CloudWatch

CloudWatch service is monitoring your cloud resources. It helps you collect and track metrics, monitor logs, set alarms, and automate actions based on predefined rules. With CloudWatch, you can keep an eye on the performance, availability, and health of your AWS infrastructure and applications, ensuring they run smoothly and efficiently.

Features

Metric Monitoring: ***CloudWatch allows users to collect and track metrics from various AWS services in real time. These metrics could include CPU utilization, network traffic, disk I/O, and many others. Users can visualize this data through graphs and charts, enabling them to understand resource usage patterns and identify performance bottlenecks.***

Log Monitoring: ***CloudWatch Logs enables users to centralize, monitor, and analyze log data from their AWS resources and applications. By collecting log files in one place, users can easily search, filter, and analyze log events to troubleshoot issues, track changes, and gain insights into system behavior.***

Alarms: ***CloudWatch Alarms enable users to set up notifications based on predefined thresholds for their metrics. When a metric breaches a specified threshold, CloudWatch triggers an alarm, notifying users via various channels such as email, SMS, or Amazon SNS (Simple Notification Service). Alarms can be configured to trigger automated actions, such as scaling AWS resources or executing AWS Lambda functions, helping users respond to incidents promptly.***

Dashboards: ***CloudWatch Dashboards allow users to create customizable dashboards to visualize their metrics data. Users can create multiple dashboards with different sets of metrics, enabling them to monitor the health and performance of their AWS resources and applications at a glance.***

Events: ***CloudWatch Events enables users to respond to changes in their AWS environment by triggering automated actions based on predefined rules. Users can create event-driven workflows to automate tasks such as starting or stopping EC2 instances, resizing RDS databases, or invoking AWS Lambda functions in response to specific events.***

Benefits

Proactive Monitoring: ***CloudWatch enables proactive monitoring of AWS resources and applications, allowing users to detect and address issues before they impact performance or availability.***

Operational Efficiency: ***By centralizing monitoring and automation tasks within CloudWatch, users can streamline operations, reduce manual intervention, and improve resource utilization.***

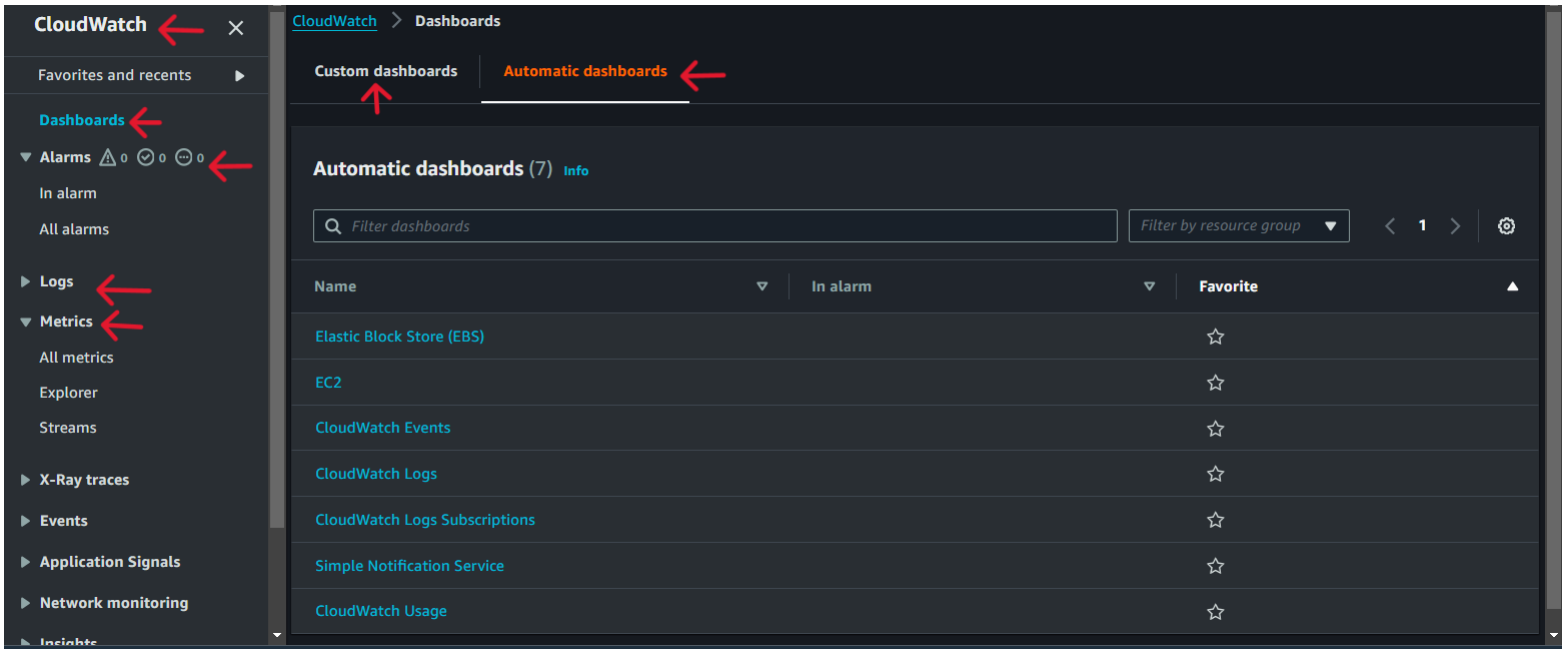
Insights and Analysis: ***CloudWatch provides valuable insights into resource usage, system behavior, and application performance through metrics visualization, log analysis, and anomaly detection.***

Scalability and Flexibility: ***CloudWatch is designed to scale with the growing needs of AWS customers, supporting large-scale deployments and diverse use cases across various industries and sectors.***

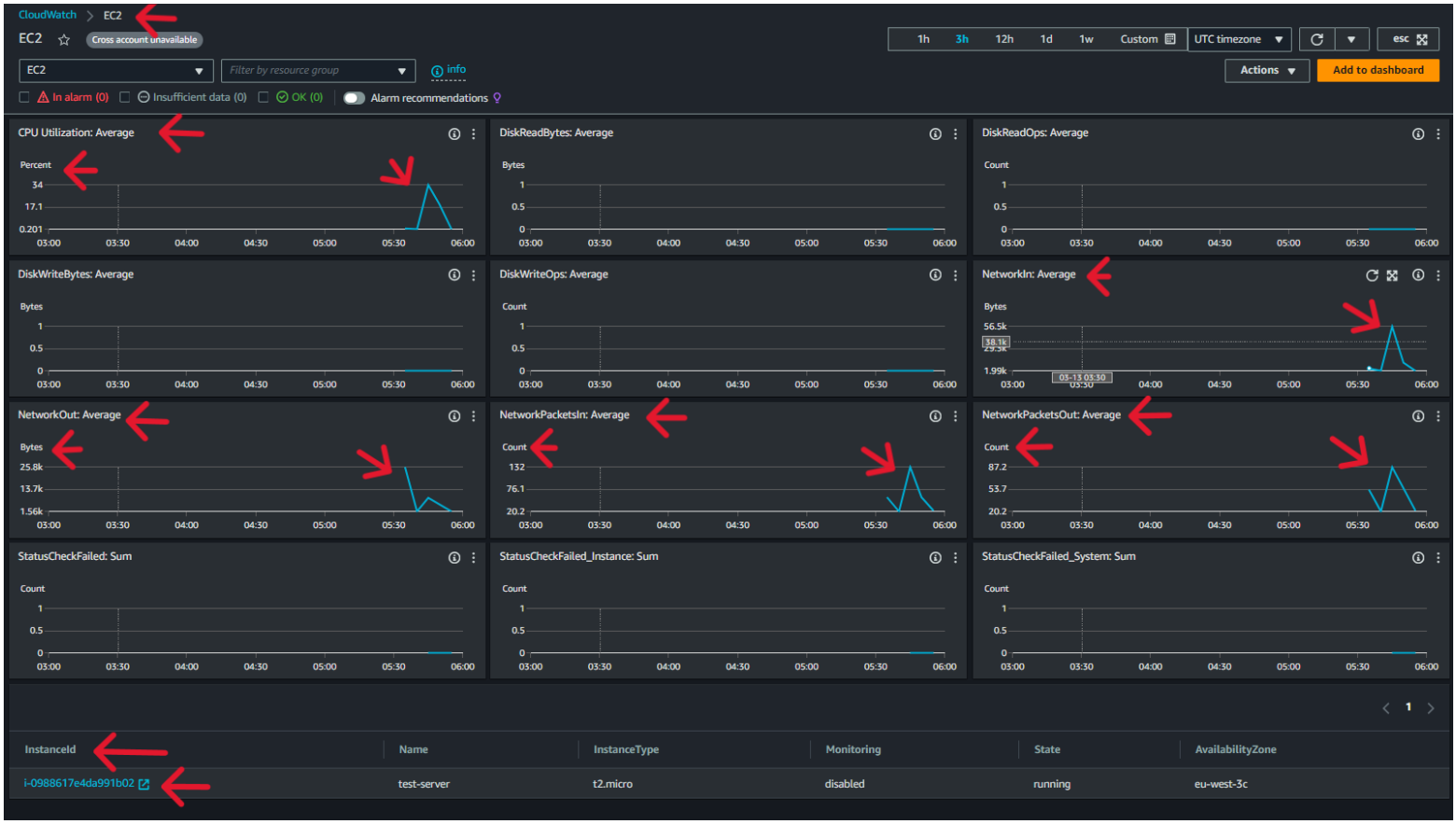
CloudWatch Dashboard

In CloudWatch Service, AWS provides an Automatic Dashboard and a Custom Dashboard.

- ***Automatic Dashboard***



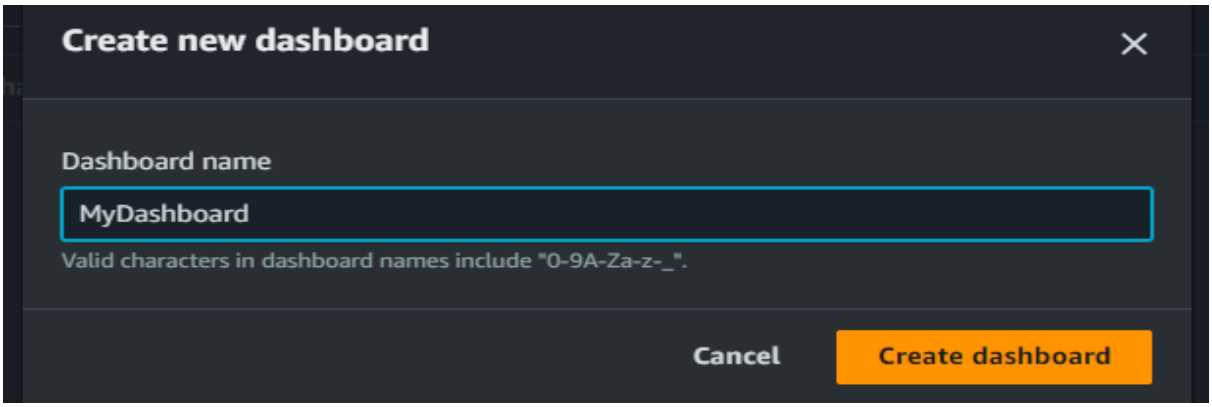
- ***Automatic Dashboard for EC2***



CloudWatch Automatic Dashboards provide instant visualizations of your AWS resources' performance metrics without manual setup. They enable you to easily monitor your infrastructure's health and quickly address any issues or anomalies (unusual or unexpected events or occurrences)

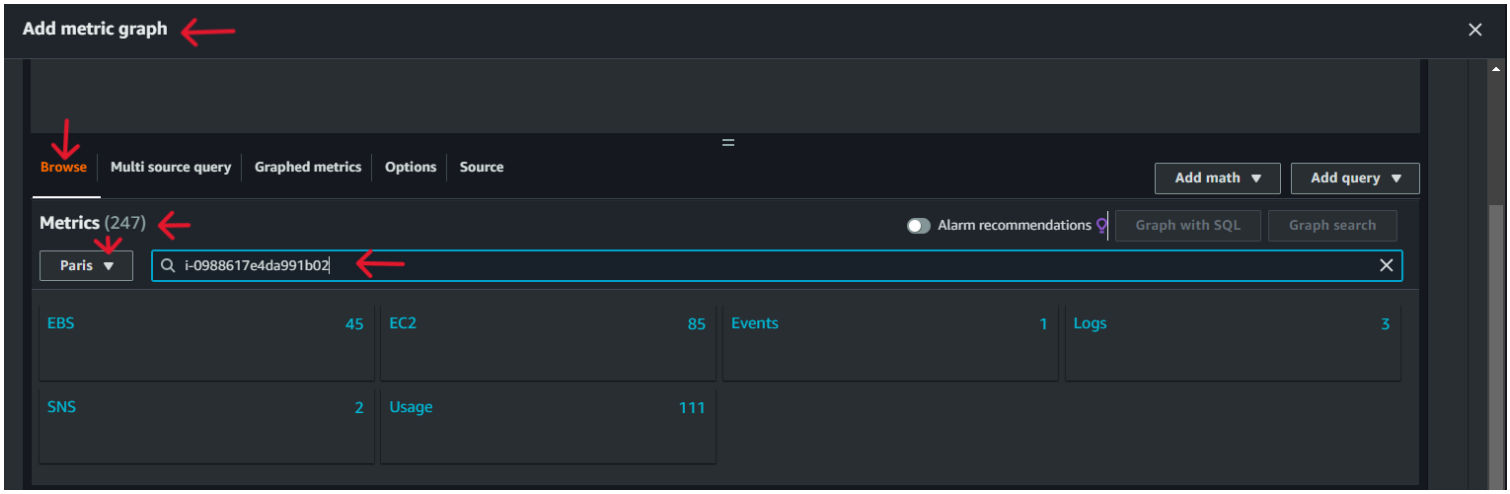
Now we create a Custom Dashboard.

1) ***Go to the Custom Dashboard option and click on the Create Dashboard Button.***
2) ***Name the dashboard and click on the Create Dashboard button.***

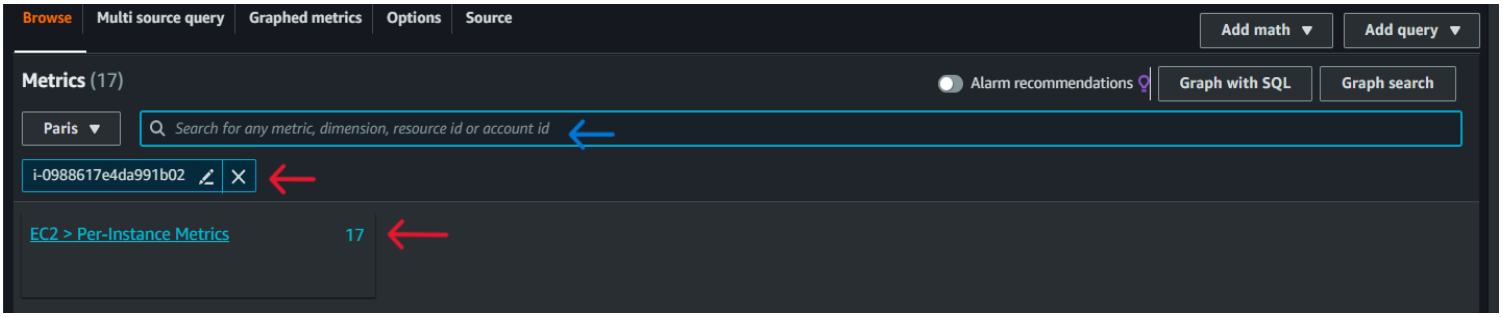


3) ***Add widget, Select widget type, and click on the next button.***

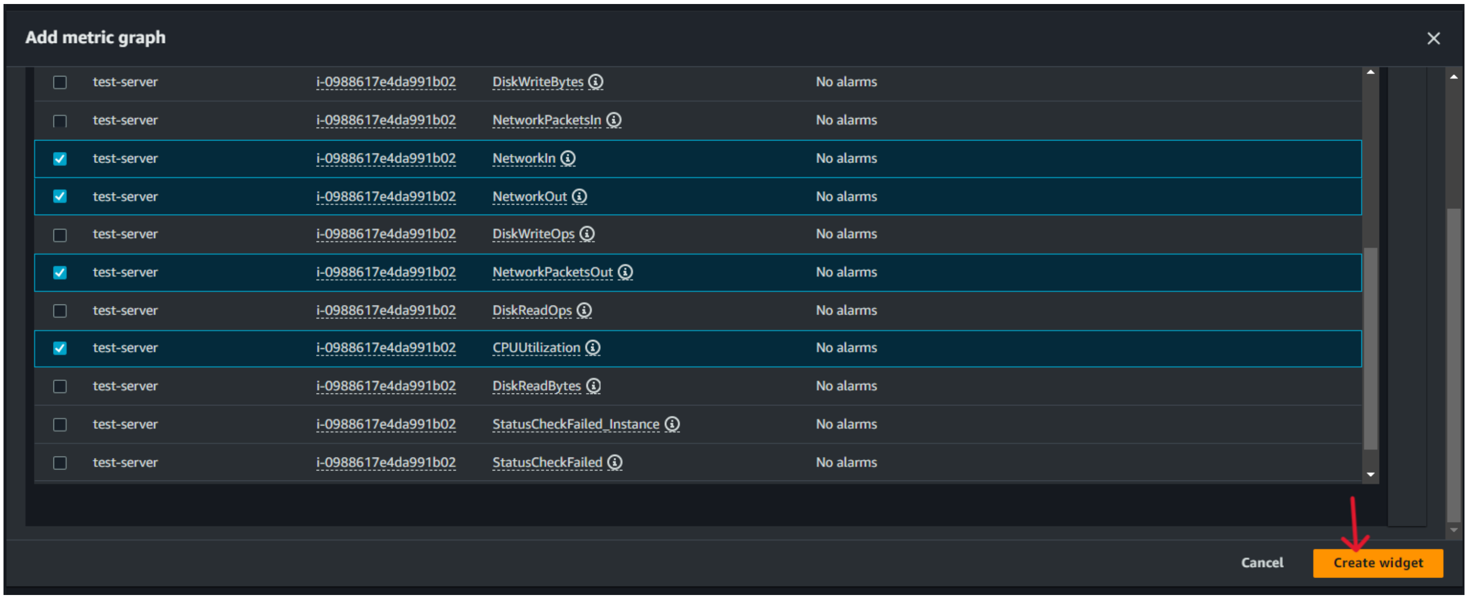
4) ***Add metric graph, enter Resource ID (now we go with ec2), and enter.***



5) ***Select Pre-Instance Metrics.***

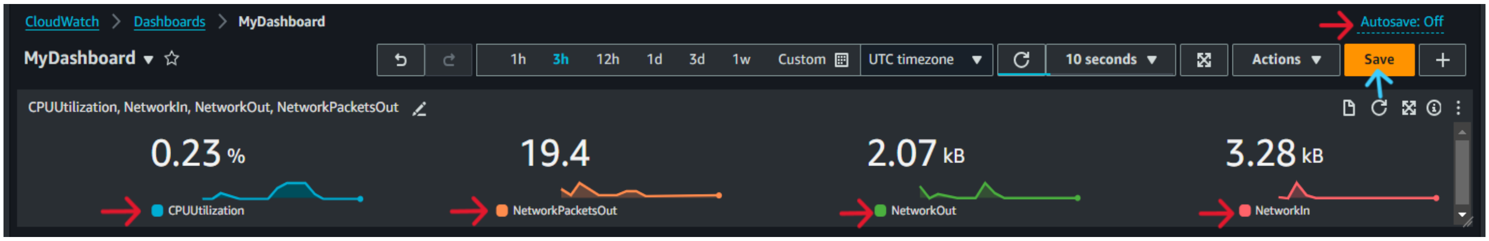


6) Select the Metrics that you want to monitor in your custom dashboard, and click on the Create Widget button.

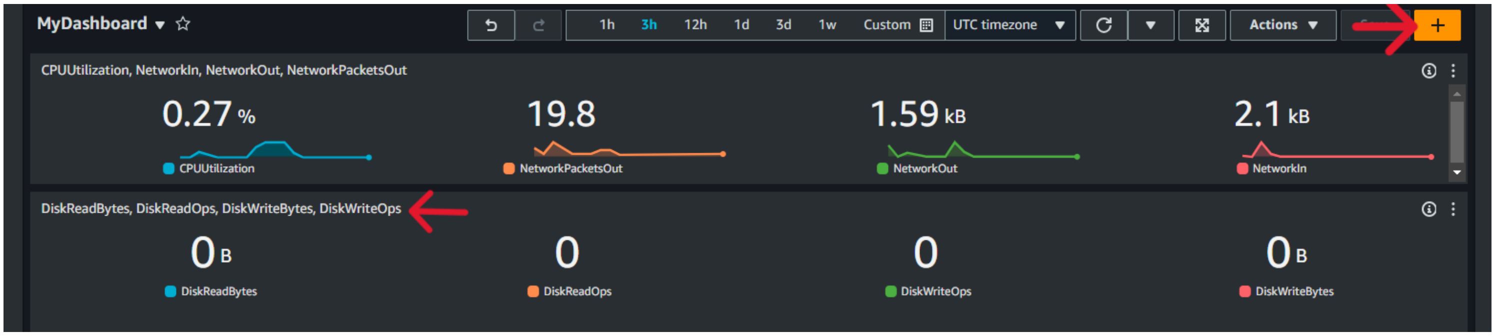


☐	test-server	i-0988617e4da991b02	DiskWriteBytes	No alarms
☐	test-server	i-0988617e4da991b02	NetworkPacketsIn	No alarms
☑	test-server	i-0988617e4da991b02	NetworkIn	No alarms
☑	test-server	i-0988617e4da991b02	NetworkOut	No alarms
☐	test-server	i-0988617e4da991b02	DiskWriteOps	No alarms
☑	test-server	i-0988617e4da991b02	NetworkPacketsOut	No alarms
☐	test-server	i-0988617e4da991b02	DiskReadOps	No alarms
☑	test-server	i-0988617e4da991b02	CPUUtilization	No alarms
☐	test-server	i-0988617e4da991b02	DiskReadBytes	No alarms
☐	test-server	i-0988617e4da991b02	StatusCheckFailed_Instance	No alarms
☐	test-server	i-0988617e4da991b02	StatusCheckFailed	No alarms

7) Now our custom dashboard is created, and the Widget of Ec2 Instance is added to it, save it.



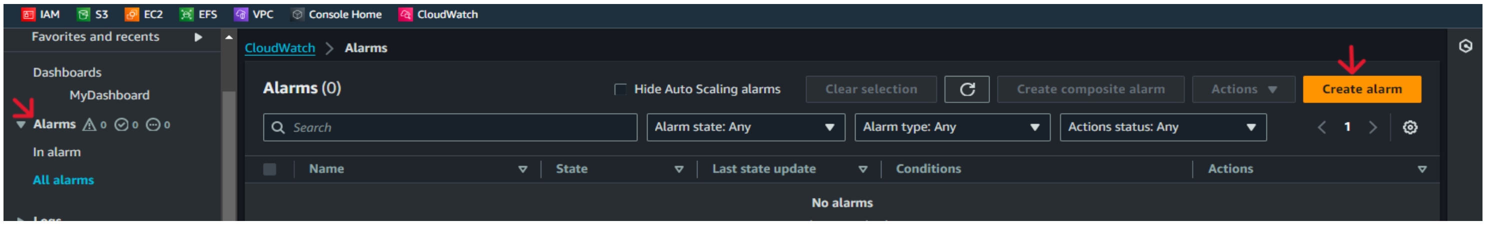
8) Now we can add more Widgets of other resources in our custom dashboard to monitor their metrics, logs, and alarms.



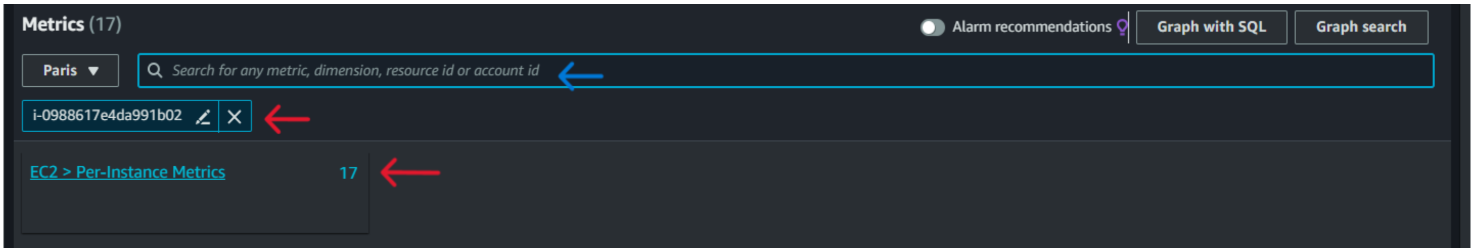
"" *""*

Now we create an Alarm

1) Click on Create Alarm in CloudWatch Service.

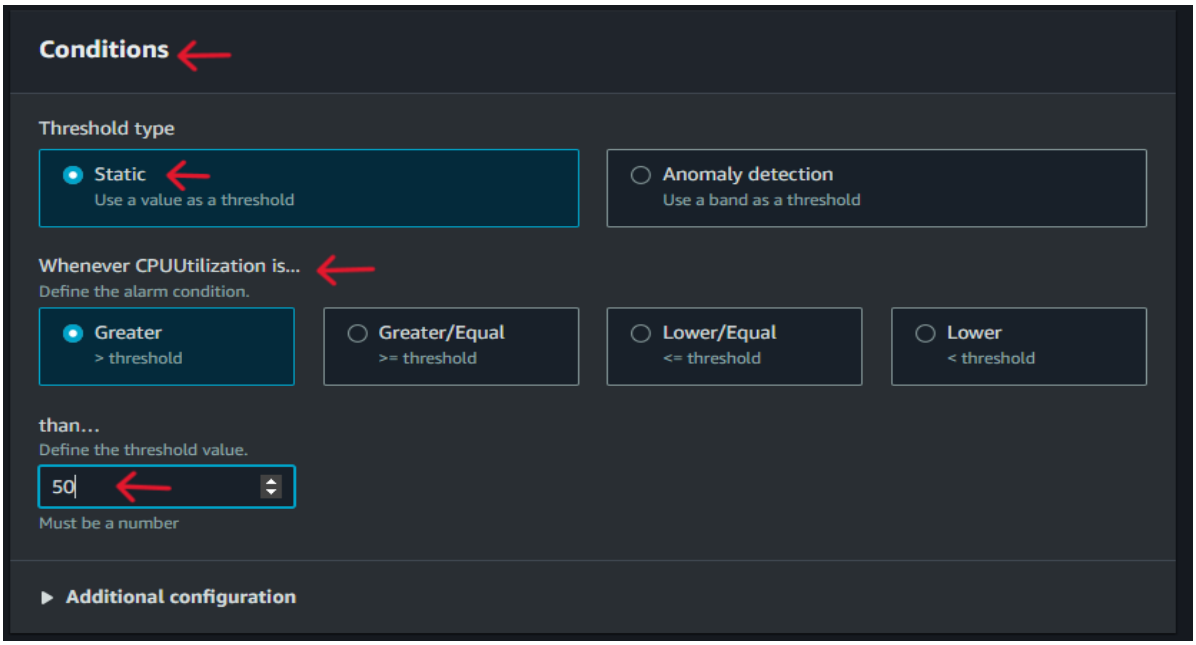


2) Click on Add Metrics, Enter Resource ID, Select Pre-Metrics, Pick Metrics CPU utilization (or more as needed), then click on Select Metrics Button.

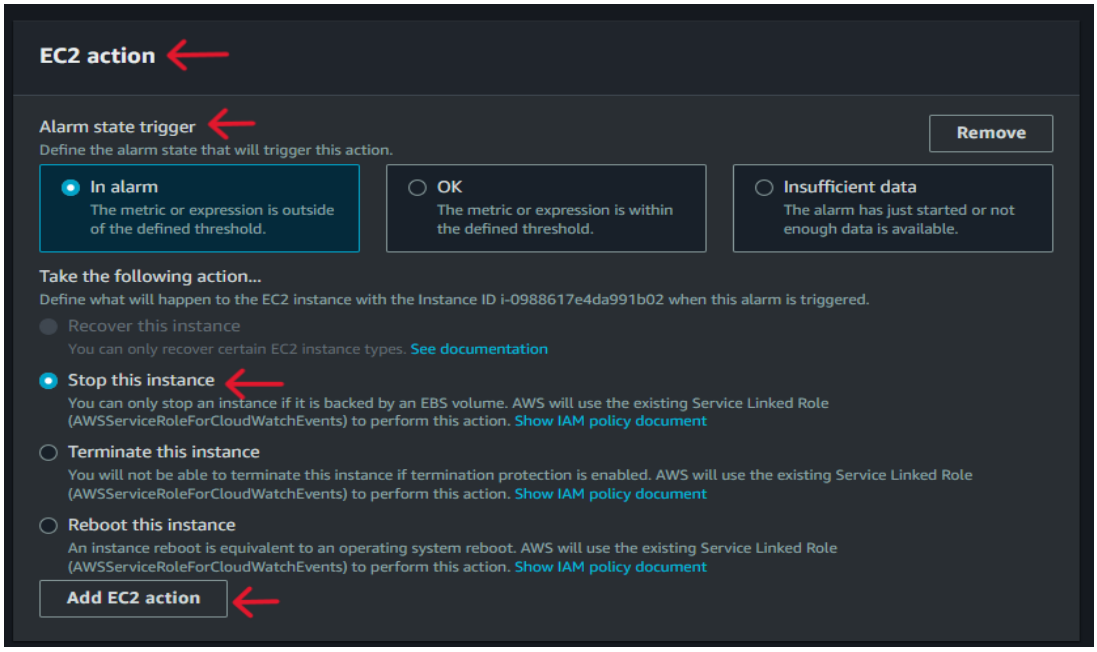


3) ***Specify metrics.***

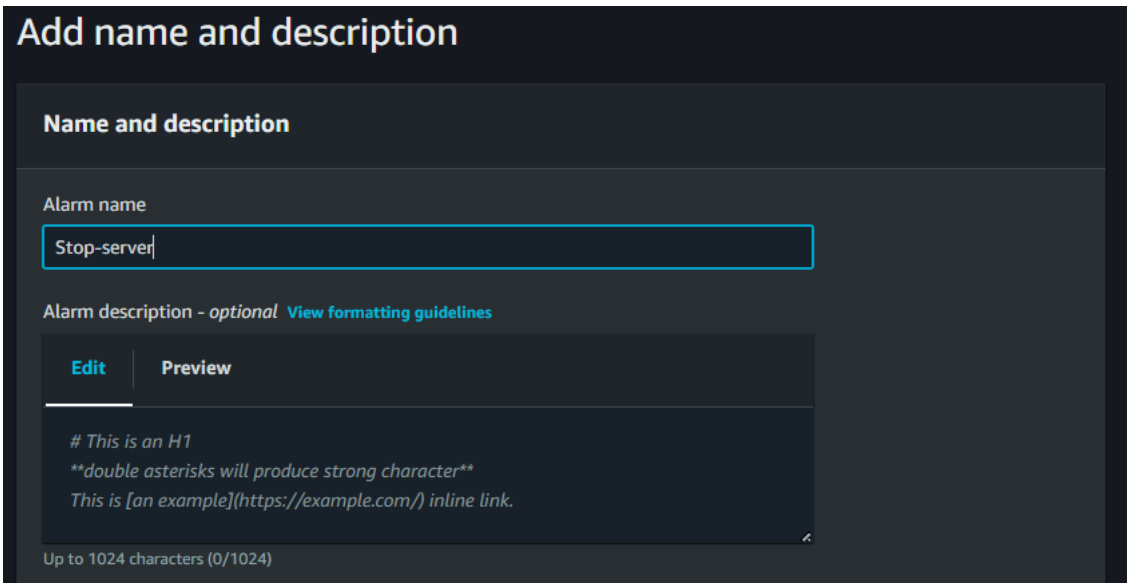
4) ***Select condition. (The threshold value is 50%.)***



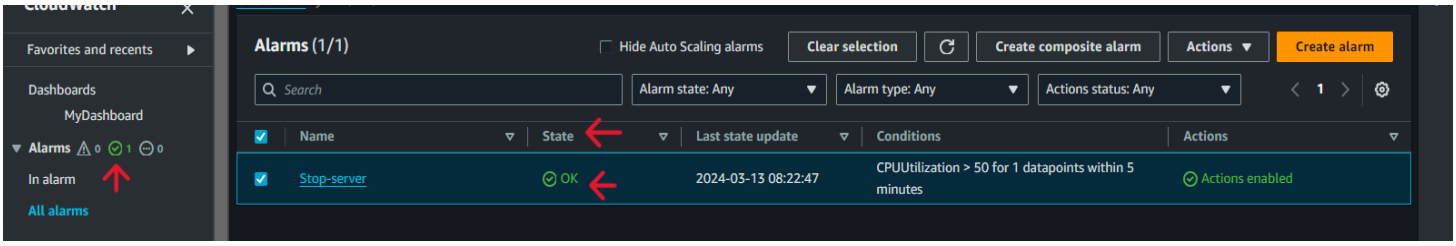
5) ***Configure what action is performed if the threshold is greater than 50%. Now we select Ec2.***



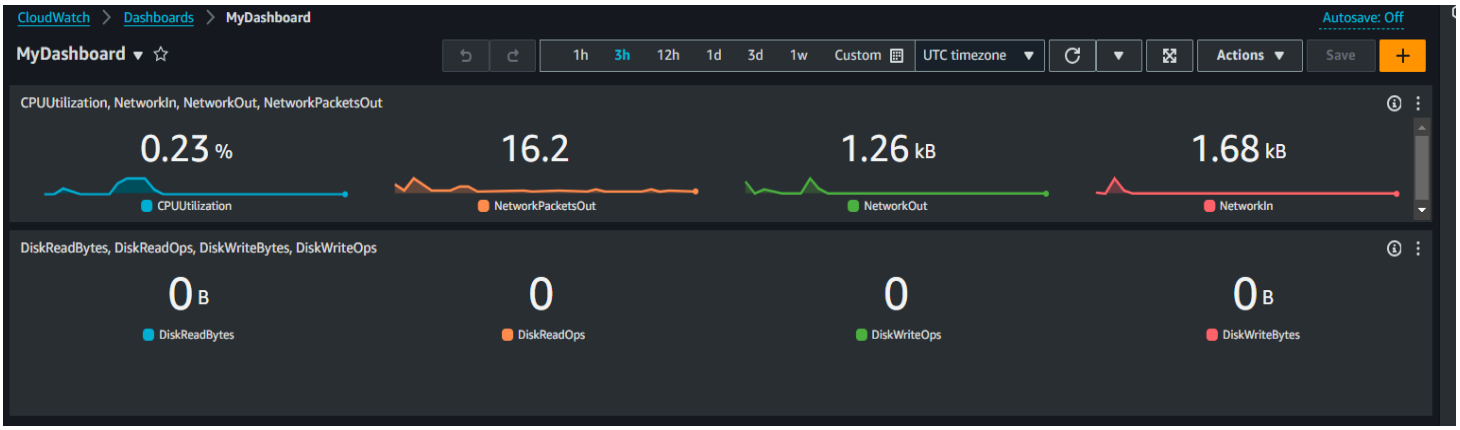
6) ***Name the alarm, and click on the next button.***



7) ***Preview the alarm, and click on the create alarm button, our alarm will created.***



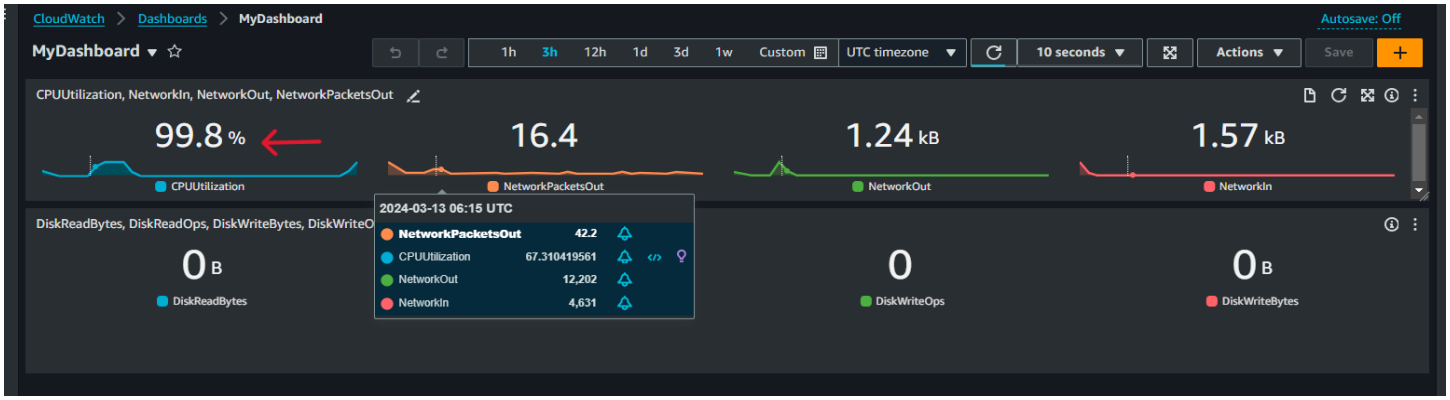
8) ***Now go the custom dashboard and check the Metrics of our Ec2 Instance.***



We can see CPU utilization is at optimal status. So here we will add artificial stress on the CPU to check that there is an alarm trigger and stop the Instance.

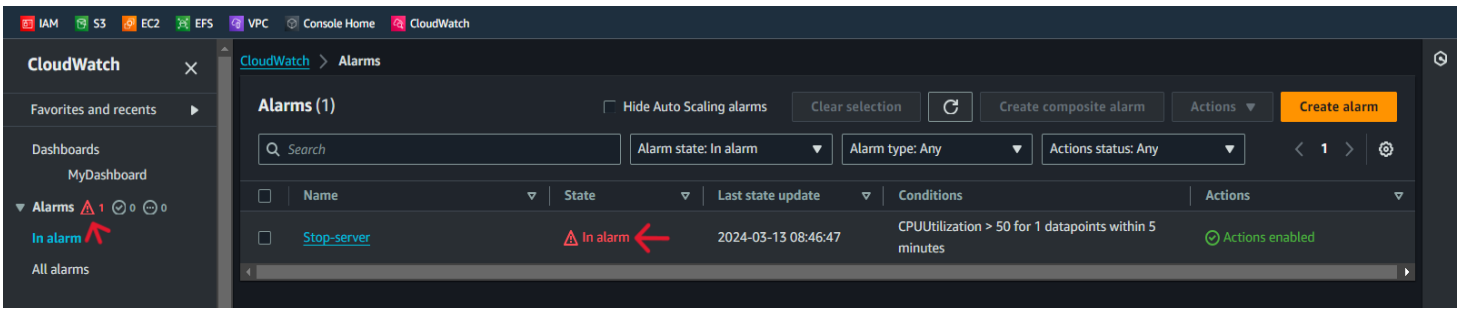
9) ***Install the Stress tool into the Instance and increase the CPU utilization artificially.***

10) ***Now check the CPU utilization.***



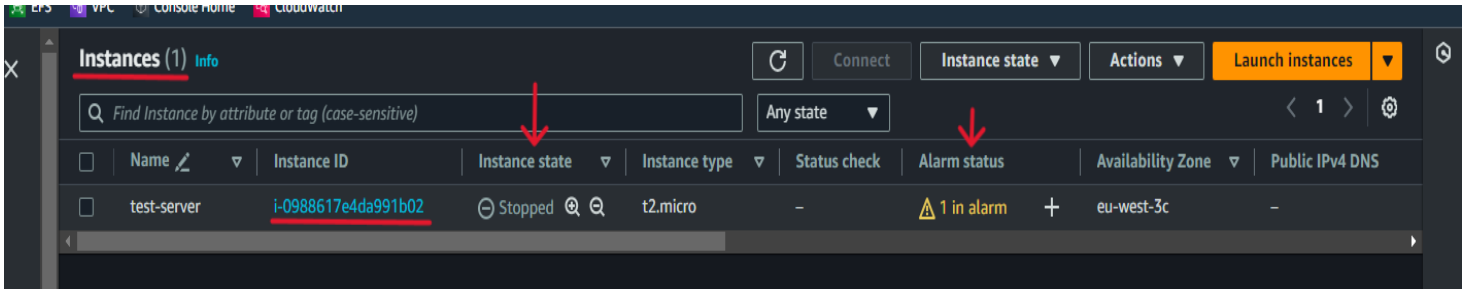
Is artificially increased

11) Now check the alarm status.



The Alarm Status is "In Alarm" which means the alarm has been triggered.

12) Now check the Instance status.



Here we can see our Instance is stopped which means our alarm function is effectively working.