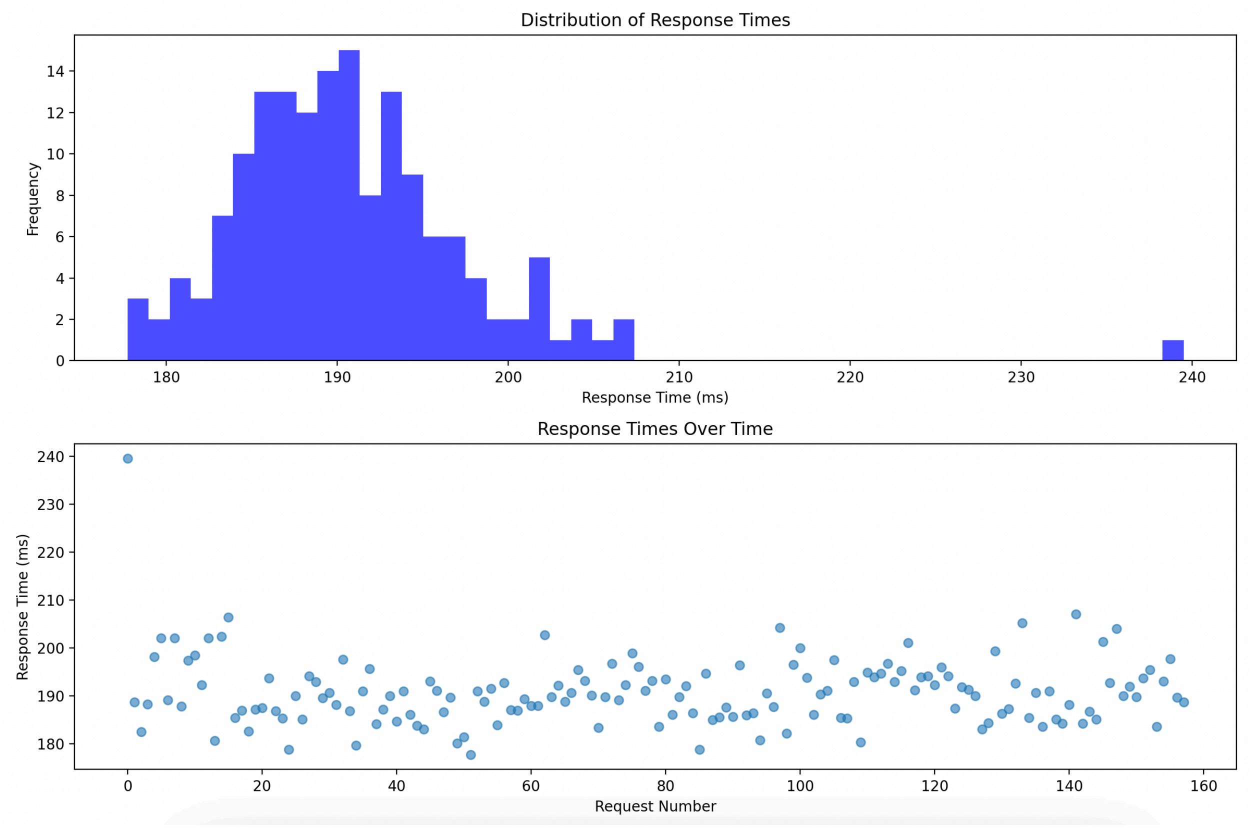
Building Scalable Distributed Systems- Assignment 1b

Load Test Results:



Observations from Load test results:

1. Distribution Shape

The histogram shows a mostly normal distribution with a slight right-hand tail. The majority of requests cluster between ~180 ms and ~200 ms, while only a small fraction exceed 200 ms, with one noticeable outlier around ~240 ms.

Only a very small percentage (<5%) of requests can be classified as "slow," indicating generally stable latency with rare spikes rather than persistent slowness.

2. Consistency

The scatter plot shows that response times remain relatively consistent across the entire 30-second window. There is no increasing trend, which suggests the system does not degrade under sustained load. While there are occasional spikes, they are isolated and non-recurring, indicating transient effects rather than systemic performance issues.

3. Percentiles

The median (50th percentile) response time is around ~188–190 ms, while the 95th percentile is closer to ~200–205 ms. The relatively small gap between the median and 95th percentile suggests low variability.

4. Infrastructure Impact

This service is deployed as a single container on a basic EC2 instance, meaning all requests share the same CPU cores, memory, and network interface.

Contributing factors include:

1. **CPU scheduling and context switching**
    - With limited vCPUs, the container competes with:
        - The host OS
        - The container runtime (Docker)
        - Other background system processes
    - This can introduce small but measurable latency variations between requests.
2. **Single-instance execution bottleneck**

- With only one container, requests cannot be distributed.
- Even sequential requests must wait for the same execution context, limiting throughput and increasing sensitivity to transient delays.

5. Scaling Implications

This test uses sequential requests from a single client, which does not reflect real-world traffic patterns. With 100 concurrent users, response times would likely:

- Increase significantly due to CPU contention
- Exhibit a longer tail in the histogram
- Show higher 95th and 99th percentile latencies

Horizontal scaling (multiple containers behind a load balancer) or vertical scaling (larger instance) would be required to maintain current performance levels.

6. Network vs Processing

The occasional high-latency outliers suggest a combination of network variability and server-side processing delays.

Further investigation could include:

- Logging server-side request processing time
- Monitoring EC2 CPU and memory usage during load
- Comparing results against a local (same-host) client to isolate network effects