

Project 1

AWS CloudWatch Dashboards for Comprehensive Monitoring

Batch

MCA 7 Feb 2025 &
DevOps 2 Jun 2025

Name

Suraj Molke

Name of the Mentor

Ravindra Bagle Sir
Swati Zampal Ma'am



Fortune Cloud Technologies

2nd Floor, Shirodkar House, Opposite To Amit Cafe, Congress House Road, Near Pune Municipal Corporation, Shivajinagar, Pune - 411005.

Project: 1 <https://github.com/Smolke9/AWS-CloudWatch-Dashboards-for-Comprehensive-Monitoring.git>

Title: AWS CloudWatch Dashboards for Comprehensive Monitoring

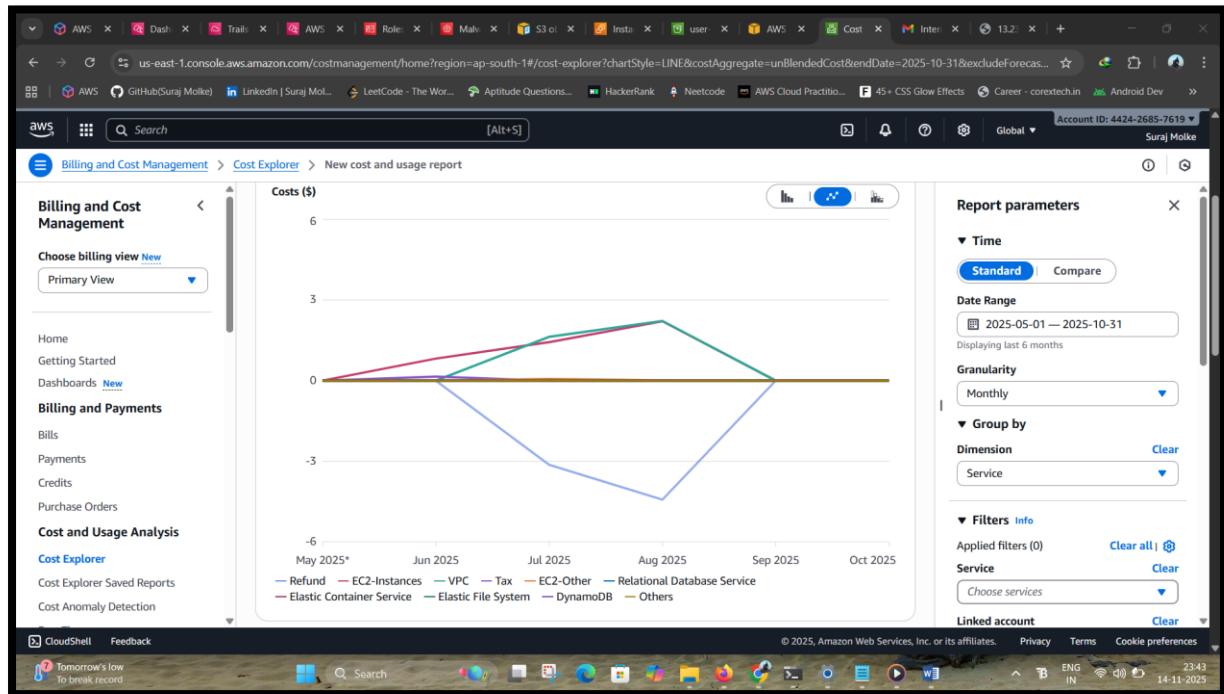
- ✓ Services used:
- ✓ Amazon CloudWatch (Dashboards, Metrics, Logs Insights)
- ✓ AWS Config (for compliance)
- ✓ AWS GuardDuty (for security threat detection)
- ✓ AWS CloudTrail (for API monitoring)
- ✓ IAM (for access control)
- ✓ EC2 (host to push logs)
- ✓ S3 (used for AWS Config)
- ✓ Application Load Balancer (for network monitoring)

Objective:

Build and configure a CloudWatch Dashboard that provides real-time visibility into key operational and financial metrics across four focus areas:

Step 1. Enable Billing Tools

1. Go to Billing → Cost Management
2. Enable Cost Explorer



The screenshot shows the AWS Cost Explorer interface. On the left, a sidebar lists navigation options such as Home, Getting Started, Dashboards, Billing and Payments, Bills, Payments, Credits, Purchase Orders, Cost and Usage Analysis, and Cost Explorer. The main area displays a 'Cost and usage breakdown' table for September 2025. The table includes columns for Total, May 2025*, June 2025, July 2025, August 2025, and September. Services listed include Total costs, EC2-Instances, VPC, Tax, EC2-Other, Relational Database Service, Elastic Container Service, Elastic File System, DynamoDB, and S3. The total cost for September is \$0.97. To the right, there is a 'Report parameters' panel with sections for Time (Date Range: 2025-05-01 — 2025-10-31), Group by (Dimension: Service), Filters (Info: Applied filters (0)), and Linked account.

Step 2. Create Reports in AWS Cost Explorer

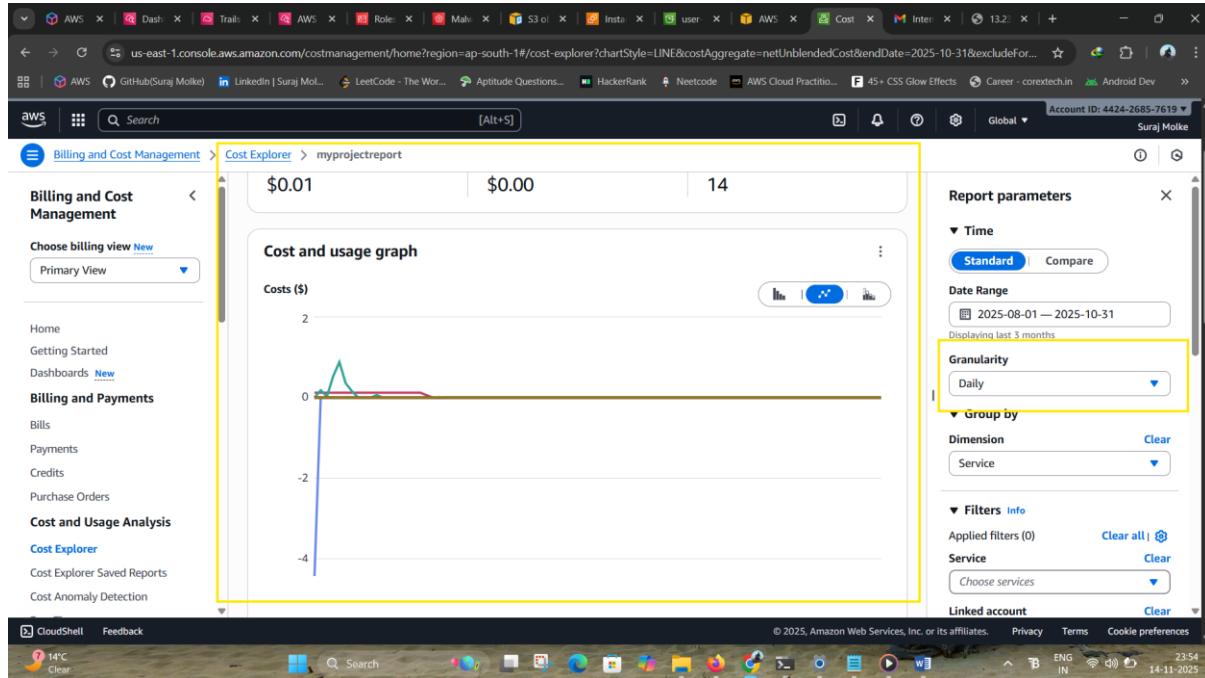
2.1 Total Cost Over Time

- Open Cost Explorer
- Granularity: Daily
- Metric: Unblended Cost
- Group by: None
- Save report

The screenshot shows the AWS Cost Explorer interface after saving a report. A green success message at the top states: "Successfully saved 'myprojectreport' report. If a dashboard widget of this report is created, it will be updated to reflect the saved changes." Below this, the report is titled "myprojectreport". It features a "Cost and usage overview" section with "Total cost" (\$0.01) and "Average monthly cost" (\$0.00). Below this is a "Cost and usage graph" showing a single data series labeled "Costs (\$)" with a value of 0.01. The graph has two horizontal axis labels: "0.01" and "0.0075". The right side of the screen contains the same "Report parameters" panel as the previous screenshot, with various filter and account selection options.

2.2. Daily Estimated Charges by Service

- Cost Explorer → Create Report
- Granularity: Daily
- Group by: Service (EC2, S3, RDS, etc.)
- Save report



2.3. Monthly Cost Breakdown

- Granularity: Monthly
- Group by: Service
- Save report

Use charts/screenshots in your dashboard report.

The screenshot shows the AWS Cost Explorer interface with a yellow box highlighting the 'Cost and usage breakdown' table. The table lists monthly costs for various AWS services, including Total costs, VPC, EC2-Instances, EC2-Other, Relational Database Service, Elastic Container Service, S3, Key Management Service, CloudWatch, Tax, and CloudFormation. The 'Report parameters' sidebar on the right is also highlighted, showing the date range from August 1, 2025, to October 31, 2025, and the granularity set to 'Monthly'. The sidebar also includes sections for 'Group by' (set to 'Service') and 'Filters'.

Service	Service total	August 2025	September 2025	October 2025
Total costs	\$0.01	\$0.01	\$0.00	\$0.00
VPC	\$2.22	\$2.22	\$0.00	\$0.00
EC2-Instances	\$2.21	\$2.21	\$0.00	\$0.00
EC2-Other	\$0.01	\$0.01	\$0.00	\$0.00
Relational Database Service	\$0.00	\$0.00	-	-
Elastic Container Service	\$0.00	-	\$0.00	-
S3	\$0.00	\$0.00	\$0.00	\$0.00
Key Management Service	\$0.00	\$0.00	\$0.00	\$0.00
CloudWatch	\$0.00	\$0.00	-	-
Tax	\$0.00	\$0.00	\$0.00	\$0.00
CloudFormation	\$0.00	-	\$0.00	-

Step 3. Application & System Logs

Steps:

3.1. Send Logs to CloudWatch Logs

From EC2

Install and configure CloudWatch Agent

```
sudo yum install amazon-cloudwatch-agent -y
```

```
sudo nano /opt/aws/amazon-cloudwatch-agent/bin/config.json
```

Add log paths:

```
{
  "logs": {
    "logs_collected": {
      "files": {
        "collect_list": [
          { "file_path": "/var/log/messages", "log_group_name": "system-logs" },
          { "file_path": "/var/log/nginx/error.log", "log_group_name": "nginx-logs" }
        ]
      }
    }
  }
}
```

Start agent:

```
sudo systemctl start amazon-cloudwatch-agent
```

Step 4. Create CloudWatch Log Insights Queries

Go to:

CloudWatch → Logs → Log Insights → Select a Log Group

Step 5. Add Log Insights to Dashboard

Go to:

CloudWatch → Dashboards → Create Dashboard → Add Widget

Add widgets:

- Log Insights → Query results (Table)
- Line graph for response times
- Bar chart for error counts
- Number widget for total errors in last 1 hour

Step 6 System Logs for CloudWatch Logs Dashboard

6.1. /var/log/messages

- This is the **main system log file** on Linux servers (Amazon Linux, CentOS, RHEL).
- Contains:
 - Kernel events
 - Service start/stop logs
 - System warnings
 - General OS activity
- Useful for identifying:
 - Server crashes
 - Restart issues
 - Hardware/network errors
 - Authentication problems

Path: /var/log/messages

2. /var/log/nginx/error.log

- This file stores all **Nginx-related errors**.
- Contains:
 - HTTP errors
 - Backend connection failures
- **Install CloudWatch Agent**
- sudo yum install amazon-cloudwatch-agent -y
- **Edit agent config**

```
{  
  "logs": {  
    "logs_collected": {  
      "files": {  
        "collect_list": [  
          {  
            "file_path": "/var/log/messages",  
            "log_group_name": "/ec2/system-messages",  
            "log_stream_name": "{instance_id}-messages",  
            "timestamp_format": "%b %d %H:%M:%S"  
          }  
        ]  
      }  
    }  
  }  
}
```

```

    },
    {
      "file_path": "/var/log/nginx/error.log",
      "log_group_name": "/ec2/nginx-error",
      "log_stream_name": "{instance_id}-nginx-error",
      "timestamp_format": "%Y/%m/%d %H:%M:%S"
    }
  ]
}

}
}
}
}
}

```

- **Start the agent**
- sudo systemctl start amazon-cloudwatch-agent
- Logs appear in:
 - **/aws/system-logs**
 - **/aws/nginx-logs**
 - Permission problems

The screenshot shows the AWS CloudWatch Log Groups interface. The left sidebar navigation includes CloudWatch, AI Operations, Alarms, Logs (with Log groups selected), Metrics, Application Signals, CloudShell, and Feedback. The main content area displays a table titled 'Log groups (5)'. The table has columns for Log group, Log class, Anomaly detection, Data processing, Sensitivity, and Retention. The first row, '/ec2/nginx-error', is highlighted with a yellow box. The other rows are: '/ecs/mytask1' (Standard, Configure, -,-, Never expire), 'RDSOSMetrics' (Standard, Configure, -,-, 1 month), 'aws-cloudtrail-logs-442426857619-0a179bec' (Standard, Configure, -,-, Never expire), and 'aws-cloudtrail-logs-442426857619-0a179becc' (Standard, Configure, -,-, Never expire). The bottom of the screen shows the Windows taskbar with various pinned icons.

Log group	Log class	Anomaly d...	Data ...	Sensi...	Retenti...
/ec2/nginx-error	Standard	Configure	-	-	Never exp...
/ecs/mytask1	Standard	Configure	-	-	Never exp...
RDSOSMetrics	Standard	Configure	-	-	1 month
aws-cloudtrail-logs-442426857619-0a179bec	Standard	Configure	-	-	Never exp...
aws-cloudtrail-logs-442426857619-0a179becc	Standard	Configure	-	-	Never exp...

Network Performance

Step 7. VPC Network Monitoring

Collect and visualize:

- VPC Flow Logs (Accepted/Rejected traffic)
- Bytes in/out
- Packet count
- Rejected connections by security groups or NACLs

Steps :

1. Enable VPC Flow Logs (VPC → Flow Logs → Create).
2. Send logs to CloudWatch Logs.
3. Use Log Insights to monitor:
 - Top IPs hitting instance
 - Rejected traffic
 - Port scans

Useful Log Insights Query:

```
fields srcAddr, dstPort, action, bytes  
| filter action="REJECT"  
| sort bytes desc
```

The screenshot shows the AWS CloudWatch Logs Insights interface. The left sidebar navigation includes 'CloudWatch' (selected), 'Dashboards', 'AI Operations', 'Alarms', 'Logs' (selected), 'Log groups', 'Log Anomalies', 'Live Tail', 'Logs Insights' (selected), 'Contributor Insights', 'Metrics', 'Application Signals (APM)'. The main content area has a title 'Logs Insights - Analyze with OpenSearch - new'. It features a 'Logs Insights Info' section with a placeholder 'Select log groups, and then run a query or choose a sample query.' Below it is a 'Logs Insights QL' editor with the following query:

```
1 fields srcAddr, dstPort, action, bytes  
2 | filter action="REJECT"  
3 | sort bytes desc
```

Below the editor are buttons for 'Start tailing', time range (30m, 3h, 1h), 'Compare (Off)', and 'Local timezone'. To the right are filters for 'Log class' (Standard), 'Account(s)' (All accounts), and 'Saved and sample queries'. A 'See recommendations' section displays the provided query. At the bottom, there's a 'Query generator' button and a toolbar with icons for copy, paste, and refresh.

Step 7.1. EC2 Network Metrics

From CloudWatch → Metrics → EC2 you can monitor:

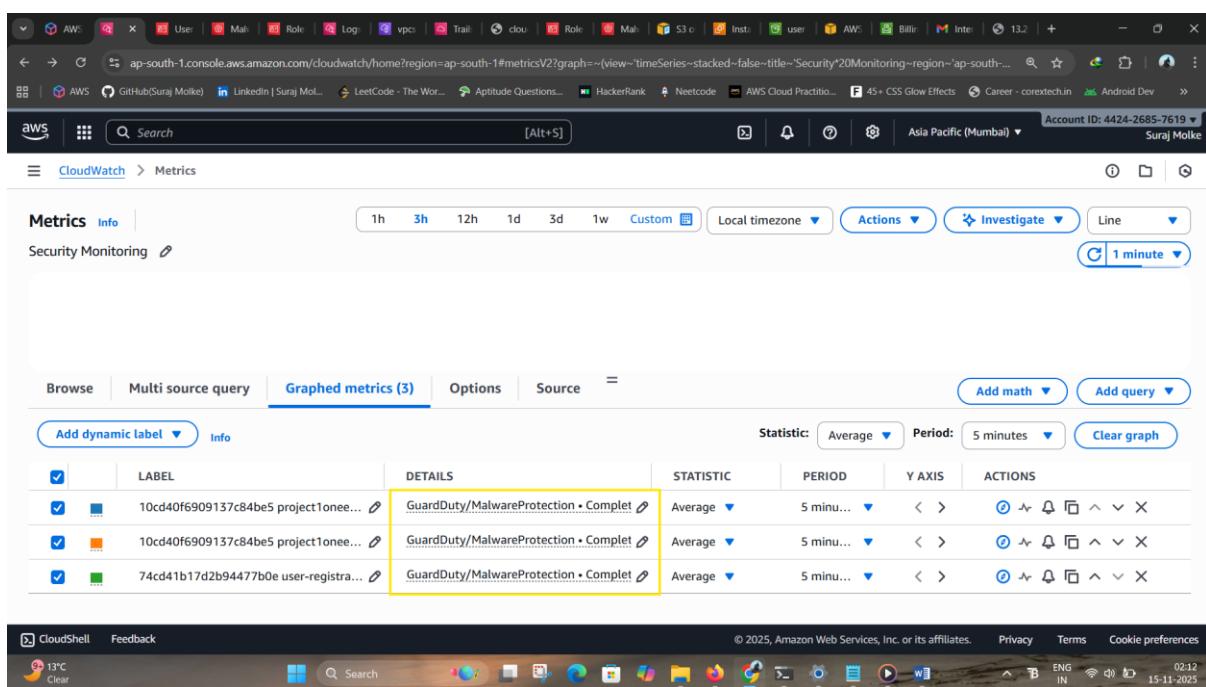
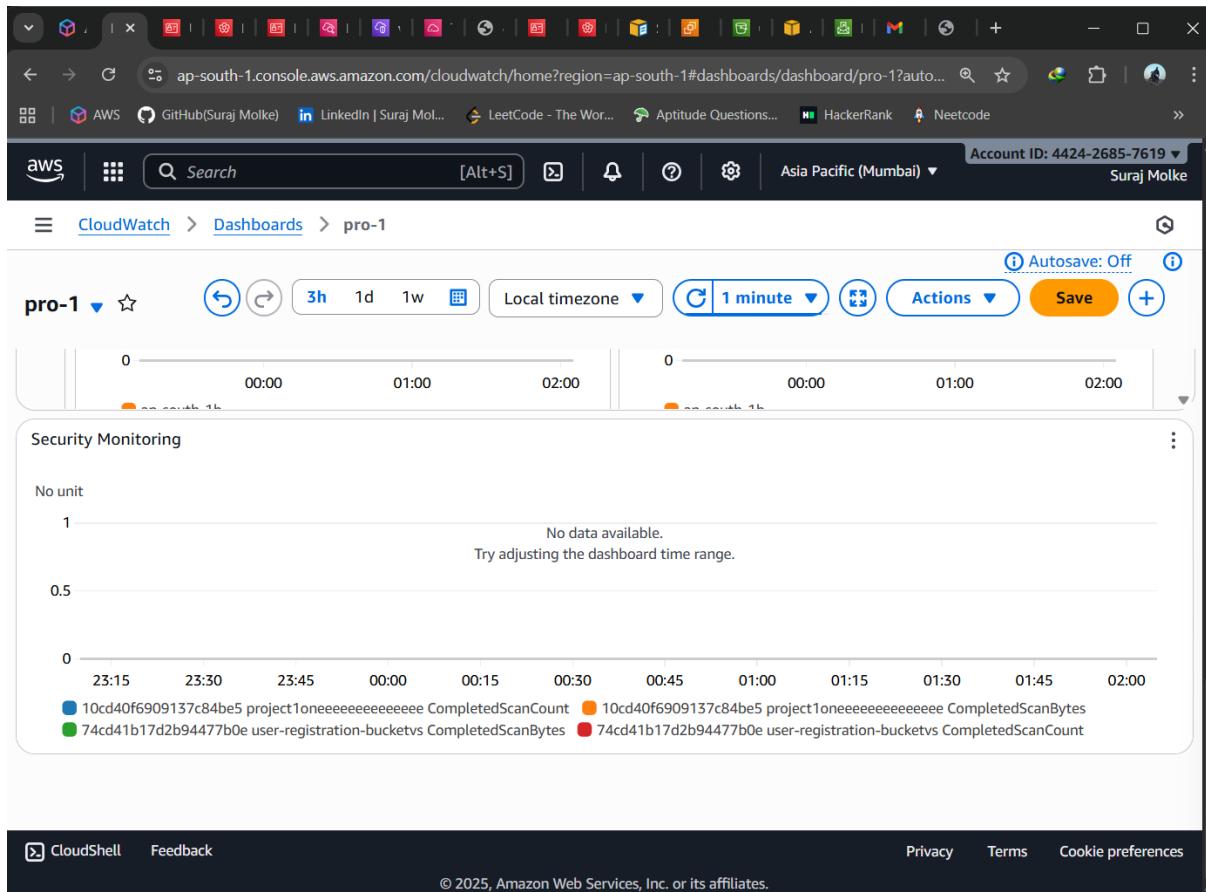
- NetworkIn (bytes received)
- NetworkOut (bytes sent)
- NetworkPacketsIn/Out
- StatusCheckFailed

The screenshot shows the AWS CloudWatch Metrics interface. At the top, there are navigation links for 'Metrics' and 'Info', and a search bar. Below the search bar are time range buttons: 1h, 3h, 12h, 1d, 3d, 1w, Custom, and Local timezone. There are also 'Actions', 'Investigate', and 'Line' buttons. A yellow box highlights the 'Metrics' tab. On the left, a sidebar shows 'Network Monitoring'. The main area displays a table of metrics for 'Server Project 1'. The columns include 'Source' (labeled 'Server Project 1'), 'Metric Name', 'Unit', and 'Alarms'. The metrics listed are: CPUUtilization, NetworkIn, NetworkPacketsOut, NetworkOut, NetworkPacketsIn, and MetadataNoTokenRejected. All metrics show 'No alarms'. At the bottom of the page, there are links for 'CloudShell', 'Feedback', and various system status indicators like temperature and battery level.

Step 8 Security Monitoring Dashboard — Short Steps

8.1. GuardDuty Monitoring

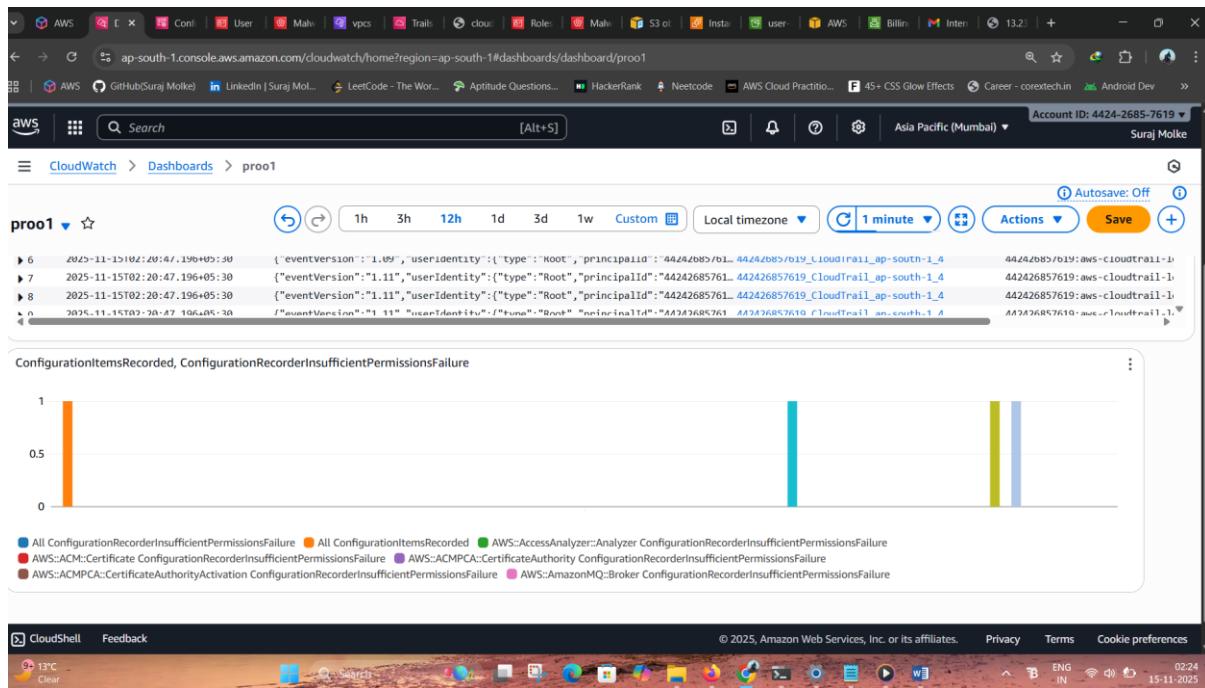
- Enable **GuardDuty** from the AWS Console.
- Go to **CloudWatch → Metrics → GuardDuty**.
- Monitor:
 - **Findings** (Severity: Low, Medium, High)
 - **Threat detections over time**
- Add widgets:
 - Line chart → *Total Findings Over Time*
 - Bar chart → *High Severity Findings*



8.2. AWS Config (Compliance Monitoring)

- Enable AWS Config → Choose recording & S3 log bucket.
- Go to CloudWatch → Metrics → Config.
- Monitor:

- **NonCompliantResourceCount**
- **ComplianceStatus**
- Dashboard widgets:
 - Number widget → *Total Non-Compliant Resources*
 - Gauge widget → *Compliance %*



8.3. CloudTrail (API Activity & Security Events)

- Ensure CloudTrail is enabled (1 trail per region).
- Use **CloudWatch Log Insights** to analyze activity:
 - Unusual logins
 - Root account usage
 - Deletion/modification events

Trails

Name	Home region	Multiregion trail	ARN	Insights	Organization trail	S3 bucket	Log file prefix	Cloud Watch Logs log group	Status
ProjectO-nemanagement-events	Asia Pacific (Mumbai)	Yes	arn:aws:cloudtrail:ap-south-1:442426857619:trail/ProjectOnemanagement-events	Disabled	No	aws-cloudtrail-logs-442426857619-0a179bcc	-	arn:aws:logs:ap-south-1:442426857619:log-group:aws-cloudtrail-logs-442426857619-0a179bcc	Disabled

Log Insights Query:

```
fields eventName, userIdentity.arn, sourceIPAddress
| filter errorCode = "AccessDenied"
| sort @timestamp desc
```

```
1 fields eventName, userIdentity.arn, sourceIPAddress
2 | filter errorCode = "AccessDenied"
3 | sort @timestamp desc
```

8.4. IAM Security Events

Monitor via CloudWatch metrics and CloudTrail logs:

- Access key usage
- Unauthorized attempts
- Password policy compliance

Useful query for Unauthorized attempts:

```
fields userIdentity.userName, eventName, errorCode, sourceIPAddress  
| filter errorCode like /AccessDenied/  
| sort @timestamp desc
```

The screenshot shows the AWS CloudWatch Logs Insights interface. The URL in the browser is [ap-south-1.console.aws.amazon.com/cloudwatch/home?region=ap-south-1#logV2:logs-insights\\$3FqueryDetail\\$3D~\(end~0~start~-3600~timeType='RELATIVE~tz='LOCAL~unit='...](https://ap-south-1.console.aws.amazon.com/cloudwatch/home?region=ap-south-1#logV2:logs-insights$3FqueryDetail$3D~(end~0~start~-3600~timeType='RELATIVE~tz='LOCAL~unit='...). The page title is "Logs Insights". The breadcrumb navigation shows "CloudWatch > Logs Insights > suraj/trails". The main area displays a query editor with the following code:

```
1 fields eventName, userIdentity.userName, sourceIPAddress  
2 | filter errorCode = "AccessDenied"  
3 | sort @timestamp desc
```

The interface includes a "Logs Insights Info" section, a time range selector (5m, 30m, 1h, 3h, 12h, Custom), a "Compare (Off)" button, and a "Local timezone" dropdown. To the right, there are sections for "Discovered fields" and "Saved and sample queries". At the bottom, there are links for "CloudShell", "Feedback", and system status indicators.

Dashboard widgets:

- Number widget → *Unauthorized API Calls Today*
- Status icon → *Access Key Rotation Age*

Conclusion

This project successfully demonstrates a complete, cost-efficient monitoring setup using AWS CloudWatch without relying on additional paid services like Lambda. By configuring four dedicated dashboards—Billing, Logs, Network Traffic, and Security—you

achieved full visibility across system performance and operational health. The EC2 instance, integrated with CloudWatch Agent and supported by appropriate IAM roles, enabled seamless log streaming and remote management through SSM. Realistic traffic generated using a load balancer with NGINX ensured practical network insights, while custom log groups and queries provided actionable details on failed logins, system events, and API activity. Overall, the solution delivers a fully functional, scalable, and low-cost monitoring architecture suitable for both learning and real-world cloud operations.