**Performance Test Results and Analysis**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to detail the performance test results and analysis for Jumia website. This document provides insights into the application's performance under load and highlights any issues or areas for improvement.

**1.2 Scope**

This performance test evaluates the application's behavior under various load conditions and assesses key performance metrics including response time, throughput, and error rate.

**2. Test Environment**

**2.1 Hardware and Software Configuration**

* **Client Configuration:**

**Client Machine Name :** Windows

**CPU:** Processor Intel(R) Xeon(R) CPU E3-1505M v5 @ 2.80GHz, 2808 MHz, 4 Core(s), 8 Logical Processor(s)

**RAM:** 16 GB

**Disk:** 26.5 GB

* **Test Tools:**

**Tool Used:** Apache JMeter

**Version:** 5.6.3

**3. Test Plan**

**3.1 Test Scenarios**

**Scenario 1:** Searching for a product , adding it to the cart and then removes it

**3.2 Test Configuration**

**Thread Group Settings:**

**Number of Threads (Users):** 300

**Ramp-Up Period:** 10

**Loop Count:** 1

* **HTTP Request Sampler Configuration:**

**Server Name or IP:** https ://www.jumia.com.eg/

**Path:** /

* **Assertions:**

**Response Code:** 200

**Response Content:** [Expected Content]

**4. Test Results**

**4.1 Response Time**

**Average Response Time:**

**Scenario 1:** 15411

* **Maximum Response Time:**

**Scenario 1:** 55297

* **Minimum Response Time:**

**Scenario 1:** 1087

**4.2 Throughput**

**Requests Per Second:**

**Scenario 1:** 5.3 /sec

**4.3 Error Rate**

* **Error Percentage:**

**Scenario 1:** 1.33 %

**4.4 Latency**

* **Average Latency:**

**Scenario 1:** 358

**5. Analysis**

**5.1 Average Response Time**

The average response time of 15411 milliseconds indicates the typical time it took for the server to process and respond to a request during the test. This value is crucial for assessing the overall performance and user experience. A lower average response time generally suggests better performance, though it should be compared against the expected performance criteria for the application.

**5.2 Minimum Response Time**

The minimum response time of 1087 milliseconds represents the fastest response recorded during the test. This value highlights the best-case scenario for the system's performance and provides insight into how quickly the server can handle requests under optimal conditions.

**5.3 Maximum Response Time**

The maximum response time of 55297 milliseconds indicates the slowest response observed. This value is critical for understanding the worst-case performance and identifying potential bottlenecks or issues that could affect user experience. High maximum response times may signal issues such as server overload, network congestion, or inefficient processing.

**5.4 Error Rate**

The error percentage of 1.33% indicates the proportion of requests that resulted in errors. A high error rate can impact the reliability of the application and should be investigated further to identify and resolve underlying issues.

**5.5 Throughput**

The throughput of: 5.3 /sec requests per second shows the rate at which the server handled requests during the test. High throughput indicates that the server can handle a large number of requests efficiently, which is important for scalability.

**5.6 Performance Insights**

* Optimal Performance: The average response time and minimum latency values are within acceptable ranges, suggesting that the system performs well under typical conditions.
* Potential Issues: The maximum response time and error rate indicate areas for improvement. Addressing these issues may involve optimizing server performance, improving application code, or enhancing network infrastructure.
* Scalability Considerations: The throughput value demonstrates the server's capacity to handle concurrent requests. Ensuring that the system can maintain performance with increased load is crucial for scalability.

**6. Recommendations**

**6.1 Performance Improvements**

**Recommendation for Server Downtime:**

**1. Immediate Actions**

**1.1 Verify the Outage**

* Check Monitoring Tools: Confirm that the server is down by reviewing monitoring tools and alerts. Tools such as Nagios, Zabbix, or New Relic can provide real-time status and historical data.
* Manual Check: Attempt to access the server through various methods (e.g., SSH, web interface) to ensure the issue is not isolated to specific users or services.

**1.2 Identify the Cause**

* Review Logs: Examine server logs for error messages or unusual activity that might indicate the cause of the downtime.
* Check Resource Utilization: Look into resource usage metrics (CPU, memory, disk space) to identify potential overuse or resource exhaustion.
* Verify Network Connectivity: Ensure there are no network issues affecting the server's connectivity.

**2. Short-Term Mitigation**

**2.1 Restart Services**

* Restart Server: If the issue appears to be temporary, restarting the server may resolve the problem. This can clear out any processes or resource locks that might be causing the downtime.
* Restart Affected Services: If a specific service or application is failing, restarting it may restore functionality.

**2.2 Communicate with Stakeholders**

* Inform Users: Notify users about the outage and provide updates on the estimated time for resolution. Clear and timely communication helps manage user expectations.
* Update Incident Management Team: Ensure that the relevant internal teams (IT support, operations) are aware of the situation and are working on a resolution.

**3. Long-Term Actions**

**3.1 Conduct a Root Cause Analysis**

* Analyze Data: Perform a detailed investigation into the root cause of the downtime. Review system logs, error messages, and performance metrics.
* Identify Weaknesses: Determine if there were any weaknesses in the infrastructure, configuration, or processes that contributed to the failure.

**3.2 Implement Preventive Measures**

* Update and Patch: Ensure that the server and all associated software are up-to-date with the latest patches and updates to prevent known issues.
* Enhance Monitoring: Improve monitoring tools and alerting systems to provide early warnings of potential issues.
* Review Configuration: Reassess server configurations and resource allocations to ensure they meet the current demand and workload.

**3.3 Plan for Redundancy and Failover**

* Implement Redundancy: Set up redundant systems or load balancers to distribute traffic and reduce the risk of complete downtime.
* Establish Failover Mechanisms: Develop and test failover procedures to ensure that if one server goes down, another can take over seamlessly.

**3.4 Document and Review**

* Document Incident: Record the details of the downtime, including causes, actions taken, and resolution. This documentation is valuable for future reference and training.
* Review and Improve: Conduct a review meeting with the team to discuss lessons learned and opportunities for improving response and prevention strategies.

**4. Conclusion**

Addressing server downtime requires a combination of immediate actions to restore service, short-term mitigation to manage user impact, and long-term measures to prevent recurrence. By following these recommendations, you can effectively manage the current outage and strengthen your infrastructure against future issues.

**7. Conclusion**

the performance testing of Jumia was conducted to assess the system’s responsiveness, reliability, and capacity under a simulated load of 300 concurrent users. The test provided valuable insights into the system's performance metrics, including response times and error rates.

**Key Metrics:**

* Average Response Time: 15411 milliseconds
* Minimum Response Time: 1087 milliseconds
* Maximum Response Time: 55297 milliseconds
* Error Percentage: 1.33%
* Throughput: 5.3 requests per second

**Identified Issues:**

* High Maximum Response Time: The maximum response time of 55297 milliseconds indicates occasional significant delays, which could impact user experience during peak load periods.
* Error Rate: An error percentage of 1.33 % suggests that a portion of the requests resulted in failures, which could affect overall reliability and user satisfaction.

**Recommended Actions:**

1. **Optimize Performance:**

Investigate and optimize server configurations, application code, and database queries to address high maximum response times.

Implement performance tuning to reduce latency and improve responsiveness.

1. **Enhance Reliability:**

Address the causes of errors by reviewing error logs and fixing identified issues to reduce the error rate.

Improve error handling and validation to prevent application failures.

1. **Increase Scalability:**

Evaluate and enhance infrastructure to support higher loads, including implementing load balancing and redundant systems.

Conduct further scalability testing to ensure the system can handle increased traffic without performance degradation.

1. **Strengthen Monitoring and Alerts:**

Upgrade monitoring tools and set up more granular alerts to detect and address performance issues proactively.

Establish a robust monitoring strategy to track performance metrics continuously and respond to issues before they affect users.

**Conclusion**

The performance test revealed that while the system generally performs well under load, there are areas requiring improvement. Addressing the identified issues and implementing the recommended actions will enhance system performance, reliability, and user experience.