# Iteration 2

## Test Records

### Testing Different Types of Requests in JMeter - Dynamic

- We will test two types of requests:
    - Static Content Requests
        - Examples: HTML files, CSS, JavaScript, images.
        - Goal: Measure how well Nginx serves cached/static content.
    - Dynamic Content Requests
        - Examples: API calls, database queries, PHP scripts.
        - Goal: Measure performance when Nginx processes requests dynamically.
- Configure Nginx for Static and Dynamic Content
    - Create a Static HTML File

```
root@ip-172-31-1-46:~# echo "<h1>Static Page Test</h1><p>This is a static HTML file served by Nginx.</p>" | sudo tee /v
tml/static.html
<h1>Static Page Test</h1><p>This is a static HTML file served by Nginx.</p>
```

        - Verify the file

```
root@ip-172-31-1-46:~# ls -l /var/www/html/
total 8
-rw-r--r-- 1 root root 615 Feb 16 07:16 index.nginx-deb
-rw-r--r-- 1 root root  76 Feb 22 11:17 static.html
```

    - Create a Dynamic PHP Script

```
root@ip-172-31-1-46:~# echo "<?php echo json_encode(['status' => 'success', 'message' => 'This is a dynamic response']);
sudo tee /var/www/html/api.php
<?php echo json_encode(['status' => 'success', 'message' => 'This is a dynamic response']); ?>
```

        - Verify the file

```
root@ip-172-31-1-46:~# ls -l /var/www/html/
total 12
-rw-r--r-- 1 root root  95 Feb 22 11:21 api.php
-rw-r--r-- 1 root root 615 Feb 16 07:16 index.nginx-debian.
-rw-r--r-- 1 root root  76 Feb 22 11:17 static.html
```

    - Configure Nginx to Support PHP
        - Open the default Nginx config file:

```
sudo nano /etc/nginx/sites-available/default
```

        - Update block
          Before:

```
location / {
    index index.html index.htm index.nginx-debian.html;
    try_files $uri $uri/ =404;
}
```

After:

```
location / {
    index index.html index.htm index.nginx-debian.html;
    try_files $uri $uri/ =404;
}

# PHP Processing
location ~ \.php$ {
    include snippets/fastcgi-php.conf;
    fastcgi_pass unix:/run/php/php-fpm.sock;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    include fastcgi_params;
}
```

- Test if Static and Dynamic Content Works
  - Test Dynamic Content: curl http://localhost/api.php

```
root@ip-172-31-1-46:~# curl http://localhost/api.php
{"status":"success","message":"This is a dynamic response"}root@ip-172-31-1-46:~#
```

  - Test Static Content

```
root@ip-172-31-1-46:~# curl http://localhost/static.html
<h1>Static Page Test</h1><p>This is a static HTML file served by Nginx.</p>
```

- Test Static
  - Create a Static Test Plan

```
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="Static Content Test" enabled="true">
      <hashTree>
        <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup" testname="Thread Group" enabled="tr
ue">
          <stringProp name="ThreadGroup.num_threads">10</stringProp> <!-- Number of users -->
          <stringProp name="ThreadGroup.ramp_time">5</stringProp> <!-- Ramp-up time -->
          <stringProp name="ThreadGroup.duration">30</stringProp> <!-- Duration -->
          <hashTree>
            <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="Static F
ile Request" e<stringProp name="HTTPSampler.domain">172.31.1.46</stringProp>
              <stringProp name="HTTPSampler.port">80</stringProp>
              <stringProp name="HTTPSampler.protocol">http</stringProp>
              <stringProp name="HTTPSampler.path">/test/sample.html</stringProp>
              <stringProp name="HTTPSampler.method">GET</stringProp>
            </HTTPSamplerProxy>
          </hashTree>
        </ThreadGroup>
      </hashTree>
    </TestPlan>
  </hashTree>
</jmeterTestPlan>
~
```

- Run the Test

```
root@ip-172-31-15-56:~/tests# vi test-static.jmx
root@ip-172-31-15-56:~/tests# jmeter -n -t test-static.jmx -l result-static.jtl -e -o report-sta
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 in a future release
Creating summariser <summary>
Created the tree successfully using test-static.jmx
Starting standalone test @ 2025 Feb 22 11:50:53 UTC (1740225053316)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary =    100 in 00:00:01 =   96.4/s Avg:    34 Min:     1 Max:   122 Err:     0 (0.00%)
Tidying up ...    @ 2025 Feb 22 11:50:54 UTC (1740225054529)
... end of run
```

- Analyze the Result

```
... end of run
root@ip-172-31-15-56:~/tests# cat result-static.jtl
timeStamp,elapsed,label,responseCode,responseMessage,threadName,dataType,success,failureMessage,bytes
,grpThreads,allThreads,URL,Latency,IdleTime,Connect
1741991139346,76,HTTP Request,200,OK,Thread Group 1-35,text,true,,857,108,42,42,http://172.31.1.46/,6
1741991139319,101,HTTP Request,200,OK,Thread Group 1-7,text,true,,857,108,42,42,http://172.31.1.46/,8
1741991139320,102,HTTP Request,200,OK,Thread Group 1-2,text,true,,857,108,42,42,http://172.31.1.46/,9
1741991139319,103,HTTP Request,200,OK,Thread Group 1-21,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139320,102,HTTP Request,200,OK,Thread Group 1-26,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139322,100,HTTP Request,200,OK,Thread Group 1-23,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139322,100,HTTP Request,200,OK,Thread Group 1-24,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139322,100,HTTP Request,200,OK,Thread Group 1-29,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139322,100,HTTP Request,200,OK,Thread Group 1-10,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139318,104,HTTP Request,200,OK,Thread Group 1-31,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139322,100,HTTP Request,200,OK,Thread Group 1-22,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139320,102,HTTP Request,200,OK,Thread Group 1-5,text,true,,857,108,42,42,http://172.31.1.46/,9
1741991139320,102,HTTP Request,200,OK,Thread Group 1-1,text,true,,857,108,42,42,http://172.31.1.46/,9
1741991139322,100,HTTP Request,200,OK,Thread Group 1-30,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139320,102,HTTP Request,200,OK,Thread Group 1-27,text,true,,857,108,42,42,http://172.31.1.46/,
1741991139363,59,HTTP Request,200,OK,Thread Group 1-36,text,true,,857,108,42,42,http://172.31.1.46/,4
1741991139319,103,HTTP Request,200,OK,Thread Group 1-3,text,true,,857,108,42,42,http://172.31.1.46/,9
1741991139369,53,HTTP Request,200,OK,Thread Group 1-37,text,true,,857,108,42,42,http://172.31.1.46/,3
1741991139320,102,HTTP Request,200,OK,Thread Group 1-8,text,true,,857,108,42,42,http://172.31.1.46/,8
```

- Test dynamic
  - Create a Dynamic Test Plan

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="5.0" jmeter="5.5">
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="Dynamic Test Plan" enabled="true">
      <stringProp name="TestPlan.comments"></stringProp>
      <boolProp name="TestPlan.functional_mode">false</boolProp>
      <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
      <elementProp name="TestPlan.user_defined_variables" elementType="Arguments" guiclass="ArgumentsPanel" testclass="Arguments" testname="
User Defined Variables" enabled="true"/>
      <stringProp name="TestPlan.user_define_classpath"></stringProp>
    </TestPlan>
    <hashTree>
      <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup" testname="Thread Group" enabled="true">
        <stringProp name="ThreadGroup.on_sample_error">continue</stringProp>
        <elementProp name="ThreadGroup.main_controller" elementType="LoopController" guiclass="LoopControlPanel" testclass="LoopController"
testname="Loop Controller" enabled="true">
          <boolProp name="LoopController.continue_forever">false</boolProp>
          <stringProp name="LoopController.loops">1</stringProp>
        </elementProp>
        <stringProp name="ThreadGroup.num_threads">100</stringProp>
        <stringProp name="ThreadGroup.ramp_time">1</stringProp>
        <longProp name="ThreadGroup.start_time">1659101628000</longProp>
        <longProp name="ThreadGroup.end_time">1659101628000</longProp>
        <boolProp name="ThreadGroup.scheduler">false</boolProp>
        <stringProp name="ThreadGroup.duration"></stringProp>
        <stringProp name="ThreadGroup.delay"></stringProp>
      </ThreadGroup>
      <hashTree>
        <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="PHP Request" enabled="true">
          <stringProp name="HTTPSampler.domain">172.31.1.46</stringProp>
          <stringProp name="HTTPSampler.port">80</stringProp>
          <stringProp name="HTTPSampler.protocol">http</stringProp>
          <stringProp name="HTTPSampler.path">/test/index.php</stringProp>
          <stringProp name="HTTPSampler.method">GET</stringProp>
        </HTTPSamplerProxy>
        <hashTree/>
      </hashTree>
    </hashTree>
  </hashTree>
</jmeterTestPlan>
```

- Run the test

```
root@ip-172-31-15-56:~/tests# jmeter -n -t test-dynamic-0312.jmx -l result-dynamic.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
Creating summariser <summary>
Created the tree successfully using test-dynamic-0312.jmx
Starting standalone test @ 2025 Mar 14 21:52:49 UTC (1741989169657)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary =    100 in 00:00:01 =   96.2/s Avg:    40 Min:     1 Max:   126 Err:     0 (0.00%)
Tidying up ...    @ 2025 Mar 14 21:52:51 UTC (1741989171042)
... end of run
```

- Analyze the Result

```
root@ip-172-31-15-56:~/tests# cat result-dynamic.jtl
timeStamp,elapsed,label,responseCode,responseMessage,threadName,dataType,success,failureMessage,bytes,sentBytes,grpThreads,allThrea
,IdleTime,Connect
1741989170368,88,PHP Request,200,OK,Thread Group 1-6,text,true,,244,115,41,41,http://172.31.1.46/api.php,86,0,56
1741989170383,75,PHP Request,200,OK,Thread Group 1-5,text,true,,244,115,42,42,http://172.31.1.46/api.php,62,0,27
1741989170378,80,PHP Request,200,OK,Thread Group 1-16,text,true,,244,115,42,42,http://172.31.1.46/api.php,76,0,36
1741989170352,106,PHP Request,200,OK,Thread Group 1-20,text,true,,244,115,42,42,http://172.31.1.46/api.php,101,0,60
1741989170378,80,PHP Request,200,OK,Thread Group 1-21,text,true,,244,115,42,42,http://172.31.1.46/api.php,75,0,36
1741989170378,83,PHP Request,200,OK,Thread Group 1-15,text,true,,244,115,42,42,http://172.31.1.46/api.php,75,0,36
1741989170378,85,PHP Request,200,OK,Thread Group 1-14,text,true,,244,115,40,40,http://172.31.1.46/api.php,74,0,36
1741989170457,7,PHP Request,200,OK,Thread Group 1-42,text,true,,244,115,38,38,http://172.31.1.46/api.php,7,0,6
1741989170379,87,PHP Request,200,OK,Thread Group 1-18,text,true,,244,115,37,37,http://172.31.1.46/api.php,73,0,34
1741989170418,49,PHP Request,200,OK,Thread Group 1-38,text,true,,244,115,37,37,http://172.31.1.46/api.php,23,0,7
1741989170378,90,PHP Request,200,OK,Thread Group 1-19,text,true,,244,115,33,33,http://172.31.1.46/api.php,74,0,35
1741989170378,91,PHP Request,200,OK,Thread Group 1-8,text,true,,244,115,33,33,http://172.31.1.46/api.php,74,0,40
1741989170367,102,PHP Request,200,OK,Thread Group 1-33,text,true,,244,115,32,32,http://172.31.1.46/api.php,85,0,53
1741989170381,89,PHP Request,200,OK,Thread Group 1-1,text,true,,244,115,31,31,http://172.31.1.46/api.php,71,0,37
1741989170448,22,PHP Request,200,OK,Thread Group 1-41,text,true,,244,115,30,30,http://172.31.1.46/api.php,4,0,4
1741989170378,93,PHP Request,200,OK,Thread Group 1-7,text,true,,244,115,29,29,http://172.31.1.46/api.php,74,0,42
1741989170366,104,PHP Request,200,OK,Thread Group 1-22,text,true,,244,115,28,28,http://172.31.1.46/api.php,86,0,53
1741989170383,89,PHP Request,200,OK,Thread Group 1-9,text,true,,244,115,27,27,http://172.31.1.46/api.php,69,0,37
1741989170356,116,PHP Request,200,OK,Thread Group 1-32,text,true,,244,115,26,26,http://172.31.1.46/api.php,96,0,64
```

- Findings:
  The dynamic file testing results indicate that the server successfully handled **100 requests** to the `/api.php` endpoint, with all responses returning **200 OK**. The **average response time was 40ms**, with a **minimum of 1ms and a maximum of 126ms**, which suggests the system was responsive but had some variability in latency. The request sizes were relatively small (**244 bytes received, 115 bytes sent per request**), meaning the payload was lightweight. No errors occurred, and thread activity was consistent, showing the system maintained stability. However, the fluctuations in response time, particularly the higher maximum latency of **126ms**, suggest potential optimizations are needed for handling dynamic content at higher loads. Future iterations should increase concurrent users and loops to determine how the server performs under heavier dynamic loads.

- Result
  The comparison between static and dynamic file testing reveals key performance differences. In **static file testing**, the response sizes were significantly larger (**857 bytes received, 108 bytes sent per request**) compared to **dynamic requests** (**244 bytes received, 115 bytes sent per request**). The static files had **higher average response times**, ranging from **76ms to 126ms**, while dynamic requests averaged **40ms**, with a max of **126ms**. This suggests that serving static content incurs more overhead, potentially due to file I/O operations or inefficient caching mechanisms. Despite this, **both tests resulted in 100% successful requests (200 OK)**, indicating system stability. However, the **higher response time variance in static requests** suggests that performance optimizations, such as caching strategies or a CDN, could improve static content delivery.

Additionally, future tests could evaluate how the server handles **higher concurrent loads and different payload sizes** to identify potential bottlenecks.

## Testing Size of Data

- Goal: identify how Nginx handles small vs. large payloads.
- Prepare Test Data
  - Create different file sizes on your Nginx server to be requested by JMeter.

```
snap
root@ip-172-31-1-46:~# cd /var/www/html/
sudo dd if=/dev/zero of=small_file.txt bs=1K count=10  # 10KB file
sudo dd if=/dev/zero of=medium_file.txt bs=1K count=500  # 500KB file
sudo dd if=/dev/zero of=large_file.txt bs=1M count=5  # 5MB file
sudo dd if=/dev/zero of=huge_file.txt bs=1M count=50  # 50MB file
10+0 records in
10+0 records out
10240 bytes (10 kB, 10 KiB) copied, 3.8614e-05 s, 265 MB/s
500+0 records in
500+0 records out
512000 bytes (512 kB, 500 KiB) copied, 0.00136536 s, 375 MB/s
5+0 records in
5+0 records out
5242880 bytes (5.2 MB, 5.0 MiB) copied, 0.00268571 s, 2.0 GB/s
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.0262147 s, 2.0 GB/s
```

- Verify the files are accessible:

```
root@ip-172-31-1-46:/var/www/html# curl -I http://172.31.1.46/small_file.txt
curl -I http://172.31.1.46/medium_file.txt
curl -I http://172.31.1.46/large_file.txt
curl -I http://172.31.1.46/huge_file.txt
HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Sat, 22 Feb 2025 13:52:28 GMT
Content-Type: text/plain
Content-Length: 10240
Last-Modified: Sat, 22 Feb 2025 13:51:00 GMT
Connection: keep-alive
ETag: "67b9d644-2800"
Accept-Ranges: bytes

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Sat, 22 Feb 2025 13:52:28 GMT
Content-Type: text/plain
Content-Length: 512000
Last-Modified: Sat, 22 Feb 2025 13:51:00 GMT
Connection: keep-alive
ETag: "67b9d644-7d000"
Accept-Ranges: bytes

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Sat, 22 Feb 2025 13:52:28 GMT
Content-Type: text/plain
Content-Length: 5242880
Last-Modified: Sat, 22 Feb 2025 13:51:00 GMT
Connection: keep-alive
ETag: "67b9d644-500000"
Accept-Ranges: bytes

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Sat, 22 Feb 2025 13:52:28 GMT
Content-Type: text/plain
Content-Length: 52428800
Last-Modified: Sat, 22 Feb 2025 13:51:00 GMT
Connection: keep-alive
ETag: "67b9d644-3200000"
Accept-Ranges: bytes
```

- Create a JMeter test plan test-size.jmx

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="5.0" jmeter="5.5">
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="Test Size Plan" enabled
      <stringProp name="TestPlan.comments">Testing different request sizes</stringProp>
      <boolProp name="TestPlan.functional_mode">false</boolProp>
      <boolProp name="TestPlan.serialize_threadgroups">false</boolProp>
      <elementProp name="TestPlan.user_defined_variables" elementType="Arguments" guiclass=
User Defined Variables" enabled="true">
        <collectionProp name="Arguments.arguments">
          <elementProp name="server" elementType="Argument">
            <stringProp name="Argument.name">server</stringProp>
            <stringProp name="Argument.value">172.31.1.46</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
          <elementProp name="port" elementType="Argument">
            <stringProp name="Argument.name">port</stringProp>
            <stringProp name="Argument.value">80</stringProp>
            <stringProp name="Argument.metadata">=</stringProp>
          </elementProp>
        </collectionProp>
      </elementProp>
      <stringProp name="TestPlan.user_define_classpath"></stringProp>
    </TestPlan>
    <hashTree>
      <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup" testname="Size Test Gr
        <stringProp name="ThreadGroup.on_sample_error">continue</stringProp>
        <elementProp name="ThreadGroup.main_controller" elementType="LoopController" guicla
testname="Loop Controller" enabled="true">
          <boolProp name="LoopController.continue_forever">false</boolProp>
          <stringProp name="LoopController.loops">1</stringProp>
        </elementProp>
        <stringProp name="ThreadGroup.num_threads">50</stringProp>
        <stringProp name="ThreadGroup.ramp_time">5</stringProp>
        <boolProp name="ThreadGroup.scheduler">false</boolProp>
      </ThreadGroup>
      <hashTree>
        <!-- Small Request (1 KB) -->
        <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testnam
```

- Run the Test
  - Small File

```
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and wil
 future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and wil
 future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and wil
 future release
Creating summariser <summary>
Created the tree successfully using test-size-0312.jmx
Starting standalone test @ 2025 Mar 14 22:33:24 UTC (1741991604207)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary =     10 in 00:00:02 =    5.4/s Avg:    13 Min:     3 Max:    60 Err:     0 (0.00%)
Tidying up ...    @ 2025 Mar 14 22:33:26 UTC (1741991606460)
... end of run
```

- Medium File
  jmeter -n -t test-size-0312.jmx -l result-medium.jtl

```
root@ip-172-31-15-56:~/tests# jmeter -n -t test-size-0312.jmx -l result-medium.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 future release
Creating summariser <summary>
Created the tree successfully using test-size-0312.jmx
Starting standalone test @ 2025 Mar 14 22:35:45 UTC (1741991745987)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary =     10 in 00:00:02 =    5.4/s Avg:    20 Min:     7 Max:    66 Err:     0 (0.00%)
Tidying up ...    @ 2025 Mar 14 22:35:48 UTC (1741991748191)
... end of run
```

- Large File

```
root@ip-172-31-15-56:~/tests# jmeter -n -t test-size-0312.jmx -l result-large.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will
 future release
Creating summariser <summary>
Created the tree successfully using test-size-0312.jmx
Starting standalone test @ 2025 Mar 14 22:37:43 UTC (1741991863191)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary =     10 in 00:00:02 =    5.3/s Avg:    69 Min:    45 Max:   155 Err:     0 (0.00%)
Tidying up ...    @ 2025 Mar 14 22:37:45 UTC (1741991865453)
... end of run
```

- Findings:
  **Analysis of Small, Medium, and Large File Test Results**

  The JMeter tests for different file sizes reveal key performance trends regarding response time, latency, and throughput.

1. **Response Time Trends**:
   - **Small file (`~10KB`)**: The average response time is between **3ms to 60ms**.
   - **Medium file (`~500KB`)**: The response time increases slightly, ranging from **7ms to 66ms**.
   - **Large file (`~5MB`)**: There is a **significant jump in response time**, with times ranging from **45ms to 155ms**.
2. **Latency and Connection Time**:
   - **For small and medium files**, the latency is relatively low (mostly under **50ms**).
   - **For large files**, latency increases significantly (ranging from **46ms to 155ms**), indicating higher I/O processing overhead.
3. **Throughput Analysis**:
   Throughput remains **high** for small and medium files, meaning the server can process multiple requests quickly.

    - For **large files**, the throughput drops as response times increase, suggesting potential bandwidth limitations or increased processing overhead.
4. **Impact on Server Load**:
    - Small and medium files perform **efficiently** under load with minimal delays.
    - Large files **consume more server resources**, increasing latency and affecting concurrent request performance.

### Key Takeaways:

- The **server handles small and medium files efficiently**, maintaining low latency and fast response times.
- **Large files introduce noticeable delays**, indicating potential bottlenecks in **network bandwidth, disk I/O, or server processing power**.
- **Possible Optimizations**:
    - **Enable caching** (e.g., `expires` and `Cache-Control` headers in Nginx).
    - **Use a CDN** for large file distribution.
    - **Optimize file compression** to reduce transfer size.
    - **Increase buffer size** in Nginx to improve large file handling.

# Stress Test Number of Users

## Number of Users = 200

- Note:
    - **Loop = 500** – ensures that the system is **continuously stressed over time**, identifying degradation effects such as **memory leaks, CPU exhaustion, and slowdowns**.
- Create the test file test-user-stress.jmx and make the number of users = 200

```
    <elementProp name="ThreadGroup.main_controller" elementType="LoopController" guiclass="Loop
er" testname="Loop Controller" enabled="true">
      <boolProp name="LoopController.continue_forever">false</boolProp>
      <stringProp name="LoopController.loops">500</stringProp>
    </elementProp>
    <stringProp name="ThreadGroup.num_threads">200</stringProp>
    <stringProp name="ThreadGroup.ramp_time">10</stringProp>
    <boolProp name="ThreadGroup.scheduler">false</boolProp>
    <stringProp name="ThreadGroup.duration"></stringProp>
    <stringProp name="ThreadGroup.delay"></stringProp>
  </ThreadGroup>
  <hashTree>
    <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="HTTP
```

- Run the test with 200 users and save the result in file: result-200user-stress.jtl

```
root@ip-172-31-15-56:~/tests# jmeter -n -t test-user-stress.jmx -l result-200user-stress.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future r
elease
Creating summariser <summary>
Created the tree successfully using test-user-stress.jmx
Starting standalone test @ 2025 Mar 14 21:40:53 UTC (1741988453073)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary +   8077 in 00:00:07 = 1230.3/s Avg:    29 Min:     1 Max:   462 Err:     0 (0.00%) Active: 132 Started: 132 Fin
ished: 0
summary +  77821 in 00:00:30 = 2594.1/s Avg:    26 Min:     1 Max:  1506 Err:     0 (0.00%) Active: 166 Started: 200 Fin
ished: 34
summary =  85898 in 00:00:37 = 2349.3/s Avg:    26 Min:     1 Max:  1506 Err:     0 (0.00%)
summary +  14102 in 00:00:05 = 2678.4/s Avg:    19 Min:     0 Max:   711 Err:     0 (0.00%) Active: 0 Started: 200 Finis
hed: 200
summary = 100000 in 00:00:42 = 2390.7/s Avg:    25 Min:     0 Max:  1506 Err:     0 (0.00%)
Tidying up ...    @ 2025 Mar 14 21:41:35 UTC (1741988495267)
... end of run
```

- View the result

```
root@ip-172-31-15-56:~/tests# tail -n 20 ~/tests/result-200user-stress.jtl
1741988576224,2,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,2,0,2
1741988576227,0,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,0,0,0
1741988576227,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,1
1741988576228,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,1
1741988576229,2,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,2,0,0
1741988576231,0,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,0,0,0
1741988576231,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,1
1741988576232,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,0
1741988576233,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,1
1741988576234,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,1
1741988576235,3,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,3,0,2
1741988576238,0,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,0,0,0
1741988576238,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,1
1741988576239,3,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,3,0,2
1741988576242,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,0
1741988576243,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,0
1741988576244,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,1
1741988576245,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,1
1741988576246,3,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,3,0,2
1741988576249,1,HTTP Request,200,OK,User Stress Test Group 1-177,text,true,,857,108,1,1,http://172.31.1.46/,1,0,0
```

- CPU usage: went up tp 32.3%

```
    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    674 www-data  20   0   13204   5848   4224 R  32.3   0.6  76:22.11 nginx
```

- Findings
    - The 200-user stress test showed that the server handled requests efficiently, with all responses returning **200 OK** and response times mostly between **1-3 milliseconds**. The **highest CPU usage recorded was 32.3%**, indicating that the

system was not overloaded and still had capacity for higher loads. The test ran consistently without significant latency spikes or failures, suggesting that the server performed well under this level of stress. Since CPU utilization remained moderate, we can increase the number of users, loops, or throughput in future tests to identify the system's breaking point and determine its true performance limits.

## Number of Users = 500

- Loop = 500

```
    <elementProp name="ThreadGroup.main_controller" elementType="LoopController
Controller" testname="Loop Controller" enabled="true">
      <boolProp name="LoopController.continue_forever">false</boolProp>
      <stringProp name="LoopController.loops">500</stringProp>
    </elementProp>
    <stringProp name="ThreadGroup.num_threads">500</stringProp>
    <stringProp name="ThreadGroup.ramp_time">10</stringProp>
    <boolProp name="ThreadGroup.scheduler">false</boolProp>
```

  - Run the test with 500 users and save the result in file: result-500user-stress.jtl

```
root@ip-172-31-15-56:~/tests# jmeter -n -t test-user-stress.jmx -l result-500user-stress.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future relea
se
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future relea
se
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future relea
se
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future relea
se
Creating summariser <summary>
Created the tree successfully using test-user-stress.jmx
Starting standalone test @ 2025 Mar 14 08:47:28 UTC (1741942048281)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary +    608 in 00:00:01 =  472.8/s Avg:    34 Min:     4 Max:   178 Err:     0 (0.00%) Active: 69 Started: 69 Finished:
 0
summary +  65973 in 00:00:30 = 2199.0/s Avg:    25 Min:     0 Max:   992 Err:     0 (0.00%) Active: 500 Started: 500 Finishe
d: 0
summary =  66581 in 00:00:31 = 2127.9/s Avg:    25 Min:     0 Max:   992 Err:     0 (0.00%)
summary +  86018 in 00:00:30 = 2866.8/s Avg:    21 Min:     0 Max:  1297 Err:     0 (0.00%) Active: 500 Started: 500 Finishe
d: 0
summary = 152599 in 00:01:01 = 2489.6/s Avg:    23 Min:     0 Max:  1297 Err:     0 (0.00%)
```

  - After 12 mins, still did not finish running, terminate it, check the result file
    - It has been running for 12 minutes

```
d: 0
summary = 2212392 in 00:10:31 = 3504.6/s Avg:    19 Min:     0 Max:  3002 Err:     0 (0.00%)
summary + 118618 in 00:00:30 = 3953.9/s Avg:    16 Min:     0 Max:   592 Err:     0 (0.00%) Active: 500 Started: 500 F
d: 0
summary = 2331010 in 00:11:01 = 3525.0/s Avg:    19 Min:     0 Max:  3002 Err:     0 (0.00%)
summary + 117223 in 00:00:30 = 3907.4/s Avg:    18 Min:     0 Max:   608 Err:     0 (0.00%) Active: 500 Started: 500 F
d: 0
summary = 2448233 in 00:11:31 = 3541.6/s Avg:    19 Min:     0 Max:  3002 Err:     0 (0.00%)
summary + 115626 in 00:00:30 = 3854.2/s Avg:    20 Min:     0 Max:  1439 Err:     0 (0.00%) Active: 500 Started: 500 F
d: 0
summary = 2563859 in 00:12:01 = 3554.6/s Avg:    19 Min:     0 Max:  3002 Err:     0 (0.00%)
^C
```

- Check the last few lines of the result file

```
root@ip-172-31-15-56:~/tests# tail -n 10 result-500user-stress.jtl
1741942775641,35,HTTP Request,200,OK,User Stress Test Group 1-204,text,true,,857,108,500,500,http://172.31.1.46/,35,0,16
1741942775676,0,HTTP Request,200,OK,User Stress Test Group 1-132,text,true,,857,108,500,500,http://172.31.1.46/,0,0,0
1741942775676,1,HTTP Request,200,OK,User Stress Test Group 1-271,text,true,,857,108,500,500,http://172.31.1.46/,1,0,0
1741942775676,1,HTTP Request,200,OK,User Stress Test Group 1-377,text,true,,857,108,500,500,http://172.31.1.46/,1,0,0
1741942775675,2,HTTP Request,200,OK,User Stress Test Group 1-400,text,true,,857,108,500,500,http://172.31.1.46/,1,0,0
1741942775642,35,HTTP Request,200,OK,User Stress Test Group 1-244,text,true,,857,108,500,500,http://172.31.1.46/,35,0,15
1741942775644,33,HTTP Request,200,OK,User Stress Test Group 1-313,text,true,,857,108,500,500,http://172.31.1.46/,33,0,12
1741942775644,34,HTTP Request,200,OK,User Stress Test Group 1-319,text,true,,857,108,500,500,http://172.31.1.46/,34,0,13
1741942773355,16,HTTP Request,200,OK,User Stress Test Group 1-218,text,true,,857,108,500,500,http://172.31.1.46/,16,0,14
1741942775646,35,HTTP Request,200,OK,User Stress Test Group 1-11,text,true,,857,108,500,500,http://172.31.1.46/,35,0,11
root@ip-172-31-15-56:~/tests#
```

- CPU Usage:
  It went up to 58.7%

```
MiB Mem :    957.4 total,     87.7 free,    325.1 used,    706.9 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.    632.3 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    674 www-data  20   0   12984   5464   3968 R  58.7   0.6  24:29.60 nginx
```

- Findings: The stress test with 500 concurrent users showed that the system maintained a stable throughput of 3500-3900 requests per second, with an average response time of 18-20ms and no recorded errors. CPU usage peaked at 58.7%, indicating moderate stress but no critical resource exhaustion. Memory usage remained stable, and the system did not crash or show signs of failure. However, the test did not complete after 12 minutes, requiring manual termination, suggesting the need for a defined stopping condition. Overall, the system handled the load well, but further testing with higher user counts is needed to determine the true capacity limit.

## Stress Test Request Speed (Throughput) - Test How Many Requests per Second Nginx Can Handle

- **Goal:** We are conducting a **throughput-based stress test** to evaluate how well the system can handle **700, 1000, and 1200, 2000 requests per second (RPS)** over a sustained period.

- Throughput = 700
  - Note:
    - **Loop = -1** (Infinite) – Ensures continuous request generation.
    - **Thread Count = 100**, which should be sufficient to sustain 700 RPS.
    - **Ramp-up time =  10 seconds**, allowing users to ramp up the load gradually.

- Create the test file

```
troller" enabled="true">
      <boolProp name="LoopController.continue_forever">false</boolProp>
      <stringProp name="LoopController.loops">-1</stringProp> <!-- Infinite loops until manually stopped -->
    </elementProp>
    <stringProp name="ThreadGroup.num_threads">100</stringProp> <!-- Adjust the number of threads as needed -->
    <stringProp name="ThreadGroup.ramp_time">10</stringProp>
    <boolProp name="ThreadGroup.scheduler">false</boolProp>
    <stringProp name="ThreadGroup.duration"></stringProp>
    <stringProp name="ThreadGroup.delay"></stringProp>
  </ThreadGroup>
  <hashTree>
    <!-- Constant Throughput Timer to maintain 700 RPS -->
    <ConstantThroughputTimer guiclass="ConstantThroughputTimerGui" testclass="ConstantThroughputTimer" testname="Throughput Ti
">
      <stringProp name="ConstantThroughputTimer.throughput">700</stringProp>
      <intProp name="ConstantThroughputTimer.calcMode">1</intProp> <!-- Mode 1: All active threads -->
    </ConstantThroughputTimer>
    <hashTree/>
```

- Run the test with 700 RPS and save the result in file: result-700rps-stress.jtl

```
root@ip-172-31-15-56:~/tests# jmeter -n -t test-throughput-stress.jmx -l result-700rps-stress.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed i
ure release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed i
ure release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed i
ure release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed i
ure release
Creating summariser <summary>
Created the tree successfully using test-throughput-stress.jmx
Starting standalone test @ 2025 Mar 14 14:21:48 UTC (1741962108959)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
```

- View the result

```
root@ip-172-31-15-56:~/tests# tail -n 10 result-700rps-stress.jtl
1741962386937,1,HTTP Request,200,OK,Throughput Stress Test Group 1-52,text,true,,857,108,100,100,http://172.31.1.46/,1,0,0
1741962386937,1,HTTP Request,200,OK,Throughput Stress Test Group 1-64,text,true,,857,108,100,100,http://172.31.1.46/,1,0,0
1741962386937,1,HTTP Request,200,OK,Throughput Stress Test Group 1-58,text,true,,857,108,100,100,http://172.31.1.46/,1,0,0
1741962386937,1,HTTP Request,200,OK,Throughput Stress Test Group 1-95,text,true,,857,108,100,100,http://172.31.1.46/,1,0,1
1741962386939,2,HTTP Request,200,OK,Throughput Stress Test Group 1-89,text,true,,857,108,100,100,http://172.31.1.46/,2,0,1
1741962386941,1,HTTP Request,200,OK,Throughput Stress Test Group 1-83,text,true,,857,108,100,100,http://172.31.1.46/,1,0,1
1741962386942,0,HTTP Request,200,OK,Throughput Stress Test Group 1-46,text,true,,857,108,100,100,http://172.31.1.46/,0,0,0
1741962386943,1,HTTP Request,200,OK,Throughput Stress Test Group 1-34,text,true,,857,108,100,100,http://172.31.1.46/,1,0,0
1741962386943,1,HTTP Request,200,OK,Throughput Stress Test Group 1-40,text,true,,857,108,100,100,http://172.31.1.46/,1,0,1
1741962386944,1,HTTP Request,200,OK,Throughput Stress Test Group 1-28,text,true,,857,108,100,100,http://172.31.1.46/,1,0,0
```

- CPU usage

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+
  674 www-data  20   0   12940   5720   4224 S  13.7   0.6  26:13.34
  670 root      20   0   11156   1752    768 S   0.0   0.2   0:00.00
```

- Throughput = 1000
  - Note:
    - Update the Constant Throughput Timer to 1000 RPS
    - Adjust the Number of Threads (Users): Ensure enough users are available to sustain 1000 RPS. We may need to increase this to 200 users for better distribution.

- Update the test file

```
<stringProp name="ThreadGroup.num_threads">200</stringProp>  <!-- Adjusted to support higher RPS -->
<stringProp name="ThreadGroup.ramp_time">10</stringProp> <!-- Smooth ramp-up -->
<boolProp name="ThreadGroup.scheduler">false</boolProp>
<stringProp name="ThreadGroup.duration"></stringProp>
<stringProp name="ThreadGroup.delay"></stringProp>
/ThreadGroup>
hashTree>
<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="HTTP Request" enabled="
  <stringProp name="HTTPSampler.domain">172.31.1.46</stringProp>
  <stringProp name="HTTPSampler.port">80</stringProp>
  <stringProp name="HTTPSampler.protocol">http</stringProp>
  <stringProp name="HTTPSampler.path">/</stringProp>
  <stringProp name="HTTPSampler.method">GET</stringProp>
</HTTPSamplerProxy>
<hashTree>
  <ConstantThroughputTimer guiclass="ConstantThroughputTimerGui" testclass="ConstantThroughputTimer" testname
enabled="true">
    <stringProp name="ConstantThroughputTimer.throughput">1000</stringProp> <!-- Target 1000 RPS -->
    <stringProp name="ConstantThroughputTimer.calcMode">1</stringProp>
  </ConstantThroughputTimer>
</hashTree>
```

- Run the test with 1000 RPS and save the result in file: result-1000rps-stress.jtl

```
root@ip-172-31-15-56:~/tests# jmeter -n -t test-throughput-stress.jmx -l result-1000rps-stress.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
Creating summariser <summary>
Created the tree successfully using test-throughput-stress.jmx
Starting standalone test @ 2025 Mar 14 14:37:35 UTC (1741963055158)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
```

- Stopped the testing after 6 mins

```
summary +  99464 in 00:00:30 = 3315.6/s Avg:     5 Min:     0 Max:   131 Err:     0 (0.00%) Active: 200 Started: 200 Finished: 0
summary = 795258 in 00:04:24 = 3006.7/s Avg:    11 Min:     0 Max:   618 Err:     0 (0.00%)
summary +  99644 in 00:00:30 = 3321.5/s Avg:     4 Min:     0 Max:   149 Err:     0 (0.00%) Active: 200 Started: 200 Finished: 0
summary = 894902 in 00:04:55 = 3038.7/s Avg:    10 Min:     0 Max:   618 Err:     0 (0.00%)
summary +  99262 in 00:00:30 = 3309.3/s Avg:     5 Min:     0 Max:   131 Err:     0 (0.00%) Active: 200 Started: 200 Finished: 0
summary = 994164 in 00:05:24 = 3063.7/s Avg:    10 Min:     0 Max:   618 Err:     0 (0.00%)
summary +  98756 in 00:00:30 = 3291.9/s Avg:     6 Min:     0 Max:   217 Err:     0 (0.00%) Active: 200 Started: 200 Finished: 0
summary = 1092920 in 00:05:54 = 3083.0/s Avg:     9 Min:     0 Max:   618 Err:     0 (0.00%)
summary +  99205 in 00:00:30 = 3306.7/s Avg:     5 Min:     0 Max:   200 Err:     0 (0.00%) Active: 200 Started: 200 Finished: 0
summary = 1192125 in 00:06:24 = 3100.5/s Avg:     9 Min:     0 Max:   618 Err:     0 (0.00%)
^C
root@ip-172-31-15-56:~/tests#
```

- Check the result

```
root@ip-172-31-15-56:~/tests# tail -n 10 result-1000rps-stress.jtl
1741963453297,3,HTTP Request,200,OK,Throughput Stress Test Group 1-147,text,true,,857,108,200,200,http://172.31.1.46/,3,0,2
1741963453296,4,HTTP Request,200,OK,Throughput Stress Test Group 1-25,text,true,,857,108,200,200,http://172.31.1.46/,4,0,2
1741963453297,4,HTTP Request,200,OK,Throughput Stress Test Group 1-24,text,true,,857,108,200,200,http://172.31.1.46/,3,0,1
1741963453297,4,HTTP Request,200,OK,Throughput Stress Test Group 1-190,text,true,,857,108,200,200,http://172.31.1.46/,4,0,1
1741963453299,2,HTTP Request,200,OK,Throughput Stress Test Group 1-65,text,true,,857,108,200,200,http://172.31.1.46/,2,0,1
1741963453297,4,HTTP Request,200,OK,Throughput Stress Test Group 1-178,text,true,,857,108,200,200,http://172.31.1.46/,4,0,1
1741963453297,4,HTTP Request,200,OK,Throughput Stress Test Group 1-27,text,true,,857,108,200,200,http://172.31.1.46/,4,0,1
1741963453297,4,HTTP Request,200,OK,Throughput Stress Test Group 1-44,text,true,,857,108,200,200,http://172.31.1.46/,4,0,1
1741963453297,4,HTTP Request,200,OK,Throughput Stress Test Group 1-195,text,true,,857,108,200,200,http://172.31.1.46/,4,0,1
1741963453297,4,HTTP Request,200,OK,Throughput Stress Test Group 1-179,text,true,,857,108,200,200,http://172.31.1.46/,4,0,1
```

- CPU usage: went up to 44%

```
MiB Swap:      0.0 total,      0.0 free,      0.0 used.    631.9 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+
    674 www-data  20   0   12940   5720   4224 R  44.0   0.6  29:11.57
    670 root      20   0   11156   1752    768 S   0.0   0.2   0:00.00
```

- Findings

  The 1000 RPS throughput stress test showed that the system maintained stable performance with an average throughput of ~3100 RPS, low response times (Avg: 9-18 ms, Max: 618 ms), and a 0% error rate. CPU usage peaked at 44%,

and memory usage remained minimal at 0.6%, indicating that the system had sufficient capacity to handle the load. With 200 concurrent users, the system continued processing requests efficiently without noticeable bottlenecks. Given these results, the system appears capable of handling even higher loads, and future tests should explore increasing the throughput to 1200 RPS to identify performance limits more accurately.

- Throughput = 1200
  - Note:
    - Update the Constant Throughput Timer to 1200 RPS
    - Adjust the Number of Threads (Users): The number of users increased to 500 to sustain the higher throughput.
  - Update the testing file

```
      <boolProp name="LoopController.continue_forever">true</boolProp>
      <stringProp name="LoopController.loops">-1</stringProp> <!-- Infinite loop -->
    </elementProp>
    <stringProp name="ThreadGroup.num_threads">500</stringProp> <!-- Infinite loop -->
    <stringProp name="ThreadGroup.ramp_time">10</stringProp>
    <boolProp name="ThreadGroup.scheduler">false</boolProp>
  </ThreadGroup>
  <hashTree>
    <ConstantThroughputTimer guiclass="ConstantThroughputTimerGui" testclass="ConstantThroughputTimer" t
enabled="true">
      <stringProp name="ConstantThroughputTimer.throughput">1200</stringProp>
    </ConstantThroughputTimer>
    <HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="HTTP Request"
      <stringProp name="HTTPSampler.domain">172.31.1.46</stringProp>
```

  - Run the test with 1200 RPS and save the result in file: result-1200rps-stress.jt

```
root@ip-172-31-15-56:~/tests# jmeter -n -t test-throughput-stress.jmx -l result-1200rps-stress.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
Creating summariser <summary>
Created the tree successfully using test-throughput-stress.jmx
Starting standalone test @ 2025 Mar 14 14:55:01 UTC (1741964101074)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary +      1 in 00:00:01 =    1.4/s Avg:    79 Min:    79 Max:    79 Err:     0 (0.00%) Active: 16 Started: 16 Finished: 0
```

  - Stopped the test after 10 mins

```
hed: 0
summary = 1520996 in 00:08:12 = 3090.6/s Avg:    18 Min:     0 Max:   983 Err:     0 (0.00%)
summary + 103141 in 00:00:30 = 3438.0/s Avg:    21 Min:     0 Max:   641 Err:     0 (0.00%) Active: 240 Started: 240 Finis
hed: 0
summary = 1624137 in 00:08:42 = 3110.5/s Avg:    18 Min:     0 Max:   983 Err:     0 (0.00%)
summary + 103996 in 00:00:30 = 3467.0/s Avg:    18 Min:     0 Max:   669 Err:     0 (0.00%) Active: 240 Started: 240 Finis
hed: 0
summary = 1728133 in 00:09:12 = 3129.8/s Avg:    18 Min:     0 Max:   983 Err:     0 (0.00%)
summary + 104657 in 00:00:30 = 3488.9/s Avg:    20 Min:     0 Max:   778 Err:     0 (0.00%) Active: 240 Started: 240 Finis
hed: 0
summary = 1832790 in 00:09:42 = 3148.3/s Avg:    19 Min:     0 Max:   983 Err:     0 (0.00%)
summary + 103567 in 00:00:30 = 3452.2/s Avg:    19 Min:     0 Max:   668 Err:     0 (0.00%) Active: 240 Started: 240 Finis
hed: 0
summary = 1936357 in 00:10:12 = 3163.2/s Avg:    19 Min:     0 Max:   983 Err:     0 (0.00%)
```

- Check the result

```
root@ip-172-31-15-56:~/tests# tail -n 10 result-1200rps-stress.jtl
1741965272522,17,HTTP Request,200,OK,Throughput Stress Test Group 1-3,text,true,,857,108,240,240,http://172.31.1.46/,17,0
13
1741965272529,10,HTTP Request,200,OK,Throughput Stress Test Group 1-187,text,true,,857,108,240,240,http://172.31.1.46/,10
0,6
1741965272521,18,HTTP Request,200,OK,Throughput Stress Test Group 1-42,text,true,,857,108,240,240,http://172.31.1.46/,18,
,12
1741965272511,29,HTTP Request,200,OK,Throughput Stress Test Group 1-182,text,true,,857,108,240,240,http://172.31.1.46/,29
0,22
1741965272521,20,HTTP Request,200,OK,Throughput Stress Test Group 1-92,text,true,,857,108,240,240,http://172.31.1.46/,20,
,17
1741965272532,9,HTTP Request,200,OK,Throughput Stress Test Group 1-119,text,true,,857,108,240,240,http://172.31.1.46/,9,0
6
1741965272530,11,HTTP Request,200,OK,Throughput Stress Test Group 1-230,text,true,,857,108,240,240,http://172.31.1.46/,11
0,9
1741965272520,21,HTTP Request,200,OK,Throughput Stress Test Group 1-15,text,true,,857,108,240,240,http://172.31.1.46/,21,
,12
1741965272206,11,HTTP Request,200,OK,Throughput Stress Test Group 1-138,text,true,,857,108,240,240,http://172.31.1.46/,11
0,10
1741965272521,21,HTTP Request,200,OK,Throughput Stress Test Group 1-77,text,true,,857,108,240,240,http://172.31.1.46/,21,
```

- CPU Usage:

```
op - 15:13:59 up  7:44,  1 user,  load average: 0.42, 0.43, 0.26
asks:   2 total,   1 running,   1 sleeping,   0 stopped,   0 zombie
Cpu(s): 11.9 us, 21.8 sy,  0.0 ni, 34.0 id,  0.0 wa,  0.0 hi, 31.2 si,  1.1
iB Mem :    957.4 total,     74.6 free,    327.0 used,    718.3 buff/cache
iB Swap:      0.0 total,      0.0 free,      0.0 used.    630.4 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+
  674 www-data  20   0   12940   5720   4224 R  46.5   0.6  33:45.79
  670 root      20   0   11156   1752    768 S   0.0   0.2   0:00.00
```

- Findings: The 1200 RPS stress test showed that the system handled the load efficiently without errors or major performance degradation. The observed **peak CPU usage was 46.5%**, and **memory usage remained low at 0.6%**, indicating that the system was not resource-constrained. The test started with **1611.6 RPS** in the first 12 seconds and later stabilized, reaching a peak of **3488.9 RPS**. Response times averaged between **17–20 ms**, with a **maximum spike of 983 ms**, but no errors were recorded throughout the test. The system remained stable, and **nginx logs showed no failures**, confirming that it can sustain at least **1200 RPS** without issue. Given these results, will proceed with further stress testing at **1500 RPS or higher** to determine the true upper limit of the system.

- Throughput = 2000
  - Note:
    - Update the Constant Throughput Timer to 2000 RPS
    - Adjust the Number of Threads (Users): The number of users increased to 5000 to sustain the higher throughput.

- Update the test file

```
el" testclass="LoopController" testname="Loop Controller" enabled="true">
    <boolProp name="LoopController.continue_forever">false</boolProp>
    <stringProp name="LoopController.loops">-1</stringProp> <!-- Continuous execution -->
  </elementProp>
  <stringProp name="ThreadGroup.num_threads">5000</stringProp> <!-- 1000 concurrent users -->
  <stringProp name="ThreadGroup.ramp_time">10</stringProp> <!-- Ramp-up time of 10 seconds -->
  <boolProp name="ThreadGroup.scheduler">false</boolProp>
  <stringProp name="ThreadGroup.duration"></stringProp>
  <stringProp name="ThreadGroup.delay"></stringProp>
</ThreadGroup>
<hashTree>
  <ConstantThroughputTimer guiclass="ConstantThroughputTimerGui" testclass="ConstantThroughputTimm
stname="Throughput Timer" enabled="true">
    <stringProp name="ConstantThroughputTimer.throughput">2000</stringProp> <!-- Targeting 1500 RR

    <stringProp name="ConstantThroughputTimer.calcMode">2</stringProp> <!-- All active threads uss
ame timer -->
  </ConstantThroughputTimer>
  <hashTree/>
```

- Run the test with 2000 RPS and save the result in file: result-2000rps-stress.jt

```
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be remo
ved in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be remo
ved in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be remo
ved in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be remo
ved in a future release
Creating summariser <summary>
Created the tree successfully using test-throughput-stress.jmx
Starting standalone test @ 2025 Mar 14 19:39:27 UTC (1741981167234)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary +   1013 in 00:00:02 =  428.7/s Avg:    56 Min:     2 Max:   406 Err:     0 (0.00%) Active: 238
```

- System experiences a stall

```
report-static       report-throughput-50RPS   result-700rps-stress.jtl   test-size.jmx        test.jmx
root@ip-172-31-15-56:~/tests# jmeter -n -t test-throughput-stress.jmx -l result-2000rps-stress.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
Creating summariser <summary>
Created the tree successfully using test-throughput-stress.jmx
Starting standalone test @ 2025 Mar 14 20:37:33 UTC (1741984653327)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary +      1 in 00:00:02 =    0.7/s Avg:   443 Min:   443 Max:   443 Err:     0 (0.00%) Active: 440 Started: 440 Finished: 0
summary +  44485 in 00:00:25 = 1794.9/s Avg:    58 Min:     1 Max:  3411 Err:     0 (0.00%) Active: 1409 Started: 1409 Finished:
summary =  44486 in 00:00:26 = 1692.4/s Avg:    58 Min:     1 Max:  3411 Err:     0 (0.00%)
```

- CPU usage: dropped to 0%

```
top - 20:46:46 up 13:17,  2 users,  load average: 0.00, 0.00, 0.02
Tasks:   2 total,   0 running,   2 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni, 94.6 id,  0.0 wa,  0.0 hi,  0.0 si,  5.4 st
MiB Mem :    957.4 total,     93.1 free,    326.2 used,    701.8 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.    631.2 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    670 root      20   0   11156   1752    768 S   0.0   0.2   0:00.00 nginx
    674 www-data  20   0   12940   5720   4224 S   0.0   0.6  76:01.13 nginx
```

- Analyze:
  - check if there are **worker process limits**

```
root@ip-172-31-1-46:~# cat /etc/nginx/nginx.conf | grep worker
worker_processes auto;
        worker_connections 768;
root@ip-172-31-1-46:~#
```

- The 2000 RPS test with 5000 users stalled, with CPU usage dropping to 0%, indicating a system-level bottleneck rather than an overload. Possible causes include Nginx hitting its connection limit, as the worker_connections setting is still at 768, which may not be sufficient for high concurrency. Additionally, open file descriptor limits (ulimit -n) might be restricting the number of simultaneous connections, preventing JMeter from fully generating the load. Another potential issue is TCP connection exhaustion, where the backlog of unprocessed requests exceeds system limits (net.core.somaxconn and tcp_max_syn_backlog). In the next iteration, I will increase worker connections in Nginx (worker_connections 10000), raise open file limits (ulimit -n 65535), and tune TCP backlog settings to handle more concurrent requests. Additionally, I will increase JMeter's heap memory to ensure it can generate high RPS loads without stalling. If the test still stalls, I will gradually scale down RPS (e.g., 3000 RPS) to determine the system's actual throughput capacity before reaching its limit.