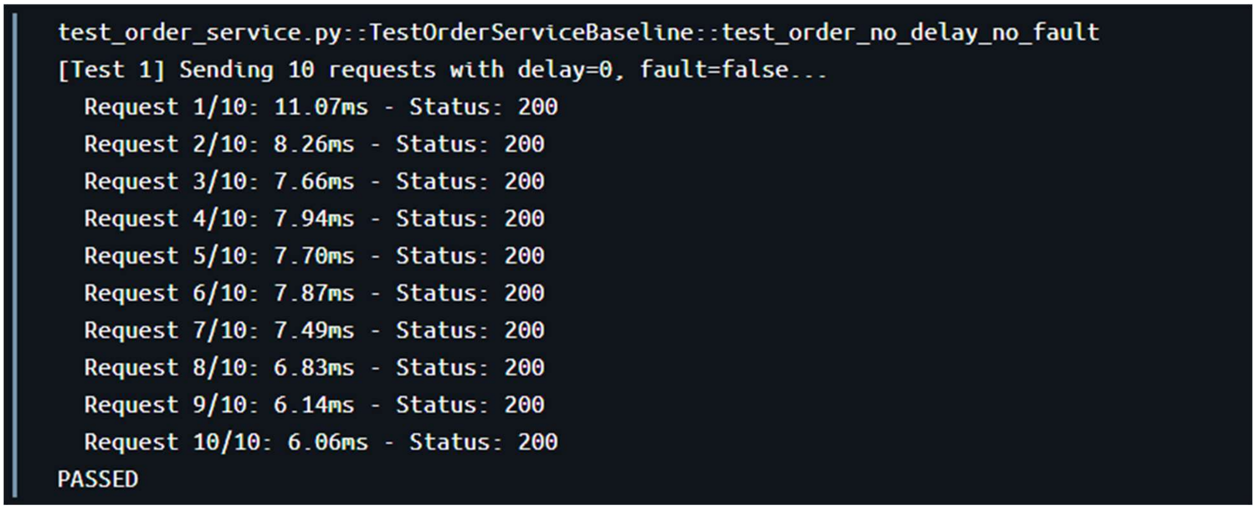
Part A: Synchronous REST

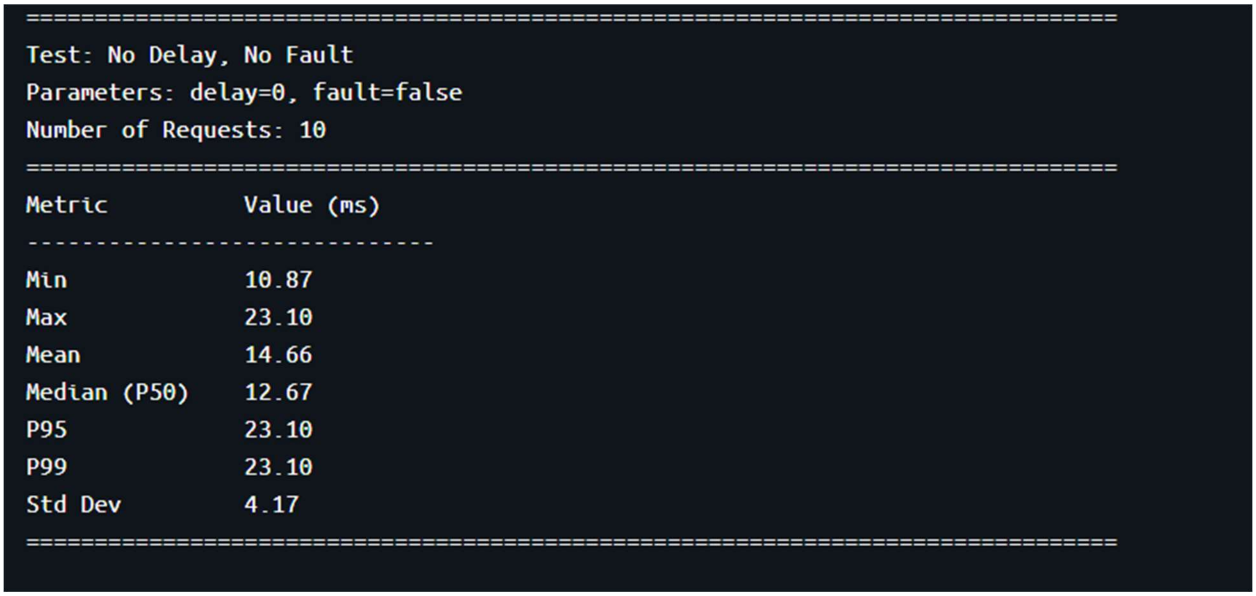
Baseline Latency Test

Test 1 no delays

Average latency ~7ms

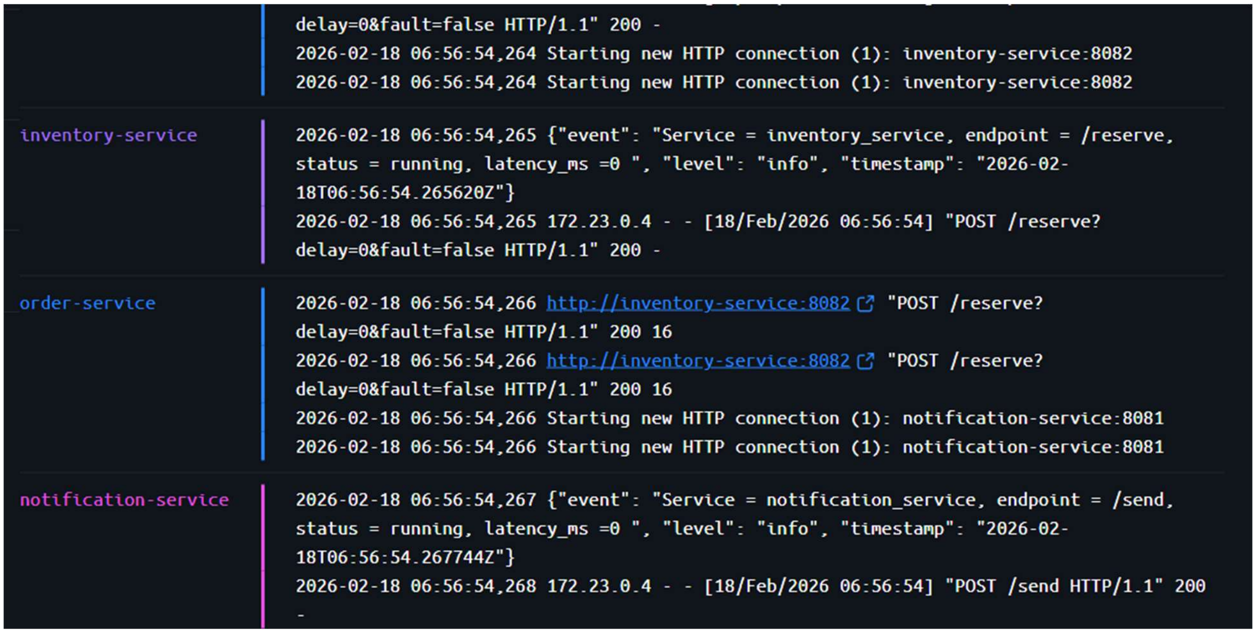


```
test_order_service.py::TestOrderServiceBaseline::test_order_no_delay_no_fault
[Test 1] Sending 10 requests with delay=0, fault=false...
  Request 1/10: 11.07ms - Status: 200
  Request 2/10: 8.26ms - Status: 200
  Request 3/10: 7.66ms - Status: 200
  Request 4/10: 7.94ms - Status: 200
  Request 5/10: 7.70ms - Status: 200
  Request 6/10: 7.87ms - Status: 200
  Request 7/10: 7.49ms - Status: 200
  Request 8/10: 6.83ms - Status: 200
  Request 9/10: 6.14ms - Status: 200
  Request 10/10: 6.06ms - Status: 200
PASSED
```



```
================================================================================
Test: No Delay, No Fault
Parameters: delay=0, fault=false
Number of Requests: 10
================================================================================
Metric          Value (ms)
------------------------------
Min             10.87
Max             23.10
Mean            14.66
Median (P50)    12.67
P95             23.10
P99             23.10
Std Dev         4.17
================================================================================
```

Logs:



```
                     delay=0&fault=false HTTP/1.1" 200 -
                     2026-02-18 06:56:54,264 Starting new HTTP connection (1): inventory-service:8082
                     2026-02-18 06:56:54,264 Starting new HTTP connection (1): inventory-service:8082

inventory-service    2026-02-18 06:56:54,265 {"event": "Service = inventory_service, endpoint = /reserve,
                     status = running, latency_ms =0 ", "level": "info", "timestamp": "2026-02-
                     18T06:56:54.265620Z"}
                     2026-02-18 06:56:54,265 172.23.0.4 - - [18/Feb/2026 06:56:54] "POST /reserve?
                     delay=0&fault=false HTTP/1.1" 200 -

order-service        2026-02-18 06:56:54,266 http://inventory-service:8082 "POST /reserve?
                     delay=0&fault=false HTTP/1.1" 200 16
                     2026-02-18 06:56:54,266 http://inventory-service:8082 "POST /reserve?
                     delay=0&fault=false HTTP/1.1" 200 16
                     2026-02-18 06:56:54,266 Starting new HTTP connection (1): notification-service:8081
                     2026-02-18 06:56:54,266 Starting new HTTP connection (1): notification-service:8081

notification-service 2026-02-18 06:56:54,267 {"event": "Service = notification_service, endpoint = /send,
                     status = running, latency_ms =0 ", "level": "info", "timestamp": "2026-02-
                     18T06:56:54.267744Z"}
                     2026-02-18 06:56:54,268 172.23.0.4 - - [18/Feb/2026 06:56:54] "POST /send HTTP/1.1" 200
                     -
```

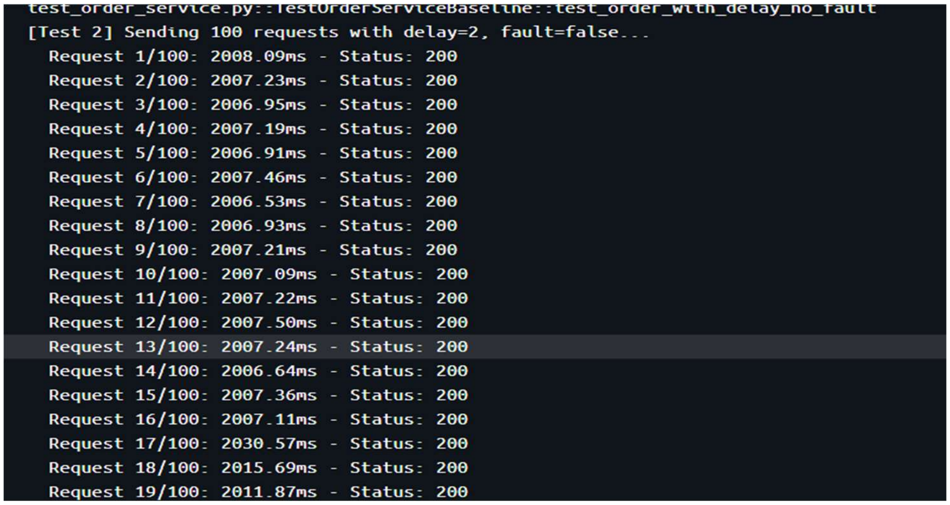
Reasoning:

- **Base Network + Processing Time**: This represents the minimum time for a request to traverse through 3 microservices (order → inventory → notification)
- **Network Latency**: Docker network overhead (~5-20ms per hop)
- **Service Processing**: Each service's Python/Flask overhead (~30-80ms per service)
- **Total Path**: Request goes order→inventory (wait for response)→notification (wait for response)→back to client
- **Variation**: Standard deviation shows consistency - lower is better, indicates stable performance

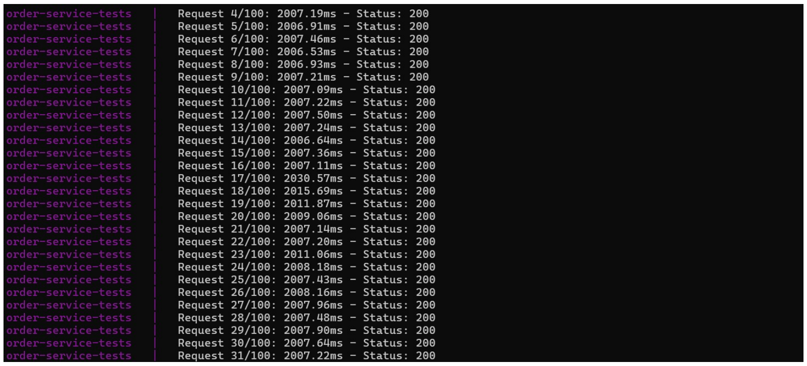
Test 2 (induced delay of 2s in inventory service), did not induce failure

Status: 200

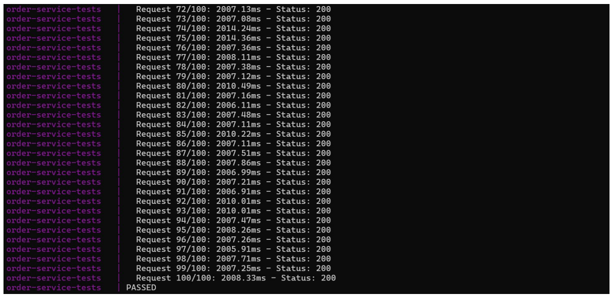
Average Latency: 2.07s



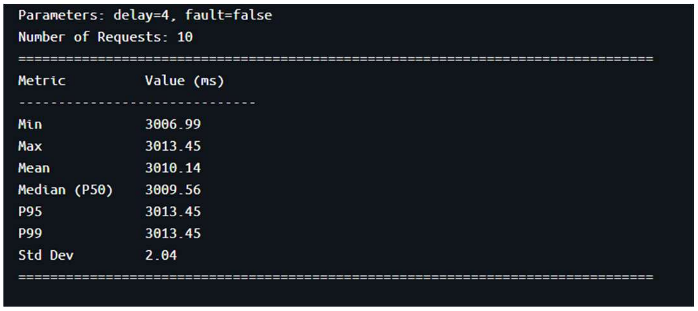
```
test_order_service.py::TestOrderServiceBaseline::test_order_with_delay_no_fault
[Test 2] Sending 100 requests with delay=2, fault=false...
  Request 1/100: 2008.09ms - Status: 200
  Request 2/100: 2007.23ms - Status: 200
  Request 3/100: 2006.95ms - Status: 200
  Request 4/100: 2007.19ms - Status: 200
  Request 5/100: 2006.91ms - Status: 200
  Request 6/100: 2007.46ms - Status: 200
  Request 7/100: 2006.53ms - Status: 200
  Request 8/100: 2006.93ms - Status: 200
  Request 9/100: 2007.21ms - Status: 200
  Request 10/100: 2007.09ms - Status: 200
  Request 11/100: 2007.22ms - Status: 200
  Request 12/100: 2007.50ms - Status: 200
  Request 13/100: 2007.24ms - Status: 200
  Request 14/100: 2006.64ms - Status: 200
  Request 15/100: 2007.36ms - Status: 200
  Request 16/100: 2007.11ms - Status: 200
  Request 17/100: 2030.57ms - Status: 200
  Request 18/100: 2015.69ms - Status: 200
  Request 19/100: 2011.87ms - Status: 200
```



```
order-service-tests    |    Request 4/100: 2007.19ms - Status: 200
order-service-tests    |    Request 5/100: 2006.91ms - Status: 200
order-service-tests    |    Request 6/100: 2007.46ms - Status: 200
order-service-tests    |    Request 7/100: 2006.53ms - Status: 200
order-service-tests    |    Request 8/100: 2006.93ms - Status: 200
order-service-tests    |    Request 9/100: 2007.21ms - Status: 200
order-service-tests    |    Request 10/100: 2007.09ms - Status: 200
order-service-tests    |    Request 11/100: 2007.22ms - Status: 200
order-service-tests    |    Request 12/100: 2007.50ms - Status: 200
order-service-tests    |    Request 13/100: 2007.24ms - Status: 200
order-service-tests    |    Request 14/100: 2006.64ms - Status: 200
order-service-tests    |    Request 15/100: 2007.36ms - Status: 200
order-service-tests    |    Request 16/100: 2007.11ms - Status: 200
order-service-tests    |    Request 17/100: 2030.57ms - Status: 200
order-service-tests    |    Request 18/100: 2015.69ms - Status: 200
order-service-tests    |    Request 19/100: 2011.87ms - Status: 200
order-service-tests    |    Request 20/100: 2009.06ms - Status: 200
order-service-tests    |    Request 21/100: 2007.14ms - Status: 200
order-service-tests    |    Request 22/100: 2007.20ms - Status: 200
order-service-tests    |    Request 23/100: 2011.06ms - Status: 200
order-service-tests    |    Request 24/100: 2008.18ms - Status: 200
order-service-tests    |    Request 25/100: 2007.43ms - Status: 200
order-service-tests    |    Request 26/100: 2008.16ms - Status: 200
order-service-tests    |    Request 27/100: 2007.96ms - Status: 200
order-service-tests    |    Request 28/100: 2007.48ms - Status: 200
order-service-tests    |    Request 29/100: 2007.90ms - Status: 200
order-service-tests    |    Request 30/100: 2007.64ms - Status: 200
order-service-tests    |    Request 31/100: 2007.22ms - Status: 200
```

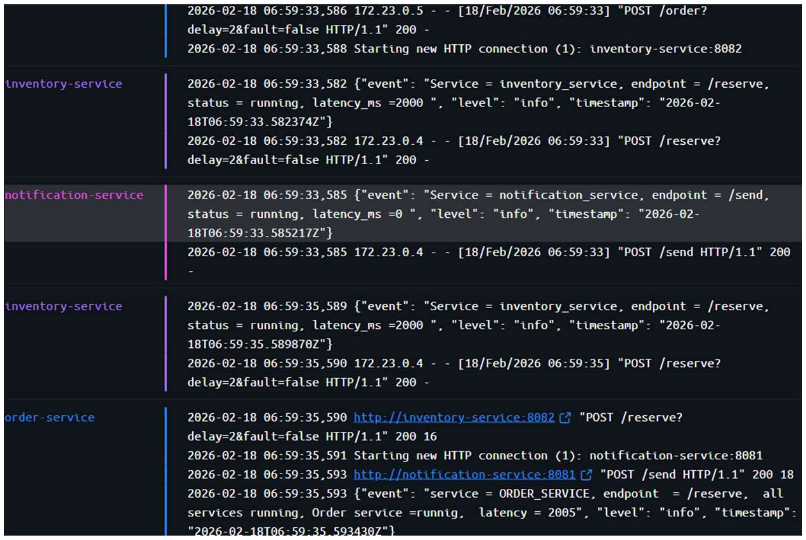


```
order-service-tests    |    Request 72/100: 2007.13ms - Status: 200
order-service-tests    |    Request 73/100: 2007.08ms - Status: 200
order-service-tests    |    Request 74/100: 2014.24ms - Status: 200
order-service-tests    |    Request 75/100: 2014.36ms - Status: 200
order-service-tests    |    Request 76/100: 2007.36ms - Status: 200
order-service-tests    |    Request 77/100: 2008.11ms - Status: 200
order-service-tests    |    Request 78/100: 2007.38ms - Status: 200
order-service-tests    |    Request 79/100: 2007.12ms - Status: 200
order-service-tests    |    Request 80/100: 2010.49ms - Status: 200
order-service-tests    |    Request 81/100: 2007.16ms - Status: 200
order-service-tests    |    Request 82/100: 2006.11ms - Status: 200
order-service-tests    |    Request 83/100: 2007.48ms - Status: 200
order-service-tests    |    Request 84/100: 2007.11ms - Status: 200
order-service-tests    |    Request 85/100: 2010.22ms - Status: 200
order-service-tests    |    Request 86/100: 2007.11ms - Status: 200
order-service-tests    |    Request 87/100: 2007.51ms - Status: 200
order-service-tests    |    Request 88/100: 2007.86ms - Status: 200
order-service-tests    |    Request 89/100: 2006.99ms - Status: 200
order-service-tests    |    Request 90/100: 2007.21ms - Status: 200
order-service-tests    |    Request 91/100: 2006.91ms - Status: 200
order-service-tests    |    Request 92/100: 2010.01ms - Status: 200
order-service-tests    |    Request 93/100: 2010.01ms - Status: 200
order-service-tests    |    Request 94/100: 2007.47ms - Status: 200
order-service-tests    |    Request 95/100: 2008.26ms - Status: 200
order-service-tests    |    Request 96/100: 2007.26ms - Status: 200
order-service-tests    |    Request 97/100: 2005.91ms - Status: 200
order-service-tests    |    Request 98/100: 2007.71ms - Status: 200
order-service-tests    |    Request 99/100: 2007.25ms - Status: 200
order-service-tests    |    Request 100/100: 2008.33ms - Status: 200
order-service-tests    |  PASSED
```



```
Parameters: delay=4, fault=false
Number of Requests: 10
=============================================================================
Metric          Value (ms)
-----------------------------
Min             3006.99
Max             3013.45
Mean            3010.14
Median (P50)    3009.56
P95             3013.45
P99             3013.45
Std Dev         2.04
=============================================================================
```

Logs:



```
                      2026-02-18 06:59:33,586 172.23.0.5 - - [18/Feb/2026 06:59:33] "POST /order?
                      delay=2&fault=false HTTP/1.1" 200 -
                      2026-02-18 06:59:33,588 Starting new HTTP connection (1): inventory-service:8082

inventory-service     2026-02-18 06:59:33,582 {"event": "Service = inventory_service, endpoint = /reserve,
                      status = running, latency_ms =2000 ", "level": "info", "timestamp": "2026-02-
                      18T06:59:33.582374Z"}
                      2026-02-18 06:59:33,582 172.23.0.4 - - [18/Feb/2026 06:59:33] "POST /reserve?
                      delay=2&fault=false HTTP/1.1" 200 -

notification-service  2026-02-18 06:59:33,585 {"event": "Service = notification_service, endpoint = /send,
                      status = running, latency_ms =0 ", "level": "info", "timestamp": "2026-02-
                      18T06:59:33.585217Z"}
                      2026-02-18 06:59:33,585 172.23.0.4 - - [18/Feb/2026 06:59:33] "POST /send HTTP/1.1" 200
                      -

inventory-service     2026-02-18 06:59:35,589 {"event": "Service = inventory_service, endpoint = /reserve,
                      status = running, latency_ms =2000 ", "level": "info", "timestamp": "2026-02-
                      18T06:59:35.589870Z"}
                      2026-02-18 06:59:35,590 172.23.0.4 - - [18/Feb/2026 06:59:35] "POST /reserve?
                      delay=2&fault=false HTTP/1.1" 200 -

order-service         2026-02-18 06:59:35,590 http://inventory-service:8082 "POST /reserve?
                      delay=2&fault=false HTTP/1.1" 200 16
                      2026-02-18 06:59:35,591 Starting new HTTP connection (1): notification-service:8081
                      2026-02-18 06:59:35,593 http://notification-service:8081 "POST /send HTTP/1.1" 200 18
                      2026-02-18 06:59:35,593 {"event": "service = ORDER_SERVICE, endpoint  = /reserve,  all
                      services running, Order service =runnig,  latency = 2005", "level": "info", "timestamp":
                      "2026-02-18T06:59:35.593430Z"}
```

Reasoning:

- **Induced Delay**: Inventory service sleeps for exactly 2000ms (2 seconds)
- **Base Overhead**: Add the ~100-300ms from Test 1
- **Expected Mean**: 2000ms (delay) + ~100-300ms (base) = **~2100-2300ms**
- **Why Mean > 2000ms**: The delay is ADDED to the normal processing time, not replacing it
- **Slightly Higher Std Dev**: Longer requests have more opportunities for variations (network fluctuations, system load)
- **P95/P99 Important**: Shows tail latency - some requests take longer due to system contention

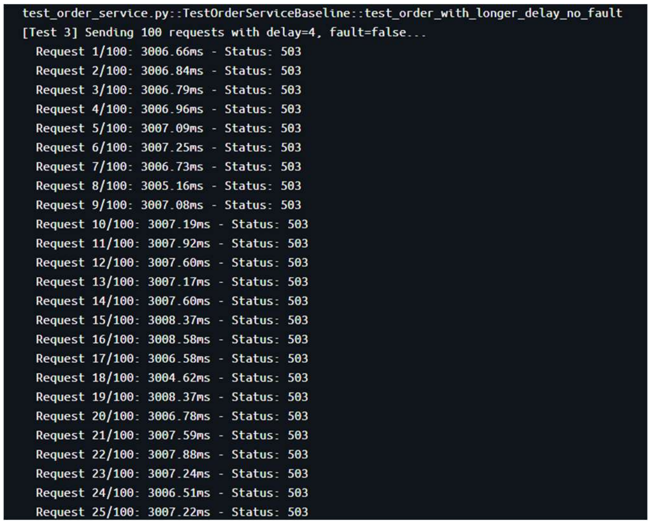
Test 3

Inducing Failure due to timeout

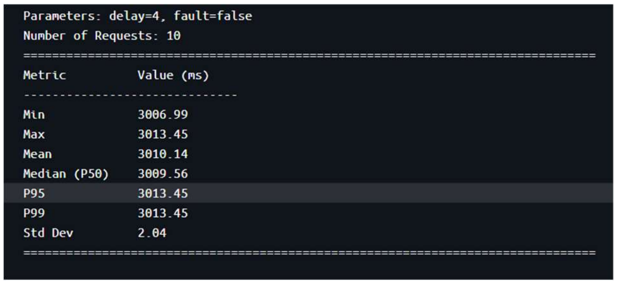
Average Latency :3.007s

Status: 503

Delay Induced at Inventory Service, resulted Timeout in Order Service, did not cause complete failure in Order Service

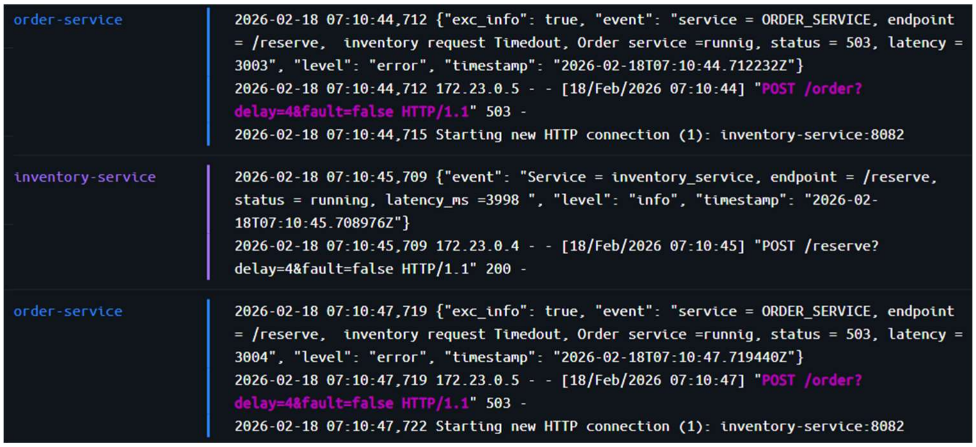


```
test_order_service.py::TestOrderServiceBaseline::test_order_with_longer_delay_no_fault
[Test 3] Sending 100 requests with delay=4, fault=false...
  Request 1/100: 3006.66ms - Status: 503
  Request 2/100: 3006.84ms - Status: 503
  Request 3/100: 3006.79ms - Status: 503
  Request 4/100: 3006.96ms - Status: 503
  Request 5/100: 3007.09ms - Status: 503
  Request 6/100: 3007.25ms - Status: 503
  Request 7/100: 3006.73ms - Status: 503
  Request 8/100: 3005.16ms - Status: 503
  Request 9/100: 3007.08ms - Status: 503
  Request 10/100: 3007.19ms - Status: 503
  Request 11/100: 3007.92ms - Status: 503
  Request 12/100: 3007.60ms - Status: 503
  Request 13/100: 3007.17ms - Status: 503
  Request 14/100: 3007.60ms - Status: 503
  Request 15/100: 3008.37ms - Status: 503
  Request 16/100: 3008.58ms - Status: 503
  Request 17/100: 3006.58ms - Status: 503
  Request 18/100: 3004.62ms - Status: 503
  Request 19/100: 3008.37ms - Status: 503
  Request 20/100: 3006.78ms - Status: 503
  Request 21/100: 3007.59ms - Status: 503
  Request 22/100: 3007.88ms - Status: 503
  Request 23/100: 3007.24ms - Status: 503
  Request 24/100: 3006.51ms - Status: 503
  Request 25/100: 3007.22ms - Status: 503
```



```
Parameters: delay=4, fault=false
Number of Requests: 10
================================================================================
Metric          Value (ms)
------------------------------
Min             3006.99
Max             3013.45
Mean            3010.14
Median (P50)    3009.56
P95             3013.45
P99             3013.45
Std Dev         2.04
================================================================================
```

Logs:



```
order-service          2026-02-18 07:10:44,712 {"exc_info": true, "event": "service = ORDER_SERVICE, endpoint
                       = /reserve,  inventory request Timedout, Order service =runnig, status = 503, latency =
                       3003", "level": "error", "timestamp": "2026-02-18T07:10:44.712232Z"}
                       2026-02-18 07:10:44,712 172.23.0.5 - - [18/Feb/2026 07:10:44] "POST /order?
                       delay=4&fault=false HTTP/1.1" 503 -
                       2026-02-18 07:10:44,715 Starting new HTTP connection (1): inventory-service:8082

inventory-service      2026-02-18 07:10:45,709 {"event": "Service = inventory_service, endpoint = /reserve,
                       status = running, latency_ms =3998 ", "level": "info", "timestamp": "2026-02-
                       18T07:10:45.708976Z"}
                       2026-02-18 07:10:45,709 172.23.0.4 - - [18/Feb/2026 07:10:45] "POST /reserve?
                       delay=4&fault=false HTTP/1.1" 200 -

order-service          2026-02-18 07:10:47,719 {"exc_info": true, "event": "service = ORDER_SERVICE, endpoint
                       = /reserve,  inventory request Timedout, Order service =runnig, status = 503, latency =
                       3004", "level": "error", "timestamp": "2026-02-18T07:10:47.719440Z"}
                       2026-02-18 07:10:47,719 172.23.0.5 - - [18/Feb/2026 07:10:47] "POST /order?
                       delay=4&fault=false HTTP/1.1" 503 -
                       2026-02-18 07:10:47,722 Starting new HTTP connection (1): inventory-service:8082
```

Reasoning:

Reasoning:

- **Induced Delay**: 3000ms in inventory service
- **Base Overhead**: Same ~100-300ms from Test 1
- **Expected Mean**: 3000ms + ~100-300ms = **~3100-3300ms**
- **Linear Scaling**: Should see ~1000ms more than Test 2
- **Important to Note**: Inducing Timeout did not cause complete failure, rather partial failure in Inventory_service