Multi-Tier Image Processing System

Amandeep - 25M0800

Overview

This project implements a multi-tier architecture for an image processing system. The system allows users to upload, download, and process images through a web-based frontend. The backend handles all computationally intensive tasks such as image resizing, conversion, thumbnail generation, and filtering. A PostgreSQL database is used to store metadata for both uploaded and processed images, while local file storage maintains the actual image files.

Architecture

The architecture follows a three-tier design pattern:

- **Client Tier:** Consists of frontend interface and users. Users can upload images, request operations, and download results.
- **Application Tier:** The backend server manages all image processing operations using worker pools and processing modules.
- **Data Tier:** Responsible for persistent storage. The PostgreSQL database stores metadata, while disk storage maintains the original and processed image files.

Architecture Diagram

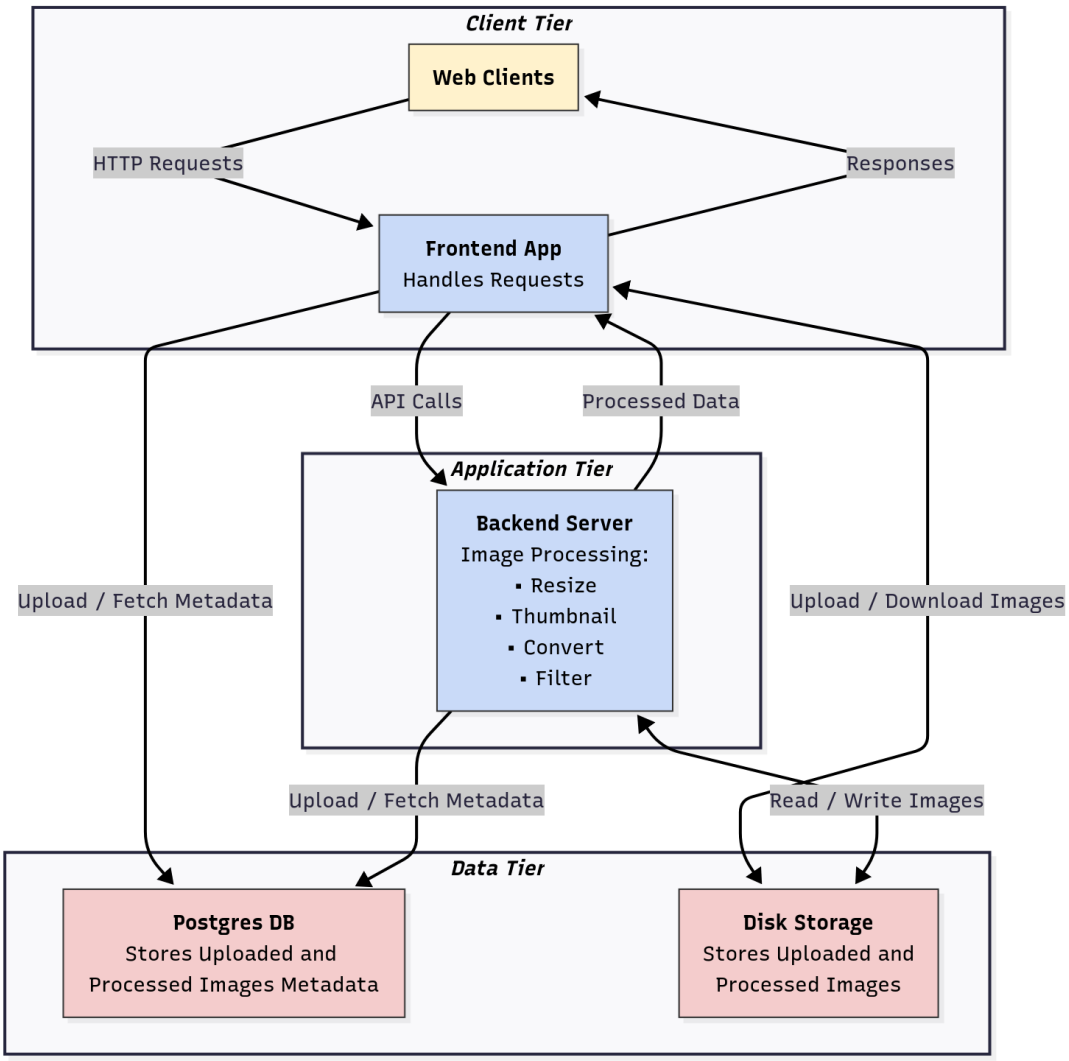

Figure 1: System Architecture Diagram

Disk I/O Bottleneck Analysis

Test Configuration

- **Test Duration:** 10 seconds per concurrent client count
- **Allocated Cores:** 10 cores on frontend server
- **Client Concurrency:** Varied from 1 to 99 clients (incremented by 2)
- **Rate Limiting:** 10 operations per second per client
- **Total Max Throughput:** 990 operations/sec (99 clients × 10 ops/sec)
- **Test Phases:** Upload phase followed by deletion phase
- **Image Size:** 32000x32000 px 16MB (per request size)

Key Observations

1. Throughput vs Concurrent Clients

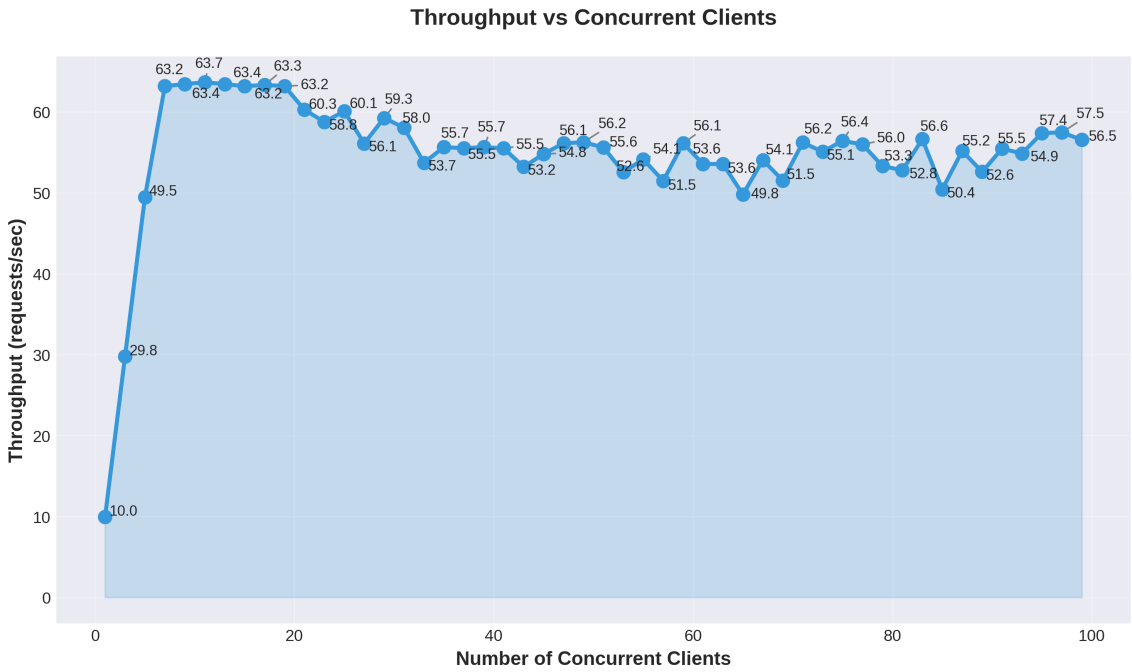

Figure 2: Throughput vs Number of Concurrent Clients

Observations:

- Throughput starts at approximately **10 requests/sec** with a single client and peaks at around **65 requests/sec** at 99 concurrent clients.
- Despite increasing the number of concurrent clients from 1 to 99, the overall system throughput **decreases by approximately 35%**, indicating severe disk I/O contention.

- The declining throughput pattern suggests that disk I/O operations (reading/writing files) become increasingly serialized as the number of competing threads increases.

2. Response Time vs Concurrent Clients

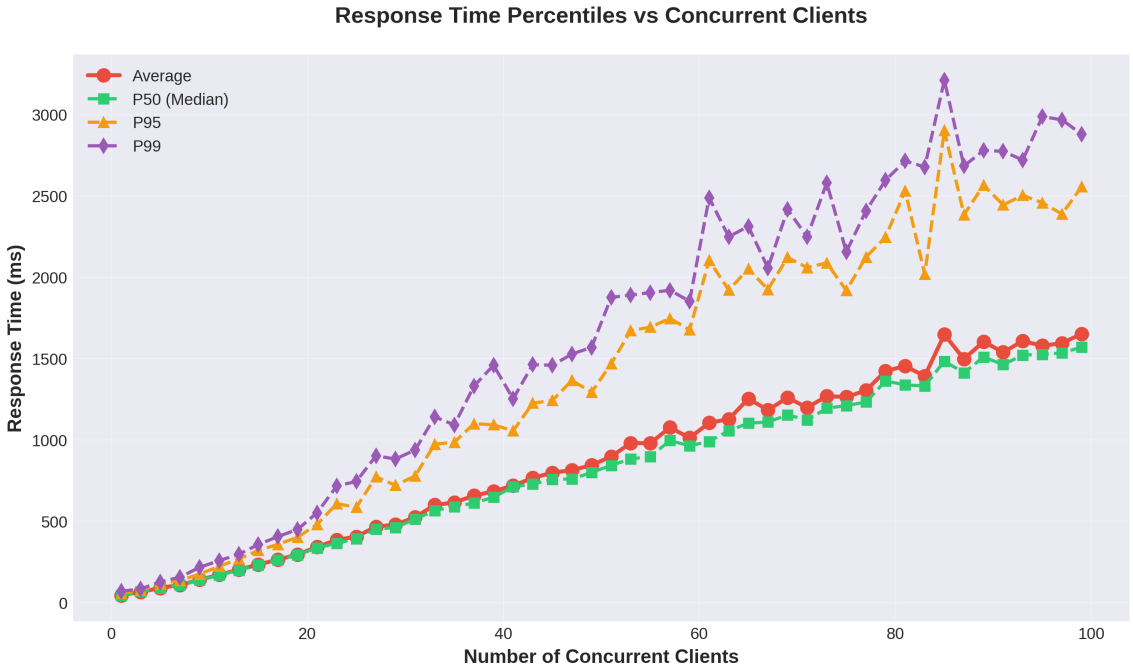

Figure 3: Response Time Percentiles vs Concurrent Clients

Observations:

- **Average response time** increases from **44ms** (1 client) to **1651ms** (99 clients), a **3,650% increase**.
- **P50 (Median) latency** grows from 43ms to 1,569ms, indicating that even median response times are severely impacted by concurrency.
- **P95 latency** escalates dramatically, reaching 2,555ms at 99 clients, meaning 95% of requests experience significant delays.
- **P99 latency** reaches **2,879ms**, indicating that worst-case scenarios become commonplace at high concurrency levels.

3. Latency Variance & Stability

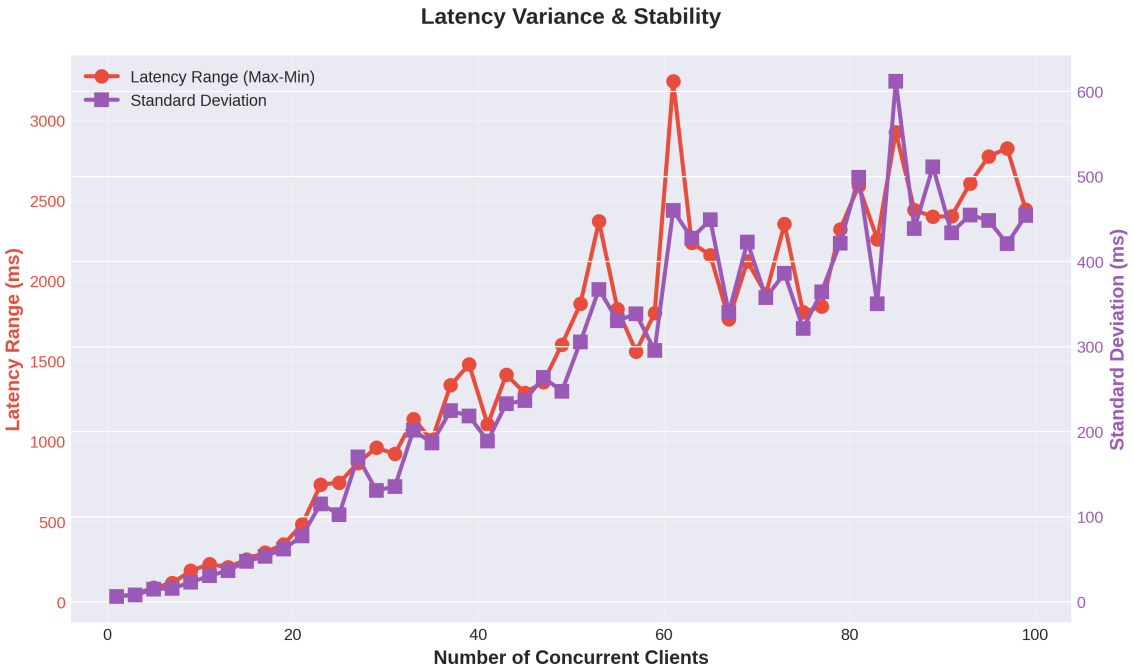

Figure 4: Latency Variance & Stability Analysis

Observations:

- **Latency range** (Max - Min) increases exponentially from **34ms** (1 client) to **2,780ms** (99 clients).
- **Standard deviation** grows from **5.96ms** to **453.96ms**, indicating highly unpredictable response times at scale.
- The exponential growth in both metrics demonstrates that system behavior becomes increasingly chaotic and unpredictable as concurrency increases.
- High variance means that clients experience highly inconsistent performance: some requests complete quickly while others experience severe delays.

4. Success Rate

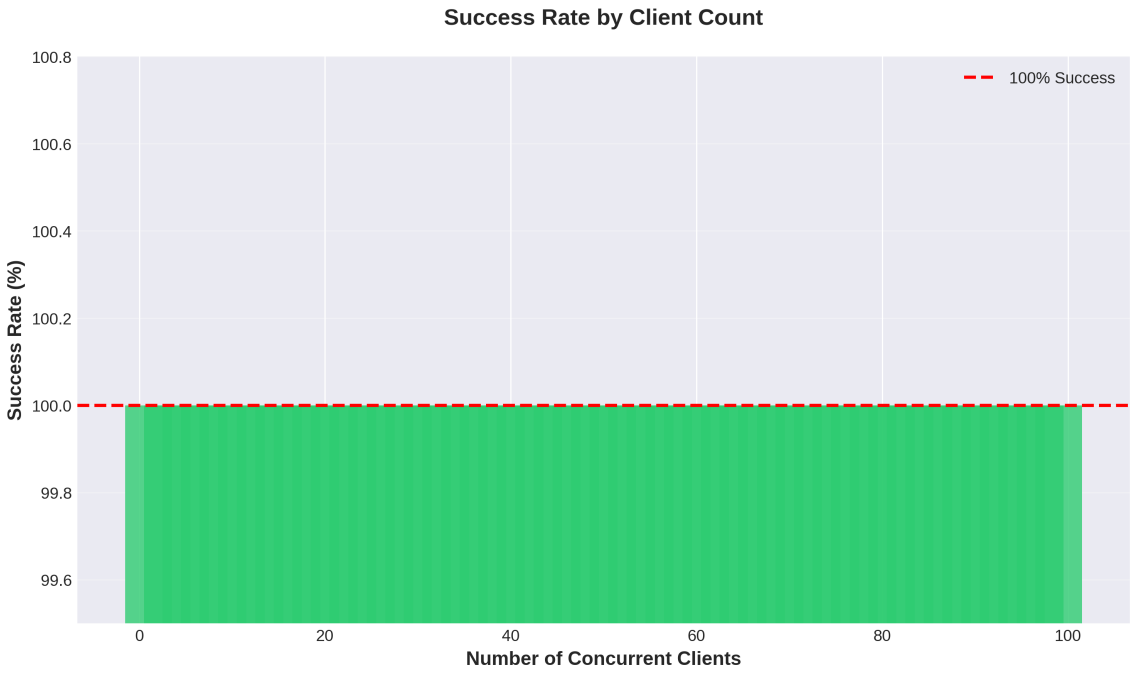

Figure 5: Success Rate by Client Count

Observations:

- **100% success rate** maintained across all concurrency levels (1 to 99 clients).
- No failed uploads or deletions were recorded, indicating excellent **system reliability**.
- Despite severe performance degradation under high concurrency, the system remains **stable** and **fault-tolerant**.
- All 15,000+ total requests completed successfully, demonstrating robust disk I/O error handling and recovery mechanisms.

5. System Load Saturation

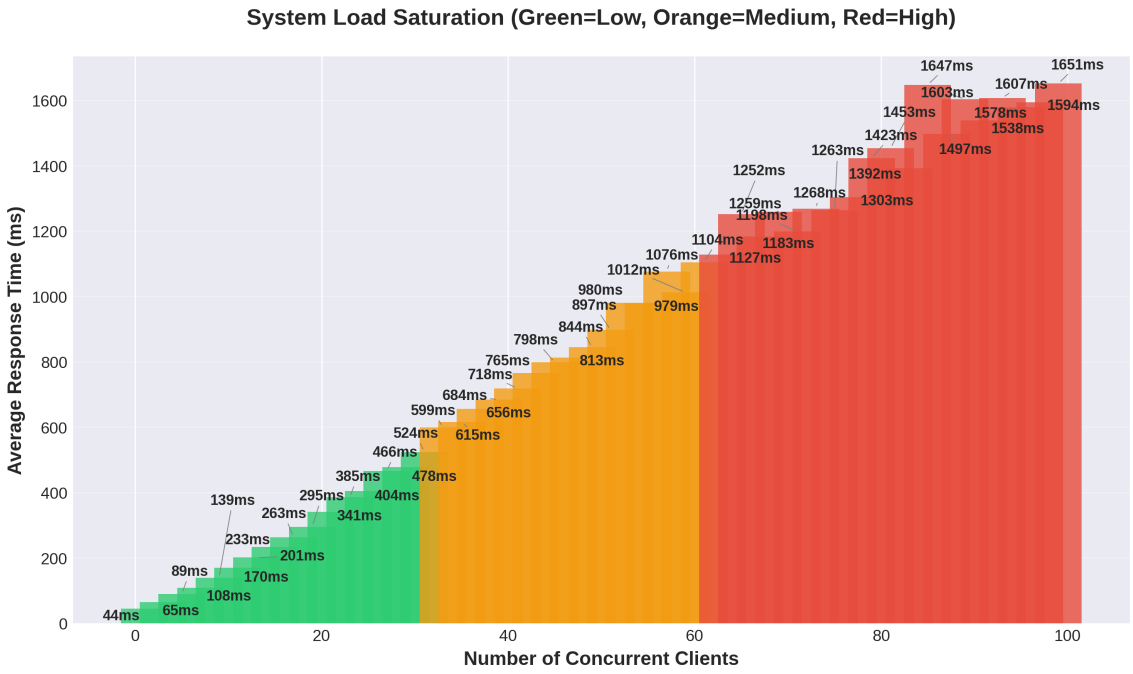

Figure 6: System Load Saturation Indicator

Observations:

- System enters the **"orange" zone (medium load)** around 20-25 concurrent clients (response time 350-400ms).
- System transitions to **"red" zone (high saturation)** beyond 35 concurrent clients (response time $>$ 600ms).
- At 99 concurrent clients, response time reaches **1,651ms**, indicating extreme saturation where disk I/O becomes the critical bottleneck.
- The saturation curve reveals that with 10 cores and rate-limited clients, the system can comfortably handle up to **15-20 concurrent clients** before experiencing noticeable degradation.

CPU vs Disk Utilization During Disk I/O Test

Test Configuration

- **Monitoring Duration:** Full test run (180 seconds)
- **Metrics Collected:** CPU percentage and Disk utilization percentage
- **Sampling Interval:** Every 2 seconds
- **CPU Cores Available:** 10 cores
- **Maximum Theoretical CPU:** 100% (1000% for 10 cores combined)

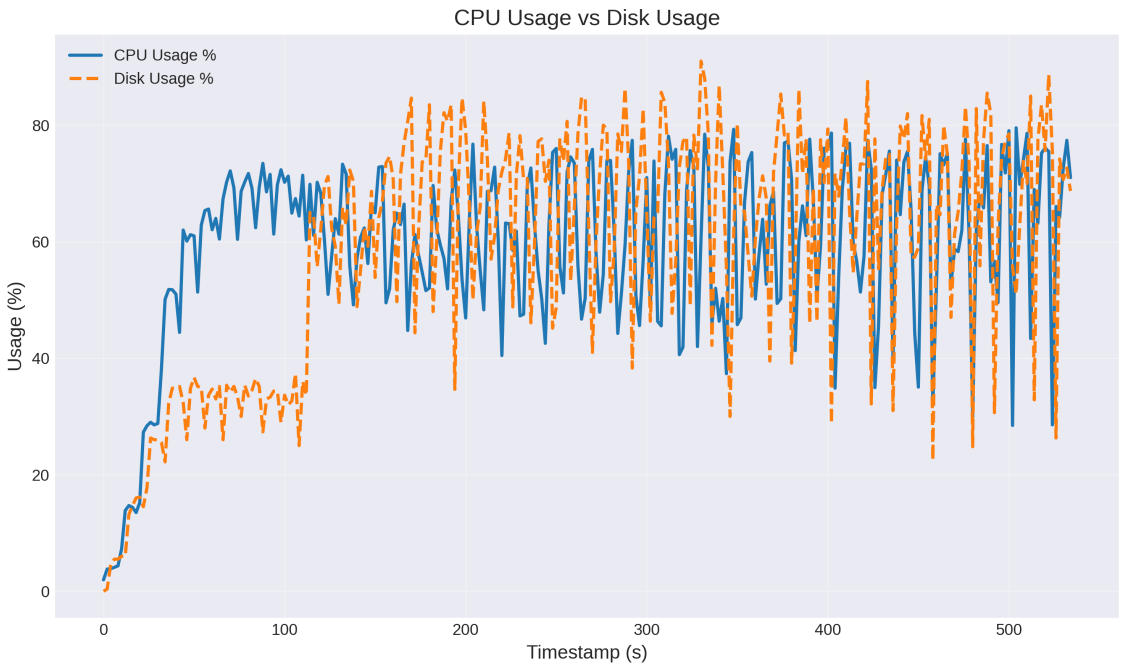

Figure 7: CPU Usage vs Disk Utilization During Disk Bottleneck Test

Disk I/O Bottleneck Conclusion

The disk I/O bottleneck analysis reveals that the system's performance is limited by disk throughput and latency, not CPU or memory. With only 10 allocated cores and mechanical disk storage, the system exhibits severe performance degradation as concurrency increases beyond 20-25 clients. Response times increase from 44ms (1 client) to 1,651ms (99 clients), while throughput paradoxically decreases due to increased context switching and I/O contention. Despite extreme performance degradation, the system maintains 100% reliability and data integrity, demonstrating robust error handling.

CPU Bottleneck Analysis

Test Configuration

- **Test Duration:** 10 seconds per concurrent client count
- **Backend Cores:** 5 cores (pinned)
- **Frontend Cores:** 2 cores (pinned)
- **Client Concurrency:** Varied from 5 to 200 clients (incremented by 5)
- **Rate Limiting:** 10 operations per second per client
- **Test Focus:** CPU-intensive image processing operations on backend
- **Key Difference:** Backend pinned to only 5 cores to create CPU bottleneck
- **Image Size:** 64x64 4KB

Key Observations

1. Throughput vs Concurrent Clients

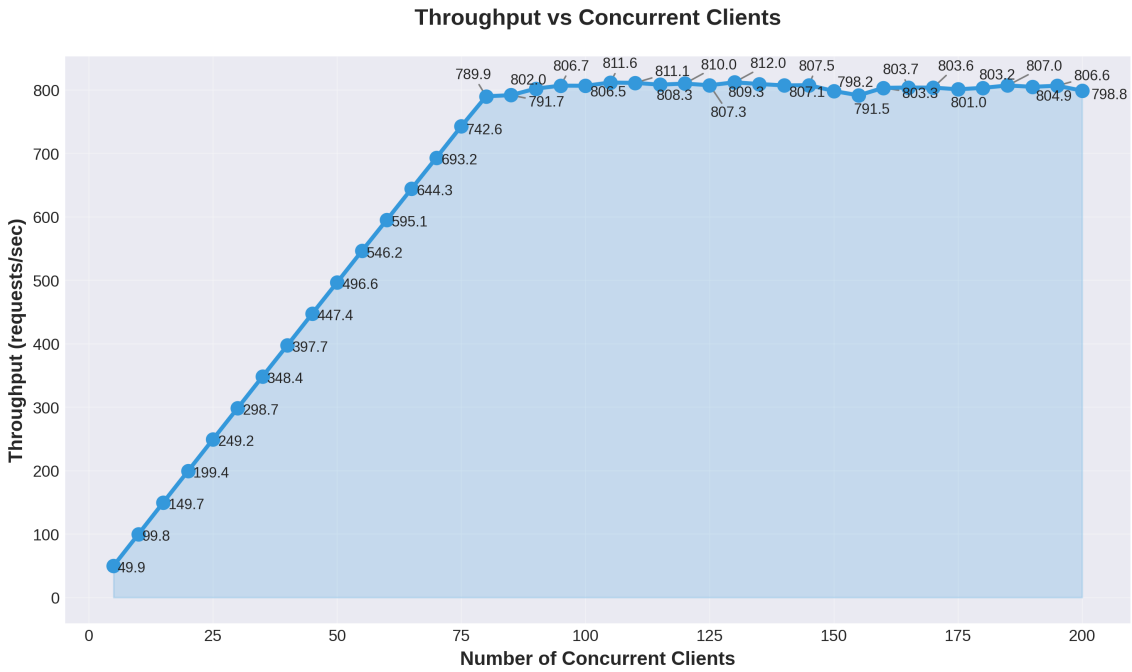

Figure 8: CPU Bottleneck: Throughput vs Number of Concurrent Clients

Observations:

- Throughput starts at approximately **50 requests/sec** with 5 clients and increases rapidly to **800+ requests/sec** at peak.

- Throughput **initially increases** as more clients are added, reaching maximum around **80-85 concurrent clients**.
- Beyond 80 clients, throughput **plateaus** at approximately **800-810 requests/sec**, indicating the system has reached its maximum CPU processing capacity with 5 backend cores.
- The plateau demonstrates clear **CPU saturation** - adding more clients does not increase throughput, confirming CPU as the bottleneck.

2. Response Time vs Concurrent Clients

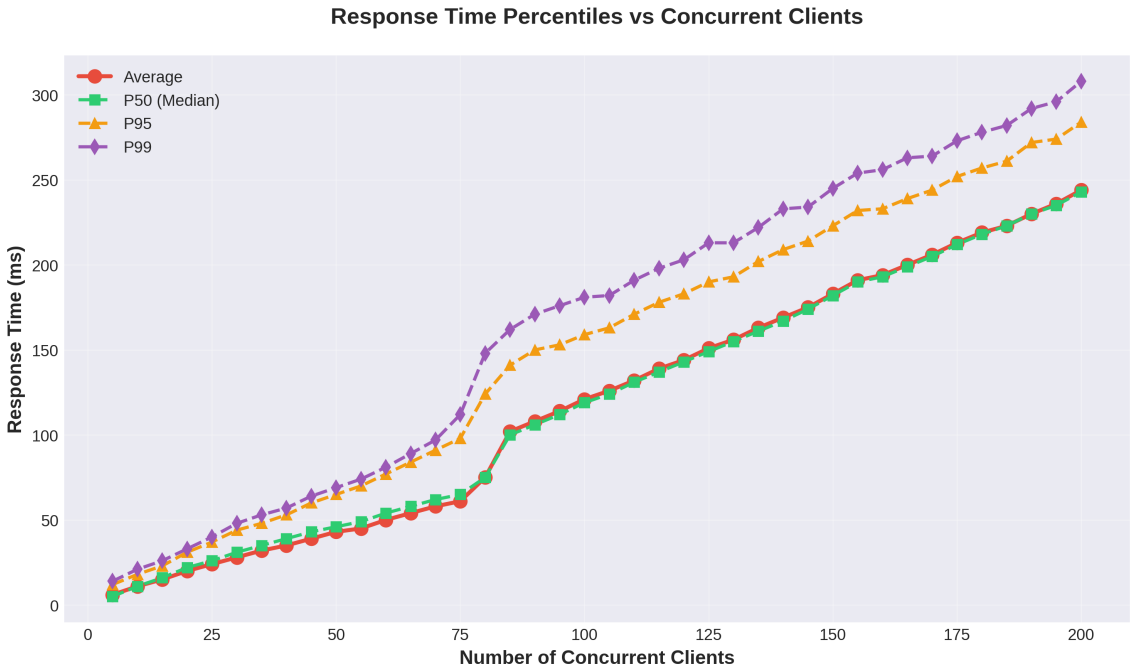

Figure 9: CPU Bottleneck: Response Time Percentiles vs Concurrent Clients

Observations:

- **Average response time** increases from **6ms** (5 clients) to **244ms** (200 clients), a **4,000% increase**.
- **P50 (Median) latency** grows from 5ms to 243ms, showing linear degradation as CPU cores become saturated.
- **P95 latency** increases from 12ms to 284ms, demonstrating relatively consistent behavior even at high loads.
- **P99 latency** reaches **308ms** at 200 clients, significantly better than the disk I/O bottleneck (2,879ms).

3. Latency Variance & Stability

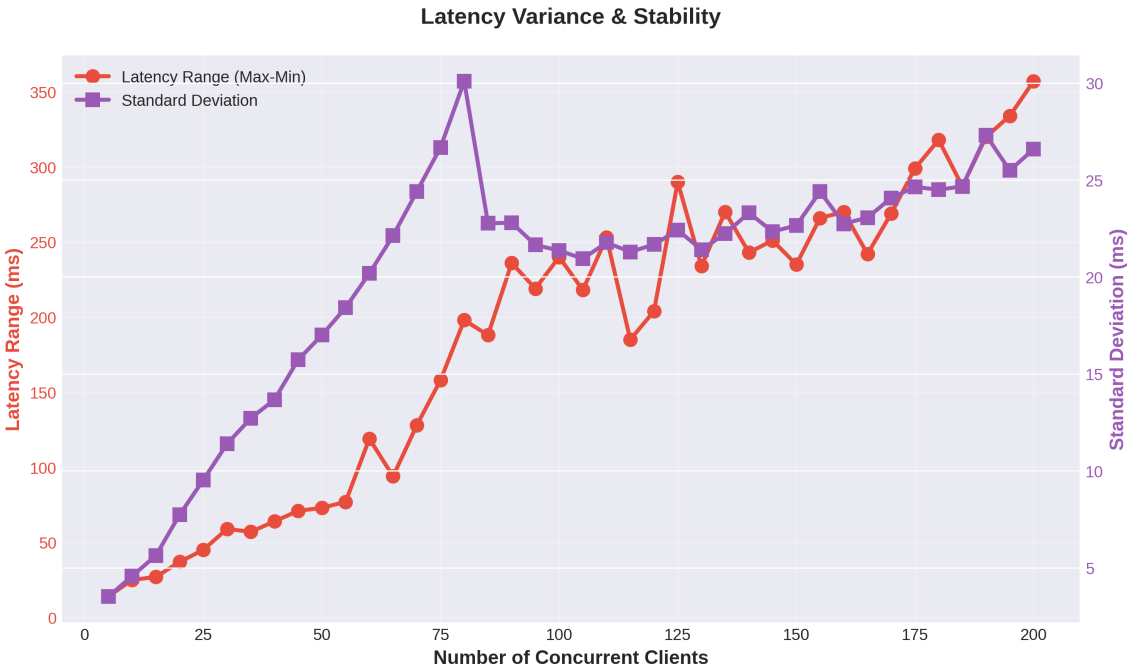

Figure 10: CPU Bottleneck: Latency Variance & Stability Analysis

Observations:

- **Latency range** (Max - Min) increases from **14ms** (5 clients) to **357ms** (200 clients).
- The relatively flat standard deviation curve indicates that CPU-bound operations provide **consistent performance characteristics** even under heavy load.
- Maximum latency range is **357ms**, compared to **2,780ms** in disk I/O test - a **7.8× difference**.

4. Success Rate

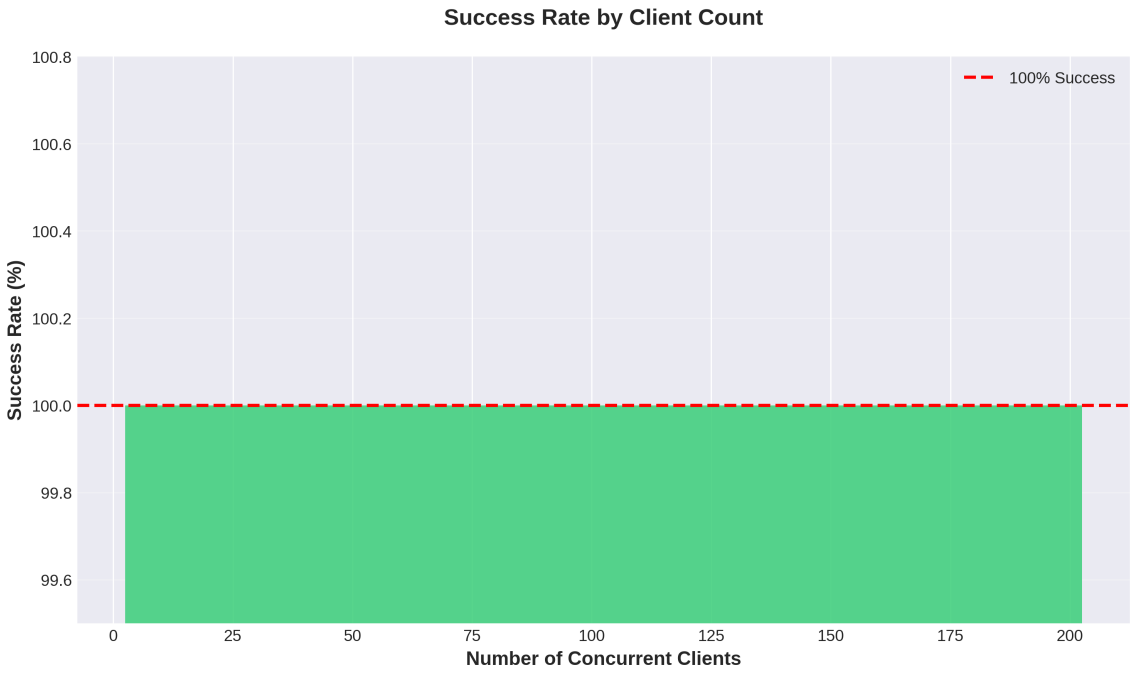

Figure 11: CPU Bottleneck: Success Rate by Client Count

Observations:

- **100% success rate** maintained across all concurrency levels (5 to 200 clients).
- No failed processing requests recorded, demonstrating excellent **system reliability** even under CPU saturation.
- Over **320,000 total requests** completed successfully during the test.
- The system remains **stable and fault-tolerant** even when CPU resources are fully saturated.

5. System Load Saturation

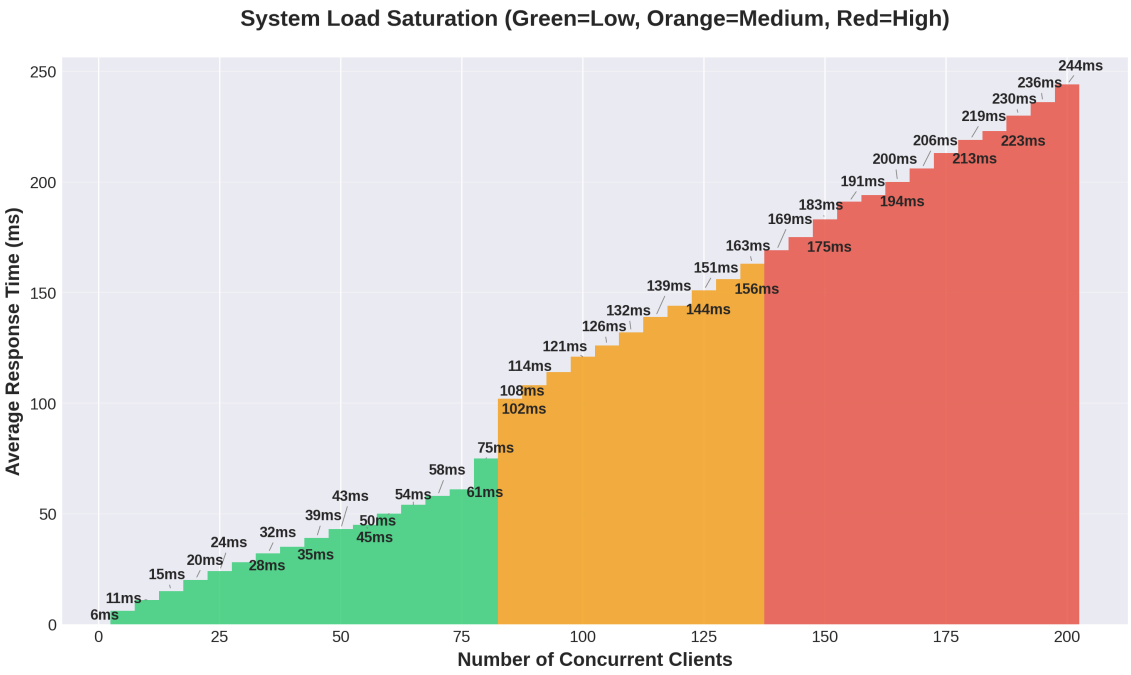

Figure 12: CPU Bottleneck: System Load Saturation Indicator

Observations:

- System remains in **"green" zone (low load)** up to approximately **40-50 concurrent clients** (response time < 50ms).
- System enters the **"orange" zone (medium load)** between 50-100 concurrent clients (response time 50-150ms).
- System transitions to **"red" zone (high saturation)** beyond 100 concurrent clients (response time > 150ms).
- With 5 backend cores, the system can comfortably handle **40-50 concurrent clients** before experiencing noticeable degradation.

CPU vs Disk Utilization During CPU Bottleneck Test

Test Configuration

- **Monitoring Duration:** Full test run (102 seconds)
- **Metrics Collected:** CPU percentage and Disk utilization percentage
- **Sampling Interval:** Every 2 seconds
- **Backend Cores:** 5 cores (pinned for CPU-intensive processing)
- **Frontend Cores:** 2 cores (pinned)

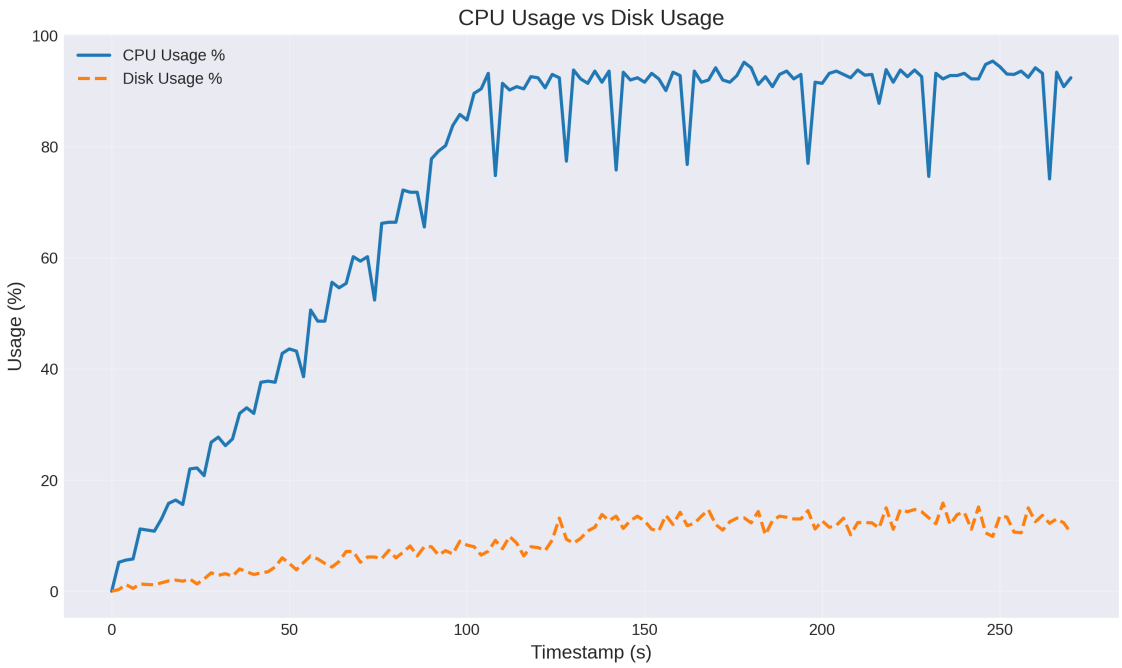

Figure 13: CPU Usage vs Disk Utilization During CPU Bottleneck Test

CPU Bottleneck Conclusion

The CPU bottleneck analysis reveals fundamentally different performance characteristics compared to disk I/O bottleneck:

- **Throughput scales linearly** up to CPU saturation point (800 req/sec), then plateaus rather than declining.
- **Response time degradation is linear and predictable** (6ms → 244ms), not exponential like disk I/O (44ms → 1,651ms).
- **Latency variance is much lower** (std dev: 3.52ms → 26.59ms vs 5.96ms → 453.96ms), providing consistent performance.
- **CPU utilization reaches 89.6%** while disk remains at 3-7%, confirming CPU as the bottleneck.
- **System handles 200 concurrent clients** with 244ms response time vs only 99 clients with 1,651ms in disk I/O test.
- **100% success rate maintained** across 320,000+ requests, demonstrating robust reliability.

Conclusion

This comprehensive analysis demonstrates that **not all bottlenecks are equal**. While both disk I/O and CPU bottlenecks limit system performance, they exhibit fundamentally different characteristics.

Github Link

https://github.com/amandeep2102/Image_Processor