**Performance Test Report - Contact List Application**

* Project Name                : Contact List Application API
* Date                                  : 03/06/2025
* Tested By                       : Adit Kurniawan
* Environment                : Public API
* Tool Used                       : JMeter 5.6.3
* Device/OS                      : Intel I5, 20GB RAM, Windows (64-bit)
* Network Speed            : Upload 22 Mbps / Download 46 Mbps
* Target Endpoint          :
  + <https://thinking-tester-contact-list.herokuapp.com/users/login>
  + <https://thinking-tester-contact-list.herokuapp.com/contacts>
  + [https://thinking-tester-contact-list.herokuapp.com/contacts/${contactId}](https://thinking-tester-contact-list.herokuapp.com/contacts/$%7bcontactId%7d)

**1. Objective**

The purpose of this performance test is to evaluate how the Contact List application behaves under increasing user loads, ranging from 50 to 200 Virtual Users (VU) over a fixed duration. The goal is to assess the application's scalability, response consistency, and identify performance bottlenecks across different CRUD operations.

**2. Test Scenarios**

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Virtual Users (VU)** | **Ramp-up Period (s)** |
| 1 | 50 | 30 |
| 2 | 100 | 30 |
| 3 | 200 | 30 |

**3. Key Metrics Captured**

* Average Response Time (ms): The average time it takes for the server to respond to a request.
* 90th Percentile Response Time (ms): 90% of requests are completed at or below this time
* Max Response Time (ms): The slowest single response recorded during the test.
* Throughput (requests/second): The number of requests the server can handle per second.
* Error Rate (%): The percentage of failed requests (e.g., timeouts, HTTP 500 errors).

**4. Test Results Summary**

Overall Performance by Load Level

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| VU | Avg Resp Time (ms) | P90 Resp Time (ms) | Max Resp Time (ms) | Throughput (req/s) | Error Rate (%) |
| 50 | 743 | 986 | 2695 | 7.06 | 0.00 |
| 100 | 1146 | 2459 | 3230 | 13.54 | 11.67 |
| 200 | 6533 | 11128 | 12361 | 14.35 | 68.39 |

Detailed Performance by Operation (100 VU Test)

**5. Error Analysis**

**Error Distribution by Load Level**

* **50 VU**: No errors (0.00%)
* **100 VU**: 11.67% error rate (70 failed requests out of 600 total)
* **200 VU**: 68.39% error rate (844 failed requests out of 1200 total)

**Error Types Identified**

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Type** | **Count** | **% of Errors** | **% of All Samples** |
| 404 Not Found | 66 | 94.29% | 11.00% |
| 400 Bad Request | 4 | 5.71% | 0.67% |

**Operations Most Affected by Errors**

1. **Delete Contact**: 40% error rate at 100 VU
2. **Patch Contact**: 30% error rate at 100 VU
3. **Get Contact**: Significant errors at 200 VU load

**6. Observations**

* **Baseline Performance (50 VU)**: The application performs well with no errors and acceptable response times (average 743ms).
* **Moderate Load (100 VU)**: Performance degrades significantly with 11.67% error rate. User Login operation shows the highest response times (2155ms average).
* **High Load (200 VU)**: System becomes unstable with 68.39% error rate, indicating capacity limits exceeded. Response times increase dramatically (6533ms average).
* **Critical Issues Identified**:
  + High 404 Not Found errors suggest resource availability issues or race conditions
  + Delete and Patch operations are particularly vulnerable to failures
  + User Login operation is a performance bottleneck across all load levels
* **Throughput Plateau**: Throughput peaks at around 14.35 req/s at 200 VU, indicating system capacity limits.

**7. Root Cause Analysis**

* **404 Errors**: Likely caused by:
  + Race conditions where contacts are deleted before other operations complete
  + Insufficient data cleanup between test iterations
  + Database connection pool limitations
* **Performance Degradation**: Attributed to:
  + Database query optimization issues
  + Insufficient connection pooling
  + Lack of proper indexing on contact tables
  + Authentication/session management bottlenecks

**8. Recommendations**

**Immediate Actions**

1. **Fix 404 Errors**: Implement proper data isolation and cleanup mechanisms
2. **Optimize User Login**: Cache authentication tokens and implement session pooling
3. **Database Optimization**: Add proper indexing and optimize CRUD queries
4. **Connection Pool Tuning**: Increase database connection pool size

**Performance Improvements**

1. **Implement Caching**: Add Redis or in-memory caching for frequently accessed contacts
2. **Database Scaling**: Consider read replicas for GET operations
3. **Load Balancing**: Implement horizontal scaling for higher concurrency
4. **Rate Limiting**: Add proper rate limiting to prevent system overload

**Testing Recommendations**

1. **Gradual Load Increase**: Test with incremental load steps (75, 125, 175 VU)
2. **Soak Testing**: Run extended duration tests to identify memory leaks
3. **Database Monitoring**: Monitor database performance metrics during tests
4. **Error Handling**: Improve application error handling and logging

**9. Conclusion**

The Contact List application demonstrates acceptable performance under light loads (50 VU) but shows significant degradation as load increases. The system appears to have a capacity limit around 100 VU, beyond which error rates become unacceptable for production use.

**Key Findings**:

* **Stable Load Capacity**: ~50 concurrent users
* **Maximum Recommended Load**: 75-80 concurrent users
* **Critical Bottlenecks**: User authentication and database operations
* **Primary Issue**: High error rates due to resource contention

**Production Readiness**: The application requires significant optimization before handling production-level traffic. Focus should be on database optimization, error handling, and implementing proper caching mechanisms.