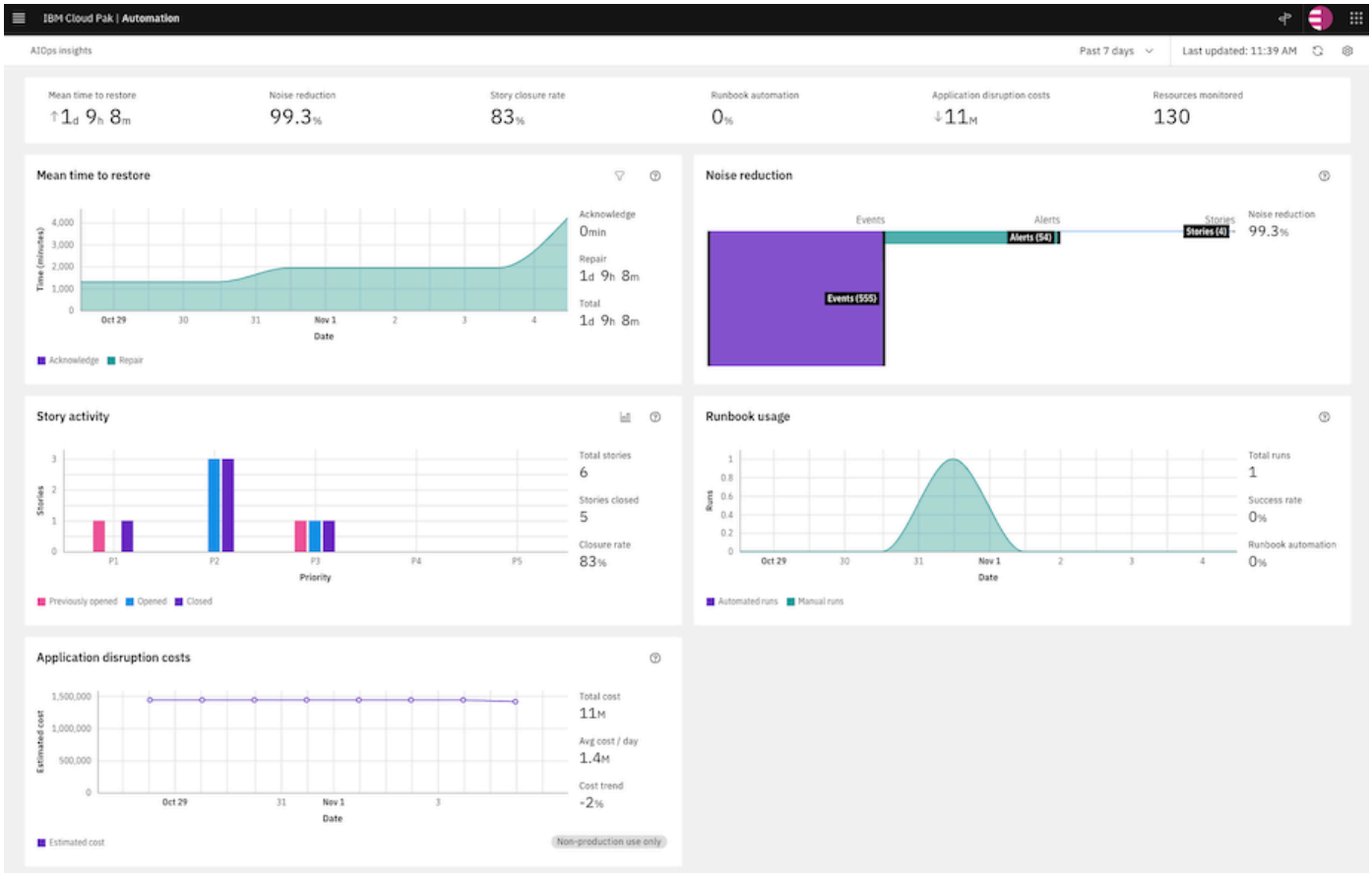
Cloud Pak for Watson AIOps

Sample Demo Script for the live demo environment



1. Introduction

This script is intended as a guide to demonstrate Cloud Pak for Watson AIOps using the live demo environment, running the Cloud Pak itself and the demo application. The script is presented in a few sections. You can utilize some or all sections depending upon your client's needs.

The script is intended to be used with live Cloud Pak for Watson AIOps 3.x demo environment that you can reserve via TechZone or install yourself.

In the demo script,

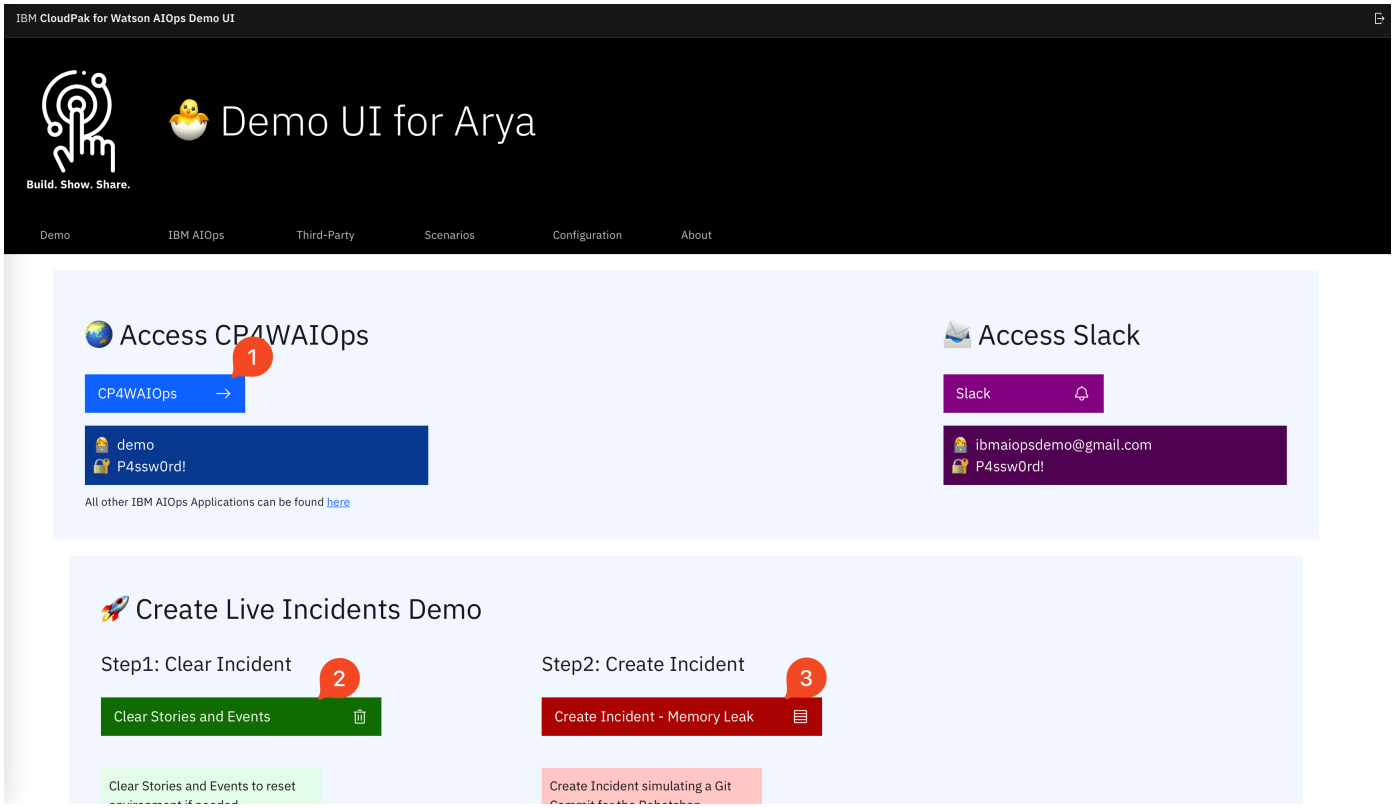
- "🚀 **Action**" denotes a setup step for the presenter.
- "📣 **Narration**" denotes what the presenter will say.
- "ℹ️ **Note**" denotes where the presenter may need to deviate from this demo script or add supplemental comments.

1.1 Key Terminology

You should be familiar with the following terminology when discussing Cloud Pak for Watson AIOps:

- **Application**: IBM Cloud Pak for Watson AIOps brings together the capability to group resources from different data types into applications. Clients can flexibly define an application to meet their business needs. With applications, you can obtain an integrated view of resources to understand inter-dependencies.
- **Event**: A point-in-time statement in Cloud Pak for Watson AIOps that tells us that something happened somewhere in a client's environment. It tells us what happened, where it happened, and when it happened. An event does not have to be exceptional or actionable, it can simply tell us something has happened.
- **Alert**: An alert in Cloud Pak for Watson AIOps represents an abnormal condition somewhere in an environment that requires resolution. It tells us what is happening, where it is happening, and when it started to happen. It may be informed by one or more events. It has a start time and end time.
- **Story**: A story in Cloud Pak for Watson AIOps represents an outage or reduction in service which is currently impacting customers and requires rapid remediation. It is created based on one or more trigger alerts that indicate the outage or reduction in service. Any alert of severity Major or Critical will act as a trigger alert. Other alerts that share the same cause may add context to the story.
- **Incident**: An incident in ServiceNow is an event of interruption disruption or degradation in normal service operation. An open incident in ServiceNow implies that the customer is impacted, or it represents the business risk.
- **Topology**: A topology is a representation of how constituent parts are interrelated. In Cloud Pak for Watson AIOps, an algorithm analyzes how the event nodes are proximate to each other and groups them into a topology-based correlation.

1.2 Navigating The Demo UI



The most important functionalities are:

1. **Open AIManager (login with the provided credentials)**
2. **Clear all existing Stories and Alerts**
3. **Create an Incident/Story**

> ℹ️ If you are asked to login to the Demo UI, please use the toekn/password `P4ssw0rd!`

> ⚠️ Before start, you should open the AIManager and check that there are no open stories and alerts pending. If there are some created few hours before (leftovers from somebody else not completing the demo) you can clean them up using AIManager Demo UI as shown below.

1.2 Demonstration scenario

1.2.1 Overview

This use case shows clients how IBM Cloud Pak for Watson AIOps proactively helps avoid application downtimes and incidents impacting end-users. You play the role of an SRE/Operations person who has received a Slack message indicating that the RobotShop application is not displaying customer ratings. This is an important feature of the RobotShop application since RobotShop is the main platform from which the fictional company sells its robots.

1.2.2 Use Case

The use case demonstrates how Cloud Pak for Watson AIOps can assist the SRE/Operations team as they identify, verify, and ultimately correct the issue. The demonstration shows integration with Instana, Turbonomic, ServiceNow, and Slack. Slack is the ChatOps environment used for working on this incident.

You will demonstrate the following major selling points around Cloud Pak for Watson AIOps:

1. **Pulls data from various IT platforms**: IBM Cloud Pak for Watson AIOps monitors incoming data feeds including logs, metrics, alerts, topologies, and tickets, highlighting potential problems across incoming data, based on trained machine learning models.
2. **Utilizes AI and natural language processing**: An insight layer connects the dots between structured and unstructured data, using AI and natural language processing technologies. This allows you to quickly understand the nature of the incident.
3. **Provides trust and transparency**: Using accurate and trustworthy recommendations, you can move forward with the diagnosis of IT system problems and the identification and prioritization of the best resolution path.
4. **Resolves rapidly**: Time and money are saved from out-of-the-box productivity that enables automation and utilizes pre-trained models. A “similar issue feature” from past incidents allows you to get services back online for customers and end-users.

1.3 Demonstration flow

1. Scenario introduction
2. Trigger problem situation [In the background]
3. Verify the status of the Robot Shop application.
4. Understanding and resolving the incident
    1. Login to AI Manager
    2. Open the Story
    3. Examining the Story
    4. Acknowledge the Story
    5. Similar Incidents
    6. Examine the Alerts
    7. Understand the Incident
    8. Examining the Topology
    9. [Optional] Topology in-depth
    10. Fixing the problem with runbook automation
    11. Resolve the Incident
5. Summary

2. Deliver the demo

2.1 Introduce the demo context

📣 **Narration**

Welcome to this demonstration of the Cloud Pak for Watson AIOps platform. In this demo, I am going to show you how Watson AIOps can help your operations team proactively identify, diagnose, and resolve incidents across mission-critical workloads.

You'll see how:

- Watson AIOps intelligently correlates multiple disparate sources of information such as logs, metrics, events, tickets and topology
- All of this information is condensed and presented in actionable alerts instead of large quantities of unrelated alerts
- You can resolve a problem within seconds to minutes of being notified using Watson AIOps' automation capabilities

During the demonstration, we will be using the sample application called RobotShop, which serves as a proxy for any type of app. The application is built on a microservices architecture, and the services are running on Kubernetes cluster.

> 🚀 **Action**
> Use demo introductory PowerPoint presentation, to illustrate the narration. Adapt your details on Slide 1 and 13

📣 **Narration**

Slide 2: Let' look at the environment that we have set up. Our sample application: "RobotShop" is running as a set of microservices in a Kubernetes cluster. Typically, the Operations team maintaining such application has a collection of tools through which they collect various data types.

Slide 3: Here we have several systems that are sending Events into WAIOPS (slide 3), like:

- GitHub
- Turbonomic
- Instana
- Selenium
- Falcon (Sysdig)

Those Events are being grouped into Alerts to massively reduce the number of signals that have to be treated. We usually observe a ratio of about 98-99% of reduction. This means that out of 20'000 events we get about 200-300 Alerts that can be further prioritised.

Slide 4: WAIOPS also ingests Logs from ElasticSearch (this could be Splunk or other Log Aggregators). The Log Anomaly detection is trained on a well running system and is able to detect anomalies and outliers. If an Anomaly is detected it will be grouped with the other Events.

Slide 5: WAIOPS also ingests Metrics from Instana (this could be Dynatrace, NewRelic or others). The Metric Anomaly detection is trained on a well running system and creates dynamic baselines. Through different algorithms it is able to detect anomalies and outliers. If an Anomaly is detected it will also be grouped with the other Events.

Slide 6: Alerts that are relevant for the same Incident are packaged into a so called Story. The Story will be enriched and updated with information as it gets available.

Slide 7: One example is the Topology information. Not only will WAIOPS tell me that I have a problem and present all relevant Events but it will also tell me where in the system topology the problem is situated.

Slide 8: Furthermore the Story is enriched with past resolution information coming from ServiceNow tickets. I'll explain this more in detail during the demo.

Slide 9: The Stories can either be examined in the WAIOPS web interface or can be pushed to Slack or Teams if your teams are using a ChatOps approach.

Slide 10: If Operations or SREs have created Runbooks, WAIOPS can automatically trigger a Runbook to mitigate the problem.

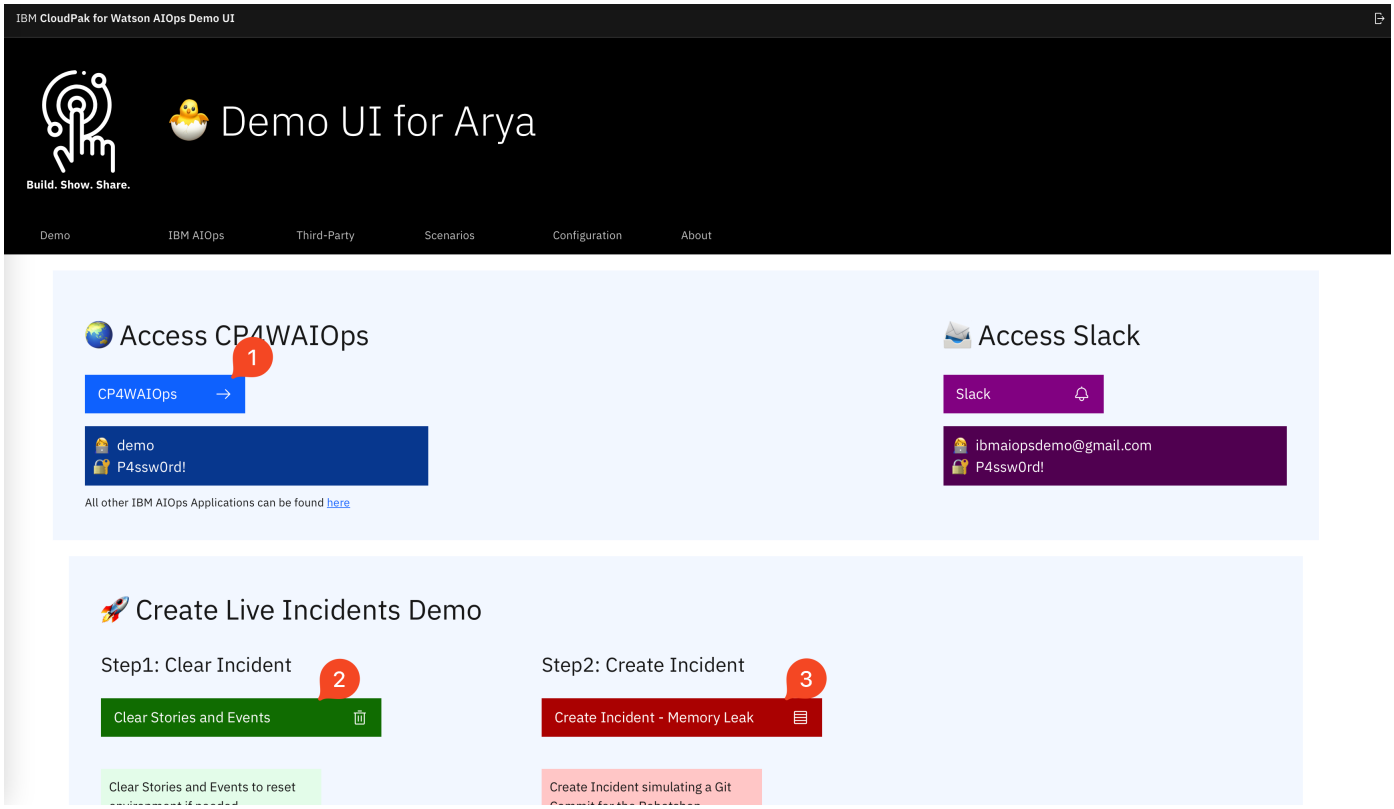
ℹ️ **Note**: We are NOT using Slack in this demo.

📣 **Narration**

Now let's start the demo.

2.2 Trigger the incident

❗ **Note**: The following step does not have to be shown to the client – you may perform the action in the background if possible.



🚀 **Action**

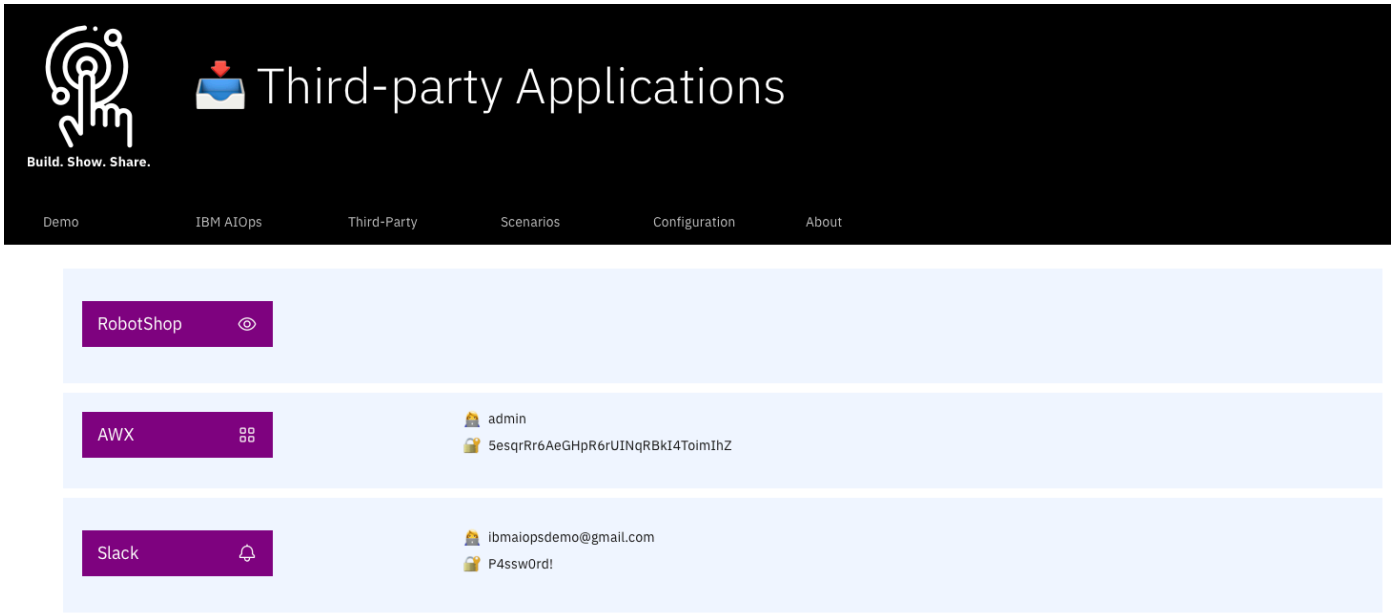
Open AIManager **Demo** UI, and trigger the incident

- Point your browser to the AIManager Demo UI,
- Login with the token “P4ssw0rd!” and
- Trigger the incident `(3)` you would like to use in your demo.

This action injects the stream of simulated events into the system, which replicates what could happen in a real life situation.

2.3 Verify the status of the Robot Shop application

2.3.1 Show the Application



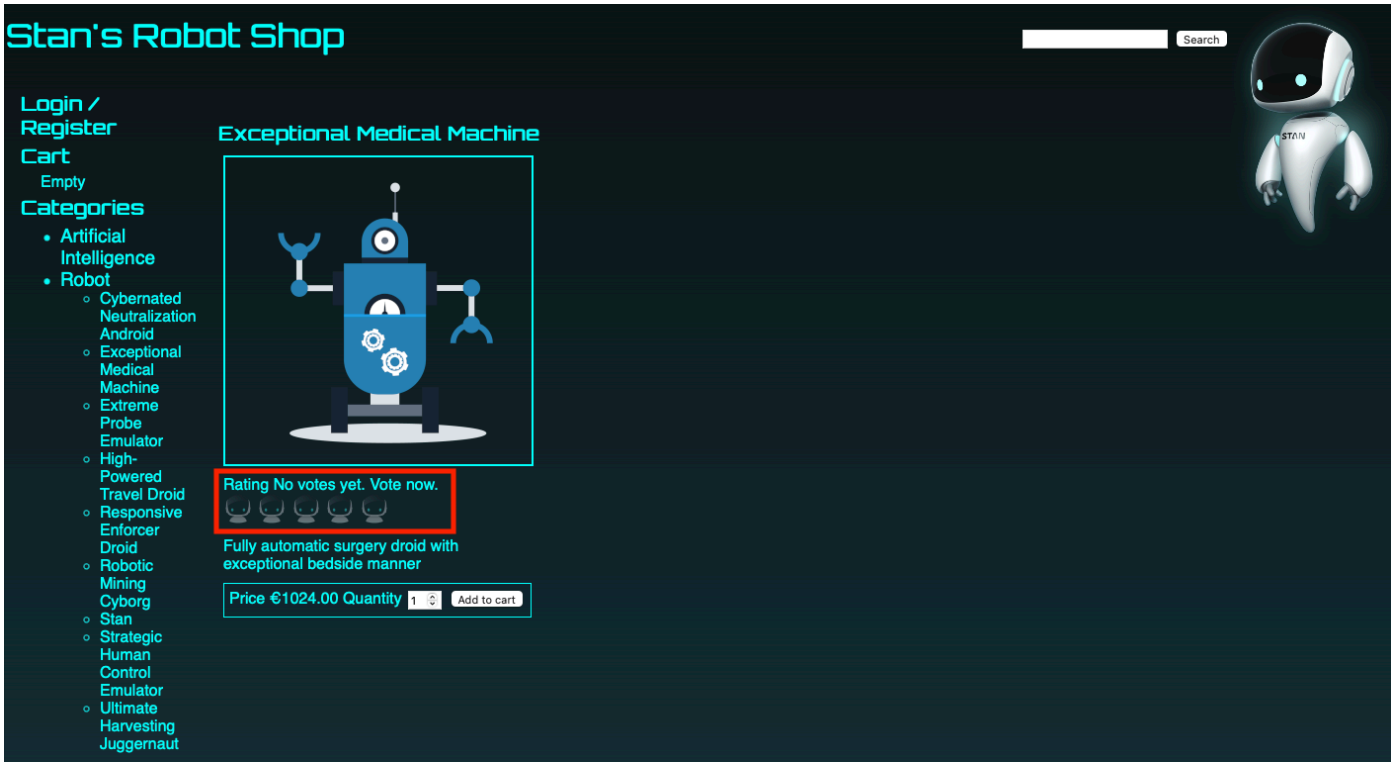
> 🚀 **Action**
> Open the RobotShop application
>
> The Link can be found in the **Demo UI** under **Third-Party**. Play with the application UI.

📣 **Narration**

In this demo I am the application SRE (Site Reliability Engineer) responsible for an e-commerce website called RobotShop, an online store operated by my company. In the middle of the day (when clients make most of the purchases) I received a slack message on my mobile, alerting me that there is some problem with the site.

Let's verify what's going on with the RobotShop site. The application is up but displays an error that it cannot get any ratings.

2.3.2 Show ratings not working



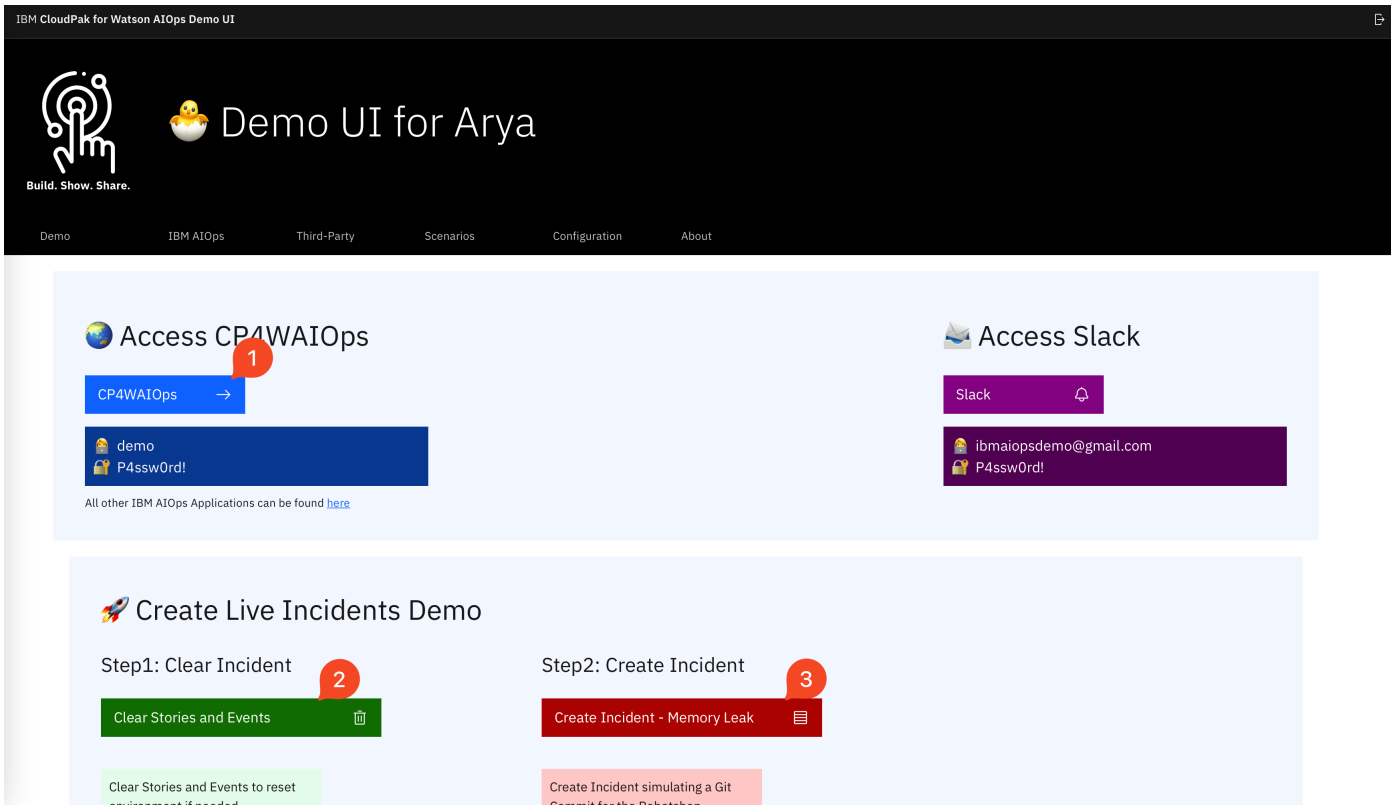
> 🚀 **Action**
> Open any robot details to show that there are no ratings displayed.

📣 Narration

I know that there are many ratings for each of the products that we sell, so when none are displayed, it means that there is a likely problem with `Ratings` service withing application that may heavily impact client's purchasing decisions, as well as may be a sign of a wider outage.

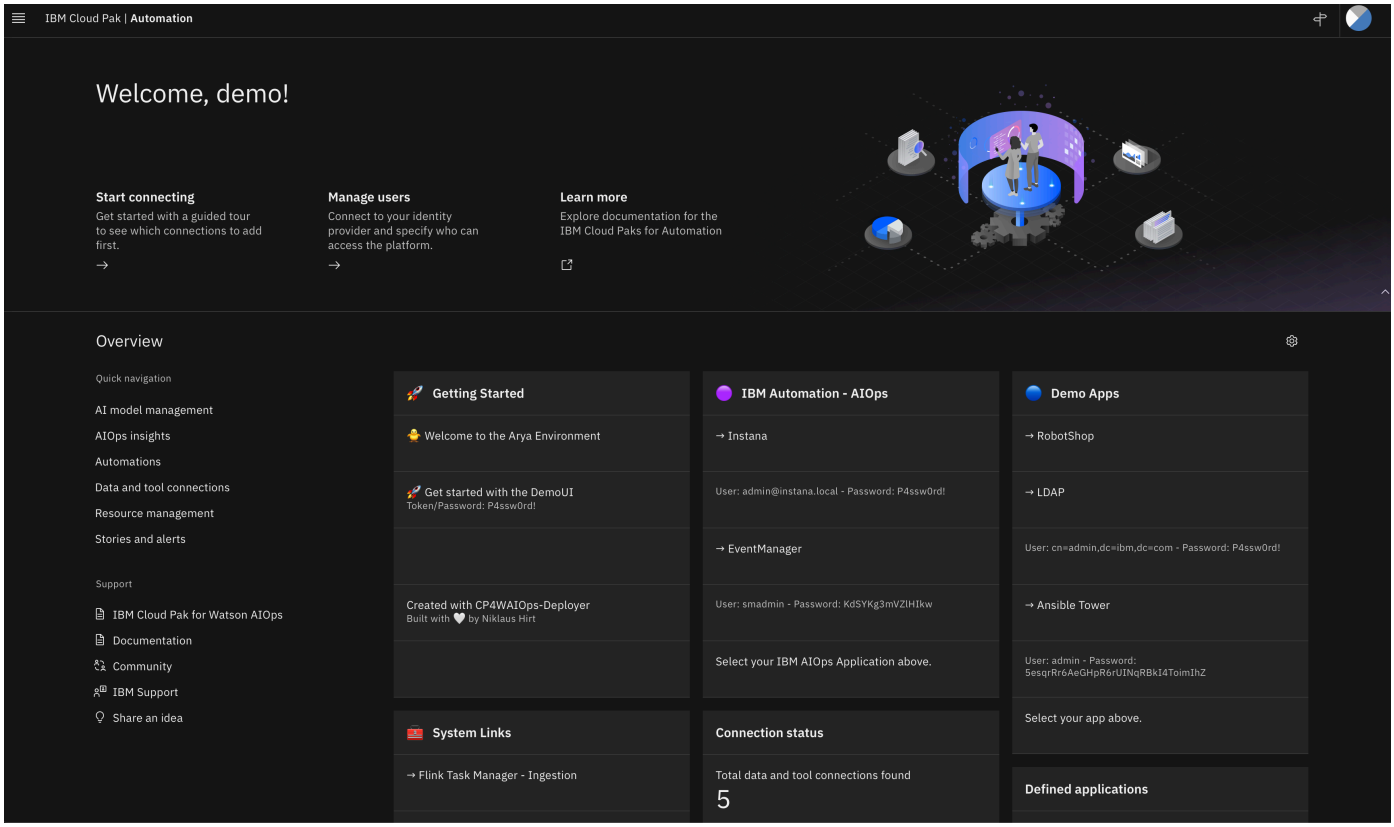
2.4 Understanding the incident

2.4.1 Login to AI Manager



Action

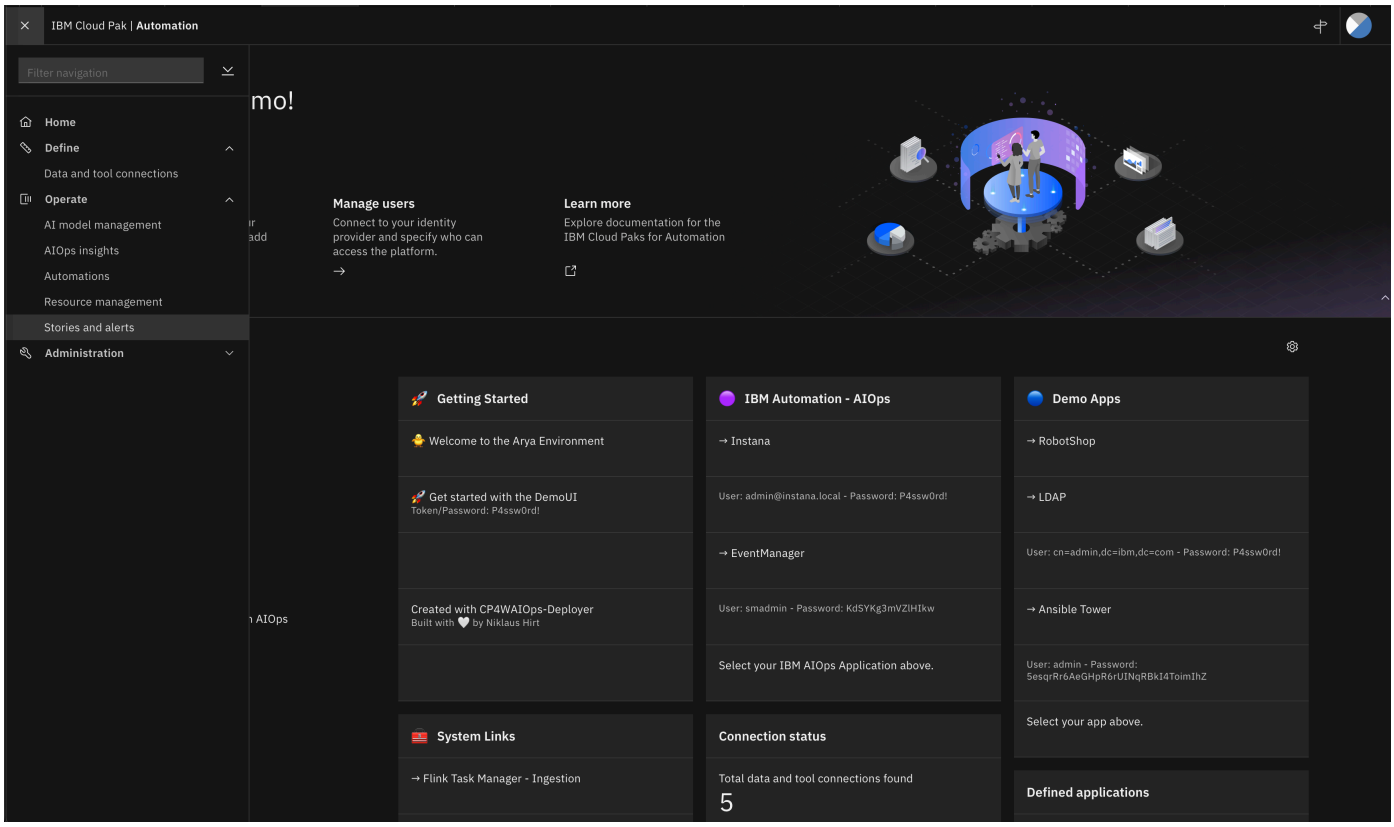
In the Demo UI, click **AI MAnager** `(1)`



📣 Narration

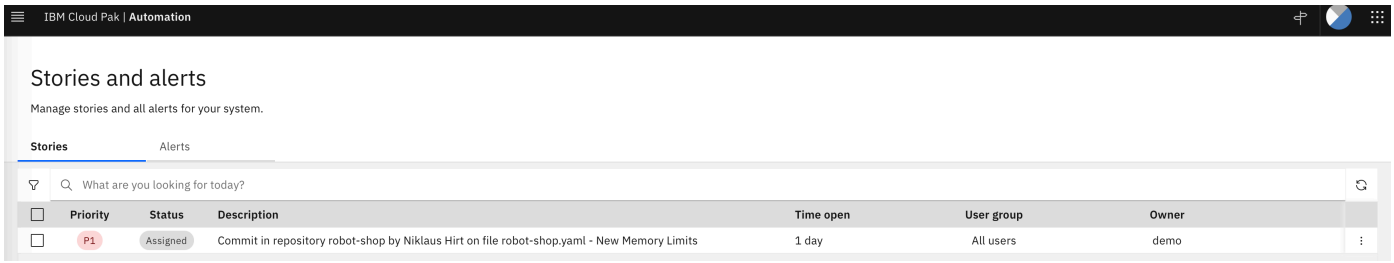
Let’s take a closer look at the incident that has been created in Watson AIOps.

2.4.2 Open the Story



Action

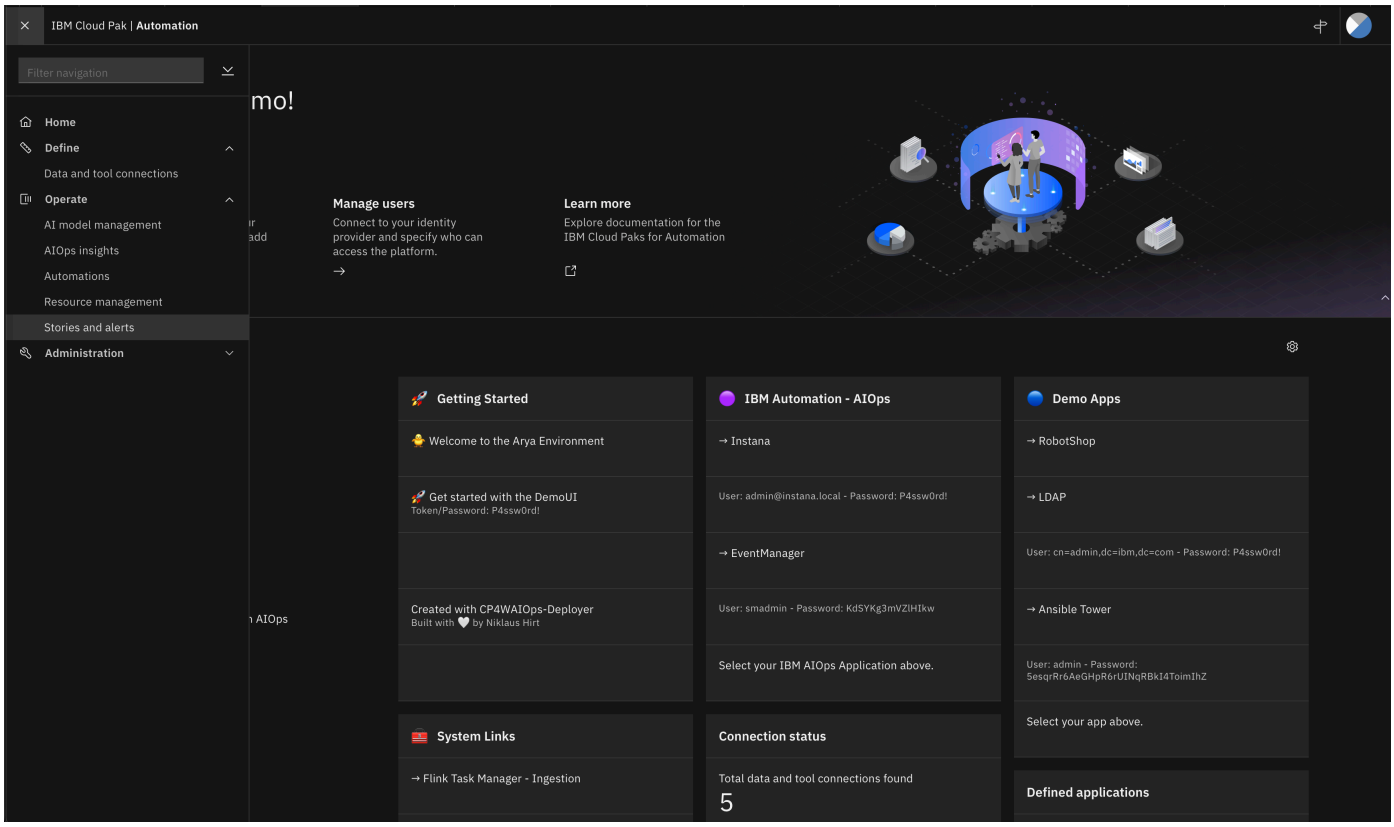
Click the "hamburger menu" on the upper left. Click **Stories and alerts**



Narration

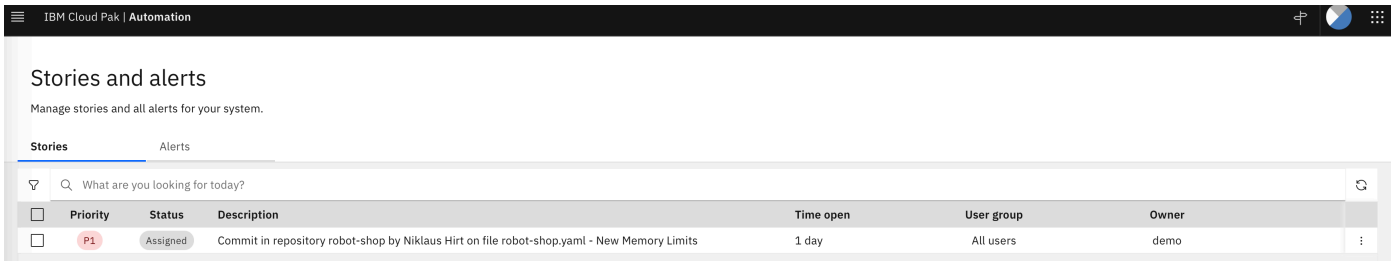
We can see that the simulation has created a **Story**. The **Story** includes grouped information related to the incident at hand. It equates to a classic War Room that are usually put in place in case of an outage.
The **Story** contains related log anomalies, topology, similar incidents, recommended actions based on past trouble tickets, relevant events, runbooks, and more.

2.4.3 Examining the Story



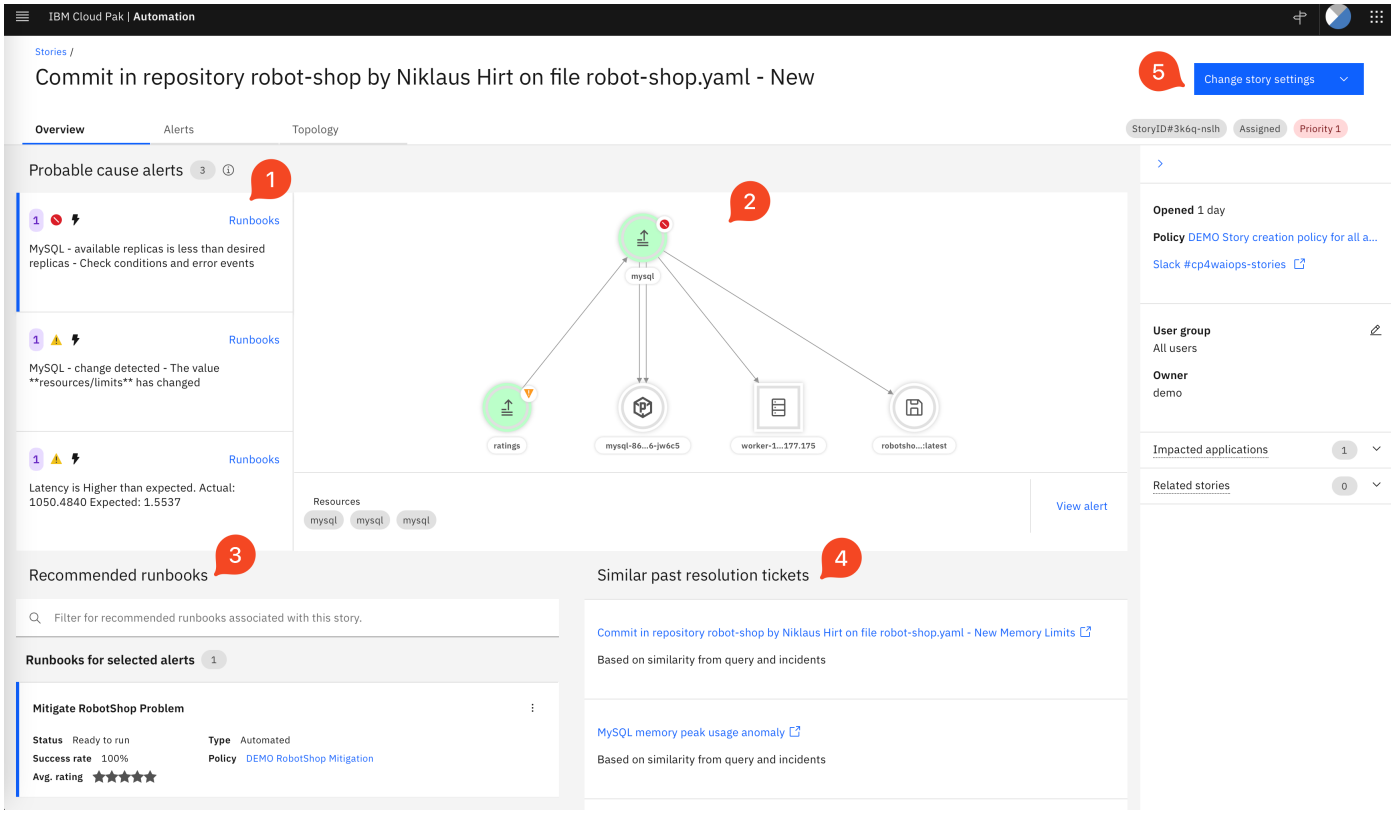
Action

Click the "hamburger menu" on the upper left. Click **Stories and alerts**



Narration

Now let's have a look at the **Story**.



As I said before, the Story regroups all relevant information concerning the incident at hand that have been identified by Watson AIOps.

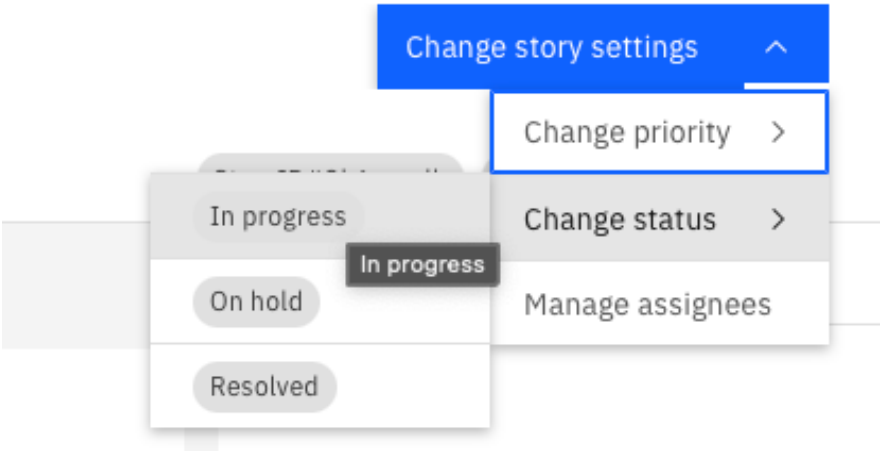
1. A list of Alerts that have been identified by Watson AIOps to be the most probable cause
2. The localization of the problem related to the Topology
3. The suggested Runbooks to automatically mitigate the incident
4. Similar Incidents that resemble the incident at hand
5. Status of the Story - here I can change the status and priority of the story

2.4.4 Acknowledge the Story

🚀 **Action**
Click on **Change Story Settings.**

Select **Change Status.**

Click on **In progress**

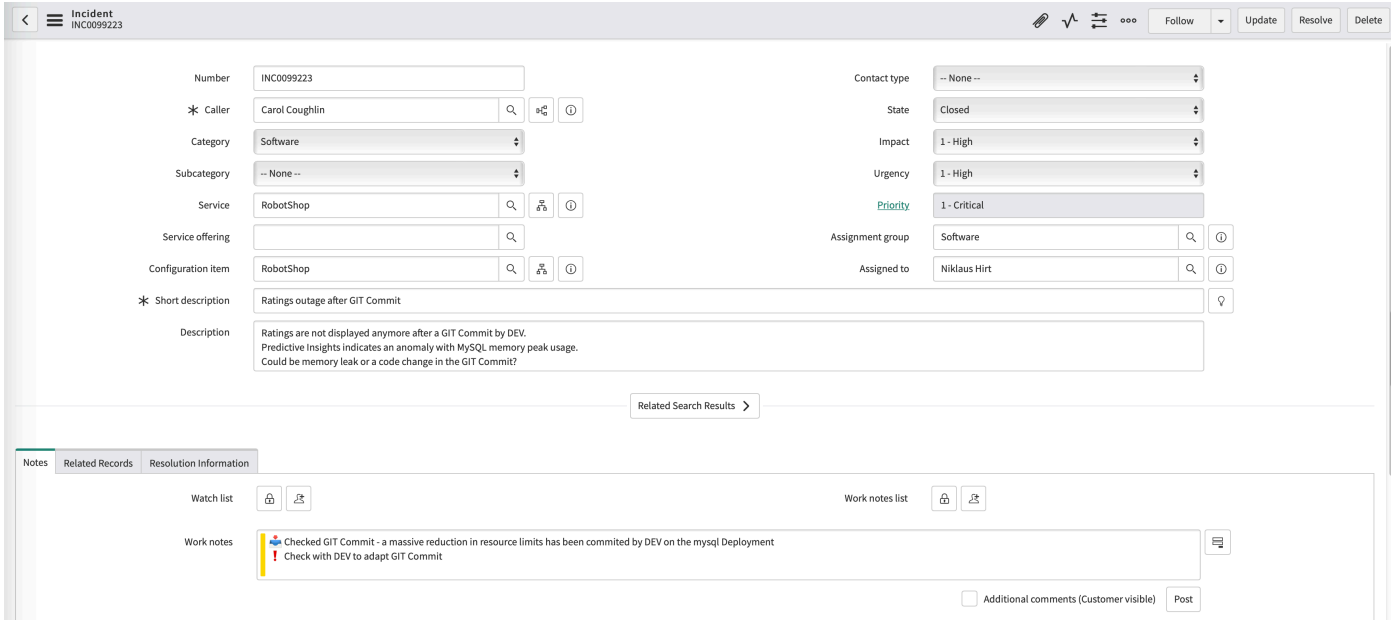


📣 **Narration**

First and before I continue examining the Story I want to let my colleagues know that I'm working on the incident. So let me set it to In Progress.

2.4.5 Similar Incidents

> 🚀 **Action**
> Click the first similar resolution ticket



📣 Narration

Most large organizations use IT Service Management tools to govern processes around IT. Our organization is using ServiceNow for that purpose. Past incidents with resolution information are ingested and analysed by Watson AIOps.

The IBM Cloud Pak for Watson AIOps trains on exisitng tickets and it extracts the steps used to fix previous incidents (if documented) and recommend resolutions using natural language processing. This AI model helps you discover historical incidents to aid in the remediation of current problems.

So for the **Story**, your team is presented with the top-ranked similar incidents from the past. These relevant similar incidents help speed up incident resolution even if the I don't have access to ServiceNow. Without these features, your team must manually search for past incidents and resolutions, which is time-consuming.

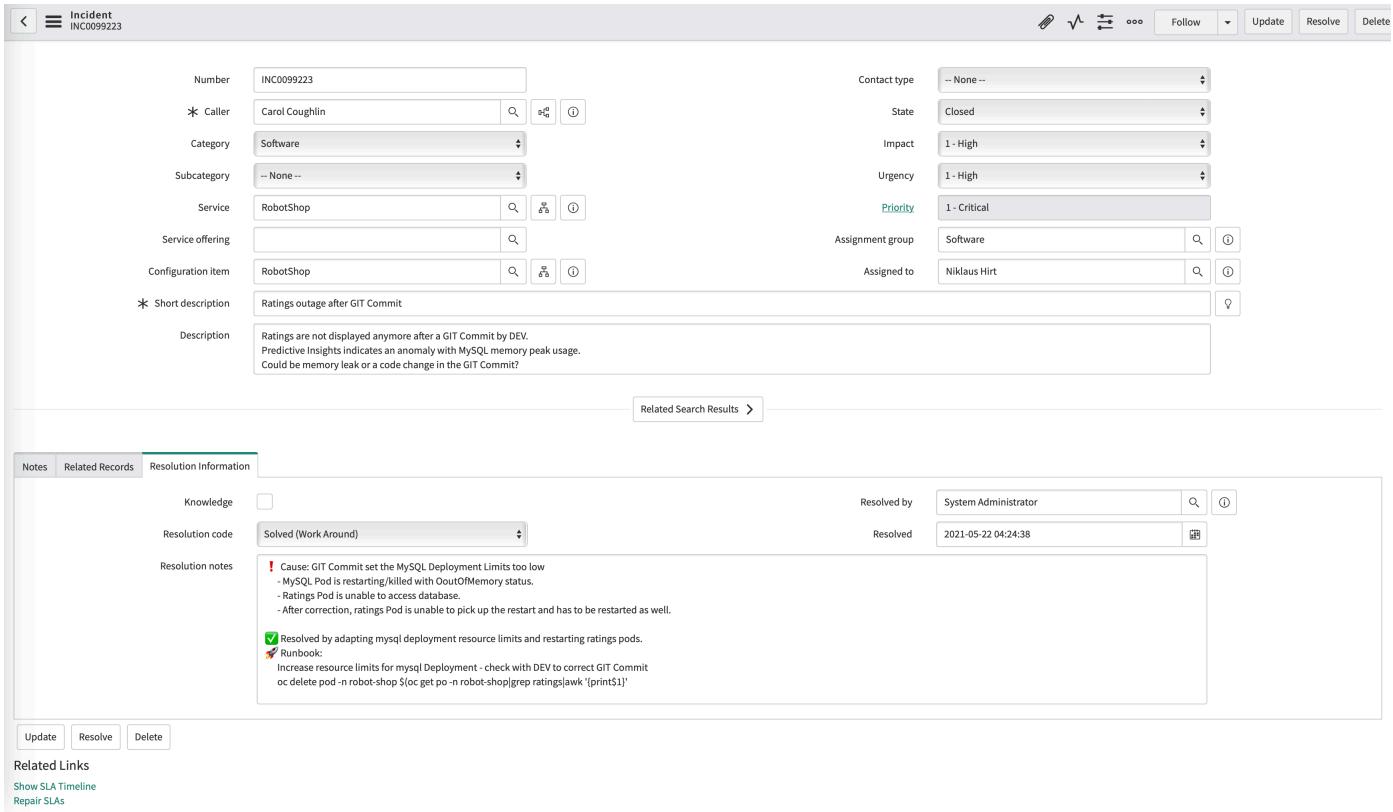
In this particular example I can see that the problem was related to a GIT Commit that massivly reduced the resource limits has been commited by DEV on the mysql Deployment.

Let me check how the problem was resolved for this incident.

❗ **Note**: In the Robot Shop demo scenario, the integration with ServiceNow is simulated with the static content.

Resolution Information

> 🚀 **Action**
> Click on the **Resolution Information** Tab

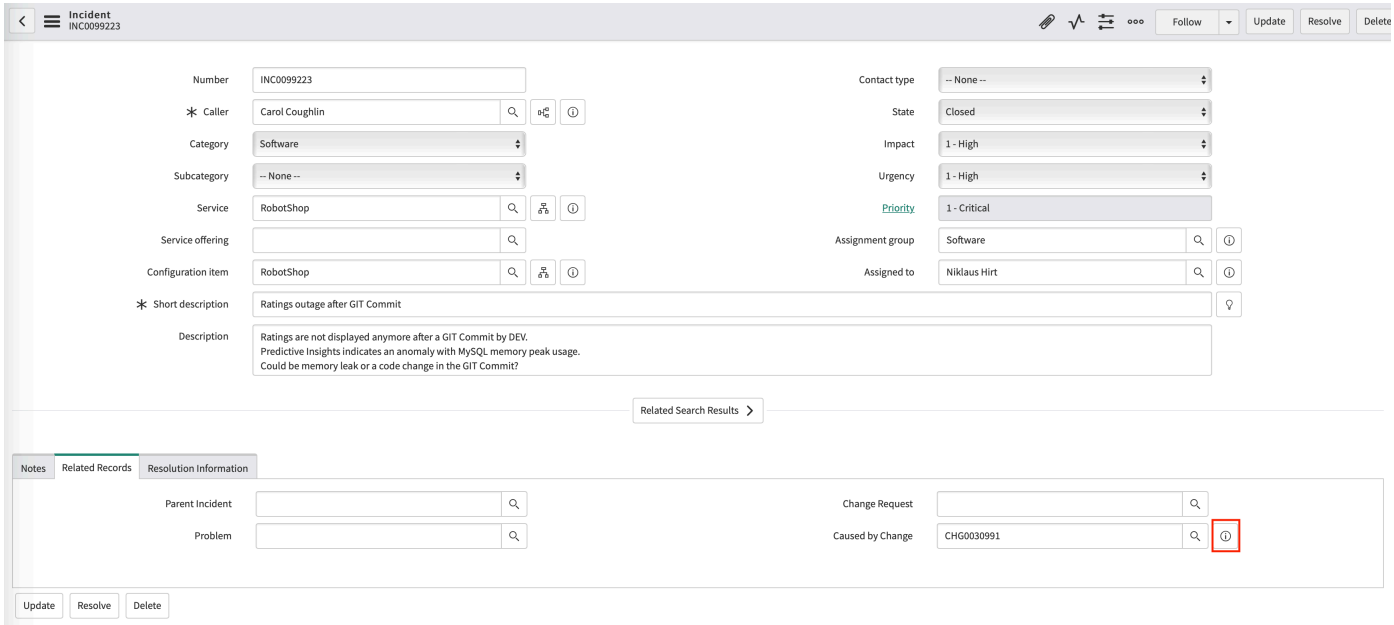


📣 Narration

It seems that it was resolved by changing the mysql deployment and a Runbook had been created to mitigate the problem.

To finish up, I will check if the incident was related to an official change.

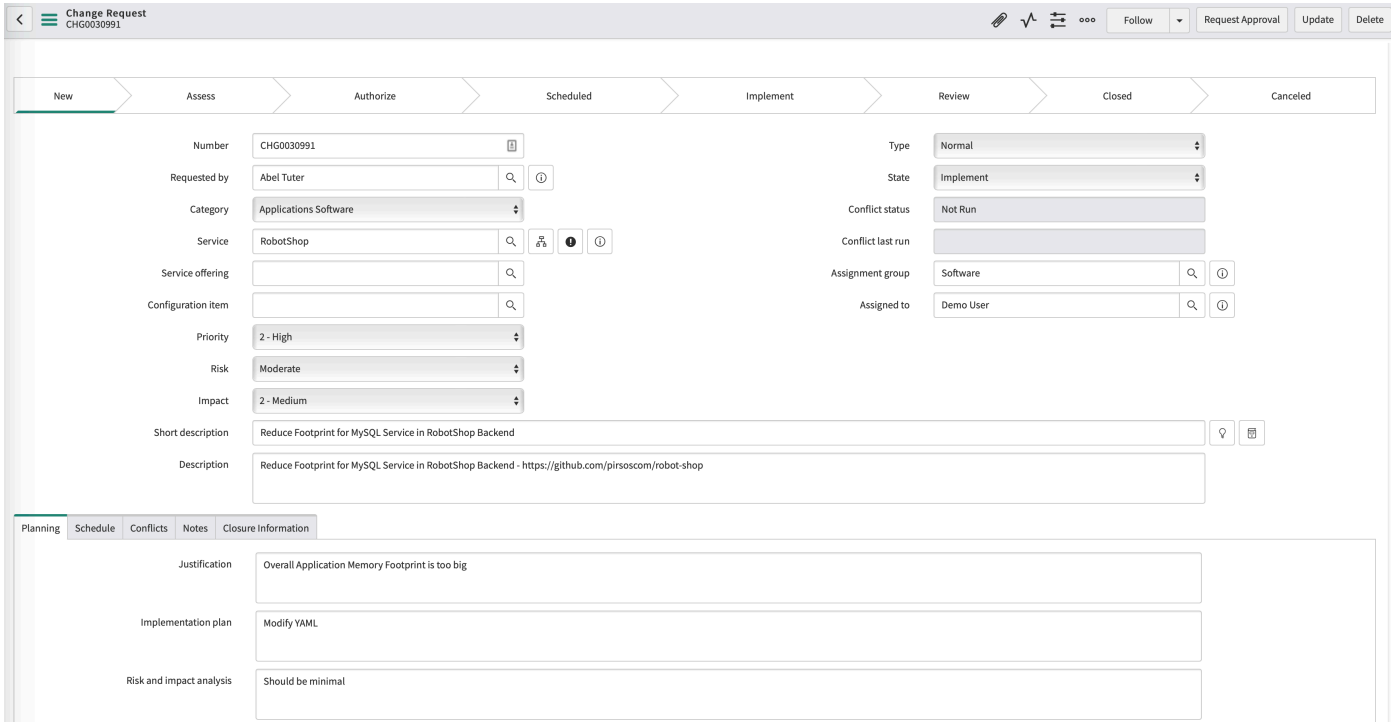
Examine the Change



🚀 Action

Click on the **Related Records** Tab

Click on the **i** Button next to **Caused by Change**



📣 Narration

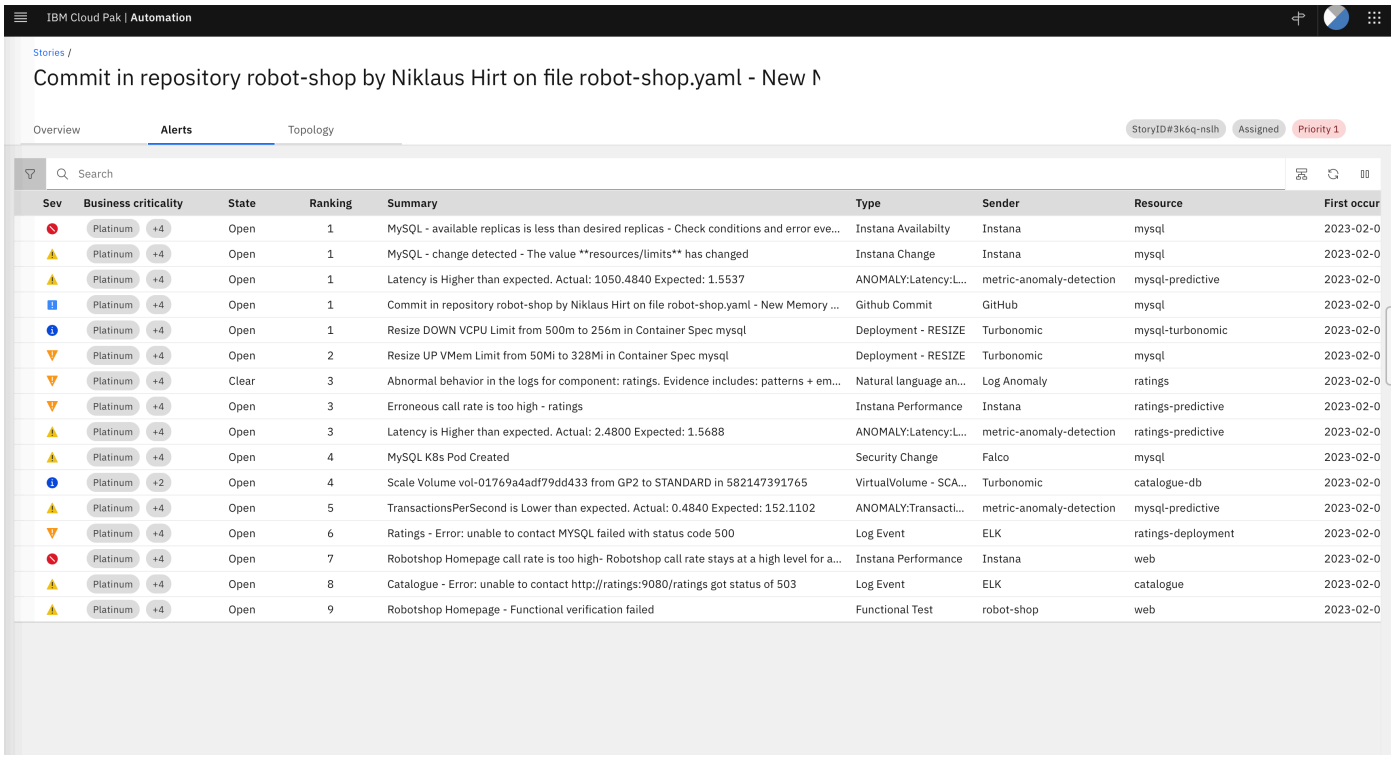
Ok, so now I can see that the problem is related to a Change that aims to reduce the footprint of the mysql database.

As it's still ongoing, chances are high, that the development team recreated a similar problem.

Obviously, in real life I would now start the Runbook to see if it resolves the problem.
But for the sake of the demo, let's dig a little deeper first.

2.4.6 Examine the Alerts

> 🚀 **Action**
> Close the ServiceNow page and click the **Alerts** Tab.



📣 Narration

Notice, that alerts are not sorted by severity, but the AI engine ranked them by relevance. The ones that are likely related to the root cause are at the top. Let's look at the first row for some more details.

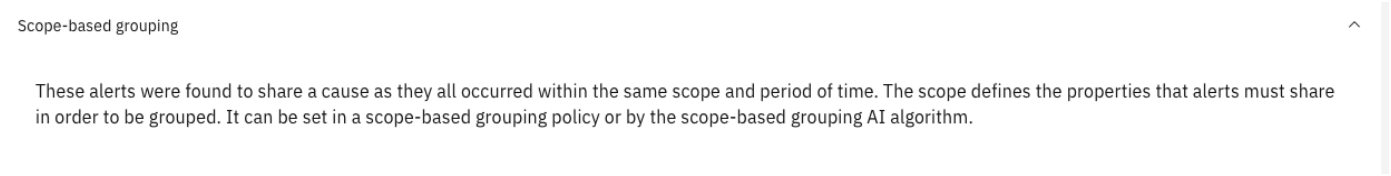
> 🚀 **Action**
> Click on the first Alert in the list.

📣 Narration

In the **Alert details,** you can see different types of groupings explaining why the specific alert was added to the story.

Scope based grouping

> 🚀 **Action**
> Click **Scope-based grouping**.

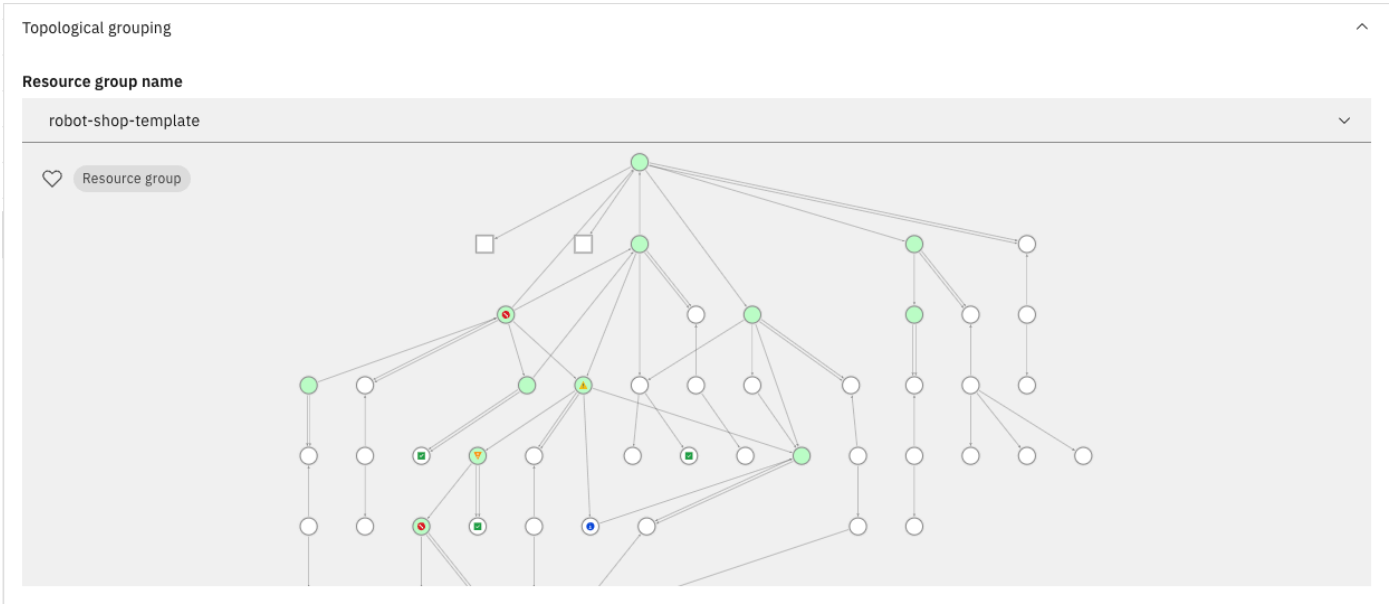


📣 Narration

Some alerts were added to the story because they occurred on the same resource within a short period (default is 15 minutes)

Topological grouping

> 🚀 **Action**
> Click **Topological grouping**.

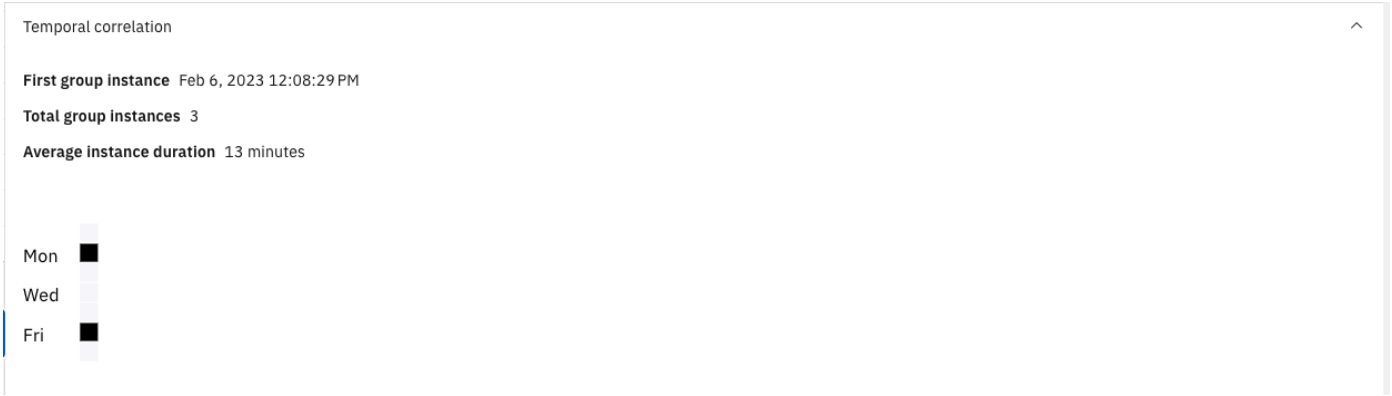


📣 Narration

Other alerts were grouped because they occurred on the logically or physically related resources. This correlation is using the application topology service that stitches topology information from different sources.

Temporal grouping

> 🚀 **Action**
> Click **Temporal correlation**.



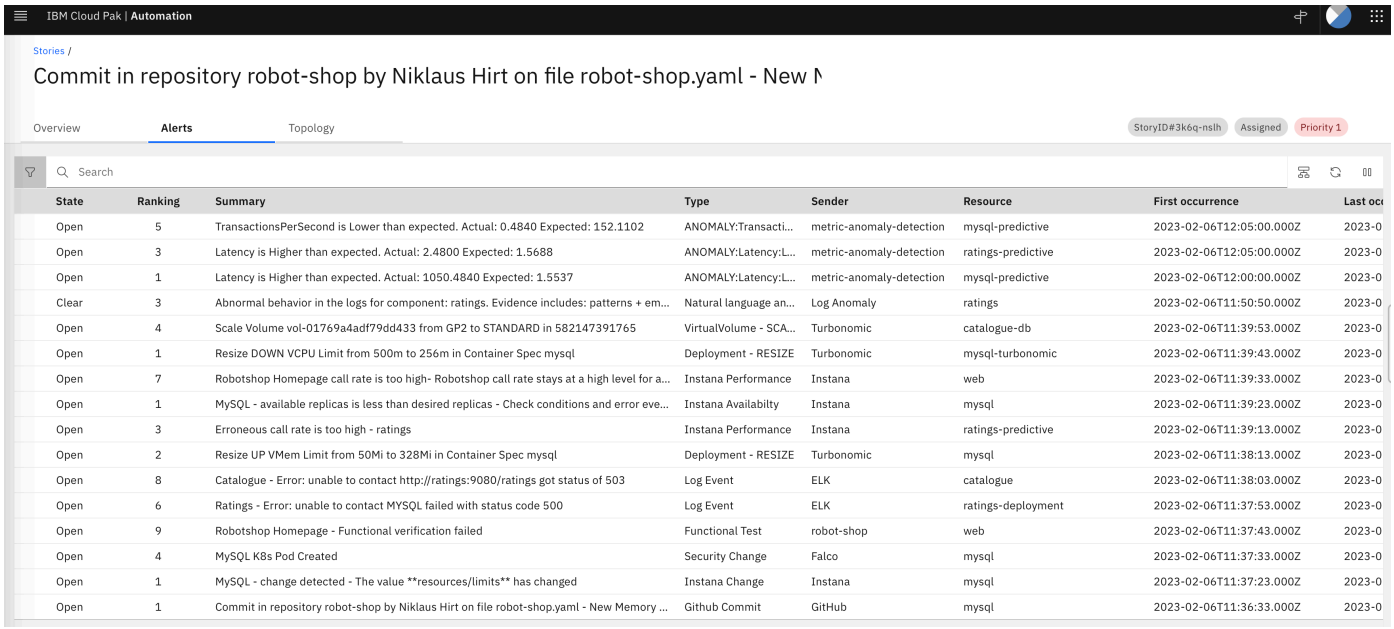
📣 Narration

Finally, the temporal correlation adds to the story events that previously, in history, are known to occur close to each other in the short time window. What is most important here is the fact that all these correlations happen automatically – there is no need to define any rules or program anything. In highly dynamic and distributed cloud-native applications this is a huge advantage that saves a lot of time and effort.

> 🚀 **Action**
> **Close** the Alert details window.

2.4.7 Incident timeline

> 🚀 **Action**
> Click twice on the **Last occurence** Header.
>
> ***Result**: The "Commit in repository robot-shop by Niklaus Hirt on file robot-shop.yaml" should be at the bottom*



State	Ranking	Summary	Type	Sender	Resource	First occurrence	Last oc
Open	5	TransactionsPerSecond is Lower than expected. Actual: 0.4840 Expected: 152.1102	ANOMALY:Transacti...	metric-anomaly-detection	mysql-predictive	2023-02-06T12:05:00.000Z	2023-0
Open	3	Latency is Higher than expected. Actual: 2.4800 Expected: 1.5688	ANOMALY:Latency:L...	metric-anomaly-detection	ratings-predictive	2023-02-06T12:05:00.000Z	2023-0
Open	1	Latency is Higher than expected. Actual: 1050.4840 Expected: 1.5537	ANOMALY:Latency:L...	metric-anomaly-detection	mysql-predictive	2023-02-06T12:00:00.000Z	2023-0
Clear	3	Abnormal behavior in the logs for component: ratings. Evidence includes: patterns + em...	Natural language an...	Log Anomaly	ratings	2023-02-06T11:50:50.000Z	2023-0
Open	4	Scale Volume vol-01769a4adf79dd433 from GP2 to STANDARD in 582147391765	VirtualVolume - SCA...	Turbonomic	catalogue-db	2023-02-06T11:39:53.000Z	2023-0
Open	1	Resize DOWN VCPU Limit from 500m to 256m in Container Spec mysql	Deployment - RESIZE	Turbonomic	mysql-turbonomic	2023-02-06T11:39:43.000Z	2023-0
Open	7	Robotshop Homepage call rate is too high- Robotshop call rate stays at a high level for a...	Instana Performance	Instana	web	2023-02-06T11:39:33.000Z	2023-0
Open	1	MySQL - available replicas is less than desired replicas - Check conditions and error eve...	Instana Availabilty	Instana	mysql	2023-02-06T11:39:23.000Z	2023-0
Open	3	Erroneous call rate is too high - ratings	Instana Performance	Instana	ratings-predictive	2023-02-06T11:39:13.000Z	2023-0
Open	2	Resize UP VMem Limit from 50Mi to 328Mi in Container Spec mysql	Deployment - RESIZE	Turbonomic	mysql	2023-02-06T11:38:13.000Z	2023-0
Open	8	Catalogue - Error: unable to contact http://ratings:9080/ratings got status of 503	Log Event	ELK	catalogue	2023-02-06T11:38:03.000Z	2023-0
Open	6	Ratings - Error: unable to contact MYSQL failed with status code 500	Log Event	ELK	ratings-deployment	2023-02-06T11:37:53.000Z	2023-0
Open	9	Robotshop Homepage - Functional verification failed	Functional Test	robot-shop	web	2023-02-06T11:37:43.000Z	2023-0
Open	4	MySQL K8s Pod Created	Security Change	Falco	mysql	2023-02-06T11:37:33.000Z	2023-0
Open	1	MySQL - change detected - The value **resources/limits** has changed	Instana Change	Instana	mysql	2023-02-06T11:37:23.000Z	2023-0
Open	1	Commit in repository robot-shop by Niklaus Hirt on file robot-shop.yaml - New Memory ...	Github Commit	GitHub	mysql	2023-02-06T11:36:33.000Z	2023-0

📣 Narration

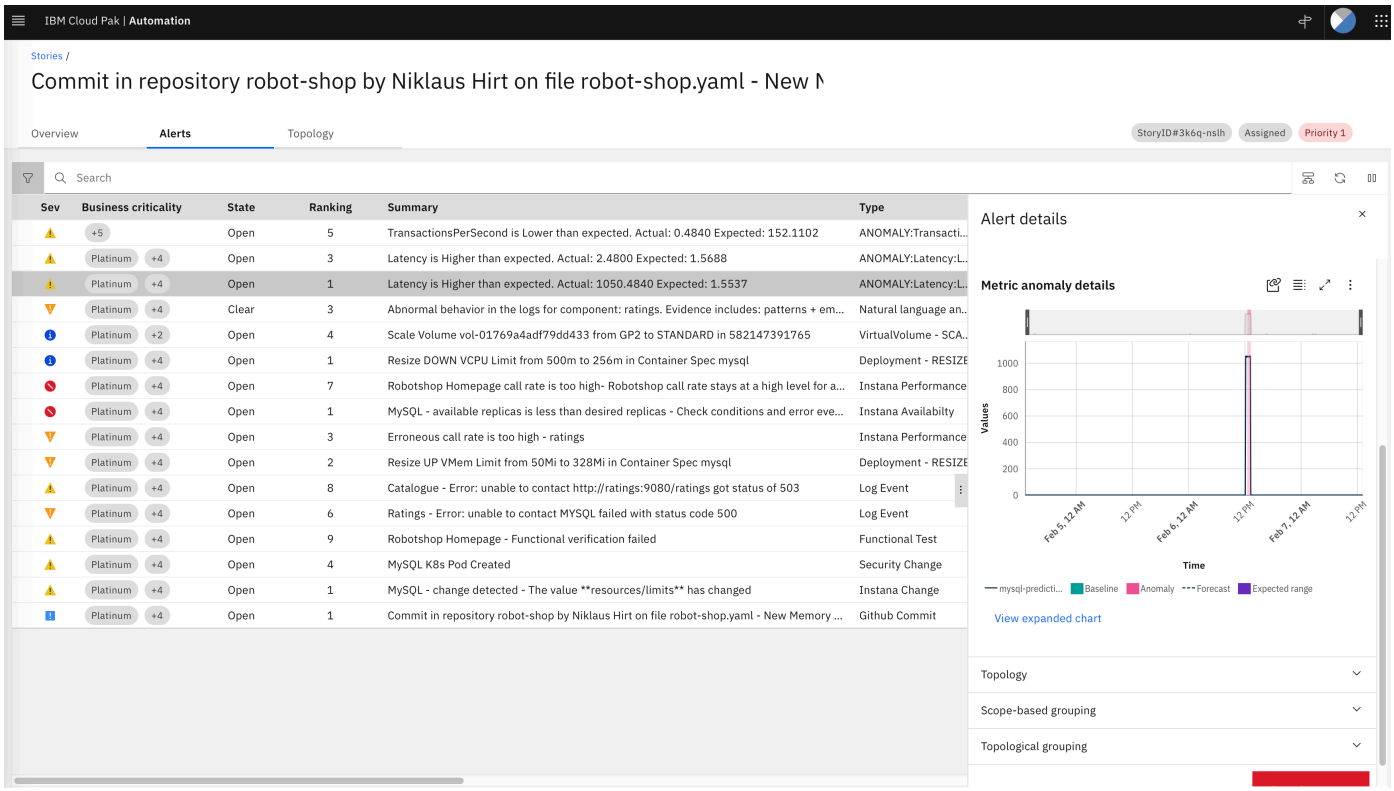
When trying to understand what happened during the incident, I sort the Alerts by occurence. This allows you to understand the chain of events.

- I can see that the first event was a code change that had been commited to **GitHub**. When I hover over the description I get the full text.
  So it seems that the Development Team has reduced the available memory for the mysql database.

Other events are confirming the hypothesis.

- I can then see the CI/CD process kick in and deploys the code change to the system detected by the Security tool and
- **Instana** has has detected the memory size change.
- Then **Functional Selenium Tests** start failing and
- **Turbonomic** tries to scale-up the mysql database.
- **Instana** tells me that the mysql Pod is not running anymore, the replicas are not matching the desired state.

- Cloud Pak for Watson AIOps has learned the normal, good patterns for logs coming from the applications. The Story contains a **Log Anomaly** that has been detected in the ratings service that cannot acces the mysql database.

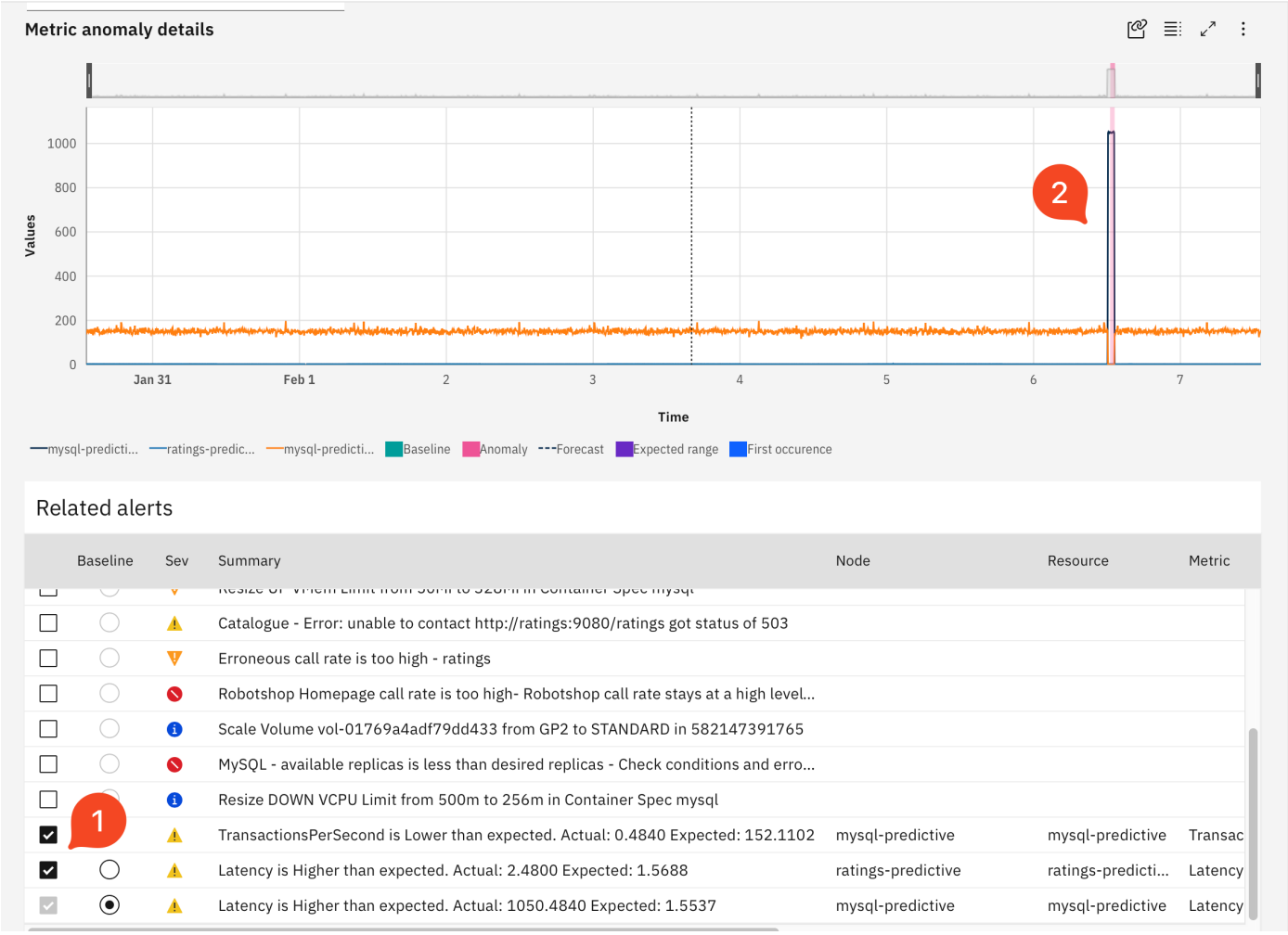


> 🚀 **Action**
> Click on a Alert line that has **ANOMALY:** in the Type column. Then open the **Metric Anomaly Details** accordion.

📣 Narration

- Cloud Pak for Watson AIOps is also capable of collecting metrics from multiple sources and detecting **Metric Anomalies**. It was trained on hundreds or thousands of metrics from the environment and constructs a dynamic baseline (shown in green). The graphic suddenly turns red which relates to detected anomaly when the database is consuming a higher amount of memory than usual.

Metric anomaly details



🚀 Action

(1) In **Related Alerts** select some additional alerts.

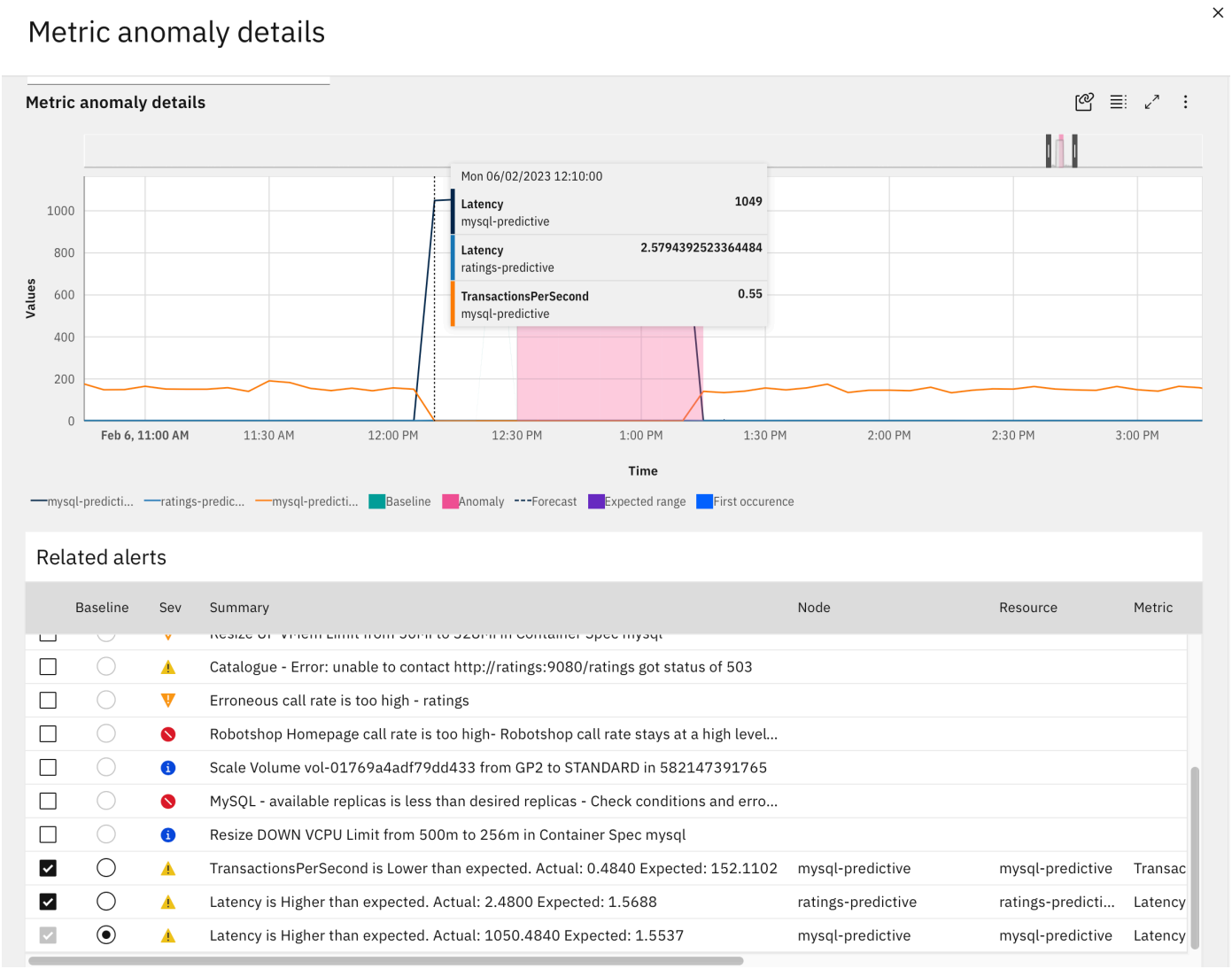
📣 Narration

You can display several alerts at the same time to better understand the temporal dependencies

> 🚀 **Action**
> (2) Select a portion of the graph with your mouse to zoom in

📣 **Narration**

Now let's zoom in to better see the anomalies



> 🚀 **Action**
> Hover over a datapoint to show the before/after values.

📣 **Narration**

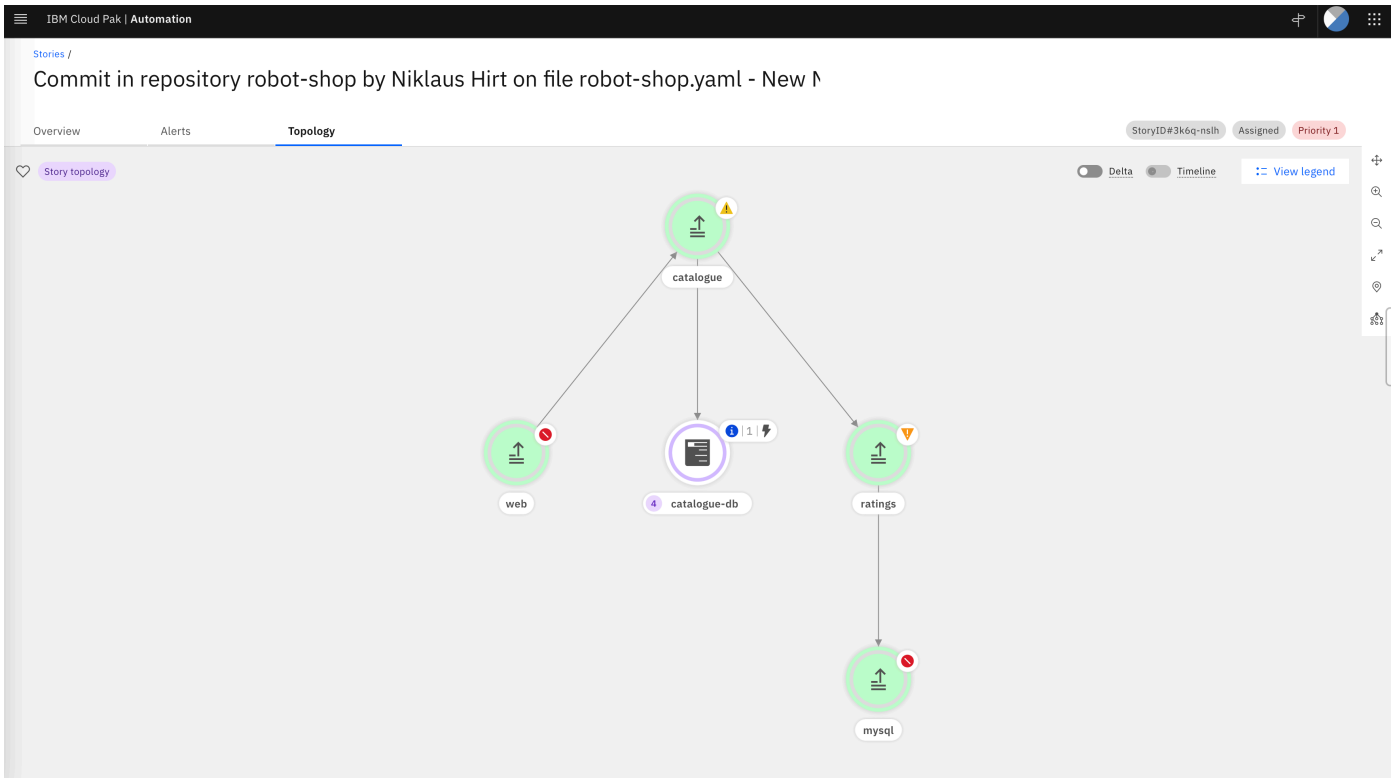
I can clearly see that the incident caused the **Latencies** to skyrocket and the **Transactions per Seconds** are almost zero. This is yet another confirmation of the source of the problem.

> 🚀 **Action**
> Close the Metric anomaly details view.

2.5 Working with Topology

2.5.1 Examining the Topology

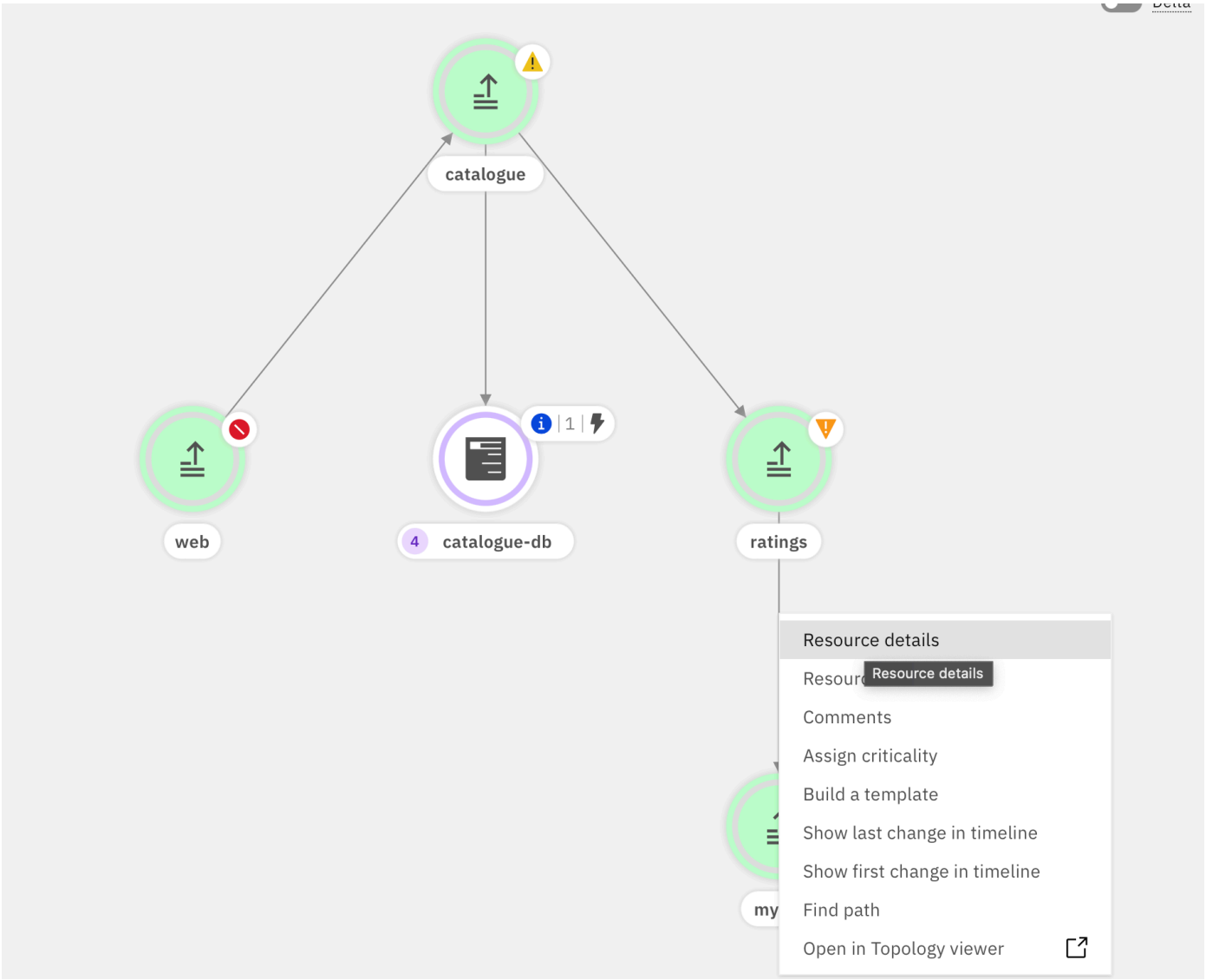
> 🚀 **Action**
> Click the **Topology** Tab.



📣 Narration

The interface shows the **topology** of the application that is relevant to the incident. IBM Cloud Pak for Watson AIOps' topology service delivers a working understanding of the resources that you have in your environment, how the resources relate to each other, and how the environment has changed over time.

You can see that there are some statuses attached to the different resources, marked with colorful dots. Let's view the details and status of the **mysql** resource with red status.

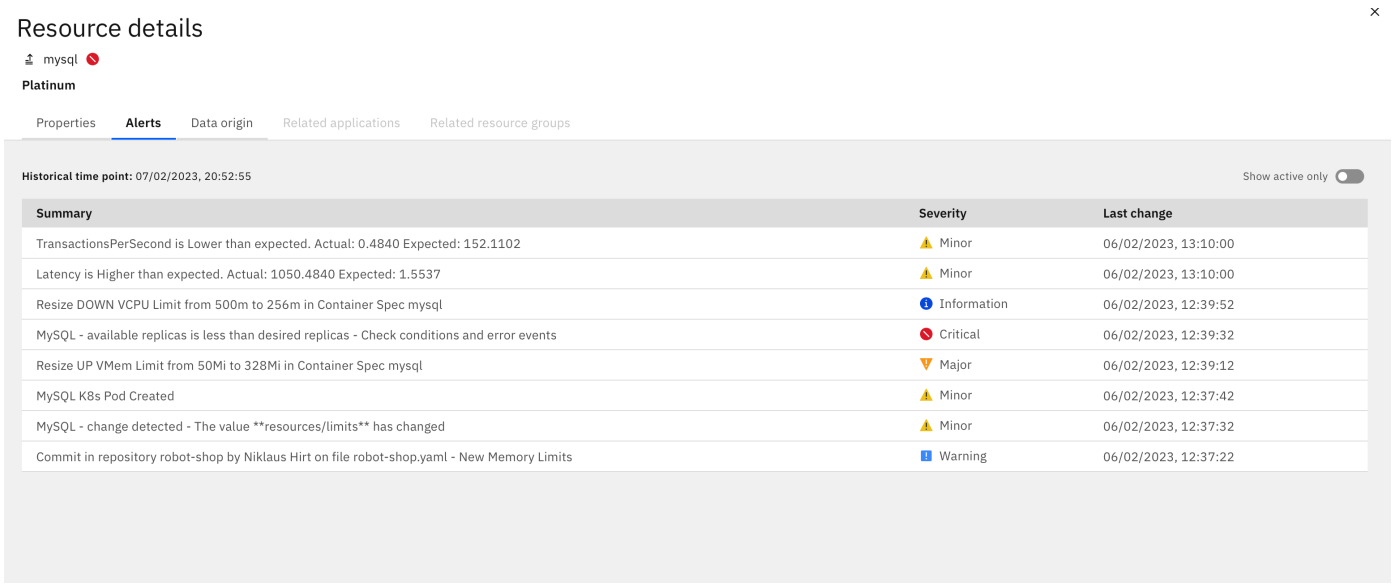


🚀 **Action**

Find the resource which displays resource name “**mysql**”. Then, right-click and select **Resource details.**

🚀 **Action**

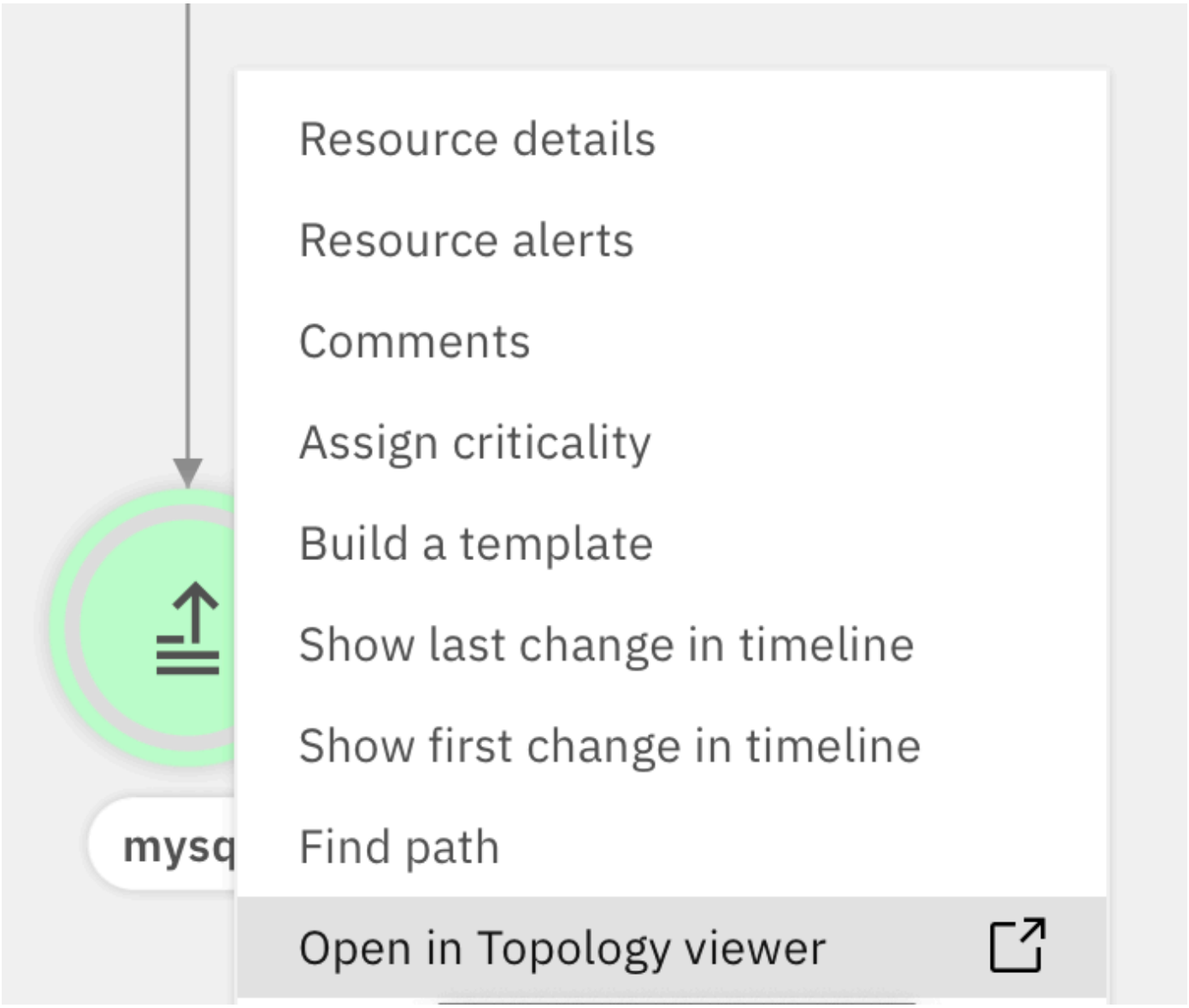
Click on Tab **Alerts**



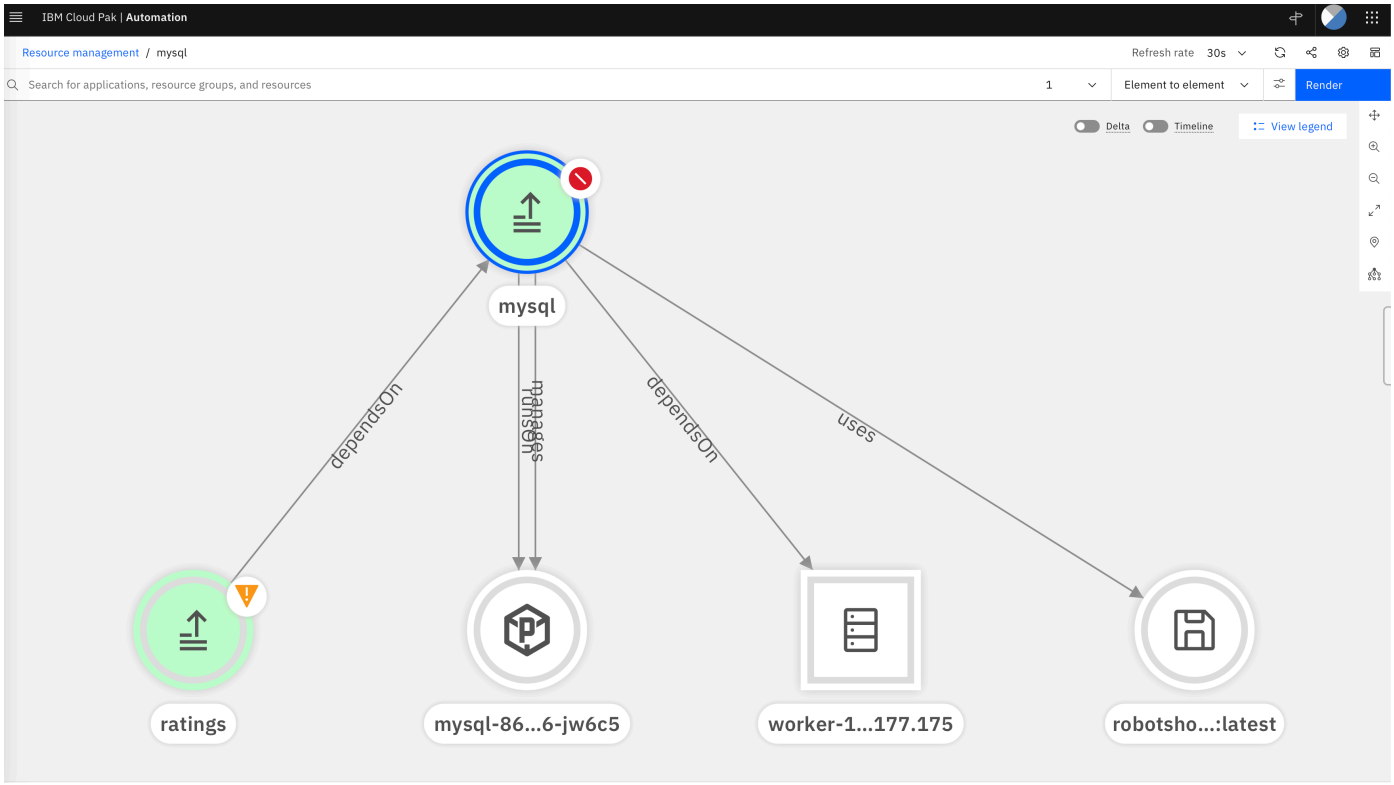
📣 Narration

The topology service provides operations teams with complete up-to-date visibility over dynamic infrastructure, resources, and services. The topology service lets you query a specific resource for details, and other relevant information. Here I can see all Alerts for the mysql database resource for example.

2.5.2 [Optional] Topology in-depth

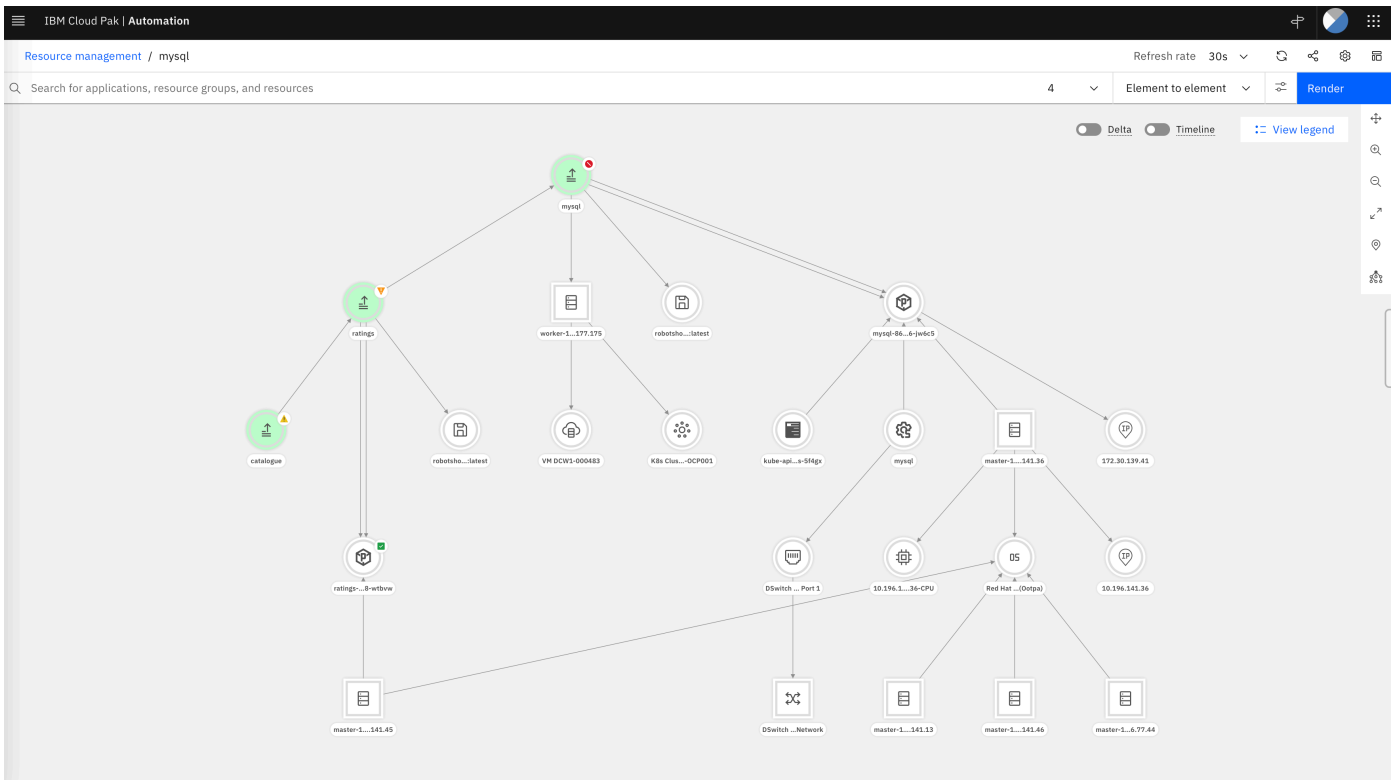


🚀 **Action**
Find the resource which displays resource name “mysql”. Then, right-click and select **Open in Topology Viewer.**



Narration

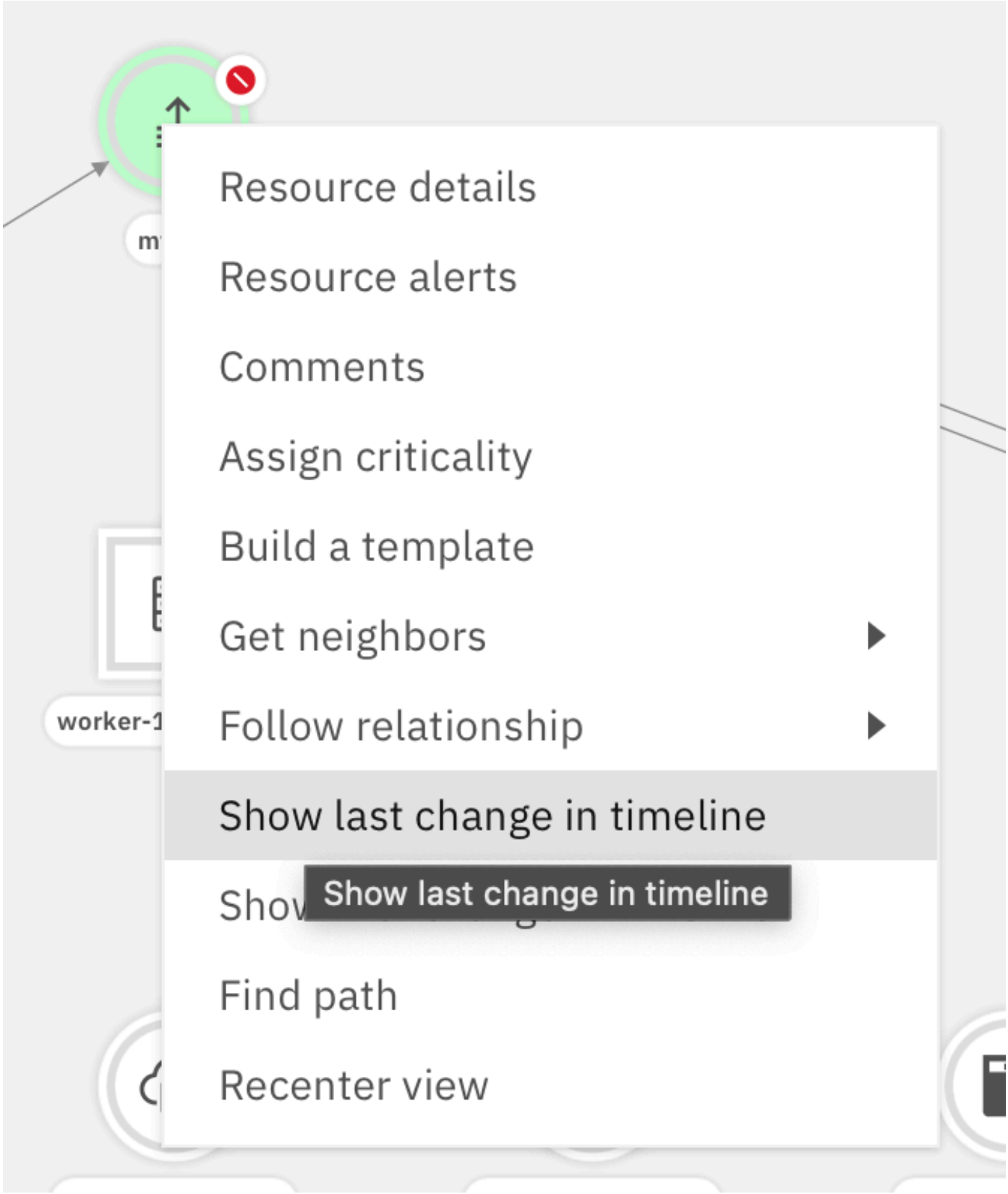
The interface shows the topology surrounding the mysql resource. I can see that the **mysql** deployment is being called by the **ratings** service and that it runs on a certain worker node.



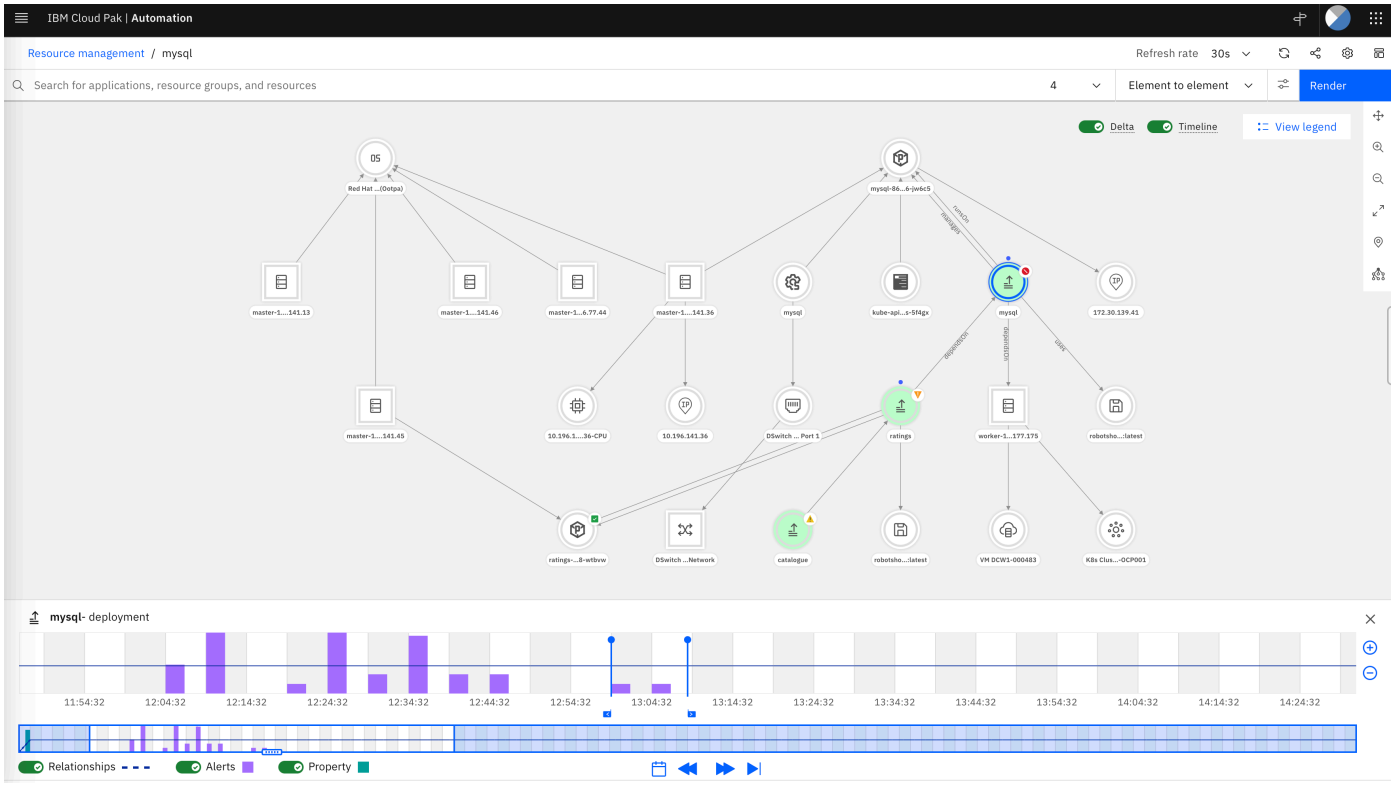
> 🚀 **Action**
> Change the number of hops to `4` and click **Render**.

🔈 **Narration**

I can also increase the size of the graph, still based on the **mysql** deployment.



🚀 **Action**
Right-click om mysql and select **Show last change in timeline** and check **Delta**



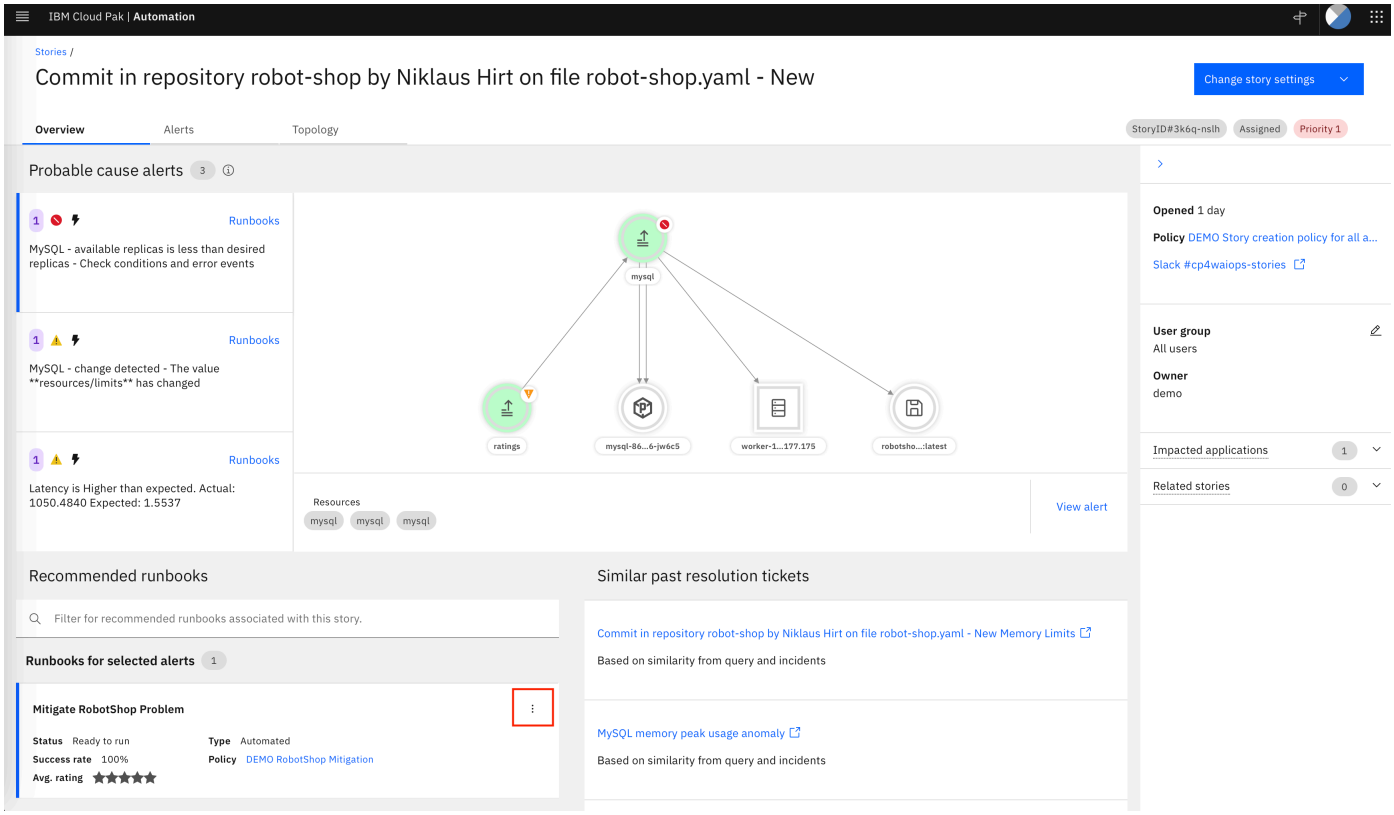
Narration

Now I will examine the historical events for the **mysql** component. I can see the **Alerts** that have been raised on the **mysql** resource over time.

2.6 Resolving the incident

2.6.1 Fixing the problem with runbook automation

> 🚀 **Action**
>
> Click on the **Overview** Tab.

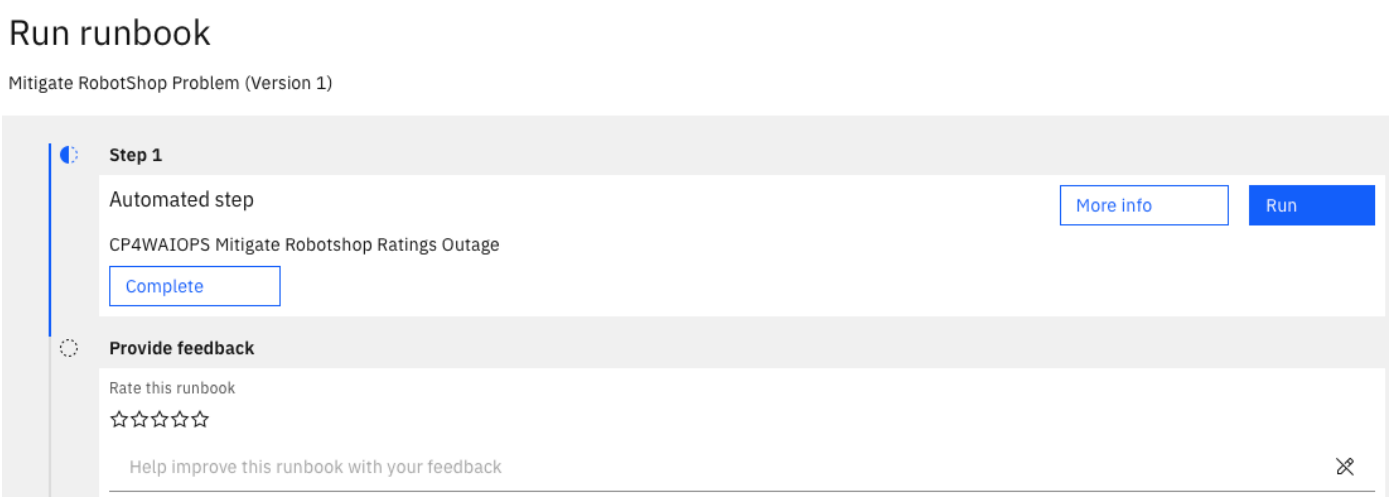


📣 Narration

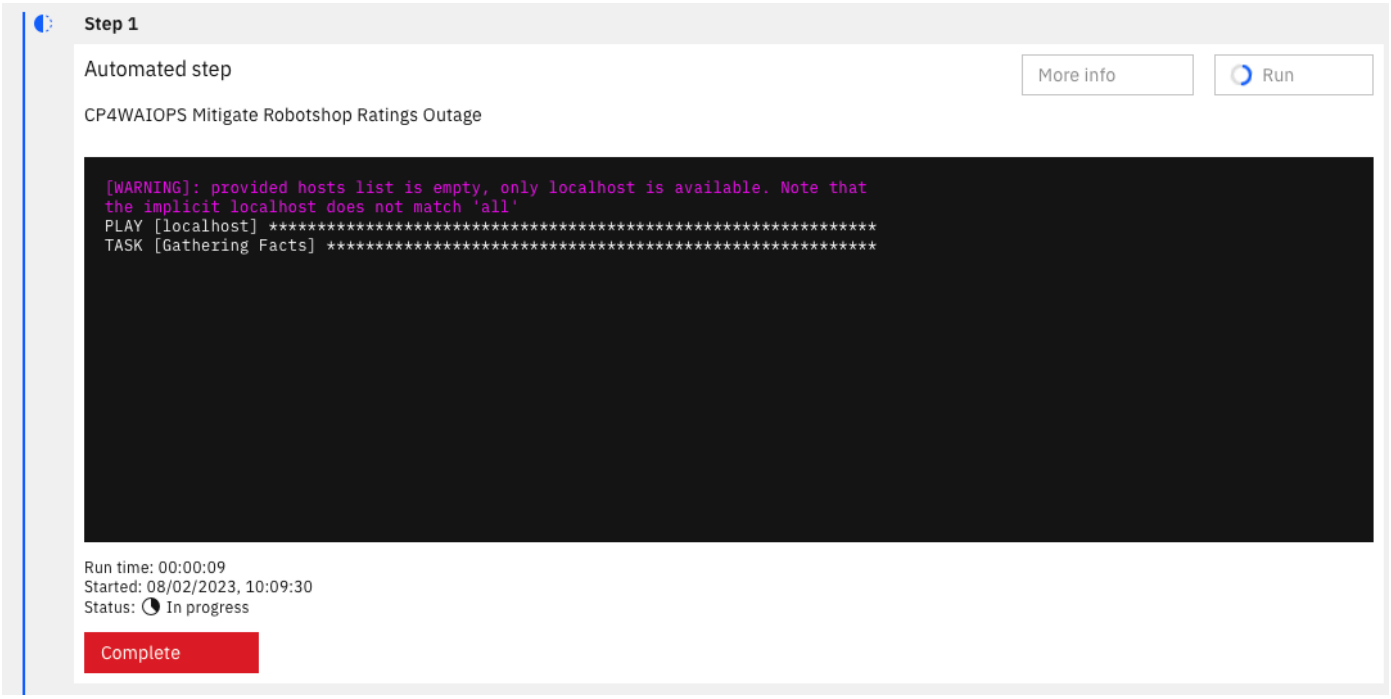
Now that we know what the problem is, let's correct what has happened. A runbook has been automatically identified but have not been executed. Runbooks are guided steps that IT operations teams use to troubleshoot and resolve problems. Some organizations might call these standard operating procedures or playbooks. When an incident occurs, IBM Cloud Pak for Watson AIOps matches an appropriate runbook to the problem. The runbook can be set to run automatically when it is matched to an incident, or it can run with user approval and participation.

Let’s execute the Runbook.

> 🚀 **Action**
>
> Click on the three dots and click **Run**
>
> Click **Start Runbook**.



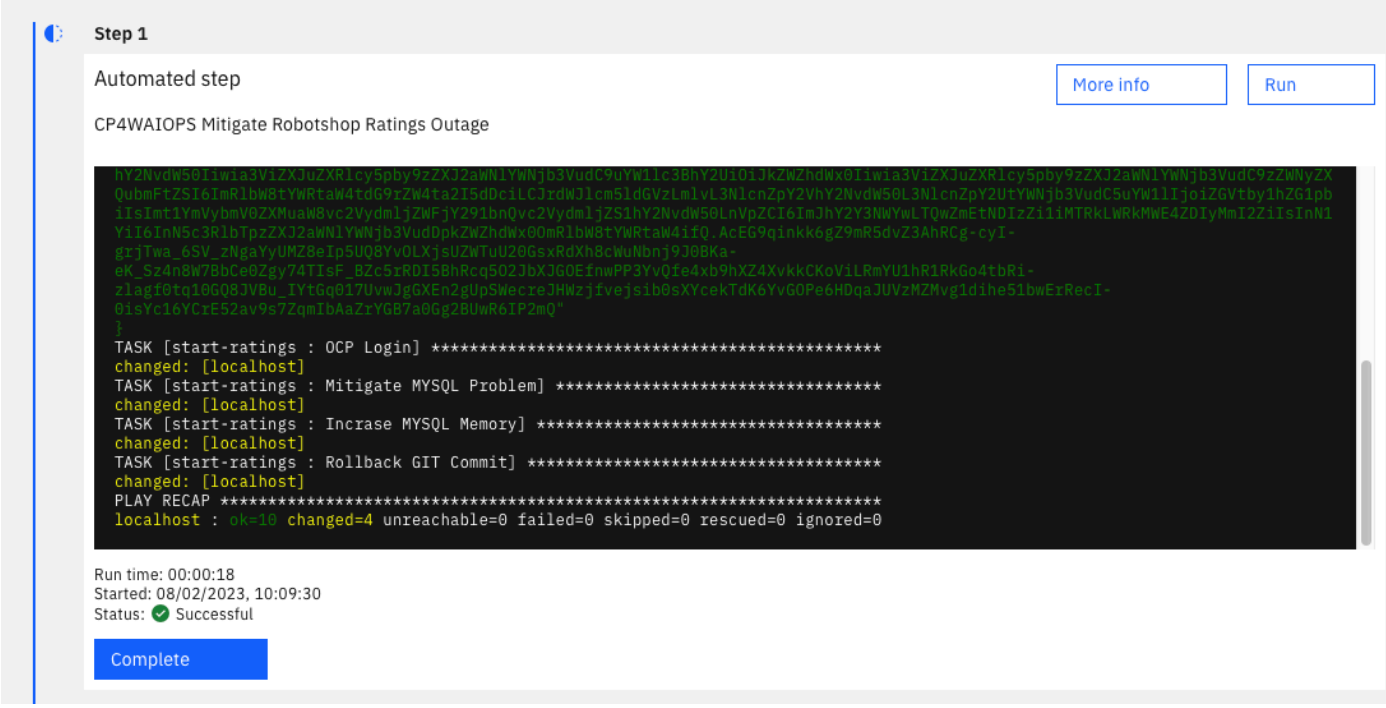
> 🚀 **Action**
>
> Click **Run** in Step 1.



❗ **Note**: The execution of the runbook can take few minutes.

📣 Narration

The Runbook that I just started kicks off a Playbook on Ansible Tower. I can follow the execution as it connects to the cluster and then scales up memory for the MySQL deployment.



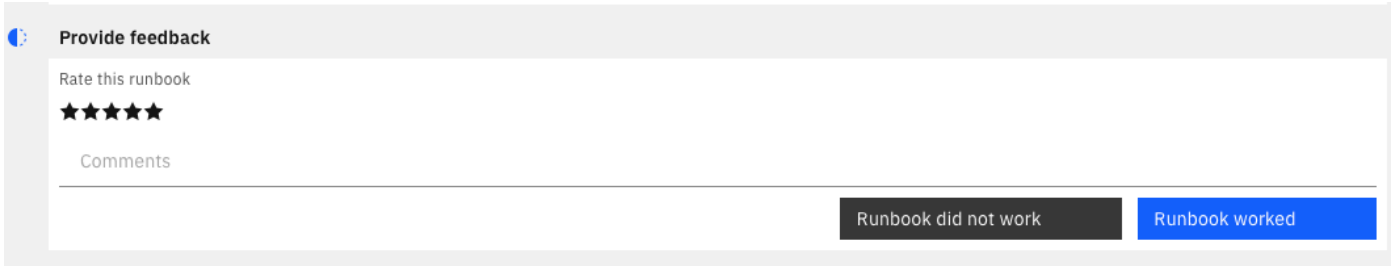
🚀 Action

When finished, click **Complete**.

Open the RobotShop application. Verify that ratings are correctly shown

Narration

Before confirming that the runbook worked as expected, I should check the RobotShop application to see if it is working as expected.



Action

Rate the Runbook

Then click **Runbook Worked**.

Narration

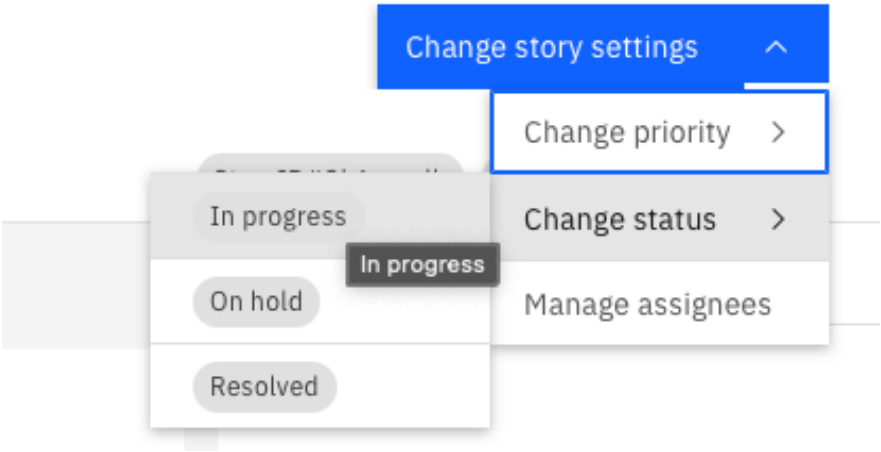
So the runbook has resolved the problem. When I tell Watson AIOps that the Runbook worked, it will learn over time to prioritize and suggest more relevant Runbooks.

2.6.2 Resolve the Incident

🚀 **Action**
Click on **Change Story Settings.**

Select **Change Status.**

Click on **Resolved**



📣 **Narration**

So now as we have resolved the problem, I will inform the development team of the problem by reopening the ServiceNow ticket and by closing the Story.

Demonstration summary

📣 **Narration**

Today, I have shown you how Cloud Pak for Watson AIOps can assist the SRE/Operations team to identify, verify, and ultimately correct an issue with a modern, distributed application running in a cloud-native environment. The presented solution provides automatic application topology discovery, anomaly detection both with metrics and logs, and sophisticated methods of correlation of events coming from different sources.