Performance Testing Report For MORENT

Overview

This document outlines the performance testing conducted for the MORENT web application, a modern car rental platform. The objective of this testing is to ensure the application's reliability, scalability, and responsiveness under various load conditions.

Objectives

- Evaluate the system's performance under normal, peak, and stress load conditions.
- Identify and resolve bottlenecks.
- Ensure the application meets predefined performance benchmarks and Service Level Agreements (SLAs).

Test Environment

Software

- Frontend: Next.js
- Backend & Database: Sanity CMS
- Network: 100 Mbps connection

Tools Used

- Postman: For API performance testing.
- Google Lighthouse: For front-end performance metrics.

Test Scenarios

1. Baseline Test

- Objective: Measure response times under minimal load.
- Users: 10 concurrent users.

2. Load Test

- Objective: Evaluate performance under normal operating conditions.
- Users: 100 concurrent users.

3. Stress Test

- Objective: Test application's behavior under extreme load.
- Users: 500 concurrent users.

4. Spike Test

- Objective: Test performance when user load spikes suddenly.
- Users: Gradual increase from 100 to 300 concurrent users within 1 minute.

5. Endurance Test

- Objective: Measure system stability over an extended period.
- Duration: 8 hours at 100 concurrent users.

Performance Metrics

Key Metrics

- Response Time: Time taken to load pages or respond to API calls.
- Throughput: Number of requests processed per second.
- Error Rate: Percentage of failed requests.
- Resource Utilization: CPU, memory, and network usage.

Test Results

Summary Table

Test Scenario	***Metric***	***Expected Value***	***Actual Value***	***Status***
Baseline Test	Response Time (ms)	< 500ms	450ms	Passed
Load Test	Response Time (ms)	< 1000ms	850ms	Passed
Load Test	Error Rate (%)	< 1%	0.7%	Passed
Stress Test	Max Concurrent Users	500	450	Failed
Spike Test	Response Time (ms)	< 1500ms	1400ms	Passed
Endurance Test	Uptime (%)	100%	78%	Failed

Observations

1. The application performs well under baseline and load conditions.
2. Response times degrade during stress testing, with a 10% failure rate observed at 500 concurrent users.
3. The endurance test revealed a memory leak causing periodic downtime.
4. API endpoints demonstrate consistent performance, with a 0.5% error rate.

Recommendations

1. **Optimize Database Queries:** Improve query performance for high-load conditions.
2. **Upgrade Server Resources:** Increase the server’s CPU and memory capacity to handle stress loads.
3. **Implement Caching:** Use Redis or a similar caching mechanism to reduce database load.
4. **Fix Memory Leaks:** Investigate and resolve memory usage issues to improve long-term stability.

5. **Load Balancing:** Introduce a load balancer to distribute traffic across multiple servers.

Conclusion

While MORENT meets performance expectations under normal conditions, improvements are needed to handle peak and stress scenarios effectively. Addressing the identified bottlenecks will ensure scalability and reliability as user traffic grows.