API Performance Testing

Approach

1. I have used a JMETER as an open source tool.

2. It's free and platform independent.

3. Using JMETER we can easily add different assertions/create virtual users/measure response time etc.

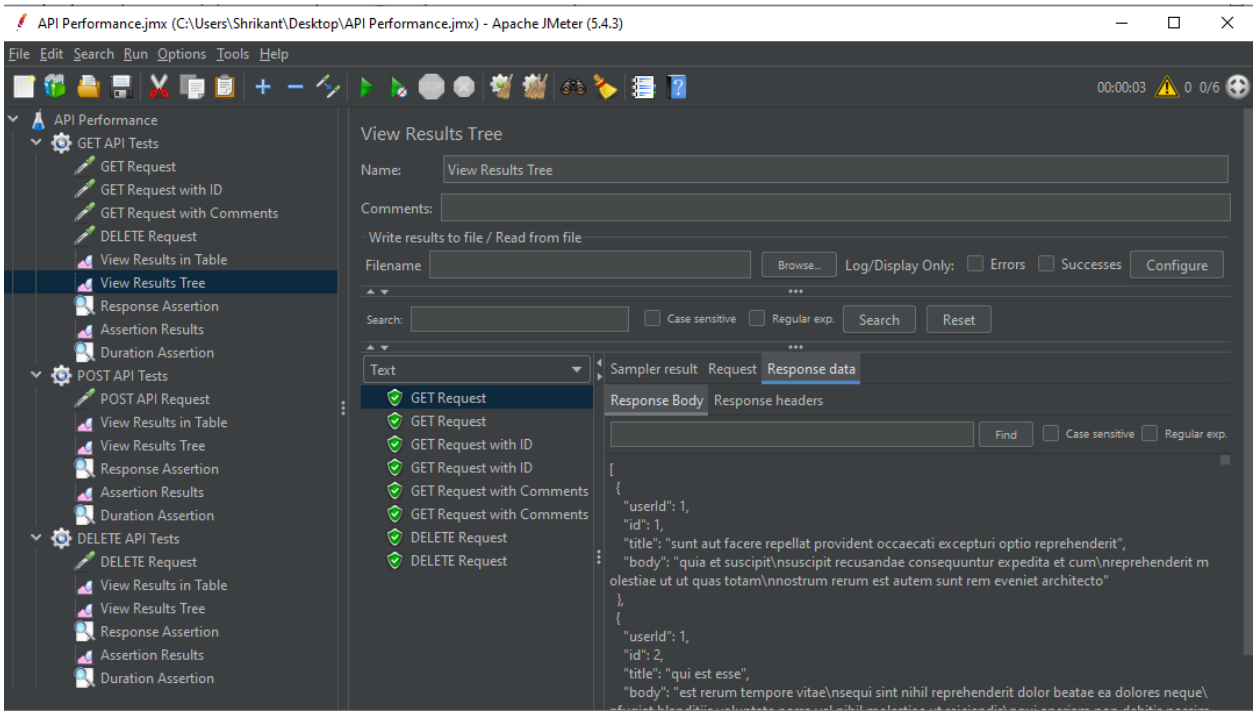
Programming language

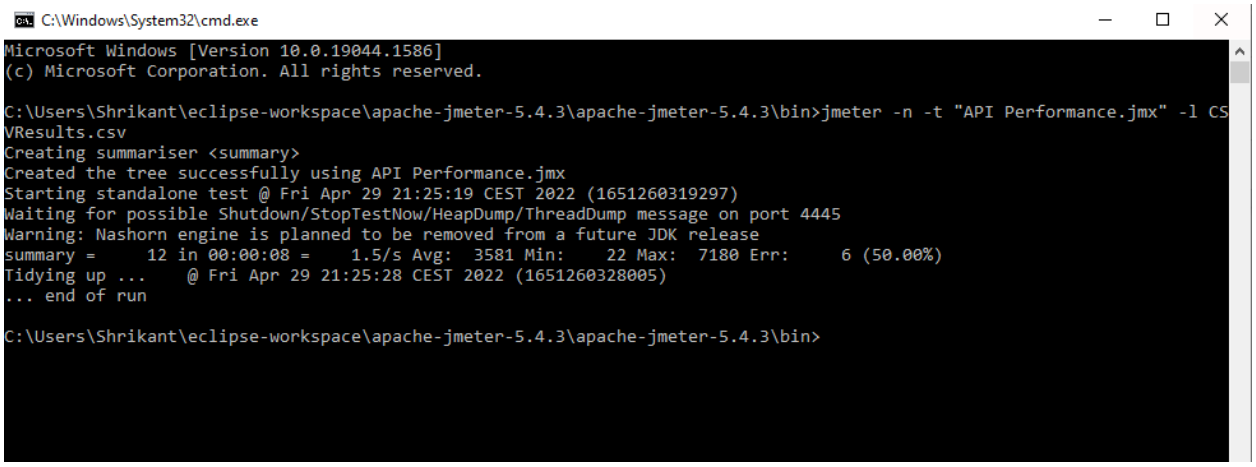
1. JMeter has their own UI + we can use bash scripting as well.

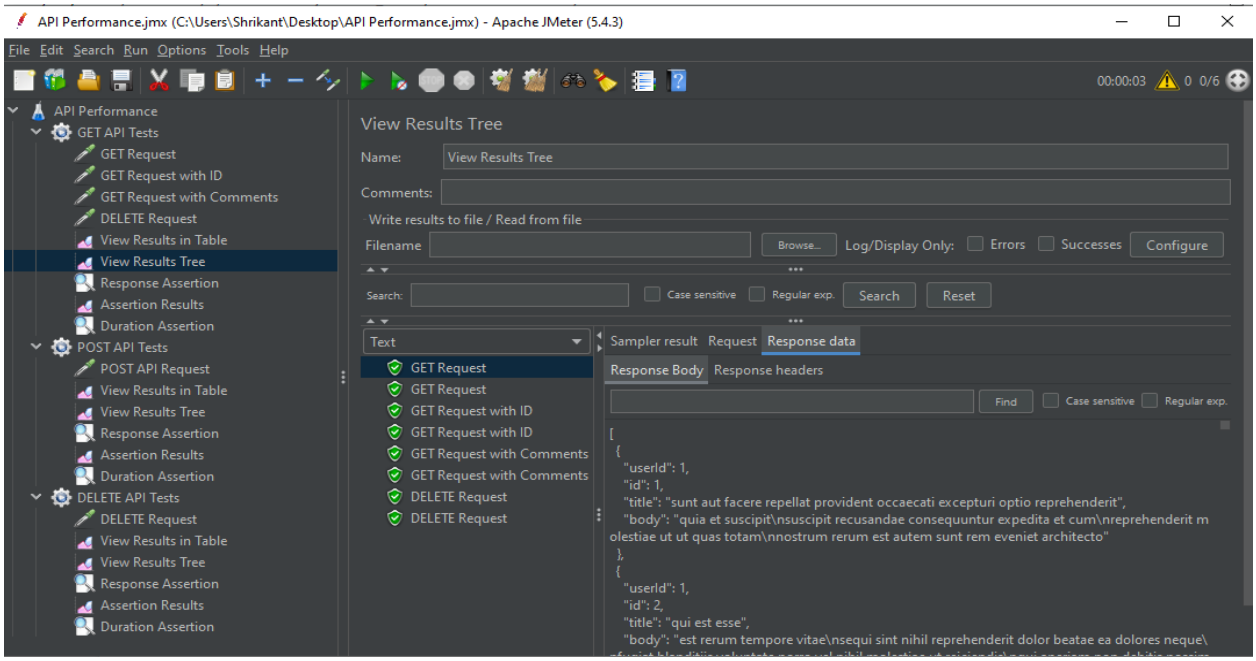
How to run

1. Clone the entire project on a local machine from GitHub.

2. Execute test from JMeter UI




C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.1586]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Shrikant\eclipse-workspace\apache-jmeter-5.4.3\apache-jmeter-5.4.3\bin>jmeter -n -t "API Performance.jmx" -l CS
VResults.csv
Creating summariser <summary>
Created the tree successfully using API Performance.jmx
Starting standalone test @ Fri Apr 29 21:25:19 CEST 2022 (1651260319297)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
Warning: Nashorn engine is planned to be removed from a future JDK release
summary = 12 in 00:00:08 = 1.5/s Avg: 3581 Min: 22 Max: 7180 Err: 6 (50.00%)
Tidying up ... @ Fri Apr 29 21:25:28 CEST 2022 (1651260328005)
... end of run
C:\Users\Shrikant\eclipse-workspace\apache-jmeter-5.4.3\apache-jmeter-5.4.3\bin>


API Performance.jmx (C:\Users\Shrikant\Desktop\API Performance.jmx) - Apache JMeter (5.4.3)
File Edit Search Run Options Tools Help
00:00:03 0 0/6
API Performance
GET API Tests
GET Request
GET Request with ID
GET Request with Comments
DELETE Request
View Results in Table
View Results Tree
Response Assertion
Assertion Results
Duration Assertion
POST API Tests
POST API Request
View Results in Table
View Results Tree
Response Assertion
Assertion Results
Duration Assertion
DELETE API Tests
DELETE Request
View Results in Table
View Results Tree
Response Assertion
Assertion Results
Duration Assertion
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
GET Request
GET Request
GET Request with ID
GET Request with ID
GET Request with Comments
GET Request with Comments
DELETE Request
DELETE Request
Sampler result Request Response data
Response Body Response headers
Find Case sensitive Regular exp.
[
{
"userId": 1,
"id": 1,
"title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
"body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto"
},
{
"userId": 1,
"id": 2,
"title": "qui est esse",
"body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\

	A	B	C	D	E	F	G	H	
1	timeStamp	elapsed	label	responseCode	responseMessage	threadName	dataType	success	failureMessage
2	1647197610180	1915	GET Request	200	OK	GET API Tests 1-1	text	false	The operation lasted too long: It took 1,915 milliseconds
3	1647197610180	1915	GET Request	200	OK	GET API Tests 1-2	text	false	The operation lasted too long: It took 1,915 milliseconds
4	1647197612112	25	GET Request with ID	200	OK	GET API Tests 1-2	text	true	
5	1647197612117	24	GET Request with ID	200	OK	GET API Tests 1-1	text	true	
6	1647197612138	20	GET Request with Comments	200	OK	GET API Tests 1-2	text	true	
7	1647197612142	31	GET Request with Comments	200	OK	GET API Tests 1-1	text	true	
8	1647197610180	2058	DELETE Request	200	OK	DELETE API Tests 3-2	text	false	The operation lasted too long: It took 2,058 milliseconds
9	1647197610180	2061	DELETE Request	200	OK	DELETE API Tests 3-1	text	false	The operation lasted too long: It took 2,061 milliseconds
10	1647197610180	2074	POST API Request	201	Created	POST API Tests 2-1	text	false	The operation lasted too long: It took 2,074 milliseconds
11	1647197610180	2074	POST API Request	201	Created	POST API Tests 2-2	text	false	The operation lasted too long: It took 2,074 milliseconds
12	1647197612174	226	DELETE Request	200	OK	GET API Tests 1-1	text	true	
13	1647197612158	283	DELETE Request	200	OK	GET API Tests 1-2	text	true	
14	1651260320684	6927	GET Request	200	OK	GET API Tests 1-2	text	false	The operation lasted too long: It took 6,927 milliseconds
15	1651260320576	7035	GET Request	200	OK	GET API Tests 1-1	text	false	The operation lasted too long: It took 7,035 milliseconds
16	1651260327637	26	GET Request with ID	200	OK	GET API Tests 1-1	text	true	
17	1651260327637	31	GET Request with ID	200	OK	GET API Tests 1-2	text	true	
18	1651260327664	22	GET Request with Comments	200	OK	GET API Tests 1-1	text	true	
19	1651260327669	22	GET Request with Comments	200	OK	GET API Tests 1-2	text	true	
20	1651260320576	7177	DELETE Request	200	OK	DELETE API Tests 3-1	text	false	The operation lasted too long: It took 7,177 milliseconds
21	1651260320576	7180	POST API Request	201	Created	POST API Tests 2-1	text	false	The operation lasted too long: It took 7,180 milliseconds

Current Automation Features -

1. Created Test Plan in JMeter for GET/POST/DELETE.

2. Added a different listeners to see details ex - response time/status codes/response body.

3. Added **assertions** on response time.

4. Able to create parallel virtual users to see performance in case of load.

What features we can add to framework in next phase

2. Integrate Automation with **Jenkins** for continuous integration.

3. Pull code at runtime from **GitHub** by Jenkins, execute, and send emailable reports to relevant audiences.

4. Create Automation dashboard to **monitor 24*7 Jenkins jobs** status.

5. Store automation reports on cloud ex – S3 bucket in AWS.

6. Manage most of Automation configuration as command-line arguments.

8. Execute same scripts on different environments ex – Integration/Staging/Preview/Production etc.

9. Integrate Automation with different Third party tools ex – Slack for notification purpose / Integrate with JIRA to update automation results.