Load Testing and Analyzing Test Results

What is achieved with Load Testing?

- What is the number of users the system can handle?
- What is the response time for each transaction?
- How each component in the system i.e Application Server, Web Server, Database etc. behave under load?
- What server configuration is best to handle the load?
- Existing hardware is enough or is there any need for additional hardware?
- What are the bottlenecks such as CPU utilization, memory usage, network delays, reading database, third party apps etc.?

Approach for Load Testing

1. Identify Business Scenarios that needs to be tested based on the main business flow. If there is an existing application, user access patterns can be obtained from server logs.

2. Identify Load Test Acceptance Criteria

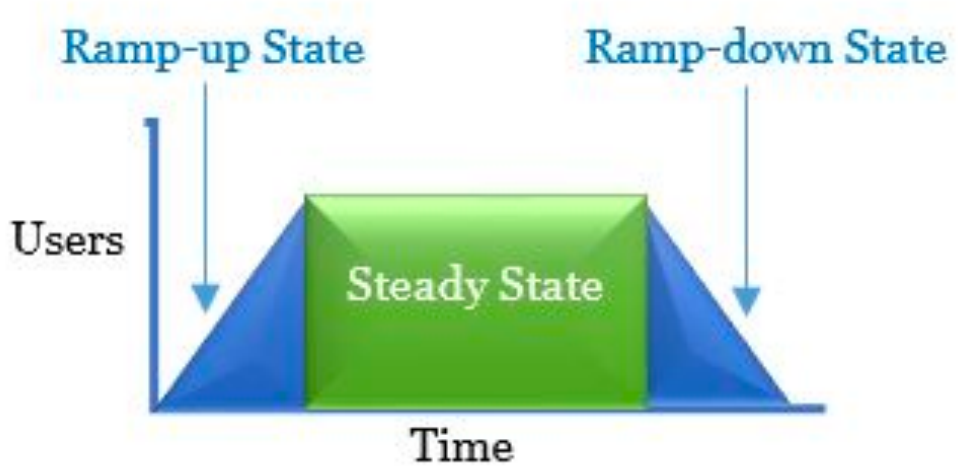
Some examples:

- The response time of the Login page shouldn't be more than 4 sec even during the max load conditions.
- CPU utilization should not be more than 85%.
- The throughput of the system should be 100 transactions per sec

3. Design the workload: Modeling the workload in such a way which mimics the actual user navigation or expected.

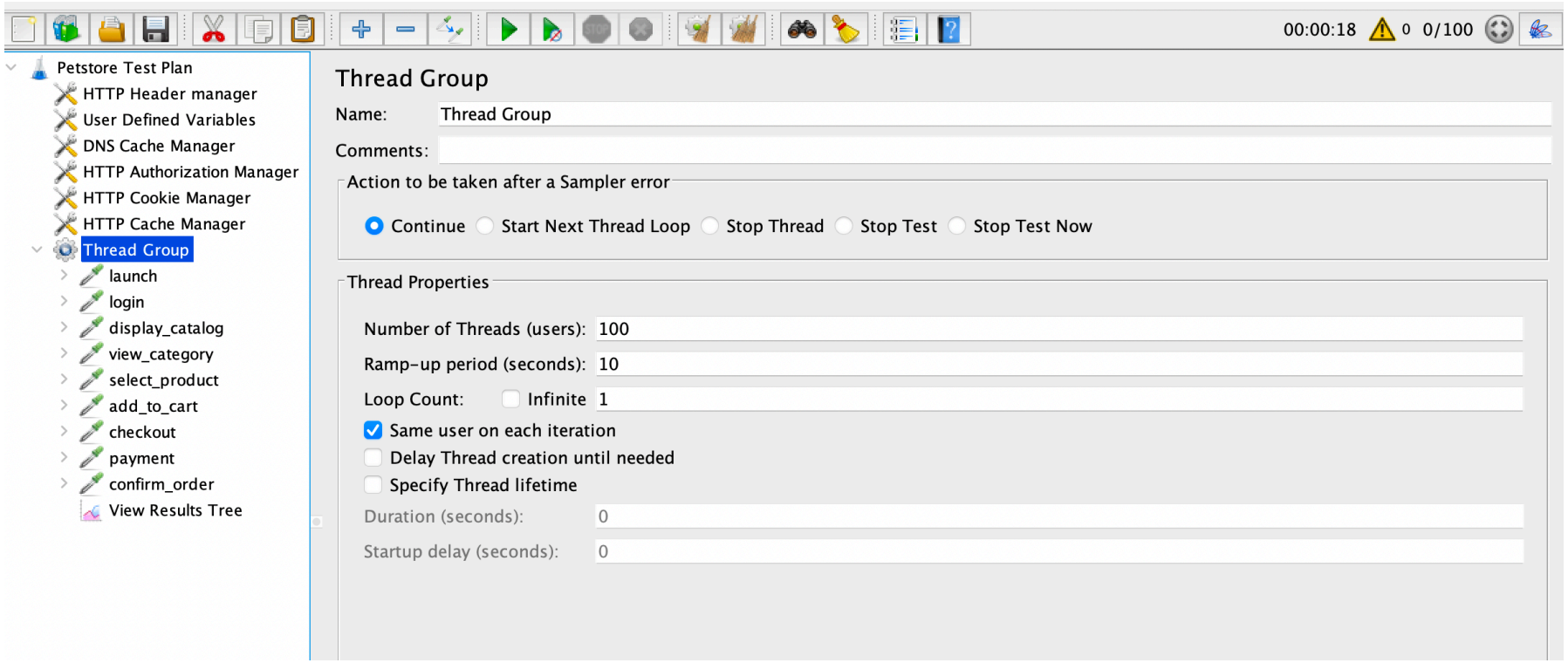
Workload

- Workload pattern shows the number of users accessing to an application, usually contains a Ramp-up, Ramp-down and a Steady State phase.



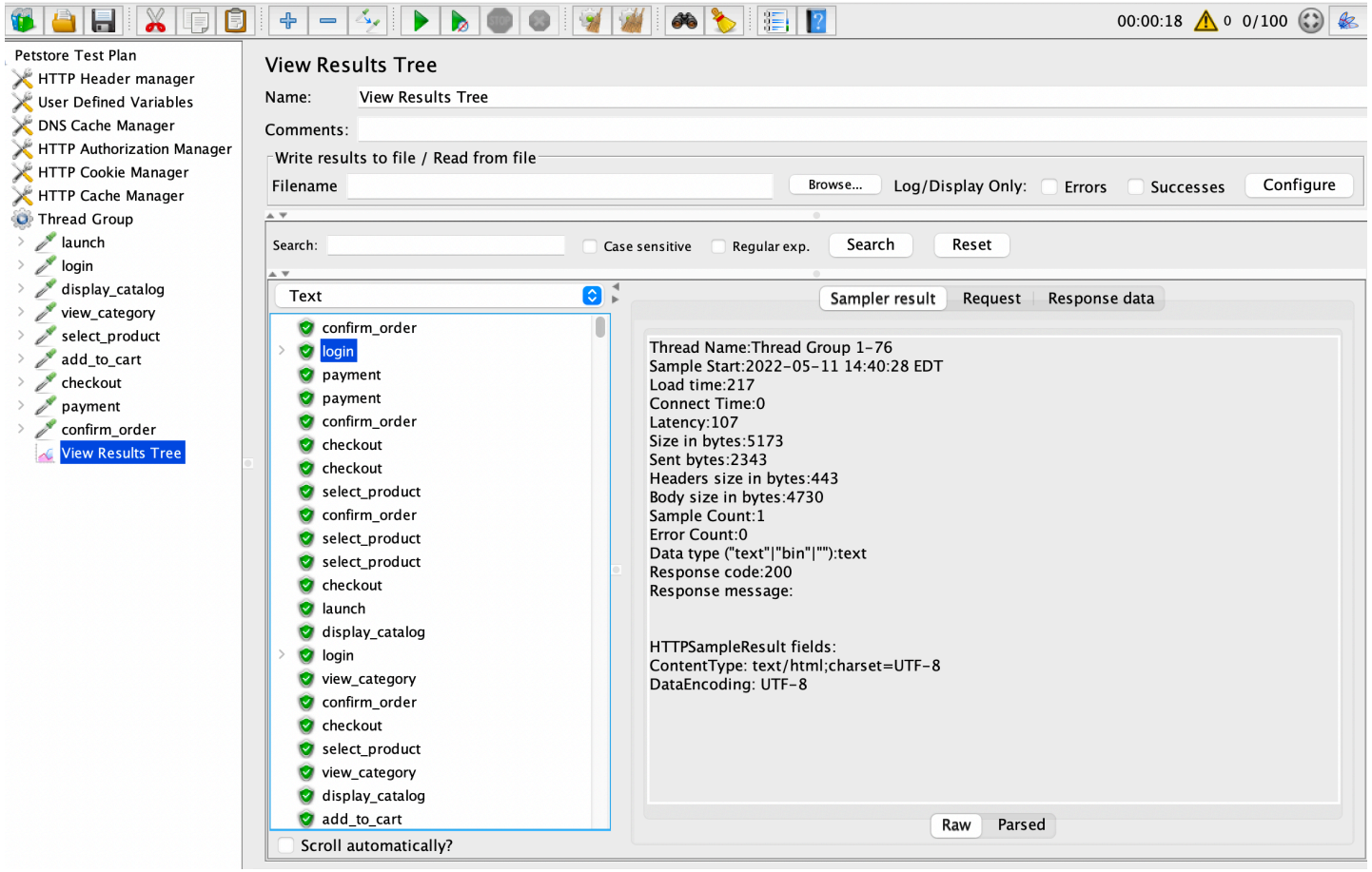
Sample Scenario - Load Testing

- Set the number of users for the Pet store scenario.
- Run the test and analyze the results.

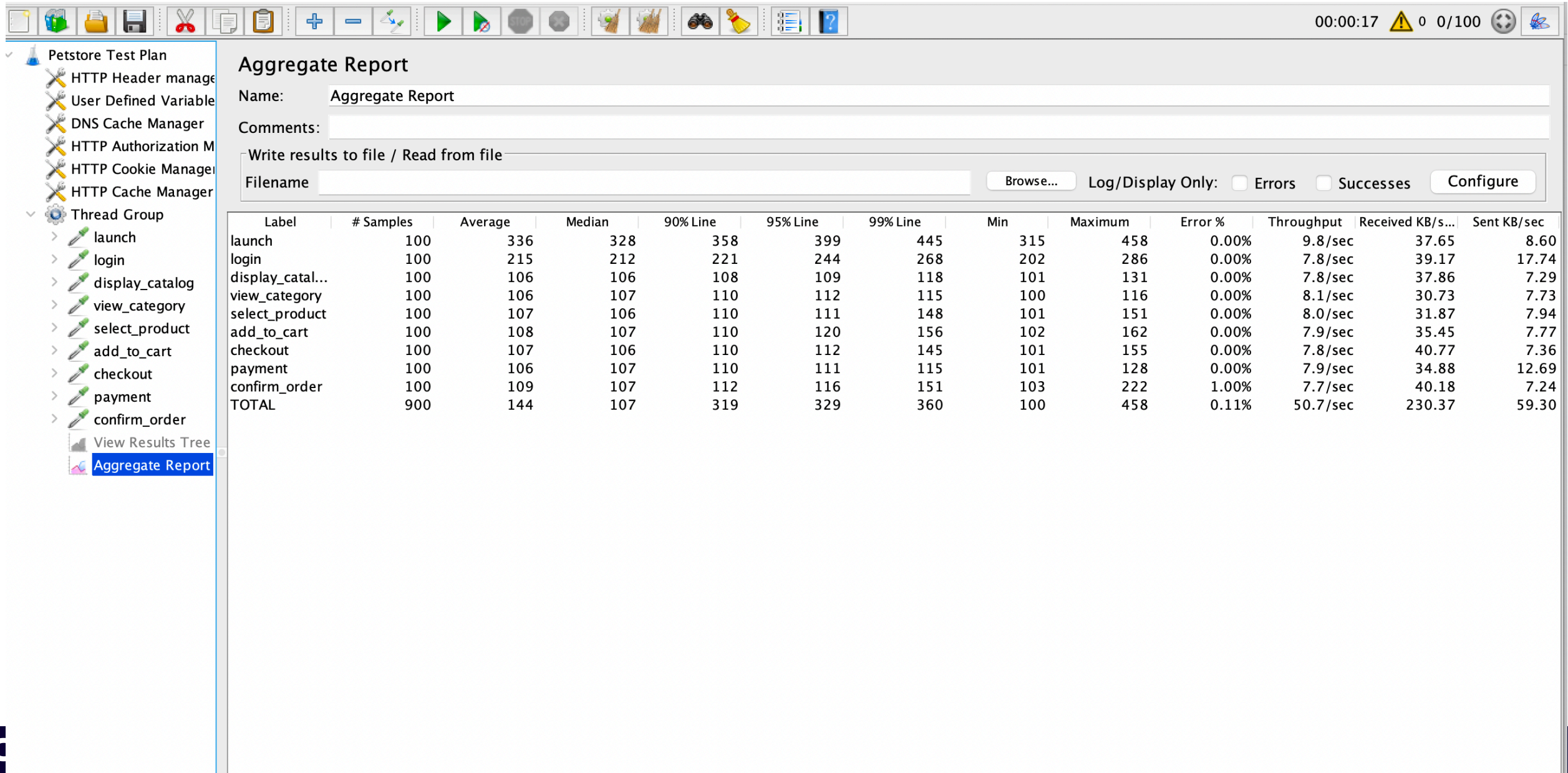


Test Results - View Results Tree Listener

- **Latency**: Measures the latency from just before sending the request to just after the first chunk of the response received
- **Connect Time**: Measures the time to establish the connection
- **Connect Time / Latency** should be as low as possible, ideally less than 1 second.



Test Results - Aggregate Report Listener



Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received KB/s...	Sent KB/sec
launch	100	336	328	358	399	445	315	458	0.00%	9.8/sec	37.65	8.60
login	100	215	212	221	244	268	202	286	0.00%	7.8/sec	39.17	17.74
display_catal...	100	106	106	108	109	118	101	131	0.00%	7.8/sec	37.86	7.29
view_category	100	106	107	110	112	115	100	116	0.00%	8.1/sec	30.73	7.73
select_product	100	107	106	110	111	148	101	151	0.00%	8.0/sec	31.87	7.94
add_to_cart	100	108	107	110	120	156	102	162	0.00%	7.9/sec	35.45	7.77
checkout	100	107	106	110	112	145	101	155	0.00%	7.8/sec	40.77	7.36
payment	100	106	107	110	111	115	101	128	0.00%	7.9/sec	34.88	12.69
confirm_order	100	109	107	112	116	151	103	222	1.00%	7.7/sec	40.18	7.24
TOTAL	900	144	107	319	329	360	100	458	0.11%	50.7/sec	230.37	59.30

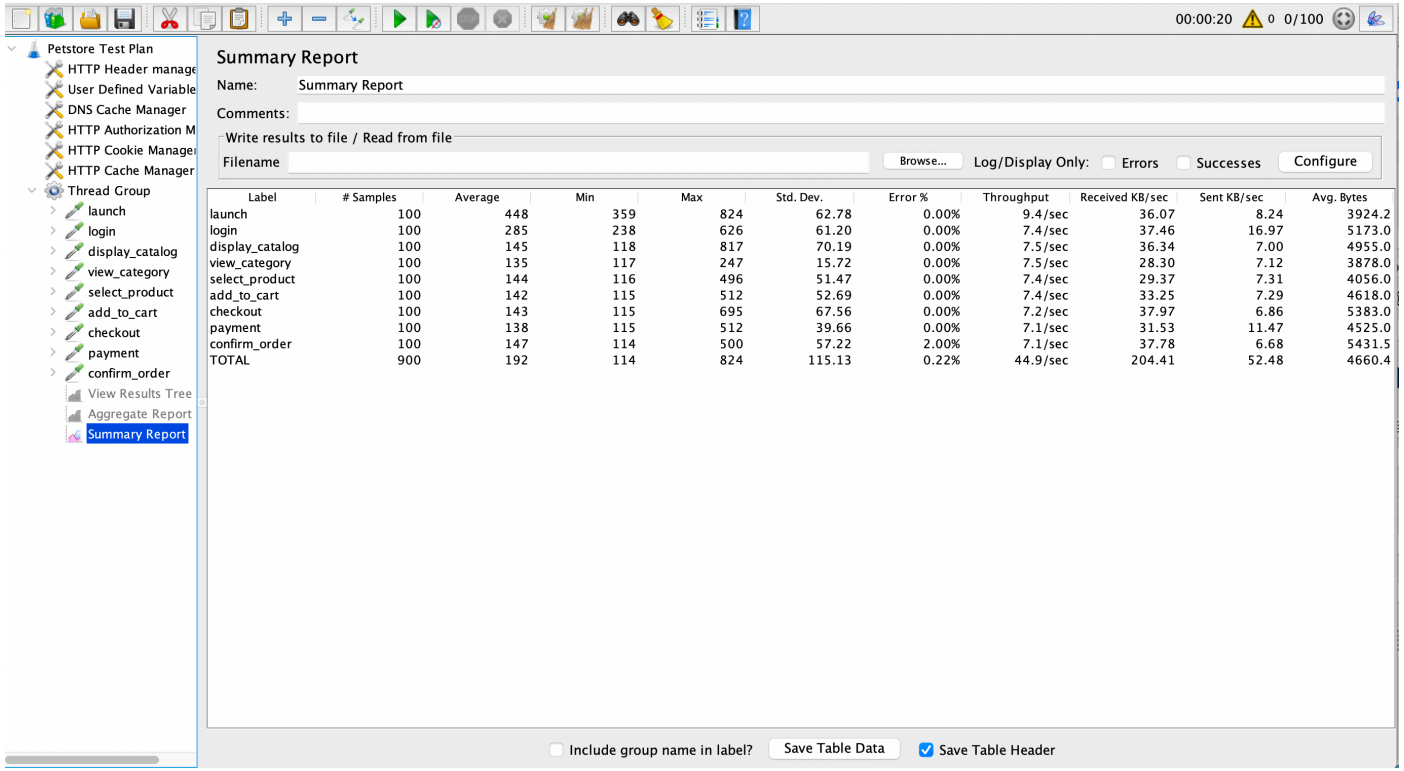
Test Results - Aggregate Report Listener (Cont.)

- **# Samples:** Number of Threads (users) * Loop Count
- **Average** (millisecond): Average **response time** of a set of results.
- **Min** (millisecond): Shortest time for the samples with the same label
- **Max** (millisecond): Longest time for the samples with the same label
- **Median:** Time in the middle of a set of samples result. It tells us that 50% of the samples took no more than this time and the remainder took at least as long.
- **Percentile** is a measure used in statistics indicating the value below which a given percentage of observations in a group of observations fall.
  - **90% Line (90th Percentile)** means 90% of the samples took **NO MORE THAN** this time. The **10%** remaining samples took at least as long as this.
  - **95% Line (95th Percentile)** means 95% of the samples took **NO MORE THAN** this time. The **5%** remaining samples took at least as long as this.
  - **99% Line (99th Percentile)** means 99% of the samples took **NO MORE THAN** this time. The **1%** remaining samples took at least as long as this.

Test Results - Summary Report Listener

- **Error%:** Percentage of failed requests per label
- **Std. Dev.:** Represents the exceptional cases which were deviating from the average value of sample response time. The lesser this value more consistent the data. Standard deviation should be less than or equal to half of the average time for a request.
- **Throughput:** Number of requests that are processed per second by the server.
- **KB/sec:** The throughput measured in Kilobytes per second. The formula is:

  Throughput KB/sec = (Throughput * Average Bytes) / 1024



Analyzing the Test Results

Response Time indicates the time which the server used to process the request, it should be as low as possible. And **throughput** indicates how many requests will be processed at a time, so it should be as high as possible.

Response Time	Throughput	Conclusions
Low	High	When the **Response time** is **low** and the **Throughput** is very **high**, it means that the Server is working well. The performance test is passed or you can consider to increase load and continue to find out the limitation of the Server.
High	Low	When the **Response time** for the request is **high** but the **Throughput** is much **lower**, it means that the Server is not capable enough to sustain/execute the request. Ask for tuning in the server side.

Test Results - Headless Listeners

- **Headless Listeners**
  - Headless listeners such as Simple Data Writer and Backend Listener are designed to work when JMeter is run from the command line.
  - These listeners use less memory than UI listeners like View Results Tree, Aggregate Report and Summary Report.