# Final Project Report

April 26, 2022


## Catalyst Team

Linni Cai, Shengnan You, Furong Tian

Northeastern University


## Overview

In this assignment, we have three consumers, two for skiers, one for resorts, but they share the same data from the queue. We chose RMQ publish/subscribe pattern, the server

publishes the data to the queue, the consumer subscribes the data from the queue. The whole workflow is that the client serves as producer, it sends plenty of posts to the server, the server delivers results to consumers.

## Github

https://github.com/linni-cai-lc/CS6650_Final_Project/

## Database Design

1. We used Redis for data storage, utilized the key/value structure to write and read efficiently. Each consumer stores the data in its own instance's Redis storage.
2. Skier Total Vertical Data Pattern:
    a. the key is TOTAL_VERTICAL
    b. the field is resortID-seasonID-dayID-skierID
    c. the value is the total vertical integer
3. Skier Total Vertical Result List Data Pattern:
    a. the key is skierID
    b. the field is seasonID
    c. the value is the total vertical integer
4. Resort Num Of Skiers Data Pattern:
    a. the key is NUM_SKIERS
    b. the field is resortID-seasonID-dayID
    c. the value is the number of skiers

## Client Preparation

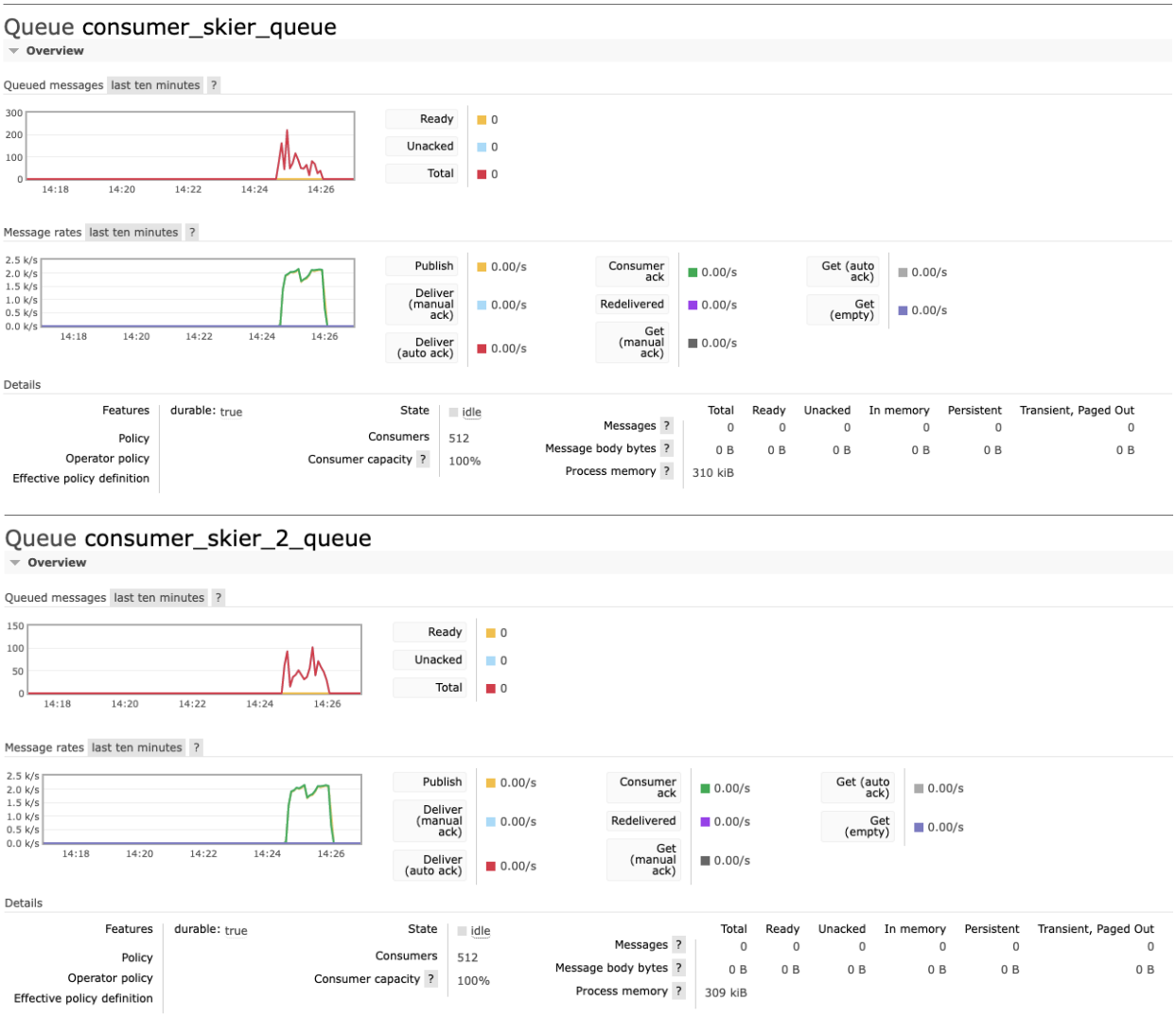
The experiments are based on 20000 skiers, 40 lifts. Overall the queue size is below 500. Our mitigation strategy continues to use the circuit breaker to limit the speed of POST generation.

## EC2 Instance

1. One Linux instance running the server

   a. provides with the skier API functionality: POST & GET
   b. publish messages to RMQ queue
2. Three Linux instances running the consumer
   a. one for skier total vertical
   b. one for skier total vertical result list
   c. one for resort
   d. run Redis and store received messages
   e. subscribe to the queues and consume messages
3. One Ubuntu instance running the RMQ server
   a. owns the queue and store messages
   b. publish the messages

# RMQ Results



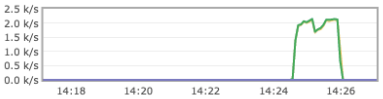
## Queue consumer_resort_queue

**Overview**

Queued messages last ten minutes ?


| | |
|---|---|
| Ready | 0 |
| Unacked | 0 |
| Total | 0 |

Message rates last ten minutes ?




| | | | | | |
|---|---|---|---|---|---|
| Publish | 0.00/s | Consumer ack | 0.00/s | Get (auto ack) | 0.00/s |
| Deliver (manual ack) | 0.00/s | Redelivered | 0.00/s | Get (empty) | 0.00/s |
| Deliver (auto ack) | 0.00/s | Get (manual ack) | 0.00/s | | |

Details

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Features | durable: true | State | idle | | Total | Ready | Unacked | In memory | Persistent | Transient, Paged Out |
| Policy | | Consumers | 512 | Messages ? | 0 | 0 | 0 | 0 | 0 | 0 |
| Operator policy | | Consumer capacity ? | 100% | Message body bytes ? | 0 B | 0 B | 0 B | 0 B | 0 B | 0 B |
| Effective policy definition | | | | Process memory ? | 309 kiB | | | | | |

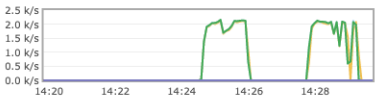
## Queue consumer_skier_queue

**Overview**

Queued messages last ten minutes ?


| | |
|---|---|
| Ready | 0 |
| Unacked | 0 |
| Total | 0 |

Message rates last ten minutes ?




| | | | | | |
|---|---|---|---|---|---|
| Publish | 0.00/s | Consumer ack | 0.00/s | Get (auto ack) | 0.00/s |
| Deliver (manual ack) | 0.00/s | Redelivered | 0.00/s | Get (empty) | 0.00/s |
| Deliver (auto ack) | 0.00/s | Get (manual ack) | 0.00/s | | |

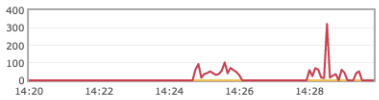
Details

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Features | durable: true | State | idle | | Total | Ready | Unacked | In memory | Persistent | Transient, Paged Out |
| Policy | | Consumers | 512 | Messages ? | 0 | 0 | 0 | 0 | 0 | 0 |
| Operator policy | | Consumer capacity ? | 100% | Message body bytes ? | 0 B | 0 B | 0 B | 0 B | 0 B | 0 B |
| Effective policy definition | | | | Process memory ? | 310 kiB | | | | | |

## Queue consumer_skier_2_queue

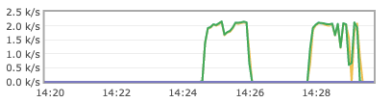
**Overview**

Queued messages last ten minutes ?




| | |
|---|---|
| Ready | 0 |
| Unacked | 0 |
| Total | 0 |

Message rates last ten minutes ?




| | | | | | |
|---|---|---|---|---|---|
| Publish | 0.00/s | Consumer ack | 0.00/s | Get (auto ack) | 0.00/s |
| Deliver (manual ack) | 0.00/s | Redelivered | 0.00/s | Get (empty) | 0.00/s |
| Deliver (auto ack) | 0.00/s | Get (manual ack) | 0.00/s | | |

Details

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Features | durable: true | State | idle | | Total | Ready | Unacked | In memory | Persistent | Transient, Paged Out |
| Policy | | Consumers | 512 | Messages ? | 0 | 0 | 0 | 0 | 0 | 0 |
| Operator policy | | Consumer capacity ? | 100% | Message body bytes ? | 0 B | 0 B | 0 B | 0 B | 0 B | 0 B |
| Effective policy definition | | | | Process memory ? | 309 kiB | | | | | |

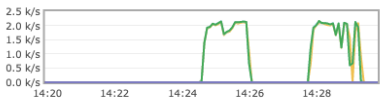
## Queue consumer_resort_queue

**Overview**

Queued messages last ten minutes ?


| | |
|---|---|
| Ready | 0 |
| Unacked | 0 |
| Total | 0 |

Message rates last ten minutes ?




| | | | | | |
|---|---|---|---|---|---|
| Publish | 0.00/s | Consumer ack | 0.00/s | Get (auto ack) | 0.00/s |
| Deliver (manual ack) | 0.00/s | Redelivered | 0.00/s | Get (empty) | 0.00/s |
| Deliver (auto ack) | 0.00/s | Get (manual ack) | 0.00/s | | |

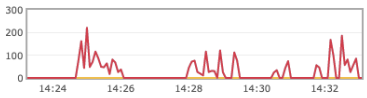
Details

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Features | durable: true | State | idle | | Total | Ready | Unacked | In memory | Persistent | Transient, Paged Out |
| Policy | | Consumers | 512 | Messages ? | 0 | 0 | 0 | 0 | 0 | 0 |
| Operator policy | | Consumer capacity ? | 100% | Message body bytes ? | 0 B | 0 B | 0 B | 0 B | 0 B | 0 B |
| Effective policy definition | | | | Process memory ? | 309 kiB | | | | | |

## Queue consumer_skier_queue

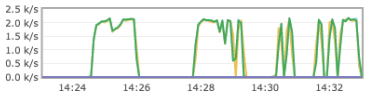
**Overview**

Queued messages last ten minutes ?




| | |
|---|---|
| Ready | 0 |
| Unacked | 0 |
| Total | 0 |

Message rates last ten minutes ?




| | | | | | |
|---|---|---|---|---|---|
| Publish | 0.00/s | Consumer ack | 0.00/s | Get (auto ack) | 0.00/s |
| Deliver (manual ack) | 0.00/s | Redelivered | 0.00/s | Get (empty) | 0.00/s |
| Deliver (auto ack) | 0.00/s | Get (manual ack) | 0.00/s | | |

Details

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Features | durable: true | State | running | | Total | Ready | Unacked | In memory | Persistent | Transient, Paged Out |
| Policy | | Consumers | 512 | Messages ? | 0 | 0 | 0 | 0 | 0 | 0 |
| Operator policy | | Consumer capacity ? | 100% | Message body bytes ? | 0 B | 0 B | 0 B | 0 B | 0 B | 0 B |
| Effective policy definition | | | | Process memory ? | 1.2 MiB | | | | | |

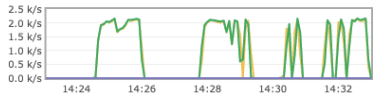
## Queue consumer_skier_2_queue

**Overview**

Queued messages last ten minutes ?


| | |
|---|---|
| Ready | 0 |
| Unacked | 0 |
| Total | 0 |

Message rates last ten minutes ?




| | | | | | |
|---|---|---|---|---|---|
| Publish | 0.00/s | Consumer ack | 0.00/s | Get (auto ack) | 0.00/s |
| Deliver (manual ack) | 0.00/s | Redelivered | 0.00/s | Get (empty) | 0.00/s |
| Deliver (auto ack) | 0.00/s | Get (manual ack) | 0.00/s | | |

Details

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Features | durable: true | State | idle | | Total | Ready | Unacked | In memory | Persistent | Transient, Paged Out |
| Policy | | Consumers | 512 | Messages ? | 0 | 0 | 0 | 0 | 0 | 0 |
| Operator policy | | Consumer capacity ? | 100% | Message body bytes ? | 0 B | 0 B | 0 B | 0 B | 0 B | 0 B |
| Effective policy definition | | | | Process memory ? | 309 kiB | | | | | |

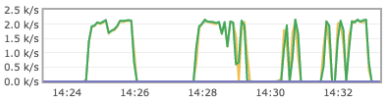
## Queue consumer_resort_queue

**Overview**

Queued messages last ten minutes ?


| | |
|---|---|
| Ready | 0 |
| Unacked | 0 |
| Total | 0 |

Message rates last ten minutes ?

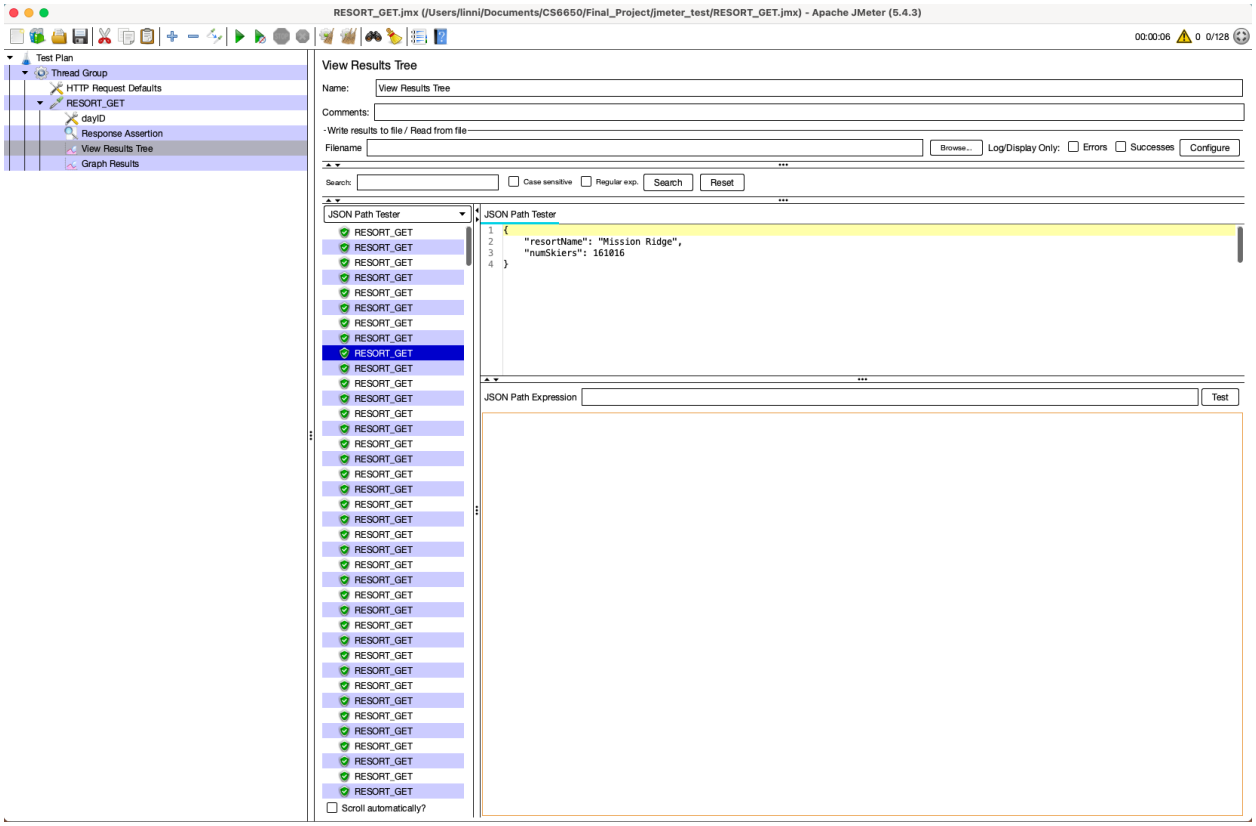


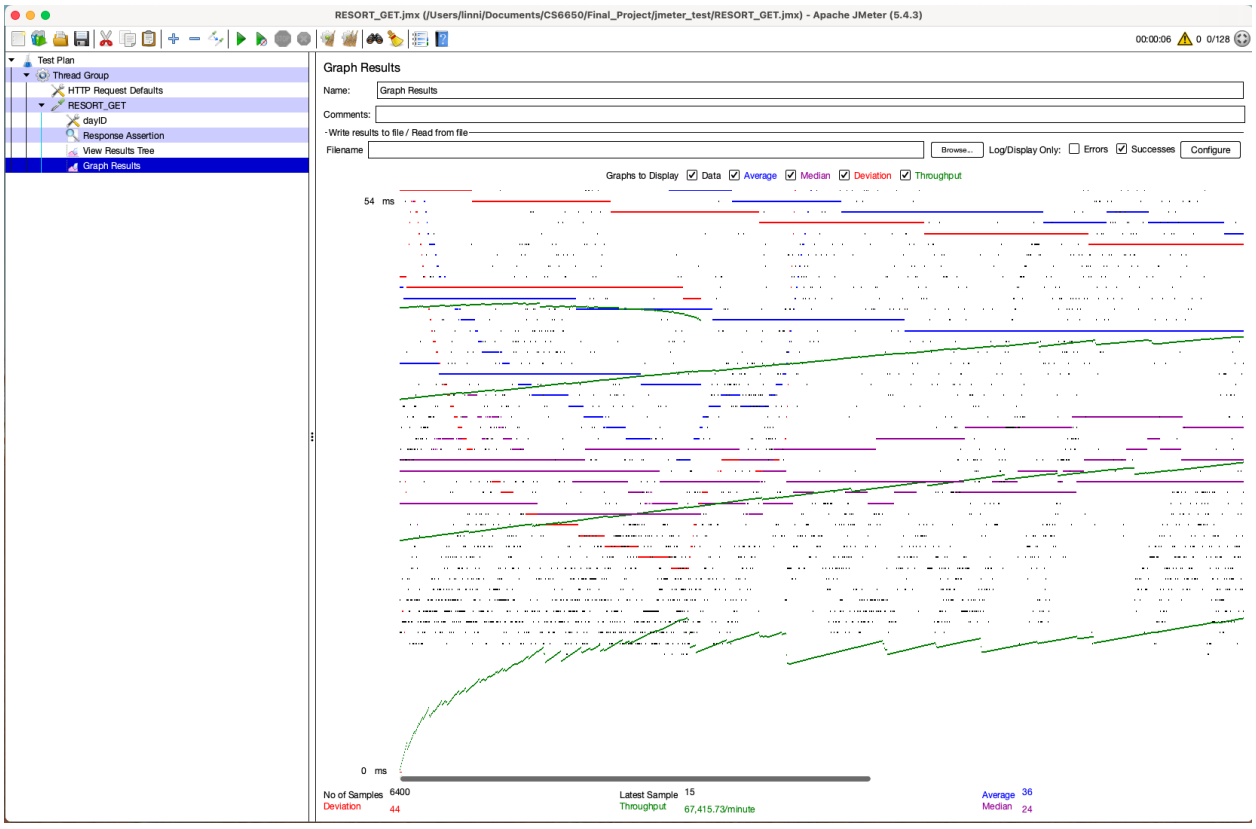

| | | | | | |
|---|---|---|---|---|---|
| Publish | 0.00/s | Consumer ack | 0.00/s | Get (auto ack) | 0.00/s |
| Deliver (manual ack) | 0.00/s | Redelivered | 0.00/s | Get (empty) | 0.00/s |
| Deliver (auto ack) | 0.00/s | Get (manual ack) | 0.00/s | | |

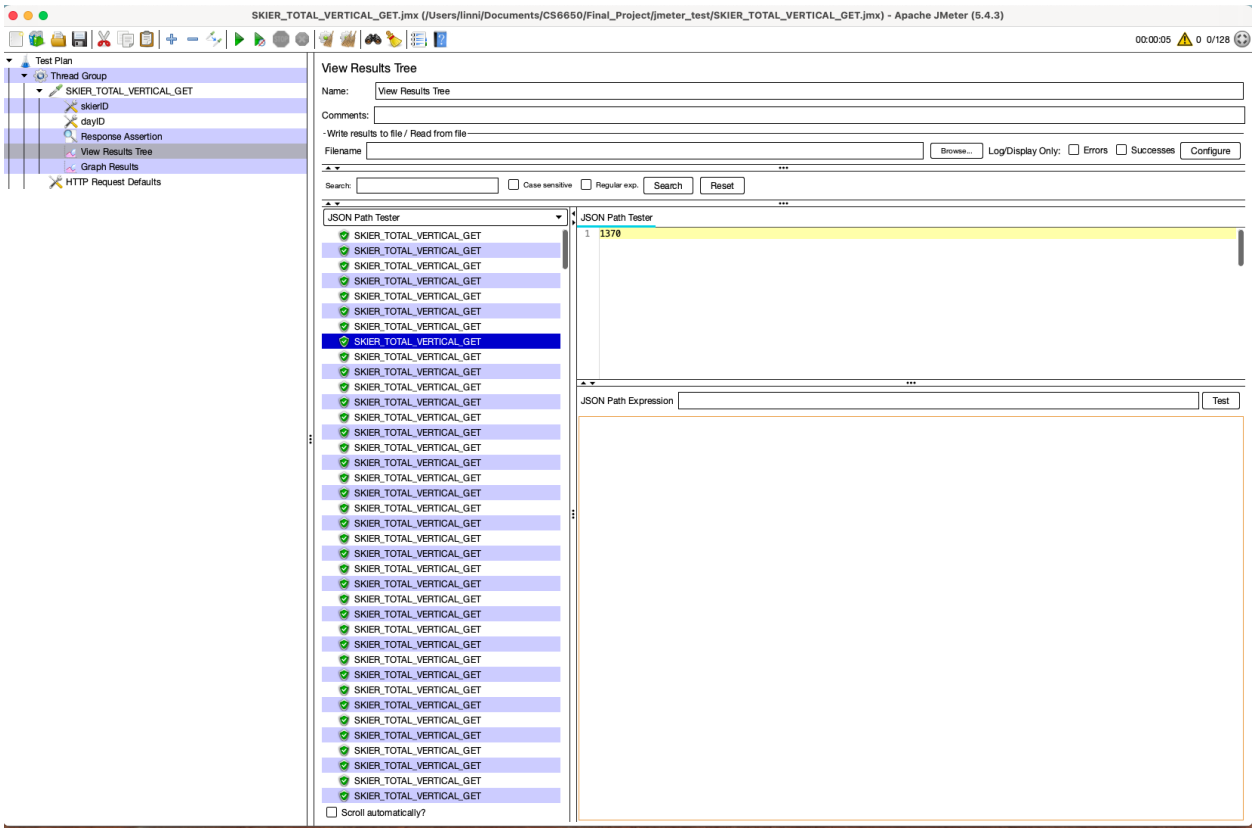
Details

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Features | durable: true | State | idle | | Total | Ready | Unacked | In memory | Persistent | Transient, Paged Out |
| Policy | | Consumers | 512 | Messages ? | 0 | 0 | 0 | 0 | 0 | 0 |
| Operator policy | | Consumer capacity ? | 100% | Message body bytes ? | 0 B | 0 B | 0 B | 0 B | 0 B | 0 B |
| Effective policy definition | | | | Process memory ? | 309 kiB | | | | | |

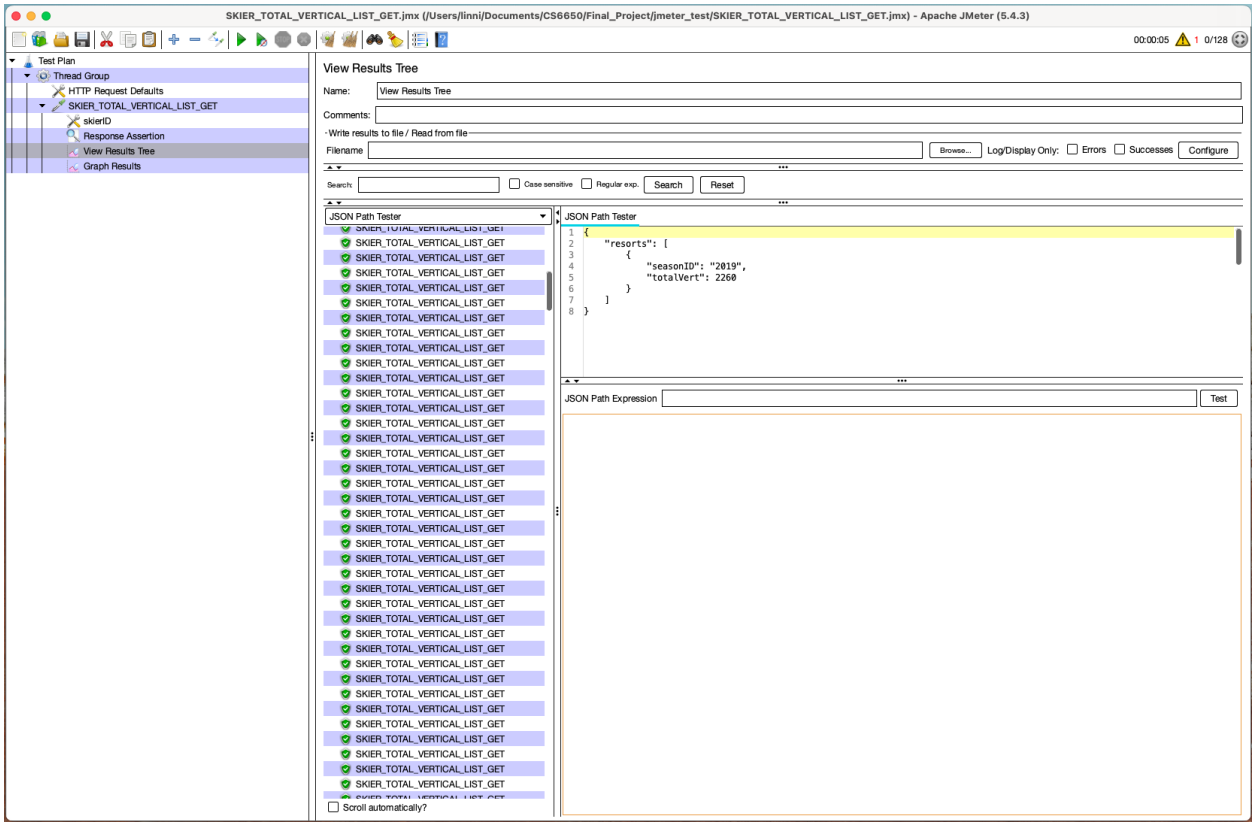
## JMeter Results

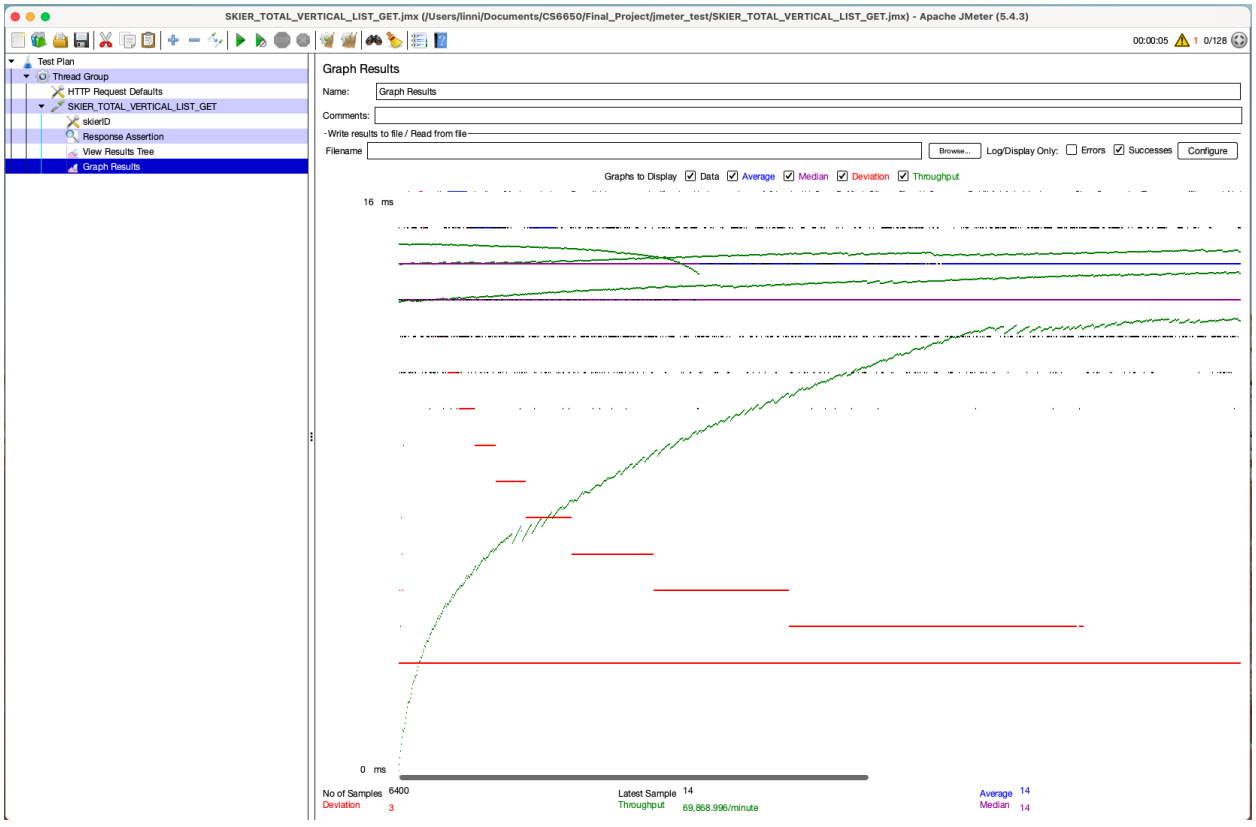

RESORT_GET.jmx (/Users/linni/Documents/CS6650/Final_Project/jmeter_test/RESORT_GET.jmx) - Apache JMeter (5.4.3)
00:00:06 0 0/128
Test Plan
Thread Group
HTTP Request Defaults
RESORT_GET
dayID
Response Assertion
View Results Tree
Graph Results
View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
JSON Path Tester
RESORT_GET
{
"resortName": "Mission Ridge",
"numSkiers": 161016
}
JSON Path Expression
Test
Scroll automatically?

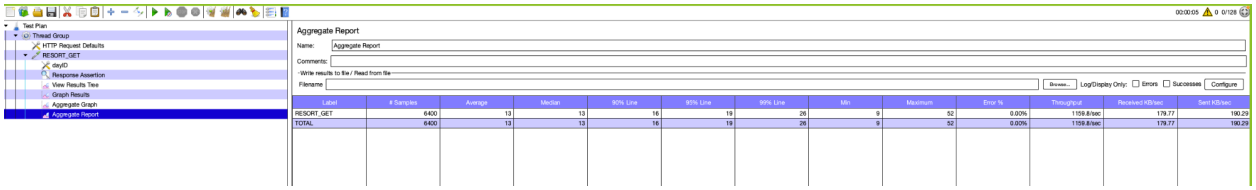

RESORT_GET.jmx (/Users/linni/Documents/CS6650/Final_Project/jmeter_test/RESORT_GET.jmx) - Apache JMeter (5.4.3)
00:00:06 0 0/128
Test Plan
Thread Group
HTTP Request Defaults
RESORT_GET
dayID
Response Assertion
View Results Tree
Graph Results
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Graphs to Display Data Average Median Deviation Throughput
54 ms
0 ms
No of Samples 6400
Deviation 44
Latest Sample 15
Throughput 67,415.73/minute
Average 36
Median 24

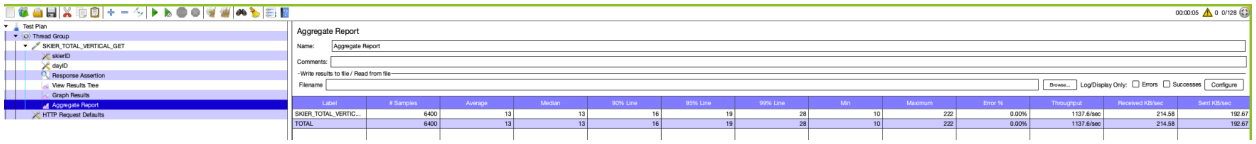

SKIER_TOTAL_VERTICAL_GET.jmx (/Users/linni/Documents/CS6650/Final_Project/jmeter_test/SKIER_TOTAL_VERTICAL_GET.jmx) - Apache JMeter (5.4.3)
00:00:05 0 0/128
Test Plan
Thread Group
SKIER_TOTAL_VERTICAL_GET
skierID
dayID
Response Assertion
View Results Tree
Graph Results
HTTP Request Defaults
View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Search:
Case sensitive
Regular exp.
Search
Reset
JSON Path Tester
SKIER_TOTAL_VERTICAL_GET
1370
JSON Path Expression
Test
Scroll automatically?

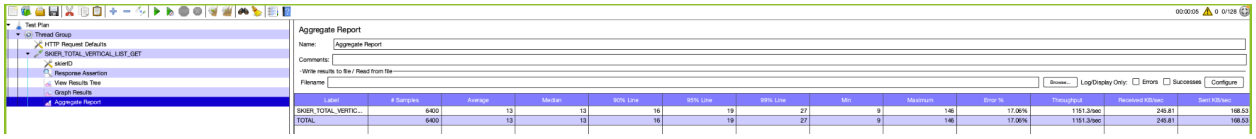

SKIER_TOTAL_VERTICAL_GET.jmx (/Users/linni/Documents/CS6650/Final_Project/jmeter_test/SKIER_TOTAL_VERTICAL_GET.jmx) - Apache JMeter (5.4.3)
00:00:05 0 0/128
Test Plan
Thread Group
SKIER_TOTAL_VERTICAL_GET
skierID
dayID
Response Assertion
View Results Tree
Graph Results
HTTP Request Defaults
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Graphs to Display Data Average Median Deviation Throughput
21 ms
0 ms
No of Samples 6400
Deviation 5
Latest Sample 13
Throughput 71,097.945/minute
Average 15
Median 14


SKIER_TOTAL_VERTICAL_LIST_GET.jmx (/Users/linni/Documents/CS6650/Final_Project/jmeter_test/SKIER_TOTAL_VERTICAL_LIST_GET.jmx) - Apache JMeter (5.4.3)
00:00:05 1 0/128
Test Plan
Thread Group
HTTP Request Defaults
SKIER_TOTAL_VERTICAL_LIST_GET
skierID
Response Assertion
View Results Tree
Graph Results
View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse... Log/Display Only: Errors Successes Configure
Search: Case sensitive Regular exp. Search Reset
JSON Path Tester
SKIER_TOTAL_VERTICAL_LIST_GET
{
"resorts": [
{
"seasonID": "2019",
"totalVert": 2260
}
]
}
JSON Path Expression Test
Scroll automatically?


SKIER_TOTAL_VERTICAL_LIST_GET.jmx (/Users/linni/Documents/CS6650/Final_Project/jmeter_test/SKIER_TOTAL_VERTICAL_LIST_GET.jmx) - Apache JMeter (5.4.3)
00:00:05 1 0/128
Test Plan
Thread Group
HTTP Request Defaults
SKIER_TOTAL_VERTICAL_LIST_GET
skierID
Response Assertion
View Results Tree
Graph Results
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse... Log/Display Only: Errors Successes Configure
Graphs to Display Data Average Median Deviation Throughput
16 ms
0 ms
No of Samples 6400
Deviation 3
Latest Sample 14
Throughput 69,868.996/minute
Average 14
Median 14







# Statistics

## I. Day 1

------ Client Post Statistics ------

------ PART 1 ------
number of successful requests sent: 160420
number of unsuccessful requests: 0
the total run time for all phases to complete: 96045
the total throughput in requests per second: 1000

------ PART 2 ------
mean response time (millisecs): 115
median response time (millisecs): 32
throughput: 1000
p99 (99th percentile) response time: 2021
min response time (millisecs): 12
max response time (millisecs): 9601

## II. Day 2

------ Client Post Statistics ------

------ PART 1 ------
number of successful requests sent: 160420
number of unsuccessful requests: 0
the total run time for all phases to complete: 96045
the total throughput in requests per second: 1000

------ PART 2 ------
mean response time (millisecs): 115
median response time (millisecs): 32
throughput: 1000
p99 (99th percentile) response time: 2021
min response time (millisecs): 12
max response time (millisecs): 9601

## III. Day 3

------ Client Post Statistics ------

------ PART 1 ------
number of successful requests sent: 160420
number of unsuccessful requests: 0
the total run time for all phases to complete: 102934
the total throughput in requests per second: 1000

------ PART 2 ------
mean response time (millisecs): 118
median response time (millisecs): 31
throughput: 1000
p99 (99th percentile) response time: 2062
min response time (millisecs): 12
max response time (millisecs): 9888