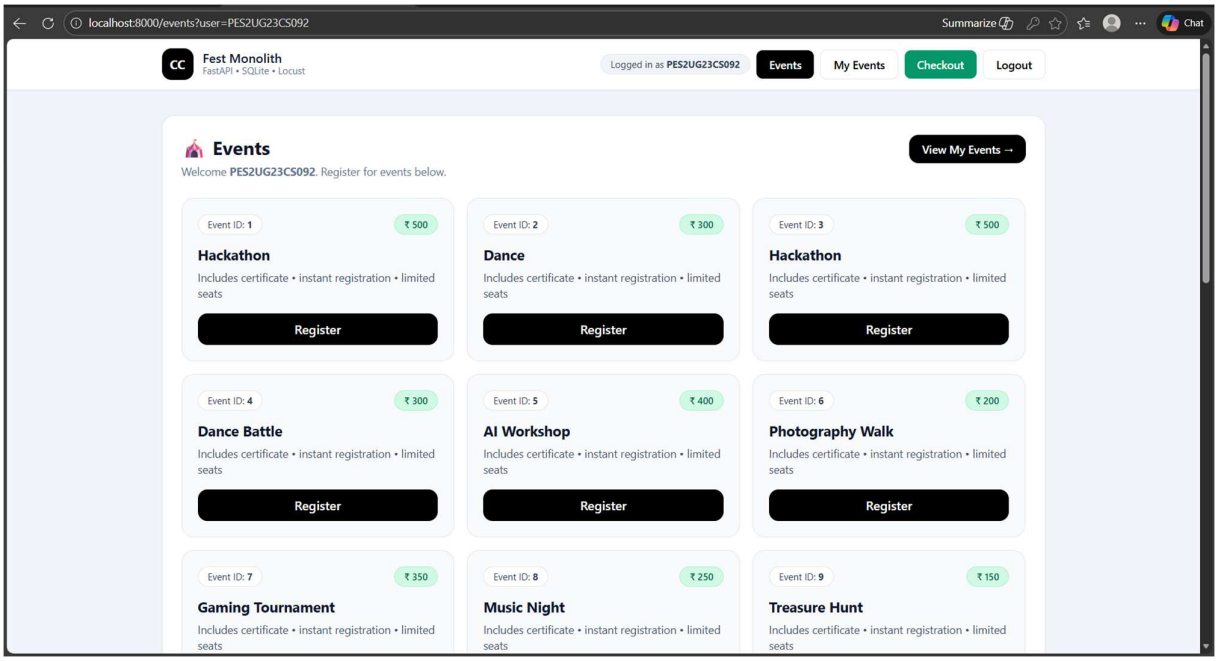
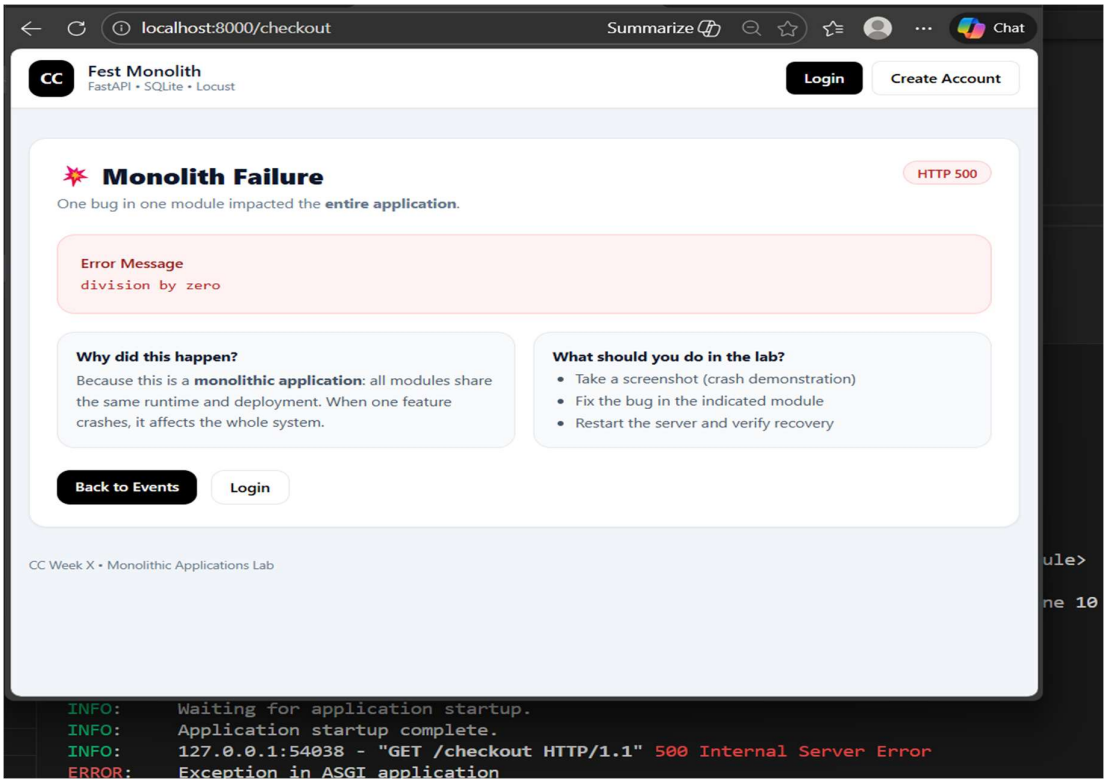
CC lab 2

Name: Arnav Sinha
SRN: PES2UG23CS092
Sec: B

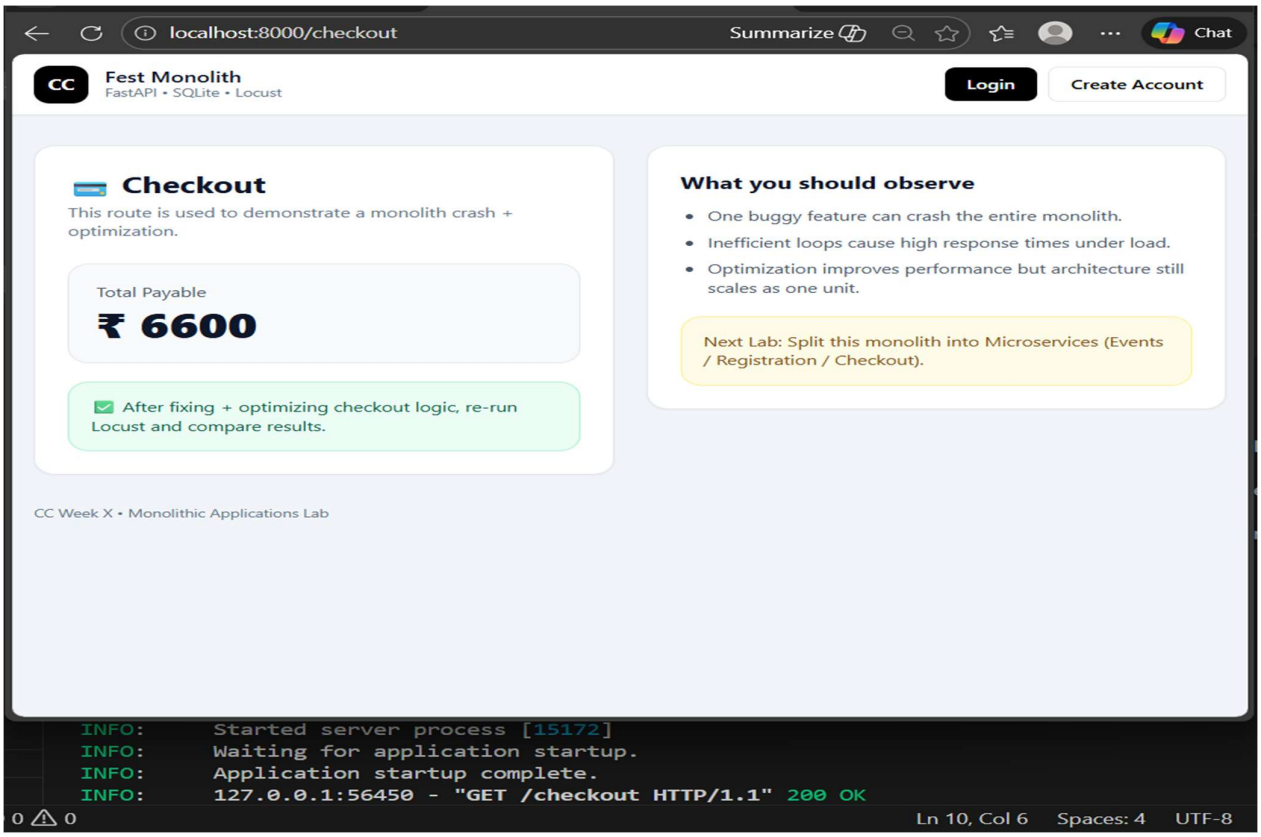
SS1



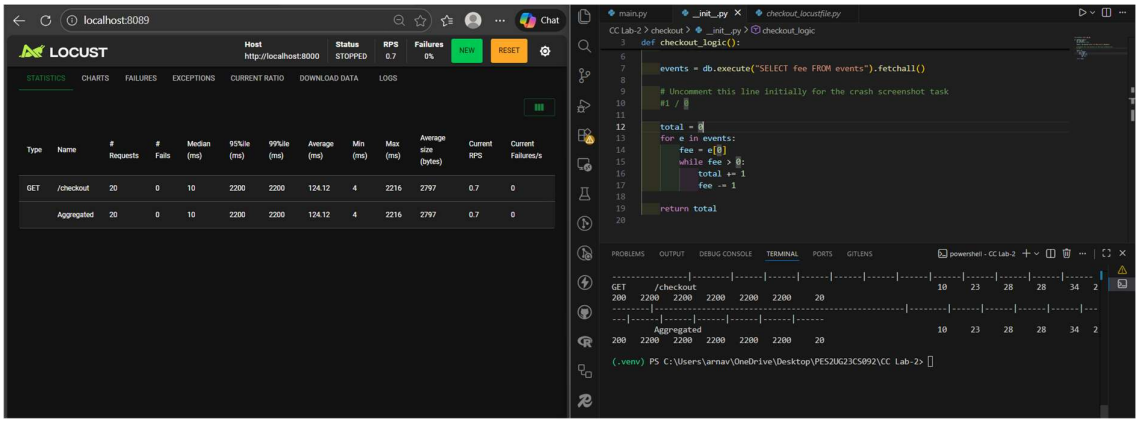
SS2


localhost:8000/checkout
Summarize
Chat
CC
Fest Monolith
FastAPI • SQLite • Locust
Login
Create Account
Monolith Failure
HTTP 500
One bug in one module impacted the entire application.
Error Message
division by zero
Why did this happen?
Because this is a monolithic application: all modules share the same runtime and deployment. When one feature crashes, it affects the whole system.
What should you do in the lab?
Take a screenshot (crash demonstration)
Fix the bug in the indicated module
Restart the server and verify recovery
Back to Events
Login
CC Week X • Monolithic Applications Lab
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: 127.0.0.1:54038 - "GET /checkout HTTP/1.1" 500 Internal Server Error
ERROR: Exception in ASGI application

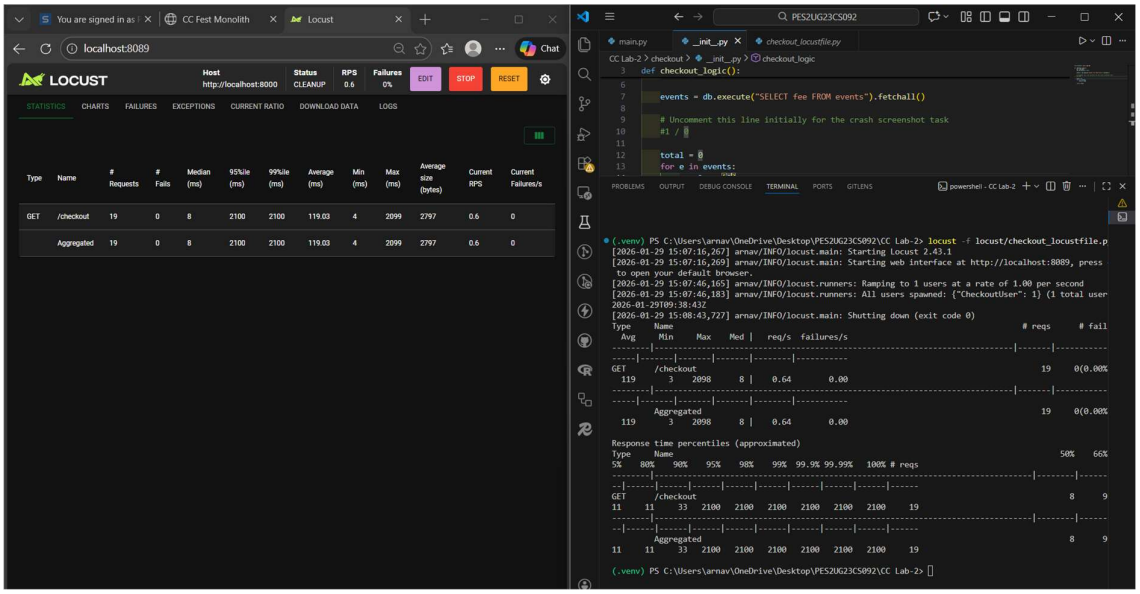
SS3


localhost:8000/checkout
Summarize
Chat
CC
Fest Monolith
FastAPI • SQLite • Locust
Login
Create Account
Checkout
This route is used to demonstrate a monolith crash + optimization.
Total Payable
₹ 6600
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: Started server process [15172]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: 127.0.0.1:56450 - "GET /checkout HTTP/1.1" 200 OK
0 0
Ln 10, Col 6
Spaces: 4
UTF-8

SS4



SS5



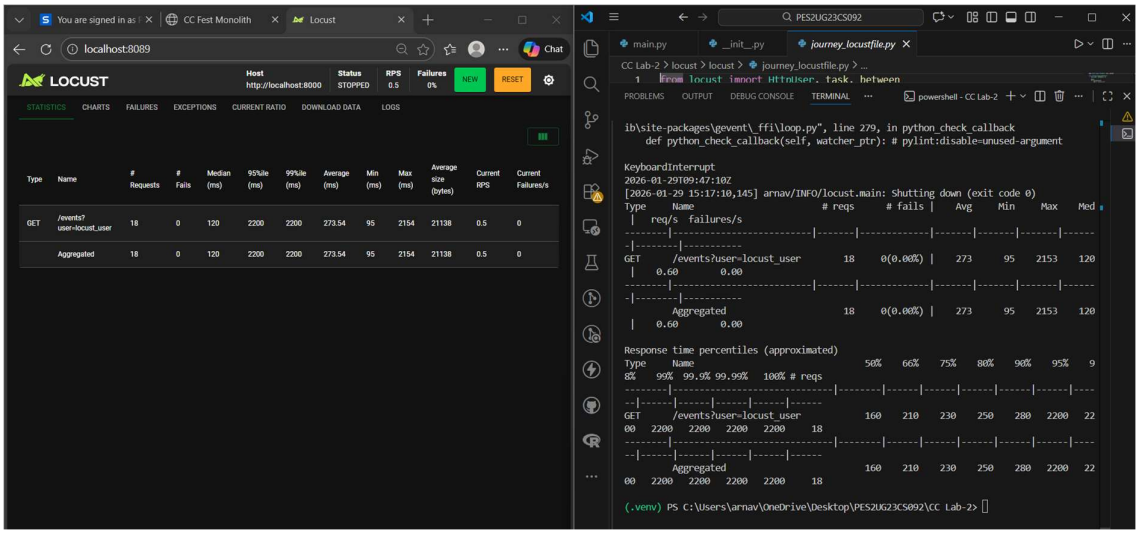
Before Optimization

- **Requests/sec (RPS): ~0.70**
- **Average Response Time: ~124 ms**

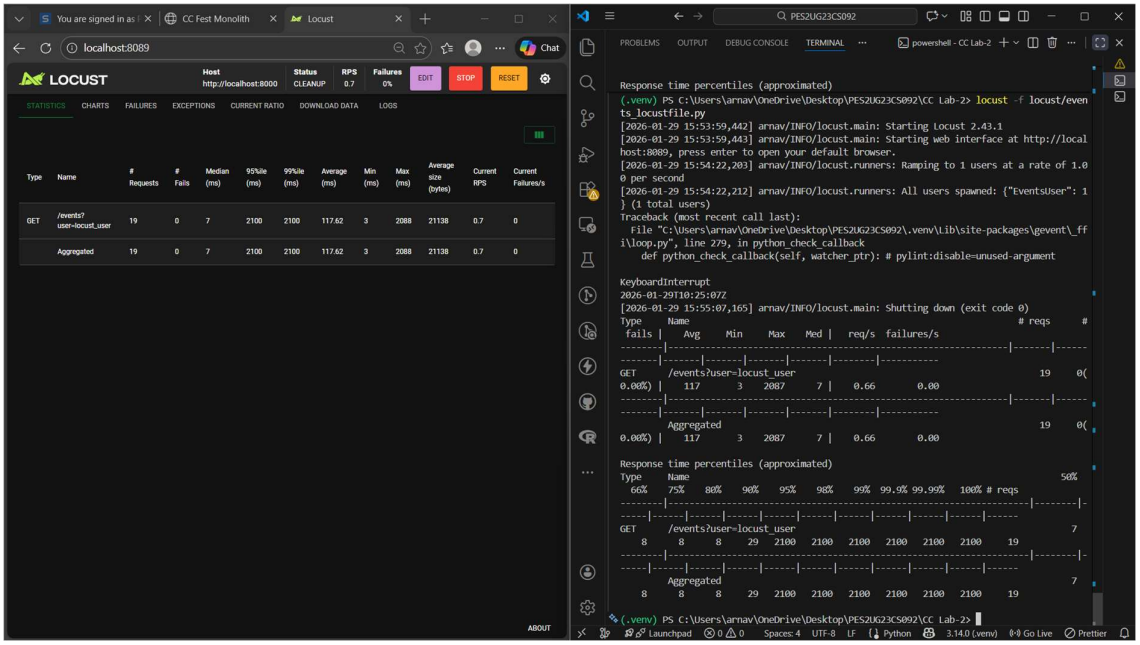
After Optimization

- **Requests/sec (RPS): ~0.64**
- **Average Response Time: ~119 ms**

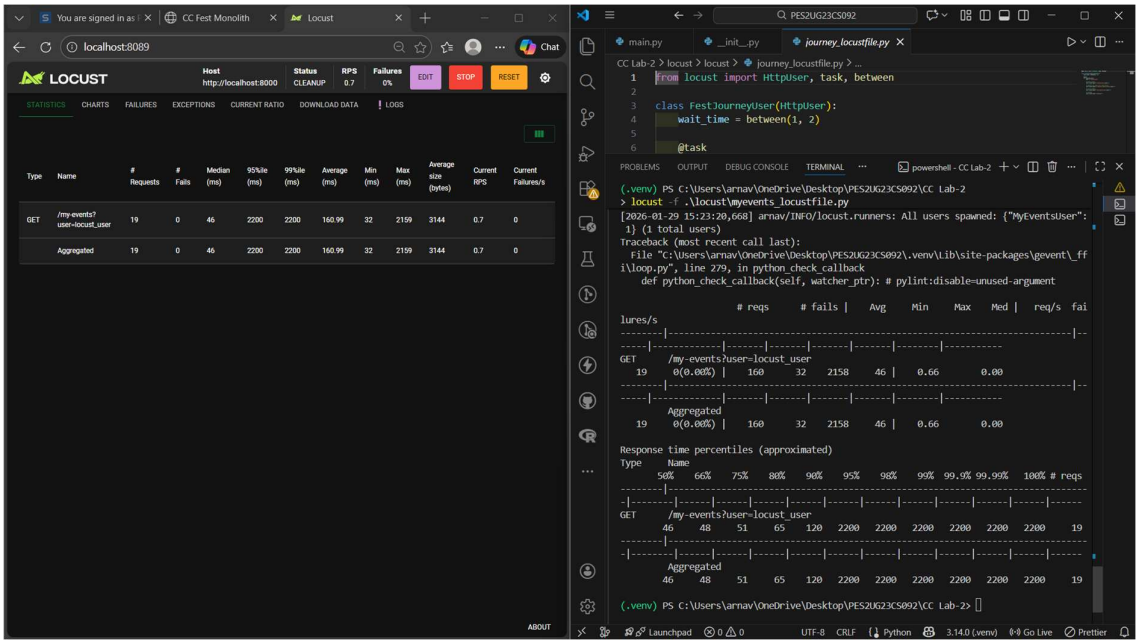
SS6


You are signed in as
CC Fest Monolith
Locust
localhost:8089
Chat
LOCUST
Host
http://localhost:8000
Status
STOPPED
RPS
0.5
Failures
0%
NEW
RESET
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
/events?user=locust_user
18
0
120
2200
2200
273.54
95
2154
21138
0.5
0
Aggregated
18
0
120
2200
2200
273.54
95
2154
21138
0.5
0
PES2UG23CS092
main.py
__init__.py
journey_locustfile.py
CC Lab-2 > locust > locust > journey_locustfile.py > ...
1 from locust import HttpUser, task, between
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
powershell - CC Lab-2
ib\site-packages\gevent_ffi\loop.py", line 279, in python_check_callback
def python_check_callback(self, watcher_ptr): # pylint:disable=unused-argument
KeyboardInterrupt
2026-01-29T09:47:10Z
[2026-01-29 15:17:10,145] arnav/INFO/locust.main: Shutting down (exit code 0)
Type Name # reqs # fails | Avg Min Max Med | req/s failures/s
GET /events?user=locust_user 18 0(0.00%) | 273 95 2153 120 | 0.60 0.00
Aggregated 18 0(0.00%) | 273 95 2153 120 | 0.60 0.00
Response time percentiles (approximated)
Type Name 50% 66% 75% 80% 90% 95% 99% 99.9% 99.99% 100% # reqs
GET /events?user=locust_user 160 210 230 250 280 2200 2200 2200 2200 2200 18
Aggregated 160 210 230 250 280 2200 2200 2200 2200 2200 18
(.venv) PS C:\Users\arnav\OneDrive\Desktop\PES2UG23CS092\CC Lab-2>

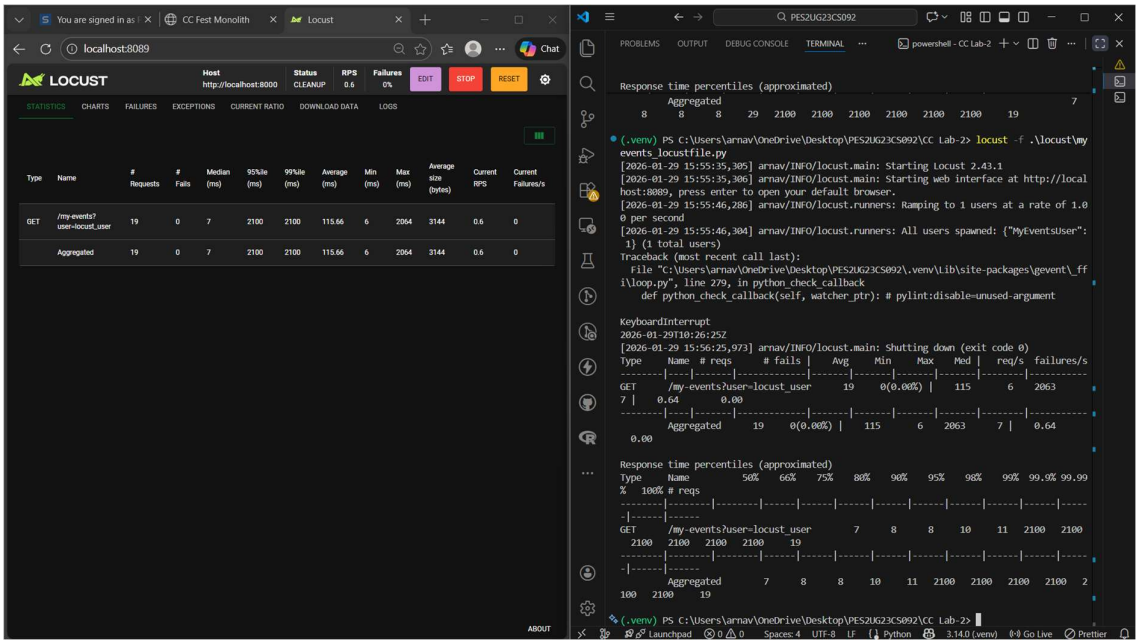
SS7


You are signed in as
CC Fest Monolith
Locust
localhost:8089
Chat
LOCUST
Host
http://localhost:8000
Status
CLEANUP
RPS
0.7
Failures
0%
EDIT
STOP
RESET
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
/events?user=locust_user
19
0
7
2100
2100
117.62
3
2088
21138
0.7
0
Aggregated
19
0
7
2100
2100
117.62
3
2088
21138
0.7
0
ABOUT
PES2UG23CS092
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
powershell - CC Lab-2
Response time percentiles (approximated)
(.venv) PS C:\Users\arnav\OneDrive\Desktop\PES2UG23CS092\CC Lab-2> locust -f locust/events_locustfile.py
[2026-01-29 15:53:59,442] arnav/INFO/locust.main: Starting Locust 2.43.1
[2026-01-29 15:53:59,443] arnav/INFO/locust.main: Starting web interface at http://localhost:8089, press enter to open your default browser.
[2026-01-29 15:54:22,203] arnav/INFO/locust.runners: Ramping to 1 users at a rate of 1.00 per second
[2026-01-29 15:54:22,212] arnav/INFO/locust.runners: All users spawned: {"EventsUser": 1} (1 total users)
Traceback (most recent call last):
File "C:\Users\arnav\OneDrive\Desktop\PES2UG23CS092\.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
2026-01-29T10:25:07Z
[2026-01-29 15:55:07,165] arnav/INFO/locust.main: Shutting down (exit code 0)
Type Name # reqs # fails | Avg Min Max Med | req/s failures/s
GET /events?user=locust_user 19 0(0.00%) | 117 3 2087 7 | 0.66 0.00
Aggregated 19 0(0.00%) | 117 3 2087 7 | 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 /events?user=locust_user 7 8 8 8 29 2100 2100 2100 2100 2100 2100 19
Aggregated 7 8 8 8 29 2100 2100 2100 2100 2100 2100 19
(.venv) PS C:\Users\arnav\OneDrive\Desktop\PES2UG23CS092\CC Lab-2>

SS8


You are signed in as
CC Fest Monolith
Locust
localhost:8089
Chat
LOCUST
Host
http://localhost:8000
Status
CLEANUP
RPS
0.7
Failures
0%
EDIT
STOP
RESET
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
/my-events?user=locust_user
19
0
46
2200
2200
160.99
32
2159
3144
0.7
0
Aggregated
19
0
46
2200
2200
160.99
32
2159
3144
0.7
0
ABOUT
PES2UG23CS092
main.py
__init__.py
journey_locustfile.py
CC Lab-2 > locust > locust > journey_locustfile.py > ...
from locust import HttpUser, task, between
class FestJourneyUser(HttpUser):
wait_time = between(1, 2)
@task
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
powershell - CC Lab-2
(.venv) PS C:\Users\arnav\OneDrive\Desktop\PES2UG23CS092\CC Lab-2
> locust -f .\locust\myevents_locustfile.py
[2026-01-29 15:23:20,668] arnav/INFO/locust.runners: All users spawned: {"MyEventsUser": 1} (1 total users)
Traceback (most recent call last):
File "C:\Users\arnav\OneDrive\Desktop\PES2UG23CS092\.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
reqs # fails | Avg Min Max Med | req/s failures/s
GET /my-events?user=locust_user
19 0(0.00%) | 160 32 2158 46 | 0.66 0.00
Aggregated
19 0(0.00%) | 160 32 2158 46 | 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 /my-events?user=locust_user 46 48 51 65 120 2200 2200 2200 2200 2200 2200 19
Aggregated 46 48 51 65 120 2200 2200 2200 2200 2200 2200 19
(.venv) PS C:\Users\arnav\OneDrive\Desktop\PES2UG23CS092\CC Lab-2>
Launchpad
UTF-8
CRLF
Python
3.14.0 (.venv)
Go Live
Prettier

SS9



/events

- Bottleneck: Server-side latency on /events (DB/cache/query work), not the Locust client. With a 1–2s wait you're issuing low RPS, so any high p95/p99 tails you see are the backend's.
- Change you have now: Simple GET with query string in locust/events_locustfile.py, using the default 1–2s wait. Commented the line in main.py

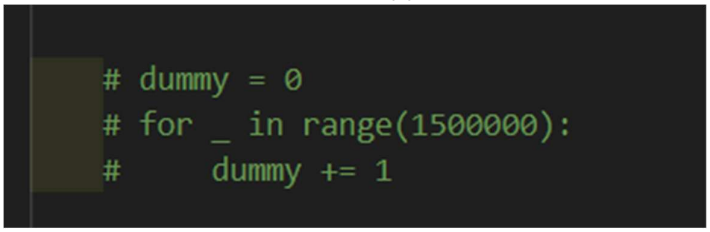
```
64
65      # waste = 0
66      # for i in range(3000000):
67      #     waste += i % 3
68
```

- Why performance improves (when it does): Stability comes from low load (long waits), which reduces contention on the backend and hides bottlenecks; it's not a client-side optimization. If tails remain high, focus on backend profiling or caching.

/my-events

- Bottleneck: Same story—backend work for /my-events (auth/session lookups, DB reads).

- Change you have now: Simple GET with query string in locust/myevents_locustfile.py, 1–2s wait. Commented the line in main.py



- Why performance improves (when it does): Lower request rate eases server pressure, so averages/tails can look better. Any remaining slowness points to the backend; you'd need shorter waits or more users to surface true capacity limits, then optimize the service (indexes, caching, N+1 fixes).

Repo: https://github.com/arnavsinha20/CC-lab2

Optimization:

- Short waits mean higher RPS so you can see how the backend behaves under load instead of idling between calls.
- Named requests and passing query params separately keep Locust stats clean and comparable.
- Timeouts and catch_response surface bad statuses quickly instead of masking errors.