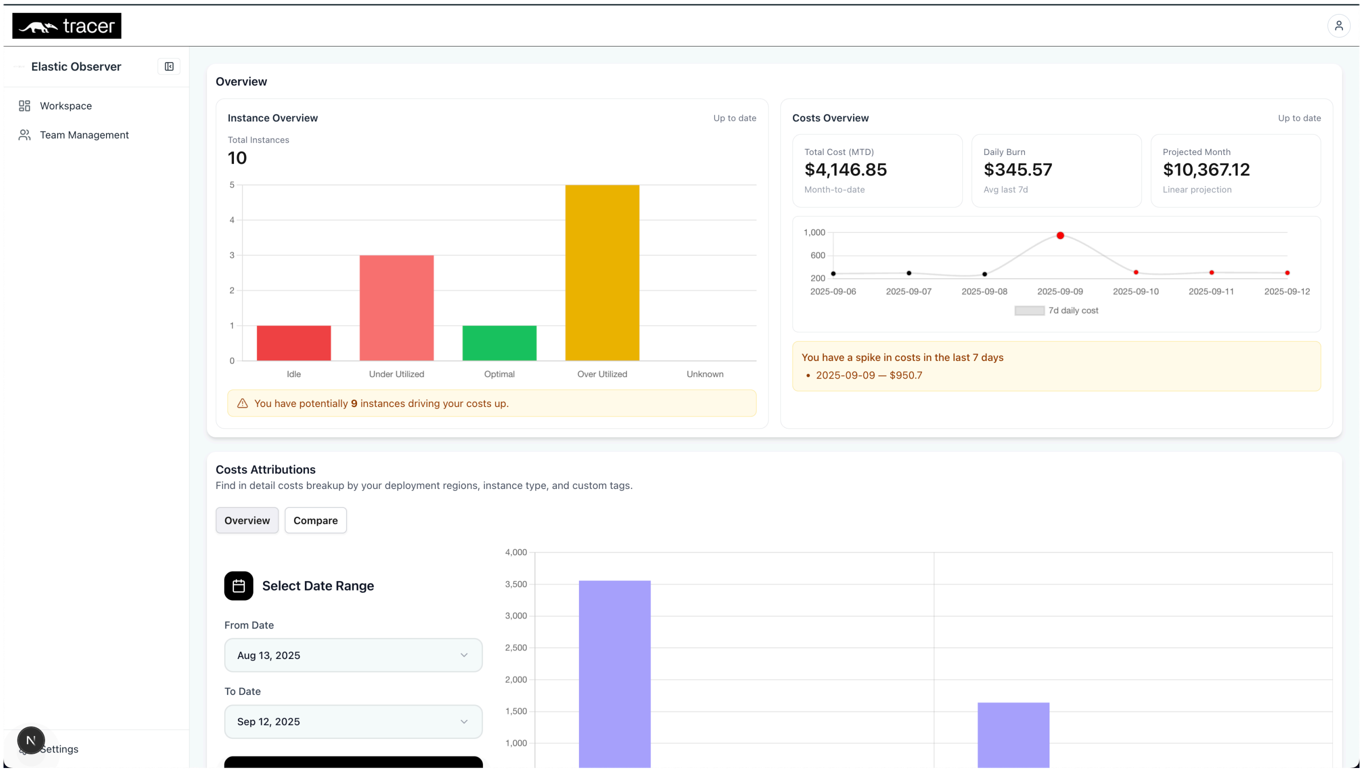
# Elastic Observer - Technical Guide

## Introduction :

This document provides a technical overview of Elastic Oberver, an EC2 Observability Prototype, built with Next.js, typescript.



Total Instances
10

You have potentially 9 instances driving your costs up.

Total Cost (MTD)
$4,146.85
Month-to-date

Daily Burn
$345.57
Avg last 7d

Projected Month
$10,367.12
Linear projection

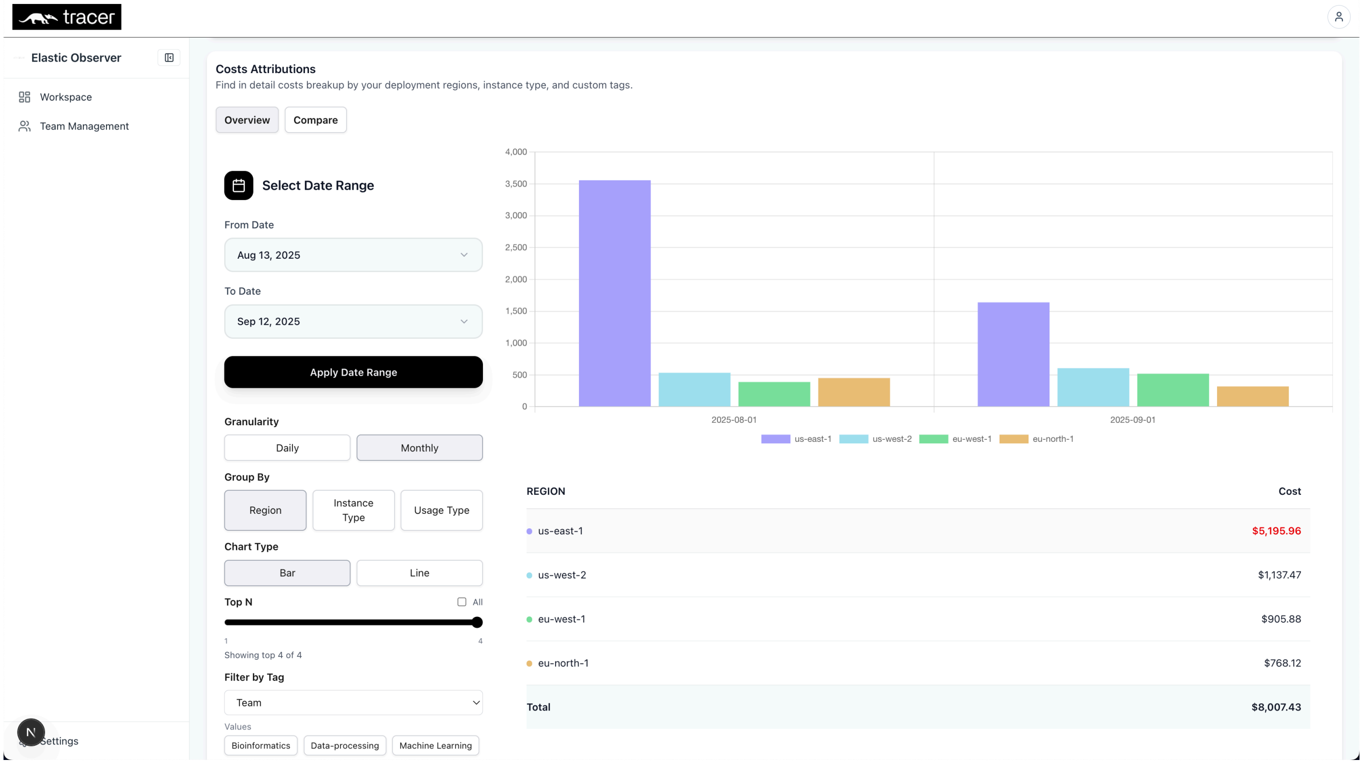
You have a spike in costs in the last 7 days

- 2025-09-09 — $950.7

Find in detail costs breakup by your deployment regions, instance type, and custom tags.

From Date
Aug 13, 2025

To Date
Sep 12, 2025



**Costs Attributions**

Find in detail costs breakup by your deployment regions, instance type, and custom tags.

Overview | Compare

**Select Date Range**

From Date: Aug 13, 2025

To Date: Sep 12, 2025

Apply Date Range

Granularity: Daily | Monthly

Group By: Region | Instance Type | Usage Type

Chart Type: Bar | Line

Top N ☐ All

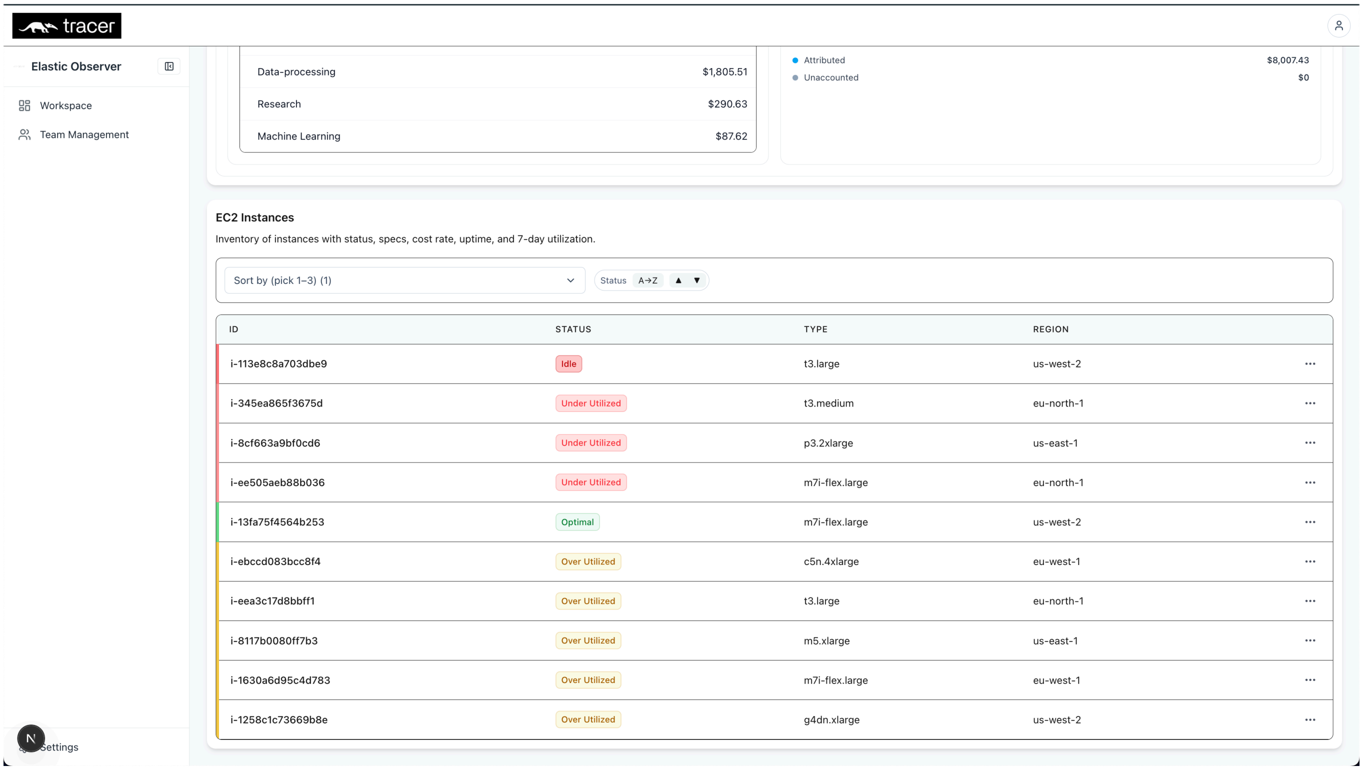
1 4

Showing top 4 of 4

Filter by Tag: Team

Values: Bioinformatics | Data-processing | Machine Learning

| REGION | Cost |
|---|---|
| ● us-east-1 | $5,195.96 |
| ● us-west-2 | $1,137.47 |
| ● eu-west-1 | $905.88 |
| ● eu-north-1 | $768.12 |
| Total | $8,007.43 |



| | |
|---|---|
| Data-processing | $1,805.51 |
| Research | $290.63 |
| Machine Learning | $87.62 |

| | |
|---|---|
| ● Attributed | $8,007.43 |
| ● Unaccounted | $0 |

**EC2 Instances**

Inventory of instances with status, specs, cost rate, uptime, and 7-day utilization.

Sort by (pick 1–3) (1) | Status A→Z ▲ ▼

| ID | STATUS | TYPE | REGION | |
|---|---|---|---|---|
| i-113e8c8a703dbe9 | Idle | t3.large | us-west-2 | ··· |
| i-345ea865f3675d | Under Utilized | t3.medium | eu-north-1 | ··· |
| i-8cf663a9bf0cd6 | Under Utilized | p3.2xlarge | us-east-1 | ··· |
| i-ee505aeb88b036 | Under Utilized | m7i-flex.large | eu-north-1 | ··· |
| i-13fa75f4564b253 | Optimal | m7i-flex.large | us-west-2 | ··· |
| i-ebccd083bcc8f4 | Over Utilized | c5n.4xlarge | eu-west-1 | ··· |
| i-eea3c17d8bbff1 | Over Utilized | t3.large | eu-north-1 | ··· |
| i-8117b0080ff7b3 | Over Utilized | m5.xlarge | us-east-1 | ··· |
| i-1630a6d95c4d783 | Over Utilized | m7i-flex.large | eu-west-1 | ··· |
| i-1258c1c73669b8e | Over Utilized | g4dn.xlarge | us-west-2 | ··· |

# Settings :

An AWS account with programmatic access (API Key and Secret)

Configured AWS IAM role with read-only permissions for EC2, CloudWatch, and Cost Explorer.

Configure Environment Variables:
Create a .env.local file in the root of the project and add your AWS credentials.

```
AWS_ACCESS_KEY_ID=your_access_key
AWS_SECRET_ACCESS_KEY=your_secret_key
AWS_REGION=your_default_region
```

The above lets you run the code with AWS access keys, however the code also runs with a configured mock API for mock data, just make all the below keys as 1 for them to work in your .env.local file

```
NEXT_PUBLIC_USE_MOCK=1
MOCK_COST=1
MOCK_METRICS=1
NEXT_PUBLIC_MOCK_OVERVIEW=1
```

# REPOSITORY

**Repository Structure**
The main structure for our use case is built across

```
src/
├── app/
│ ├── api/ # AWS & Mock API routes
│ └── screens/ # Main pages: Overview, Cost Attributions, EC2 Table
├── components/ # Reusable UI components
├── context/ # Global state: CostContext, EC2Context, Filters
└── utils/ # Helper functions (date utils, formatting)
```

# DATA

**Note on mock data** :
To design a decent product system, a big part of this task was to understand how AWS handled cost explorer API and other ec2 instance realted API's. For this a solid set of data was really important. Rather than setup mock-data files, I setup a mock database system which mostly mimicked the AWS API barring some changes.

## How Mock Data Was Generated

The mock cost data is **deterministic** and designed to look realistic while mirroring AWS Cost Explorer's shape.

- **Instances** are loaded from `synthetic-ec2-data.json` with IDs, types, regions, tags, and launch times.
- **Profiles** ( `Idle` , `Under Utilized` , `Optimal` , `Over Utilized` ) are inferred from instance family, tags, and a seeded random jitter.

- **CPU, memory, uptime, and infra signals** (EBS type/size, NAT, ALB, Elastic IP) are generated per instance to simulate real usage.
- **Daily/Monthly costs** are computed by iterating periods, capping hours by profile uptime, multiplying by hourly prices, and adding storage, network, and infra charges.
- **Anomalies**: 15% of days are “spike days,” with compute cost multiplied 3×–7× to create realistic cost spikes.
- **Filters and grouping** follow AWS Cost Explorer semantics for region, instance type, tags, and usage type.
- All randomness is **seeded**, ensuring the same inputs always produce the same results.

# Core Components

## EC2 Instance Utilisation Table

This table provides a detailed, at-a-glance view of all running EC2 instances.

**Features:**

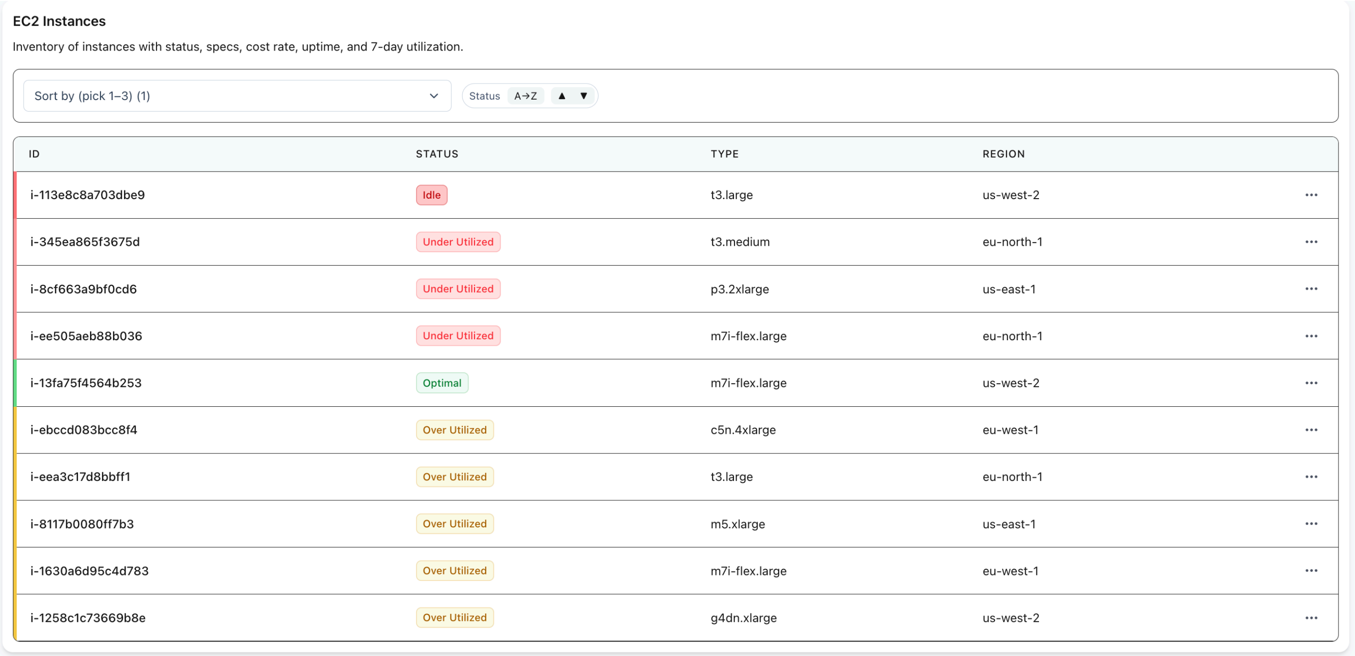
- **Columns:** ID, Wastage Status, Instance Type, Region
- **Sub-metrics:** CPU, RAM, GPU, Uptime, Memory, Cost/hour
- **Visual cues:** Color-coded bars to indicate Idle, Under-utilized, Optimal, Over-utilized
- **Sorting & Filtering:** By region, instance type, and utilisation status
- **Waste Indicator:** Flags instances with consistently low CPU usage over long uptimes

**Creative Element:**
I implemented a **utilisation score** using a simple heuristic that combines CPU usage, memory usage, and uptime coverage. This approach allows bioinformaticians to quickly spot cost-driving idle or oversized instances without performing manual calculations.

I initially considered applying an **unsupervised algorithm** such as k-means clustering to automatically group instances by utilization patterns. However, I decided against it, as clustering can introduce complexity and make the results less transparent without a strong, well-communicated rationale for how clusters are defined.

By using a heuristic-based scoring system, the classification should remain **interpretable and actionable**, helping users make confident infrastructure decisions with minimal cognitive overhead.

# INSTANCES TABLE VIEW

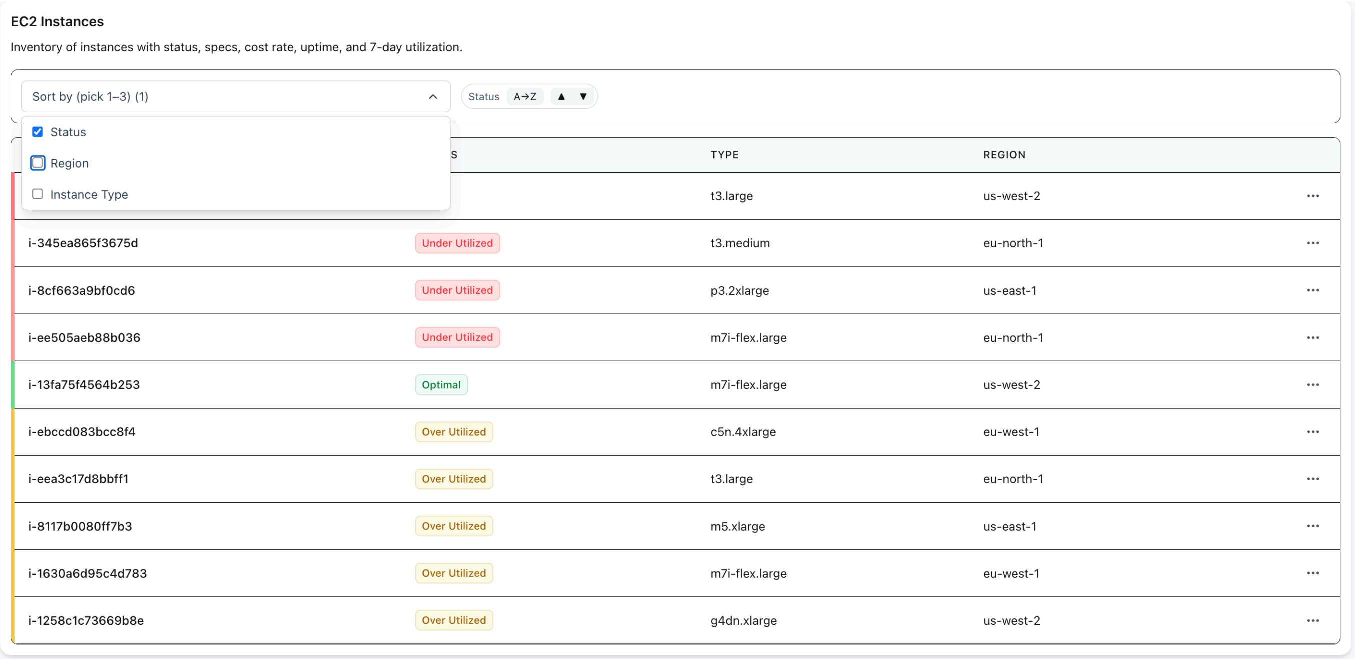


EC2 Instances

Inventory of instances with status, specs, cost rate, uptime, and 7-day utilization.

| ID | STATUS | TYPE | REGION | |
|---|---|---|---|---|
| i-113e8c8a703dbe9 | Idle | t3.large | us-west-2 | ··· |
| i-345ea865f3675d | Under Utilized | t3.medium | eu-north-1 | ··· |
| i-8cf663a9bf0cd6 | Under Utilized | p3.2xlarge | us-east-1 | ··· |
| i-ee505aeb88b036 | Under Utilized | m7i-flex.large | eu-north-1 | ··· |
| i-13fa75f4564b253 | Optimal | m7i-flex.large | us-west-2 | ··· |
| i-ebccd083bcc8f4 | Over Utilized | c5n.4xlarge | eu-west-1 | ··· |
| i-eea3c17d8bbff1 | Over Utilized | t3.large | eu-north-1 | ··· |
| i-8117b0080ff7b3 | Over Utilized | m5.xlarge | us-east-1 | ··· |
| i-1630a6d95c4d783 | Over Utilized | m7i-flex.large | eu-west-1 | ··· |
| i-1258c1c73669b8e | Over Utilized | g4dn.xlarge | us-west-2 | ··· |

# INSTANCES FILTER



EC2 Instances

Inventory of instances with status, specs, cost rate, uptime, and 7-day utilization.

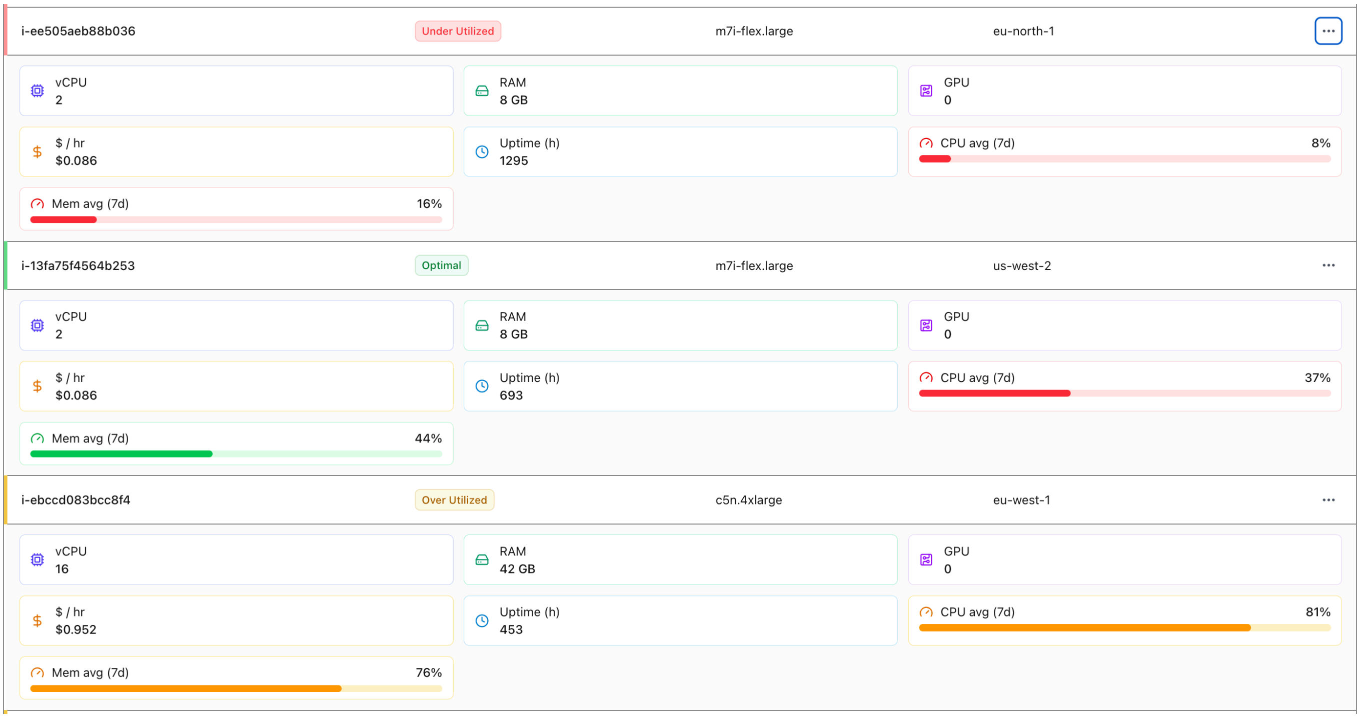
☑ Status

☐ Region

☐ Instance Type

| | S | TYPE | REGION | |
|---|---|---|---|---|
| | | t3.large | us-west-2 | ··· |
| i-345ea865f3675d | Under Utilized | t3.medium | eu-north-1 | ··· |
| i-8cf663a9bf0cd6 | Under Utilized | p3.2xlarge | us-east-1 | ··· |
| i-ee505aeb88b036 | Under Utilized | m7i-flex.large | eu-north-1 | ··· |
| i-13fa75f4564b253 | Optimal | m7i-flex.large | us-west-2 | ··· |
| i-ebccd083bcc8f4 | Over Utilized | c5n.4xlarge | eu-west-1 | ··· |
| i-eea3c17d8bbff1 | Over Utilized | t3.large | eu-north-1 | ··· |
| i-8117b0080ff7b3 | Over Utilized | m5.xlarge | us-east-1 | ··· |
| i-1630a6d95c4d783 | Over Utilized | m7i-flex.large | eu-west-1 | ··· |
| i-1258c1c73669b8e | Over Utilized | g4dn.xlarge | us-west-2 | ··· |

## INSTANCES SUB METRICS(CPU, MEMORY etc.)



i-ee505aeb88b036 Under Utilized m7i-flex.large eu-north-1

vCPU 2

RAM 8 GB

GPU 0

$ / hr $0.086

Uptime (h) 1295

CPU avg (7d) 8%

Mem avg (7d) 16%

i-13fa75f4564b253 Optimal m7i-flex.large us-west-2

vCPU 2

RAM 8 GB

GPU 0

$ / hr $0.086

Uptime (h) 693

CPU avg (7d) 37%

Mem avg (7d) 44%

i-ebccd083bcc8f4 Over Utilized c5n.4xlarge eu-west-1

vCPU 16

RAM 42 GB

GPU 0

$ / hr $0.952

Uptime (h) 453

CPU avg (7d) 81%

Mem avg (7d) 76%

---

# Cost Attribution Panel

This panel helps users understand **how EC2 costs map back** to scientific jobs, teams, or infrastructure dimensions.

### Features:

- **Breakdowns:** Group costs by **Region**, **Instance Type**, **Usage Type**
- **Views:** Toggle between **Bar** and **Line** charts for time-series or categorical comparison
- **Comparison: Month-over-Month** comparison for the same grouping
- **Filters:** Multi-select **Tag values** and **Dimension values** that persist across the panel
- **Totals:** Clear separation of **Total**, **Attributed**, and **Unaccounted** cost for Tags

### Visual cues & anomaly handling:

- **Spikes:** Local spike detection highlights pronounced peaks (value > ~1.5× neighboring mean)
- **Future estimates:** Future periods render as **dashed lines** or **outlined bars** with a banner note

### Totals explained:

- **Total** — Sum of all costs for the selected range and filters
- **Attributed** — Portion matched to the selected **Tag key** (e.g., `Team` )

- **Unaccounted** — Portion **without that tag**, surfacing data-hygiene gaps for chargeback/showback

**Why these metadata dimensions:**

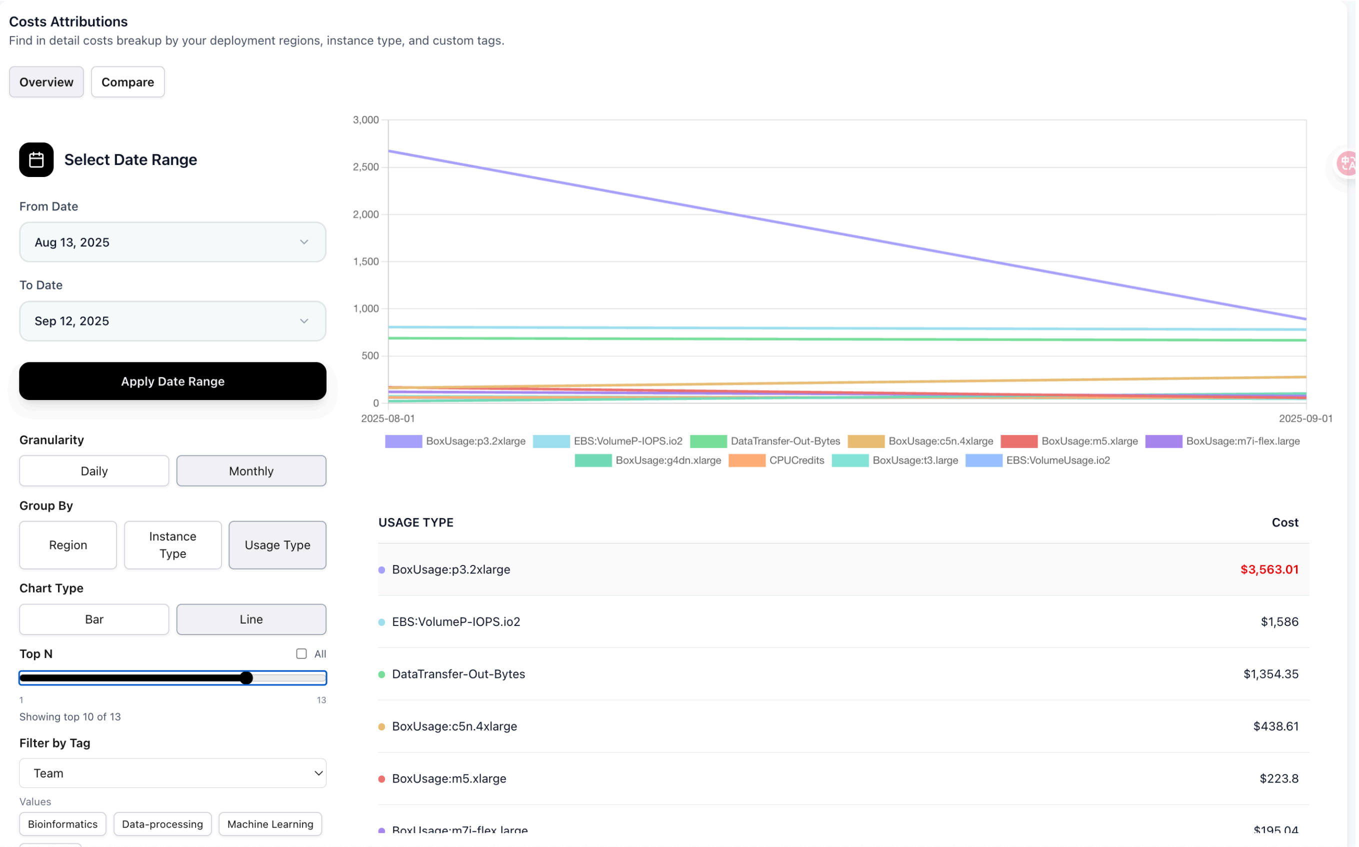
Region-based deployments, usage types (such as CloudFront data transfers), and instance sizes are key drivers of cloud costs. These dimensions make it easier to pinpoint where optimizations can have the most impact — for example, consolidating workloads in fewer regions or right-sizing instance families.

Additionally, **Team** and **Project** tags could be assumed as critical for collaborative research environments, enabling teams to attribute costs to the right workflows or departments.

The UI intentionally limits grouping and filtering options to these key dimensions, helping users reach conclusions quickly without being overwhelmed by too many variables.

## COST ATTRIBUTION COMPONENT

(Overview with line chart)



Costs Attributions

Find in detail costs breakup by your deployment regions, instance type, and custom tags.

From Date: Aug 13, 2025

To Date: Sep 12, 2025

Granularity: Monthly

Group By: Usage Type

Chart Type: Line

Top N ☐ All

Showing top 10 of 13

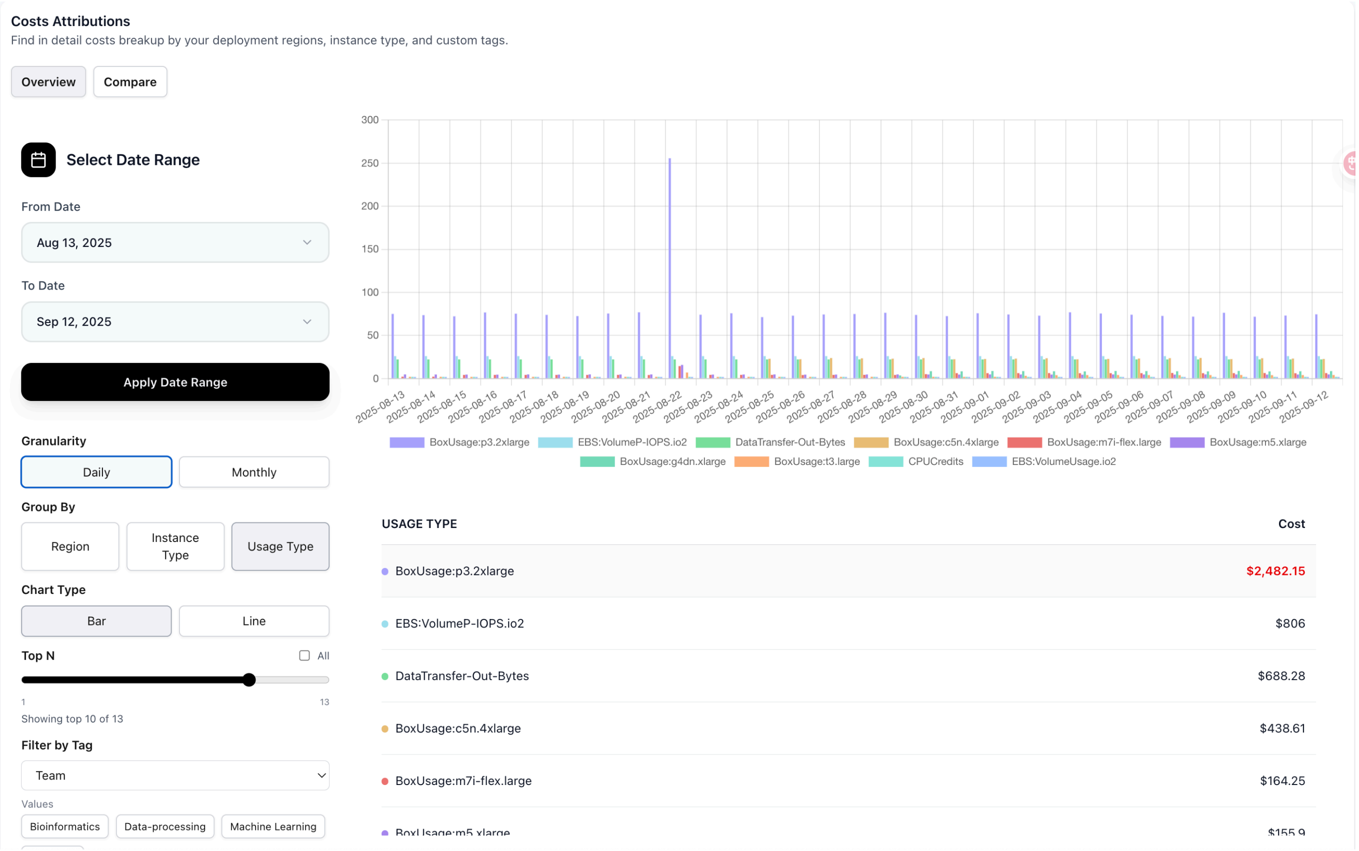
Filter by Tag: Team

Values: Bioinformatics, Data-processing, Machine Learning

| USAGE TYPE | Cost |
|---|---|
| ● BoxUsage:p3.2xlarge | $3,563.01 |
| ● EBS:VolumeP-IOPS.io2 | $1,586 |
| ● DataTransfer-Out-Bytes | $1,354.35 |
| ● BoxUsage:c5n.4xlarge | $438.61 |
| ● BoxUsage:m5.xlarge | $223.8 |
| ● BoxUsage:m7i-flex.large | $195.04 |

## COST ATTRIBUTION COMPONENT

(Overview with bar chart)



**Costs Attributions**

Find in detail costs breakup by your deployment regions, instance type, and custom tags.

Overview | Compare

**Select Date Range**

From Date: Aug 13, 2025

To Date: Sep 12, 2025

Apply Date Range

Granularity: [✓ Daily] Monthly

Group By: Region | Instance Type | [✓ Usage Type]

Chart Type: [✓ Bar] Line

Top N ☐ All

1 — 13

Showing top 10 of 13

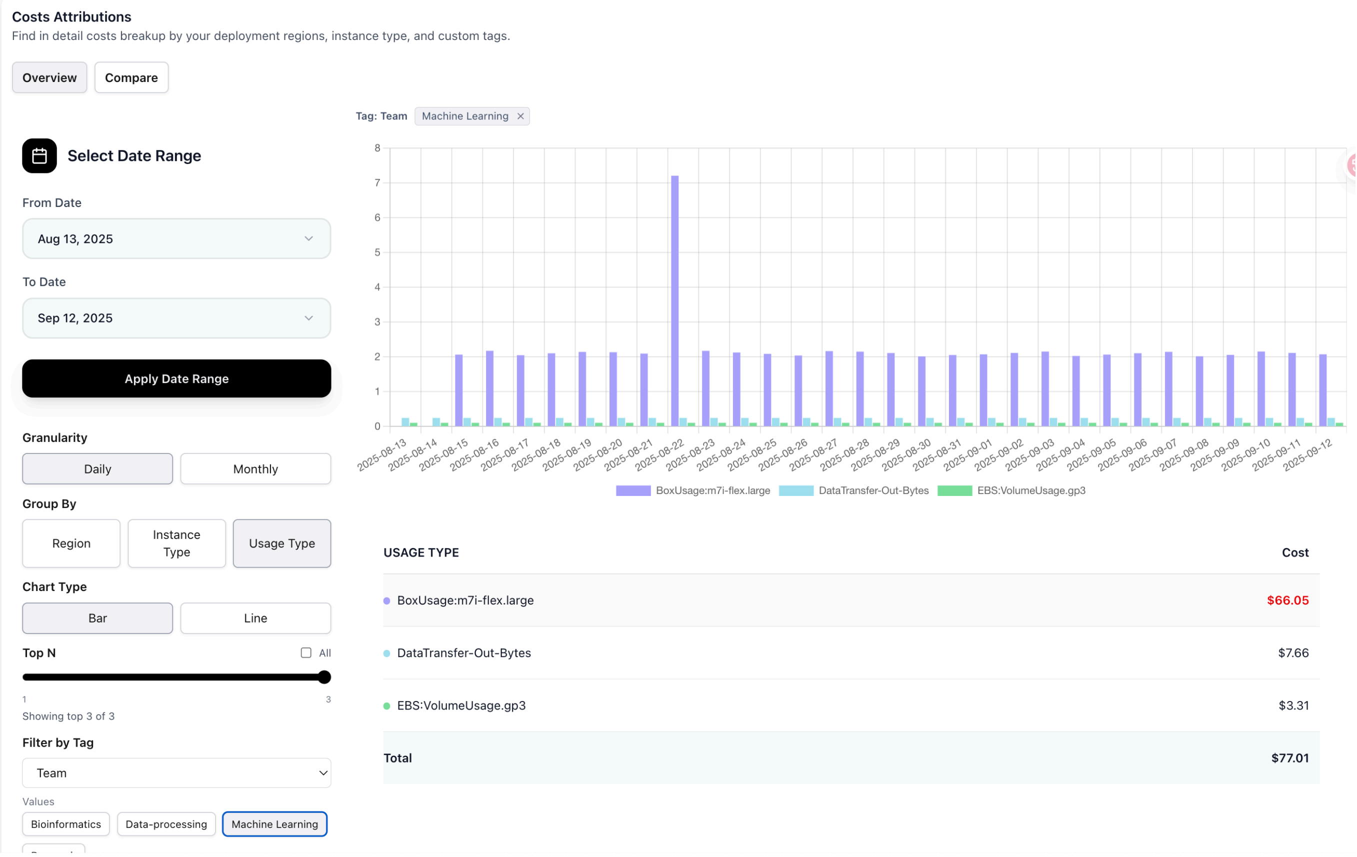
Filter by Tag: Team

Values: Bioinformatics | Data-processing | Machine Learning

| USAGE TYPE | Cost |
|---|---|
| ● BoxUsage:p3.2xlarge | $2,482.15 |
| ● EBS:VolumeP-IOPS.io2 | $806 |
| ● DataTransfer-Out-Bytes | $688.28 |
| ● BoxUsage:c5n.4xlarge | $438.61 |
| ● BoxUsage:m7i-flex.large | $164.25 |
| ● BoxUsage:m5.xlarge | $155.9 |

## COST ATTRIBUTION COMPONENT

(Overview with tag filters)



**Costs Attributions**

Find in detail costs breakup by your deployment regions, instance type, and custom tags.

Overview | Compare

Tag: Team | Machine Learning ×

**Select Date Range**

From Date: Aug 13, 2025

To Date: Sep 12, 2025

Apply Date Range

Granularity: [✓ Daily] Monthly

Group By: Region | Instance Type | [✓ Usage Type]

Chart Type: [✓ Bar] Line

Top N ☐ All

1 — 3

Showing top 3 of 3

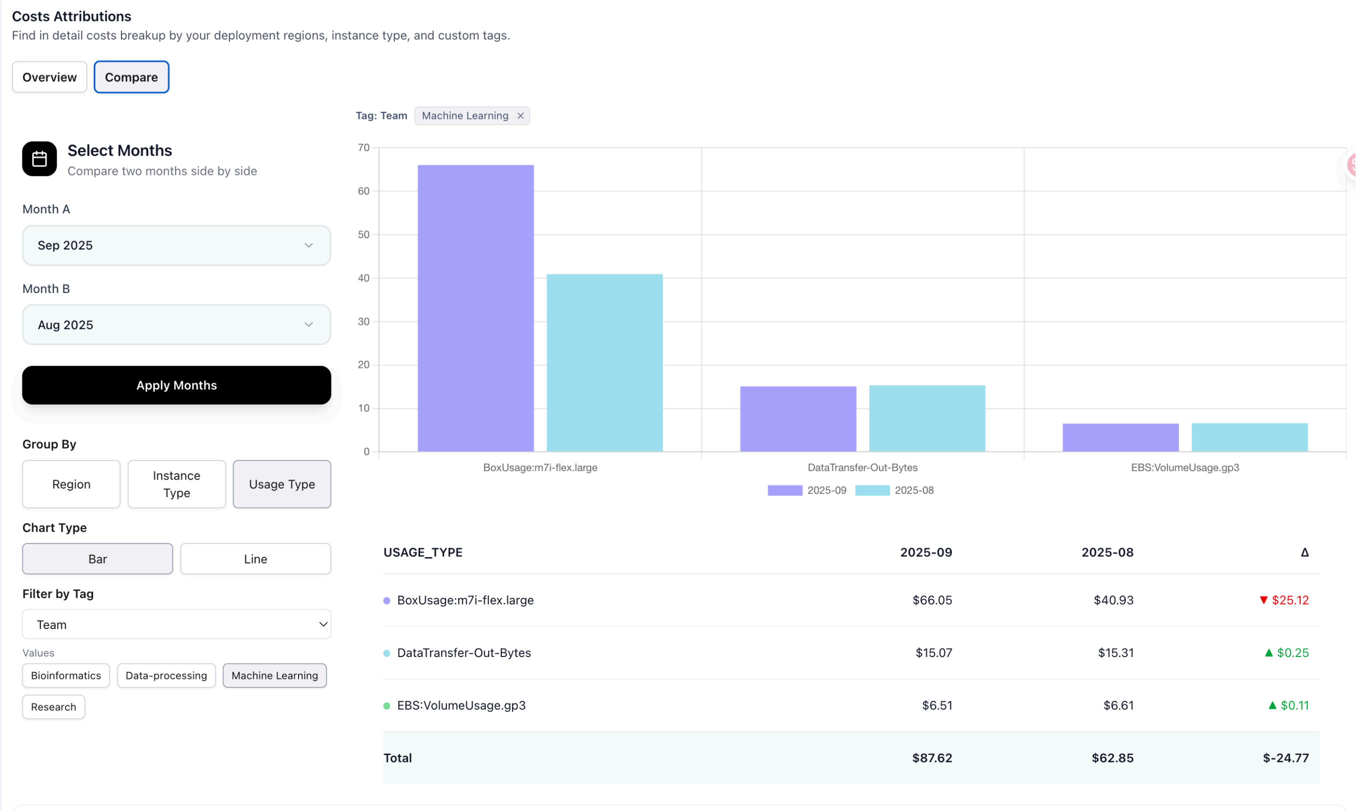
Filter by Tag: Team

Values: Bioinformatics | Data-processing | [✓ Machine Learning]

| USAGE TYPE | Cost |
|---|---|
| ● BoxUsage:m7i-flex.large | $66.05 |
| ● DataTransfer-Out-Bytes | $7.66 |
| ● EBS:VolumeUsage.gp3 | $3.31 |
| Total | $77.01 |

## COST ATTRIBUTION COMPONENT

(Comparision with line chart)



Month A: Sep 2025

Month B: Aug 2025

Filter by Tag: Team

| USAGE_TYPE | 2025-09 | 2025-08 | Δ |
|---|---|---|---|
| ● BoxUsage:m7i-flex.large | $66.05 | $40.93 | ▼ $25.12 |
| ● DataTransfer-Out-Bytes | $15.07 | $15.31 | ▲ $0.25 |
| ● EBS:VolumeUsage.gp3 | $6.51 | $6.61 | ▲ $0.11 |
| Total | $87.62 | $62.85 | $-24.77 |

# Live Cloud Cost Overview

This panel sits at the top of the dashboard and provides an overall summary to the user.

**Features:**
This section is divided into two parts:

### First part shows:

- **KPIs:** Total Cost (MTD), Daily Burn (7-day average), Projected Monthly Spend
- **Trend:** 7-day line chart of daily cost
- **Anomaly Cues:** Highlights spikes (≥ ~2σ) with red points and a callout list of spike dates and amounts.
  - Mean and standard deviation across the 7-day values are computed, and points with `(v − mean)/sd > 2` are flagged.
  - Spikes are visually emphasised (larger red points) and listed in **SpikeCallouts**.

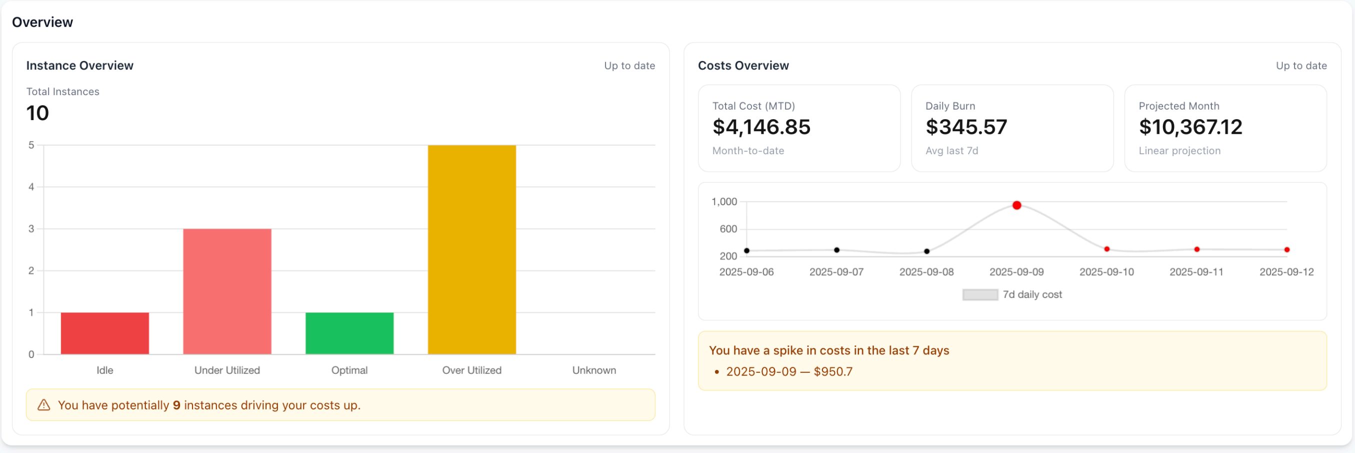
### Second part shows:

- **Risk Strip:** A compact bar chart of instance utilisation states (Idle / Under / Optimal / Over / Unknown), plus an alert summarising the count of potentially cost-driving instances (Idle + Under + Over).

- **SpikeCallouts:** A lightweight anomaly panel listing outlier days so decision-makers can act quickly without combing through the entire chart.

These two elements were chosen to give the user a high-level, quick view of what might be going wrong.
I used static data for the anomaly detection chart since neither the mock data nor the AWS data produced a clear spike during testing.

## OVERVIEW COMPONENT



Overview

Instance Overview

Up to date

Total Instances

10

⚠ You have potentially **9** instances driving your costs up.

Costs Overview

Up to date

Total Cost (MTD)
$4,146.85
Month-to-date

Daily Burn
$345.57
Avg last 7d

Projected Month
$10,367.12
Linear projection

You have a spike in costs in the last 7 days

- 2025-09-09 — $950.7