LOAD TESTING REPORT

Tool: The tool being used for the load testing this time will be apache jMeter version 5.6.3.

Login Load Test Cases

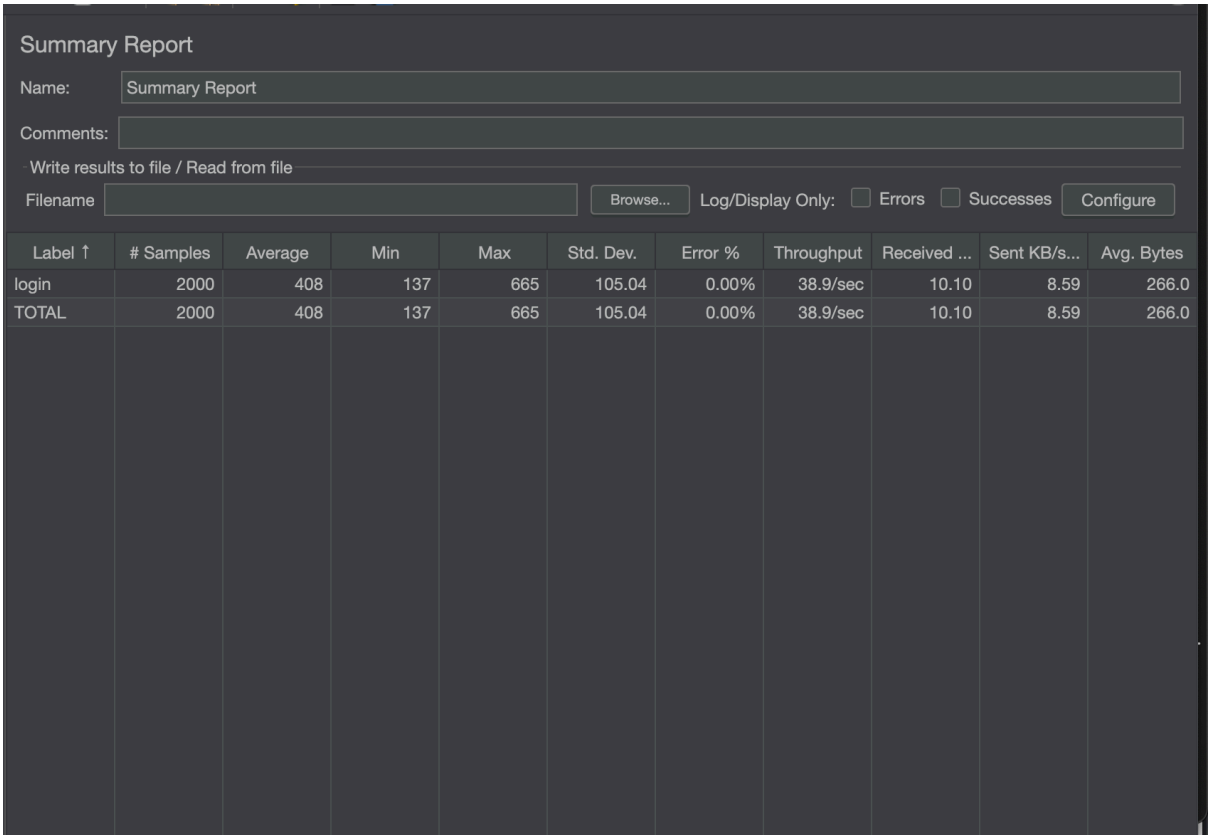
Test Case 1: Normal Login Scenario (Valid Username and Password)

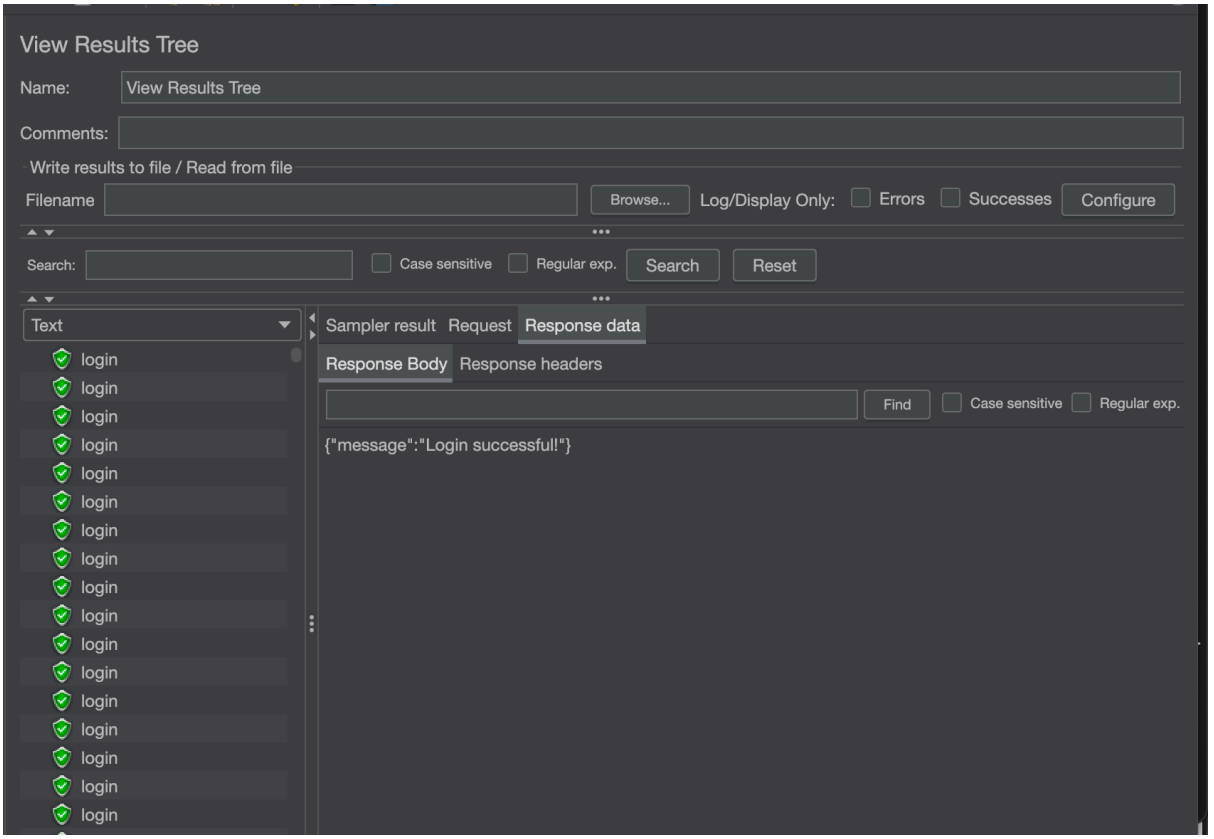
- Objective: To test the system's ability to handle multiple users logging in with valid credentials.
- Input Data: A set of valid usernames and passwords (can be sourced from CSV or manually configured).
- Expected Output: A successful login with a 200 OK status code.
- Simulated Users: Simulate 20 users, each sending a login request with valid credentials

Test Configuration

- Number of Users (Threads): 20 users
- Ramp-Up Time: 12 seconds (users start 1 by 1 with 0.6-second intervals)
- Test Duration: Run indefinitely, simulating 100 requests per minute for 10 minutes.
- Test Scenario:
    - Login API (valid credentials)

Test case load test report.





Test Case 2: Valid Registration Scenario

Objective:

To test the system's ability to handle user registrations with valid credentials, ensuring the system correctly processes and registers users.

Input Data:

A set of valid usernames, passwords, and confirm passwords.

Expected Output:

A successful registration with a 201 Created status code and a response message such as "User registered successfully."

Simulated Users:

Simulate 100 users, each sending a registration request with valid data (valid username, password, and confirm password).

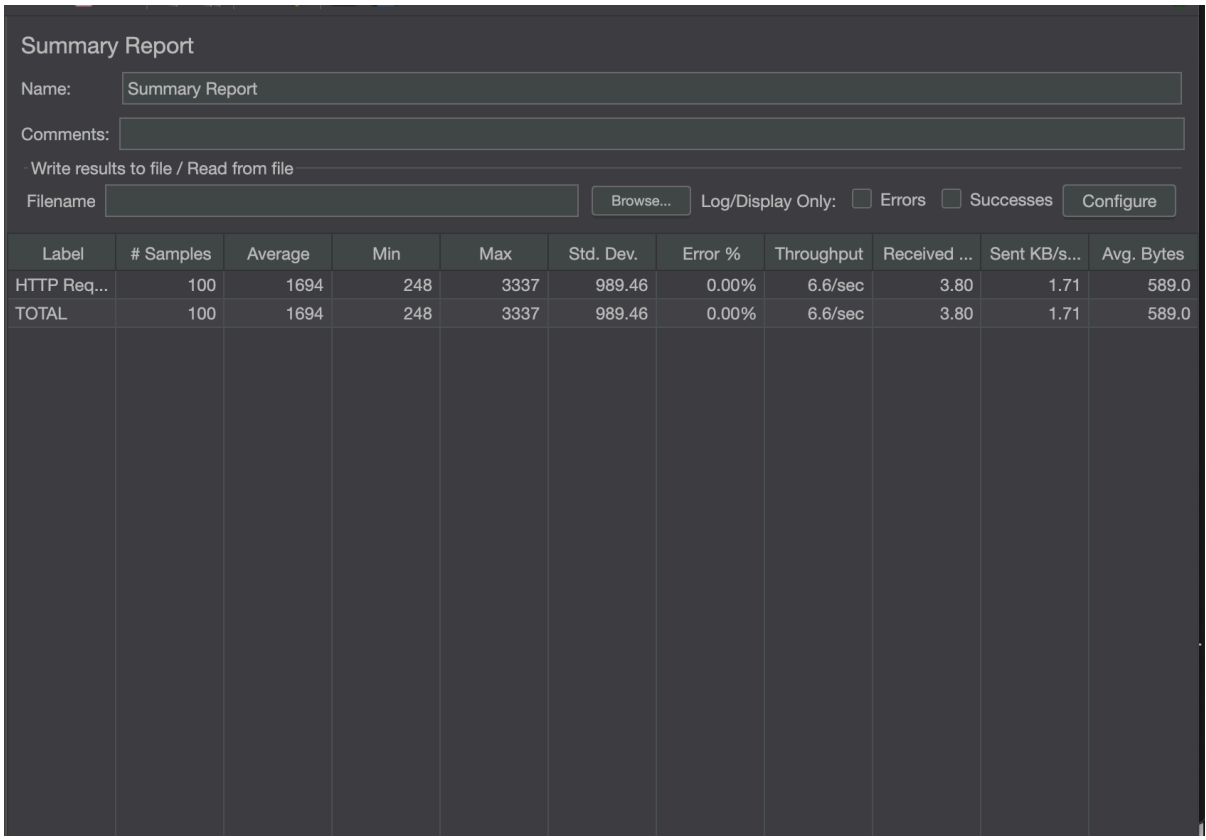
Jmeter Test Configuration:

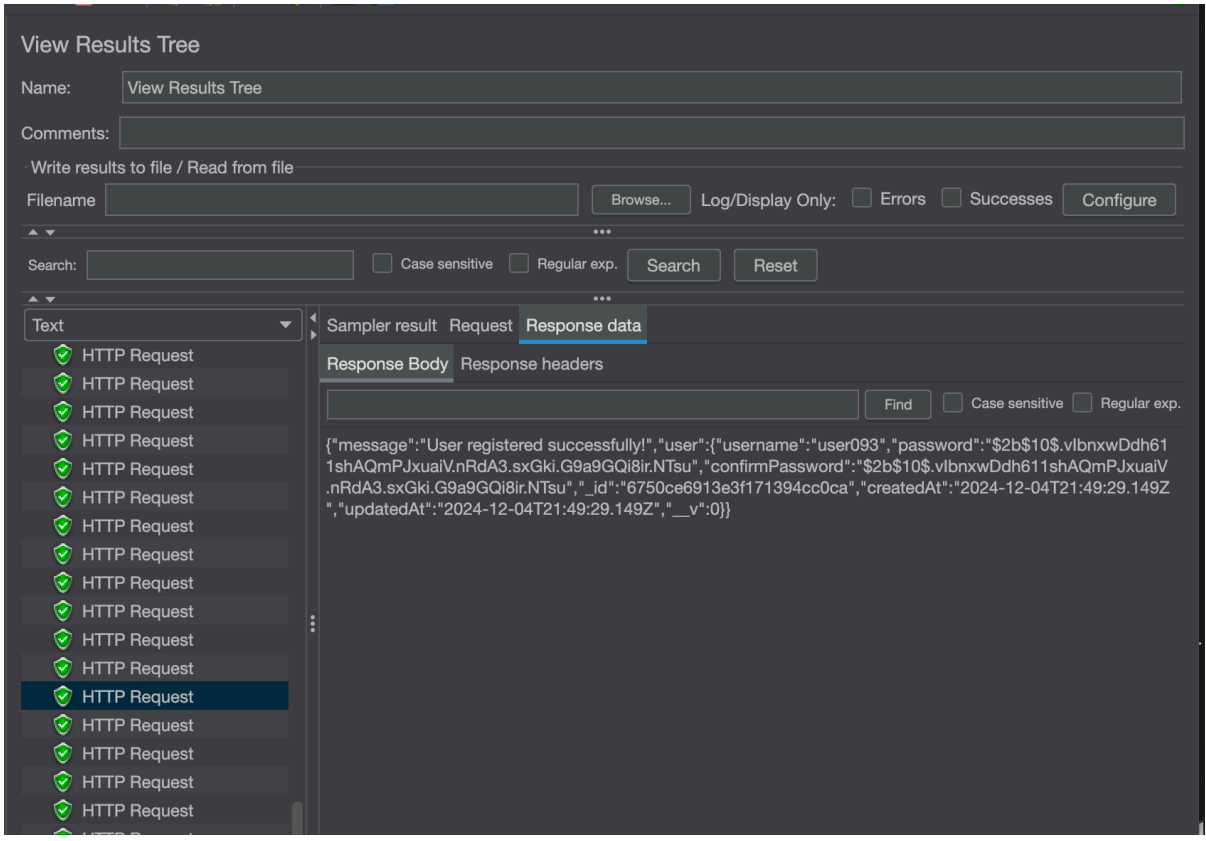
- Number of Users (Threads): 100 users
    - This means that 100 virtual users will attempt to register simultaneously with valid credentials.
- Ramp-Up Time: 5 seconds
    - Users will be simulated starting in 5 seconds with each user starting 0.05 seconds apart. For example, the first user will start immediately, the second user will start after 0.05 seconds, and so on.
- Test Duration: Run Once
    - The test will run once and not indefinitely. With the given ramp-up time and the 100 users, the total test duration will be relatively short.
- Test Scenario: Registration API (Valid Data)
    - In this scenario, users will send valid registration data (a valid username, password, and confirm password) to the registration endpoint.

Expected Behavior:

201 Created: Each user's registration should succeed, and the user should be created in the system.

Test case load test report.





Test Case 3: Recipe Search Scenario

Objective:
To test the system's ability to handle multiple users making search requests for recipes using valid query parameters, simulating frequent and repeated searches as part of a major feature of the application.

Input Data:
Valid query parameters for the /recipe/search endpoint:

- keyword: A valid search term like "chicken".
- mealType: A valid meal type such as "lunch", "dinner", etc.
- time: A valid time parameter such as "30" (maximum cooking time of 30 minutes).

Expected Output:

- A successful search with a 200 OK status code.
- The response should return relevant recipe data in JSON format, which includes recipes matching the search parameters.

Simulated Users:
Simulate 20 users, each sending 100 requests in a loop.

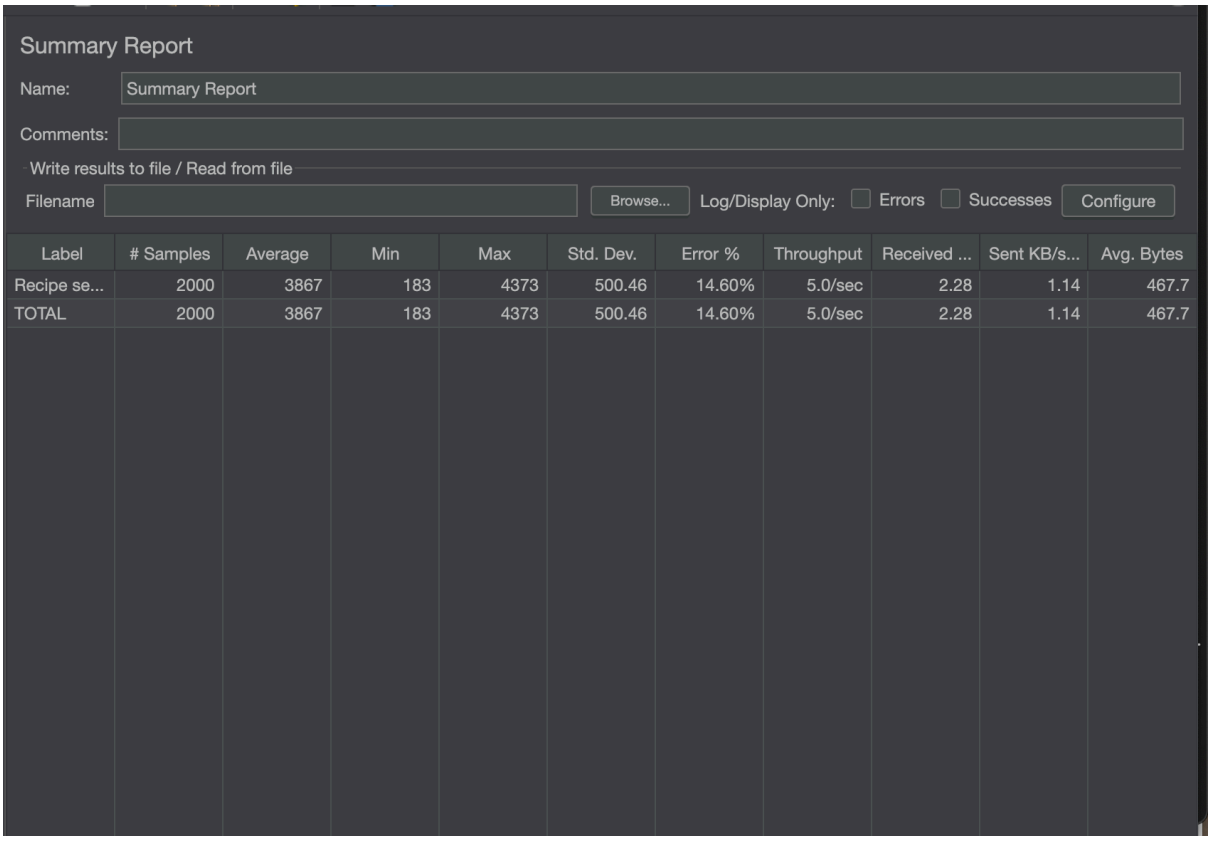
Test Configuration:

- Number of Users (Threads):
  20 users (20 virtual users will simultaneously send requests to search for recipes).
- Ramp-Up Time:
  20 seconds (the users will be simulated starting one by one, with 1 user starting every second).
- Loop Count:
  100 (Each of the 20 users will send 100 requests during the test).
- Test Duration:
  This test will simulate continuous searching for a duration required to send 100 requests per user, and the test will run for as long as it takes to send all 100 requests per user.

JMeter Configuration:

1. Thread Group Configuration:
   - Number of Threads (Users): 20
   - Ramp-Up Period**:** 20 seconds
   - Loop Count: 100 (this means each user will repeat the request 100 times)
2. Request:
   - Set up an HTTP Request for `/recipe/search` with the parameters `keyword`, `mealType`, and `time`.
3. Assertions:
   - Add a Response Assertion to check if the response code is `200 OK`.

Test case load test report.



Summary Report

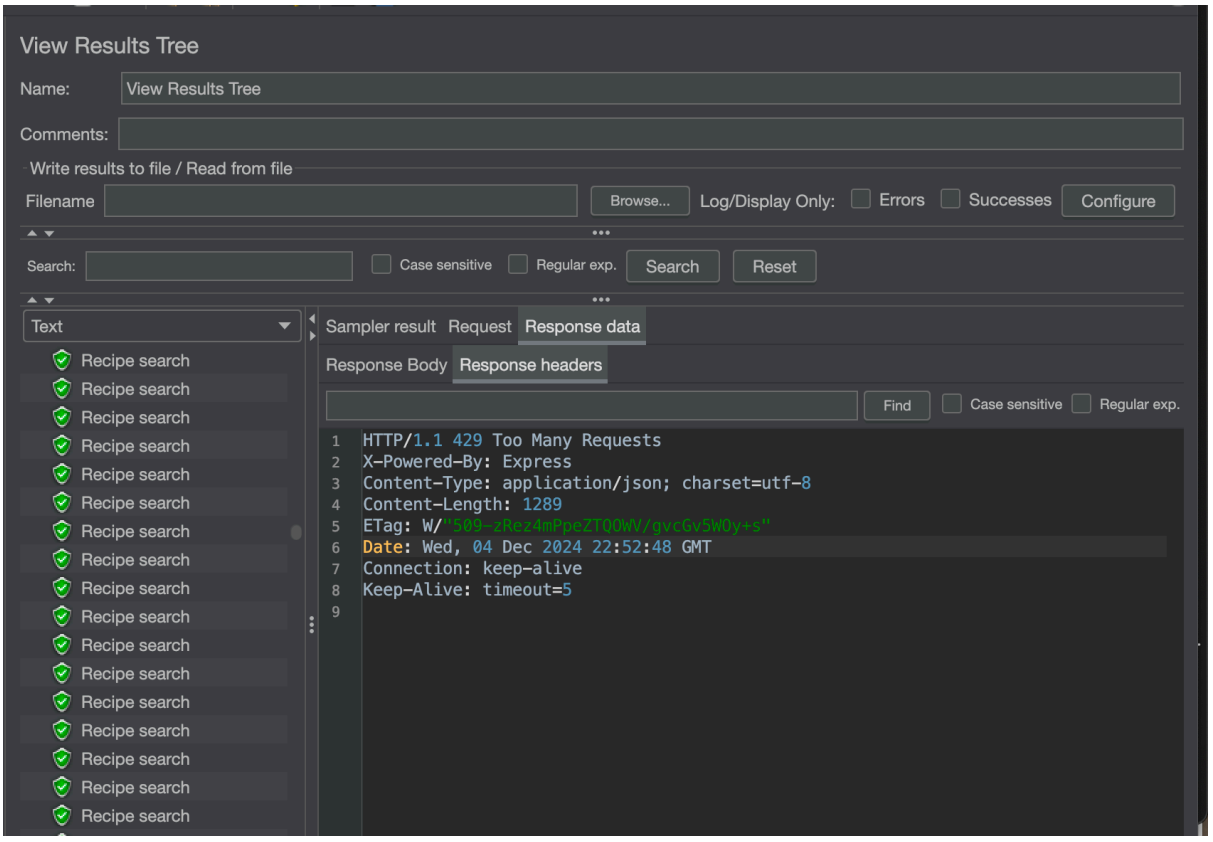
Name: Summary Report

Comments:

Write results to file / Read from file

Filename Browse... Log/Display Only: Errors Successes Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received ...	Sent KB/s...	Avg. Bytes
Recipe se...	2000	3867	183	4373	500.46	14.60%	5.0/sec	2.28	1.14	467.7
TOTAL	2000	3867	183	4373	500.46	14.60%	5.0/sec	2.28	1.14	467.7



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

Text

Recipe search

Sampler result Request Response data

Response Body Response headers

Find Case sensitive Regular exp.

```
1  HTTP/1.1 429 Too Many Requests
2  X-Powered-By: Express
3  Content-Type: application/json; charset=utf-8
4  Content-Length: 1289
5  ETag: W/"509-zRez4mPpeZTQOWV/gvcGv5WOy+s"
6  Date: Wed, 04 Dec 2024 22:52:48 GMT
7  Connection: keep-alive
8  Keep-Alive: timeout=5
9
```

Test Case 4: Favorites Endpoint Scenario

Objective

To test the system's ability to handle multiple users adding a recipe to their favorites using valid query parameters, simulating concurrent user actions interacting with this feature.

Input Data

Valid parameters for the /favorites/:recipeId endpoint:

- recipeId: A valid recipe ID (e.g., 3f40351ef85b4323b4c9bf654355cafe).
- username: A valid username (e.g., pops).
- title: A valid recipe title (e.g., shredded chicken).

Expected Output

- A successful request with a 201 Created status code when the recipe is stored successfully.

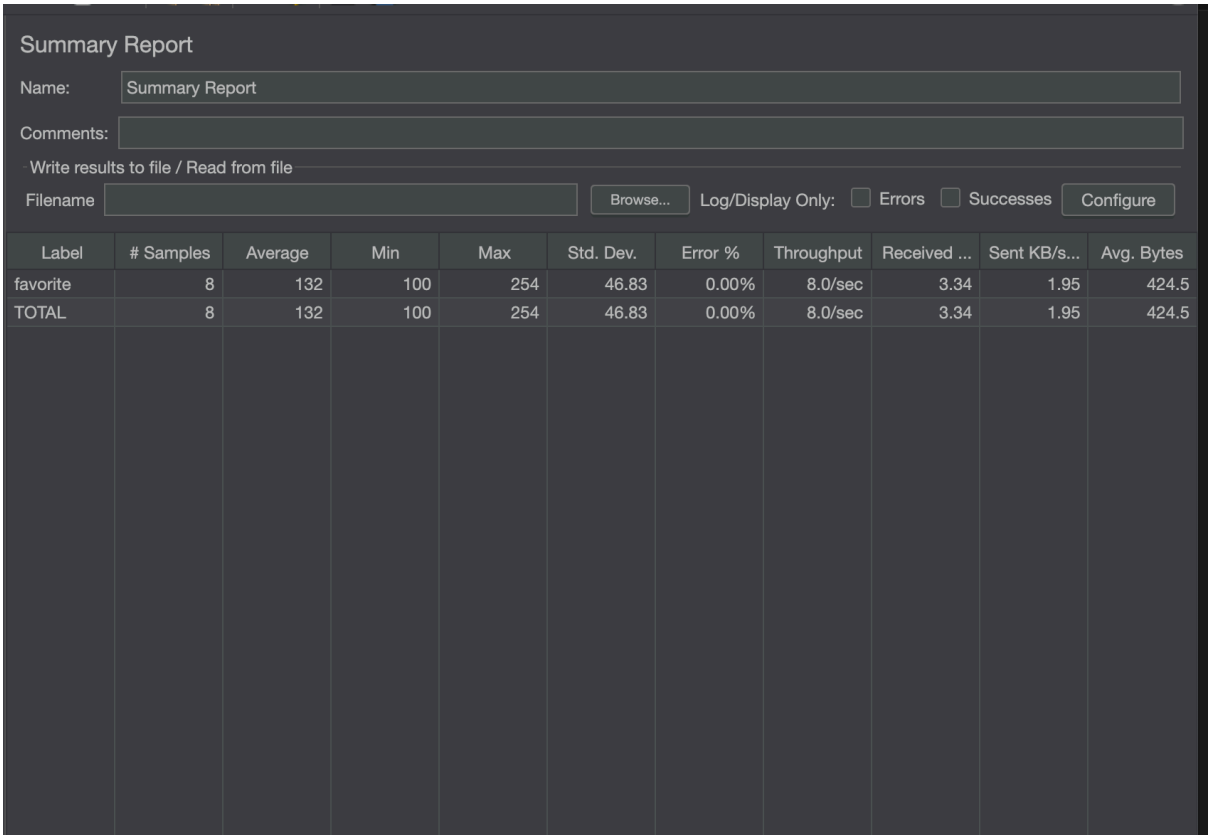
Simulated Users

Simulate 8 users, each sending 1 request to add a recipe to their favorites.

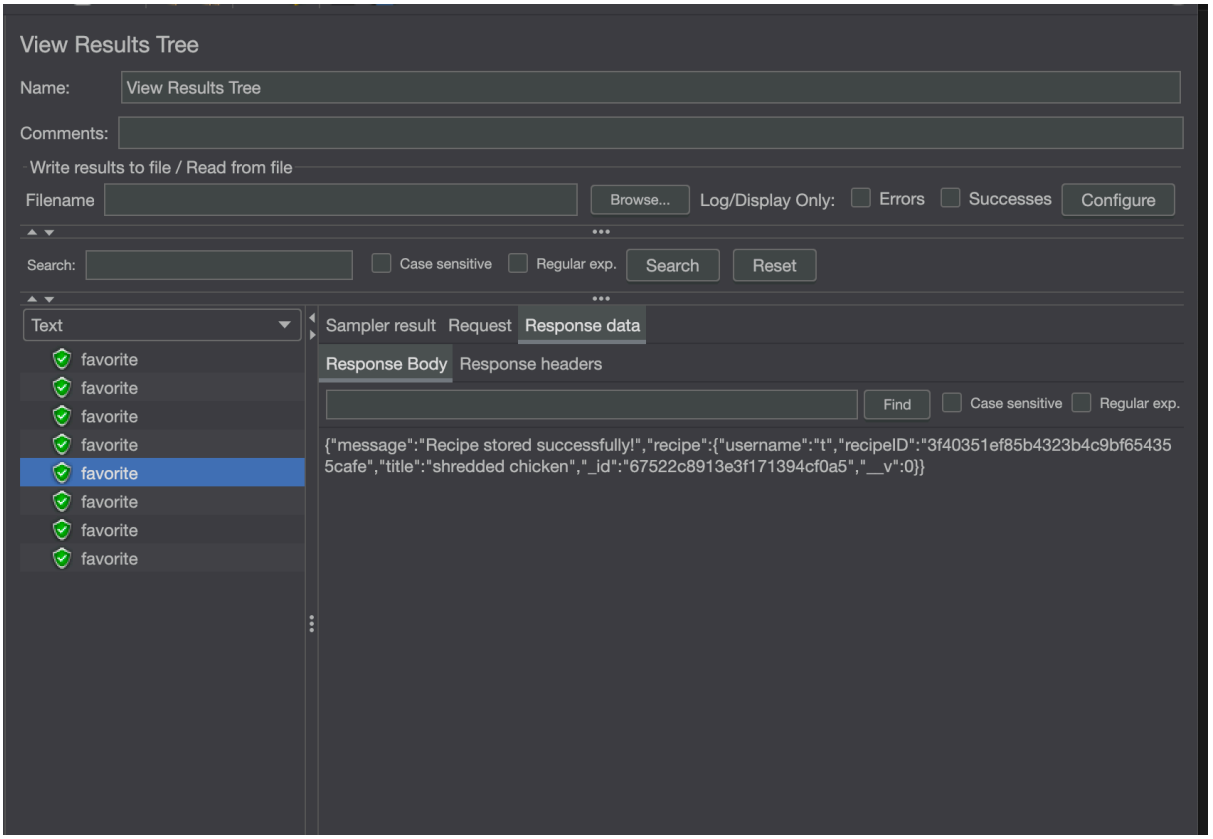
Test Configuration

- Number of Users (Threads): 8 users
- Ramp-Up Time: 1 second (1 user starts every 0.125 seconds).
- Loop Count: 1 (Each user will send 1 request).
- Test Duration: Runs for a short period, as only 1 request per user is sent.

TEST CASE LOAD REPORT



Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received ...	Sent KB/s...	Avg. Bytes
favorite	8	132	100	254	46.83	0.00%	8.0/sec	3.34	1.95	424.5
TOTAL	8	132	100	254	46.83	0.00%	8.0/sec	3.34	1.95	424.5



Bottleneck: Edamam API Rate Limiting

Issue Description:

During the load testing of the Recipe Search API endpoint, a bottleneck was identified where 14.6% of requests resulted in 429 (Too Many Requests) errors. This occurred because the Edamam API, which powers the recipe search feature, imposed rate limiting on the number of incoming requests.

Impact:

- User Experience: A significant portion of users experienced failed searches during high traffic periods.
- System Reliability: The overall success rate of the Recipe Search endpoint decreased, causing degraded performance in one of the core features.

Root Cause:

The Edamam API enforces a rate limit on incoming requests per minute, which was exceeded during the load test that simulated 20 users sending 100 requests per minute.

DID WE MEET OUR GOALS

Objective: Ensure that the Recipe Search API can handle 20 users sending 100 requests per minute.
Outcome: The system handled most requests successfully, but 14.6% of requests failed with a 429 Too Many Requests error due to Edamam's API rate limiting. Therefore, the non-functional requirement of handling high traffic was partially met.

- Successes:
    - The internal API and server infrastructure handled the load without significant latency or server crashes.
- Challenges:
    - The primary bottleneck was rate limiting by the Edamam API, which restricted the number of allowed requests per minute, leading to failures during high-traffic periods.

COULD WE MEET GOALS WITH MONEY

Edamam Api offers different plans, perhaps upgrading the plan to a higher tier could perhaps solve a lot of these issues. There could also be other alternatives that can be explored even without money though, like implementing a load balancer.