

CLOUD COMPUTING

LAB – 2: MONOLITHIC ARCHITECTURE

NAME: ADITYA VENKATESH

SRN: PES1UG23AM024

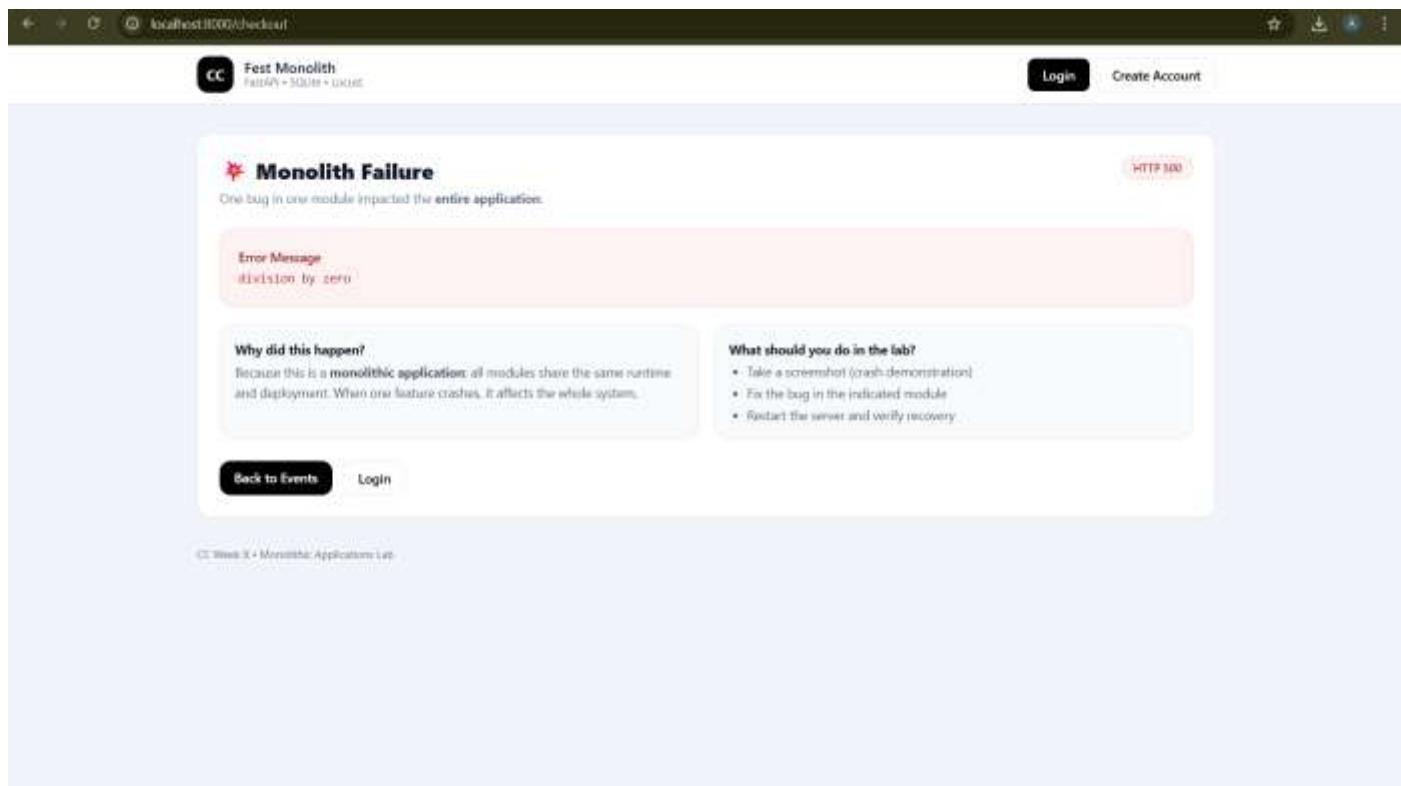
SCREENSHOT – 1: EVENTS PAGE LOADED

The screenshot shows a web browser window for the 'Fest Monolith' application at the URL `localhost:8000/events/user=PES1UG23AM024`. The page is titled 'Events' and features a header with navigation links for 'Events', 'My Events', 'Checkout', and 'Logout'. A sub-header says 'Welcome PES1UG23AM024. Register for events below.' Below this, there are nine event cards arranged in a 3x3 grid:

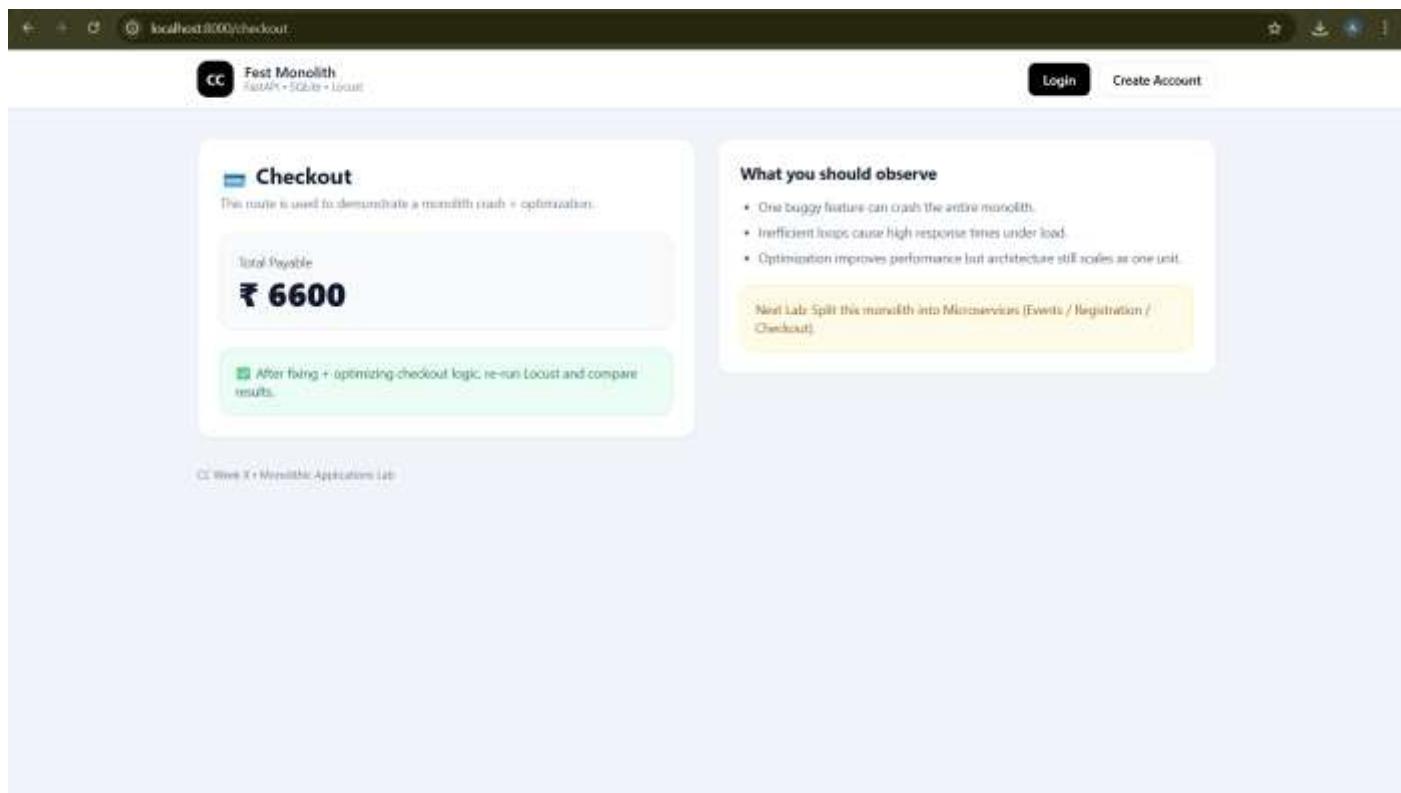
- Event ID: 1** (₹ 500)
Hackathon
Includes certificate • instant registration • limited seats
Register
- Event ID: 2** (₹ 300)
Dance
Includes certificate • instant registration • limited seats
Register
- Event ID: 3** (₹ 500)
Hackathon
Includes certificate • instant registration • limited seats
Register
- Event ID: 4** (₹ 500)
Dance Battle
Includes certificate • instant registration • limited seats
Register
- Event ID: 5** (₹ 400)
AI Workshop
Includes certificate • instant registration • limited seats
Register
- Event ID: 6** (₹ 200)
Photography Walk
Includes certificate • instant registration • limited seats
Register
- Event ID: 7** (₹ 250)
Gaming Tournament
Includes certificate • instant registration • limited seats
Register
- Event ID: 8** (₹ 250)
Music Night
Includes certificate • instant registration • limited seats
Register
- Event ID: 9** (₹ 150)
Treasure Hunt
Includes certificate • instant registration • limited seats
Register

SCREENSHOT – 2: MONOLITH FAILURE

```
INFO:    127.0.0.1:51934 - "GET /my-events?user=PES1UG23AM024 HTTP/1.1" 200 OK
INFO:    127.0.0.1:51934 - "GET /checkout HTTP/1.1" 500 Internal Server Error
ERROR:   Exception in ASGI application
Traceback (most recent call last):
```



SCREENSHOT – 3: FIXED PAGE



SCREENSHOT – 4: LOCUST DASHBOARD AND TERMINAL

**LOCUST**Performance Testing for Microservices

Host: http://localhost:8000 Status: RUNNING Users: 1 RPS: 0.43 Failures: 0%

STATISTICS CHARTS FAILURES EXCEPTIONS CURRENT RATIO DOWNLOAD DATA LOGS

Type	Name	# Requests	# Fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	itcheckout	5	0	18	2000	2000	419.05	5	2035	2797	0.43	0
	Aggregated	5	0	18	2000	2000	419.05	5	2035	2797	0.43	0

SCREENSHOT – 5: DASHBOARD AFTER CODE UPDATE

Locust Performance Test Report										Host	Status	RPS	Failures	Actions		
										http://localhost:8000	STOPPED	0.0	0%	<button>NEW</button>	<button>RESET</button>	
Statistics		Charts		Failures		Exceptions		Current Ratio		Download Data		Logs		Metrics		
Type	Name	# Requests	# Fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures	Success Rate	Latency	Throughput	
GET	checkout	10	0	10	2100	2100	122.52	8	2063	2797	0.0	0	100.0%	100-200 ms	1000 RPS	
Aggregated		10	0	10	2100	2100	122.52	8	2063	2797	0.0	0	100.0%	100-200 ms	1000 RPS	

SCREENSHOT – 6: EVENTS ROUTE BEFORE OPTIMIZATION

Locust Performance Test Results												
Test Configuration		Performance Metrics			Request Data							
Type	Name	# Requests	# Fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/events?user=locust_user	10	0	530	2600	2600	720.75	381	2547	21138	0.6	0
Aggregated		10	0	530	2600	2600	720.75	381	2547	21138	0.6	0

```

<venv> PS C:\Users\adity\Documents\PES1UG23AM024\CC_Lab2> locust -f locust/events_locustfile.py
[2026-01-19 15:04:30,489] Adriana/INFO/locust.main: Starting Locust 2.43.1
[2026-01-19 15:04:38,458] Adriana/INFO/locust.main: Starting web interface at http://localhost:8089, press enter to open your default browser.
[2026-01-19 15:05:58,293] Adriana/INFO/locust.runners: Ramping to 1 users at a rate of 1.00 per second
[2026-01-19 15:05:58,299] Adriana/INFO/locust.runners: All users spawned: {'EventsUser': 1} {1 total users}
Traceback (most recent call last):
  File "C:\Users\adity\Documents\PES1UG23AM024\CC_Lab2\venv\Lib\site-packages\gevent\_ffi\loop.py", line 279, in python_check_callback
    def python_check_callback(self, watcher_ptr): # pylint:disable=unused-argument

KeyboardInterrupt
2026-01-19T09:37:10Z
[2026-01-19 15:07:18,662] Adriana/INFO/locust.main: Shutting down (exit code 8)
Type      Name          # reqs   # fails | Avg     Min     Max     Med | req/s  failures/s
GET      /events?user=locust_user           14      8(6.00%) | 671     389     2546    538 | 0.49    0.00
Aggregated                         14      8(6.00%) | 671     389     2546    538 | 0.49    0.00

Response time percentiles (approximated)
Type      Name          50%   60%   75%   80%   90%   95%   98%   99%   99.9% 99.99
% 18% # reqs
|-----|
|-----|
GET      /events?user=locust_user           530   578   579   600   660   2588   2588   2588   2588   258
0  2500   14
|-----|
Aggregated                         530   578   579   600   660   2588   2588   2588   2588   258
0  2500   14

<venv> PS C:\Users\adity\Documents\PES1UG23AM024\CC_Lab2>

```

SCREENSHOT – 7: EVENTS ROUTE AFTER OPTIMIZATION

Locust Performance Test Report												
Summary Metrics												
Type	Name	# Requests	# Fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures
GET	/events	28	0	17	33	2100	91.81	13	2056	21138	1	0
	Aggregated	28	0	17	33	2100	91.81	13	2056	21138	1	0

1. What was the bottleneck?

Unnecessary random wait calculation and dynamic URL aggregation during results reporting.

2. What change did you make?

Replaced randomized wait with a constant delay and added a static endpoint name in events_locustfile.py. In main.py, I reduced the loop from 3,000,000 to 30,000 cutting unnecessary CPU work by 100x, significantly lowering endpoint response time.

3. Why did the performance improve?

Reduced per-request overhead and reporting noise, allowing Locust to spend more time generating load and less time computing waits and formatting stats.

SCREENSHOT – 8: MY-EVENTS ROUTE BEFORE OPTIMIZATION

The screenshot shows the Locust web interface with the following details:

- Host:** http://localhost:8000
- Status:** CLEANUP
- RPS:** 0.6
- Failures:** 0%

Below the header, there are tabs for STATISTICS, CHARTS, FAILURES, EXCEPTIONS, CURRENT RATIO, DOWNLOAD DATA, and LOGS. The STATISTICS tab is selected.

Type	Name	# Requests	# Fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures
GET	my-events?user=locust_user	17	0	240	2200	2200	346.2	155	2189	3144	0.6	0
Aggregated												
17 requests in 0:00:00.000, 0.6 RPS, 0.00% errors												

SCREENSHOT – 9: MY-EVENTS ROUTE AFTER OPTIMIZATION

Locust Performance Test Report												
Host: http://localhost:8000												
Statistics		Charts		Failures		Exceptions		Current Ratio				
Type	Name	# Requests	# fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	my-events	28	28	7	11	2000	79.81	4	2044	90	1	1
	Aggregated	28	28	7	11	2000	79.81	4	2044	90	1	1

```

[venv] PS C:\Users\adity\Documents\PES1UG23AM024\CC_Lab2> locust -f locust/myevents_locustfile.py
[2026-01-19 15:19:18,898] Adriana/INFO/locust.main: Starting Locust 2.43.1
[2026-01-19 15:19:18,891] Adriana/INFO/locust.main: Starting web interface at http://localhost:8089, press enter to open your default browser.
[2026-01-19 15:19:28,446] Adriana/INFO/locust.runners: Spawning 1 users at a rate of 1.00 per second
[2026-01-19 15:19:28,456] Adriana/INFO/locust.runners: All users spawned: {"MyEventsUser": 1} (1 total users)
Traceback (most recent call last):
  File "C:\Users\adity\Documents\PES1UG23AM024\CC_Lab2\.venv\Lib\site-packages\gevent\_ffibuilder.py", line 279, in python_check_callback
    def python_check_callback(self, watcher_ptr): # pylint:disable=unused-argument
KeyboardInterrupt
[2026-01-19T09:58:32Z] Adriana/INFO/locust.main: Shutting down (exit code 1)

```

Type	Name	# reqs	# fails	Avg	Min	Max	Med	req/s	Failures/s
GET	/my-events	28	28(100.00%)	79	3	2013	7	0.95	0.95
	Aggregated	28	28(100.00%)	79	3	2043	7	0.95	0.95

Response time percentiles (approximated)

Type	Name	50%	66%	75%	88%	90%	95%	98%	99%	99.9%	99.99%
% 100% # reqs											
GET	/my-events	7	7	8	8	11	11	2889	2888	2888	2888
	Aggregated	7	7	8	8	11	11	2889	2888	2888	2888
0	2888	28									

1. What was the bottleneck?

Randomized wait-time calculation and dynamic URL aggregation during stats reporting.

2. What change did you make?

Replaced between(1,2) with a constant delay and set a static request name in myevents_locustfile.py. In main.py, I reduced the loop from 3,000,000 to 30,000 cutting unnecessary CPU work by 100x, significantly lowering endpoint response time.

3. Why did the performance improve?

Reduced overhead for wait computation and results grouping, letting Locust generate more consistent load and cleaner metrics.