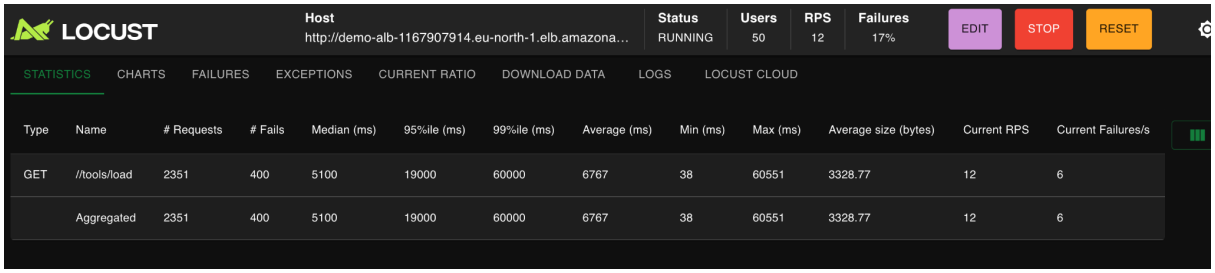
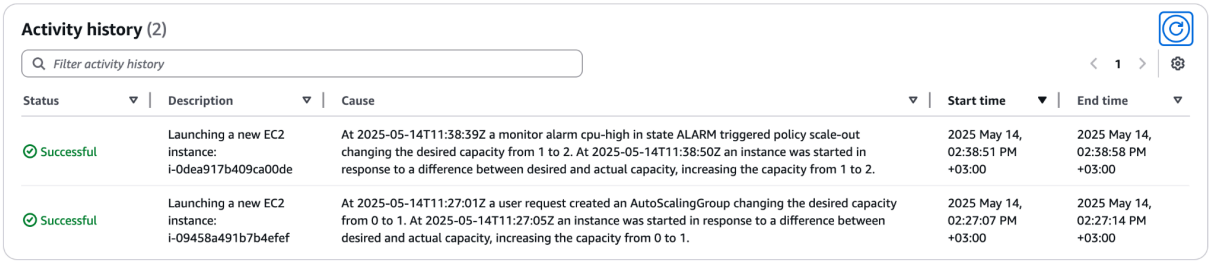
ALB+ASG+EC2

Started from 1 instance with 50 users and 5 users started/per second, min amount of user

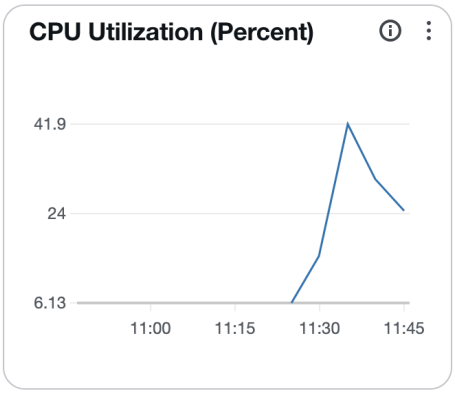


Type	Name	# Requests	# Fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	//tools/load	2351	400	5100	19000	60000	6767	38	60551	3328.77	12	6
	Aggregated	2351	400	5100	19000	60000	6767	38	60551	3328.77	12	6

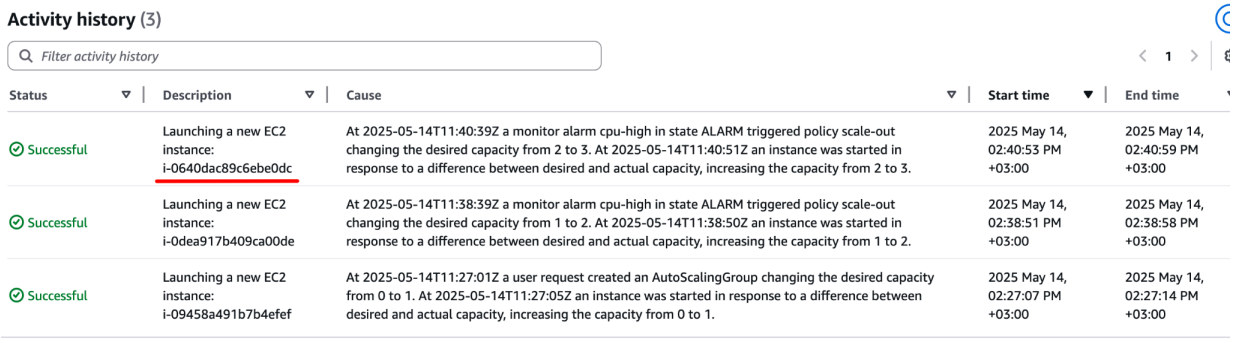
Here you can see loading test(there was acreage 8000+ ms) after adding one more instance to target group



Status	Description	Cause	Start time	End time
Successful	Launching a new EC2 instance: i-0dea917b409ca00de	At 2025-05-14T11:38:39Z a monitor alarm cpu-high in state ALARM triggered policy scale-out changing the desired capacity from 1 to 2. At 2025-05-14T11:38:50Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 2.	2025 May 14, 02:38:51 PM +03:00	2025 May 14, 02:38:58 PM +03:00
Successful	Launching a new EC2 instance: i-09458a491b7b4efef	At 2025-05-14T11:27:01Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 1. At 2025-05-14T11:27:05Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 1.	2025 May 14, 02:27:07 PM +03:00	2025 May 14, 02:27:14 PM +03:00

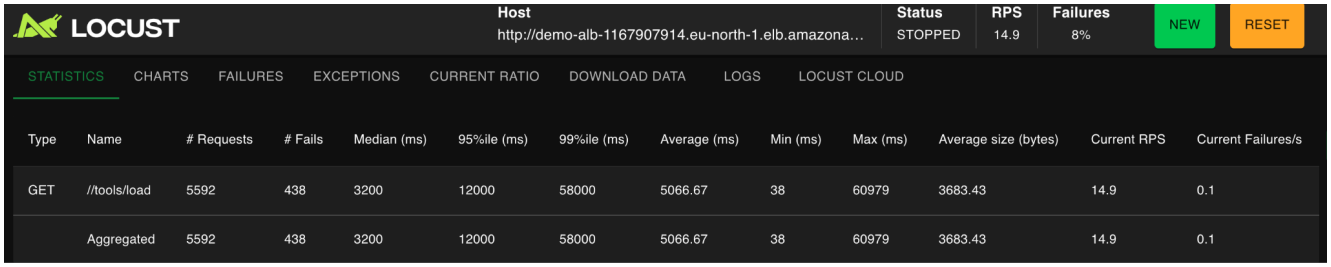


It wasn't enough for defined maximum cpu load (30%) so cloudwatch alarms triggered asg to create one more instance



Status	Description	Cause	Start time	End time
Successful	Launching a new EC2 instance: i-0640dac89c6ebe0dc	At 2025-05-14T11:40:39Z a monitor alarm cpu-high in state ALARM triggered policy scale-out changing the desired capacity from 2 to 3. At 2025-05-14T11:40:51Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 2 to 3.	2025 May 14, 02:40:53 PM +03:00	2025 May 14, 02:40:59 PM +03:00
Successful	Launching a new EC2 instance: i-0dea917b409ca00de	At 2025-05-14T11:38:39Z a monitor alarm cpu-high in state ALARM triggered policy scale-out changing the desired capacity from 1 to 2. At 2025-05-14T11:38:50Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 2.	2025 May 14, 02:38:51 PM +03:00	2025 May 14, 02:38:58 PM +03:00
Successful	Launching a new EC2 instance: i-09458a491b7b4efef	At 2025-05-14T11:27:01Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 1. At 2025-05-14T11:27:05Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 1.	2025 May 14, 02:27:07 PM +03:00	2025 May 14, 02:27:14 PM +03:00

In context of 3 instances there was such load



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	//tools/load	5592	438	3200	12000	58000	5066.67	38	60979	3683.43	14.9	0.1
	Aggregated	5592	438	3200	12000	58000	5066.67	38	60979	3683.43	14.9	0.1

Deployment Method Comparison: EC2 Auto Scaling Group vs. ECS Fargate

EC2 + Auto Scaling Group (Implemented)

Complexity of Setup:
Requires manual setup of launch templates, user data scripts, ALB, target groups, scaling policies, and CloudWatch alarms. More control, but more complexity.

Cost:
Fixed costs due to running EC2 instances regardless of traffic(can be used with different pay strategies). ALB adds cost based on active hours and request volume.

Scalability:
Supports horizontal scaling via Auto Scaling policies. Reaction time depends on CloudWatch alarm thresholds and cooldown settings. More customizable but slower to scale under high load.

ECS Fargate

Complexity of Setup:
Simpler setup using task definitions and service configurations. No server management required. Native integration with load balancers and auto scaling.

Cost:
Pay-per-use pricing based on CPU and memory used per task. More cost-efficient for applications with fluctuating or intermittent traffic.

Scalability:
Faster scaling. Tasks launch on demand without delays.