Software Test Plan (STP) - API Rate Limiter

Project:API Rate Limiter
Version: 1.0
Authors: QuadCore
Date: 03/09/2025
Status: Sample / Draft

1. Introduction

Purpose: This document outlines the test plan for the API Rate Limiter v1.0. It defines the overall strategy, scope, objectives, schedule, resources, and responsibilities for testing. The primary goal is to verify that the rate limiter effectively and correctly enforces predefined rate limits on API requests, ensuring stability and preventing abuse.

Scope: Testing covers core features of the API Rate Limiter such as enforcing request limits, handling burst traffic, generating proper error responses (e.g., HTTP 429 Too Many Requests), supporting reset/retry mechanisms, and ensuring fairness across multiple clients. Internal business logic of API endpoints and backend database operations are excluded

References: API Rate Limiter SRS v1.0, Design Specifications v1.0, HTTP RFC standards.

Definitions:

API:(Application Programming Interface)
Rate Limiter: (Middleware/service that enforces request limits per user/IP in a defined timeframe)
HTTP 429: (Standard status code for “Too Many Requests”)
SRS: Software Requirements Specification
RTM: (Requirements Traceability Matrix)

2. Test Items

- **Rate Limiting Middleware:** The core component that handles and enforces rate limits.- Cash withdrawal module
- **Request tracking Module:** Tracks the number of requests made per client.
- **Caching/State Management:** The mechanism used to store and retrieve rate limit data, which may be a simple cache or a distributed store like Redis.
- **Admin/Configuration Interface:** Allows admins to define rate limit policies.
- **Logging & Monitoring Module:** Logs blocked requests, usage statistics, and alerts for abnormal traffic patterns.

3. Features to be Tested

Features mapped to SRS requirement IDs:
-**RL-F-001:** Enforce maximum request limit per client (e.g., 100 requests/minute).
- **RL-F-002:** Block or throttle requests exceeding the allowed rate.
- **RL-F-003:** Return proper error response (`HTTP 429 Too Many Requests`) for blocked requests.
- **RL-F-004:** Support different rate limit policies.
- **RL-F-005:** Allow admin configuration of rate limits.
- **RL-NF-001:** Response time ≤ 200 ms for rate limiting decision.
- **RL-NF-002:** High availability (99.9%) under normal and peak loads.

4. Features Not to be Tested

- Business logic of the underlying API endpoints.
-External database or cache performance optimizations
- Third-party API integrations or external gateway services.

5. Test Approach / Strategy

Levels:

- **Unit Tests (module-level):** Validate request counter, block handler, retry timer, and logging functions individually.
- **Integration tests**: Verify that the rate limiter integrates correctly with the API gateway or the protected application.
- **System Tests (end-to-end):** Ensure complete rate-limiting workflow.
- **Acceptance tests**: Verification that the rate limiter meets the business and security requirements of the application.

Types:

- **Functional Testing:** Verify core rate-limiting rules, request blocking, retry-after behavior.
- **Regression testing**: Ensure new code changes or bug fixes don't break existing rate-limiting functionality.
-**Performance Testing:** Measure response latency, throughput under peak load, burst handling.
- **Scalability Testing:** Validate system's ability to handle large-scale traffic.

Entry Criteria: Stable build of API Rate Limiter delivered. Test environment and tools configured.

Exit Criteria: 100% of planned test cases executed. 0 critical defects open. All acceptance criteria satisfied.

5.1 Security Validation

- Validate that no sensitive client information (tokens, IPs) is logged in plain text.

- Ensure secure handling of client identifiers .
- Validate use of TLS 1.2+ or higher for secure communication.
- Fuzz testing with malformed requests to ensure the limiter handles them gracefully.
- Penetration testing to confirm the limiter cannot be bypassed by attackers .

6. Test Environment

Hardware: Standard server/VM or Cloud/on-prem test servers.
Software: API Rate Limiter v1.0
Tools: Postman (manual API testing), JMeter (load & performance testing), cURL / Python scripts (for automated request generation) and Jira (defect tracking)
Test Data: Dummy API keys, user IDs, IP addresses

7. Test Schedule

Milestones:
- Test case design: 10-Sep-2025
- Environment setup: 12-Sep-2025
- Test execution start: 13-Sep-2025
- Test execution end: 22-Sep-2025
- UAT: 24-Sep-2025 to 26-Sep-2025

8. Test Deliverables

- Test Plan (this document)
- Test Cases (manual & automated)
- Test Scripts
- Test Data
- Test Execution Logs
- Defect Reports
- Test Summary Report

9. Roles and Responsibilities

Role	Name	Responsibility
QA Lead	Ankita Muni	Prepare plan, coordinate execution
Test Engineer	Manasa S A	Design & execute test cases, log defects
Developer	Likitha H	Support defect fixes and triage
Product Owner	Arya S	Approve test results, sign-off readiness

10. Risks and Mitigation

Risk	Mitigation
Delay in stable build delivery	Request early smoke builds from dev team
Test environment downtime	Maintain backup environment on cloud VM
Dependency on external API gateways or middleware	Create stubs/mocks to simulate gateway behavior during testing
High traffic simulation may overload test infrastructure	Use cloud-based load testing tools (e.g., JMeter on distributed mode) or throttle simulation batches

11. Assumptions & Dependencies

- APIs under test will be stable and accessible.
- Test data (API keys, users) will be available before execution.
- Required tools and environment (Postman, JMeter, server) will be ready.

12. Suspension & Resumption Criteria

Suspend testing if:

- API Environment unavailable for >4 hours
- Build is too unstable and blocks more than 30% of test cases.

Resume testing if:

- Critical/blocking defects are fixed
- API environment is stable and accessible.

13. Test Case Management & Traceability

The Requirements Traceability Matrix (RTM) ensures that each SRS requirement is mapped to corresponding test cases for complete coverage.

Example:

- RL-F-001 (Enforce request limit) → TC-RL-01, TC-RL-02
- RL-F-003 (Return HTTP 429 error) → TC-ERR-01, TC-ERR-02
- RL-F-004 (Retry-After header support) → TC-Retry-01
- RL-NF-001 (Response time ≤ 200 ms) → TC-Perf-01
- RL-NF-003 (Scalability under burst traffic) → TC-Scale-01

14. Test Metrics & Reporting

Metrics collected:

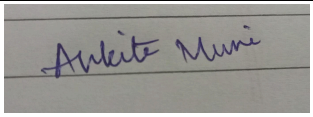
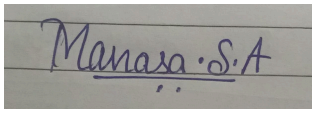
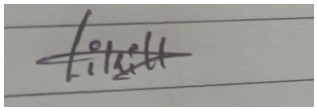
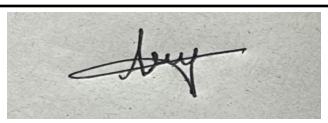
- % test cases executed
- % passed/failed
- Defect density (defects per module/feature)
- Defect aging (time taken to fix critical issues)

- Requirement coverage (mapped via RTM)

Reports:

- Daily execution status
- Final Test Summary Report

15. Approvals

Role	Name	Signature / Date
QA Lead	Ankita Muni	
Test Engineer	Manasa S A	
Dev Lead	Likitha H	
Product Owner	Arya S	 03/09/2025