

# CLOUD COMPUTING

## LAB 2

NAME: PRANITH K  
SRN: PES2UG23CS434  
SECTION: G

SS1:

CC

Fest Monolith  
FastAPI • SQLite • Locust


Logged in as PES2UG23CS434

Events

My Events

Checkout

Logout

 **Events**

Welcome PES2UG23CS434. Register for events below.

View My Events →

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

₹ 300

**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

₹ 350

**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

Event ID: 10

₹ 300

**Stand-up Comedy**  
Includes certificate • instant registration • limited seats

Register

Event ID: 11

₹ 450

**Robo Race**  
Includes certificate • instant registration • limited seats

Register

Event ID: 12

₹ 500

**Hackathon**  
Includes certificate • instant registration • limited seats

Register

## SS2:

### 🌟 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](#)

```
ZeroDivisionError: division by zero
INFO:      127.0.0.1:62763 - "GET /checkout HTTP/1.1" 500 Internal Server Error
or
ERROR:      Exception in ASGI application
```

## SS3:

### 🛒 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).

# SS4: PRE OPTIMIZATION

LOCUST

Host  
http://localhost:8000

Status  
STOPPED

RPS  
0.6

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  | /checkout  | 20         | 0       | 15          | 2100        | 2100        | 120.59       | 13       | 2114     | 2798                 | 0.6         | 0                  |
|      | Aggregated | 20         | 0       | 15          | 2100        | 2100        | 120.59       | 13       | 2114     | 2798                 | 0.6         | 0                  |

[2026-01-29 14:54:36,366] Pran-Vivobook/INFO/locust.main: Shutting down (exit code 0)

| Type | Name       | # reqs | # fails  | Avg | Min | Max  | Med | req/s | failures/s |
|------|------------|--------|----------|-----|-----|------|-----|-------|------------|
| GET  | /checkout  | 20     | 0(0.00%) | 120 | 12  | 2114 | 15  | 0.67  | 0.00       |
|      | Aggregated | 20     | 0(0.00%) | 120 | 12  | 2114 | 15  | 0.67  | 0.00       |

| Type | Name       | # reqs | # fails | Avg | Min | Max | Med  | req/s | failures/s |
|------|------------|--------|---------|-----|-----|-----|------|-------|------------|
| GET  | /checkout  | 15     | 18      | 19  | 19  | 20  | 2100 | 2100  | 2100       |
|      | Aggregated | 15     | 18      | 19  | 19  | 20  | 2100 | 2100  | 2100       |

(.venv) C:\Users\Prany\OneDrive\Desktop\PES2UG23CS434\CC\_LAB2>

# SS5: POST OPTIMIZATION - MINOR OPTIMIZATION OBSERVED

LOCUST

Host  
http://localhost:8000

Status  
STOPPED

RPS  
0.6

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  | /checkout  | 18         | 0       | 15          | 2000        | 2000        | 129.16       | 13       | 2049     | 2797                 | 0.6         | 0                  |
|      | Aggregated | 18         | 0       | 15          | 2000        | 2000        | 129.16       | 13       | 2049     | 2797                 | 0.6         | 0                  |

File Edit Selection View Go Run ...

CC\_Lab-2

\_init\_.py

checkout > \_init\_.py > checkout\_logic

1 from database import get\_db

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

| Type | Name       | # reqs | # fails  | Avg | Min | Max  | Med | req/s | failures/s |
|------|------------|--------|----------|-----|-----|------|-----|-------|------------|
| GET  | /checkout  | 18     | 0(0.00%) | 129 | 12  | 2049 | 15  | 0.63  | 0.00       |
|      | Aggregated | 18     | 0(0.00%) | 129 | 12  | 2049 | 15  | 0.63  | 0.00       |

Response time percentiles (approximated)

| Type | Name       | 50% | 66% | 75% | 80% | 90% | 95%  | 98%  | 99%  | 99.9% | 99.99% | 100% | # reqs |
|------|------------|-----|-----|-----|-----|-----|------|------|------|-------|--------|------|--------|
| GET  | /checkout  | 16  | 18  | 19  | 19  | 20  | 2000 | 2000 | 2000 | 2000  | 2000   | 2000 | 18     |
|      | Aggregated | 16  | 18  | 19  | 19  | 20  | 2000 | 2000 | 2000 | 2000  | 2000   | 2000 | 18     |

(.venv) C:\Users\Prany\OneDrive\Desktop\PES2UG23CS434\CC\_Lab-2>

## SS6: EVENT OPTIMIZATION

The screenshot shows the Locust web interface at the top, indicating the host is `http://localhost:8000`, status is `STOPPED`, RPS is `0.6`, and failures are `0%`. Below the interface, the VS Code terminal displays the output of the `__init__.py` script. The terminal output shows a table of response time percentiles for the `GET /events?user=locust_user` endpoint. The table includes columns for Type, Name, # Requests, # Fails, Median (ms), 95%ile (ms), 99%ile (ms), Average (ms), Min (ms), Max (ms), Average size (bytes), Current RPS, and Current Failures/s. The data shows that the endpoint is performing well with a median response time of 280ms and a 99th percentile of 2500ms.

| 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 | 16         | 0       | 280         | 2500        | 2500        | 419.97       | 227      | 2480     | 21138                | 0.6         | 0                  |
| Aggregated |                          | 16         | 0       | 280         | 2500        | 2500        | 419.97       | 227      | 2480     | 21138                | 0.6         | 0                  |

## SS7: OPTIMIZED EVENTS

The screenshot shows the Locust web interface at the top, indicating the host is `http://localhost:8000`, status is `STOPPED`, RPS is `0.6`, and failures are `0%`. Below the interface, the VS Code terminal displays the output of the `main.py` script. The terminal output shows a table of response time percentiles for the `GET /events?user=locust_user` endpoint. The table includes columns for Type, Name, # Requests, # Fails, Median (ms), 95%ile (ms), 99%ile (ms), Average (ms), Min (ms), Max (ms), Average size (bytes), Current RPS, and Current Failures/s. The data shows that the endpoint is performing well with a median response time of 19ms and a 99th percentile of 2100ms.

| 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       | 19          | 2100        | 2100        | 126.34       | 14       | 2090     | 21138                | 0.6         | 0                  |
| Aggregated |                          | 19         | 0       | 19          | 2100        | 2100        | 126.34       | 14       | 2090     | 21138                | 0.6         | 0                  |

## SS8: PRE OPTIMIZATION

The screenshot shows a web browser displaying a statistics table and a terminal window.

**Statistics Table:**

| Type | Name                        | # Requests | # Fails | Median (ms) | 95kile (ms) | 99kile (ms) | Average (ms) | Min (ms) | Max (ms) | Average size (bytes) | Current RPS | Current Failures/s |
|------|-----------------------------|------------|---------|-------------|-------------|-------------|--------------|----------|----------|----------------------|-------------|--------------------|
| GET  | /my-events?user=locust_user | 17         | 0       | 100         | 2200        | 2200        | 228.47       | 96       | 2177     | 3144                 | 0.6         | 0                  |
|      | Aggregated                  | 17         | 0       | 100         | 2200        | 2200        | 228.47       | 96       | 2177     | 3144                 | 0.6         | 0                  |

**Terminal Window:**

```

main.py > events
58

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

GET      /my-events?user=locust_user    17    0(0.00%) | 228    96    2176    100 | 0.59    0.00
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
Aggregated    17    0(0.00%) | 228    96    2176    100 | 0.59    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    100    110    110    120    160    2200    2200    2200    2200    2200    17
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
Aggregated    100    110    110    120    160    2200    2200    2200    2200    2200    17
  
```

## SS9: POST OPTIMIZATION

| 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       | 17          | 2100        | 2100        | 125.83       | 14       | 2089     | 3144                 | 0.7         | 0                  |
|      | Aggregated                  | 19         | 0       | 17          | 2100        | 2100        | 125.83       | 14       | 2089     | 3144                 | 0.7         | 0                  |

```

2026-01-29T10:10:39Z
[2026-01-29 15:40:39,399] Pran-Vivobook/INFO/locust.main: Shutting down (exit code 0)
Type      Name                                     # reqs      # fails | Avg      Min      Max      Med | req/s  failures/s
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
GET      /my-events?user=locust_user                19          0(0.00%) | 125      13      2089     17 | 0.64    0.00
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
Aggregated                                  19          0(0.00%) | 125      13      2089     17 | 0.64    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                17      18      19      19      21      2100    2100    2100    2100    210
0 2100 19
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
Aggregated                                  17      18      19      19      21      2100    2100    2100    2100    210
0 2100 19

(.venv) C:\Users\Prany\OneDrive\Desktop\PES2UG23CS434\CC_Lab-2>
(.venv) C:\Users\Prany\OneDrive\Desktop\PES2UG23CS434\CC_Lab-2>

```

## **QNA:**

### **Short Explanation for Both Routes**

#### **Route: /events**

##### **What was the bottleneck?**

The events route had extra processing logic that was executed on every request, which increased the response time.

##### **What change did you make?**

I removed the unnecessary processing while keeping the database access and page output the same.

##### **Why did the performance improve?**

By reducing the amount of work done per request, the server was able to handle the request more efficiently.

#### **Route: /my-events**

##### **What was the bottleneck?**

The route included redundant operations that caused additional CPU usage during each request.

##### **What change did you make?**

I removed the redundant code without changing the functionality of the route.

##### **Why did the performance improve?**

Since fewer computations were performed, the response time improved during load testing.