nova
net

Caching og K6

Hans Arne Vartdal
Lars Alexander Jakobsen

nova
net

Utvikler på **platform**teamet med fokus på **API.**

Brukes av Smartbokser, Smart TV, AppleTV,

AndroidTV, iOS, Android, og web.

RiksTV

nova
net

Noen tall fra siste 7 dager:

94,5 millioner requests

~800 requests per second (peak)

Snitt responstid **38,9ms**

1,8GB komprimert JSON

RiksTV

nova
net

Lazy loading

Distribuert

Events

Cache

Write through

InMemory

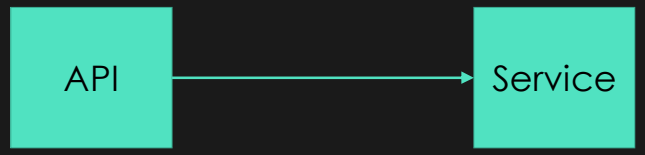
«Cache aside»

nova
net

Ingen cache

Trenger du egentlig cache?

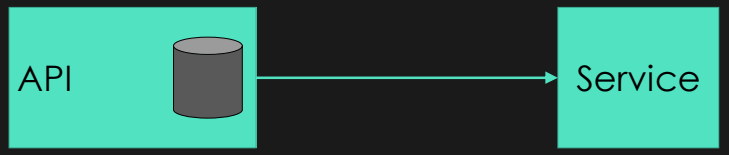
Systemer uten cache er **lette å forstå**, og
lette å teste.



nova
net

ConcurrentDictionary

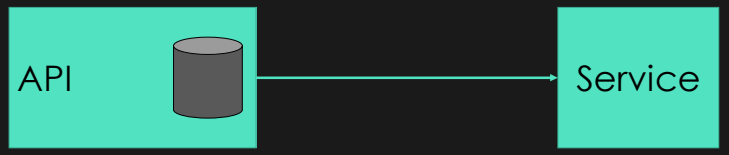
- Lett å implementere, og lett å teste
- Lite "out of the box"
- Må implementere strategier for oppdatering/invalidering
- Relativt små datasett med lav endringstakt



nova
net

MemoryCache

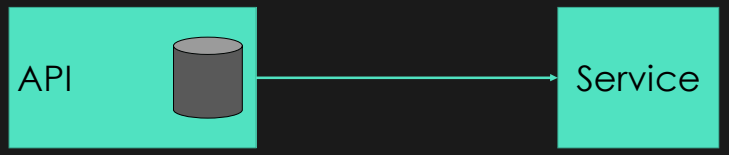
- Bruk **Microsoft.Extensions.Caching.Memory**
- Godt egnet for «cache aside» pga støtte for **Time To Live** (TTL)
- Lar deg konfigurere **minneforbruket**
- Lar deg håndtere når data kastes ut (cache eviction)



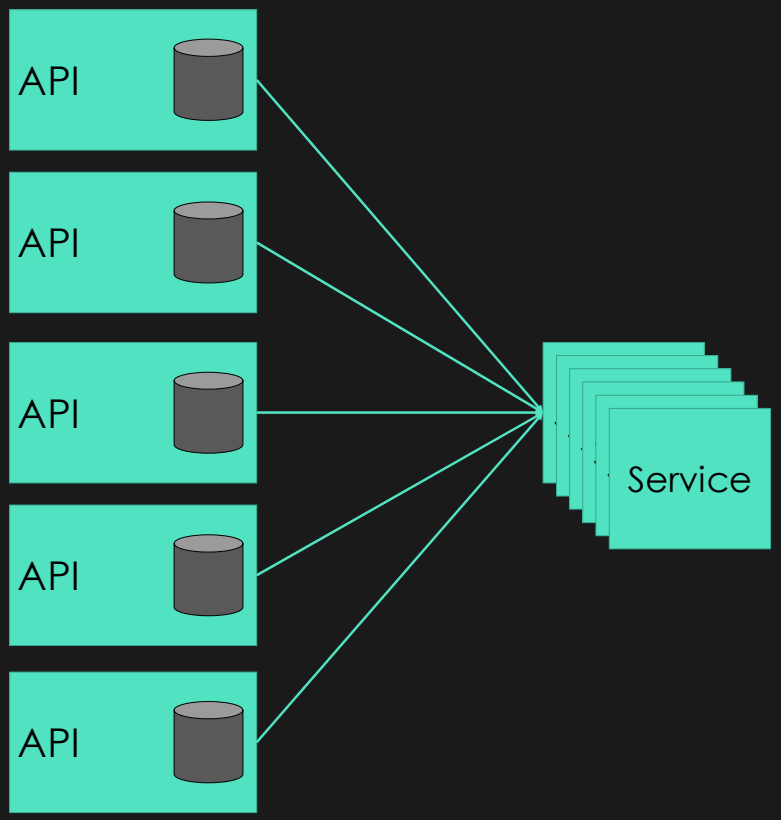
nova
net

LazyCache

- Open source bibliotek som **utvider MemoryCache**
- Garanterer at oppdatering av cache bare kjøres en gang
- **Unngå flere samtidige kall** mot en treg tjeneste



nova
net



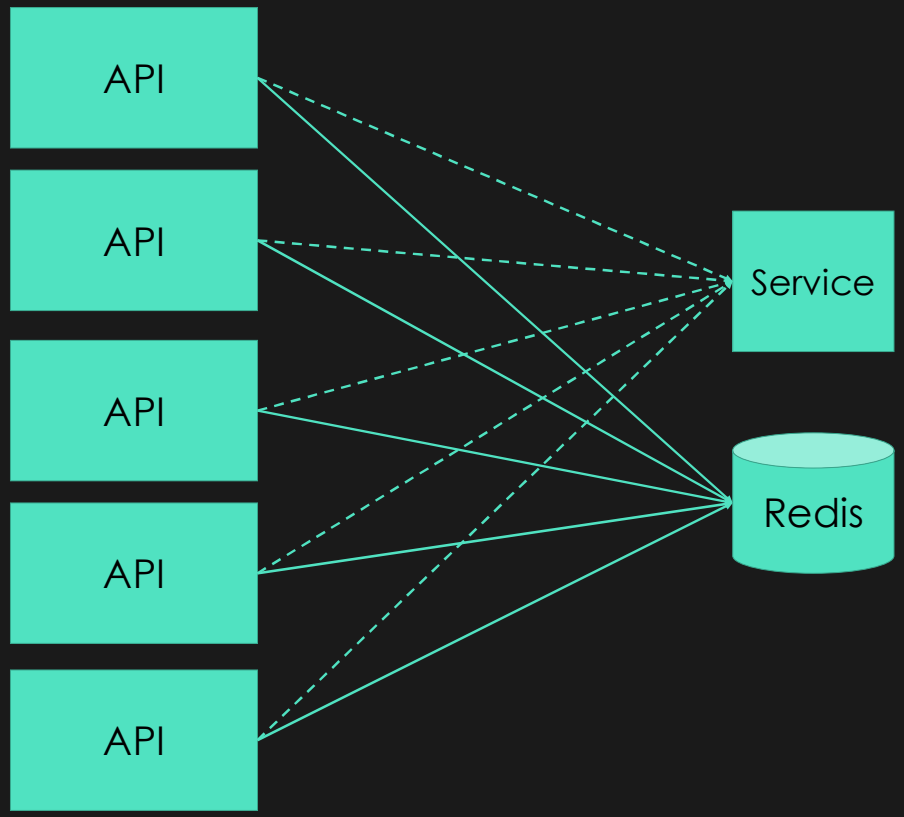
nova
net

Distribuert cache

- Distribuert cache; Ikke i samme prosess som applikasjonen
- For eksempel **Redis**
- Det finnes flere strategier for å oppdatere en slik cache!

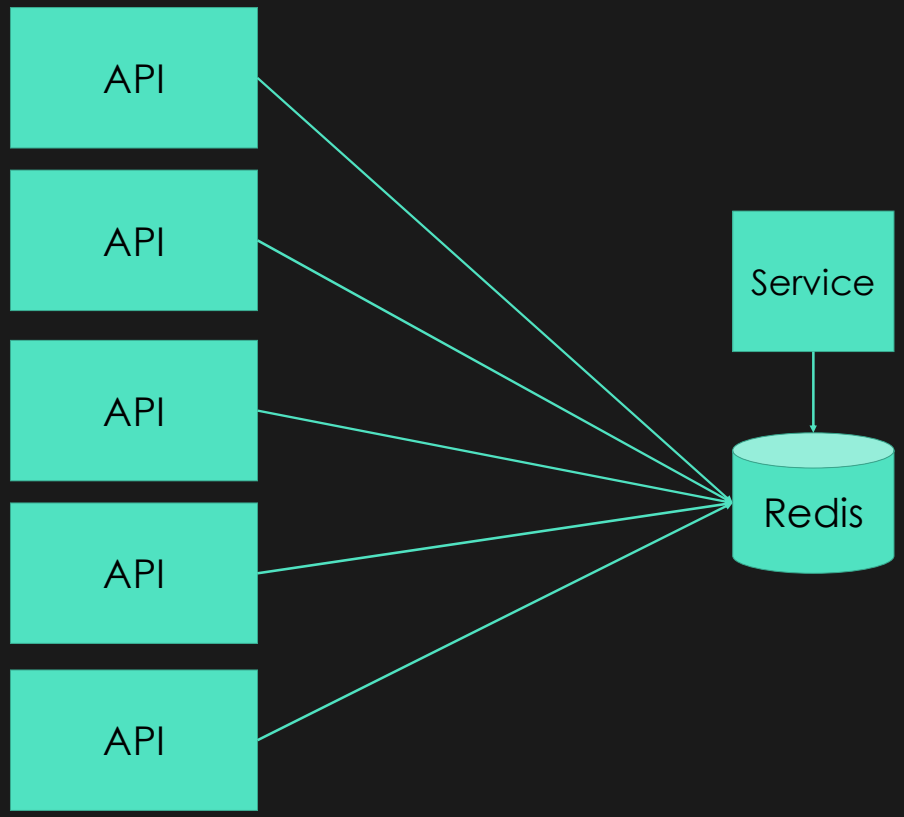
nova
net

Lazy loading



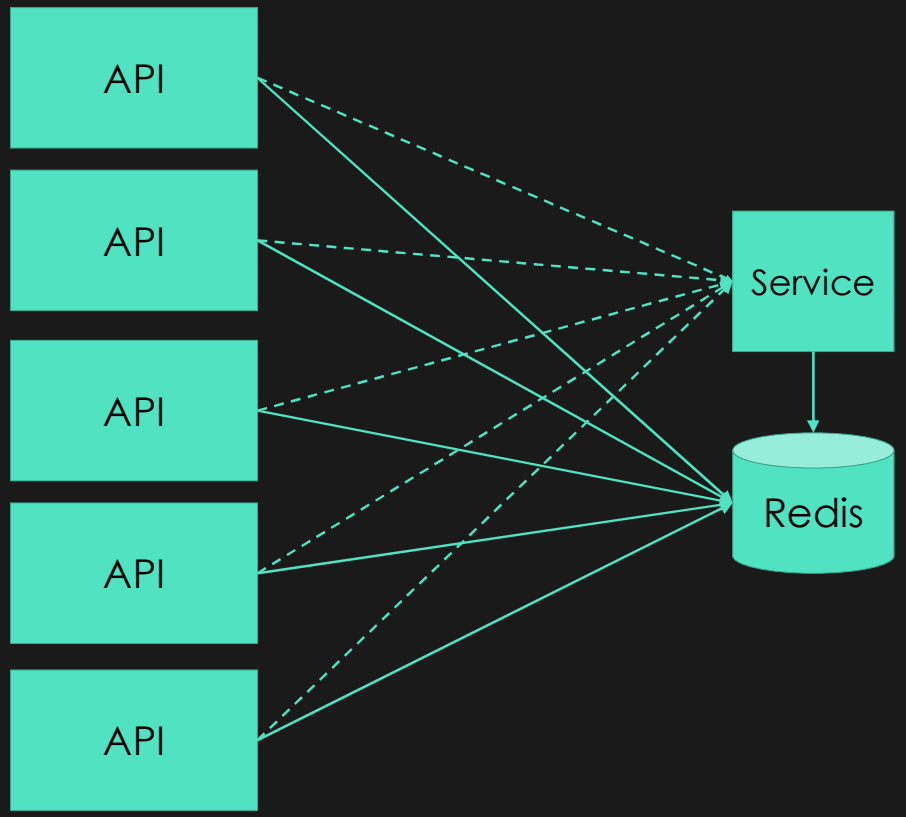
nova
net

Write through



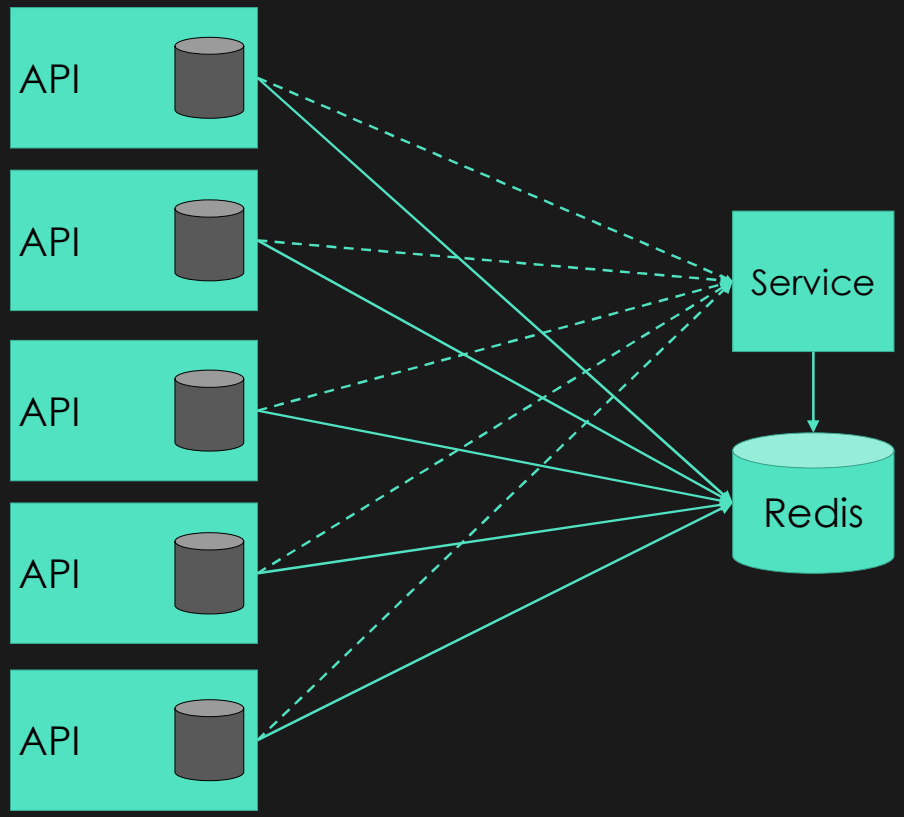
nova
net

Lazy loading + Write through



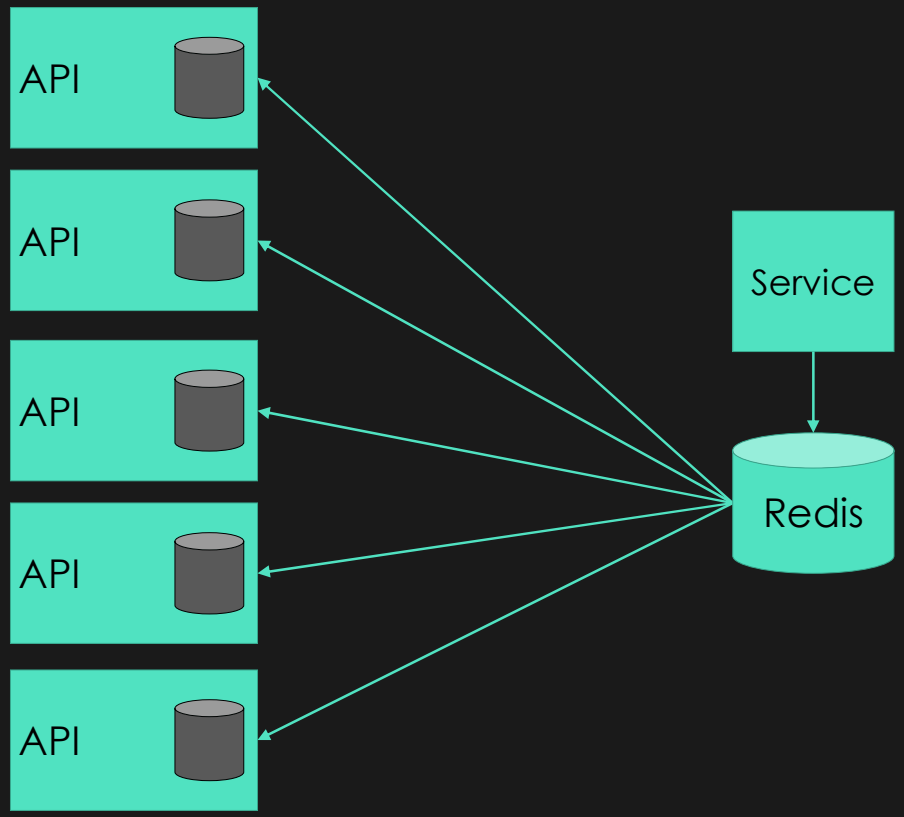
nova
net

In memory
Lazy cache
+
Redis
Lazy loading
+
Redis
Write through



nova
net

Event driven cache



nova
net

Ting å tenke på

- Må **første request** være like rask?
- **Hvor mye** data har jeg?
- **Hvor ofte** må jeg oppdatere dataene?
- Kaldstart
- **Versjonering** av distribuert cache?
- Distribuert cache kan bli «single point of failure»
- Tenk på **latency** ved distribuert cache.
- **Immutability** for InMemory cache

nova
net

K6

- Verktøy for ytelsestesting
  - Tester i Javascript
  - Command Line verktøy

Linux:
sudo apt-get install k6

Mac:
brew install k6

Windows: https://dl.bintray.com/loadimpact/windows/k6-v0.26.1-amd64.msi

Docker:
docker pull loadimpact/k6

nova
net

K6

- Open source, gratis og laget for utviklere
- Lett å komme i gang
  - Kan starte med ytelsestesting fra utviklermaskin
- Kan integreres i CI/CD

nova
net

DEMO

nova net

Enkel test

```
import http from 'k6/http';
import { sleep } from 'k6';

export default function() {

  let result = http.get('https://novanetk6api.azurewebsites.net/nocache/99');

  sleep(1); //sekunder

}
```

nova net

Enkel test: Resultat

```
execution: local-
   output: -
   script: ./scripts_01.js

  duration: -,  iterations: 1
       vus: 1, max: 1

  done [==========================================================] 1 / 1i

  data_received..............: 6.7 kB 2.8 kB/s
  data_sent..................: 565 B  240 B/s
  http_req_blocked...........: avg=292.92ms min=292.92ms med=292.92ms max=292.92ms p(90)=292.92ms p(95)=292.92ms
  http_req_connecting........: avg=38.99ms  min=38.99ms  med=38.99ms  max=38.99ms  p(90)=38.99ms  p(95)=38.99ms
  http_req_duration..........: avg=1.05s    min=1.05s    med=1.05s    max=1.05s    p(90)=1.05s    p(95)=1.05s
  http_req_receiving.........: avg=0s       min=0s       med=0s       max=0s       p(90)=0s       p(95)=0s
  http_req_sending...........: avg=0s       min=0s       med=0s       max=0s       p(90)=0s       p(95)=0s
  http_req_tls_handshaking...: avg=133.96ms min=133.96ms med=133.96ms max=133.96ms p(90)=133.96ms p(95)=133.96ms
  http_req_waiting...........: avg=1.05s    min=1.05s    med=1.05s    max=1.05s    p(90)=1.05s    p(95)=1.05s
  http_reqs..................: 1      0.425313/s
  iteration_duration.........: avg=2.35s    min=2.35s    med=2.35s    max=2.35s    p(90)=2.35s    p(95)=2.35s
  iterations.................: 1      0.425313/s
  vus........................: 1      min=1 max=1
  vus_max....................: 1      min=1 max=1

PS D:\dev\novanet\k6-caching\Novanet.K6.Api\tests>
```

nova net

Options

```
import http from 'k6/http';
import { sleep } from 'k6';

export let options = {
  vus: 5,
  duration: "30s"
};

export default function() {

  let result = http.get('https://novanetk6api.azurewebsites.net/nocache/99');

  sleep(1); //sekunder

}
```

nova net

Checks

```
import http from 'k6/http';
import { sleep, check } from 'k6';

export let options = {
  vus: 5,
  duration: "30s"
};

export default function() {

  let result = http.get('https://novanetk6api.azurewebsites.net/nocache/99');

  check(result, {
    "Status is 200": (r) => r.status == 200,
    "Duration < 1000ms": (r) => r.timings.duration < 1000
  });

  sleep(1); //sekunder
}
```

nova net

Checks : Resultat

```
✓ Status is 200
✗ Duration < 1000ms
  ↳  0% — ✓ 0 / ✗ 70

checks.....................: 50.00% ✓ 70  ✗ 70
data_received..............: 70 kB  2.3 kB/s
data_sent..................: 12 kB  386 B/s
http_req_blocked...........: avg=21.45ms min=0s    med=0s    max=305.23ms p(90)=0s       p(95)=299.22ms
http_req_connecting........: avg=2.95ms  min=0s    med=0s    max=43ms     p(90)=0s       p(95)=40.99ms
http_req_duration..........: avg=1.06s   min=1.04s med=1.04s max=2.04s    p(90)=1.06s    p(95)=1.06s
http_req_receiving.........: avg=98.97µs min=0s    med=0s    max=998.5µs  p(90)=777.93µs p(95)=838.47µs
http_req_sending...........: avg=0s      min=0s    med=0s    max=0s       p(90)=0s       p(95)=0s
http_req_tls_handshaking...: avg=10.19ms min=0s    med=0s    max=148ms    p(90)=0s       p(95)=141.09ms
http_req_waiting...........: avg=1.06s   min=1.04s med=1.04s max=2.04s    p(90)=1.06s    p(95)=1.06s
http_reqs..................: 70     2.333333/s
iteration_duration.........: avg=2.08s   min=2.04s med=2.04s max=3.34s    p(90)=2.06s    p(95)=2.36s
iterations.................: 70     2.333333/s
vus........................: 5      min=5 max=5
vus_max....................: 5      min=5 max=5
```

nova net

Tresholds

```
import http from 'k6/http';
import { sleep, check } from 'k6';

export let options = {
  vus: 5,
  duration: "30s",
  thresholds: {
    "errors": ["rate<0.1"], // <10% errors
  }
};

export default function() {

  let result = http.get('https://novanetk6api.azurewebsites.net/nocache/99');

  check(result, {
    "Status is 200": (r) => r.status == 200,
    "Duration < 2000ms": (r) => r.timings.duration < 2000
  });

  sleep(1); //sekunder
}
```

nova net

Tresholds : Resultat

```
WARN[0003] No data generated, because no script iterations finished, consider making the test duration longer

    ✓ Status is 200
    ✗ Duration < 2000ms
     ↳  0% — ✓ 0 / ✗ 4

    checks.....................: 50.00% ✓ 4    ✗ 4
    http_req_blocked...........: avg=310.94ms min=310.94ms med=310.94ms max=310.94ms p(90)=310.94ms p(95)=310.94ms
    http_req_connecting........: avg=39.95ms  min=39.95ms  med=39.95ms  max=39.95ms  p(90)=39.95ms  p(95)=39.95ms
  ✗ http_req_duration..........: avg=1.05s    min=1.05s    med=1.05s    max=1.05s    p(90)=1.05s    p(95)=1.05s
    http_req_receiving.........: avg=0s       min=0s       med=0s       max=0s       p(90)=0s       p(95)=0s
    http_req_sending...........: avg=0s       min=0s       med=0s       max=0s       p(90)=0s       p(95)=0s
    http_req_tls_handshaking...: avg=148.07ms min=148.07ms med=148.07ms max=148.07ms p(90)=148.07ms p(95)=148.07ms
    http_req_waiting...........: avg=1.05s    min=1.05s    med=1.05s    max=1.05s    p(90)=1.05s    p(95)=1.05s
    http_reqs..................: 4      2.019838/s
    vus........................: 5      min=5 max=5
    vus_max....................: 5      min=5 max=5

ERRO[0003] some thresholds have failed
```

nova net

Stages

```
import http from 'k6/http';
import { sleep, check } from 'k6';

export let options = {
  thresholds: {
    "errors": ["rate<0.1"], // <10% errors
  },
  stages: [
    { duration: '10s', target: 10 },
    { duration: '15s', target: 20 },
    { duration: '20s', target: 30 },
    { duration: '30s', target: 100 },
  ],
};

export default function() {

  let result = http.get('https://novanetk6api.azurewebsites.net/nocache/99');

  check(result, {
    "Status is 200": (r) => r.status == 200,
    "Duration < 2000ms": (r) => r.timings.duration < 2000
  });

  sleep(1); //sekunder
}
```

nova net

Sammenligning: Resultat

```
Dictionary Cache duration...: avg=43.3176     min=39.9998   med=42.2671   max=51.2598   p(90)=47.1982    p(95)=49.10078
Lazy Cache duration.........: avg=43.752602   min=40.0301   med=42.9399   max=51.0018   p(90)=48.50005   p(95)=49.69604
Memory Cache duration.......: avg=44.792578   min=40.8522   med=43.0006   max=71.3846   p(90)=49.0873    p(95)=50.86346
No Cache duration...........: avg=1082.556407 min=1042.4406 med=1054.5934 max=1997.289 p(90)=1060.61254 p(95)=1065.64368
Redis Cache duration........: avg=91.621389   min=76.1802   med=80.01685  max=698.2008  p(90)=85.53435   p(95)=87.9904
checks.....................: 100.00% ✓ 574 ✗ 0
data_received..............: 149 kB  5.0 kB/s
data_sent..................: 58 kB   1.9 kB/s
http_req_blocked...........: avg=5.22ms      min=0s        med=0s        max=304.77ms  p(90)=0s         p(95)=0s
http_req_connecting........: avg=747.87µs    min=0s        med=0s        max=46.12ms   p(90)=0s         p(95)=0s
http_req_duration..........: avg=266.76ms    min=39.99ms   med=47ms      max=1.99s     p(90)=1.05s      p(95)=1.05s
http_req_receiving.........: avg=134.09µs    min=0s        med=0s        max=7ms       p(90)=931.1µs    p(95)=999.67µs
http_req_sending...........: avg=0s          min=0s        med=0s        max=0s        p(90)=0s         p(95)=0s
http_req_tls_handshaking....: avg=2.42ms      min=0s        med=0s        max=145.01ms  p(90)=0s         p(95)=0s
http_req_waiting...........: avg=266.63ms    min=39.93ms   med=46.99ms   max=1.99s     p(90)=1.05s      p(95)=1.05s
http_reqs..................: 287     9.566361/s
iteration_duration.........: avg=2.58s       min=2.5s      med=2.52s     max=3.47s     p(90)=2.81s      p(95)=2.83s
iterations.................: 54      1.799943/s
vus........................: 5       min=5 max=5
vus_max....................: 5       min=5 max=5
```

nova net

Azure Devops Pipeline

Novanet.K6.Api / azure-pipelines-load-tests.yml

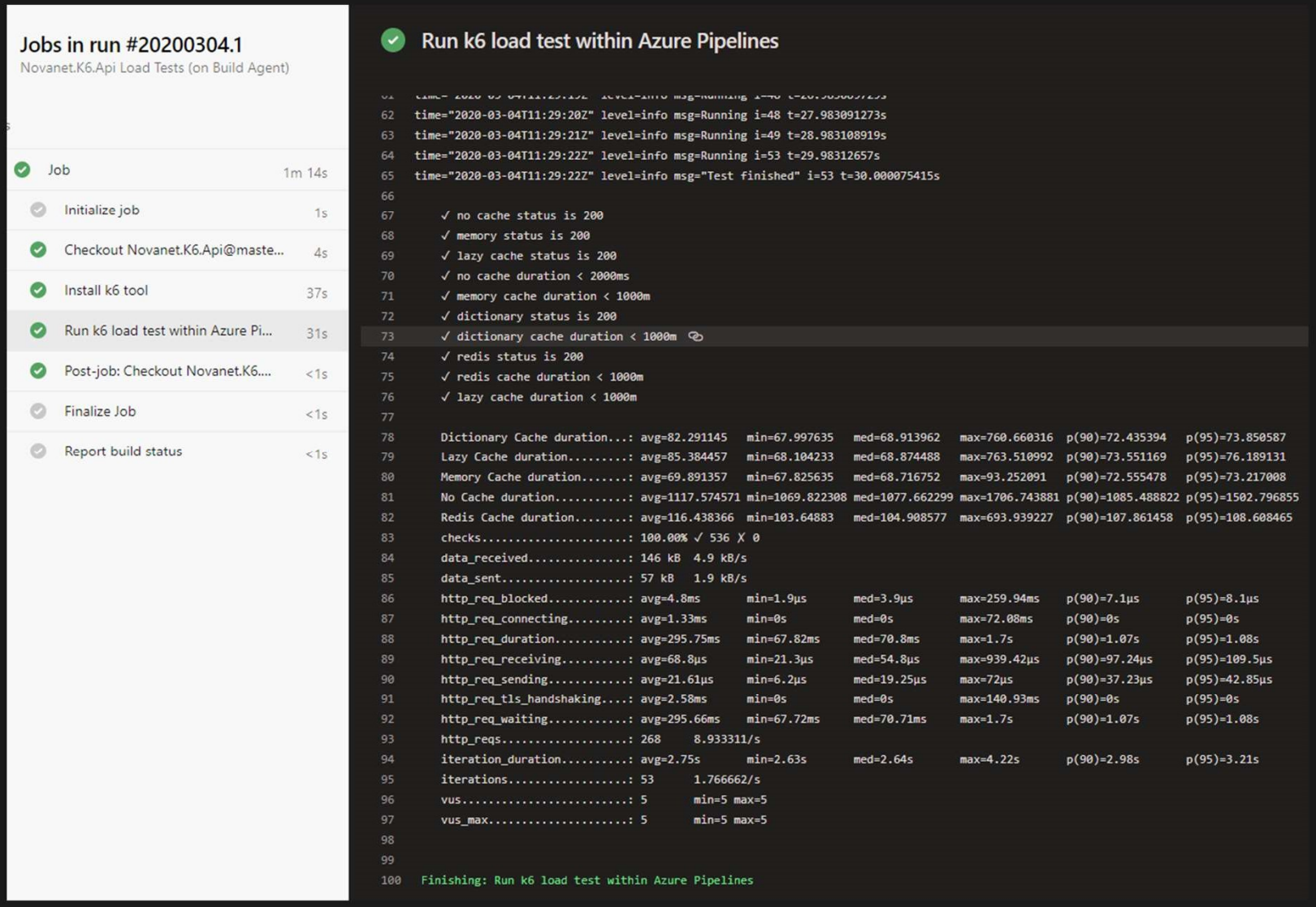
```yaml
# trigger:
# - master

pool:
  vmImage: 'ubuntu-latest'

steps:
- script: |
    sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 379CE192D401AB61
    echo "deb https://dl.bintray.com/loadimpact/deb stable main" | sudo tee -a /etc/apt/sources.list
    sudo apt-get update
    sudo apt-get install k6
  displayName: Install k6 tool

- script: |
    k6 run tests/scripts_99.js
  displayName: Run k6 load test within Azure Pipelines
```

nova net


Jobs in run #20200304.1
Novanet.K6.Api Load Tests (on Build Agent)
Job 1m 14s
Initialize job 1s
Checkout Novanet.K6.Api@maste... 4s
Install k6 tool 37s
Run k6 load test within Azure Pi... 31s
Post-job: Checkout Novanet.K6.... <1s
Finalize Job <1s
Report build status <1s
Run k6 load test within Azure Pipelines
time="2020-03-04T11:29:20Z" level=info msg=Running i=48 t=27.983091273s
time="2020-03-04T11:29:21Z" level=info msg=Running i=49 t=28.983108919s
time="2020-03-04T11:29:22Z" level=info msg=Running i=53 t=29.98312657s
time="2020-03-04T11:29:22Z" level=info msg="Test finished" i=53 t=30.000075415s
✓ no cache status is 200
✓ memory status is 200
✓ lazy cache status is 200
✓ no cache duration < 2000ms
✓ memory cache duration < 1000m
✓ dictionary status is 200
✓ dictionary cache duration < 1000m
✓ redis status is 200
✓ redis cache duration < 1000m
✓ lazy cache duration < 1000m
Dictionary Cache duration...: avg=82.291145 min=67.997635 med=68.913962 max=760.660316 p(90)=72.435394 p(95)=73.850587
Lazy Cache duration.....: avg=85.384457 min=68.104233 med=68.874488 max=763.510992 p(90)=73.551169 p(95)=76.189131
Memory Cache duration.....: avg=69.891357 min=67.825635 med=68.716752 max=93.252091 p(90)=72.555478 p(95)=73.217008
No Cache duration.....: avg=1117.574571 min=1069.822308 med=1077.662299 max=1706.743881 p(90)=1085.488822 p(95)=1502.796855
Redis Cache duration.....: avg=116.438366 min=103.64883 med=104.908577 max=693.939227 p(90)=107.861458 p(95)=108.608465
checks.....: 100.00% ✓ 536 ✗ 0
data_received.....: 146 kB 4.9 kB/s
data_sent.....: 57 kB 1.9 kB/s
http_req_blocked.....: avg=4.8ms min=1.9µs med=3.9µs max=259.94ms p(90)=7.1µs p(95)=8.1µs
http_req_connecting.....: avg=1.33ms min=0s med=0s max=72.08ms p(90)=0s p(95)=0s
http_req_duration.....: avg=295.75ms min=67.82ms med=70.8ms max=1.7s p(90)=1.07s p(95)=1.08s
http_req_receiving.....: avg=68.8µs min=21.3µs med=54.8µs max=939.42µs p(90)=97.24µs p(95)=109.5µs
http_req_sending.....: avg=21.61µs min=6.2µs med=19.25µs max=72µs p(90)=37.23µs p(95)=42.85µs
http_req_tls_handshaking....: avg=2.58ms min=0s med=0s max=140.93ms p(90)=0s p(95)=0s
http_req_waiting.....: avg=295.66ms min=67.72ms med=70.71ms max=1.7s p(90)=1.07s p(95)=1.08s
http_reqs.....: 268 8.933311/s
iteration_duration.....: avg=2.75s min=2.63s med=2.64s max=4.22s p(90)=2.98s p(95)=3.21s
iterations.....: 53 1.766662/s
vus.....: 5 min=5 max=5
vus_max.....: 5 min=5 max=5
Finishing: Run k6 load test within Azure Pipelines

nova
net

Visualisering

- K6 Cloud (i skyen)
  - SaaS, betalt løsning
  - Testene kjøres i K6 sin sky
  - Får automatisk visualisering og historikk

- InfluxDB og Grafana (on prem eller sky)
  - InfluxDB is the open source time series database
  - Grafana is the open source analytics & monitoring solution for every database.

```
k6 login cloud
k6 cloud loadtest.js
```

```
docker pull influxdb
docker run -p 8086:8086 -v c:temp/influxdb/var/lib/influxdb influxdb
k6 run –out influxdb:http://localhost:8086/ .\loadtest.js
```

```
docker pull grafana/Grafana
docker run -d --name=grafana -p 3000:3000 grafana/grafana
```

nova
net

Ting å tenke på:

- Når bør ytelsestestene kjøre?
  - Manuelt
  - Ved hver release
  - Nattlig
- Hvordan stoppe/rulle tilbake en release når testene feiler?
  - Alert + rulle tilbake manuelt
  - Deployment slots
- Vær snill, ingen DDoS'ing!