Lab 5 Findings - Chelle Davies

Test Setup

- **Tool:** k6 for load testing the API.
- **Variable:** `CACHE_RATE`, which was adjusted to assess its impact on API performance.
- **Metrics Collected:**
  - **Response Time:** How quickly the API responds under load.

- **Throughput:** The number of requests processed per unit time.
- **Error Rate:** The frequency of failed requests.
- **Latency and Other KPIs:** Additional details on resource utilization and performance trends.

Introduction

This document analyzes the API performance based on load tests conducted with k6. The tests were executed under varying `CACHE_RATE` values to observe how caching influences key performance indicators such as response time, throughput, and error rate. The results are grouped by test runs and are illustrated in the screenshots below.

Performance Results

Group 1: Baseline and Detailed Metrics

The first group of screenshots provides an overall view of the API's performance under a given `CACHE_RATE` setting, including both aggregate and detailed breakdowns.

- **Overall Performance Overview:**
  - ```
    (base) michelledavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run --env CACHE_RATE=0 load.js

            /\      Grafana   /‾‾/
       /\  /  \     |\  __   /  /
      /  \/    \    | |/ /  /   ‾‾\
     /          \   |   (  |  (‾)  |
    / __________ \  |_|_\  _____/

         execution: local
            script: load.js
            output: -

         scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
                  * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

    running (2m04.4s), 100/100 VUs, 22583 complete and 0 interrupted iterations
    default   [===========>--------------------------] 100/100 VUs  2m04.4s/6m05.0s
    ```
  - ```
    (base) michelledavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run --env CACHE_RATE=0 load.js

            /\      Grafana   /‾‾/
       /\  /  \     |\  __   /  /
      /  \/    \    | |/ /  /   ‾‾\
     /          \   |   (  |  (‾)  |
    / __________ \  |_|\_\  \_____/

         execution: local
            script: load.js
            output: -

         scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
                  * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

    running (3m27.1s), 100/100 VUs, 34542 complete and 0 interrupted iterations
    default   [====================>-----------------] 100/100 VUs  3m27.1s/6m05.0s
    ```

```
(base) michelledavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run --env CACHE_RATE=0 load.js

        /\      Grafana   /‾‾/
   /\  /  \     |\  __   /  /
  /  \/    \    | |/ /  /   ‾‾\
 /          \   |   (  |  (‾)  |
/ ********** \  |_|\_\  \_____/

     execution: local
        script: load.js
        output: -

     scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
              * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

running (5m43.9s), 100/100 VUs, 54170 complete and 0 interrupted iterations
default   [==================================>---] 100/100 VUs  5m43.9s/6m05.0s
```

Observations:
These images show the initial performance baseline, revealing the average response times and throughput. The detailed breakdown helps identify peak load times and potential bottlenecks before comparing with other `CACHE_RATE` settings.

Group 2: Varying `CACHE_RATE` Values

lab-5-load-testing-michelleddavies / Findings.md ↑ Top

Preview Code Blame Raw

- **Test Run with Lower `CACHE_RATE`:**

```
(base) michelledavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run --env CACHE_RATE=0 load.js

        /\      Grafana   /‾‾/
   /\  /  \     |\  __   /  /
  /  \/    \    | |/ /  /   ‾‾\
 /          \   |   (  |  (‾)  |
/ ********** \  |_|\_\  \_____/

     execution: local
        script: load.js
        output: -

     scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
              * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)


     ✓ prediction responded with 200

     checks.........................: 100.00% 56872 out of 56872
     data_received..................: 13 MB   36 kB/s
     data_sent......................: 18 MB   50 kB/s
     http_req_blocked...............: avg=544.33µs min=0s      med=1µs      max=429.11ms p(90)=1µs      p(95)=1µs
     http_req_connecting............: avg=172.25µs min=0s      med=0s       max=142.03ms p(90)=0s       p(95)=0s
   × http_req_duration..............: avg=584.39ms min=95.03ms med=647.44ms max=1.22s    p(90)=763.51ms p(95)=798.71ms
       { expected_response:true }...: avg=584.39ms min=95.03ms med=647.44ms max=1.22s    p(90)=763.51ms p(95)=798.71ms
     http_req_failed................: 0.00%   0 out of 56872
     http_req_receiving.............: avg=1.55ms   min=6µs     med=359µs    max=404.37ms p(90)=4.76ms   p(95)=6.55ms
     http_req_sending...............: avg=184.1µs  min=10µs    med=81µs     max=15.65ms  p(90)=216µs    p(95)=321.44µs
     http_req_tls_handshaking.......: avg=368.81µs min=0s      med=0s       max=295.06ms p(90)=0s       p(95)=0s
     http_req_waiting...............: avg=582.65ms min=94.98ms med=644.71ms max=1.22s    p(90)=761.5ms  p(95)=797.51ms
     http_reqs......................: 56872   155.684544/s
     iteration_duration.............: avg=585.29ms min=95.1ms  med=647.89ms max=1.22s    p(90)=763.94ms p(95)=799.15ms
     iterations.....................: 56872   155.684544/s
     vus............................: 10      min=2             max=100
     vus_max........................: 100     min=100           max=100


running (6m05.3s), 000/100 VUs, 56872 complete and 0 interrupted iterations
default ✓ [======================================] 000/100 VUs  6m5s
ERRO[0365] thresholds on metrics 'http_req_duration' have been crossed
```

Key Observations:
At a lower `CACHE_RATE` (`CACHE_RATE=0`), the API shows moderate throughput with slightly higher response times. The reduced cache hit rate may be contributing to longer processing times per request.

- **Test Run with Medium `CACHE_RATE`:**

```
(base) michelleddavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run --env CACHE_RATE=0.5 load.js

        /\      Grafana   /‾‾/
     /\  /  \     |\  __   /  /
    /  \/    \    | |/ /  /   ‾‾\
   /          \   |   (  |  (‾)  |
  / __________ \  |_|\_\  \_____/

     execution: local
        script: load.js
        output: -

     scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
              * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

WARN[0000] Error from API server                         error="listen tcp 127.0.0.1:6565: bind: address already in use"

     ✓ prediction responded with 200

     checks.........................: 100.00% 55850 out of 55850
     data_received..................: 13 MB   36 kB/s
     data_sent......................: 18 MB   48 kB/s
     http_req_blocked...............: avg=550.87µs min=0s      med=1µs      max=373.55ms p(90)=1µs      p(95)=1µs
     http_req_connecting............: avg=177.05µs min=0s      med=0s       max=122.94ms p(90)=0s       p(95)=0s
   × http_req_duration..............: avg=594.67ms min=96.09ms med=637.65ms max=1.3s     p(90)=767.51ms p(95)=797.7ms
       { expected_response:true }...: avg=594.67ms min=96.09ms med=637.65ms max=1.3s     p(90)=767.51ms p(95)=797.7ms
     http_req_failed................: 0.00%   0 out of 55850
     http_req_receiving.............: avg=1.02ms   min=6µs     med=112µs    max=413.03ms p(90)=3.21ms   p(95)=5.07ms
     http_req_sending...............: avg=199.95µs min=12µs    med=87µs     max=17.88ms  p(90)=210µs    p(95)=335µs
     http_req_tls_handshaking.......: avg=372.15µs min=0s      med=0s       max=266.45ms p(90)=0s       p(95)=0s
     http_req_waiting...............: avg=593.45ms min=95.73ms med=636.08ms max=1.3s     p(90)=766.25ms p(95)=796.54ms
     http_reqs......................: 55850   152.899346/s
     iteration_duration.............: avg=595.62ms min=96.5ms  med=638.26ms max=1.3s     p(90)=768.18ms p(95)=798.47ms
     iterations.....................: 55850   152.899346/s
     vus............................: 3       min=2              max=100
     vus_max........................: 100     min=100            max=100


running (6m05.3s), 000/100 VUs, 55850 complete and 0 interrupted iterations
default ✓ [======================================] 000/100 VUs  6m5s
ERRO[0365] thresholds on metrics 'http_req_duration' have been crossed
```

Key Observations:
With a medium `CACHE_RATE` (`CACHE_RATE=0.5`), improvements are seen in response times and throughput. The caching mechanism appears to optimize data retrieval, reducing the server load and processing time.

- **Test Run with Higher `CACHE_RATE`:**

```
(base) michelleddavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run --env CACHE_RATE=1 load.js

        /\      Grafana   /‾‾/
     /\  /  \     |\  __   /  /
    /  \/    \    | |/ /  /   ‾‾\
   /          \   |   (  |  (‾)  |
  / __________ \  |_|\_\  \_____/

     execution: local
        script: load.js
        output: -

     scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
              * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

WARN[0000] Error from API server                         error="listen tcp 127.0.0.1:6565: bind: address already in use"

     ✓ prediction responded with 200

     checks.........................: 100.00% 69397 out of 69397
     data_received..................: 16 MB   44 kB/s
     data_sent......................: 22 MB   59 kB/s
     http_req_blocked...............: avg=459.01µs min=0s      med=0s       max=464.01ms p(90)=1µs      p(95)=1µs
     http_req_connecting............: avg=144.41µs min=0s      med=0s       max=160.06ms p(90)=0s       p(95)=0s
   × http_req_duration..............: avg=478.41ms min=94.98ms med=510.01ms max=1.09s    p(90)=663.16ms p(95)=677.96ms
       { expected_response:true }...: avg=478.41ms min=94.98ms med=510.01ms max=1.09s    p(90)=663.16ms p(95)=677.96ms
     http_req_failed................: 0.00%   0 out of 69397
     http_req_receiving.............: avg=1.84ms   min=6µs     med=733µs    max=428.41ms p(90)=5.36ms   p(95)=7.1ms
     http_req_sending...............: avg=165.42µs min=12µs    med=71µs     max=20.34ms  p(90)=181µs    p(95)=291µs
     http_req_tls_handshaking.......: avg=312.97µs min=0s      med=0s       max=360.71ms p(90)=0s       p(95)=0s
     http_req_waiting...............: avg=476.39ms min=94.8ms  med=507.7ms  max=1.09s    p(90)=660.21ms p(95)=674.95ms
     http_reqs......................: 69397   198.128132/s
     iteration_duration.............: avg=479.21ms min=95.14ms med=510.69ms max=1.09s    p(90)=663.69ms p(95)=678.5ms
     iterations.....................: 69397   198.128132/s
     vus............................: 3       min=2              max=100
     vus_max........................: 100     min=100            max=100


running (6m05.0s), 000/100 VUs, 69397 complete and 0 interrupted iterations
default ✓ [======================================] 000/100 VUs  6m5s
ERRO[0365] thresholds on metrics 'http_req_duration' have been crossed
```

Key Observations:
At a higher `CACHE_RATE` (`CACHE_RATE=1`), the API delivers the best performance in terms of throughput and response time. However, this setting may also introduce higher cache overhead or potential stale data issues if not managed properly.

- **Additional Detailed Views:**

```
(base) michelledavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run --env CACHE_RATE=0.25 load.js

     execution: local
        script: load.js
        output: -

     scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
              * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

WARN[0000] Error from API server                         error="listen tcp 127.0.0.1:6565: bind: address already in use"

     ✓ prediction responded with 200

     checks.........................: 100.00% 55041 out of 55041
     data_received..................: 13 MB   35 kB/s
     data_sent......................: 18 MB   48 kB/s
     http_req_blocked...............: avg=556.02µs min=0s      med=1µs      max=412.96ms p(90)=1µs      p(95)=1µs
     http_req_connecting............: avg=177.26µs min=0s      med=0s       max=128.18ms p(90)=0s       p(95)=0s
   × http_req_duration..............: avg=603.62ms min=93.6ms  med=648.74ms max=1.3s     p(90)=766.38ms p(95)=798.74ms
       { expected_response:true }...: avg=603.62ms min=93.6ms  med=648.74ms max=1.3s     p(90)=766.38ms p(95)=798.74ms
     http_req_failed................: 0.00%   0 out of 55041
     http_req_receiving.............: avg=1.34ms   min=5µs     med=181µs    max=87.95ms  p(90)=4.4ms    p(95)=6.18ms
     http_req_sending...............: avg=189.41µs min=11µs    med=82µs     max=18.09ms  p(90)=210µs    p(95)=321µs
     http_req_tls_handshaking.......: avg=376.98µs min=0s      med=0s       max=307.54ms p(90)=0s       p(95)=0s
     http_req_waiting...............: avg=602.09ms min=92.22ms med=646.46ms max=1.3s     p(90)=764.79ms p(95)=797.35ms
     http_reqs......................: 55041   150.702402/s
     iteration_duration.............: avg=604.55ms min=94.75ms med=649.18ms max=1.3s     p(90)=766.78ms p(95)=799.25ms
     iterations.....................: 55041   150.702402/s
     vus............................: 6       min=2              max=100
     vus_max........................: 100     min=100            max=100


running (6m05.2s), 000/100 VUs, 55041 complete and 0 interrupted iterations
default ✓ [======================================] 000/100 VUs  6m5s
ERRO[0365] thresholds on metrics 'http_req_duration' have been crossed
```

```
(base) michelledavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run --env CACHE_RATE=0.75 load.js

     execution: local
        script: load.js
        output: -

     scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
              * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

WARN[0000] Error from API server                         error="listen tcp 127.0.0.1:6565: bind: address already in use"

     ✓ prediction responded with 200

     checks.........................: 100.00% 60358 out of 60358
     data_received..................: 14 MB   38 kB/s
     data_sent......................: 19 MB   52 kB/s
     http_req_blocked...............: avg=512.24µs min=0s      med=1µs      max=447.07ms p(90)=1µs      p(95)=1µs
     http_req_connecting............: avg=167.34µs min=0s      med=0s       max=173.73ms p(90)=0s       p(95)=0s
   × http_req_duration..............: avg=550.13ms min=97.54ms med=568.33ms max=1.3s     p(90)=751.77ms p(95)=784.31ms
       { expected_response:true }...: avg=550.13ms min=97.54ms med=568.33ms max=1.3s     p(90)=751.77ms p(95)=784.31ms
     http_req_failed................: 0.00%   0 out of 60358
     http_req_receiving.............: avg=898.23µs min=6µs     med=107µs    max=402.7ms  p(90)=2.93ms   p(95)=4.51ms
     http_req_sending...............: avg=193.41µs min=12µs    med=85µs     max=16.76ms  p(90)=208µs    p(95)=331µs
     http_req_tls_handshaking.......: avg=343.33µs min=0s      med=0s       max=272.92ms p(90)=0s       p(95)=0s
     http_req_waiting...............: avg=549.04ms min=96.8ms  med=567.04ms max=1.3s     p(90)=750.62ms p(95)=782.96ms
     http_reqs......................: 60358   165.349879/s
     iteration_duration.............: avg=551.03ms min=98.05ms med=569.04ms max=1.3s     p(90)=752.5ms  p(95)=785.04ms
     iterations.....................: 60358   165.349879/s
     vus............................: 3       min=2              max=100
     vus_max........................: 100     min=100            max=100


running (6m05.0s), 000/100 VUs, 60358 complete and 0 interrupted iterations
default ✓ [======================================] 000/100 VUs  6m5s
ERRO[0365] thresholds on metrics 'http_req_duration' have been crossed
```

```
(base) michelledavies@MacBook-Pro lab-5-load-testing-michelleddavies % k6 run load.js

     execution: local
        script: load.js
        output: -

     scenarios: (100.00%) 1 scenario, 100 max VUs, 6m35s max duration (incl. graceful stop):
              * default: Up to 100 looping VUs for 6m5s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

WARN[0000] Error from API server                         error="listen tcp 127.0.0.1:6565: bind: address already in use"

     ✓ prediction responded with 200

     checks.........................: 100.00% 100973 out of 100973
     data_received..................: 23 MB   63 kB/s
     data_sent......................: 32 MB   87 kB/s
     http_req_blocked...............: avg=312.47µs min=0s      med=1µs      max=432.62ms p(90)=1µs      p(95)=2µs
     http_req_connecting............: avg=99.51µs  min=0s      med=0s       max=173.15ms p(90)=0s       p(95)=0s
   × http_req_duration..............: avg=328.66ms min=91.91ms med=176.71ms max=1.12s    p(90)=667.66ms p(95)=720.27ms
       { expected_response:true }...: avg=328.66ms min=91.91ms med=176.71ms max=1.12s    p(90)=667.66ms p(95)=720.27ms
     http_req_failed................: 0.00%   0 out of 100973
     http_req_receiving.............: avg=782.76µs min=6µs     med=84µs     max=421.24ms p(90)=2.4ms    p(95)=4.14ms
     http_req_sending...............: avg=171.12µs min=11µs    med=97µs     max=15.76ms  p(90)=222µs    p(95)=295µs
     http_req_tls_handshaking.......: avg=210.72µs min=0s      med=0s       max=306.81ms p(90)=0s       p(95)=0s
     http_req_waiting...............: avg=327.71ms min=91.76ms med=175.97ms max=1.12s    p(90)=665.35ms p(95)=718.87ms
     http_reqs......................: 100973  276.633578/s
     iteration_duration.............: avg=329.33ms min=92.19ms med=177.02ms max=1.41s    p(90)=668.46ms p(95)=722.18ms
     iterations.....................: 100973  276.633578/s
     vus............................: 3       min=2                max=100
     vus_max........................: 100     min=100              max=100


running (6m05.0s), 000/100 VUs, 100973 complete and 0 interrupted iterations
default ✓ [======================================] 000/100 VUs  6m5s
ERRO[0365] thresholds on metrics 'http_req_duration' have been crossed
```

- This is showing the default value `CACHE_RATE=0.9` .

Key Observations:
These screenshots offer further insights into latency distributions, error occurrences, and resource utilization. They confirm that as `CACHE_RATE` increases, the API benefits from improved data retrieval times, but it is important to balance cache freshness and performance.

Synthesis of Findings

- **Performance Trade-offs:**
  Increasing the `CACHE_RATE` generally enhances API throughput and reduces response times, as the system benefits from caching frequently requested data. However, there is a balance to be maintained—excessively high caching might risk serving outdated information or increase overhead if the cache management is not optimized.

- **Metric Correlations:**
  The detailed metrics indicate a strong correlation between `CACHE_RATE` and key KPIs:

  - **Lower `CACHE_RATE`:** Results in higher response times and lower throughput.
  - **Medium `CACHE_RATE`:** Provides a balanced performance with moderate improvements.
  - **Higher `CACHE_RATE`:** Offers the best performance in terms of speed and load handling, with the caveat of potential data staleness.

Conclusion

The k6 load tests clearly demonstrate that the `CACHE_RATE` value is a significant factor in API performance. By fine-tuning this parameter, teams can achieve optimal performance—balancing fast response times with efficient throughput while minimizing errors. The comprehensive visual data from the screenshots underscores the importance of ongoing performance monitoring and the need for tailored caching strategies to meet specific application requirements.