---

Performance Test Report for Digikala

Test Duration: 5 minutes

Tested URL: https://www.digikala.com

Tool Used: Locust

---

1. Test Overview

- Test Type: Load Testing
- Number of Virtual Users (Peak Concurrency): 100 users
- Ramp-Up (Users Started per Second): 1 user/sec
- Total Number of Requests: 2609
- Requests per Second (RPS): 33

---

2. Test Results Summary

2.1. Homepage (GET /)

- Total Requests: 1304
- Failed Requests: 881
- Median Response Time: 150 ms
- 95th Percentile Response Time: 530 ms
- 99th Percentile Response Time: 1700 ms
- Average Response Time: 225.55 ms
- Minimum Response Time: 117 ms
- Maximum Response Time: 3320 ms
- Average Size of Response: 11569.98 bytes
- Current RPS: 16 requests/sec
- Current Failures/sec: 10.3 failures/sec

2.2. Product Page (GET /product/dkp-4278853)

- Total Requests: 1305
- Failed Requests: 884
- Median Response Time: 84 ms
- 95th Percentile Response Time: 240 ms
- 99th Percentile Response Time: 1000 ms
- Average Response Time: 120.41 ms
- Minimum Response Time: 62 ms
- Maximum Response Time: 1879 ms
- Average Size of Response: 11127.83 bytes
- Current RPS: 17 requests/sec
- Current Failures/sec: 12.7 failures/sec

2.3. Aggregated Results

- Total Requests: 2609
- Failed Requests: 1765 (68% of total requests)
- Median Response Time: 130 ms
- 95th Percentile Response Time: 390 ms
- 99th Percentile Response Time: 1600 ms
- Average Response Time: 172.96 ms
- Minimum Response Time: 62 ms
- Maximum Response Time: 3320 ms
- Average Size of Response: 11348.82 bytes
- Current RPS: 33 requests/sec
- Current Failures/sec: 23 failures/sec

---

3. Analysis & Observations

- High Failure Rate: Approximately 68% of the total requests failed, indicating that there may be an issue with the server handling high concurrency or with specific endpoints.

- Response Time Issues: While most of the responses are fast, the 99th percentile time (1700 ms) shows that some requests are significantly delayed. This suggests potential bottlenecks in the system that need optimization.
- High Traffic Load: The server is handling around 33 requests per second on average, which may be putting a strain on system resources.

---

4. Recommendations for Improvement

1. Optimize Server Handling:
    - Investigate server performance under load to identify any bottlenecks.
    - Ensure that the server can handle higher traffic volumes efficiently.
2. Database Optimization:
    - Look into database query performance and optimize slow queries.
    - Consider adding indexes or optimizing database schema if needed.
3. Caching:
    - Implement caching mechanisms for frequently accessed data to reduce load on the server.
4. Scaling:
    - Consider scaling the server infrastructure horizontally to handle more concurrent users (load balancing).
5. Monitoring and Alerts:
    - Set up monitoring and alerts to detect and address performance issues early.

---

5. Conclusion

The system showed significant issues with handling high traffic, particularly in terms of high failure rates and delayed response times. It is recommended to address the optimization and scaling strategies to ensure