Ascend app

Stress Testing Report – Sign-In API

By Testing Team Members: Shehap Elhadary - Hamza Ahmed
Team 2

📅 Test Summary

This report outlines the results of a stress test conducted on the **Sign-In endpoint** of the system, with the objective of determining the **maximum throughput (users/second)** the server can handle before it begins to exhibit partial failures.

🎯 Test Objective

To identify the threshold at which the **Sign-In API** begins to fail under high load by incrementally increasing the number of concurrent users and adjusting the ramp-up period to simulate different request rates.

🧪 Test Configuration

Parameter	Value
Target Endpoint	`/auth/login`
Test Tool	Apache JMeter
Protocol	HTTPS
Payload Type	JSON
Authentication	N/A (Testing sign-in itself)
Server Environment	Production
Monitoring Tools	JMeter listeners (Aggregate Report, View Results Tree)

📊 Test Scenarios

Trial	Users	Ramp-up period (sec)	Load rate (users/second)	Observations
1	2000	200	10	Server stable, all requests successful
2	3000	200	15	Server stable, all requests successful
3	6000	300	20	Server stable, no errors observed
4	6000	150	40	High throughput, server maintained stability
5	9000	150	60	Elevated stress, server still performed well
6	10000	125	80	**Partial failures observed**, indicating the threshold

🧠 Analysis

- The **Sign-In API** sustained consistent performance up to **60 users/second** without degradation.
- At **80 users/second**, the server began to exhibit **intermittent request failures** (e.g., timeouts, socket errors).
- Despite partial failures, the server remained responsive — a sign of **graceful degradation**, not a crash.
- The system demonstrates **strong concurrency handling** up to at least **9000 users** with no instability.

📈 Key Findings

- The server can **withstand up to 15,000 total requests** when the **user rate is low** (i.e., longer ramp-up period), maintaining stable performance without failures.
- However, when the **user rate increases to 80 users/second** (i.e., lower ramp-up period), the server starts to **partially fail at around 10,000 users**.
- This indicates that the **rate of incoming requests** (not just total users) plays a critical role in system stability.
- Beyond 60 users/second, the backend begins to strain under load, highlighting potential optimization areas such as **thread pool sizing, database limits, or caching**.
- Importantly, no complete service outage occurred even under peak conditions, suggesting a **resilient and recoverable architecture**.

📌 Conclusion

Stress testing of the Sign-In API confirms that the system is capable of handling **up to 15,000 requests successfully** under a **gradual load increase** (low request rate per second).

As the **request rate intensifies**—specifically at **80 users/second**—the system begins to show **partial failures at around 10,000 users**, marking this rate as the **current performance threshold**.

These results demonstrate strong overall resilience, with clear insight into where **scaling limitations** emerge under pressure.