" với file locustfile.py:

**3.2. Quy trình thực hiện**

* **Số người dùng**: 100, 500, 1000 (tăng dần).
* **Spawn rate**: 10 người/giây.
* **Thời gian chạy**: 5 phút mỗi kịch bản.
* **Endpoint**: http://localhost:5001.

**3.3. Kết quả Load Testing**

| **Số người dùng** | **RPS** | **Response Time (ms)** | **Failure Rate (%)** | **Ghi chú** |
| --- | --- | --- | --- | --- |
| 100 | 50 | 150 | 0 | Hệ thống hoạt động ổn định |
| 500 | 220 | 300 | 2 | Chậm nhẹ, lỗi nhỏ |
| 1000 | 350 | 800 | 10 | Hiệu suất giảm, lỗi tăng |

* **Nhận xét**:
  + Hệ thống xử lý tốt với 100-500 người dùng, thời gian phản hồi dưới 300ms, không đáng kể lỗi.
  + Với 1000 người dùng, thời gian phản hồi tăng lên 800ms, tỷ lệ lỗi 10%, cho thấy giới hạn thiết kế gần đạt ngưỡng.

4. KIỂM THỬ CĂNG THẲNG (STRESS TESTING)

**4.1. Kịch bản kiểm thử**

Kịch bản đẩy hệ thống vượt giới hạn với file locustfile.py:

**4.2. Quy trình thực hiện**

* **Số người dùng**: 1000, 2000, 5000 (tăng dần để tìm điểm phá vỡ).
* **Spawn rate**: 100 người/giây.
* **Thời gian chạy**: 5 phút mỗi kịch bản hoặc đến khi hệ thống sập.

**4.3. Kết quả Stress Testing**

| **Số người dùng** | **RPS** | **Response Time (ms)** | **Failure Rate (%)** | **Ghi chú** |
| --- | --- | --- | --- | --- |
| 1000 | 350 | 800 | 10 | Hiệu suất giảm |
| 2000 | 600 | 2000 | 30 | Chậm nghiêm trọng, lỗi tăng |
| 5000 | 700 | 15000 | 80 | Hệ thống gần sập, lỗi lớn |

* **Điểm phá vỡ**: Khoảng 4000-5000 người dùng, khi RPS đạt đỉnh 700, response time vượt 15 giây, và 80% yêu cầu thất bại (timeout hoặc lỗi 500).
* **Nguyên nhân lỗi**:
  + Cơ sở dữ liệu quá tải (query chậm).
  + Server hết tài nguyên (CPU 100%, RAM đầy).

5. PHÂN TÍCH VÀ ĐỀ XUẤT CẢI THIỆN

* **Load Testing**: Hệ thống hoạt động ổn định dưới 500 người dùng, phù hợp với tải thông thường (ví dụ: ngày thường). Tuy nhiên, với 1000 người dùng (tải cao như ngày khuyến mãi), hiệu suất bắt đầu giảm.
* **Stress Testing**: Hệ thống chịu được tối đa khoảng 2000 người dùng trước khi xuất hiện lỗi nghiêm trọng. Điểm phá vỡ tại 5000 người dùng cho thấy giới hạn phần cứng và tối ưu hóa hiện tại.

6. KẾT LUẬN

Hệ thống "Bookstore" đáp ứng tốt tải bình thường (dưới 500 người dùng) với thời gian phản hồi dưới 300ms và không lỗi. Tuy nhiên, khi vượt quá 1000 người dùng, hiệu suất giảm rõ rệt, và hệ thống sập hoàn toàn ở mức 5000 người dùng. Các cải tiến về cơ sở dữ liệu, tài nguyên server, và kiến trúc phân tán là cần thiết để nâng cao khả năng chịu tải và đảm bảo hoạt động ổn định trong các tình huống tải cao (như đợt khuyến mãi lớn).