Stress Test Report

Project: Dummy JSON API Testing

Prepared by: Nada Abdelrahim

Date: 22-October-2025

Contents

1.Overview

The objective of this performance test was to evaluate the stability and scalability of the Dummy JSON APIs under increasing user load (ramp-up scenario), capture p99 latency and error behavior, and identify system bottlenecks impacting API performance and reliability

2.Test Configuration

Parameter	**Value**
Executor	JMeter via Taurus
Ramp Strategy	Gradual ramp-up until SLA is violated
Assertions	p99 > 1200 ms for 30 s, failures > 2 %, success rate < 98 %
Duration	≈ 2 minutes before SLA stop triggered
Samples Collected	≈ 6 k requests
Threads (Users)	Ramp from 0 → ~400 active users

3. Key Metrics Summary

APIs	Assertion Applied
Total Samples	Status Code / Duration Assertion Json Field Exists
Failure Rate	≈ 49 % overall (73 % in final run)
Avg Response Time	22.3 s
Median Response Time	11.8 s
p99 Latency	≈ 84 s
Throughput	≈ 4 TPS
Apdex Score	0.148 → POOR user satisfaction

4.SLA Breach and Ramp Behavior

- SLA Trigger: At ~32 seconds runtime, p95 latency exceeded 1200 ms → Taurus auto-stopped the test.
- Observation: As load ramped beyond 200 threads, response times rose non-linearly and error rates spiked (> 50 %).
- Indication: Server-side bottleneck on API endpoints ("Fetch Products List" and "Login") once CPU or connection pools saturated.

5.Bottleneck Identification

API	Avg RT	Error %	Issue
Fetch Products List	11,683 ms	81 %	High I/O or DB query latency
Login	7,072 ms	85 %	Session/auth lock contention
Add To Cart	3,229 ms	64 %	Application logic delays under load

6.Graphs Highlights

All the graphs will found in html-report, for more reference please follow the setup file.

1. **Response Time Percentiles**

   Steep rise past 90th percentile → extreme tail latencies (~80 s max).

   Latency spread shows system instability under concurrency.

2. **Response Time Overview**

   ~300 requests failed or exceeded 1.5 s; only ~25 % under 500 ms.

   Confirms degradation beyond target tolerance.

3. **Time vs Threads**

   Latency oscillates violently as threads increase → thread-pool starvation or network queuing.

4. **Response Time Distribution**

   Heavy right-tail indicates occasional very slow transactions affecting user experience.

5. **Apdex & Request Summary**

   Apdex ≤ 0.15 → users mostly frustrated. Only ≈ 27 % requests met "satisfied" criteria.

6. **Error Analysis**

Error Type	% of Errors
Assertion Failed (58 %)	Most requests breached timing assertion.
SocketException (41 %)	Connection drops → server not accepting new connections.
HTTP 400 (< 1 %)	Minor functional error.

7. Hypotheses & Next Steps

- Server **saturation** — Application/thread pool limits exceeded → SocketException and timeouts.
- **Database bottleneck** — Long query response or connection pool exhaustion on Fetch Products List.
- **Auth service latency** — Login endpoint degradation under ramp load suggests synchronous blocking operations.
- **Network timeout tuning** — Connection reset errors point to insufficient keep-alive or short timeouts.
- **Infrastructure scaling** — Consider horizontal autoscaling or load balancer tuning.

8. Summary

This performance test successfully revealed the system’s SLA breaking point and main bottlenecks using **JMeter** and **Taurus**.

System performance degraded after roughly 200–300 concurrent users, with **p95 latency >17s**, **p99 up to 84s**, and **~49% failures**.

Most errors were due to **Assertion Failures (58%)** and **SocketExceptions (41%)**, indicating backend saturation.

Throughput fell below 5 TPS, and the **Apdex score (0.148)** confirmed poor user experience.

The APIs require optimization in backend concurrency, database response times, and connection handling before scaling further.