**FUNCTIONAL AND PERFORMANC TESTING**

|  |  |
| --- | --- |
| Date | 29-05-2025 |
| Team ID | LTVIP2025TMID28821 |
| Project Name | TO SUPPLY LEFTOVER FOOD TO POOR |
| Maximum Marks | 4 Marks |

**Performance Testing Strategy**

**1. Introduction**

Performance testing is a critical quality assurance activity that evaluates how well an application behaves under various workloads. In the context of the project *“To Supply Leftover Food to the Poor,”* which aims to manage time-sensitive food donations via the Salesforce-powered NourishBridge platform, performance testing ensures system responsiveness, reliability, scalability, and stability.

This section outlines the objectives, tools, scope, metrics, scenarios, and outcomes related to performance testing for the project’s end-to-end ecosystem comprising donors, NGOs, volunteers, and administrators.

**2. Objectives of Performance Testing**

The core goals of the performance testing strategy include:

* Verifying that all pages and components load within the defined thresholds (e.g., 3 seconds or less).
* Ensuring the backend automation (flows, Apex triggers) completes within expected limits.
* Confirming system behavior under concurrent user load (scalability testing).
* Measuring API and integration response times (e.g., Twilio SMS delivery).
* Detecting and resolving any performance bottlenecks before go-live.

**3. Scope of Performance Testing**

The following modules and processes were included in the scope of testing:

* Donor Portal
  + Login, Post Donation form, Dashboard views.
* NGO Portal
  + Donation claim process, Volunteer assignment, Status updates.
* Admin Panel
  + Verification workflows, dashboard reporting, user audit logs.
* Notification System
  + Email/SMS delivery and queue performance.
* Automated Matching Engine
* Execution time for identifying eligible NGOs from available donations.

**4. Key Performance Metrics**

| Metric | Target Threshold |
| --- | --- |
| Page Load Time | ≤ 3 seconds |
| Concurrent Users | ≥ 100 users |
| Flow Execution Time | ≤ 1 second per flow |
| Apex CPU Time | ≤ 5000 ms per transaction |
| SMS Delivery Latency | ≤ 30 seconds |
| Dashboard Refresh | ≤ 5 seconds for 100K records |

These thresholds were based on Salesforce best practices, platform governor limits, and usability expectations of non-technical users.

**5. Tools and Environment Setup**

**5.1 Testing Tools**

* JMeter: Simulated concurrent users accessing donor and NGO portals.
* Salesforce Developer Console: Tracked Apex execution times and SOQL/DML usage.
* Chrome DevTools: Measured client-side load times for LWC-based components.
* Log Inspector: Identified flow performance and decision-point delays.
* Twilio Console: Monitored SMS dispatch status and delivery timestamps.

**5.2 Test Environment**

* A Salesforce sandbox environment configured with production-like data (~100,000 donation records).
* External Twilio sandbox linked with Salesforce for messaging simulations.
* Simulated load generated by test users with varied roles and regional access.

**6. Test Scenarios**

Several performance scenarios were executed to simulate real-world usage:

**6.1 Load Testing**

* Simulated 100 users logging in simultaneously, posting donations, and accessing dashboards.
* Observed portal response times, record submission delays, and CPU usage.

**6.2 Stress Testing**

* Increased concurrent users to over 250 to assess system limits under extreme load.
* Tested record insertions beyond normal thresholds to evaluate system degradation.

**6.3 Soak Testing**

* Kept the system under load for 4 continuous hours with staggered traffic.
* Verified memory leaks, log size accumulation, and flow behavior over time.

**6.4 Spike Testing**

* Added 30–40 concurrent donation posts within 30 seconds.
* Observed performance of matching logic and notification queue**.**

**7. Performance Bottlenecks Identified & Resolutions**

| Issue | Observation | Resolution |
| --- | --- | --- |
| Apex CPU Limit Breach | Detected in donor matching logic | Refactored loop logic and bulkified queries |
| Slow Dashboard Load | When dataset exceeded 50K records | Added filters and optimized report types |
| SMS Delay in Peak Load | Slight delay (>30 seconds) | Enabled asynchronous queue batch processing |

**8. Test Results Summary**

| Module | Avg. Load Time | Status |
| --- | --- | --- |
| Donor Login & Dashboard | 2.4 seconds | Pass |
| Donation Submission | 1.7 seconds | Pass |
| NGO Matching Flow | 1.2 seconds | Pass |
| Admin Dashboard Reports | 4.8 seconds (with filters) | Pass |
| Twilio SMS Notification | Avg. 19.6 seconds | Pass |

All major processes passed with acceptable buffer under the defined thresholds.

**9. Post-Testing Recommendations**

* Use batch processing for heavy reporting workloads.
* Schedule dashboard refreshes during off-peak hours if datasets grow.
* Continue monitoring Twilio API quotas and upgrade if traffic increases.
* Adopt Salesforce Event Monitoring tokeep track of real-time performance in production.

**10. Conclusion**

The performance testing strategy for *“To Supply Leftover Food to the Poor”* validated the platform’s ability to handle real-time donation and distribution activities effectively. By ensuring fast response times, smooth automation, and scalable design, the platform is now fully equipped to support NGOs and donors without disruption—ensuring that each surplus meal reaches someone in need, on time**.**