UE23CS351B: Cloud Computing

Lab-2: Monolithic Architecture

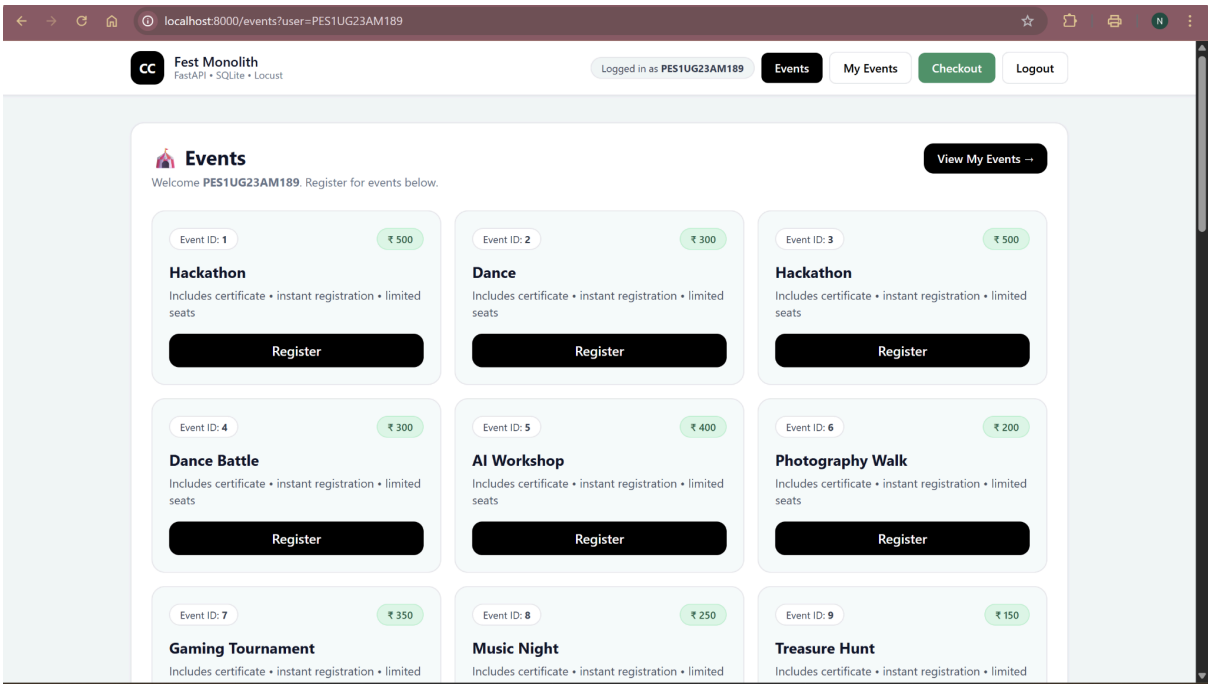
Name: Niharika Paul
SRN: PES1UG23AM189
Semester: 6; Section: AIML 'D'
Date of submission: 21-01-2026

PART-1

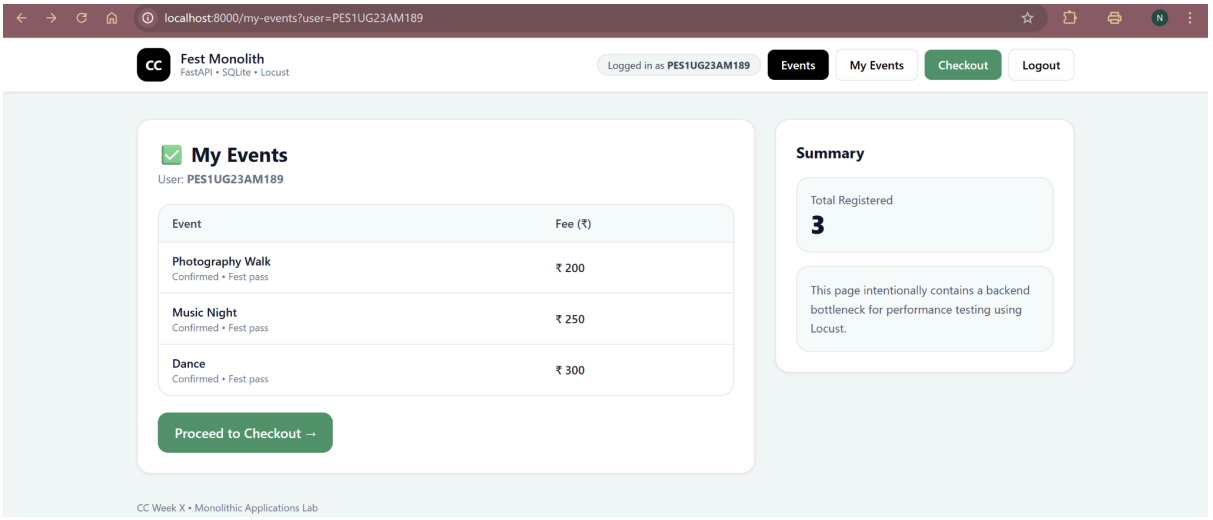
```
(.venv) C:\SEM-6\CC_Lab\PES1UG23AM189\CC_Lab2>python insert_events.py
✅ Events inserted successfully!

(.venv) C:\SEM-6\CC_Lab\PES1UG23AM189\CC_Lab2>uvicorn main:app --reload
INFO:     Will watch for changes in these directories: ['C:\\SEM-6\\CC_Lab\\PES1UG23AM189\\CC_Lab2']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [4600] using StatReload
INFO:     Started server process [25840]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

PART-2

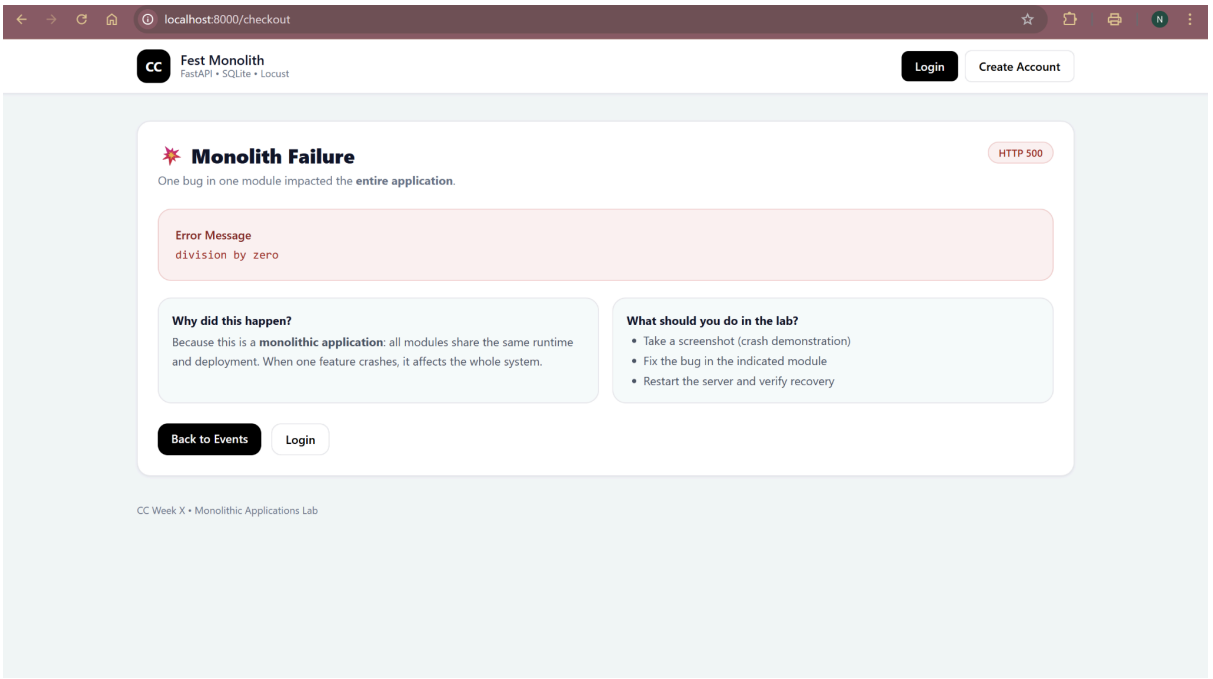
SS1:



PART-3



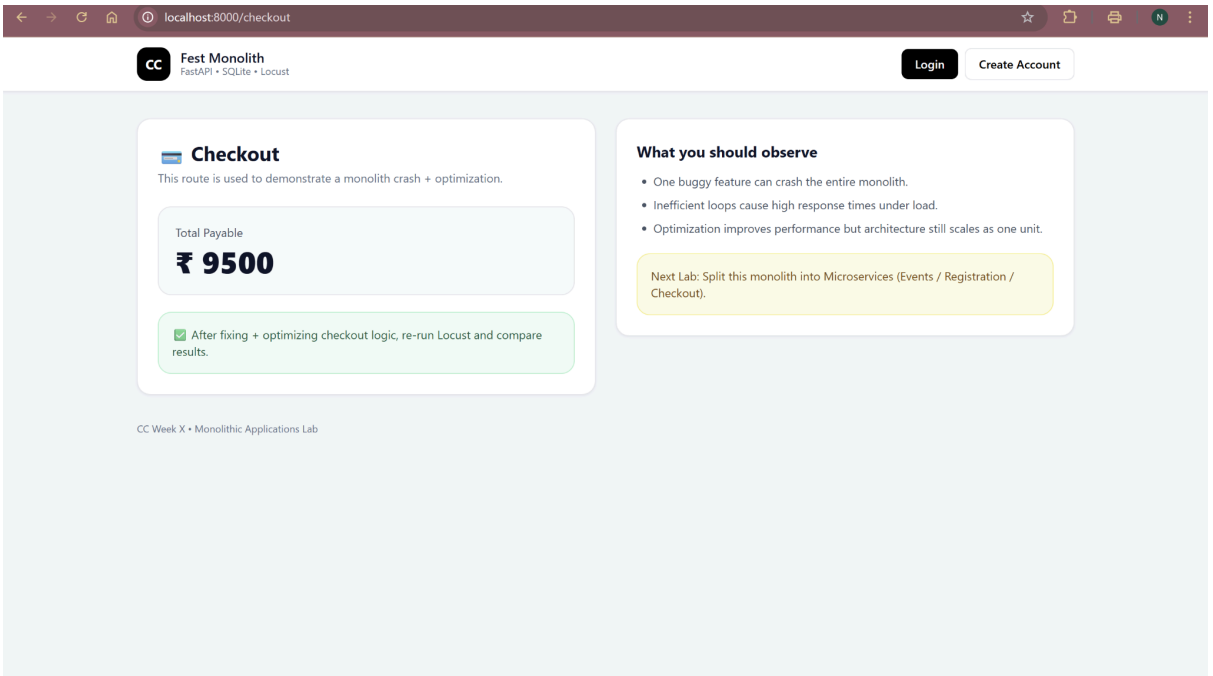
SS2:




```
INFO:     127.0.0.1:51731 - "GET /checkout HTTP/1.1" 500 Internal Server Error
ERROR:    Exception in ASGI application
```

PART-4

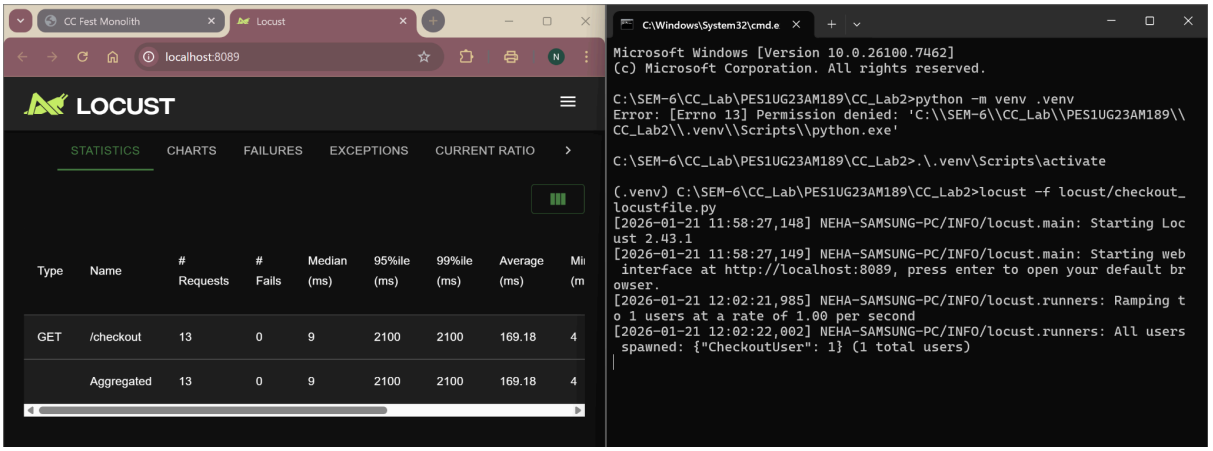
SS3:

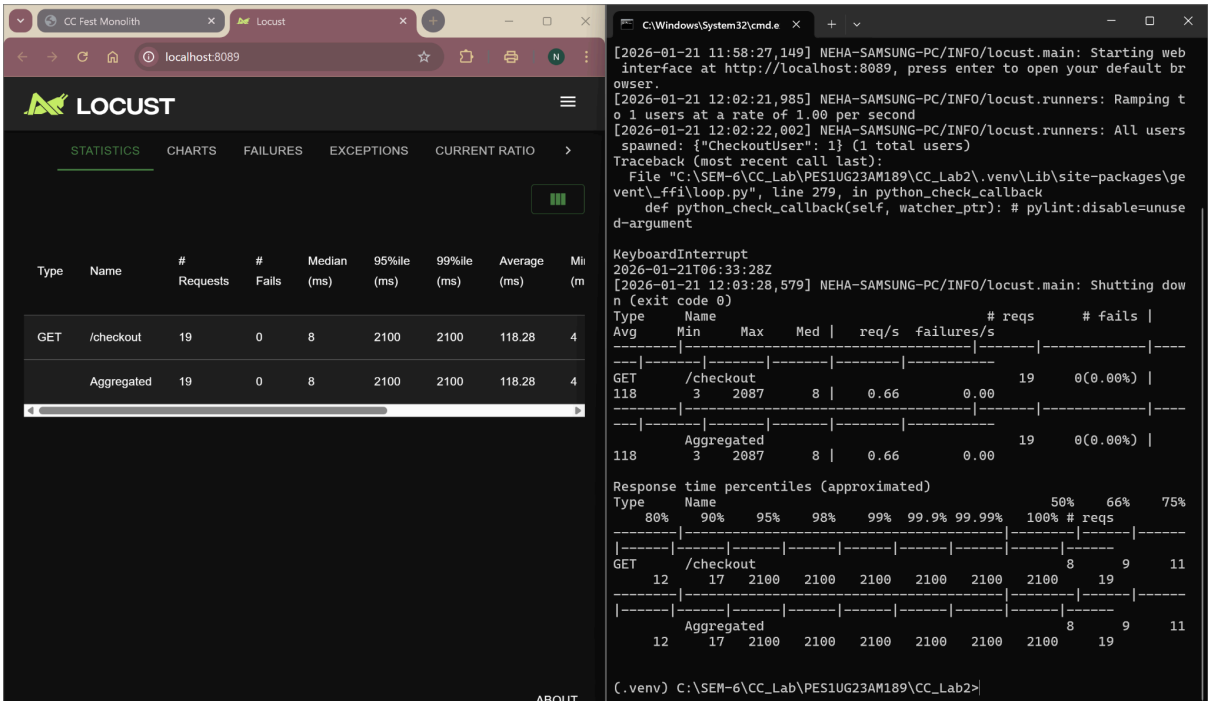

localhost:8000/checkout
CC
Fest Monolith
FastAPI • SQLite • Locust
Login
Create Account
Checkout
This route is used to demonstrate a monolith crash + optimization.
Total Payable
₹ 9500
After fixing + optimizing checkout logic, re-run Locust and compare results.
What you should observe
One buggy feature can crash the entire monolith.
Inefficient loops cause high response times under load.
Optimization improves performance but architecture still scales as one unit.
Next Lab: Split this monolith into Microservices (Events / Registration / Checkout).
CC Week X • Monolithic Applications Lab


INFO: 127.0.0.1:57012 - "GET /checkout HTTP/1.1" 200 OK

PART-5

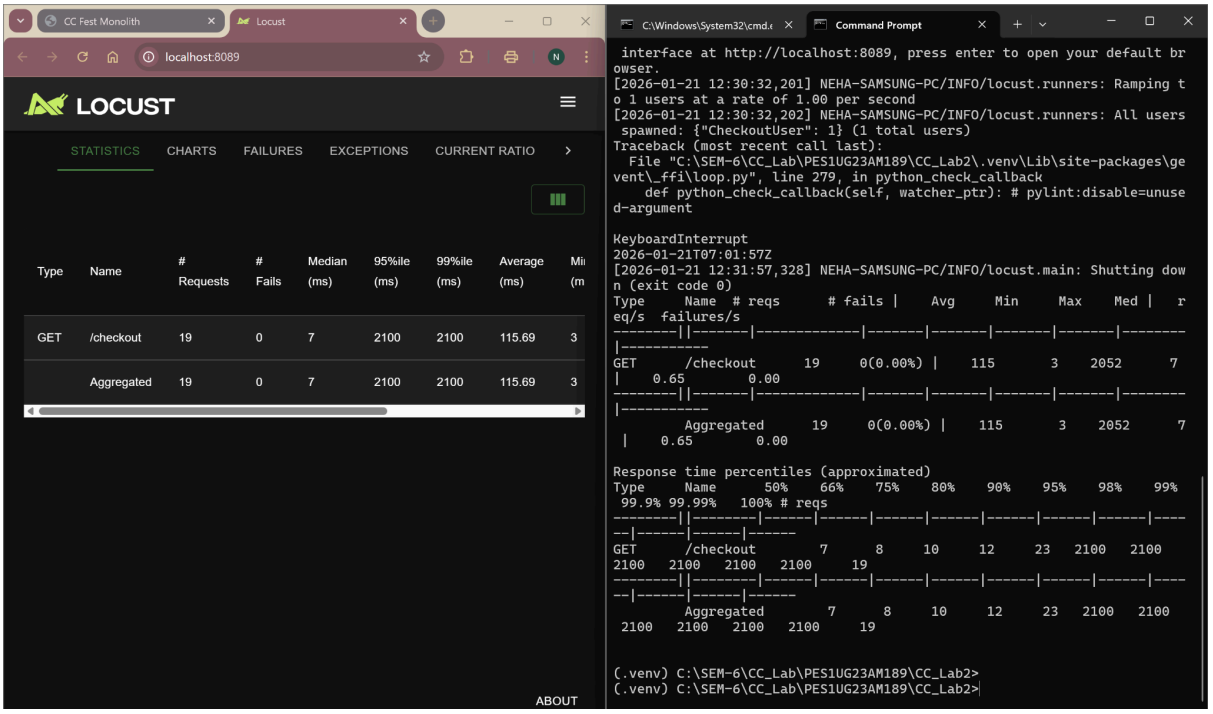
SS4:


CC Fest Monolith
Locust
localhost:8089
LOCUST
STATISTICS
CHARTS
FAILURES
EXCEPTIONS
CURRENT RATIO
Type
Name
Requests
Fails
Median (ms)
95%ile (ms)
99%ile (ms)
Average (ms)
GET
/checkout
13
0
9
2100
2100
169.18
4
Aggregated
13
0
9
2100
2100
169.18
4
C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.26100.7462]
(c) Microsoft Corporation. All rights reserved.
C:\SEM-6\CC_Lab\PES1UG23AM189\CC_Lab2>python -m venv .venv
Error: [Errno 13] Permission denied: 'C:\\SEM-6\\CC_Lab\\PES1UG23AM189\\CC_Lab2\\.venv\\Scripts\\python.exe'
C:\SEM-6\CC_Lab\PES1UG23AM189\CC_Lab2>.\.venv\Scripts\activate
(.venv) C:\SEM-6\CC_Lab\PES1UG23AM189\CC_Lab2>locust -f locust/checkout_locustfile.py
[2026-01-21 11:58:27,148] NEHA-SAMSUNG-PC/INFO/locust.main: Starting Locust 2.43.1
[2026-01-21 11:58:27,149] NEHA-SAMSUNG-PC/INFO/locust.main: Starting web interface at http://localhost:8089, press enter to open your default browser.
[2026-01-21 12:02:21,985] NEHA-SAMSUNG-PC/INFO/locust.runners: Ramping to 1 users at a rate of 1.00 per second
[2026-01-21 12:02:22,002] NEHA-SAMSUNG-PC/INFO/locust.runners: All users spawned: {"CheckoutUser": 1} (1 total users)


CC Fest Monolith
Locust
localhost:8089
LOCUST
STATISTICS
CHARTS
FAILURES
EXCEPTIONS
CURRENT RATIO
Type
Name
Requests
Fails
Median (ms)
95%ile (ms)
99%ile (ms)
Average (ms)
GET
/checkout
19
0
8
2100
2100
118.28
4
Aggregated
19
0
8
2100
2100
118.28
4
ABOUT
C:\Windows\System32\cmd.e
[2026-01-21 11:58:27,149] NEHA-SAMSUNG-PC/INFO/locust.main: Starting web interface at http://localhost:8089, press enter to open your default browser.
[2026-01-21 12:02:21,985] NEHA-SAMSUNG-PC/INFO/locust.runners: Ramping to 1 users at a rate of 1.00 per second
[2026-01-21 12:02:22,002] NEHA-SAMSUNG-PC/INFO/locust.runners: All users spawned: {"CheckoutUser": 1} (1 total users)
Traceback (most recent call last):
File "C:\SEM-6\CC_Lab\PES1UG23AM189\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
2026-01-21T06:33:28Z
[2026-01-21 12:03:28,579] NEHA-SAMSUNG-PC/INFO/locust.main: Shutting down (exit code 0)
Type Name # reqs # fails | Avg Min Max Med | req/s failures/s
GET /checkout 19 0(0.00%) | 118 3 2087 8 | 0.66 0.00
Aggregated 19 0(0.00%) | 118 3 2087 8 | 0.66 0.00
Response time percentiles (approximated)
Type Name 50% 66% 75% 80% 90% 95% 98% 99% 99.9% 99.99% 100% # reqs
GET /checkout 8 9 11 12 17 2100 2100 2100 2100 2100 2100 19
Aggregated 8 9 11 12 17 2100 2100 2100 2100 2100 2100 19
(.venv) C:\SEM-6\CC_Lab\PES1UG23AM189\CC_Lab2>

PART-6

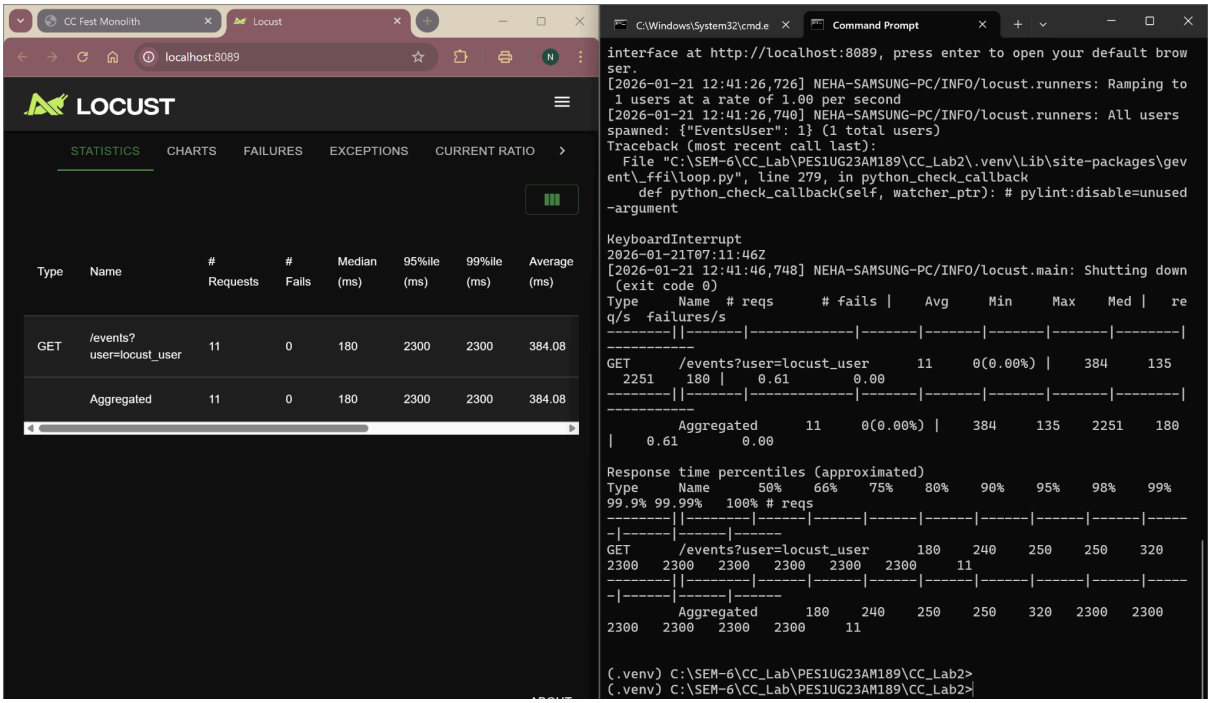
SS5:


CC Fest Monolith
Locust
localhost:8089
LOCUST
STATISTICS
CHARTS
FAILURES
EXCEPTIONS
CURRENT RATIO
Type
Name
Requests
Fails
Median (ms)
95%ile (ms)
99%ile (ms)
Average (ms)
GET
/checkout
19
0
7
2100
2100
115.69
3
Aggregated
19
0
7
2100
2100
115.69
3
ABOUT
C:\Windows\System32\cmd.e
Command Prompt
interface at http://localhost:8089, press enter to open your default browser.
[2026-01-21 12:30:32,201] NEHA-SAMSUNG-PC/INFO/locust.runners: Ramping to 1 users at a rate of 1.00 per second
[2026-01-21 12:30:32,202] NEHA-SAMSUNG-PC/INFO/locust.runners: All users spawned: {"CheckoutUser": 1} (1 total users)
Traceback (most recent call last):
File "C:\SEM-6\CC_Lab\PES1UG23AM189\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
2026-01-21T07:01:57Z
[2026-01-21 12:31:57,328] NEHA-SAMSUNG-PC/INFO/locust.main: Shutting down (exit code 0)
Type Name # reqs # fails | Avg Min Max Med | req/s failures/s
GET /checkout 19 0(0.00%) | 115 3 2052 7 | 0.65 0.00
Aggregated 19 0(0.00%) | 115 3 2052 7 | 0.65 0.00
Response time percentiles (approximated)
Type Name 50% 66% 75% 80% 90% 95% 98% 99% 99.9% 99.99% 100% # reqs
GET /checkout 7 8 10 12 23 2100 2100 2100 2100 2100 2100 19
Aggregated 7 8 10 12 23 2100 2100 2100 2100 2100 2100 19
(.venv) C:\SEM-6\CC_Lab\PES1UG23AM189\CC_Lab2>
(.venv) C:\SEM-6\CC_Lab\PES1UG23AM189\CC_Lab2>

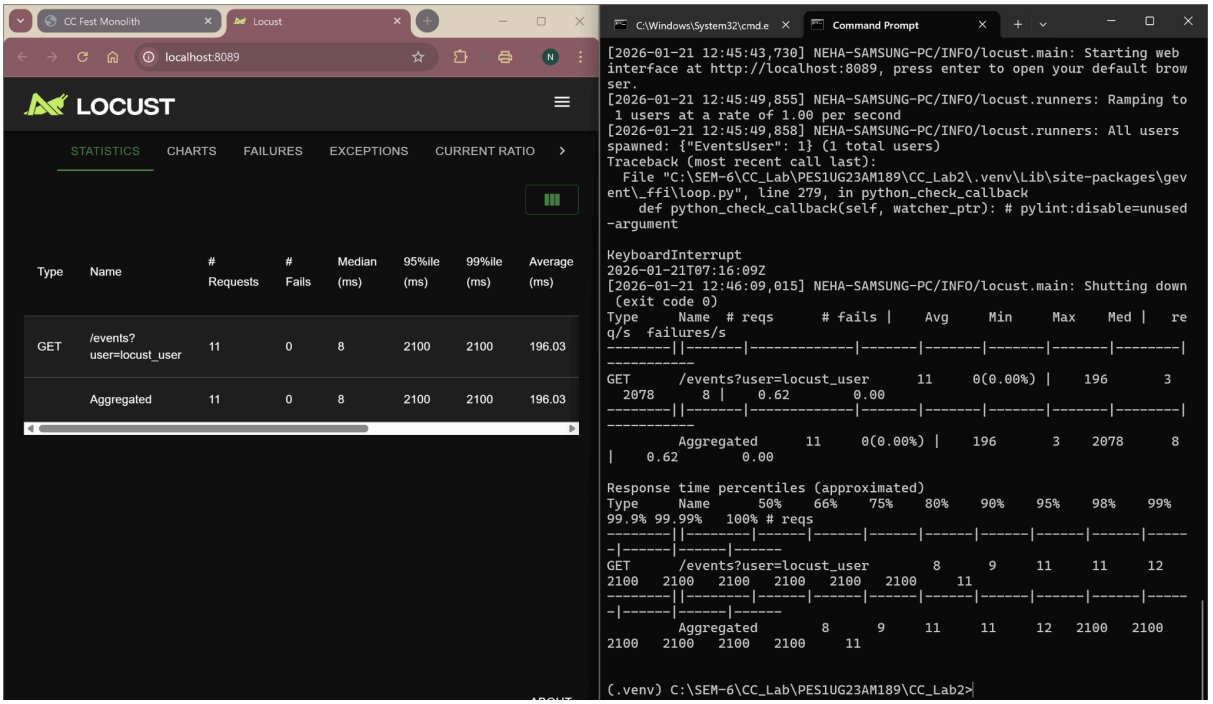
PART-7

Route 1: /events

SS6 *(BEFORE optimization)*:

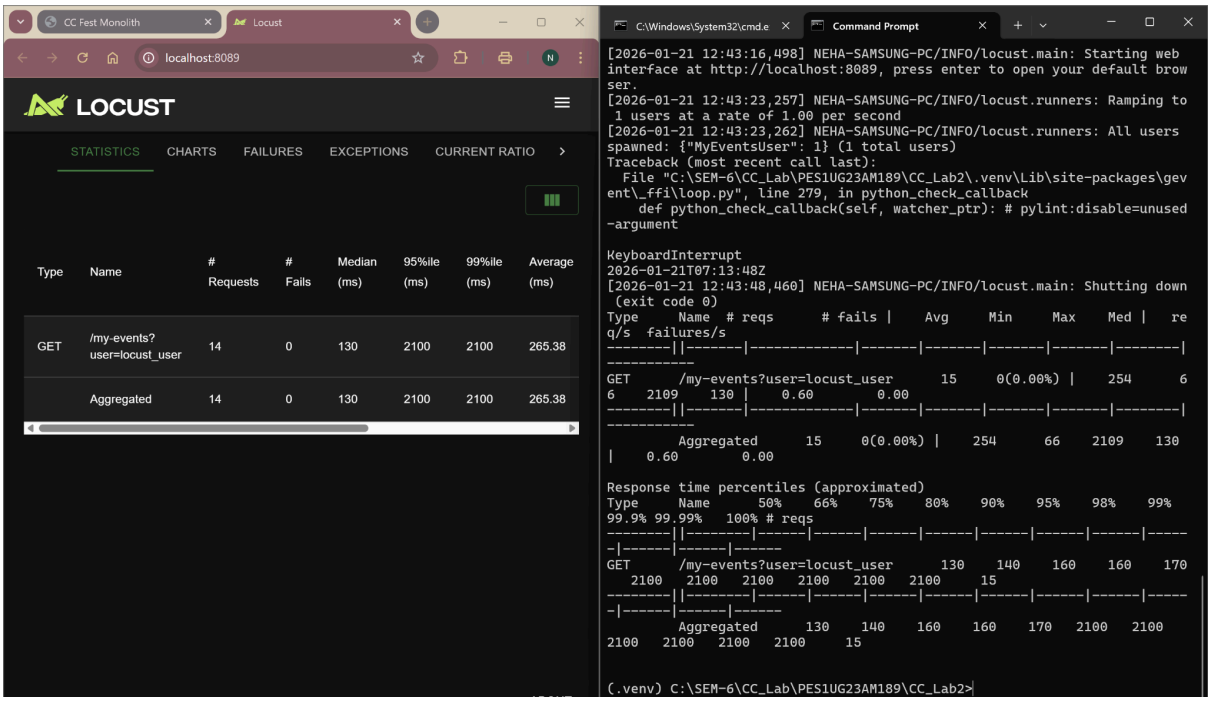


SS7 *(AFTER optimization)*:

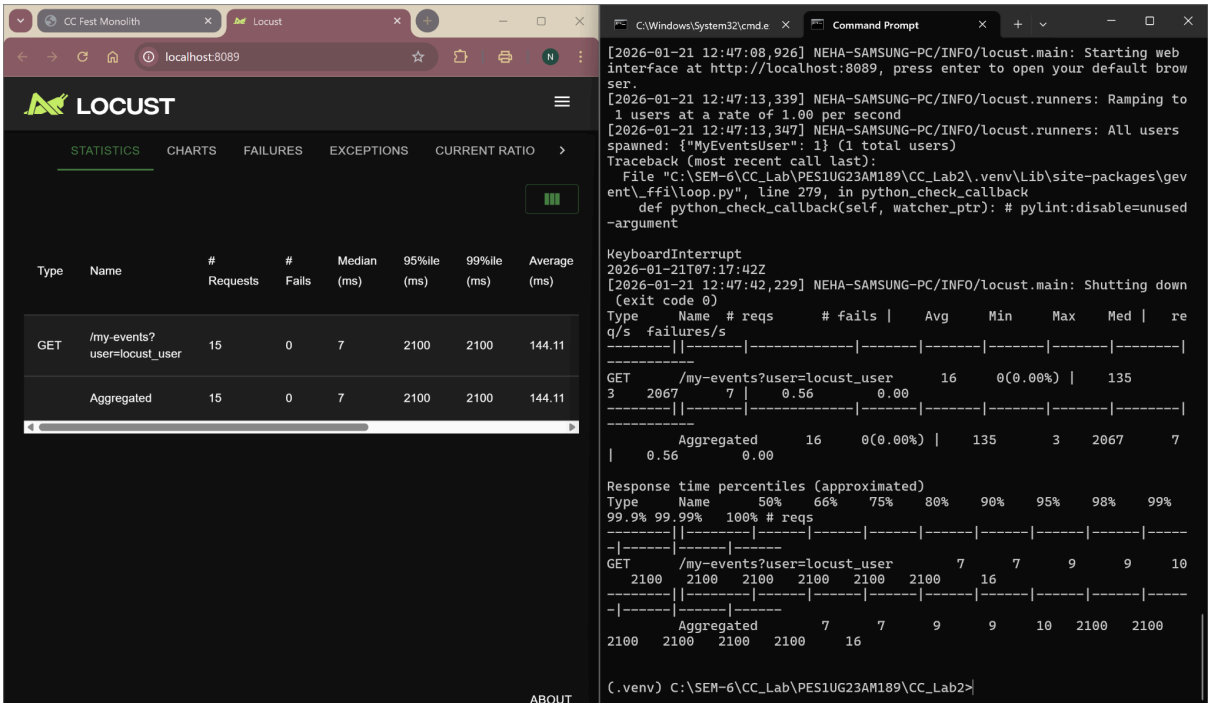


Route 2: /my-events

SS8 *(BEFORE optimization)*:



SS9 *(AFTER optimization)*:



Explanation of optimizations:

Route 1: /events

- **What was the bottleneck?**
  The bottleneck was unnecessary CPU-heavy computation inside the route (for i in range(3000000)), which wasted time on every request.
- **What change did you make?**
  Removed/reduced the unnecessary loop (dummy computation) so the route mainly performs the database query and returns the template.
- **Why did the performance improve?**
  Because the server stopped spending extra CPU time on useless calculations, the request finished faster and response time reduced.

Route 2: /my-events

- **What was the bottleneck?**
  The bottleneck was extra CPU work inside the route (for _ in range(1500000)), which slowed down every request even after fetching the required data.
- **What change did you make?**
  Removed/reduced the dummy loop and kept only the required database query + response rendering.
- **Why did the performance improve?**
  Because the endpoint no longer wastes CPU cycles, it processes requests quicker, so the average response time improved under Locust load.
