

Project Report - Lab 1

ENTERPRISE DISTRIBUTED SYSTEMS

Shivang Mistry | 013823108 | 17th March 2019

Part 1 - Calculator

Introduction

This application implements a calculator with basic functionalities like addition, subtraction, multiplication and division. The goal is to respond to the requests efficiently with optimum performance.

System Design

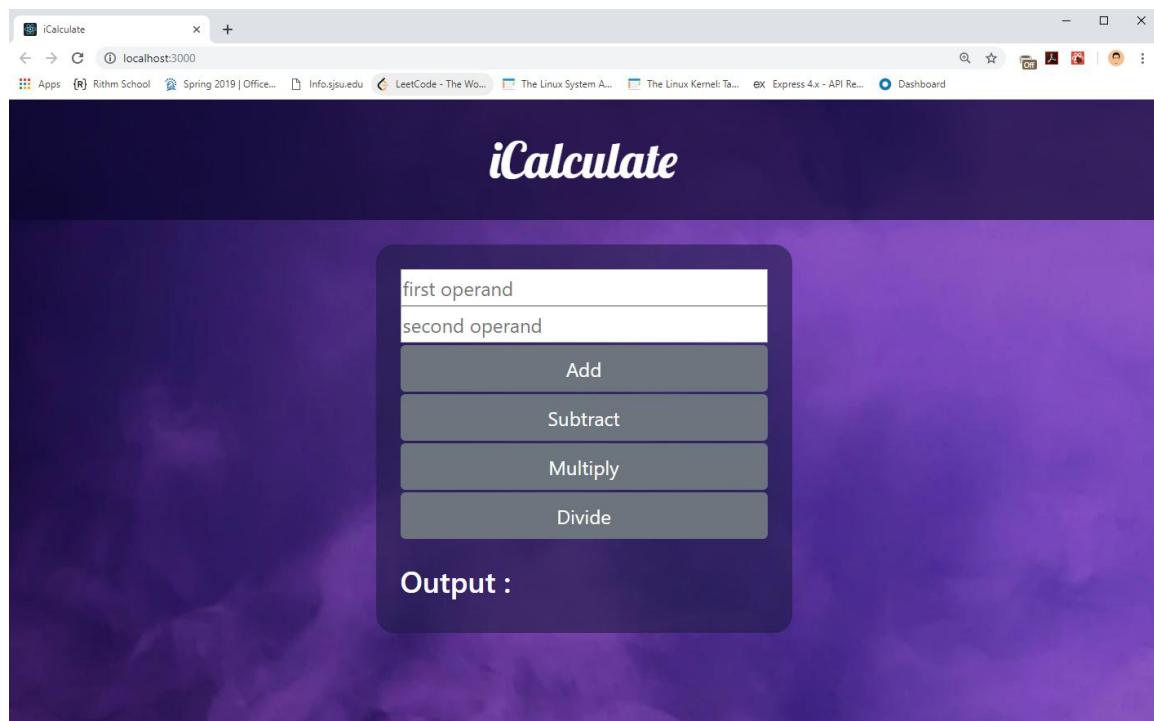
The system is composed of a server and multiple clients. The server is responsible for performing basic operations like addition, subtraction, multiplication and division, using RESTful services. It maintains a minimal design and simple layout for the client to enter data and get the result. The input fields are validated to accept a valid number. All the elements are responsive to varying browser width and hence appears compatible with mobile/tablet view.

Software: HTML, Javascript, CSS, Node.js, React, Bootstrap.

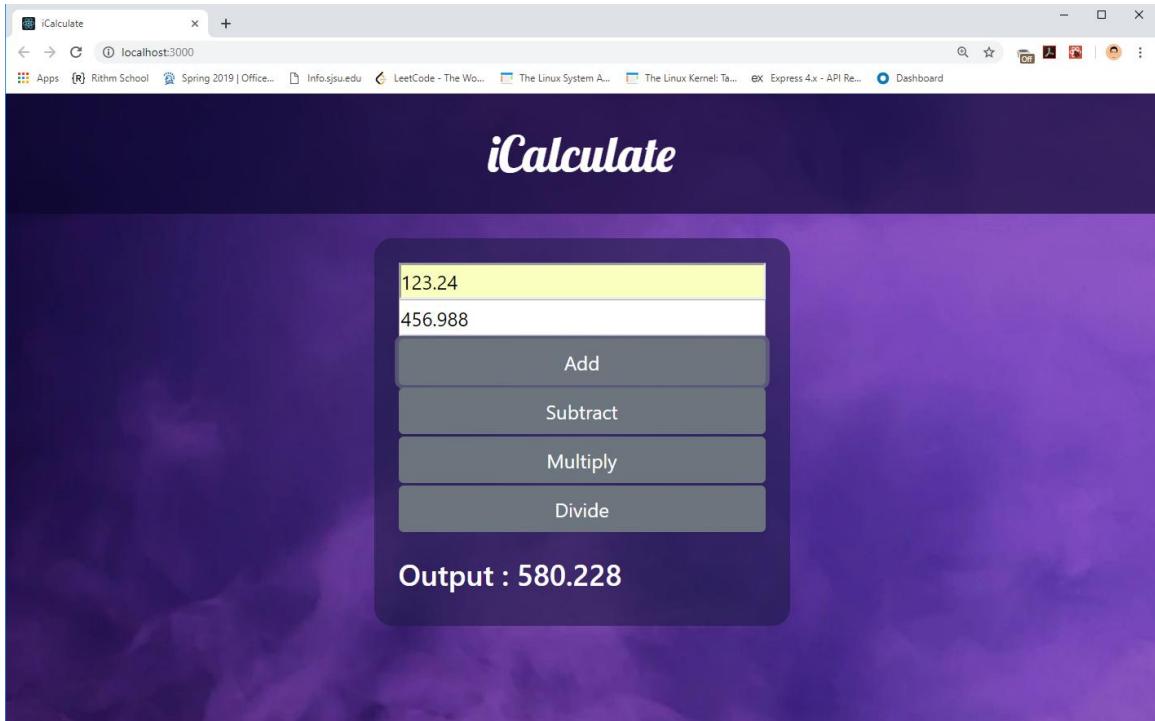
Hardware: Any computer having Chrome and required packages installed.

Results

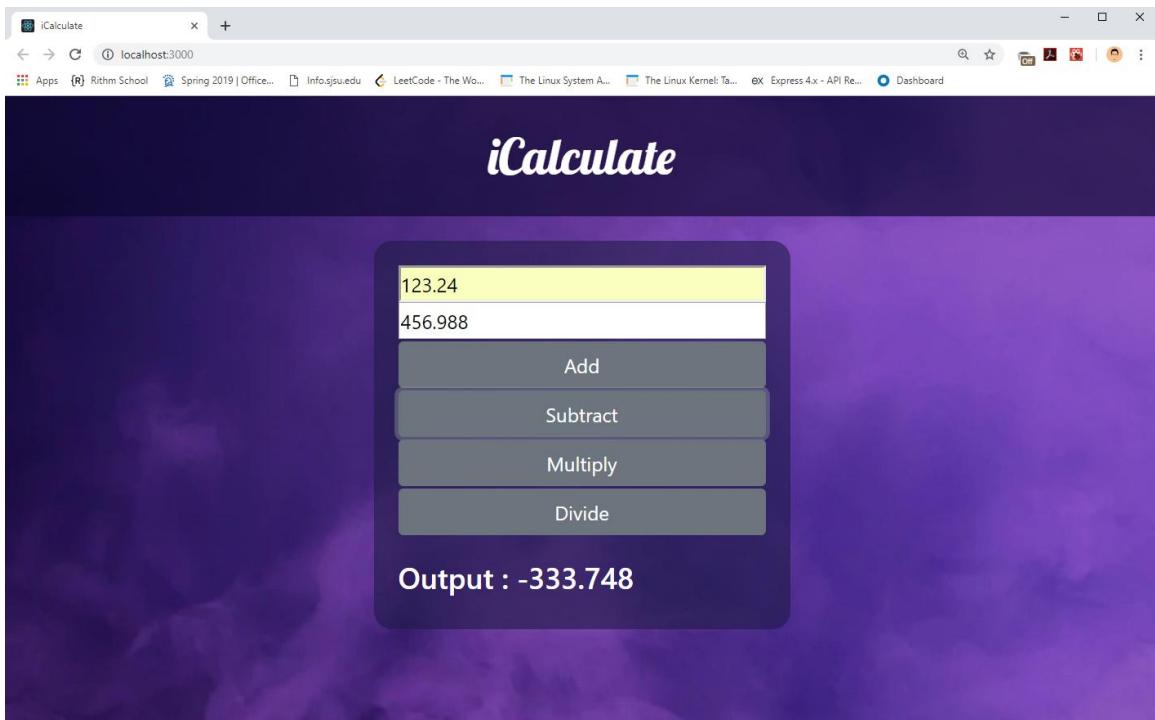
Homepage



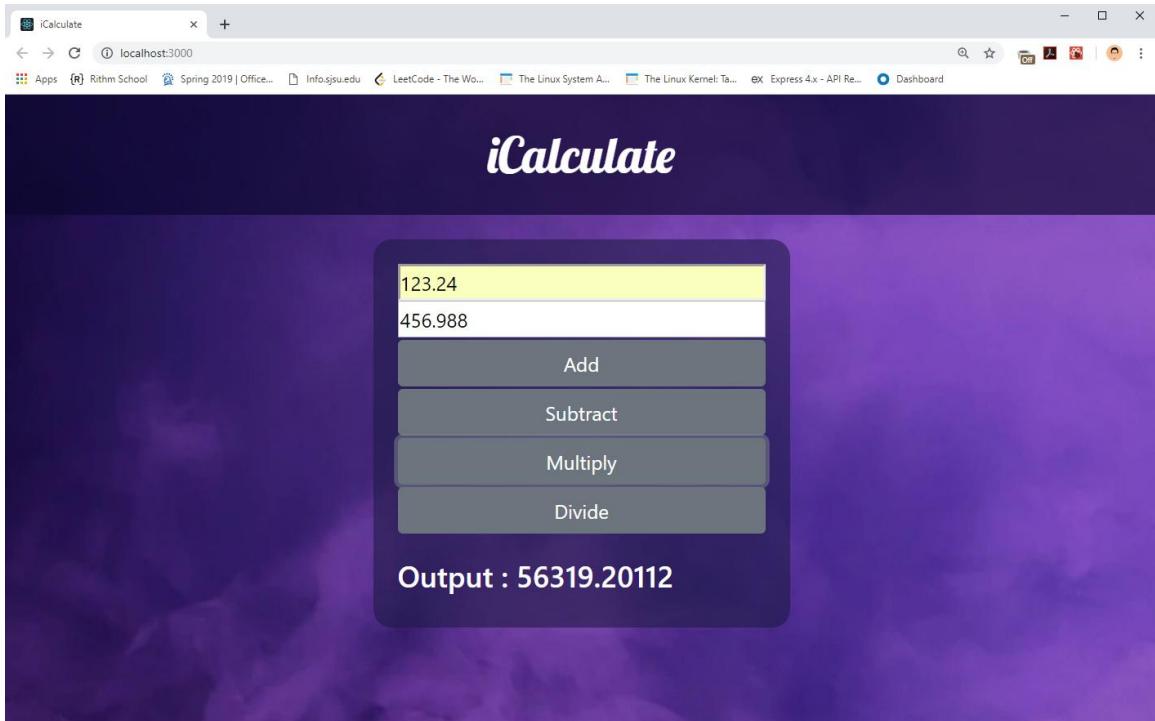
User input and corresponding output for addition operation:



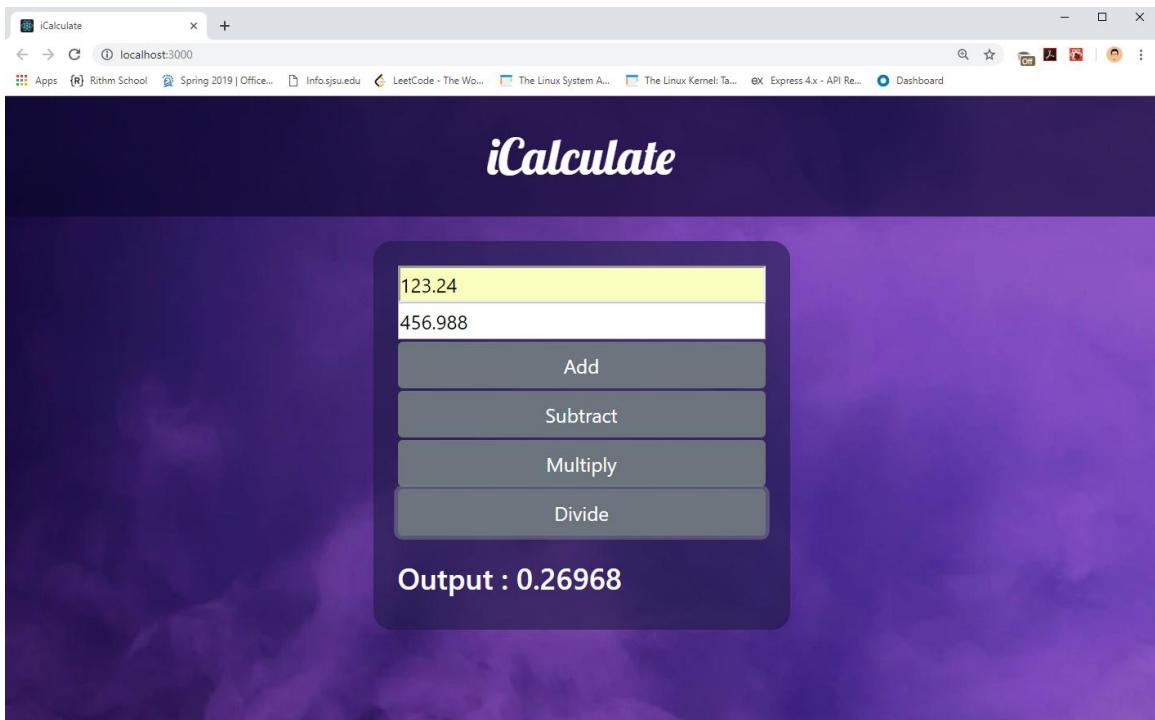
User input and corresponding output for subtraction operation:



User input and corresponding output for multiplication operation:



User input and corresponding output for division operation:



Performance

Test Operations:

Addition

Send Parameters With the Request:					
Name:	Value	URL Encode?	Content-Type	Include Equals?	
op1	123.45	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	
op2	456.78	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	
op	add	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	

Subtraction

Send Parameters With the Request:					
Name:	Value	URL Encode?	Content-Type	Include Equals?	
op1	123.45	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	
op2	456.78	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	
op	sub	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	

Multiplication

Send Parameters With the Request:					
Name:	Value	URL Encode?	Content-Type	Include Equals?	
op1	123.45	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	
op2	456.78	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	
op	mul	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	

Division

Send Parameters With the Request:					
Name:	Value	URL Encode?	Content-Type	Include Equals?	
op1	123.45	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	
op2	456.78	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	
op	div	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>	

Test Case 1: Invoking 1000 random calls:

Thread Group

Name: Users
Comments:
-Action to be taken after a Sampler error
 Continue Start Next Thread Loop Stop Thread Stop Test Stop Test Now

Thread Properties
Number of Threads (users): 1
Ramp-Up Period (in seconds): 1
Loop Count: Forever 1000

Operation = addition, average = 1 ms

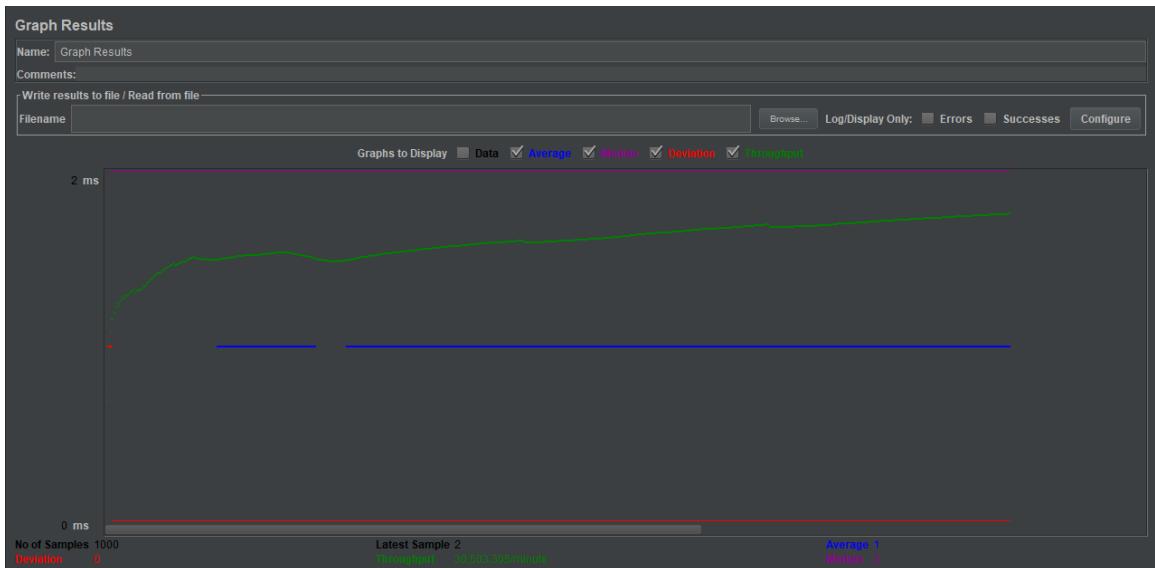
View Results in Table

Name: View Results in Table
Comments:
-Write results to file / Read from file
Filename

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

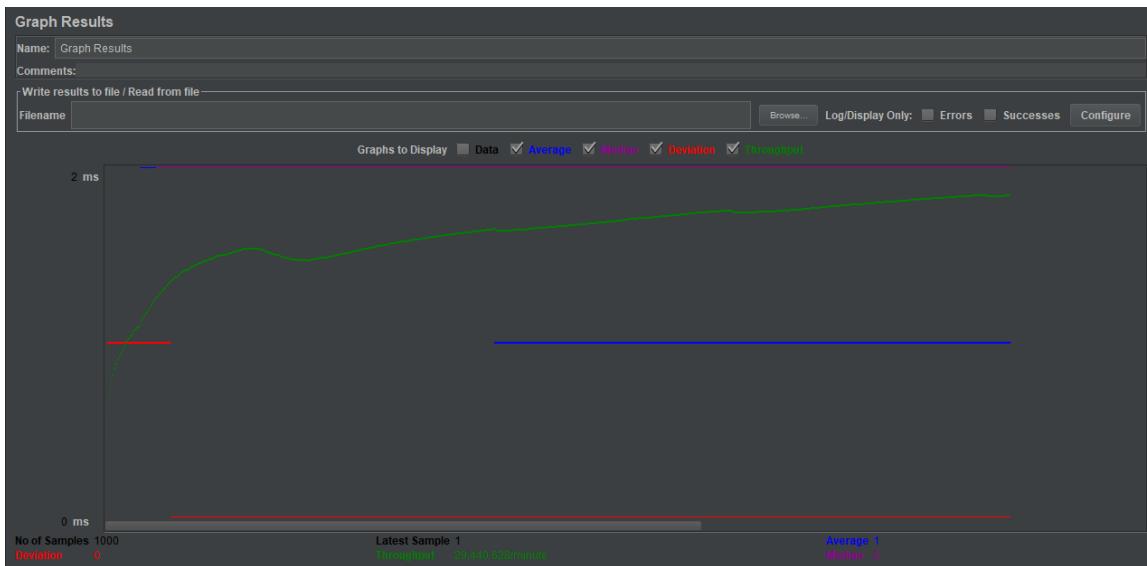
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
1	02:01:23.490	Users 1-1	HTTP Request	5	Success	278	231	5	3
2	02:01:23.496	Users 1-1	HTTP Request	3	Success	278	231	3	1
3	02:01:23.499	Users 1-1	HTTP Request	2	Success	278	231	2	1
4	02:01:23.502	Users 1-1	HTTP Request	2	Success	278	231	2	0
5	02:01:23.504	Users 1-1	HTTP Request	2	Success	278	231	2	1
6	02:01:23.506	Users 1-1	HTTP Request	2	Success	278	231	2	1
7	02:01:23.509	Users 1-1	HTTP Request	2	Success	278	231	2	0
8	02:01:23.511	Users 1-1	HTTP Request	3	Success	278	231	3	2
9	02:01:23.514	Users 1-1	HTTP Request	2	Success	278	231	2	1
10	02:01:23.517	Users 1-1	HTTP Request	2	Success	278	231	2	1
11	02:01:23.519	Users 1-1	HTTP Request	2	Success	278	231	2	1
12	02:01:23.522	Users 1-1	HTTP Request	2	Success	278	231	2	0
13	02:01:23.524	Users 1-1	HTTP Request	2	Success	278	231	2	1
14	02:01:23.527	Users 1-1	HTTP Request	2	Success	278	231	2	1
15	02:01:23.530	Users 1-1	HTTP Request	2	Success	278	231	2	1
16	02:01:23.532	Users 1-1	HTTP Request	2	Success	278	231	2	1
17	02:01:23.534	Users 1-1	HTTP Request	2	Success	278	231	2	1
18	02:01:23.537	Users 1-1	HTTP Request	2	Success	278	231	2	1
19	02:01:23.540	Users 1-1	HTTP Request	2	Success	278	231	2	1
20	02:01:23.542	Users 1-1	HTTP Request	2	Success	278	231	2	1
21	02:01:23.544	Users 1-1	HTTP Request	3	Success	278	231	3	1
22	02:01:23.547	Users 1-1	HTTP Request	2	Success	278	231	2	1
23	02:01:23.549	Users 1-1	HTTP Request	3	Success	278	231	3	1
24	02:01:23.552	Users 1-1	HTTP Request	2	Success	278	231	2	1
25	02:01:23.554	Users 1-1	HTTP Request	3	Success	278	231	3	1

Scroll automatically? Child samples? No of Samples 1000 Latest Sample 2 Average 1 Deviation 0



Operation = subtraction, average = 1 ms

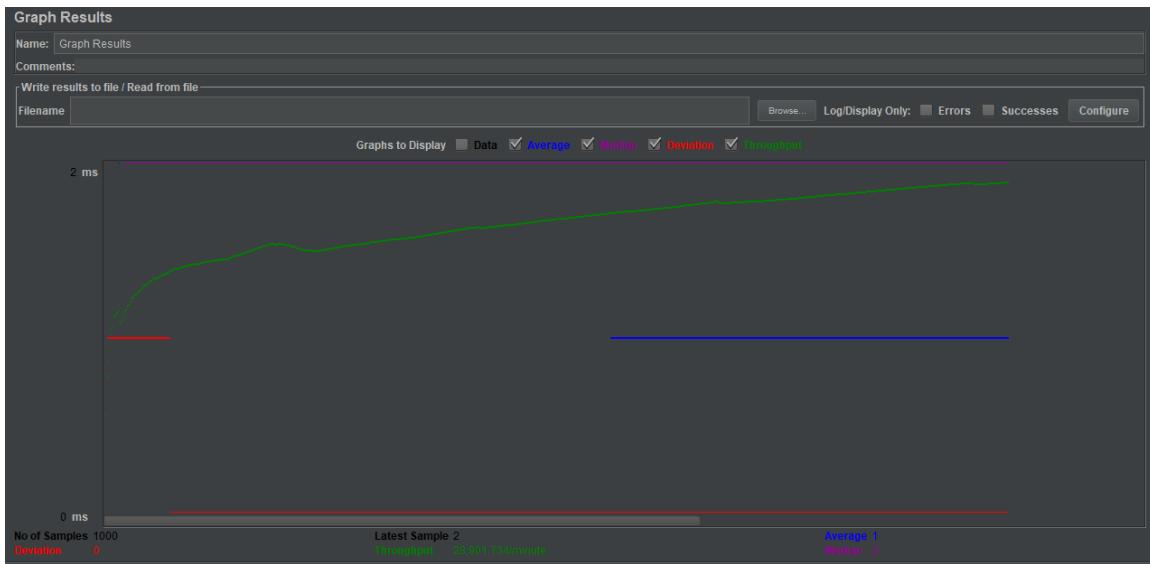
View Results in Table										
Name: View Results in Table										
Comments:										
Write results to file / Read from file										
Filename										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	02:02:18.578	Users 1-1	HTTP Request	8	Success	278	231	8	5	
2	02:02:18.587	Users 1-1	HTTP Request	5	Success	278	231	5	1	
3	02:02:18.592	Users 1-1	HTTP Request	4	Success	278	231	4	2	
4	02:02:18.597	Users 1-1	HTTP Request	3	Success	278	231	3	1	
5	02:02:18.600	Users 1-1	HTTP Request	4	Success	278	231	4	1	
6	02:02:18.604	Users 1-1	HTTP Request	3	Success	278	231	3	2	
7	02:02:18.608	Users 1-1	HTTP Request	3	Success	278	231	3	1	
8	02:02:18.612	Users 1-1	HTTP Request	3	Success	278	231	3	1	
9	02:02:18.615	Users 1-1	HTTP Request	3	Success	278	231	3	2	
10	02:02:18.619	Users 1-1	HTTP Request	9	Success	278	231	3	1	
11	02:02:18.622	Users 1-1	HTTP Request	3	Success	278	231	3	1	
12	02:02:18.626	Users 1-1	HTTP Request	3	Success	278	231	3	1	
13	02:02:18.630	Users 1-1	HTTP Request	2	Success	278	231	2	1	
14	02:02:18.633	Users 1-1	HTTP Request	3	Success	278	231	3	1	
15	02:02:18.636	Users 1-1	HTTP Request	3	Success	278	231	3	1	
16	02:02:18.639	Users 1-1	HTTP Request	3	Success	278	231	3	2	
17	02:02:18.643	Users 1-1	HTTP Request	3	Success	278	231	3	1	
18	02:02:18.646	Users 1-1	HTTP Request	3	Success	278	231	3	1	
19	02:02:18.649	Users 1-1	HTTP Request	3	Success	278	231	3	1	
20	02:02:18.652	Users 1-1	HTTP Request	3	Success	278	231	3	2	
21	02:02:18.655	Users 1-1	HTTP Request	3	Success	278	231	3	2	
22	02:02:18.659	Users 1-1	HTTP Request	2	Success	278	231	2	1	
23	02:02:18.662	Users 1-1	HTTP Request	2	Success	278	231	2	1	
24	02:02:18.665	Users 1-1	HTTP Request	2	Success	278	231	2	1	
25	02:02:18.668	Users 1-1	HTTP Request	2	Success	278	231	2	1	



Operation = multiplication, average = 1 ms

View Results in Table										
<input type="text"/> Name: View Results in Table <input type="text"/> Comments: <input type="checkbox"/> Write results to file / Read from file <input type="text"/> Filename <input type="button" value="Browse..."/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	02:03:11.411	Users 1-1	HTTP Request	7	✓	278	231	7	3	
2	02:03:11.412	Users 1-1	HTTP Request	3	✓	278	231	3	1	
3	02:03:11.412	Users 1-1	HTTP Request	4	✓	278	231	4	1	
4	02:03:11.427	Users 1-1	HTTP Request	2	✓	278	231	2	1	
5	02:03:11.429	Users 1-1	HTTP Request	3	✓	278	231	3	1	
6	02:03:11.432	Users 1-1	HTTP Request	2	✓	278	231	2	1	
7	02:03:11.435	Users 1-1	HTTP Request	2	✓	278	231	2	1	
8	02:03:11.438	Users 1-1	HTTP Request	3	✓	278	231	3	1	
9	02:03:11.441	Users 1-1	HTTP Request	3	✓	278	231	3	1	
10	02:03:11.444	Users 1-1	HTTP Request	2	✓	278	231	2	1	
11	02:03:11.447	Users 1-1	HTTP Request	2	✓	278	231	2	1	
12	02:03:11.449	Users 1-1	HTTP Request	4	✓	278	231	4	1	
13	02:03:11.453	Users 1-1	HTTP Request	3	✓	278	231	3	2	
14	02:03:11.455	Users 1-1	HTTP Request	3	✓	278	231	3	2	
15	02:03:11.460	Users 1-1	HTTP Request	2	✓	278	231	2	1	
16	02:03:11.462	Users 1-1	HTTP Request	2	✓	278	231	2	1	
17	02:03:11.465	Users 1-1	HTTP Request	8	✓	278	231	8	1	
18	02:03:11.473	Users 1-1	HTTP Request	3	✓	278	231	3	1	
19	02:03:11.477	Users 1-1	HTTP Request	2	✓	278	231	2	1	
20	02:03:11.479	Users 1-1	HTTP Request	2	✓	278	231	2	1	
21	02:03:11.482	Users 1-1	HTTP Request	2	✓	278	231	2	1	
22	02:03:11.484	Users 1-1	HTTP Request	2	✓	278	231	2	1	
23	02:03:11.487	Users 1-1	HTTP Request	3	✓	278	231	3	2	
24	02:03:11.489	Users 1-1	HTTP Request	2	✓	278	231	2	1	
25	02:03:11.492	Users 1-1	HTTP Request	2	✓	278	231	2	1	

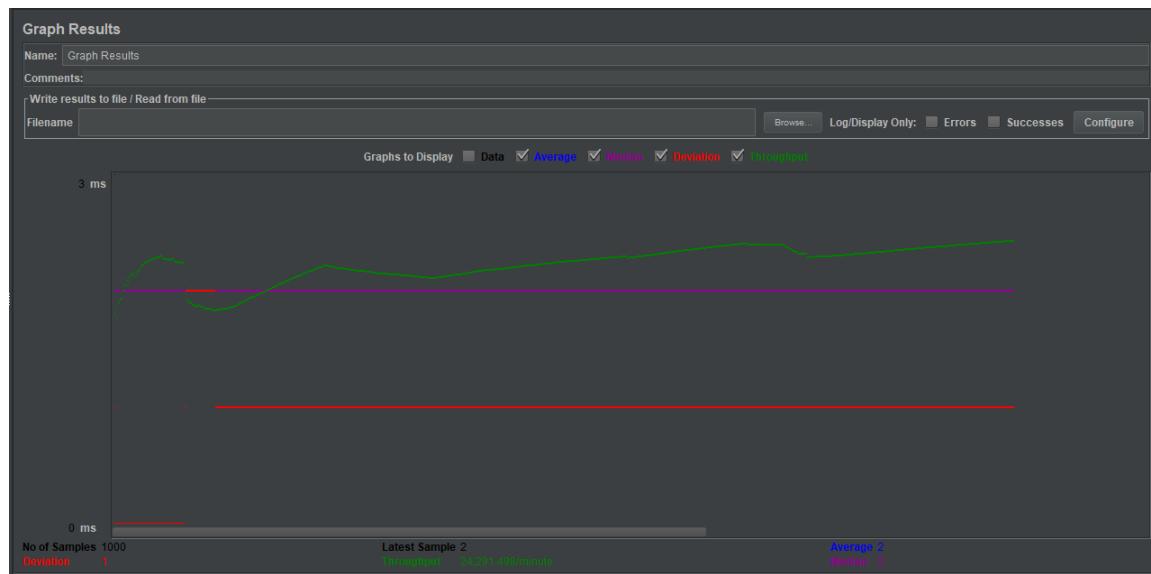
Scroll automatically? Child samples? No of Samples 1000 Latest Sample 2 Average: 1 Deviation: 0



Operation = division, average = 2 ms

View Results in Table										
Name: View Results in Table										
Comments:										
- Write results to file / Read from file										
Filename										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	4	3
1	02:04:00.863	Users 1-1	HTTP Request	4	Success	278	231	4	3	
2	02:04:00.868	Users 1-1	HTTP Request	2	Success	278	231	2	1	
3	02:04:00.871	Users 1-1	HTTP Request	2	Success	278	231	2	1	
4	02:04:00.874	Users 1-1	HTTP Request	3	Success	278	231	2	1	
5	02:04:00.877	Users 1-1	HTTP Request	3	Success	278	231	3	1	
6	02:04:00.880	Users 1-1	HTTP Request	2	Success	278	231	2	1	
7	02:04:00.883	Users 1-1	HTTP Request	2	Success	278	231	2	1	
8	02:04:00.885	Users 1-1	HTTP Request	3	Success	278	231	3	1	
9	02:04:00.888	Users 1-1	HTTP Request	3	Success	278	231	3	1	
10	02:04:00.891	Users 1-1	HTTP Request	3	Success	278	231	3	1	
11	02:04:00.894	Users 1-1	HTTP Request	2	Success	278	231	2	1	
12	02:04:00.896	Users 1-1	HTTP Request	3	Success	278	231	3	1	
13	02:04:00.899	Users 1-1	HTTP Request	2	Success	278	231	2	1	
14	02:04:00.901	Users 1-1	HTTP Request	2	Success	278	231	2	1	
15	02:04:00.904	Users 1-1	HTTP Request	3	Success	278	231	3	1	
16	02:04:00.907	Users 1-1	HTTP Request	2	Success	278	231	2	1	
17	02:04:00.910	Users 1-1	HTTP Request	2	Success	278	231	2	1	
18	02:04:00.912	Users 1-1	HTTP Request	2	Success	278	231	2	1	
19	02:04:00.915	Users 1-1	HTTP Request	2	Success	278	231	2	1	
20	02:04:00.917	Users 1-1	HTTP Request	2	Success	278	231	2	1	
21	02:04:00.920	Users 1-1	HTTP Request	2	Success	278	231	2	1	
22	02:04:00.923	Users 1-1	HTTP Request	2	Success	278	231	2	1	
23	02:04:00.925	Users 1-1	HTTP Request	3	Success	278	231	2	1	
24	02:04:00.928	Users 1-1	HTTP Request	3	Success	278	231	3	0	
25	02:04:00.931	Users 1-1	HTTP Request	2	Success	278	231	2	1	

Scroll automatically? Child samples? No of Samples 1000 Latest Sample 2 Average 2 Deviation 1



Test Case 2: Invoking 5000 random calls:

Thread Group

Name: Users

Comments:

Action to be taken after a Sampler error:

- Continue
- Start Next Thread Loop
- Stop Thread
- Stop Test
- Stop Test Now

Thread Properties

Number of Threads (users): 1

Ramp-Up Period (in seconds): 1

Loop Count: Forever 5000

Operation = addition, average = 2 ms

View Results in Table

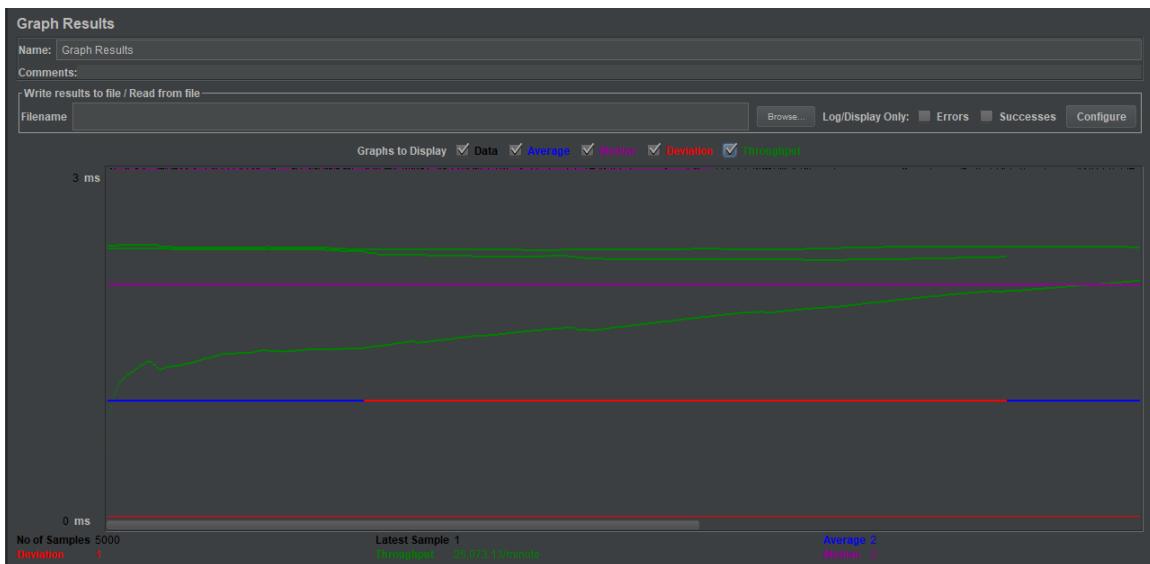
Name: View Results in Table

Comments:

Write results to file / Read from file

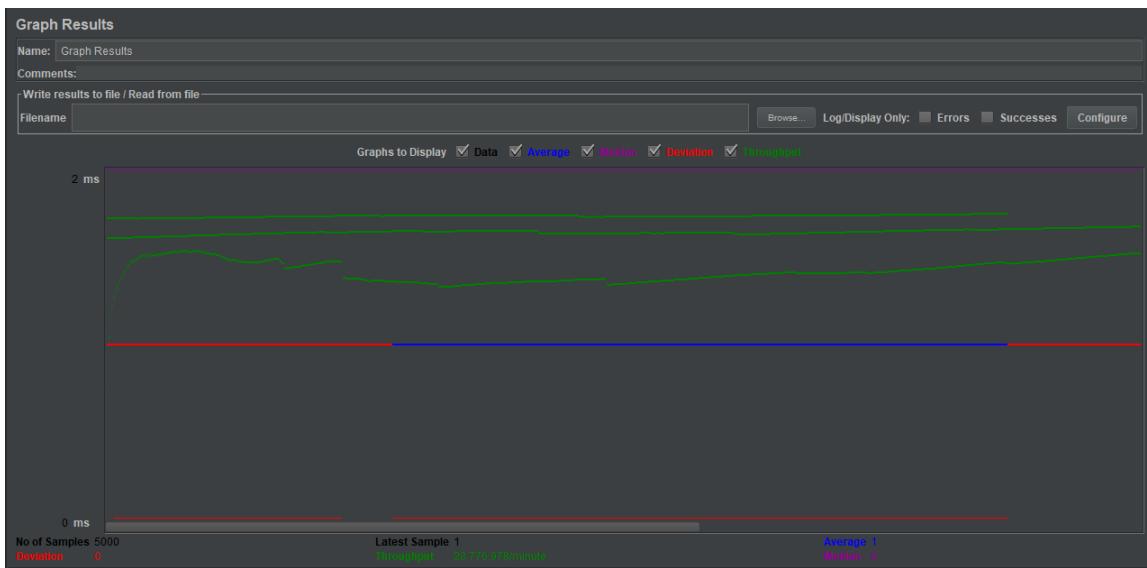
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connected Time(ms)
1	01:23:40.695	Users 1-1	HTTP Request	9	✓	278	231	9	5
2	01:23:40.705	Users 1-1	HTTP Request	5	✓	278	231	5	1
3	01:23:40.711	Users 1-1	HTTP Request	4	✓	278	231	4	2
4	01:23:40.716	Users 1-1	HTTP Request	3	✓	278	231	3	1
5	01:23:40.720	Users 1-1	HTTP Request	3	✓	278	231	3	1
6	01:23:40.724	Users 1-1	HTTP Request	3	✓	278	231	3	2
7	01:23:40.728	Users 1-1	HTTP Request	4	✓	278	231	4	2
8	01:23:40.732	Users 1-1	HTTP Request	4	✓	278	231	4	2
9	01:23:40.737	Users 1-1	HTTP Request	3	✓	278	231	3	2
10	01:23:40.741	Users 1-1	HTTP Request	3	✓	278	231	3	2
11	01:23:40.745	Users 1-1	HTTP Request	3	✓	278	231	3	2
12	01:23:40.749	Users 1-1	HTTP Request	3	✓	278	231	3	1
13	01:23:40.753	Users 1-1	HTTP Request	3	✓	278	231	3	1
14	01:23:40.757	Users 1-1	HTTP Request	3	✓	278	231	3	1
15	01:23:40.761	Users 1-1	HTTP Request	3	✓	278	231	3	1
16	01:23:40.765	Users 1-1	HTTP Request	3	✓	278	231	3	1
17	01:23:40.769	Users 1-1	HTTP Request	3	✓	278	231	3	1
18	01:23:40.772	Users 1-1	HTTP Request	4	✓	278	231	4	2
19	01:23:40.776	Users 1-1	HTTP Request	4	✓	278	231	4	2
20	01:23:40.780	Users 1-1	HTTP Request	4	✓	278	231	4	2
21	01:23:40.784	Users 1-1	HTTP Request	4	✓	278	231	4	2
22	01:23:40.788	Users 1-1	HTTP Request	3	✓	278	231	3	2
23	01:23:40.792	Users 1-1	HTTP Request	3	✓	278	231	3	1
24	01:23:40.796	Users 1-1	HTTP Request	4	✓	278	231	4	1
25	01:23:40.800	Users 1-1	HTTP Request	4	✗	278	231	4	2

Scroll automatically? Child samples? No of Samples 5000 Latest Sample 1 Average 2 Deviation 1



Operation = subtraction, average = 2 ms

View Results in Table										
<input type="checkbox"/> Name: View Results in Table <input type="checkbox"/> Comments: <input type="checkbox"/> Write results to file / Read from file <input type="checkbox"/> Filename: <input type="text"/> <input type="button"/> Browse... <input type="checkbox"/> Log/Display Only. <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button"/> Configure										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	01:25:08.101	Users 1-1	HTTP Request	5	✓	278	231	5	3	
2	01:25:08.106	Users 1-1	HTTP Request	2	✓	278	231	2	1	
3	01:25:08.109	Users 1-1	HTTP Request	3	✓	278	231	3	0	
4	01:25:08.113	Users 1-1	HTTP Request	2	✓	278	231	1	1	
5	01:25:08.115	Users 1-1	HTTP Request	2	✓	278	231	2	1	
6	01:25:08.117	Users 1-1	HTTP Request	2	✓	278	231	2	1	
7	01:25:08.120	Users 1-1	HTTP Request	2	✓	278	231	2	1	
8	01:25:08.122	Users 1-1	HTTP Request	2	✓	278	231	2	1	
9	01:25:08.124	Users 1-1	HTTP Request	3	✓	278	231	3	1	
10	01:25:08.127	Users 1-1	HTTP Request	2	✓	278	231	2	1	
11	01:25:08.130	Users 1-1	HTTP Request	1	✓	278	231	1	0	
12	01:25:08.132	Users 1-1	HTTP Request	2	✓	278	231	2	1	
13	01:25:08.134	Users 1-1	HTTP Request	3	✓	278	231	3	1	
14	01:25:08.137	Users 1-1	HTTP Request	2	✓	278	231	2	1	
15	01:25:08.139	Users 1-1	HTTP Request	2	✓	278	231	2	1	
16	01:25:08.141	Users 1-1	HTTP Request	2	✓	278	231	2	1	
17	01:25:08.144	Users 1-1	HTTP Request	2	✓	278	231	2	0	
18	01:25:08.146	Users 1-1	HTTP Request	2	✓	278	231	2	1	
19	01:25:08.148	Users 1-1	HTTP Request	2	✓	278	231	2	1	
20	01:25:08.150	Users 1-1	HTTP Request	3	✓	278	231	3	1	
21	01:25:08.153	Users 1-1	HTTP Request	2	✓	278	231	2	1	
22	01:25:08.155	Users 1-1	HTTP Request	2	✓	278	231	2	1	
23	01:25:08.157	Users 1-1	HTTP Request	2	✓	278	231	2	1	
24	01:25:08.160	Users 1-1	HTTP Request	1	✓	278	231	1	0	
25	01:25:08.162	Users 1-1	HTTP Request	2	✓	278	231	2	1	



Operation = multiplication, average = 1 ms

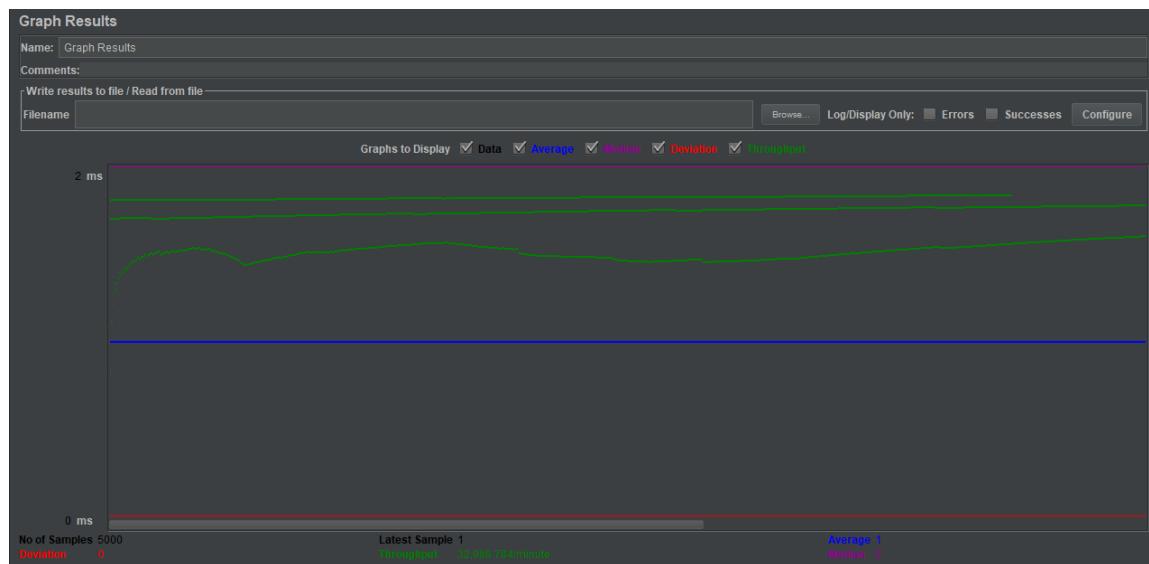
View Results in Table

Name: View Results in Table
Comments:
- Write results to file / Read from file —
Filename:

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

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
1	01:26:11.090	Users 1-1	HTTP Request	4	Success	278	231	4	2
2	01:26:11.094	Users 1-1	HTTP Request	2	Success	278	231	2	1
3	01:26:11.097	Users 1-1	HTTP Request	2	Success	278	231	2	0
4	01:26:11.099	Users 1-1	HTTP Request	2	Success	278	231	2	1
5	01:26:11.101	Users 1-1	HTTP Request	2	Success	278	231	2	1
6	01:26:11.104	Users 1-1	HTTP Request	1	Success	278	231	1	0
7	01:26:11.105	Users 1-1	HTTP Request	2	Success	278	231	2	0
8	01:26:11.108	Users 1-1	HTTP Request	2	Success	278	231	2	1
9	01:26:11.110	Users 1-1	HTTP Request	2	Success	278	231	2	1
10	01:26:11.112	Users 1-1	HTTP Request	2	Success	278	231	2	1
11	01:26:11.115	Users 1-1	HTTP Request	2	Success	278	231	2	1
12	01:26:11.117	Users 1-1	HTTP Request	2	Success	278	231	2	1
13	01:26:11.119	Users 1-1	HTTP Request	2	Success	278	231	2	1
14	01:26:11.122	Users 1-1	HTTP Request	1	Success	278	231	1	0
15	01:26:11.124	Users 1-1	HTTP Request	2	Success	278	231	2	0
16	01:26:11.126	Users 1-1	HTTP Request	2	Success	278	231	2	1
17	01:26:11.128	Users 1-1	HTTP Request	2	Success	278	231	2	1
18	01:26:11.130	Users 1-1	HTTP Request	2	Success	278	231	2	1
19	01:26:11.133	Users 1-1	HTTP Request	2	Success	278	231	2	1
20	01:26:11.135	Users 1-1	HTTP Request	2	Success	278	231	2	1
21	01:26:11.137	Users 1-1	HTTP Request	2	Success	278	231	2	1
22	01:26:11.139	Users 1-1	HTTP Request	2	Success	278	231	2	1
23	01:26:11.141	Users 1-1	HTTP Request	2	Success	278	231	2	1
24	01:26:11.143	Users 1-1	HTTP Request	2	Success	278	231	2	1
25	01:26:11.146	Users 1-1	HTTP Request	1	Success	278	231	1	0

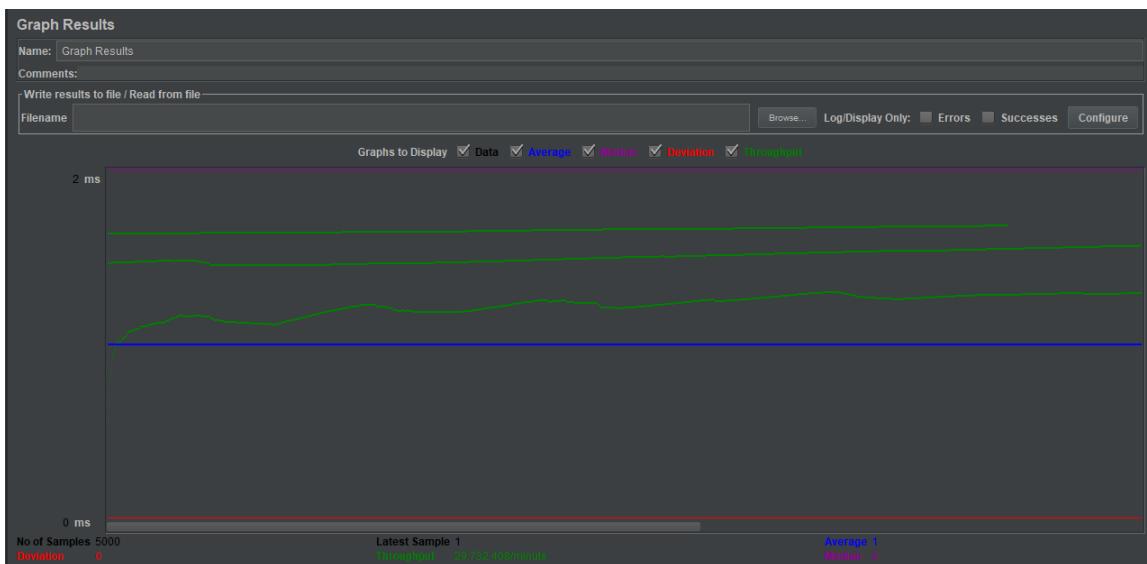
Scroll automatically? Child samples? No of Samples 5000 Latest Sample 1 Average 1 Deviation 0



Operation = division, average = 1 ms

View Results in Table										
<input type="checkbox"/> Name: View Results in Table <input type="checkbox"/> Comments: <input type="checkbox"/> Write results to file / Read from file <input type="checkbox"/> Filename: <input type="text"/> <input type="button"/> Browse... <input type="checkbox"/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button"/> Configure										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	01:27:10.959	Users 1-1	HTTP Request	6	✓	278	231	6	4	
2	01:27:10.966	Users 1-1	HTTP Request	3	✓	278	231	3	1	
3	01:27:10.970	Users 1-1	HTTP Request	2	✓	278	231	2	1	
4	01:27:10.972	Users 1-1	HTTP Request	3	✓	278	231	3	2	
5	01:27:10.975	Users 1-1	HTTP Request	3	✓	278	231	3	1	
6	01:27:10.979	Users 1-1	HTTP Request	2	✓	278	231	2	1	
7	01:27:10.982	Users 1-1	HTTP Request	2	✓	278	231	2	1	
8	01:27:10.985	Users 1-1	HTTP Request	2	✓	278	231	2	1	
9	01:27:10.987	Users 1-1	HTTP Request	3	✓	278	231	3	1	
10	01:27:10.990	Users 1-1	HTTP Request	3	✓	278	231	3	1	
11	01:27:10.993	Users 1-1	HTTP Request	3	✓	278	231	3	1	
12	01:27:10.996	Users 1-1	HTTP Request	3	✓	278	231	3	1	
13	01:27:10.999	Users 1-1	HTTP Request	3	✓	278	231	3	1	
14	01:27:11.003	Users 1-1	HTTP Request	3	✓	278	231	3	1	
15	01:27:11.006	Users 1-1	HTTP Request	3	✓	278	231	3	1	
16	01:27:11.009	Users 1-1	HTTP Request	3	✓	278	231	3	2	
17	01:27:11.012	Users 1-1	HTTP Request	3	✓	278	231	3	2	
18	01:27:11.015	Users 1-1	HTTP Request	3	✓	278	231	3	1	
19	01:27:11.018	Users 1-1	HTTP Request	3	✓	278	231	3	1	
20	01:27:11.021	Users 1-1	HTTP Request	3	✓	278	231	3	1	
21	01:27:11.024	Users 1-1	HTTP Request	3	✓	278	231	3	1	
22	01:27:11.027	Users 1-1	HTTP Request	3	✓	278	231	3	1	
23	01:27:11.030	Users 1-1	HTTP Request	2	✓	278	231	2	1	
24	01:27:11.033	Users 1-1	HTTP Request	2	✓	278	231	2	1	
25	01:27:11.036	Users 1-1	HTTP Request	2	✓	278	231	2	1	

Scroll automatically? Child samples?
 No of Samples 5000 Latest Sample 1 Average 1 Deviation 0



Test Case 3: Invoking 1000 calls from 100 concurrent users:

Thread Group

Name: Users
Comments:
Action to be taken after a Sampler error: Continue

Thread Properties
Number of Threads (users): 100
Ramp-Up Period (in seconds): 1
Loop Count: Forever 1000

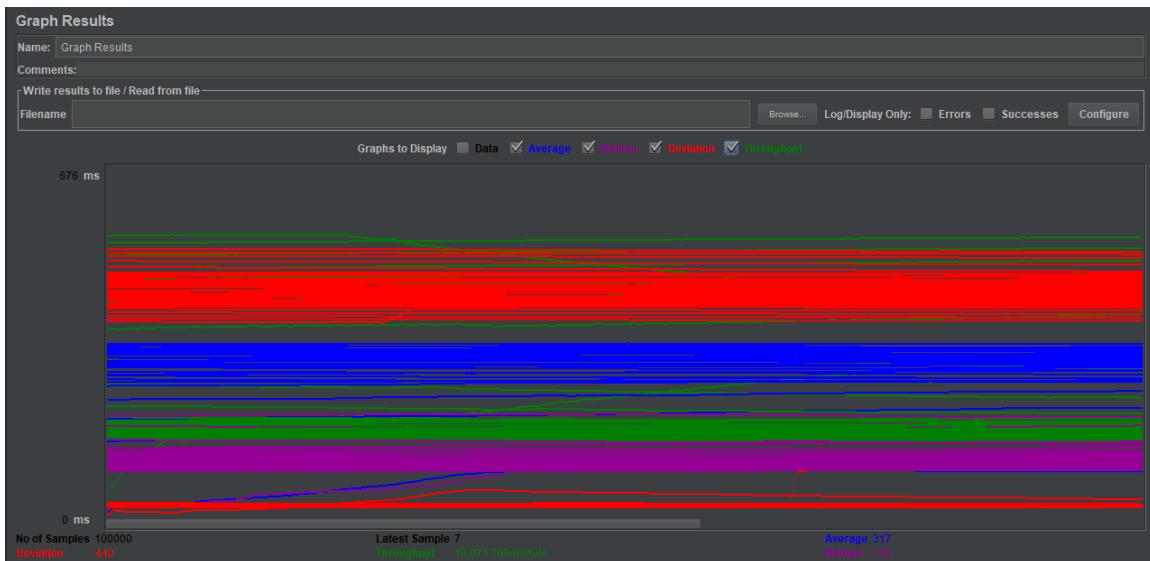
Operation = addition, average = 317 ms

View Results in Table

Name: View Results in Table
Comments:
Write results to file / Read from file
Filename

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
1	01:29:54.136	Users 1-1	HTTP Request	12	✓	278	231	12	7
2	01:29:54.146	Users 1-2	HTTP Request	4	✓	278	231	4	1
3	01:29:54.149	Users 1-1	HTTP Request	6	✓	278	231	6	1
4	01:29:54.150	Users 1-2	HTTP Request	9	✓	278	231	9	2
5	01:29:54.155	Users 1-1	HTTP Request	24	✓	278	231	24	2
6	01:29:54.159	Users 1-2	HTTP Request	27	✓	278	231	27	2
7	01:29:54.164	Users 1-3	HTTP Request	36	✓	278	231	36	2
8	01:29:54.175	Users 1-4	HTTP Request	33	✓	278	231	33	2
9	01:29:54.179	Users 1-1	HTTP Request	32	✓	278	231	32	2
10	01:29:54.187	Users 1-2	HTTP Request	28	✓	278	231	28	3
11	01:29:54.195	Users 1-5	HTTP Request	23	✓	278	231	23	1
12	01:29:54.200	Users 1-3	HTTP Request	26	✓	278	231	26	2
13	01:29:54.208	Users 1-4	HTTP Request	19	✓	278	231	19	1
14	01:29:54.210	Users 1-6	HTTP Request	18	✓	278	231	18	2
15	01:29:54.212	Users 1-1	HTTP Request	20	✓	278	231	20	1
16	01:29:54.216	Users 1-2	HTTP Request	18	✓	278	231	18	1
17	01:29:54.219	Users 1-5	HTTP Request	16	✓	278	231	16	1
18	01:29:54.223	Users 1-9	HTTP Request	18	✓	278	231	18	2
19	01:29:54.226	Users 1-3	HTTP Request	15	✓	278	231	15	1
20	01:29:54.227	Users 1-4	HTTP Request	15	✓	278	231	15	1
21	01:29:54.228	Users 1-6	HTTP Request	16	✓	278	231	16	2
22	01:29:54.231	Users 1-10	HTTP Request	14	✓	278	231	14	1
23	01:29:54.232	Users 1-1	HTTP Request	16	✓	278	231	16	1
24	01:29:54.235	Users 1-2	HTTP Request	14	✓	278	231	14	1
25	01:29:54.236	Users 1-5	HTTP Request	14	✗	278	231	14	1

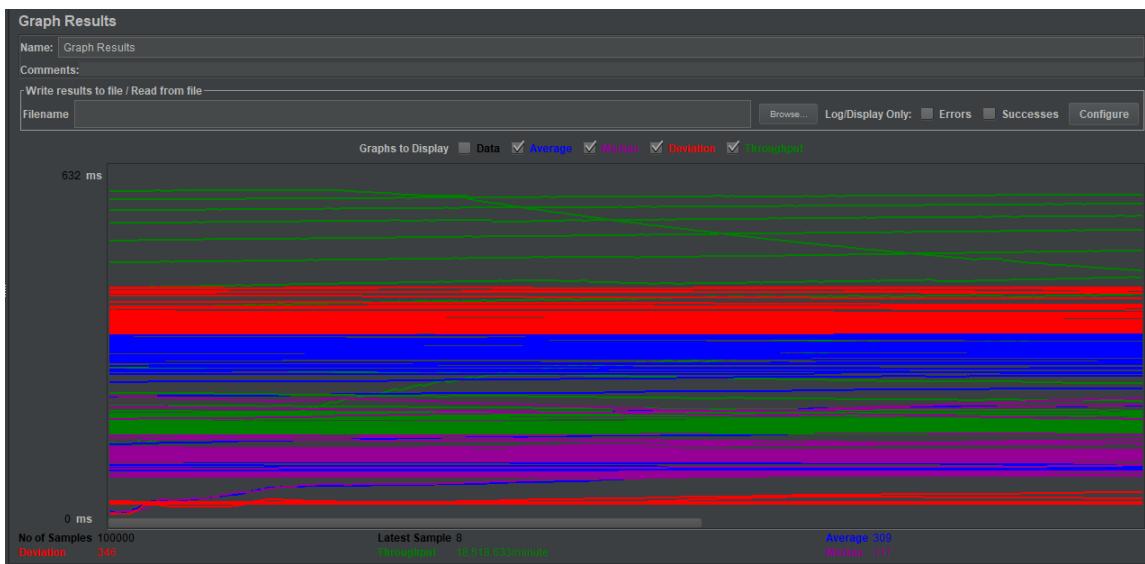
Scroll automatically? Child samples? No of Samples 100000 Latest Sample 7 Average 317 Deviation 440



Operation = subtraction, average = 309 ms

View Results in Table										
<input type="text"/> Name: View Results in Table <input type="text"/> Comments: <input type="checkbox"/> Write results to file / Read from file <input type="text"/> Filename <input type="button" value="Browse..."/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	01:36:45.781	Users 1-1	HTTP Request	10	✓	278	231	10	4	
2	01:36:45.791	Users 1-2	HTTP Request	4	✓	278	231	4	2	
3	01:36:45.791	Users 1-1	HTTP Request	4	✓	278	231	4	1	
4	01:36:45.793	Users 1-2	HTTP Request	3	✓	278	231	3	1	
5	01:36:45.795	Users 1-1	HTTP Request	7	✓	278	231	7	1	
6	01:36:45.797	Users 1-2	HTTP Request	8	✓	278	231	8	0	
7	01:36:45.802	Users 1-3	HTTP Request	5	✓	278	231	5	1	
8	01:36:45.802	Users 1-1	HTTP Request	6	✓	278	231	6	1	
9	01:36:45.805	Users 1-2	HTTP Request	5	✓	278	231	5	1	
10	01:36:45.808	Users 1-3	HTTP Request	6	✓	278	231	6	1	
11	01:36:45.808	Users 1-1	HTTP Request	7	✓	278	231	7	1	
12	01:36:45.810	Users 1-2	HTTP Request	7	✓	278	231	7	1	
13	01:36:45.812	Users 1-4	HTTP Request	8	✓	278	231	8	1	
14	01:36:45.814	Users 1-3	HTTP Request	9	✓	278	231	9	2	
15	01:36:45.815	Users 1-1	HTTP Request	9	✓	278	231	9	1	
16	01:36:45.817	Users 1-2	HTTP Request	9	✓	278	231	9	2	
17	01:36:45.820	Users 1-4	HTTP Request	9	✓	278	231	9	2	
18	01:36:45.823	Users 1-5	HTTP Request	7	✓	278	231	7	2	
19	01:36:45.823	Users 1-3	HTTP Request	8	✓	278	231	8	1	
20	01:36:45.825	Users 1-1	HTTP Request	12	✓	278	231	12	1	
21	01:36:45.827	Users 1-2	HTTP Request	14	✓	278	231	14	1	
22	01:36:45.830	Users 1-4	HTTP Request	13	✓	278	231	13	1	
23	01:36:45.831	Users 1-5	HTTP Request	14	✓	278	231	14	1	
24	01:36:45.832	Users 1-3	HTTP Request	18	✓	278	231	18	1	
25	01:36:45.835	Users 1-6	HTTP Request	21	✗	278	231	21	1	

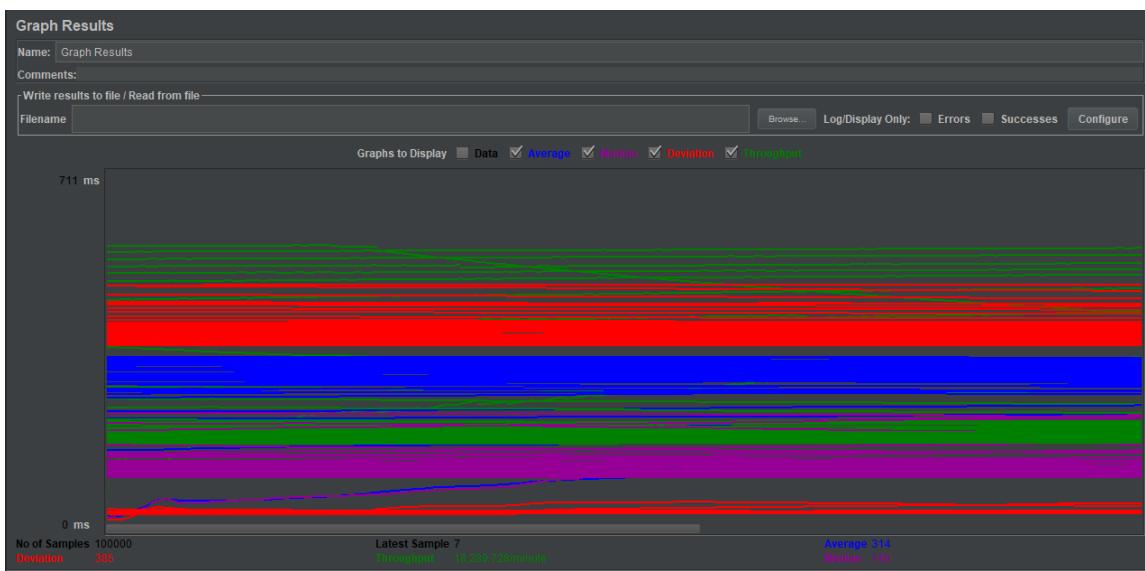
Scroll automatically? Child samples? No of Samples 100000 Latest Sample 8 Average 309 Deviation 346



Operation = multiplication, average = 314 ms

View Results in Table										
<input type="checkbox"/> Name: View Results in Table <input type="checkbox"/> Comments: <input type="checkbox"/> Write results to file / Read from file <input type="checkbox"/> Filename: <input type="text"/> <input type="button"/> Browse... <input type="checkbox"/> Log/Display Only. <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button"/> Configure										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	01:43:11.972	Users 1-1	HTTP Request	14	✓	278	231	14	6	
2	01:43:11.979	Users 1-2	HTTP Request	9	✓	278	231	9	2	
3	01:43:11.988	Users 1-1	HTTP Request	7	✓	278	231	7	2	
4	01:43:11.988	Users 1-2	HTTP Request	7	✓	278	231	7	2	
5	01:43:11.993	Users 1-1	HTTP Request	7	✓	278	231	7	2	
6	01:43:11.995	Users 1-3	HTTP Request	12	✓	278	231	12	2	
7	01:43:11.996	Users 1-2	HTTP Request	13	✓	278	231	13	2	
8	01:43:12.001	Users 1-1	HTTP Request	10	✓	278	231	10	1	
9	01:43:12.004	Users 1-4	HTTP Request	15	✓	278	231	15	2	
10	01:43:12.008	Users 1-3	HTTP Request	12	✓	278	231	12	1	
11	01:43:12.009	Users 1-2	HTTP Request	14	✓	278	231	14	1	
12	01:43:12.012	Users 1-1	HTTP Request	12	✓	278	231	12	2	
13	01:43:12.016	Users 1-5	HTTP Request	12	✓	278	231	12	2	
14	01:43:12.019	Users 1-4	HTTP Request	10	✓	278	231	10	1	
15	01:43:12.020	Users 1-3	HTTP Request	16	✓	278	231	16	1	
16	01:43:12.023	Users 1-2	HTTP Request	14	✓	278	231	14	2	
17	01:43:12.024	Users 1-1	HTTP Request	16	✓	278	231	16	2	
18	01:43:12.027	Users 1-6	HTTP Request	20	✓	278	231	20	1	
19	01:43:12.029	Users 1-5	HTTP Request	22	✓	278	231	22	2	
20	01:43:12.030	Users 1-4	HTTP Request	25	✓	278	231	25	2	
21	01:43:12.036	Users 1-3	HTTP Request	26	✓	278	231	26	2	
22	01:43:12.038	Users 1-2	HTTP Request	29	✓	278	231	29	1	
23	01:43:12.041	Users 1-8	HTTP Request	27	✓	278	231	27	2	
24	01:43:12.041	Users 1-1	HTTP Request	30	✓	278	231	30	1	
25	01:43:12.046	Users 1-1	HTTP Request	36	✓	278	231	36	2	

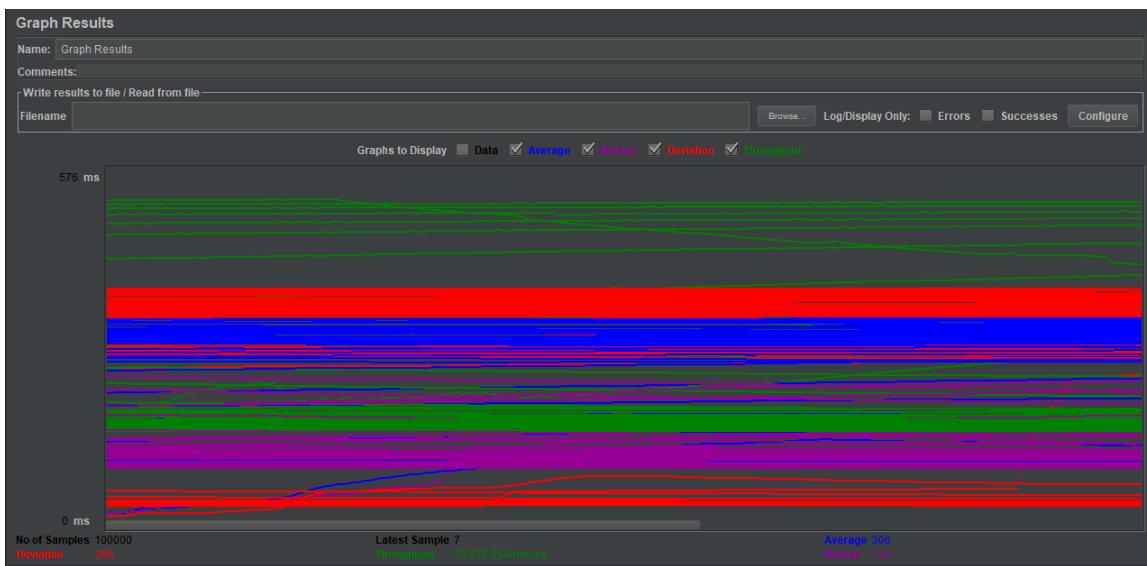
Scroll automatically? Child samples?
 No of Samples 100000 Latest Sample 7 Average 314 Deviation 385



Operation = division, average = 306ms

View Results in Table										
Name: <input type="text" value="View Results in Table"/> Comments: <input type="text"/> <input type="checkbox"/> Write results to file / Read from file <input type="button" value="Browse..."/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	01:50:14.000	Users 1-1	HTTP Request	10	✓	278	231	10	6	
2	01:50:14.010	Users 1-1	HTTP Request	5	✓	278	231	5	2	
3	01:50:14.008	Users 1-2	HTTP Request	12	✓	278	231	12	6	
4	01:50:14.016	Users 1-1	HTTP Request	7	✓	278	231	7	1	
5	01:50:14.019	Users 1-3	HTTP Request	6	✓	278	231	6	1	
6	01:50:14.021	Users 1-2	HTTP Request	5	✓	278	231	5	1	
7	01:50:14.023	Users 1-1	HTTP Request	8	✓	278	231	8	1	
8	01:50:14.026	Users 1-3	HTTP Request	7	✓	278	231	7	2	
9	01:50:14.027	Users 1-2	HTTP Request	7	✓	278	231	7	1	
10	01:50:14.032	Users 1-1	HTTP Request	5	✓	278	231	5	1	
11	01:50:14.034	Users 1-3	HTTP Request	6	✓	278	231	6	1	
12	01:50:14.035	Users 1-2	HTTP Request	8	✓	278	231	8	1	
13	01:50:14.037	Users 1-4	HTTP Request	8	✓	278	231	8	1	
14	01:50:14.038	Users 1-1	HTTP Request	8	✓	278	231	8	1	
15	01:50:14.041	Users 1-3	HTTP Request	11	✓	278	231	11	1	
16	01:50:14.044	Users 1-2	HTTP Request	9	✓	278	231	9	1	
17	01:50:14.045	Users 1-4	HTTP Request	9	✓	278	231	9	2	
18	01:50:14.046	Users 1-1	HTTP Request	10	✓	278	231	10	2	
19	01:50:14.049	Users 1-5	HTTP Request	10	✓	278	231	10	2	
20	01:50:14.052	Users 1-3	HTTP Request	8	✓	278	231	8	1	
21	01:50:14.053	Users 1-2	HTTP Request	11	✓	278	231	11	1	
22	01:50:14.054	Users 1-4	HTTP Request	13	✓	278	231	13	1	
23	01:50:14.056	Users 1-1	HTTP Request	17	✓	278	231	17	1	
24	01:50:14.059	Users 1-5	HTTP Request	20	✓	278	231	20	2	
25	01:50:14.061	Users 1-3	HTTP Request	19	✓	278	231	19	1	

Scroll automatically? Child samples? No of Samples 100000 Latest Sample 7 Average 306 Deviation 306



Conclusion

The throughput is almost same for test cases 1,2 while that for test case 3 is less due to more concurrent requests.

The response time is being between 1-2 ms for test case 1 and 2. This implies the despite increasing the number of requests in thousands, the change in average response time is nearly the same. The average rises drastically for concurrent users. Hence it can be said that it highly affected by concurrency.

The type of operation does not matter in any case as all operations produce nearly same average.

Part 2 - Application

Introduction

This project aims to implement the major functionalities of *Canvas* application which is a web portal accessible to students and the faculty of San Jose State University. It will basically be a virtual classroom involving process like basic sign up, searching and enrolling courses, submitting quiz and assignments, grading of the same, making announcements, thus connecting the student and faculty's world to a common accessible platform.

System Design

The application is made to revolve around two types of its users which are faculty and student. Detailed information and functionalities can be inferred from the following: -

Assumptions

- Every user must have a unique SJSU ID prior to using this application.
- All input fields are validated.
- Profile image can be updated by an image URL.
- Passwords are encrypted using bcrypt.
- Permission code is based on timestamp so that it is unique.

Functionalities

- Setting up and maintaining user account and profile information.
- Faculty can create a course.
- Student can search among all the available courses.
- Student can enroll, waitlist or drop from a course.
- User can view students enrolled in a class.
- Faculty can remove a student from a class; generate unique permission code for waitlisted students.
- Faculty can make and view announcement for corresponding course; students can view them.
- Faculties can create and view quiz. Students can take the quiz and have their grades displayed under Grades.
- Faculty can upload lecture notes as files, which the student can download.

- Faculty can upload assignments. Student can submit the assignment which will be graded by Faculty and the score will be displayed under Grades.

Database Design

User

sys.user						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
◇ about	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ city	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ cno	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ company	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ country	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ email	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ gender	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ hometown	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ id	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ image	varchar(1024)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ language	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ name	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ password	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ role	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ school	varchar(45)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Course

sys.course						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
◇ ccap	int(11)		NO			select,insert,update,references
◇ cdept	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ cdesc	varchar(2048)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ cid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ cname	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ croom	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ cterm	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ cwait	int(11)		NO			select,insert,update,references
◇ fid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Class

sys.class						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
◇ cid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ fid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ sid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ stat	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Announcement

sys.announcement						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
adesc	varchar(2048)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
aname	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
atime	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
cid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Quiz

sys.quiz						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
cid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
cor1	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
cor2	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
d1	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
d2	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
op11	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
op12	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
op13	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
op14	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
op21	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
op22	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
op23	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
op24	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
q1	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
q2	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
qid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
qname	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Grade

sys.grade						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
cid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
grade	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
sid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
typeid	varchar(255)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
typeof	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Lecture

sys.lecture						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
cid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
fname	varchar(512)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
fpath	varchar(512)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Assignment

sys.assignment						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
◇ asname	varchar(512)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ aspath	varchar(512)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ cid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references

Submission

sys.submission						
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
◇ asname	varchar(512)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ cid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ sid	varchar(45)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ subname	varchar(512)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references
◇ subpath	varchar(512)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references

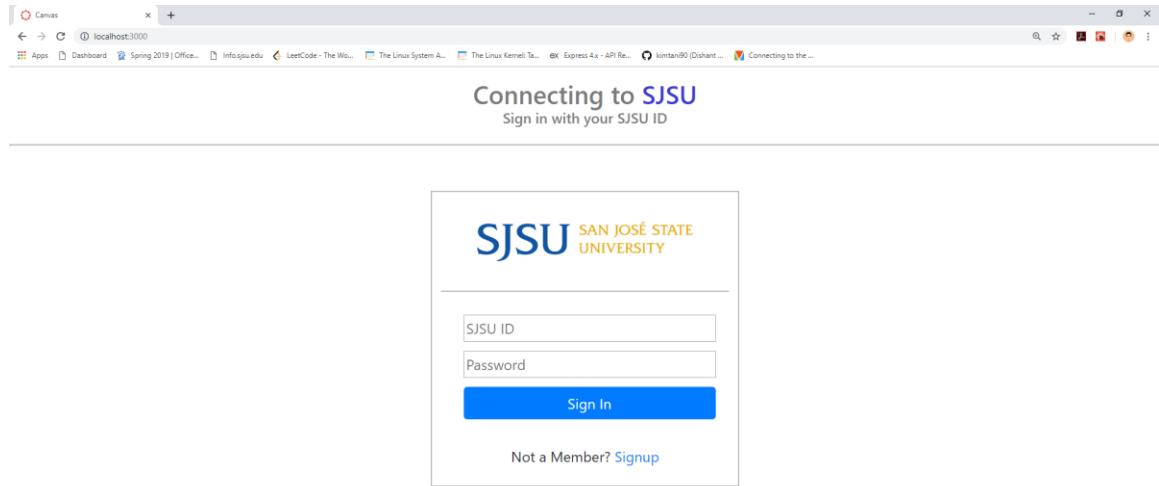
Technologies

Software: HTML, CSS, Javascript, Node, React, MySQL, Material UI, Bootstrap, Font Awesome.

Hardware: Any computer having Chrome along with required packages installed.

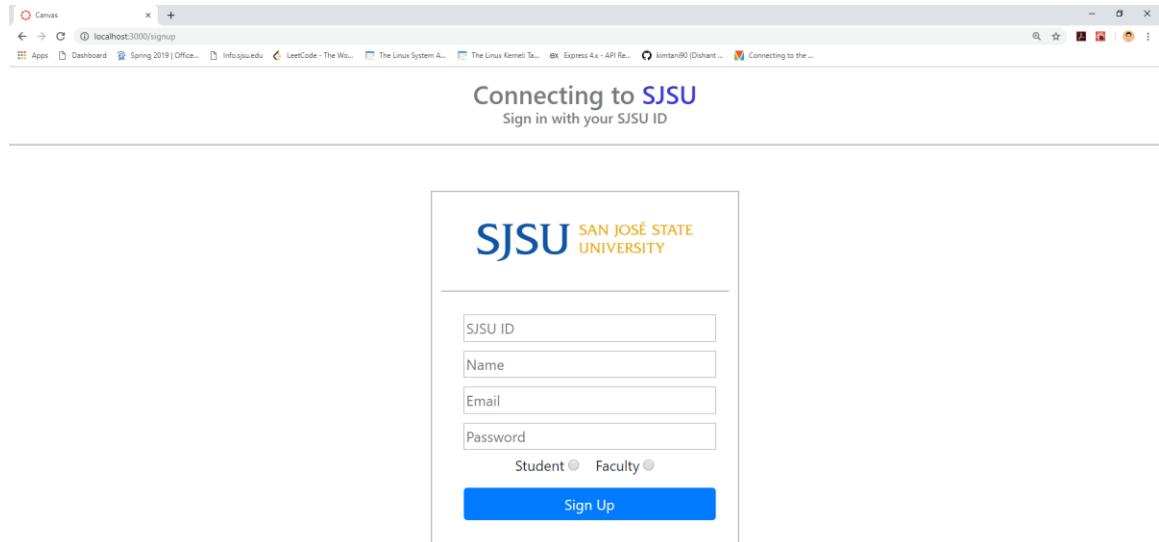
Results

Login page:



A screenshot of a web browser window showing the SJSU login page. The title bar reads "Connecting to SJSU" and "Sign in with your SJSU ID". The main form has the SJSU logo at the top. It contains two input fields: "SJSU ID" and "Password", followed by a blue "Sign In" button. Below the buttons is a link "Not a Member? [Signup](#)". The browser's address bar shows "localhost:3000".

Signup page:



A screenshot of a web browser window showing the SJSU signup page. The title bar reads "Connecting to SJSU" and "Sign in with your SJSU ID". The main form has the SJSU logo at the top. It contains four input fields: "SJSU ID", "Name", "Email", and "Password". Below these fields are two radio buttons: "Student" and "Faculty". At the bottom is a blue "Sign Up" button. The browser's address bar shows "localhost:3000/signup".

Faculty signup:

The screenshot shows a web browser window with the URL `localhost:3000/signup`. The page title is "Connecting to SJSU" and the sub-instruction is "Sign in with your SJSU ID". The main form is titled "SJSU SAN JOSÉ STATE UNIVERSITY". It contains four input fields: "104", "Shim Sang", "shim@sjsu.edu", and "....". Below the fields are two radio buttons: "Student" and "Faculty", with "Faculty" selected. A blue "Sign Up" button is at the bottom.

```
Server started on port 3001
Database connection successful.
104 Shim Sang shim@sjsu.edu $2b$10$68YT.9U9XAyVLypa.GkssOrUh073ofGGA2GA4ETIUnouAhAHeTuNm faculty
Row affected: 1
[]
```

Faculty login:

The screenshot shows a web browser window with the URL `localhost:3000/signup`. The page title is "Connecting to SJSU" and the sub-instruction is "Sign in with your SJSU ID". The main form is titled "SJSU SAN JOSÉ STATE UNIVERSITY". It contains three input fields: "104", "....", and a password field. Below the fields is a blue "Sign In" button. At the bottom of the form, there is a link "Not a Member? [Signup](#)".

```
Server started on port 3001
Database connection successful.
[ RowDataPacket {
    password:
      '$2b$10$68YT.9U9XAyVLypa.GkssOrUh073ofGGA2GA4ETIUnouAhAHeTuNm',
    role: 'faculty' } ]
[]
```

Faculty homepage:

The faculty homepage features a vertical blue sidebar on the left with the SJSU logo at the top. Below the logo are three icons: a person icon labeled "Profile", a book icon labeled "Courses", and a circular arrow icon labeled "Log Out". To the right of the sidebar is a white area titled "My Courses" in bold. At the top right of this area is a blue "Add Course" button.

Student homepage:

The student homepage has a similar layout to the faculty version. It includes a vertical blue sidebar with the SJSU logo, a "Profile" icon, a "Courses" icon, and a "Log Out" icon. The main content area is titled "My Courses" and contains a blue "Search Course" button. Below the main content is a dark terminal window displaying the following text:

```
Server started on port 3001
Database connection successful.
[]
```

Faculty homepage after several courses are created by them:

The screenshot shows a faculty homepage with a sidebar on the left and a main content area. The sidebar contains:

- SJSU logo
- Profile icon
- Courses icon
- Add Course button
- Log Out icon

The main content area is titled "My Courses" and displays four course cards:

Course ID	Course Name	Term
CMPE202	Software Systems Engineering	Spring 2019
CMPE256	Large-Scale Analytics	Fall 2019
CMPE272	Enterprise Software Platforms	Spring 2019
CMPE273	Enterprise Distributed Systems	Spring 2019

Below the courses, there is a code snippet representing the data structure of the courses:

```
[ RowDataPacket {  
    cid: 'CMPE202',  
    cname: 'Software Systems Engineering',  
    cterm: 'Spring 2019' },  
RowDataPacket {  
    cid: 'CMPE256',  
    cname: 'Large-Scale Analytics',  
    cterm: 'Fall 2019' },  
RowDataPacket {  
    cid: 'CMPE272',  
    cname: 'Enterprise Software Platforms',  
    cterm: 'Spring 2019' },  
RowDataPacket {  
    cid: 'CMPE273',  
    cname: 'Enterprise Distributed Systems',  
    cterm: 'Spring 2019' } ]
```

Student homepage after enrolling/wait listing for course:

The screenshot shows a student homepage with a sidebar on the left and a main content area on the right.

SJSU (Logo)

My Courses

Profile (Icon: User)

Courses (Icon: Book)

Search Course (Blue button)

Log Out (Icon: Logout)

CMPE256
Large-Scale Analytics
Fall 2019

CMPE202
Software Systems Engineering
Spring 2019

```
[ RowDataPacket {  
    cid: 'CMPE256',  
    cname: 'Large-Scale Analytics',  
    cterm: 'Fall 2019' },  
  RowDataPacket {  
    cid: 'CMPE202',  
    cname: 'Software Systems Engineering',  
    cterm: 'Spring 2019' } ]
```

User Profile:

The screenshot shows the 'My Profile' section of the SJSU website. On the left, there's a sidebar with icons for 'Profile' (user icon), 'Courses' (book icon), and 'Log Out' (exit icon). The main area has a large placeholder profile picture. It displays 'Account Information' with fields: SJSU ID: 104, Name: Shim Sang, and Email: shim@sjsu.edu. Below that is 'Personal Information' with fields for Contact Number, About, City, Country, Company, School, Hometown, Language, and Gender, all currently set to null. A blue 'Edit Profile' button is at the bottom.

```
RowDataPacket {  
    id: '104',  
    name: 'Shim Sang',  
    email: 'shim@sjsu.edu',  
    password:  
        '$2b$10$68YT.9U9XAyVLypa.GkssOrUh073oFGGA2G4ETIUnouAhAHeTuNm',  
    cno: null,  
    city: null,  
    country: null,  
    company: null,  
    school: null,  
    hometown: null,  
    gender: null,  
    about: null,  
    role: 'faculty',  
    image: null,  
    language: null }  
[]
```

Profile edit page:

SJSU

My Profile

Enter new information

Profile Picture	:	image url
Contact Number	:	0123456789
About Me	:	
City	:	
Country	:	
Company	:	
School	:	
Hometown	:	
Language	:	
Gender	:	<input type="radio"/> Male <input checked="" type="radio"/> Female <input type="radio"/> Other

Update **Cancel**

Profile
Courses
Log Out

SJSU

My Profile

Enter new information

Profile Picture	:	https://cmpe.sjsu.edu/file/4169821220
Contact Number	:	4169821220
About Me	:	
City	:	San Jose
Country	:	United States
Company	:	
School	:	
Hometown	:	
Language	:	
Gender	:	<input checked="" type="radio"/> Male <input type="radio"/> Female <input type="radio"/> Other

Update **Cancel**

Profile
Courses
Log Out

https://cmpe.sjsu.edu/files/public/styles/mugshot/public/shim-simon.jpg?itok=lITtCfp 4169821220 San Jose United States Male
Rows affected: 1

My Profile



Profile



Courses



Log Out



Account Information

SJSU ID: 104
Name : Shim Sang
Email : shim@sjsu.edu

Personal Information

Contact Number: 4169821220
About :
City : San Jose
Country : United States
Company :
School :
Hometown :
Language :
Gender : Male

[Edit Profile](#)

```
RowDataPacket {  
    id: '104',  
    name: 'Shim Sang',  
    email: 'shim@sjsu.edu',  
    password:  
        '$2b$10$68YT.9U9XayVLypa.Gkss0rUh073ofGGA2G44ETIUouAhAHeTuNm',  
    cno: '4169821220',  
    city: 'San Jose',  
    country: 'United States',  
    company: '',  
    school: '',  
    hometown: '',  
    gender: 'Male',  
    about: '',  
    role: 'faculty',  
    image:  
        'https://cmpe.sjsu.edu/files/public/styles/mugshot/public/shim-simon.jpg?itok=lITTcFqp',  
    language: '' }  
[]
```

Course creation by faculty:

SJSU

 Profile
 Courses
 Log Out

My Courses

Course ID :
Course Name :
Course Department:

Course Description :

Course Room :
Course Capacity :
Waitlist Capacity :
Course Term :

Add Course **Cancel**

SJSU

 Profile
 Courses
 Log Out

My Courses

Course ID : CMPE282
Course Name : Cloud Services
Course Department: Computer Engineering

Course Description : studies and team projects.
Prerequisites: CMPE 281 or
instructor consent. Computer
Engineering and Software
Engineering majors only.

Course Room : ENGR258
Course Capacity : 20
Waitlist Capacity : 10
Course Term : Fall 2020

Add Course **Cancel**

CMPE282 Cloud Services Computer Engineering Cloud service architecture and layering, administrative issues, resiliency and security considerations; business development, operations and business support service, case studies and team projects. Prerequisites: CMPE 281 or instructor consent. Computer Engineering and Software Engineering majors only. ENGR258 20 10 Fall 2020
Rows affected: 1

Navigation to home page after creating course:

The screenshot shows the SJSU My Courses interface. On the left is a vertical sidebar with icons for Profile, Courses, and Log Out. The main area is titled "My Courses" and displays a single course card for "CMPE282 Cloud Services Fall 2020". Below the card is a blue "Add Course" button. At the bottom of the page, there is a code snippet in a terminal window.

```
[ RowDataPacket { cid: 'CMPE282', cname: 'Cloud Services', cterm: 'Fall 2020' } ]
```

Course searching by student:

The screenshot shows the SJSU Course Catalog page. The sidebar on the left includes Profile, Courses, and Log Out options. The main content area is titled "Course Catalog" and features a search bar labeled "Search course by id,name or term". Below the search bar are two rows of course cards. The first row contains four courses: CMPE202 Software Systems Engineering (Spring 2019), CMPE255 Data Mining (Fall 2019), CMPE256 Large-Scale Analytics (Fall 2019), and CMPE257 Machine Learning (Spring 2020). The second row contains four courses: CMPE272 Enterprise Software, CMPE273 Enterprise Distributed, CMPE275 Enterprise Application, and CMPE281 Cloud Technologies.

```
[ RowDataPacket {
    cid: 'CMPE202',
    cname: 'Software Systems Engineering',
    cterm: 'Spring 2019' },
RowDataPacket { cid: 'CMPE255', cname: 'Data Mining', cterm: 'Fall 2019' },
RowDataPacket {
    cid: 'CMPE256',
    cname: 'Large-Scale Analytics',
    cterm: 'Fall 2019' },
RowDataPacket {
    cid: 'CMPE257',
    cname: 'Machine Learning',
    cterm: 'Spring 2020' },
RowDataPacket {
    cid: 'CMPE272',
    cname: 'Enterprise Software Platforms',
    cterm: 'Spring 2019' },
RowDataPacket {
    cid: 'CMPE273',
    cname: 'Enterprise Distributed Systems',
    cterm: 'Spring 2019' },
RowDataPacket {
    cid: 'CMPE275',
    cname: 'Enterprise Application Development',
    cterm: 'Fall 2019' },
RowDataPacket {
    cid: 'CMPE281',
    cname: 'Cloud Technologies',
    cterm: 'Fall 2019' },
RowDataPacket { cid: 'CMPE282', cname: 'Cloud Services', cterm: 'Fall 2020' } ]
```

The screenshot shows a user interface for a course catalog. On the left is a vertical sidebar with a blue header containing the "SJSU" logo. Below the logo are three menu items: "Profile" (with a person icon), "Courses" (with a book icon), and "Log Out" (with a right-pointing arrow icon). The main content area has a light gray background. At the top center is the title "Course Catalog". Below the title is a search bar with the word "fall" typed into it. The main content area displays five course cards arranged in two rows. The first row contains four cards: CMPE255 (Data Mining, Fall 2019), CMPE256 (Large-Scale Analytics, Fall 2019), CMPE275 (Enterprise Application Development, Fall 2019), and CMPE281 (Cloud Technologies, Fall 2019). The second row contains one card: CMPE282 (Cloud Services).

Course ID	Course Name	Term
CMPE255	Data Mining	Fall 2019
CMPE256	Large-Scale Analytics	Fall 2019
CMPE275	Enterprise Application Development	Fall 2019
CMPE281	Cloud Technologies	Fall 2019
CMPE282	Cloud Services	



Profile



Courses



Log Out

er|

CMPE202 Software Systems Engineering Spring 2019	CMPE272 Enterprise Software Platforms Spring 2019	CMPE273 Enterprise Distributed Systems Spring 2019	CMPE275 Enterprise Application Development Fall 2019
--	---	--	--

Course Homepage for faculty:



Profile



Courses



Log Out

Home

Announcements

Assignments

Grades

People

Files

Quiz

Cloud Services

Department: Computer Engineering

Description : Cloud service architecture and layering, administrative issues, resiliency and security considerations; business development, operations and business support service, case studies and team projects.

Prerequisites: CMPE 281 or instructor consent. Computer Engineering and Software Engineering majors only.

Classroom : ENGR258

Capacity : 20

Waitlist : 10

Term : Fall 2020

Course Homepage for student, having the option to enroll/wait-list:



Profile



Courses



Log Out

Cloud Services

Department : Computer Engineering

Description : Cloud service architecture and layering, administrative issues, resiliency and security considerations;

business development, operations and business support service, case studies and team projects.

Prerequisites: CMPE 281 or instructor consent. Computer Engineering and Software Engineering majors only.

Classroom : ENGR258

Capacity : 20

Waitlist : 10

Term : Fall 2020

[Enroll](#) [Waitlist](#)

```
[{"Routepacket": {
    "cid": "CMPE282",
    "fid": "104",
    "cname": "Cloud Services",
    "cddept": "Computer Engineering",
    "cdesc": "Cloud service architecture and layering, administrative issues, resiliency and security considerations; business development, operations and business support service, case studies and team projects. Prerequisite: CMPE 281 or instructor consent. Computer Engineering and Software Engineering majors only.",
    "croom": "ENGR258",
    "ccap": 20,
    "cwait": 10,
    "cterm": "Fall 2020"}]
```

Student course view if enrolled:



Profile



Courses



Log Out

Cloud Services

Department : Computer Engineering

Description : Cloud service architecture and layering, administrative issues, resiliency and security considerations;

business development, operations and business support service, case studies and team projects.

Prerequisites: CMPE 281 or instructor consent. Computer Engineering and Software Engineering majors only.

Classroom : ENGR258

Capacity : 20

Waitlist : 10

Term : Fall 2020

[Drop](#)

Student course view if waitlisted:

The screenshot shows a student's course view for CMPE202. The left sidebar has icons for Profile, Courses, and Log Out. The main content area displays course details: Software Systems Engineering, Department: Software Engineering, Description: Integrated approach to software design and development including requirements elicitation and analysis, system design and construction through studying multiple facets of software development processes, design methodologies, modeling approaches, and implementation techniques. Prerequisite: Classified graduate standing or instructor consent. Computer Engineering and Software Engineering majors only. Misc/Lab: Lecture 3 hours. Classroom: ENG337, Capacity: 20, Waitlist: 10, Term: Spring 2019, Status: Waitlisted. A blue "Drop" button is at the bottom.

Enrolled/ wait-listed course will be shown on student's homepage in form of cards as shown before.

People view for faculty:

The screenshot shows a faculty's people view for CMPE202. The left sidebar has icons for Profile, Courses, and Log Out, with People highlighted. The main content area shows a table titled "People" with four rows. The first three rows have "REMOVE" buttons, and the last row has an "ENROLL" button. The data is as follows:

Profile Picture	Name	Action
	Shivang Mistry	REMOVE
	Jon Snow	REMOVE
	Harry Potter	REMOVE
	Sherlock Holmes	ENROLL

```
[ RowDataPacket {
    cid: 'CMPE202',
    name: 'Some announcement',
    adesc: 'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.',
    atime: 'Sat Mar 16 2019 13:33:35 AM' } ]
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
[nodemon] starting node index.js
Server started on port 3001
Database connection successful.
[ RowDataPacket {
    id: '201',
    name: 'Shivang Mistry',
    image: 'https://lh3.googleusercontent.com/pyAziLmQx0Pj6dzfZ0tnc4-00Jldmx1qfDfWfpnHkR9FVxyis1kP2p-aimGxjX1_Cfm2qeNsBjC26h-UyviixXeKoEvSfna29dHQhy802maJ4w-ZFmSVgqfQ9GrcGgg6Y90VP901beM39d8kicpp1F0CCVhdByglzbZtpglikmr81pxdvcv415002d1MfJcIpbkvkubQby1ryzbljepb4h0sbgw8965f28f8fKErshqfQCr1kGs9tXegf368190EH1lgr1Yd5ny9xa3690enCLk65QAGh8dPj18q3dnk-prSVYH223dx_-gaLu2z1khSeL_-c1jQd4031DFw4055f8boweo298MNL_HrJsm5saB0h1h2291gkx0f9q65-c72554-MD01PK1ox38sysqj0zur1510gfbx9fhl_UbYQsQPRtzynhs1uk_2_tf-F-3VSCHN0_995jfrt3Gf1921QMFKc_j4UCh-TqslUftyKqavInrjgsM51lyscet9fx3X_p3FouwHw1pkeshyFu272-7xuErRmp3-SuhSHmfGeyak2juaW5h7xswy8hErXWVipHSn5169sk8C0mnssevEpvnwXv_fby5uzb8quz70sdg-w798-h958-n',
    stat: 'enroll' },
RowDataPacket {
    id: '203',
    name: 'Jon Snow',
    image: 'http://www.minimalisticlist.com/wp-content/uploads/minimal-GoT/minimal-game-of-thrones-by-jerry-liu-29-jon-snow.jpg',
    stat: 'enroll' },
RowDataPacket { id: '202', name: 'Harry Potter', image: '', stat: 'enroll' },
RowDataPacket {
    id: '204',
    name: 'Sherlock Holmes',
    image: null,
    stat: 'waitlist' } ]
```

People view for student:



SJSU

- [!\[\]\(ce7f9d1c6591dd360cc3fdef91bc6f1b_img.jpg\) Profile](#)
- [!\[\]\(ddf4c47366385fe173b81b76e5d85787_img.jpg\) Courses](#)
- [!\[\]\(709838b4b8c96990b2864638b1f39974_img.jpg\) Log Out](#)

CMPE202

People

Name
 Shivang Mistry
 Jon Snow
 Harry Potter

Faculty removing a student (Harry Potter):

The screenshot shows a web application interface. On the left is a vertical sidebar with the SJSU logo at the top. Below it are icons for Profile (user), Courses (book), and Log Out. The main content area has a header "People". On the left is a sidebar with links: Home, Announcements, Assignments, Grades, People, Files, and Quiz. The "People" link is highlighted. The main content area displays a list of four people with profile icons and names: Shivang Mistry, Jon Snow, Harry Potter, and Sherlock Holmes. To the right of each name is a red "REMOVE" button. Below the list is a blue "ENROLL" button. A modal dialog box is open at the top center with the text "Action Performed." and an "OK" button.

```
RowDataPacket { id: '202', name: 'Harry Potter', image: '', stat: 'enroll' },  
RowDataPacket {  
  id: '204',  
  name: 'Sherlock Holmes',  
  image: null,  
  stat: 'waitlist' } ]  
OkPacket {  
  fieldCount: 0,  
  affectedRows: 1,  
  insertId: 0,  
  serverStatus: 2,  
  warningCount: 0,  
  message: '',  
  protocol41: true,  
  changedRows: 0 }
```

This screenshot shows the same web application after an action has been performed. The sidebar and header are identical to the first screenshot. The main content area now shows the same list of people, but the "ENROLL" button for Sherlock Holmes has been replaced by a blue "ENROLLED" button with a checkmark icon. The other three people (Shivang Mistry, Jon Snow, and Harry Potter) still have their original "REMOVE" buttons.

Faculty generating permission code for waitlisted student (Sherlock Holmes):

SJSU

Profile Courses Log Out

Home Announcements Assignments Grades People Files Quiz

People

	Shivang Mistry	REMOVE
	Jon Snow	REMOVE
	Sherlock Holmes	ENROLL

```
RowDataPacket {  
    id: '204',  
    name: 'Sherlock Holmes',  
    image: null,  
    stat: 'waitlist' } ]  
OkPacket {  
    fieldCount: 0,  
    affectedRows: 1,  
    insertId: 0,  
    serverStatus: 2,  
    warningCount: 0,  
    message: '(Rows matched: 1  Changed: 1  Warnings: 0',  
    protocol41: true,  
    changedRows: 1 }
```

SJSU CMPE202

Profile Courses Log Out

Home Announcements Assignments Grades People Files Quiz

People

	Shivang Mistry	REMOVE
	Jon Snow	REMOVE
	Sherlock Holmes	REMOVE

Faculty announcement view, initially:



Announcements

New Announcement

Student announcement view, initially:



Announcements

Faculty adding announcement:



Home
Announcements
Assignments
Grades
People
Files
Quiz

New Announcements

Title

Announcement



Home
Announcements
Assignments
Grades
People
Files
Quiz

New Announcements

Title

Announcement

Some announcement Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure
Rows affected: 1



- [Home](#)
- [Announcements](#)
- [Assignments](#)
- [Grades](#)
- [People](#)
- [Files](#)
- [Quiz](#)

Announcements

Some announcement Sat Mar 16 2019 3:33:35 AM

>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore ...

[New Announcement](#)

```
[ RowDataPacket {
    cid: 'CMPE282',
    aname: 'Some announcement',
    adesc:
      'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.',
    atime: 'Sat Mar 16 2019 3:33:35 AM' } ]
```

Opening an announcement:



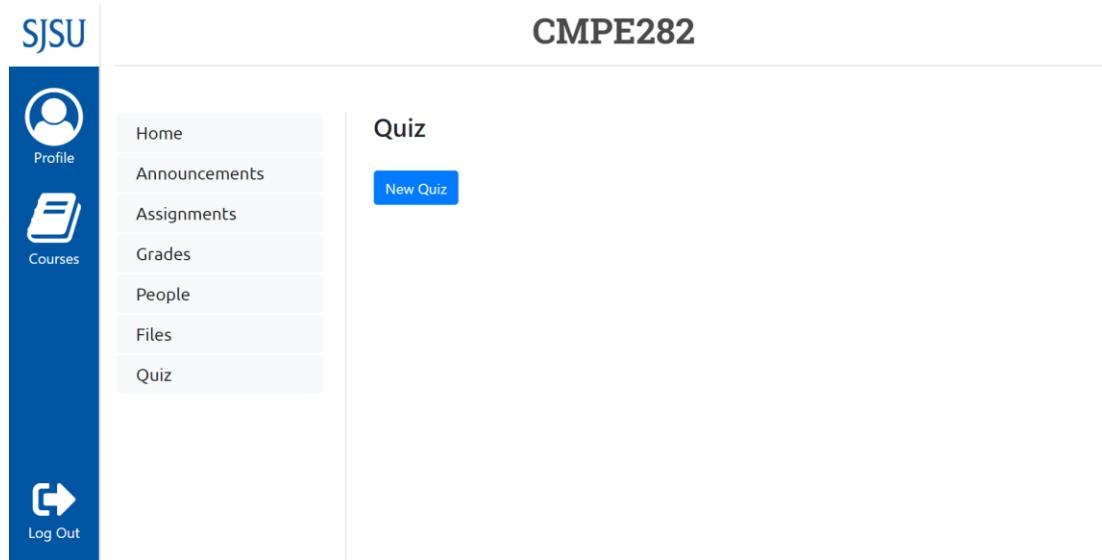
- [Home](#)
- [Announcements](#)
- [Assignments](#)
- [Grades](#)
- [People](#)
- [Files](#)
- [Quiz](#)

Some announcement

Sat Mar 16 2019 3:33:35 AM

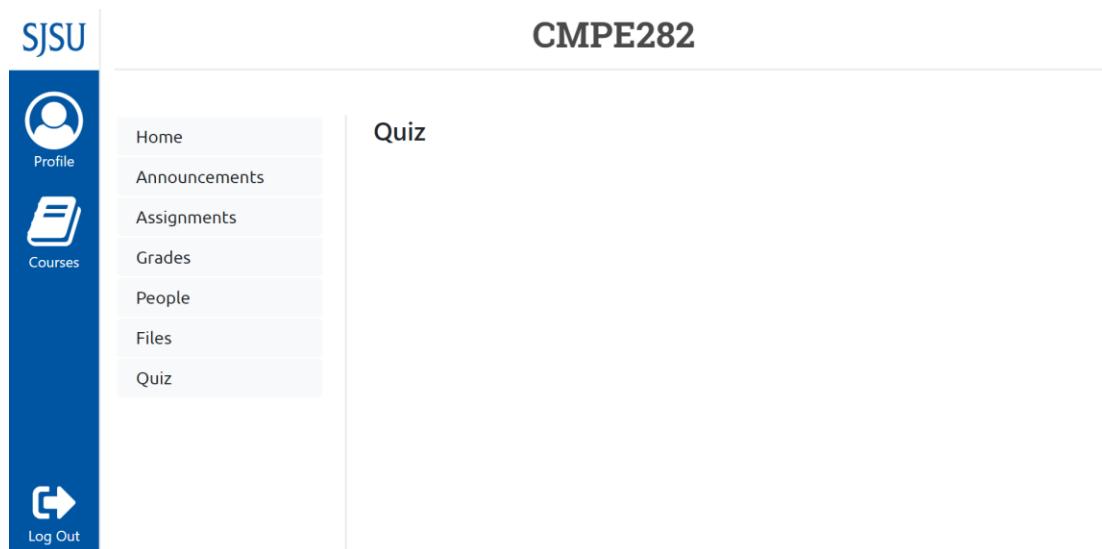
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

Faculty quiz view initially:



The screenshot shows the faculty view of a quiz page. At the top, it displays "SJSU" and "CMPE282". On the left, there's a vertical sidebar with icons for Profile (user), Courses (book), and Log Out. The main content area is titled "Quiz" and contains a "New Quiz" button. A sidebar menu on the right lists "Home", "Announcements", "Assignments", "Grades", "People", "Files", and "Quiz".

Student Quiz view initially:



The screenshot shows the student view of a quiz page. At the top, it displays "SJSU" and "CMPE282". On the left, there's a vertical sidebar with icons for Profile (user), Courses (book), and Log Out. The main content area is titled "Quiz". A sidebar menu on the right lists "Home", "Announcements", "Assignments", "Grades", "People", "Files", and "Quiz".

Faculty creating quiz:

SJSU

CMPE282

Create Quiz

Enter details and choose correct option for each question.

Quiz ID :

Name :

Question 1 :
Option A : Option B :
Option C : Option D :

Question 2 :
Option A : Option B :
Option C : Option D :

Available from: Available till:

Publish

SJSU

Create Quiz

Enter details and choose correct option for each question.

Quiz ID : Quiz - 01

Name : Cloud Basics

Question 1 : Applications that work with cloud comput
Option A : high touch Option B : low touch
Option C : moderate touch Option D : all of the mentioned

Question 2 : A service that concentrates on hardware fo
Option A : IaaS Option B : CaaS
Option C : PaaS Option D : All of the mentioned

Available from: Available till:

Publish

```
OkPacket {  
    fieldCount: 0,  
    affectedRows: 1,  
    insertId: 0,  
    serverStatus: 2,  
    warningCount: 0,  
    message: '',  
    protocol41: true,  
    changedRows: 0 }
```

Faculty view:

The dashboard for the CMPE282 course is displayed. At the top, the course code "CMPE282" is shown. On the left, a vertical sidebar contains the SJSU logo and a navigation menu with links: Profile, Courses, Home, Announcements, Assignments, Grades, People, Files, and Quiz. A blue "New Quiz" button is located below the Quiz link. At the bottom of the sidebar is a "Log Out" button.

Student view:

The dashboard for the CMPE282 course is displayed. At the top, the course code "CMPE282" is shown. On the left, a vertical sidebar contains the SJSU logo and a navigation menu with links: Profile, Courses, Home, Announcements, Assignments, Grades, People, Files, and Quiz. Below the navigation menu, there is a link to "Quiz - 01 - Cloud Basics". At the bottom of the sidebar is a "Log Out" button.

```
[ RowDataPacket { qid: 'Quiz - 01', qname: 'Cloud Basics' } ]
```

Faculty quiz info:

SJSU



Profile



Courses

 Log Out

CMPE282

Quiz - 01 - Cloud Basics

Applications that work with cloud computing that have low margins and usually low risk are

- high touch
- low touch
- moderate touch
- all of the mentioned

A service that concentrates on hardware follows the _____ as a Service model.

- IaaS
- Caas
- PaaS
- All of the mentioned

```
[ RowDataPacket {  
    cid: 'CMPE282',  
    qid: 'Quiz - 01',  
    qname: 'Cloud Basics',  
    ql:  
        'Applications that work with cloud computing that have low margins and usually low risk are',  
        op11: 'high touch',  
        op12: 'low touch',  
        op13: 'moderate touch',  
        op14: 'all of the mentioned',  
        cor1: 'b',  
        q2:  
            'A service that concentrates on hardware follows the _____ as a Service model.',  
        op21: 'IaaS',  
        op22: 'Caas',  
        op23: 'PaaS',  
        op24: 'All of the mentioned',  
        cor2: 'a',  
        d1: '2019-03-15',  
        d2: '2019-03-18' } ]
```

Student quiz info and submission:

The screenshot shows a web browser window with a quiz interface. The left sidebar has icons for Profile, Courses, and Log Out. The main area shows a quiz titled "Quiz - 01 - Cloud Basics". Two questions are listed:

- Question 1: Applications that work with cloud computing that have low margins and usually low risk are
 - high touch
 - low touch
 - moderate touch
 - all of the mentioned
- Question 2: A service that concentrates on hardware follows the _____ as a Service model.
 - IaaS
 - Caas
 - PaaS
 - All of the mentioned

A modal dialog box at the top right says "localhost:3000 says Score: 2 OK". Below the quiz interface is a large black box containing the JSON data structure used to store the quiz information.

```
[ RowDataPacket {  
    cid: 'CMPE282',  
    qid: 'Quiz - 01',  
    qname: 'Cloud Basics',  
    q1:  
      'Applications that work with cloud computing that have low margins and usually low risk are',  
      op11: 'high touch',  
      op12: 'low touch',  
      op13: 'moderate touch',  
      op14: 'all of the mentioned',  
      cor1: 'b',  
    q2:  
      'A service that concentrates on hardware follows the _____ as a Service model.',  
      op21: 'IaaS',  
      op22: 'Caas',  
      op23: 'PaaS',  
      op24: 'All of the mentioned',  
      cor2: 'a',  
      d1: '2019-03-15',  
      d2: '2019-03-18' } ]  
Answers: b a  
[ RowDataPacket { cor1: 'b', cor2: 'a' } ]
```

Student Grade view:

SJSU

CMPE202

Profile

Announcements

Assignments

Grades

People

Files

Quiz

Courses

Log Out

Grades

Quiz

Quiz 1 2/2

Assignments

02 - OOP.pptx 10/10

```
[ RowDataPacket {  
    cid: 'CMPE202',  
    sid: '201',  
    typeof: 'quiz',  
    typeid: 'Quiz 1',  
    grade: '2' },  
RowDataPacket {  
    cid: 'CMPE202',  
    sid: '201',  
    typeof: 'ass',  
    typeid: '02 - OOP.pptx',  
    grade: '10' } ]
```

Faculty Grade view:

SJSU



Profile



Courses

 Log Out

CMPE202

Grades

Quiz

201	Quiz 1	2/2
-----	--------	-----

Assignments

201	02 - OOP.pptx	10/10
203	02 - OOP.pptx	10/10
202	02 - OOP.pptx	5/10

```
[ RowDataPacket {
    cid: 'CMPE202',
    sid: '201',
    typeof: 'quiz',
    typeid: 'Quiz 1',
    grade: '2' },
  RowDataPacket {
    cid: 'CMPE202',
    sid: '201',
    typeof: 'ass',
    typeid: '02 - OOP.pptx',
    grade: '10' },
  RowDataPacket {
    cid: 'CMPE202',
    sid: '203',
    typeof: 'ass',
    typeid: '02 - OOP.pptx',
    grade: '10' },
  RowDataPacket {
    cid: 'CMPE202',
    sid: '202',
    typeof: 'ass',
    typeid: '02 - OOP.pptx',
    grade: '5' } ]
```

Faculty File view:

SJSU

CMPE282

Profile

Courses

Log Out

Files

Choose File No file chosen Upload

Home Announcements Assignments Grades People Files Quiz

Student view for some lecture files uploaded:

SJSU

CMPE202

Profile

Courses

Log Out

Files

04 - GoF Design Patterns - Adapter (powerpoint).pptx
04 - GoF Design Patterns - Observer (powerpoint).pptx
CMPE 273 Lab 1.pdf

Home Announcements Assignments Grades People Files Quiz

Clicking on the file name will download the file:

The screenshot shows the SJSU CMPE202 course interface. On the left, there is a vertical sidebar with icons for Profile, Courses, and Log Out. Below these are links for Home, Announcements, Assignments, Grades, People, Files (which is currently selected), and Quiz. The main content area is titled "CMPE202" and shows a "Files" section. It lists several files: "04 - GoF Design Patterns - Adapter (powerpoint).pptx", "04 - GoF Design Patterns - Observer (powerpoint).pptx", and "CMPE 273 Lab 1.pdf". At the bottom of the page, there are two preview thumbnails for the first two powerpoint files.

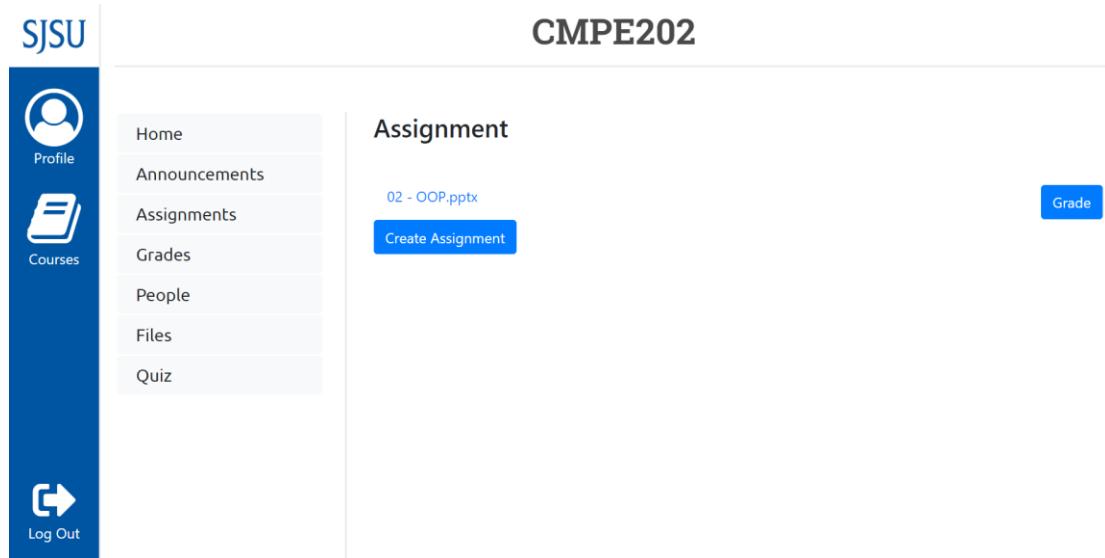
Faculty can upload new file:

The screenshot shows the SJSU CMPE202 course interface. On the left, there is a vertical sidebar with icons for Profile, Courses, and Log Out. Below these are links for Home, Announcements, Assignments, Grades, People, Files (which is currently selected), and Quiz. The main content area is titled "CMPE202" and shows a "Files" section. There is a "Choose File" input field containing "00-Introduction (2).pdf" and a blue "Upload" button. A modal dialog box is displayed in the center, showing the message "localhost:3000 says File Uploaded." with an "OK" button. The browser's address bar at the top shows "localhost:3000/course/CMPE202/file".

```
{ fieldName: 'lecturefile',
originalname: '00-Introduction (2).pdf',
encoding: '7bit',
mimetype: 'application/pdf',
destination: './public/uploads/',
filename: '00-Introduction (2).pdf1552820224155.pdf',
path: 'public\uploads\00-Introduction (2).pdf1552820224155.pdf',
size: 413526 }
OkPacket {
  fieldCount: 0,
  affectedRows: 1,
  insertId: 0,
  serverStatus: 2,
  warningCount: 0,
  message: '',
  protocol41: true,
  changedRows: 0 }
```

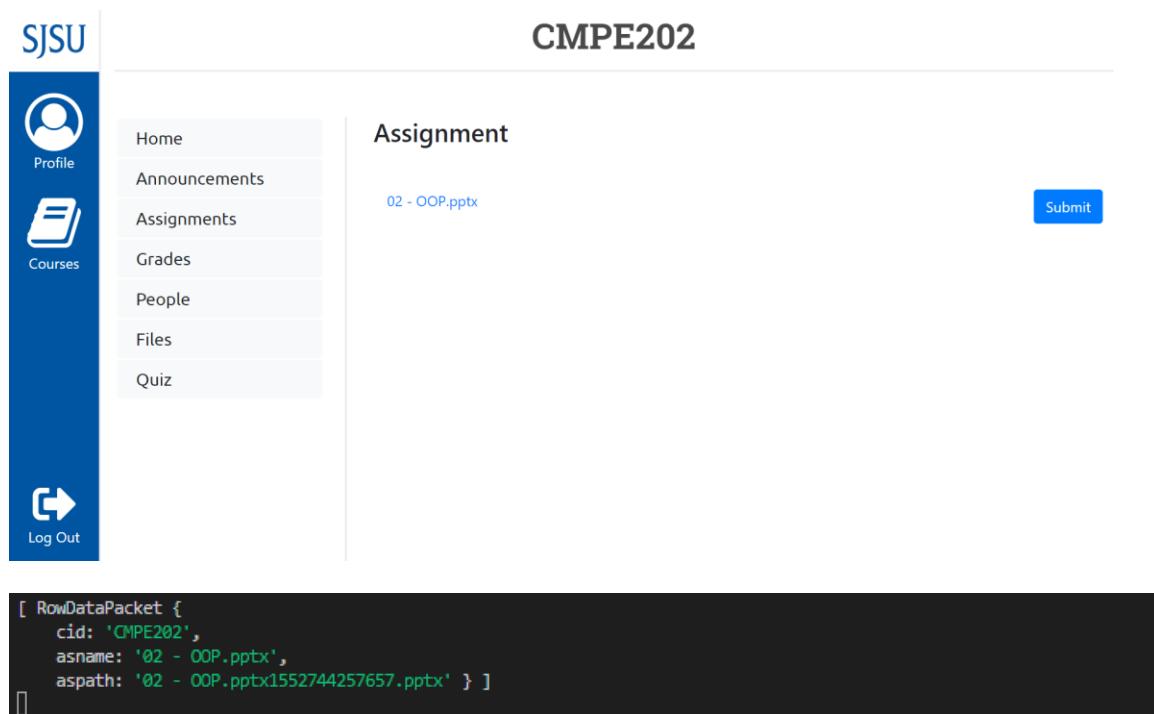
The screenshot shows a user interface for a learning management system. On the left is a vertical sidebar with a blue header containing the SJSU logo. Below the header, the sidebar includes icons for Profile (user), Courses (book), and Log Out (exit). The main content area has a header "CMPE282". On the left side of the main area, there is a navigation menu with links: Home, Announcements, Assignments, Grades, People, Files, and Quiz. The "Files" link is currently selected, indicated by a grey background. The main content area is titled "Files" and displays a file entry for "00-Introduction (2).pdf". It shows a "Choose File" button, a text input field "No file chosen", and a blue "Upload" button.

Faculty Assignment view:



The screenshot shows the faculty assignment view. On the left is a vertical sidebar with the SJSU logo at the top, followed by icons for Profile (user), Courses (book), and Log Out. Below these are links for Home, Announcements, Assignments (which is highlighted in blue), Grades, People, Files, and Quiz. The main content area is titled "Assignment" and shows a file named "02 - OOP.pptx". A blue button labeled "Create Assignment" is visible. In the top right corner, there is a blue button labeled "Grade".

Student assignment view:



The screenshot shows the student assignment view. The layout is identical to the faculty view, with the SJSU sidebar on the left and the "Assignment" section on the right. The "Assignments" link in the sidebar is also highlighted in blue. The main content area shows the same "02 - OOP.pptx" file. A blue button labeled "Submit" is present in the top right corner. At the bottom of the page, there is a dark box containing the following JSON-like code:

```
[ RowDataPacket {  
    cid: 'CMPE202',  
    asname: '02 - OOP.pptx',  
    aspath: '02 - OOP.pptx1552744257657.pptx' } ]
```

Faculty posting assignment:

The screenshot shows the SJSU Learning Management System (LMS) interface. On the left is a vertical sidebar with icons for Profile, Courses, and Log Out. The main content area has a header "New Assignment". A file upload dialog box is overlaid on the page, showing the message "File Uploaded." with an "OK" button. Below the dialog, there is a "Choose File" button with the path "UML_example_source.pdf" and a "Publish" button.

```
{ fieldname: 'assignment',
originalname: 'UML_example_source.pdf',
encoding: '7bit',
mimetype: 'application/pdf',
destination: './public/uploads/',
filename: 'UML_example_source.pdf1552834681197.pdf',
path: 'public\uploads\\UML_example_source.pdf1552834681197.pdf',
size: 52013 }
```

The screenshot shows the CMPE202 course page within the SJSU LMS. The left sidebar includes Profile, Courses, and Log Out. The main content area has a header "Assignment". It lists two files: "02 - OOP.pptx" and "UML_example_source.pdf". There are two blue "Grade" buttons on the right. A "Create Assignment" button is located below the file list.

```
[ RowDataPacket {
    cid: 'CMPE202',
    asname: '02 - OOP.pptx',
    aspath: '02 - OOP.pptx1552744257657.pptx' },
RowDataPacket {
    cid: 'CMPE202',
    asname: 'UML_example_source.pdf',
    aspath: 'UML_example_source.pdf1552834681197.pdf' } ]
```

Student submitting an assignment:

The screenshot shows a web application interface for a student. On the left is a vertical sidebar with the SJSU logo at the top. Below it are icons for Profile (user), Courses (book), and Log Out. The main content area has a header "Submission". On the right, there is a file input field labeled "Choose File" with "Capture.PNG" selected, and a blue "Submit" button. A modal dialog box is open in the center, showing the message "localhost:3000 says File Uploaded." with an "OK" button. The browser's address bar shows the URL "localhost:3000/course/CMPE202/assignment/UML_example_source.pdf". The status bar at the bottom of the browser window shows "Connecting to the...".

```
{ fieldname: 'submission',
originalname: 'Capture.PNG',
encoding: '7bit',
mimetype: 'image/png',
destination: './public/uploads/',
filename: 'Capture.PNG1552835316218.PNG',
path: 'public\uploads\Capture.PNG1552835316218.PNG',
size: 120266 }
OkPacket {
fieldCount: 0,
affectedRows: 1,
insertId: 0,
serverStatus: 2,
warningCount: 0,
message: '',
protocol41: true,
changedRows: 0 }
```

Faculty grading an assignment:

The screenshot shows a faculty member grading an assignment. The interface has a left sidebar with icons for Profile, Courses, and Log Out. The main area is titled "CMPE202 Submission". It shows a file named "204 Capture.PNG" with a grade of "8" entered in a text field, followed by a "Grade" button.

```
{ sid: '204', grade: '8' }
OkPacket {
  fieldCount: 0,
  affectedRows: 1,
  insertId: 0,
  serverStatus: 2,
  warningCount: 0,
  message: '',
  protocol41: true,
  changedRows: 0 }
```

Student grades being updated:

The screenshot shows a student updating their grades. The interface has a left sidebar with icons for Profile, Courses, and Log Out. The main area is titled "CMPE202 Grades". It shows a "Quiz" section with a grade of "8/10" and an "Assignments" section showing a file named "UML_example_source.pdf".

```
[ RowDataPacket {  
    cid: 'CMPE202',  
    sid: '204',  
    typeof: 'ass',  
    typeid: 'UML_example_source.pdf',  
    grade: '8' } ]
```

- On any page where the file is uploaded like as a lecture note, assignment or submission, clicking on the file name will download the file.

Performance

Jmeter Testing

100 concurrent users, without connection pooling, avg time = 516ms

View Results in Table										
<input type="text"/> Name: View Results in Table <input type="text"/> Comments: <input type="checkbox"/> Write results to file / Read from file: <input type="text"/> Filename <input type="button" value="Browse..."/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:10:07.587	Users 1-1	HTTP Request	23	✓	919	124	23	6	
2	13:10:07.602	Users 1-3	HTTP Request	53	✓	919	124	53	3	
3	13:10:07.645	Users 1-4	HTTP Request	36	✓	919	124	36	2	
4	13:10:07.644	Users 1-5	HTTP Request	56	✓	919	124	56	3	
5	13:10:07.651	Users 1-6	HTTP Request	70	✓	919	124	70	1	
6	13:10:07.664	Users 1-7	HTTP Request	88	✓	919	124	88	1	
7	13:10:07.667	Users 1-2	HTTP Request	96	✓	919	124	96	1	
8	13:10:07.672	Users 1-8	HTTP Request	129	✓	919	124	129	1	
9	13:10:07.687	Users 1-9	HTTP Request	131	✓	919	124	131	2	
10	13:10:07.696	Users 1-10	HTTP Request	136	✓	919	124	136	1	
11	13:10:07.713	Users 1-11	HTTP Request	125	✓	919	124	125	2	
12	13:10:07.716	Users 1-12	HTTP Request	306	✓	919	124	305	1	
13	13:10:07.728	Users 1-13	HTTP Request	321	✓	919	124	321	2	
14	13:10:07.742	Users 1-14	HTTP Request	339	✓	919	124	339	2	
15	13:10:07.747	Users 1-15	HTTP Request	392	✓	919	124	392	1	
16	13:10:07.760	Users 1-16	HTTP Request	405	✓	919	124	405	2	
17	13:10:07.778	Users 1-17	HTTP Request	410	✓	919	124	410	4	
18	13:10:07.787	Users 1-18	HTTP Request	416	✓	919	124	416	1	
19	13:10:07.850	Users 1-19	HTTP Request	377	✓	919	124	377	1	
20	13:10:07.854	Users 1-20	HTTP Request	399	✓	919	124	399	2	
21	13:10:07.874	Users 1-21	HTTP Request	445	✓	919	124	445	3	
22	13:10:07.883	Users 1-22	HTTP Request	485	✓	919	124	485	2	
23	13:10:07.885	Users 1-23	HTTP Request	524	✓	919	124	524	3	
24	13:10:07.890	Users 1-24	HTTP Request	528	✓	919	124	528	2	
25	13:10:07.904	Users 1-25	HTTP Request	572	✗	919	124	572	1	

Scroll automatically? Child samples?
No of Samples 100 Latest Sample 509 Average 516 Deviation 171

100 concurrent users, with connection pooling, avg time = 128ms

View Results in Table										
<input type="text"/> Name: View Results in Table <input type="text"/> Comments: <input type="checkbox"/> Write results to file / Read from file: <input type="text"/> Filename <input type="button" value="Browse..."/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:31:07.649	Users 1-1	HTTP Request	34	✓	919	124	34	2	
2	13:31:07.673	Users 1-3	HTTP Request	20	✓	919	124	20	2	
3	13:31:07.665	Users 1-2	HTTP Request	36	✓	919	124	36	1	
4	13:31:07.699	Users 1-4	HTTP Request	13	✓	919	124	12	1	
5	13:31:07.709	Users 1-5	HTTP Request	13	✓	919	124	13	1	
6	13:31:07.721	Users 1-6	HTTP Request	13	✓	919	124	13	1	
7	13:31:07.732	Users 1-7	HTTP Request	13	✓	919	124	13	1	
8	13:31:07.741	Users 1-8	HTTP Request	19	✓	919	124	19	1	
9	13:31:07.752	Users 1-11	HTTP Request	16	✓	919	124	16	1	
10	13:31:07.756	Users 1-9	HTTP Request	15	✓	919	124	15	1	
11	13:31:07.773	Users 1-10	HTTP Request	12	✓	919	124	12	1	
12	13:31:07.801	Users 1-12	HTTP Request	8	✓	919	124	8	1	
13	13:31:07.814	Users 1-13	HTTP Request	6	✓	919	124	6	1	
14	13:31:07.824	Users 1-14	HTTP Request	8	✓	919	124	8	1	
15	13:31:07.832	Users 1-15	HTTP Request	9	✓	919	124	9	1	
16	13:31:07.842	Users 1-16	HTTP Request	7	✓	919	124	7	1	
17	13:31:07.853	Users 1-17	HTTP Request	7	✓	919	124	7	1	
18	13:31:07.866	Users 1-18	HTTP Request	10	✓	919	124	10	1	
19	13:31:07.874	Users 1-19	HTTP Request	11	✓	919	124	11	1	
20	13:31:07.884	Users 1-20	HTTP Request	8	✓	919	124	8	1	
21	13:31:07.897	Users 1-21	HTTP Request	8	✓	919	124	8	1	
22	13:31:07.904	Users 1-22	HTTP Request	11	✓	919	124	11	2	
23	13:31:07.917	Users 1-23	HTTP Request	6	✓	919	124	6	1	
24	13:31:07.925	Users 1-24	HTTP Request	8	✓	919	124	8	1	
25	13:31:07.935	Users 1-25	HTTP Request	7	✗	919	124	7	2	

Scroll automatically? Child samples?
No of Samples 100 Latest Sample 461 Average 128 Deviation 116

200 concurrent users, without connection pooling, avg time = 940ms

View Results in Table										
Name: View Results in Table										
Comments:										
Write results to file / Read from file										
Filename										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:13:34.299	Users 1-3	HTTP Request	23	✓	919	124	23	2	
2	13:13:34.304	Users 1-4	HTTP Request	61	✓	919	124	61	1	
3	13:13:34.303	Users 1-2	HTTP Request	65	✓	919	124	65	5	
4	13:13:34.284	Users 1-1	HTTP Request	84	✓	919	124	84	13	
5	13:13:34.362	Users 1-5	HTTP Request	74	✓	919	124	74	1	
6	13:13:34.360	Users 1-6	HTTP Request	79	✓	919	124	79	1	
7	13:13:34.356	Users 1-7	HTTP Request	88	✓	919	124	88	2	
8	13:13:34.446	Users 1-16	HTTP Request	32	✓	919	124	32	1	
9	13:13:34.435	Users 1-11	HTTP Request	51	✓	919	124	51	10	
10	13:13:34.433	Users 1-12	HTTP Request	60	✓	919	124	60	2	
11	13:13:34.448	Users 1-13	HTTP Request	45	✓	919	124	45	1	
12	13:13:34.446	Users 1-14	HTTP Request	92	✓	919	124	92	2	
13	13:13:34.446	Users 1-10	HTTP Request	161	✓	919	124	161	9	
14	13:13:34.487	Users 1-9	HTTP Request	122	✓	919	124	122	1	
15	13:13:34.487	Users 1-19	HTTP Request	122	✓	919	124	122	2	
16	13:13:34.488	Users 1-15	HTTP Request	120	✓	919	124	120	1	
17	13:13:34.490	Users 1-18	HTTP Request	118	✓	919	124	118	1	
18	13:13:34.490	Users 1-23	HTTP Request	117	✓	919	124	117	1	
19	13:13:34.491	Users 1-20	HTTP Request	115	✓	919	124	115	1	
20	13:13:34.492	Users 1-17	HTTP Request	112	✓	919	124	112	42	
21	13:13:34.535	Users 1-26	HTTP Request	68	✓	919	124	68	1	
22	13:13:34.538	Users 1-22	HTTP Request	303	✓	919	124	303	77	
23	13:13:34.538	Users 1-21	HTTP Request	473	✓	919	124	473	115	
24	13:13:34.653	Users 1-58	HTTP Request	410	✓	919	124	410	1	
25	13:13:34.657	Users 1-25	HTTP Request	579	✓	919	124	579	1	

200 concurrent users, with connection pooling, avg time = 401ms

View Results in Table										
Name: View Results in Table										
Comments:										
Write results to file / Read from file										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:32:02.152	Users 1-1	HTTP Request	11	✓	919	124	11	3	
2	13:32:02.162	Users 1-2	HTTP Request	10	✓	919	124	10	1	
3	13:32:02.167	Users 1-3	HTTP Request	15	✓	919	124	15	1	
4	13:32:02.173	Users 1-4	HTTP Request	18	✓	919	124	18	1	
5	13:32:02.178	Users 1-5	HTTP Request	79	✓	919	124	79	1	
6	13:32:02.185	Users 1-6	HTTP Request	89	✓	919	124	89	1	
7	13:32:02.191	Users 1-7	HTTP Request	95	✓	919	124	95	1	
8	13:32:02.196	Users 1-8	HTTP Request	101	✓	919	124	101	1	
9	13:32:02.206	Users 1-10	HTTP Request	151	✓	919	124	151	0	
10	13:32:02.233	Users 1-13	HTTP Request	130	✓	919	124	130	0	
11	13:32:02.239	Users 1-14	HTTP Request	128	✓	919	124	128	1	
12	13:32:02.252	Users 1-15	HTTP Request	118	✓	919	124	118	1	
13	13:32:02.202	Users 1-9	HTTP Request	174	✓	919	124	174	1	
14	13:32:02.221	Users 1-11	HTTP Request	158	✓	919	124	158	1	
15	13:32:02.227	Users 1-12	HTTP Request	170	✓	919	124	170	1	
16	13:32:02.262	Users 1-16	HTTP Request	139	✓	919	124	139	1	
17	13:32:02.265	Users 1-17	HTTP Request	155	✓	919	124	155	1	
18	13:32:02.389	Users 1-49	HTTP Request	61	✓	919	124	61	1	
19	13:32:02.394	Users 1-50	HTTP Request	61	✓	919	124	61	1	
20	13:32:02.409	Users 1-51	HTTP Request	53	✓	919	124	53	2	
21	13:32:02.414	Users 1-18	HTTP Request	53	✓	919	124	53	1	
22	13:32:02.443	Users 1-53	HTTP Request	34	✓	919	124	34	1	
23	13:32:02.427	Users 1-52	HTTP Request	54	✓	919	124	54	1	
24	13:32:02.506	Users 1-19	HTTP Request	15	✓	919	124	15	1	
25	13:32:02.508	Users 1-20	HTTP Request	16	✓	919	124	15	2	

300 concurrent users, without connection pooling, avg time = 1022ms

View Results in Table										
<input type="text"/> Name: View Results in Table <input type="text"/> Comments: <input type="checkbox"/> Write results to file / Read from file <input type="text"/> Filename <input type="button" value="Browse..."/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:14:33.442	Users 1-1	HTTP Request	311	✓	919	124	311	3	
2	13:14:33.444	Users 1-3	HTTP Request	357	✓	919	124	357	1	
3	13:14:33.446	Users 1-4	HTTP Request	411	✓	919	124	411	1	
4	13:14:33.447	Users 1-6	HTTP Request	422	✓	919	124	422	1	
5	13:14:33.450	Users 1-8	HTTP Request	433	✓	919	124	433	10	
6	13:14:33.450	Users 1-7	HTTP Request	438	✓	919	124	438	8	
7	13:14:33.465	Users 1-5	HTTP Request	427	✓	919	124	427	1	
8	13:14:33.471	Users 1-9	HTTP Request	429	✓	919	124	429	1	
9	13:14:33.473	Users 1-10	HTTP Request	435	✓	919	124	435	1	
10	13:14:33.474	Users 1-11	HTTP Request	441	✓	919	124	441	1	
11	13:14:33.475	Users 1-12	HTTP Request	448	✓	919	124	448	2	
12	13:14:33.478	Users 1-13	HTTP Request	456	✓	919	124	456	2	
13	13:14:33.480	Users 1-14	HTTP Request	462	✓	919	124	462	1	
14	13:14:33.488	Users 1-27	HTTP Request	461	✓	919	124	461	1	
15	13:14:33.488	Users 1-18	HTTP Request	472	✓	919	124	472	1	
16	13:14:33.488	Users 1-25	HTTP Request	483	✓	919	124	483	2	
17	13:14:33.489	Users 1-16	HTTP Request	485	✓	919	124	485	1	
18	13:14:33.491	Users 1-26	HTTP Request	491	✓	919	124	491	1	
19	13:14:33.493	Users 1-30	HTTP Request	503	✓	919	124	503	1	
20	13:14:33.495	Users 1-21	HTTP Request	503	✓	919	124	503	1	
21	13:14:33.493	Users 1-24	HTTP Request	505	✓	919	124	505	1	
22	13:14:33.496	Users 1-19	HTTP Request	509	✓	919	124	509	1	
23	13:14:33.497	Users 1-22	HTTP Request	514	✓	919	124	514	1	
24	13:14:33.498	Users 1-2	HTTP Request	520	✓	919	124	520	1	
25	13:14:33.499	Users 1-15	HTTP Request	545	✗	919	124	545	1	

Scroll automatically? Child samples?
No of Samples 300 Latest Sample 893 Average 1022 Deviation 250

300 concurrent users, with connection pooling, avg time = 616ms

View Results in Table										
<input type="text"/> Name: View Results in Table <input type="text"/> Comments: <input type="checkbox"/> Write results to file / Read from file <input type="text"/> Filename <input type="button" value="Browse..."/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:32:48.416	Users 1-1	HTTP Request	21	✓	919	124	21	3	
2	13:32:48.417	Users 1-2	HTTP Request	22	✓	919	124	22	1	
3	13:32:48.417	Users 1-3	HTTP Request	27	✓	919	124	27	1	
4	13:32:48.426	Users 1-4	HTTP Request	30	✓	919	124	30	2	
5	13:32:48.429	Users 1-5	HTTP Request	35	✓	919	124	35	2	
6	13:32:48.435	Users 1-6	HTTP Request	58	✓	919	124	58	1	
7	13:32:48.442	Users 1-7	HTTP Request	56	✓	919	124	56	1	
8	13:32:48.445	Users 1-8	HTTP Request	60	✓	919	124	60	0	
9	13:32:48.448	Users 1-9	HTTP Request	60	✓	919	124	60	0	
10	13:32:48.453	Users 1-10	HTTP Request	59	✓	919	124	59	1	
11	13:32:48.460	Users 1-13	HTTP Request	54	✓	919	124	54	1	
12	13:32:48.462	Users 1-11	HTTP Request	56	✓	919	124	56	1	
13	13:32:48.469	Users 1-14	HTTP Request	53	✓	919	124	53	1	
14	13:32:48.473	Users 1-15	HTTP Request	51	✓	919	124	51	2	
15	13:32:48.471	Users 1-12	HTTP Request	56	✓	919	124	56	5	
16	13:32:48.528	Users 1-25	HTTP Request	9	✓	919	124	9	1	
17	13:32:48.541	Users 1-16	HTTP Request	11	✓	919	124	11	1	
18	13:32:48.544	Users 1-29	HTTP Request	15	✓	919	124	15	1	
19	13:32:48.544	Users 1-17	HTTP Request	19	✓	919	124	19	2	
20	13:32:48.548	Users 1-18	HTTP Request	20	✓	919	124	20	1	
21	13:32:48.555	Users 1-19	HTTP Request	23	✓	919	124	23	1	
22	13:32:48.560	Users 1-20	HTTP Request	21	✓	919	124	21	1	
23	13:32:48.564	Users 1-21	HTTP Request	20	✓	919	124	20	1	
24	13:32:48.589	Users 1-22	HTTP Request	15	✓	919	124	15	2	
25	13:32:48.592	Users 1-32	HTTP Request	21	✓	919	124	21	1	

Scroll automatically? Child samples?
No of Samples 300 Latest Sample 855 Average 616 Deviation 281

400 concurrent users, without connection pooling, avg time = 1121ms

View Results in Table										
Name: View Results in Table										
Comments:										
Write results to file / Read from file:										
<input type="text"/> Filename										
<input type="button" value="Browse ..."/>										
<input type="checkbox"/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:16:03.237	Users 1-1	HTTP Request	16	✓	919	124	16	5	
2	13:16:03.311	Users 1-2	HTTP Request	45	✓	919	124	45	1	
3	13:16:03.353	Users 1-20	HTTP Request	88	✓	919	124	88	1	
4	13:16:03.353	Users 1-9	HTTP Request	88	✓	919	124	88	1	
5	13:16:03.351	Users 1-19	HTTP Request	90	✓	919	124	90	1	
6	13:16:03.343	Users 1-3	HTTP Request	102	✓	919	124	102	1	
7	13:16:03.346	Users 1-17	HTTP Request	99	✓	919	124	99	2	
8	13:16:03.343	Users 1-15	HTTP Request	102	✓	919	124	102	1	
9	13:16:03.339	Users 1-14	HTTP Request	107	✓	919	124	107	2	
10	13:16:03.338	Users 1-4	HTTP Request	108	✓	919	124	108	3	
11	13:16:03.337	Users 1-13	HTTP Request	109	✓	919	124	109	0	
12	13:16:03.334	Users 1-12	HTTP Request	112	✓	919	124	112	1	
13	13:16:03.332	Users 1-11	HTTP Request	114	✓	919	124	114	1	
14	13:16:03.330	Users 1-10	HTTP Request	117	✓	919	124	117	1	
15	13:16:03.325	Users 1-8	HTTP Request	122	✓	919	124	122	1	
16	13:16:03.320	Users 1-6	HTTP Request	127	✓	919	124	127	1	
17	13:16:03.317	Users 1-5	HTTP Request	130	✓	919	124	130	1	
18	13:16:03.349	Users 1-18	HTTP Request	101	✓	919	124	101	0	
19	13:16:03.349	Users 1-7	HTTP Request	101	✓	919	124	101	1	
20	13:16:03.453	Users 1-47	HTTP Request	74	✓	919	124	74	1	
21	13:16:03.452	Users 1-56	HTTP Request	76	✓	919	124	76	1	
22	13:16:03.452	Users 1-43	HTTP Request	76	✓	919	124	76	1	
23	13:16:03.450	Users 1-46	HTTP Request	79	✓	919	124	79	1	
24	13:16:03.450	Users 1-21	HTTP Request	80	✓	919	124	80	1	
25	13:16:03.454	Users 1-55	HTTP Request	82	✗	919	124	82	1	

Scroll automatically? Child samples? No of Samples 400 Latest Sample 1131 Average 1121 Deviation 359

400 concurrent users, with connection pooling, avg time = 858ms

View Results in Table										
Name: View Results in Table										
Comments:										
Write results to file / Read from file:										
<input type="text"/> Filename										
<input type="button" value="Browse ..."/>										
<input type="checkbox"/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:33:35.411	Users 1-1	HTTP Request	19	✓	919	124	19	4	
2	13:33:35.432	Users 1-2	HTTP Request	20	✓	919	124	20	1	
3	13:33:35.434	Users 1-7	HTTP Request	25	✓	919	124	25	1	
4	13:33:35.437	Users 1-3	HTTP Request	25	✓	919	124	25	3	
5	13:33:35.440	Users 1-4	HTTP Request	30	✓	919	124	30	1	
6	13:33:35.450	Users 1-8	HTTP Request	29	✓	919	124	28	1	
7	13:33:35.456	Users 1-5	HTTP Request	27	✓	919	124	27	1	
8	13:33:35.458	Users 1-6	HTTP Request	29	✓	919	124	29	1	
9	13:33:35.530	Users 1-9	HTTP Request	15	✓	919	124	15	2	
10	13:33:35.532	Users 1-10	HTTP Request	22	✓	919	124	22	1	
11	13:33:35.536	Users 1-11	HTTP Request	27	✓	919	124	26	1	
12	13:33:35.540	Users 1-12	HTTP Request	38	✓	919	124	38	1	
13	13:33:35.544	Users 1-13	HTTP Request	37	✓	919	124	37	3	
14	13:33:35.548	Users 1-14	HTTP Request	35	✓	919	124	35	1	
15	13:33:35.551	Users 1-15	HTTP Request	34	✓	919	124	34	1	
16	13:33:35.553	Users 1-16	HTTP Request	35	✓	919	124	35	1	
17	13:33:35.577	Users 1-17	HTTP Request	22	✓	919	124	22	1	
18	13:33:35.592	Users 1-18	HTTP Request	13	✓	919	124	13	1	
19	13:33:35.597	Users 1-19	HTTP Request	16	✓	919	124	16	1	
20	13:33:35.606	Users 1-20	HTTP Request	20	✓	919	124	20	2	
21	13:33:35.610	Users 1-21	HTTP Request	33	✓	919	124	33	1	
22	13:33:35.613	Users 1-22	HTTP Request	35	✓	919	124	35	1	
23	13:33:35.616	Users 1-23	HTTP Request	47	✓	919	124	47	1	
24	13:33:35.620	Users 1-24	HTTP Request	56	✓	919	124	56	2	
25	13:33:35.623	Users 1-25	HTTP Request	58	✗	919	124	58	1	

Scroll automatically? Child samples? No of Samples 400 Latest Sample 906 Average 858 Deviation 297

500 concurrent users, without connection pooling, avg time = 1152ms

View Results in Table										
Name: View Results in Table										
Comments:										
Write results to file / Read from file										
Filename					Browse...	Log/Display Only:	<input type="checkbox"/> Errors	<input type="checkbox"/> Successes	<input type="button" value="Configure"/>	
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:16:49.245	Users 1-1	HTTP Request	18	✓	919	124	18	3	
2	13:16:49.259	Users 1-8	HTTP Request	33	✓	919	124	33	1	
3	13:16:49.258	Users 1-5	HTTP Request	38	✓	919	124	38	1	
4	13:16:49.254	Users 1-3	HTTP Request	45	✓	919	124	45	2	
5	13:16:49.252	Users 1-2	HTTP Request	50	✓	919	124	50	1	
6	13:16:49.251	Users 1-4	HTTP Request	50	✓	919	124	50	2	
7	13:16:49.290	Users 1-13	HTTP Request	32	✓	919	124	32	1	
8	13:16:49.288	Users 1-18	HTTP Request	34	✓	919	124	34	1	
9	13:16:49.296	Users 1-17	HTTP Request	28	✓	919	124	28	12	
10	13:16:49.296	Users 1-9	HTTP Request	130	✓	919	124	130	13	
11	13:16:49.293	Users 1-10	HTTP Request	234	✓	919	124	234	16	
12	13:16:49.309	Users 1-7	HTTP Request	255	✓	919	124	255	1	
13	13:16:49.306	Users 1-11	HTTP Request	307	✓	919	124	307	15	
14	13:16:49.375	Users 1-31	HTTP Request	296	✓	919	124	296	1	
15	13:16:49.373	Users 1-21	HTTP Request	301	✓	919	124	301	2	
16	13:16:49.370	Users 1-27	HTTP Request	309	✓	919	124	309	2	
17	13:16:49.368	Users 1-26	HTTP Request	316	✓	919	124	316	1	
18	13:16:49.364	Users 1-25	HTTP Request	326	✓	919	124	326	2	
19	13:16:49.360	Users 1-24	HTTP Request	335	✓	919	124	335	2	
20	13:16:49.355	Users 1-20	HTTP Request	354	✓	919	124	354	1	
21	13:16:49.350	Users 1-19	HTTP Request	362	✓	919	124	362	1	
22	13:16:49.342	Users 1-16	HTTP Request	379	✓	919	124	379	1	
23	13:16:49.336	Users 1-15	HTTP Request	386	✓	919	124	386	3	
24	13:16:49.330	Users 1-14	HTTP Request	393	✓	919	124	393	1	
25	13:16:49.311	Users 1-12	HTTP Request	416	✗	919	124	416	13	

500 concurrent users, with connection pooling, avg time = 814ms

View Results in Table										
Name: View Results in Table										
Comments:										
Write results to file / Read from file										
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)	
1	13:34:09.024	Users 1-2	HTTP Request	232	✓	919	124	232	3	
2	13:34:09.030	Users 1-4	HTTP Request	230	✓	919	124	230	2	
3	13:34:09.031	Users 1-6	HTTP Request	232	✓	919	124	232	1	
4	13:34:09.039	Users 1-11	HTTP Request	228	✓	919	124	228	2	
5	13:34:09.034	Users 1-8	HTTP Request	235	✓	919	124	235	7	
6	13:34:09.044	Users 1-10	HTTP Request	228	✓	919	124	228	1	
7	13:34:09.046	Users 1-5	HTTP Request	230	✓	919	124	230	1	
8	13:34:09.050	Users 1-12	HTTP Request	231	✓	919	124	231	1	
9	13:34:09.056	Users 1-13	HTTP Request	229	✓	919	124	229	2	
10	13:34:09.057	Users 1-9	HTTP Request	233	✓	919	124	233	1	
11	13:34:09.058	Users 1-14	HTTP Request	238	✓	919	124	238	1	
12	13:34:09.061	Users 1-17	HTTP Request	239	✓	919	124	239	1	
13	13:34:09.060	Users 1-15	HTTP Request	252	✓	919	124	252	2	
14	13:34:09.062	Users 1-7	HTTP Request	252	✓	919	124	252	1	
15	13:34:09.066	Users 1-18	HTTP Request	251	✓	919	124	251	1	
16	13:34:09.069	Users 1-27	HTTP Request	249	✓	919	124	249	1	
17	13:34:09.069	Users 1-32	HTTP Request	254	✓	919	124	253	1	
18	13:34:09.069	Users 1-24	HTTP Request	257	✓	919	124	257	3	
19	13:34:09.071	Users 1-22	HTTP Request	258	✓	919	124	258	1	
20	13:34:09.072	Users 1-23	HTTP Request	265	✓	919	124	265	1	
21	13:34:09.073	Users 1-21	HTTP Request	267	✓	919	124	267	1	
22	13:34:09.074	Users 1-33	HTTP Request	270	✓	919	124	270	0	
23	13:34:09.076	Users 1-34	HTTP Request	271	✓	919	124	271	1	
24	13:34:09.076	Users 1-20	HTTP Request	280	✓	919	124	280	1	
25	13:34:09.077	Users 1-30	HTTP Request	290	✗	919	124	290	1	

Conclusion

The average response time increases gradually along with number of users till the users are 400, then it decreases slightly in both cases.

The response time for connection pooling is always less than without pooling.

The difference of average between with and without pooling decreases as the number of users increases. Setting a higher pool limit may decrease the average response time.

Mocha Test

```
Canvas
  ✓ GET /course
  ✓ GET /course/search
  ✓ GET /profile
  ✓ GET /course/:id/people
  ✓ GET /course/:id/home

it('GET /course', () => {
  agent.get('/course').query({ currentUser:"101"})
    .then((res)=>{
      let expect = chai.expect;
      expect(res.body.courses).to.have.lengthOf(4);
    });
});

it('GET /course/search', () => {
  agent.get('/course/search').query({ currentUser:"101"})
    .then((res)=>{
      let expect = chai.expect;
      expect(res.body.courses).to.have.lengthOf(9);
    });
});

it('GET /profile', () => {
  agent.get('/profile').query({ currentUser:"101"})
    .then((res)=>{
      let expect = chai.expect;
      expect(res.body.name).to.equal("Colt Steele");
    });
});

it('GET /course/:id/people', () => {
  agent.get('/course/:id/people').query({ currentUser:"101", cid:"CMPE202"})
    .then((res)=>{
      let expect = chai.expect;
      expect(res.body.people).to.have.lengthOf(4);
    });
});

it('GET /course/:id/home', () => {
  agent.get('/course/:id/home').query({ currentUser:"101", cid:"CMPE202"})
    .then((res)=>{
      let expect = chai.expect;
      expect(res.body.course.fid).to.equal("101");
    });
});
```

Questions

1) Broadly speaking, passwords can be stored in any of the following ways: -

Plaintext: The exact password entered by user is stored in database which offers no security against attacks.

Encryption: It converts a plaintext to ciphertext using encryption algorithm and a key. Decryption follows decryption algorithm using the same key. It requires a key for both processes.

Hashing: It requires no key. Passwords are encrypted using a one-way hash function which are computationally very expensive to crack. Two users cannot have same salt implies their passwords can never be same.

Hash and Salt: Salt is a randomly generated string of characters which can be append or prepend to the passwords. Incorporating salts and hashing the same makes many attacks like dictionary or brute-force attacks futile.

Modern hashing algorithm are MD-5, SHA-1, SHA-2 and SHA-3. There are currently 3 main password hashing algorithms:

- PBKDF2
- bcrypt
- scrypt.

The passwords in this application are encrypted using bcrypt algorithm which is based on Blowfish Cipher. Since it's a type of Blowfish cipher, it provides good encryption rate and currently no effective cryptanalysis of it has been found. To generate hash, it uses look-up table which is loaded into memory; so it will utilize memory space. It uses salt iteratively to resist rainbow and brute-force attack even with increasing computational power. It uses timestamp so even the same password will be hashed to different value at different instance of time and generates long hash containing numbers, letters and special characters.

This algorithm is programmed by the bcrypt node module available to generate salt, hash it and compare password with its hash. Higher salt rounds can achieve stronger passwords but it is memory consuming.

2) Connection pooling is a technique where database connections are cached so that they can be reused in future instead of establishing and releasing a connection every time data needs to be accessed. It is used to improve performance of executing codes in database. It also cuts down time when a user must wait to establish a new connection.

Comparing the results of jmeter test, it is clear that the average time increases in a polynomial manner as the concurrent users increases. The time is always less for pooled connection and the difference between pooled and without pooled connection decrease as number of concurrent users increases. In the testing case, the poolLimit is set to 100. As the number of concurrent users grow, the response time slope gradually decreases. This declination can be assumed to decrease to a lesser extent if the poolLimit is set higher which in turn will give better performance. But eventually, as the number of users approach or cross the poolLimit, the graph gets shallow.

If I were to implement a connection pooling technique, it would work as follows:

Since threads are light-weight processes, they can be used to create and release a connection to the database. We can have a pool of reusable threads. The pool data structure provides with a free thread to an incoming request. If the number of requests passes the number of threads, the system will facilitate by adding a new thread to the pool. The number of threads added this way must have a certain limit. When the server is less busy, the newly created threads can be released. This way, connection to database will be established only once, hence reducing overhead due to creating new connections.

As long as data is being read, the threads can work concurrently. Similarly, when threads are being used to write data to different locations they will work concurrently. When threads are writing to same data, they must wait in a queue. The concept of semaphores can be helpful for the same. Threads will be maintained in first-come-first-serve bases written accordingly. We can also use thread priority to surpass or condone the operation. For example, a system admin's thread priority will be higher than normal user and it must access the database first. Trivial processes which doesn't affect much to the database can have less thread priority.

3) Caching refers to storing frequently used data so that it can be retrieved quickly, which otherwise is a resource intensive process. The idea is same in SQL caching. SQL has a buffer cache which it uses to store table or index data pages once they are modified or read from the disk. These pages are same as they appear on the disk. If the application has to access data, it will search in cache and if it is not found, only then it will search for the same in database. The cached data can be read as long as it is not changed. Once the data changes, the system must update the cache. Thus, SQL caching aims to reduce I/O tasks and hence reduce data retrieval and response time.

There are several caching strategies, which depends on data and data access pattern They are: -

- Cache-aside
- Read-through Cache
- Write-through Cache
- Write-around
- Write-back

According to me, combination of Read-through and Write-through technique will be the most efficient way to cache data. These techniques have a direct access to cache and indirect access to database, through cache. Database is only accessed if a cache-miss occurs in case of read; in case of write, the data is always written to cache before writing to database.

Since the number of students is way more than faculty, the number of read/write operations for students will always be more than that for faculty. Suppose a faculty makes a new announcement or uploads some files, then the students under those faculty and his course will read the same data in large number. Therefore, new announcement or data common for students can be stored in the cache for faster access. Similarly, submitting assignments will be in large number and in case of quiz it can be concurrent. In this case, the write-through cache will store the data in cache prior to writing it to database. By the time it starts caching new write data, the previous ones can be written to database simultaneously. For faculty, the read/write data can be stored in cache but only temporarily. Also, it is assumed the database connection is made through pooling which is much lighter than the traditional way.

4) Scalability means a system, network or process capable of handling growing workload or the potential to accommodate the same.

Vertically scalability refers to extending the capacity of server by adding CPU, memory, I/O and networking components to address workload requirements.

When the server gets overloaded with incoming requests and does not have enough resources, they can be allocated dynamically. Horizontally scalable allows adding multiple hardware which work as a single entity. If the server is clustered, it can be scaled horizontally to provide availability and performance. It allows scalability on the fly. Some examples of horizontally scalable are MongoDB, Cassandra, Google Cloud Spanner, etc.

My current application's session strategy is not horizontally scalable because sessions are stored as states. The session is maintained using cookies for each user in their browser as well as in the server's database. Which means authenticating user for every request requires accessing the database and validating the same. This can be very expensive process. This can be overcome by using what is known as JSON Web Token (JWT). JWT is basically a string that is sent as part of http request to validate authenticity of user. This string is not stored in database but can be stored on client side. JWT is made with a secret which is private to the user. Every request can be verified for authenticity through this secret key. Attempts to change the JWT will lead to verification failure. Hence the application can be made horizontally scalable with help of JWT. This resolves overhead due to accessing stored session cookie for every user for every request.

Github commit history

History for CMPE273-SP19-8 / Lab1-013823108

- ↳ Commits on Mar 17, 2019
 - Download Files
shivangmistry committed 6 hours ago
- ↳ Commits on Mar 16, 2019
 - Assignment and Submission
shivangmistry committed 2 days ago
 - Lecture Notes
shivangmistry committed 2 days ago
- ↳ Commits on Mar 14, 2019
 - Grades
shivangmistry committed 3 days ago
- ↳ Commits on Mar 9, 2019
 - Quiz
shivangmistry committed 9 days ago
- ↳ Commits on Mar 8, 2019
 - Announcements
shivangmistry committed 10 days ago

- ↳ Commits on Mar 7, 2019
 - Classmates
shivangmistry committed 10 days ago
 - Course Menu and navigation
shivangmistry committed 11 days ago
- ↳ Commits on Mar 6, 2019
 - Enrollment, Drop, Waitlist
shivangmistry committed 11 days ago
- ↳ Commits on Mar 5, 2019
 - Display Courses
shivangmistry committed 12 days ago
 - Create and Search Courses
shivangmistry committed 13 days ago
- ↳ Commits on Mar 4, 2019
 - View and Edit Profile
shivangmistry committed 13 days ago
 - Landing Page and Session Handling
shivangmistry committed 13 days ago
- ↳ Commits on Mar 3, 2019
 - Canvas - Initial Commit
shivangmistry committed 15 days ago

- ↳ Commits on Feb 28, 2019
 - Final Commit
shivangmistry committed 18 days ago
- ↳ Commits on Feb 26, 2019
 - Calculator - initial commit
shivangmistry committed 19 days ago

Newer Older