ARCTURUS
CONSTANT INNOVATION

APPLICARE 9.2.X

Using Applicare

Dashboards, applications, transactions, infrastructure and AIOps features.

Version 9.2

Arcturus Technologies Inc. · Published November 3, 2026
Generated from the Applicare master documentation. 210 of 210 articles were migrated from earlier documentation and are pending technical review.
The latest version is always available at https://helpdesk.arcturustech.com/hc/en-us

Contents

USING APPLICARE

App Monitoring Dashboard 4
Operations Center 10
Overview 12
Server Comparison 14
Events Calendar 15
Enterprise 16
Unattached servers 21
Log Analyzer 22
Alerts 33
Custom Search in LogAnalyzer 38
Date and Regex pattern format 43
LAMA 45
Business Transactions User Guide 61
Automatically add KBT configuration 73
Download IntelliTrace report 75
Configuration for KBT with parameter 78
Server Analyzer 82
Applicare Docker Monitoring 97
Operating System Analyzer 100
JVM Analyzer 102
GC Analyzer 103
Network Analyzer 105
Sensor Analyzer 108
Applicare Custom JMX Analysis: Create a Sensor 109
Monitoring MSSQL Metrics with DB Query Sensors 111
Server Availability Monitoring 113
User Guide for Performance Comparison 115
Applicare IntelliSense 119
Arcturus Insights - ArcIn 122
Applicare Genie 124
ArcIn Database Configuration 126

Set Up Predictive Analysis in Applicare on Linux 128
End User Experience Monitoring (EUM)/User Experience Monitoring (UEM) Analyzer 131
EUM Real-Time Transaction Monitoring 141
Applicare RUM/EUM Integration Script 145
UEM Injection through the Applicare Reverse Proxy Server 146
Apache User Experience 153
UEM IP Address Translation 157
iOS Mobile App Performance Monitoring 158
Monitoring performance for Android mobile apps 164
Measuring User Satisfaction - Apdex 169

USING APPLICARE › DASHBOARDS

App Monitoring Dashboard

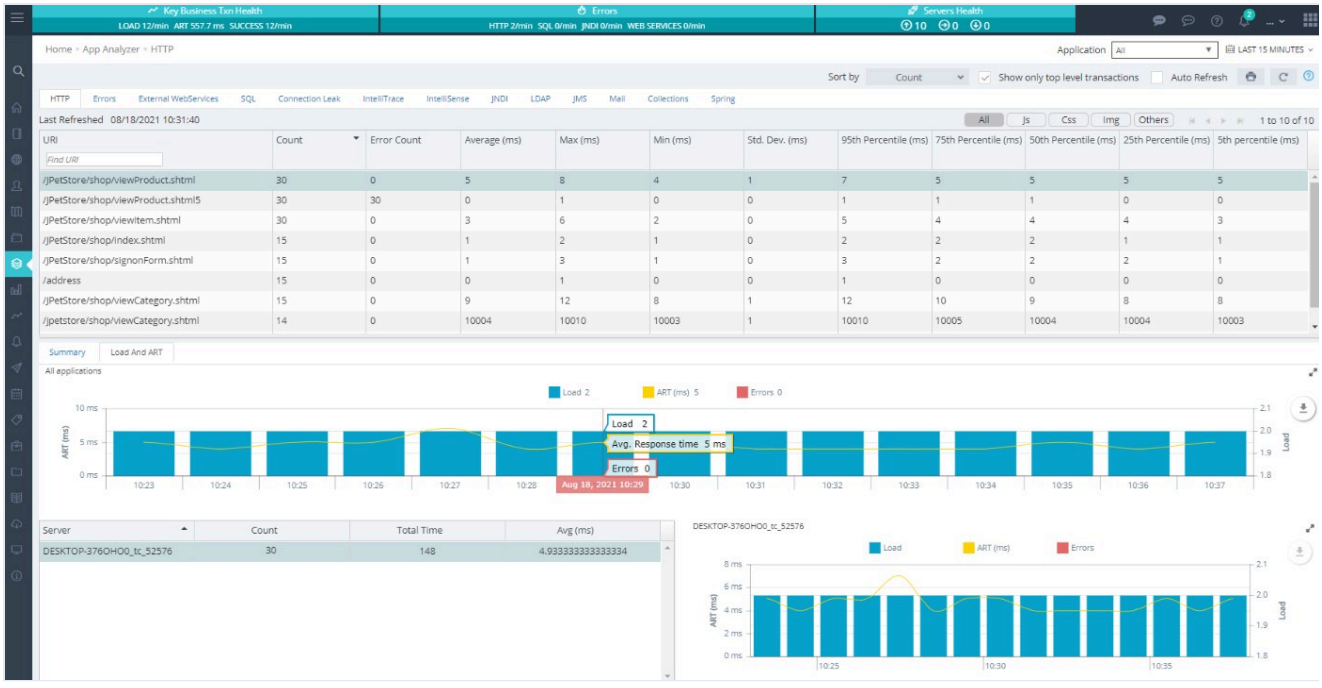
Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Overview

The App Analyzer is designed to evaluate the performance of an entire enterprise application, providing comprehensive insights into various transactions, SQL queries and other metrics across all servers that support the application.

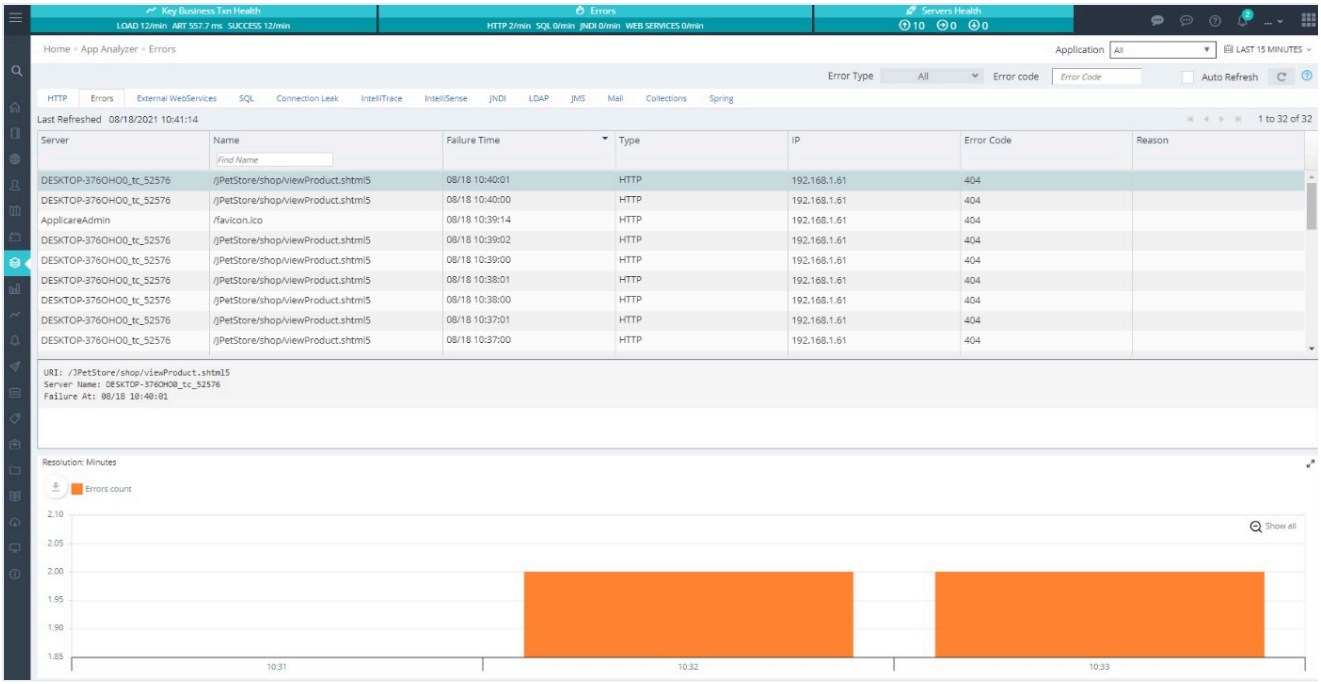
HTTP

The top table displays URI along with its performance metrics, including count, average and performance percentiles. Selecting a row will reveal the performance details for that URI within the chosen time period in the central chart. On bottom left, performance is shown on all the servers in the application and on bottom right is the chart showing comparative performance across all servers.



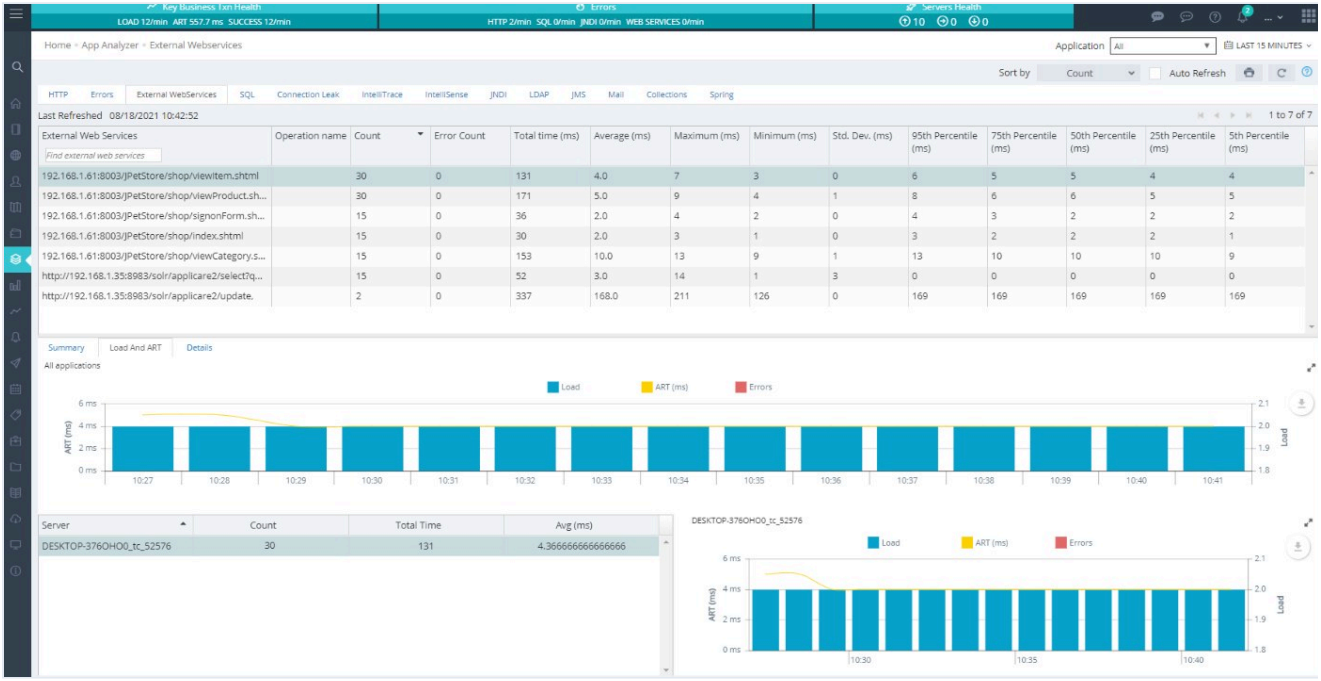
HTTP Errors

The HTTP Errors tab provides details of HTTP errors and allows drilling down into any specific http error. Details available for each http error are server name, URI, failure time, IP of the user who experienced the error, HTTP error code (404, 500 etc.) and HTTP exception details. Exception stack is available in the middle text box on selecting a transaction row in the table on top.



External WebServices

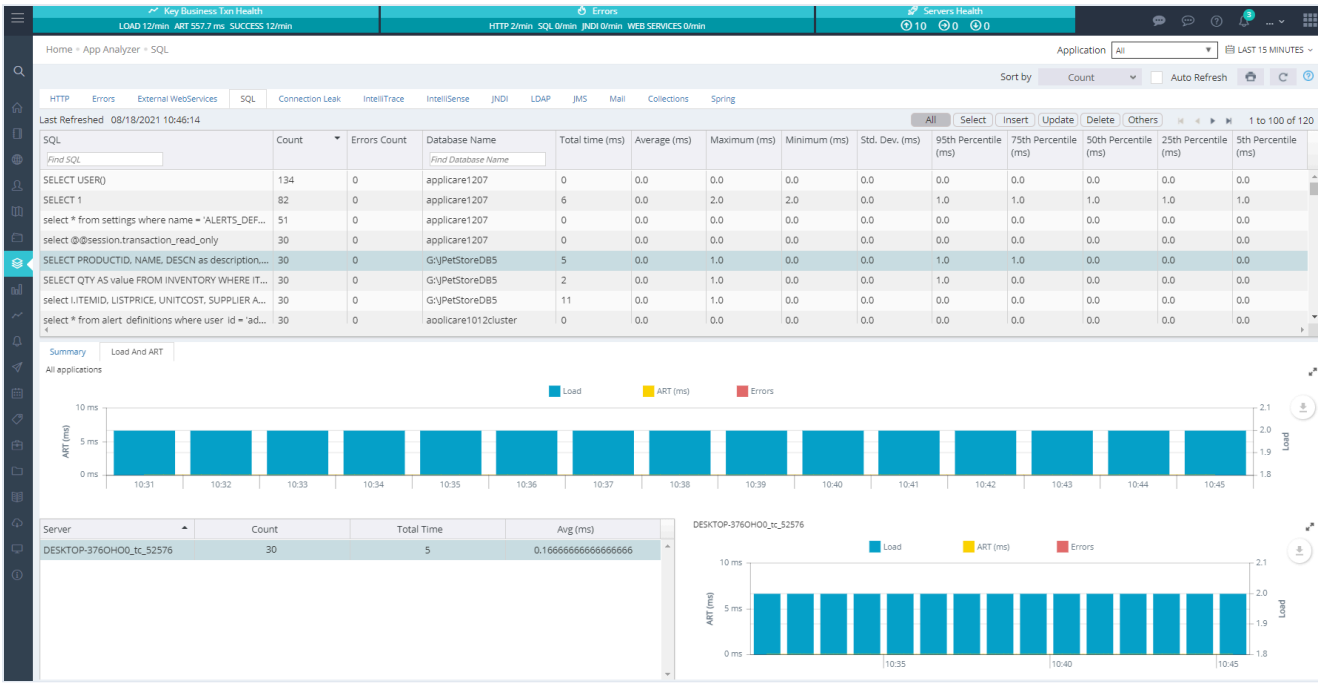
The External Web Services tab works very similar to HTTP tab, and provides performance data for external web service calls made from your application. Top table presents performance data in a tabular format, while the middle charts illustrate the performance of web services across all servers over time. Bottom left table shows performance on individual servers and bottom right chart either shows comparative performance or performance on an individual server if a server or row is selected in the bottom left table.



SQL

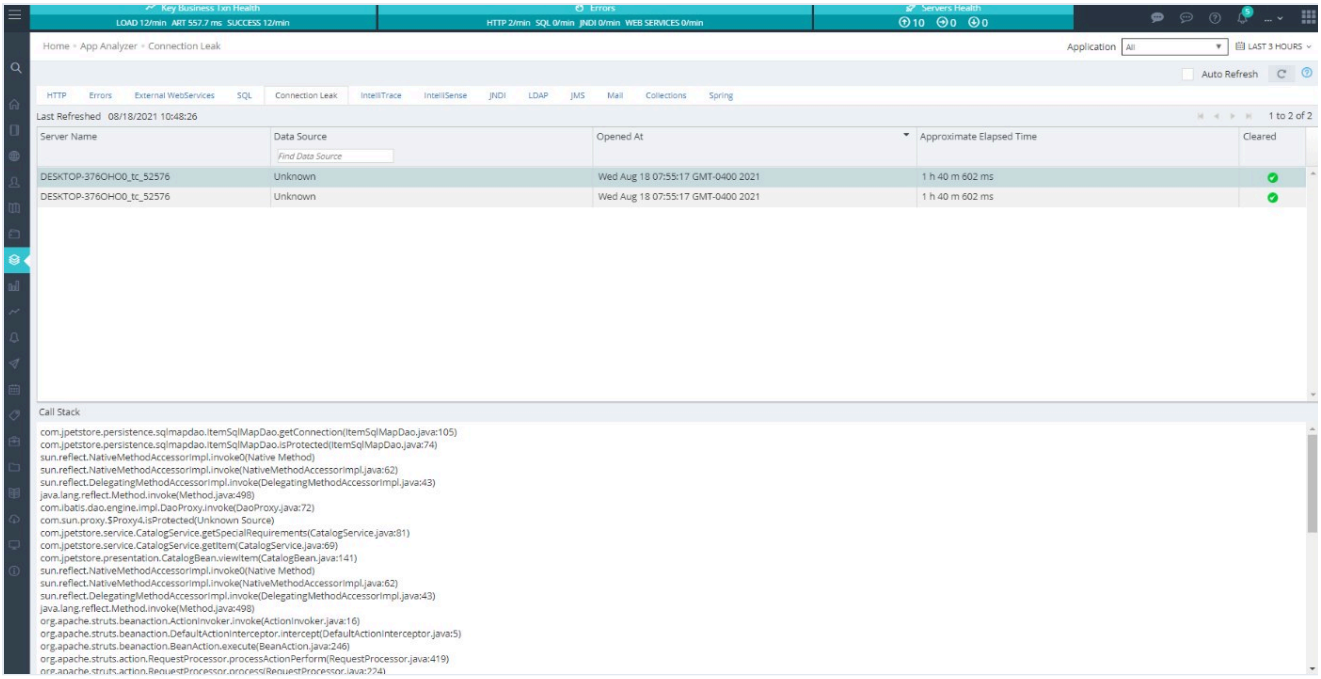
The SQL tab displays cumulative performance data for various SQL queries executed on the server within the app. Initially, a pie chart reveals the percentage of total time consumed by different SQL queries.

When you select a specific SQL query, the middle chart will present its cumulative performance across the application, while the bottom right chart provides a comparative view of performance across all servers. The bottom left section details performance on individual servers, allowing you to a server and view its performance in the bottom right chart.



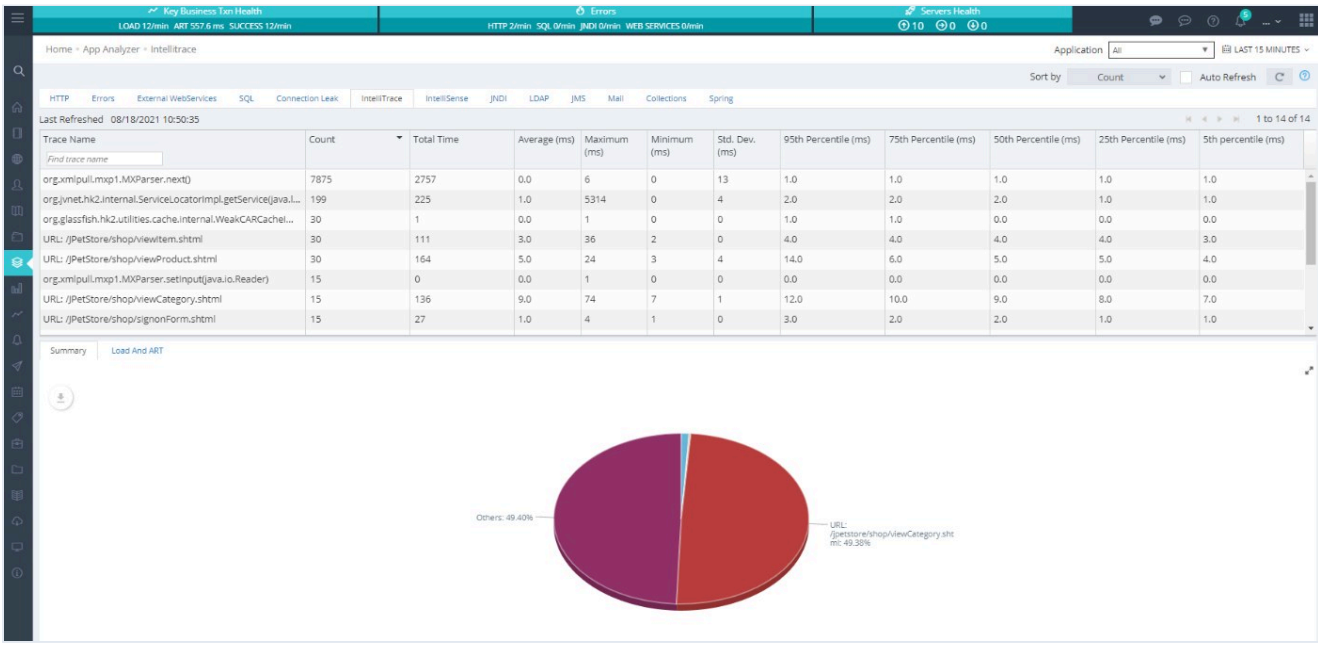
Connection Leak

The Connection Leak tab shows detailed information about each connection leak, including the code responsible for the leak, the associated datasource, the time the connection was opened, and the duration for which it has been held.

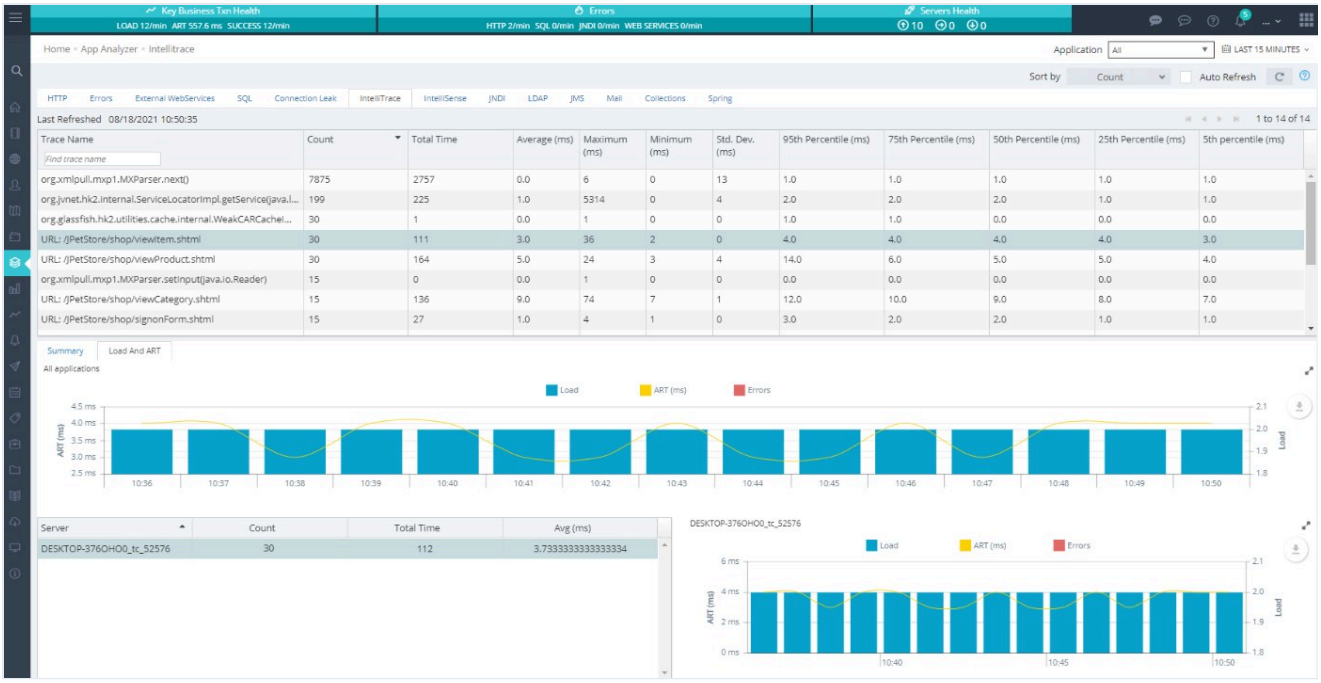


IntelliTrace

IntelliTrace tab shows various transactions including java, web sphere and HTTP calls flowing through the system. Bottom pie chart shows % based on total time consumed by a particular transaction and shows the impact of the transaction in grand schema of things.

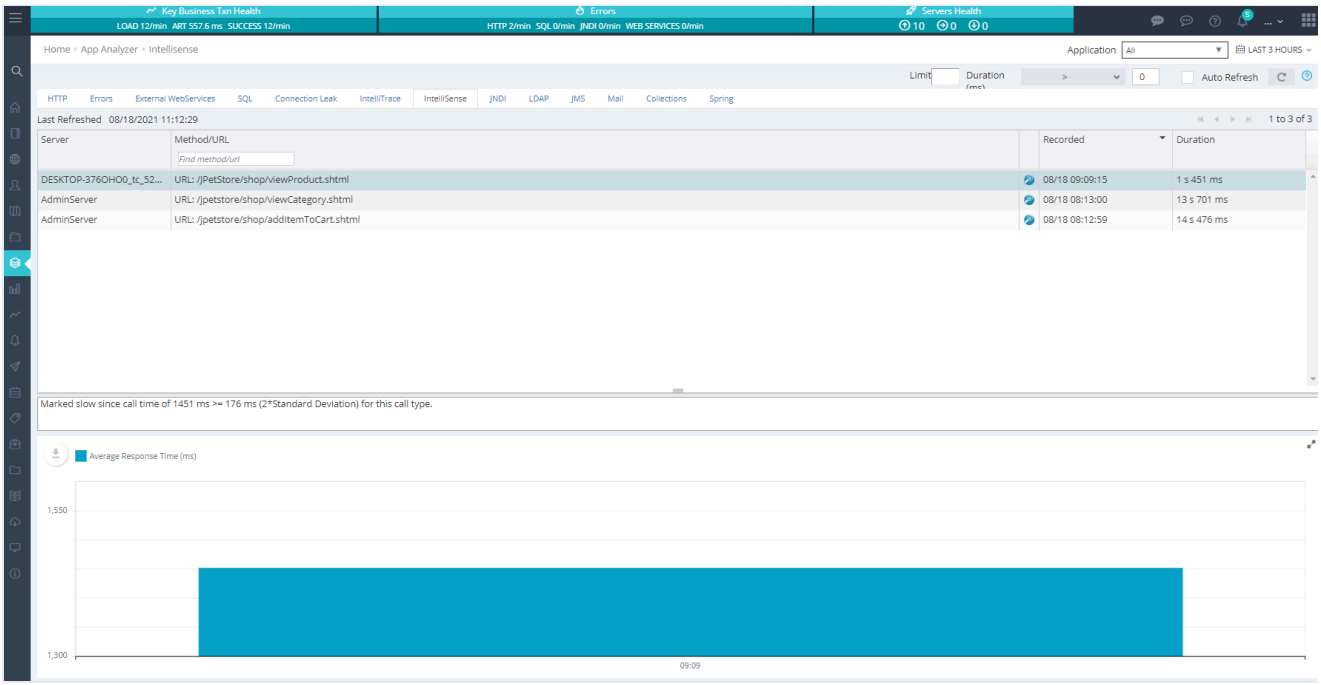


When you select a transaction in top table, the middle chart shows cumulative performance. Bottom left table shows transaction's performance across servers. Bottom right chart shows comparative performance or performance on a particular server if a sever row is selected in the table on the left.



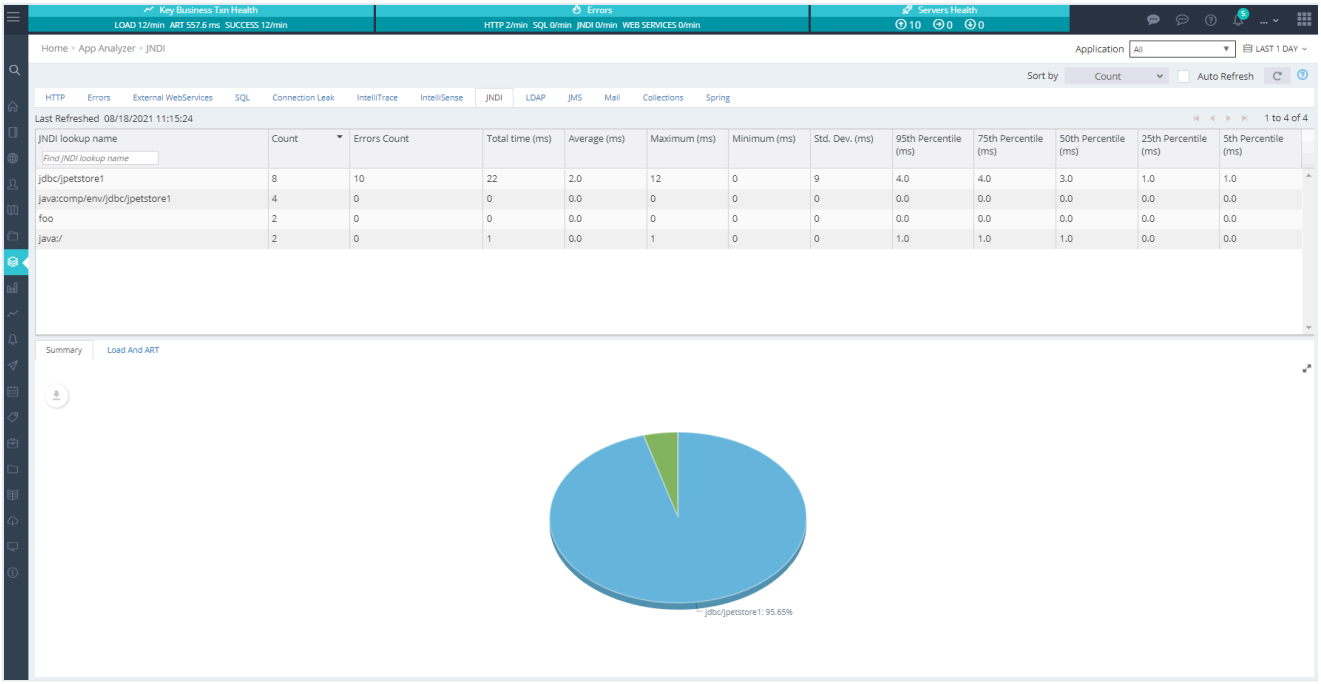
IntelliSense

IntelliSense automatically collects in-depth performance data for transactions that exceeds predefined criteria such as those with duration more than a twice the standard deviation for the specified category call type. For detailed transaction performance stats, simply double-click the corresponding row.

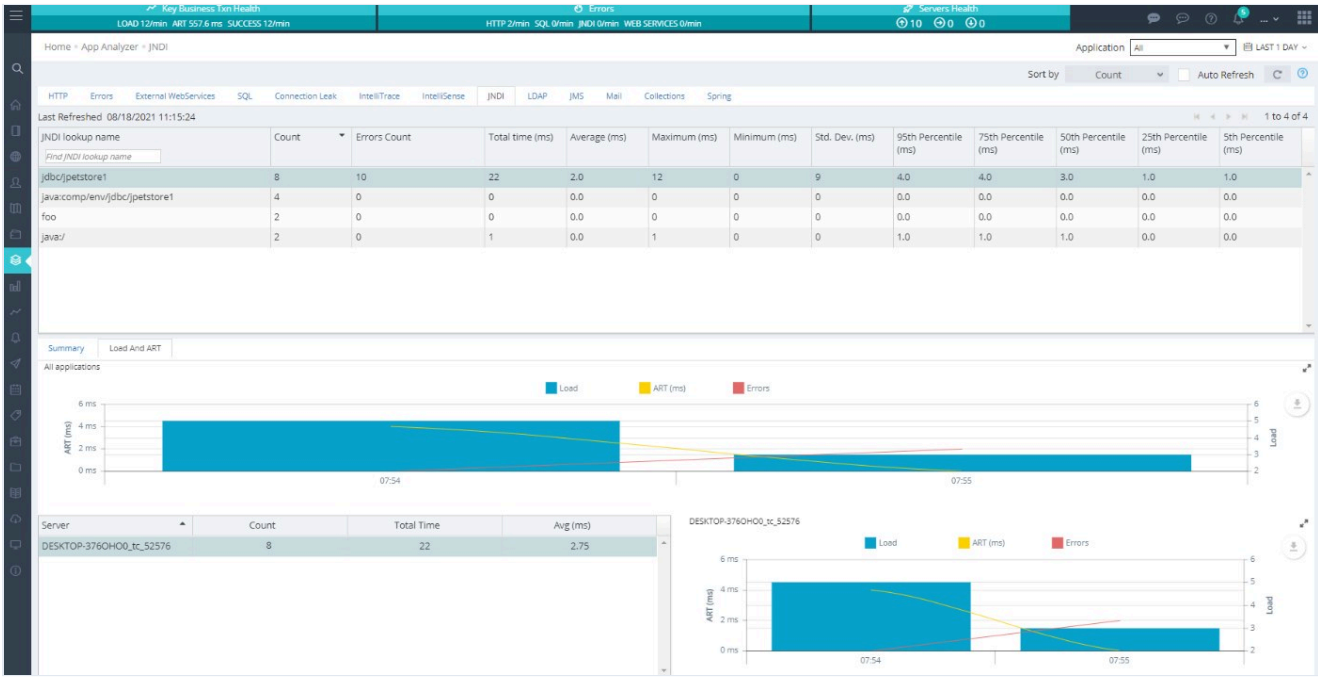


JNDI

The JNDI tab displays performance details for JNDI lookups. The initial view represent a pie chart with illustrating the percentage of time taken for these lookups.



When you select a specific row, detailed information will be displayed in both the middle and bottom tables/charts.



Monitoring with Applicare

USING APPLICARE › DASHBOARDS

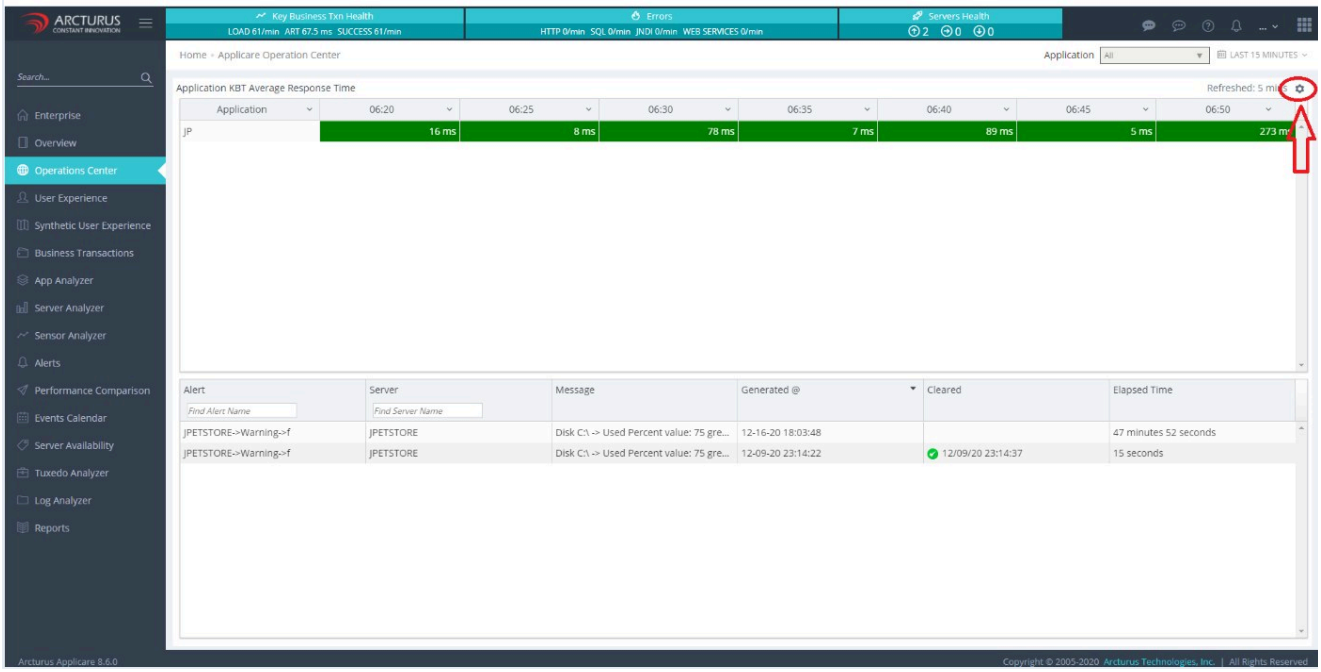
Operations Center

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

The Operation center shows Average Performance Time (ART) of KBT performance based on applications. It displays from last 60 mins data and refreshes automatically every 5 mins.

Applications KBT transactions in last 60 minutes will not be displayed. At the bottom of the page grid, the grid shows alerts from the last 60 minutes for the Application listed in the KBT's ART grid at the top.

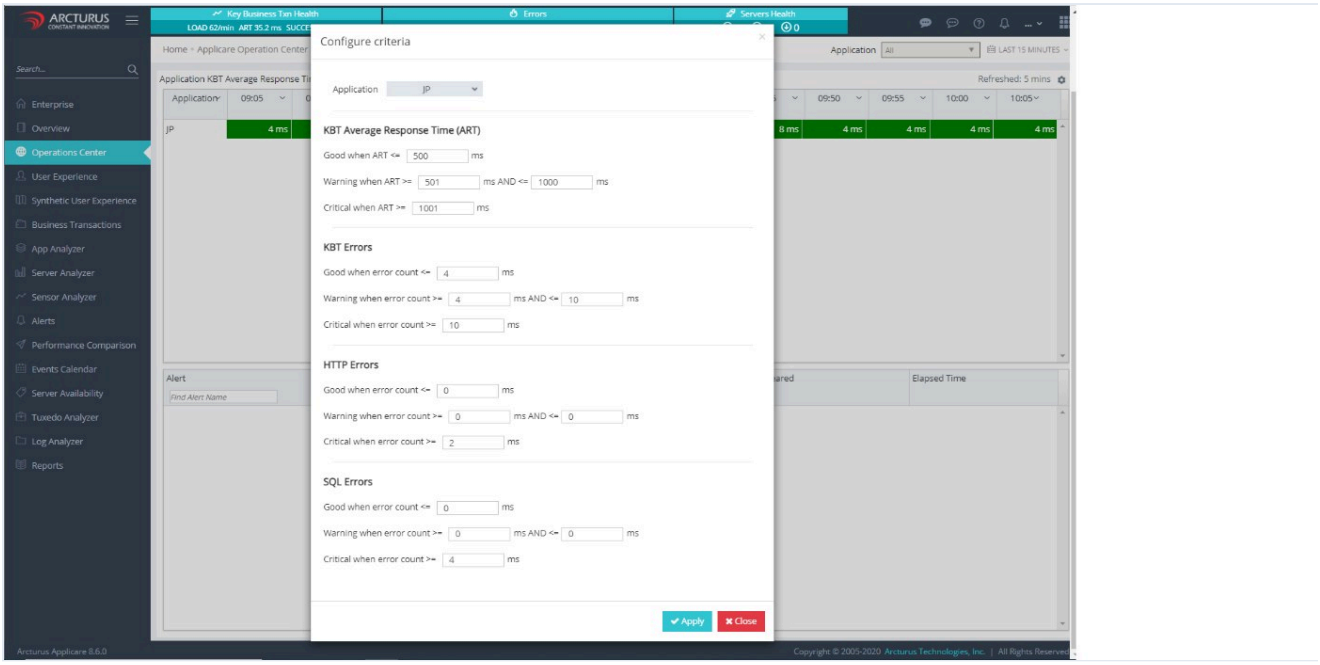
Configure criteria



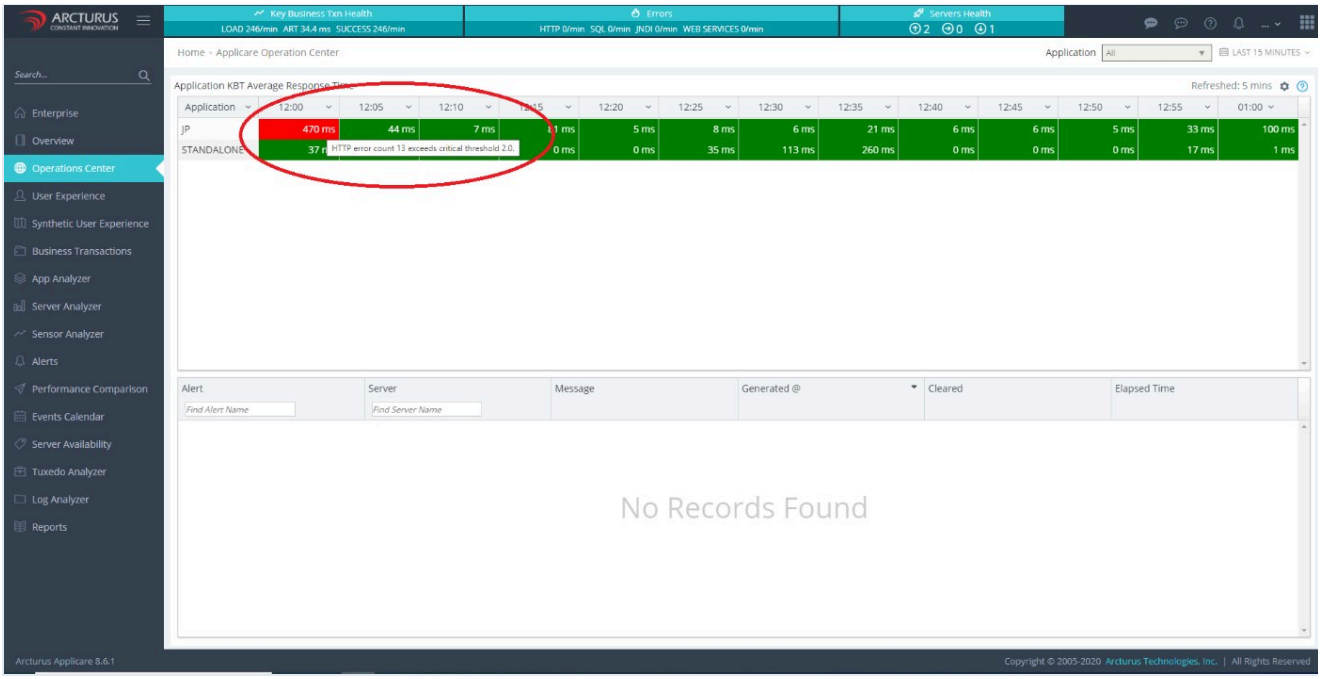
> [Screenshot 2 could not be downloaded during import — see migration report]

The "Configure Criteria" feature helps users easily differentiate between best, average, and worst performance using color coding. You can set color codes for KBT ART, KBT errors, HTTP errors, and SQL error counts. Values within the good range are shown in green, those in the warning range in yellow, and critical values in red.

You can configure KBT criteria for individual applications.



> _[Screenshot 4 could not be downloaded during import — see migration report]_To view more details, hover your mouse over any color-coded cell in the KBT's ART grid and a tooltip will display the information.



> _[Screenshot 6 could not be downloaded during import — see migration report]_By default, Application with more critical state are displayed on the top and least one are displayed at the bottom (i.e Apps with more Red are displayed on the top and least Red are displayed at the bottom). It helps to easily identity the apps in trouble.

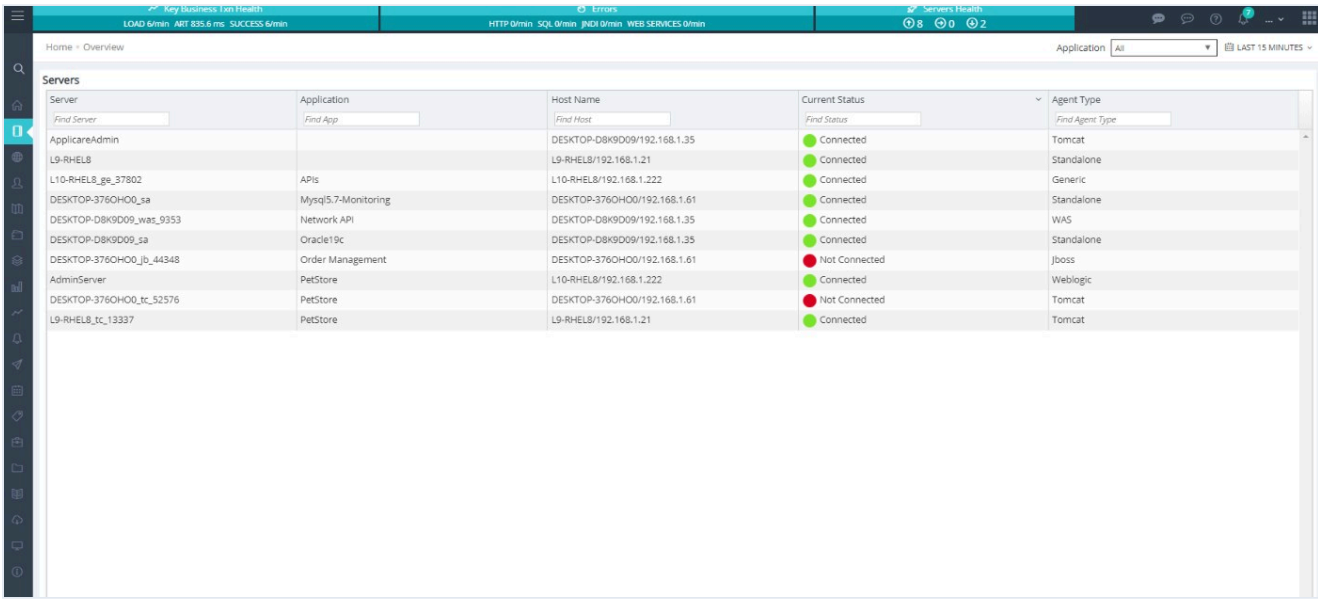
Monitoring with Applicare

USING APPLICARE › DASHBOARDS

Overview

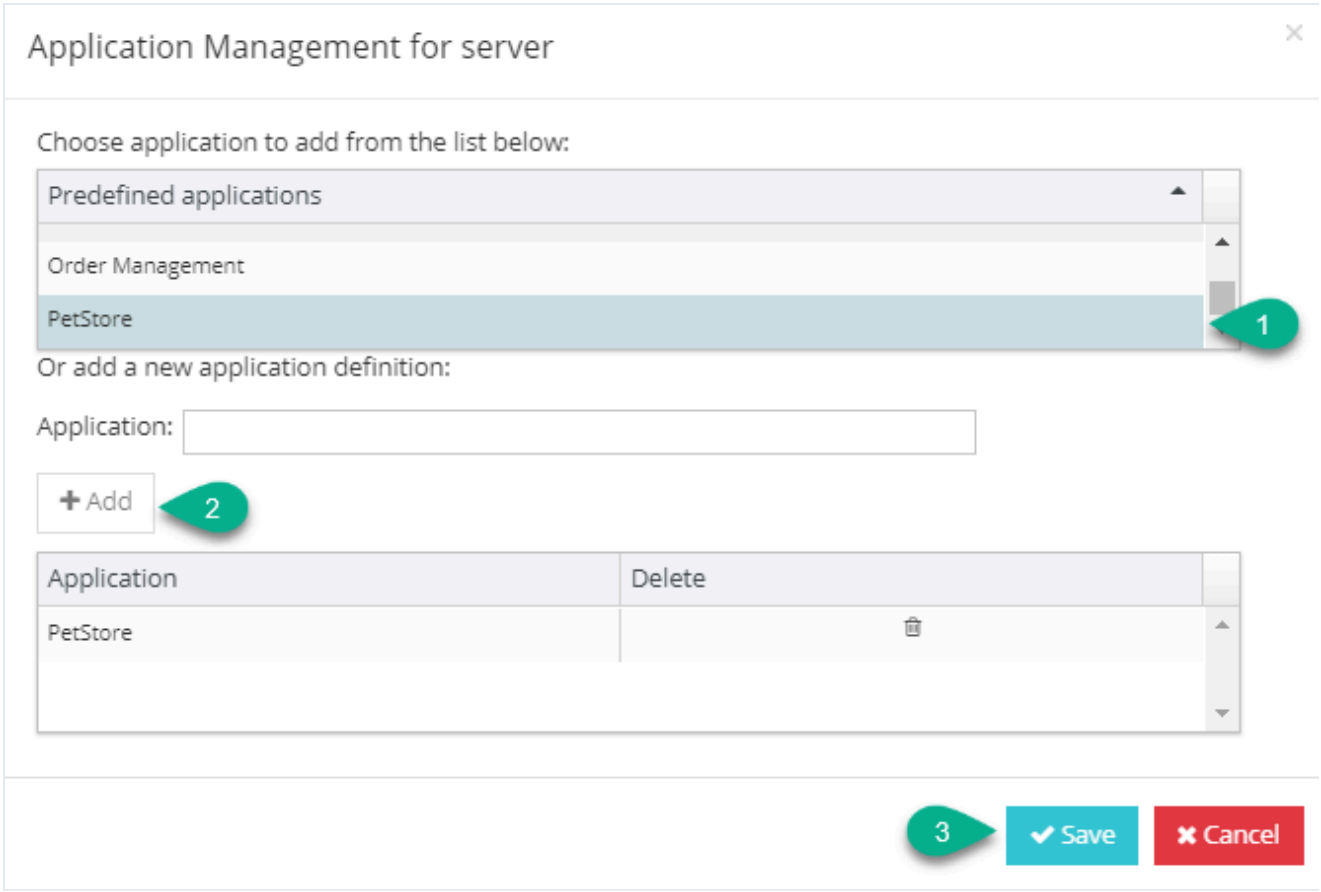
Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

The Overview tab provides information on the server's health & other details.



Servers

After adding an Applicare agent here, you can assign it to an application. An application serves as an administrative entity, enabling the grouping of data.



For Agent Auto Update and Restart Click here

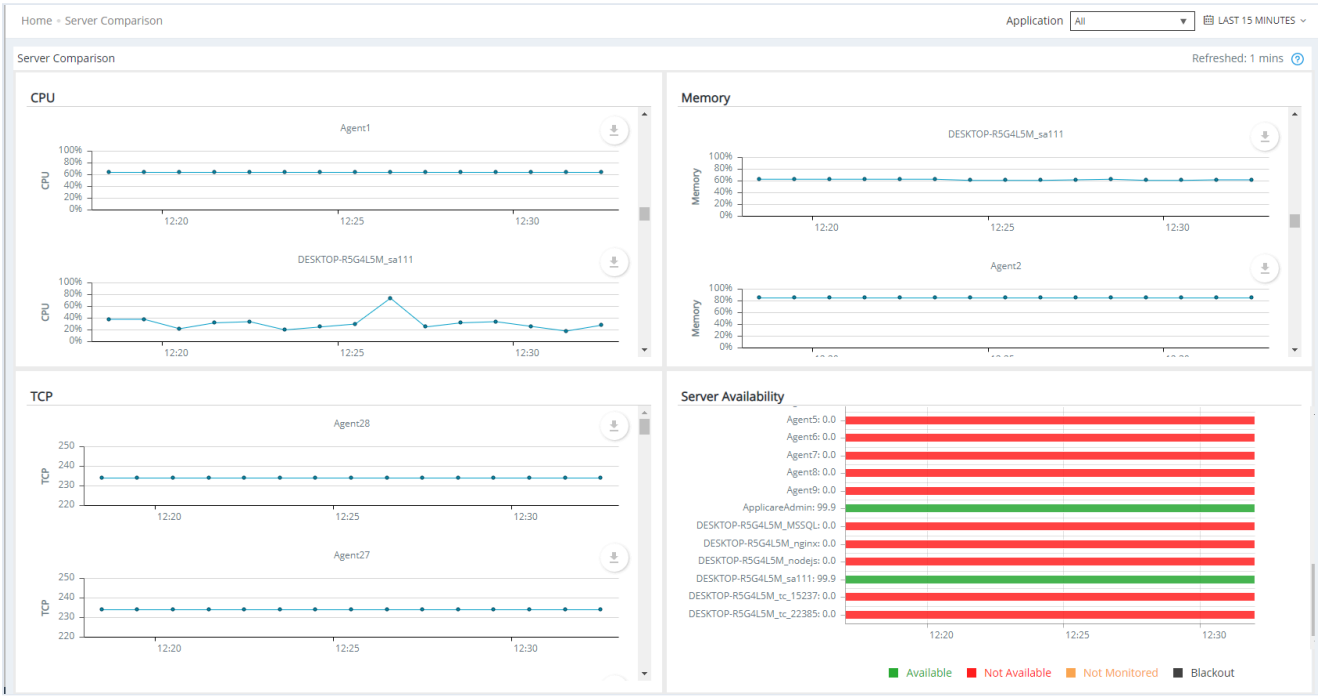
Monitoring with Applicare

USING APPLICARE › DASHBOARDS

Server Comparison

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Server Comparison offers a quick and convenient view of your server's CPU, memory, tcp and health details.



Monitoring with Applicare

USING APPLICARE › DASHBOARDS

Events Calendar

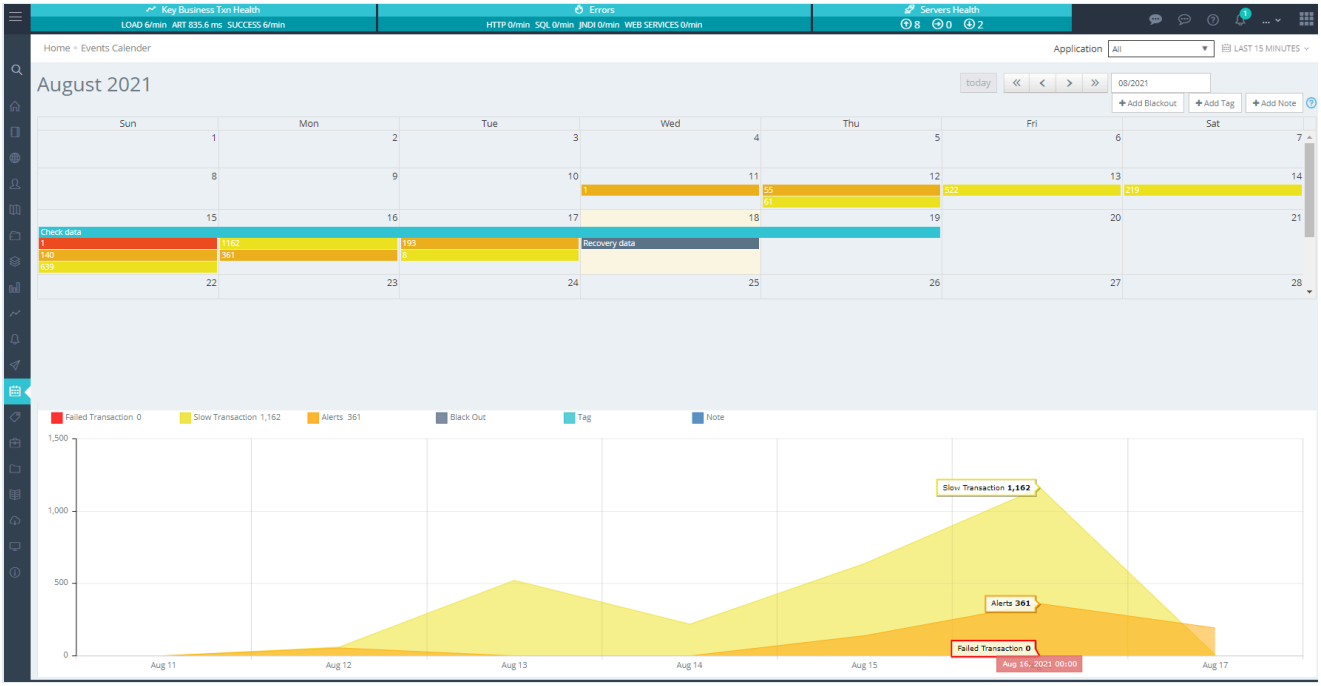
Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Overview

Events Calendar provides a monthly overview of entire enterprise's performance or an application's performance in a calendar format. Here you easily see how the app performance was for a particular day or for the entire month. You see number of alerts, failed transactions, slow transactions etc on a particular day and for the entire month. This way you know how things are improving

- Reduction in number of alerts over time
- Transaction's performance improving
- Reduction in number of failed transactions
- Reduction in number of slow transactions

You can add notes on any day to keep track of relevant data such as an outage occured on Aug 18 due to an app update or why that outage occurred. This way you don't have to dig root cause details/discussions/notes in your email box, they can all be available here.



Monitoring with Applicare

USING APPLICARE › DASHBOARDS

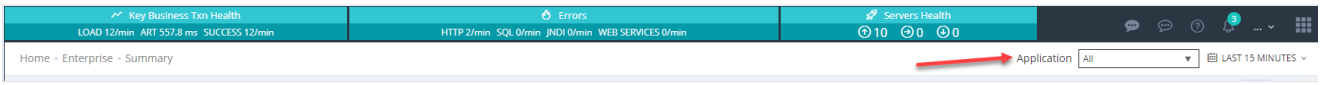
Enterprise

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Applicare Home

The Applicare home page displays the health and performance of the monitored applications. At top is the application selector, time selector, transactions score card and servers health status.

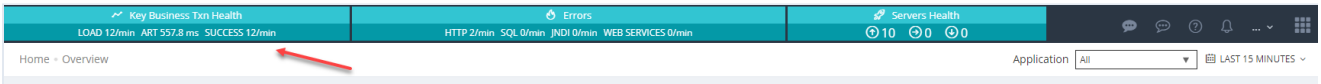
The application selector dropdown in the top right allows selection of an application to focus the scope of the analyzers. A server is assigned to an application at the time of adding it to Applicare. Application assignments can be managed from 'Overview' tab -> servers. A double click on an existing server allows editing the assigned application. Adding a server to an application allows grouping of data.



The time period selection on the top bar lets you narrow the focus to a specific time range.



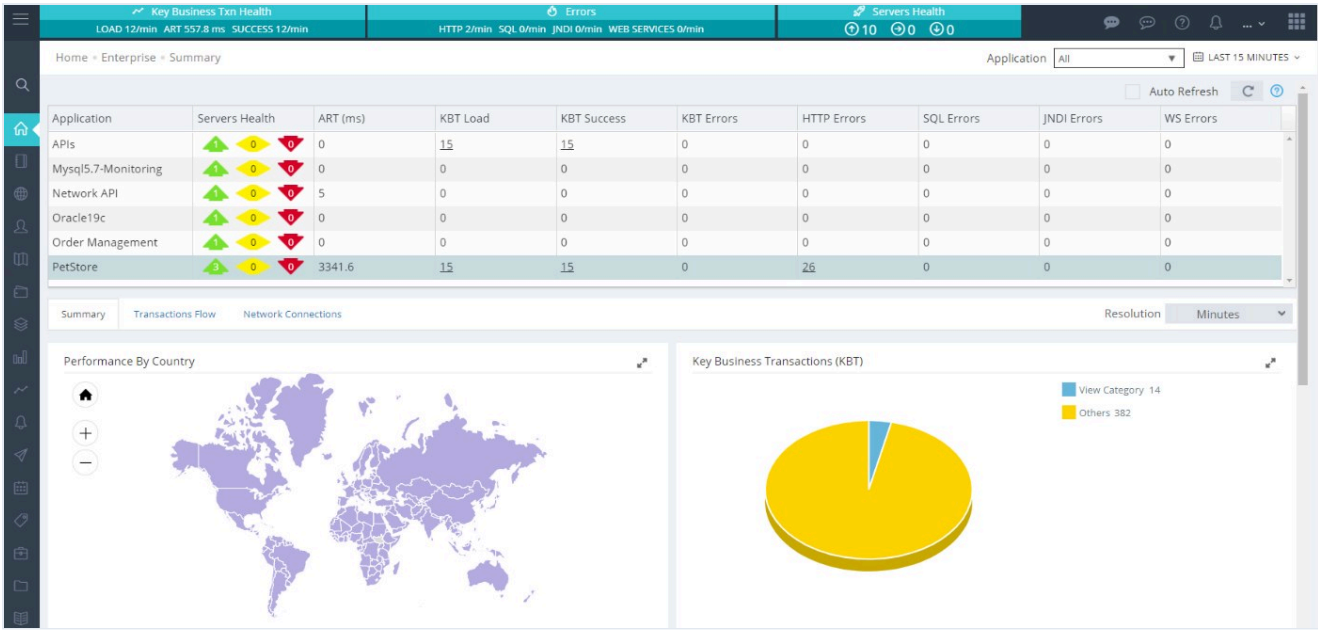
The **Business Transactions** & **Servers Health** shows the data for the selected application or all applications if 'All' is selected in application selection dropdown on left.



Below the top scorecards and selectors are various tabs that provide visibility into different aspects of the enterprise applications.

Enterprise Overview Dashboard

The **Enterprise Dashboard** provides a quick overview of the health of enterprise applications. It displays the average response time, business transactions, business transaction's load, success/errors, page load performance and health of monitored servers.



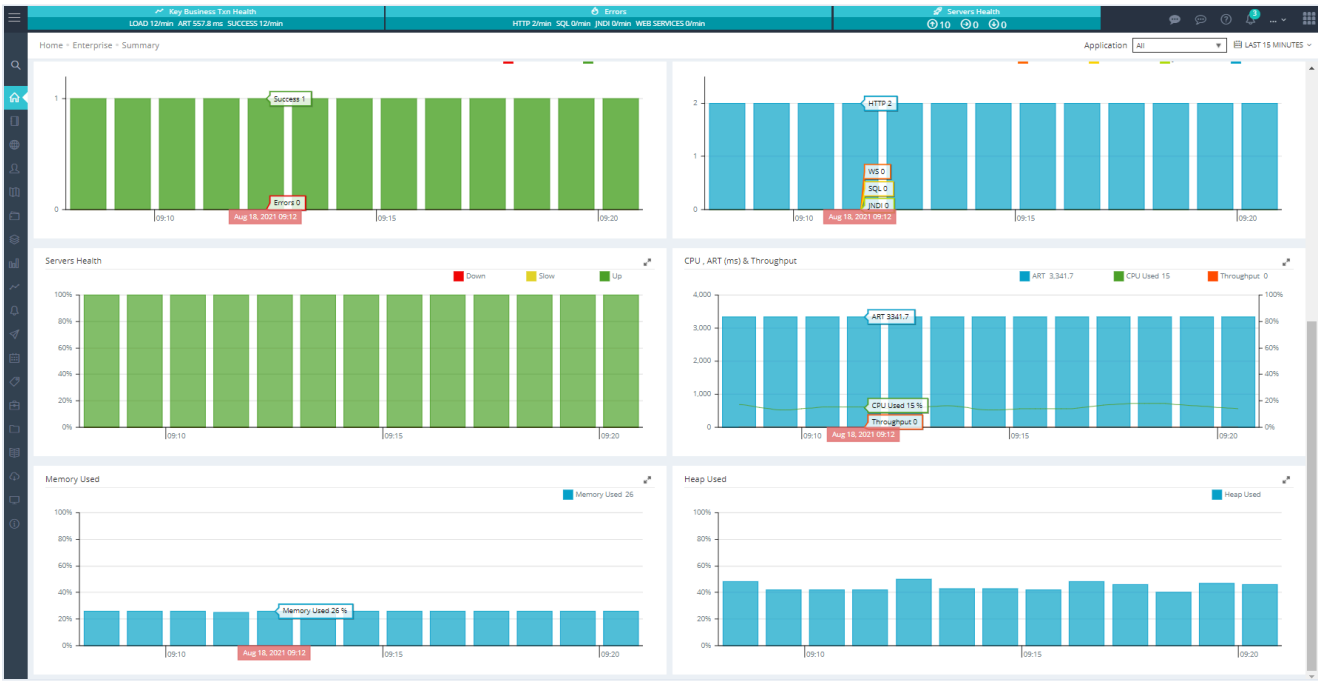
The **Enterprise Dashboard** offers a convenient to way identify performance issues and drill down into the details as needed:

- **Increase in average response time (ART)** for applications

- **Business transactions:** load, success rates, failures

- **Page Load Performance:** to show the user's experience from around the globe

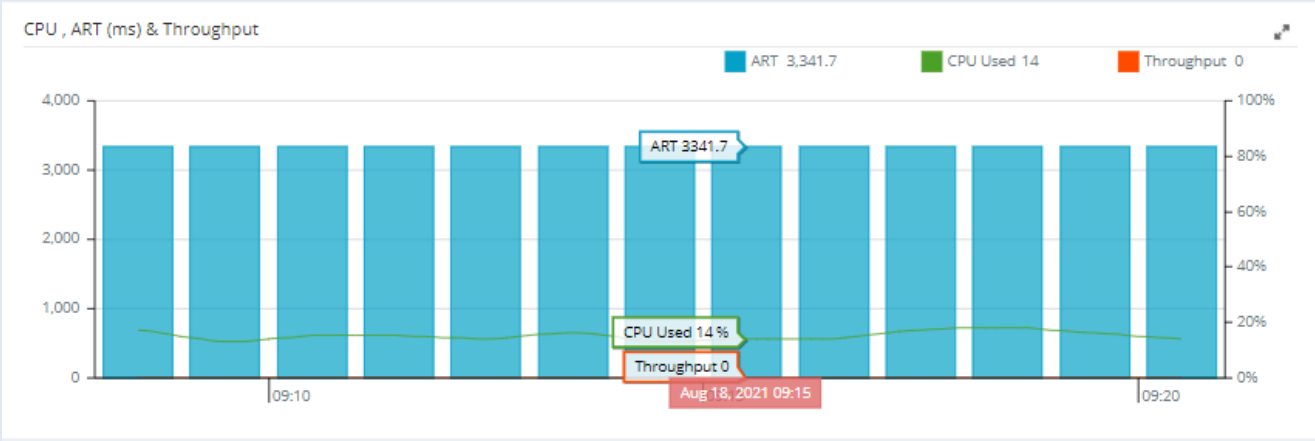
- **Servers Issues:** becoming slow or going down



On the Enterprise Dashboard, the top table allows you to select an application. Double-clicking an application table switches to the 'Transaction Flow' tab, which displays the flow and performance at each leg of the application.

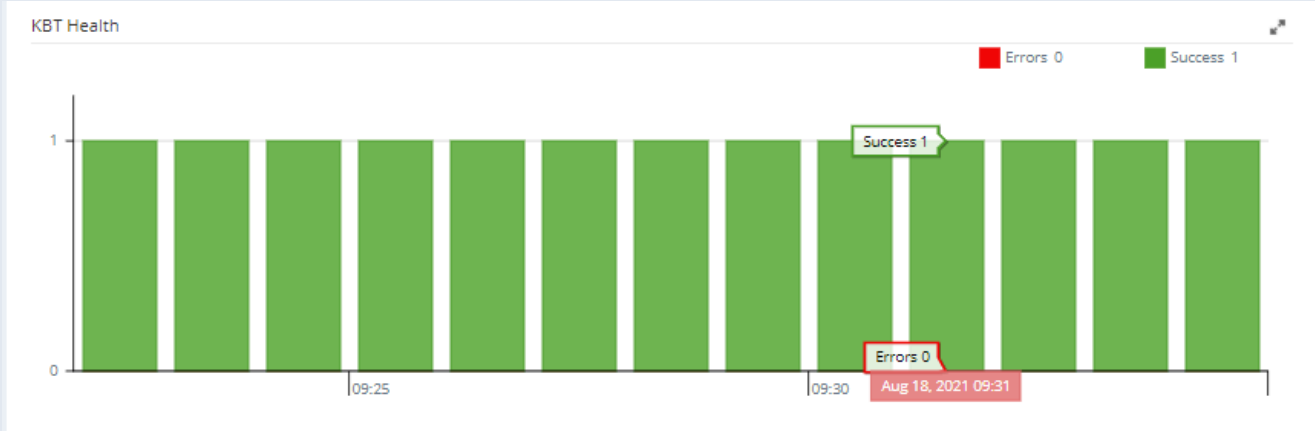
What is the average response time for the app and how does it vary across each participating server?

Select the bar indicating an increase in ART and it will display a list of servers in the selected app along with their respective ARTs, making easy to identify any server/s has/have slowed down.



Which business transactions are being invoked, and what is their performance?

The Business Transactions Health section displays load, success rates and errors and provides the option to drill into detailed information.

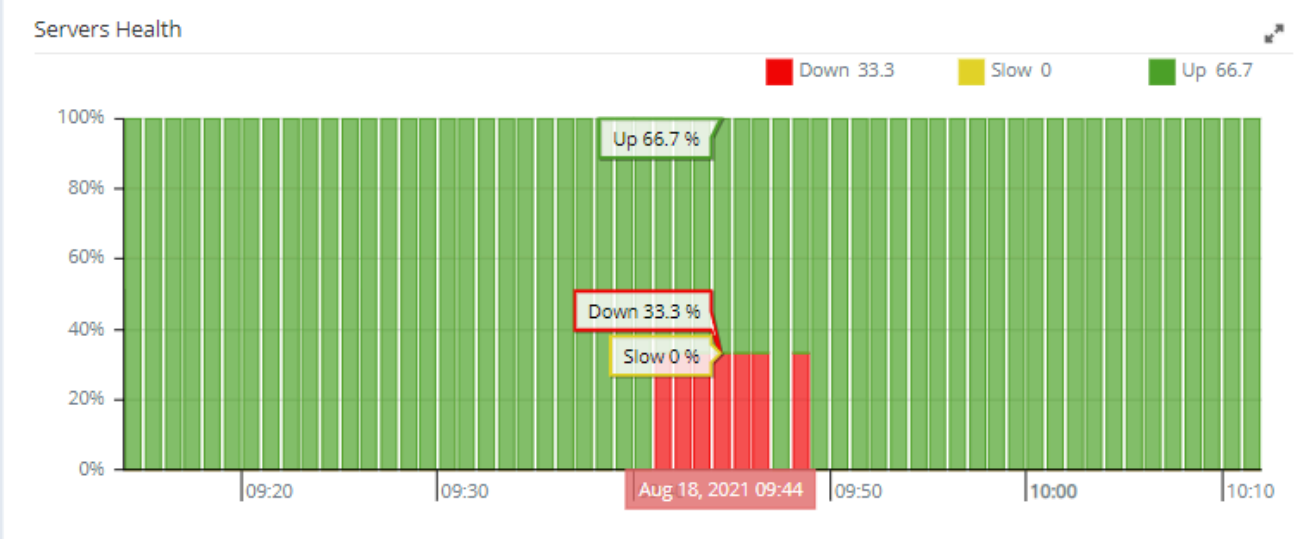


Where are the users located, and how is their experience with app?

Performance by Country/City provides insights into end user's experience with the selected application, including average app server response times, network latency, and browser's page rendering times. It also allows drill down to 'User Experience' tab for detailed information on individual slow transactions.

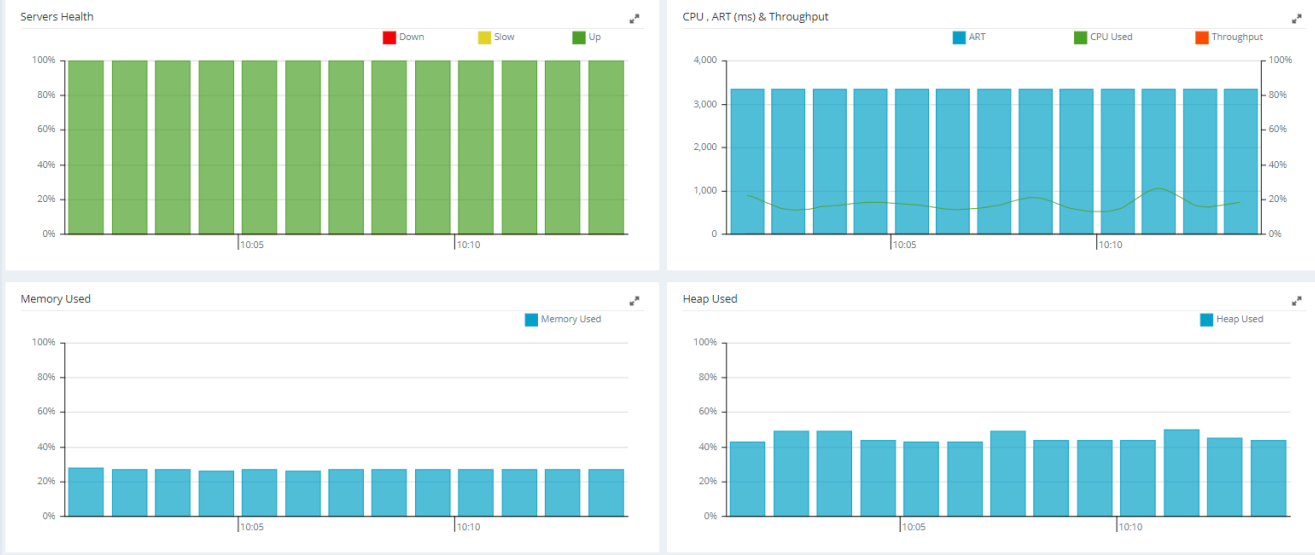


What servers are currently slow or have experienced downtime?



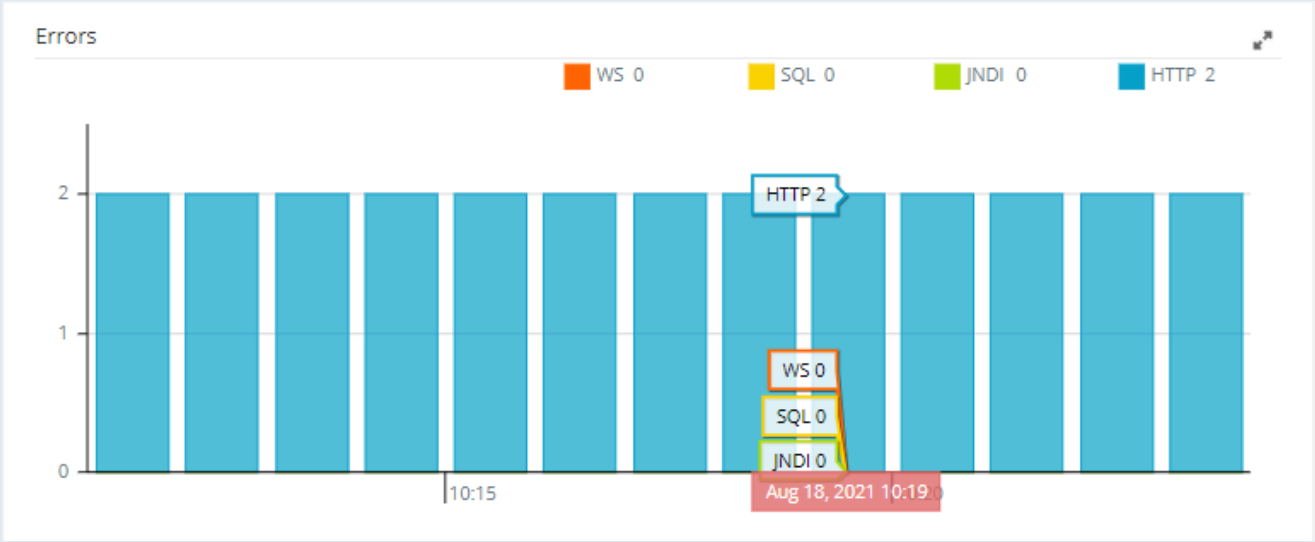
How is my application doing in terms of capacity?

The Enterprise Dashboard offers a clear view of an application's performance with respect to CPU, memory and heap usage. The three charts in the bottom right display cumulative usage by all servers within the selected application. i.e. in the JPetStore application shown below, the three servers each use 6% CPU, 26% memory, and 51% heap.



What is the transaction flow of my app and where are the errors occurring?

In the Enterprise dashboard, the **Network Connections** section displays the connections made by application's member servers. Clicking on server name hyperlink in bottom left table will provide detailed information about these connections, including their endpoints and current state.



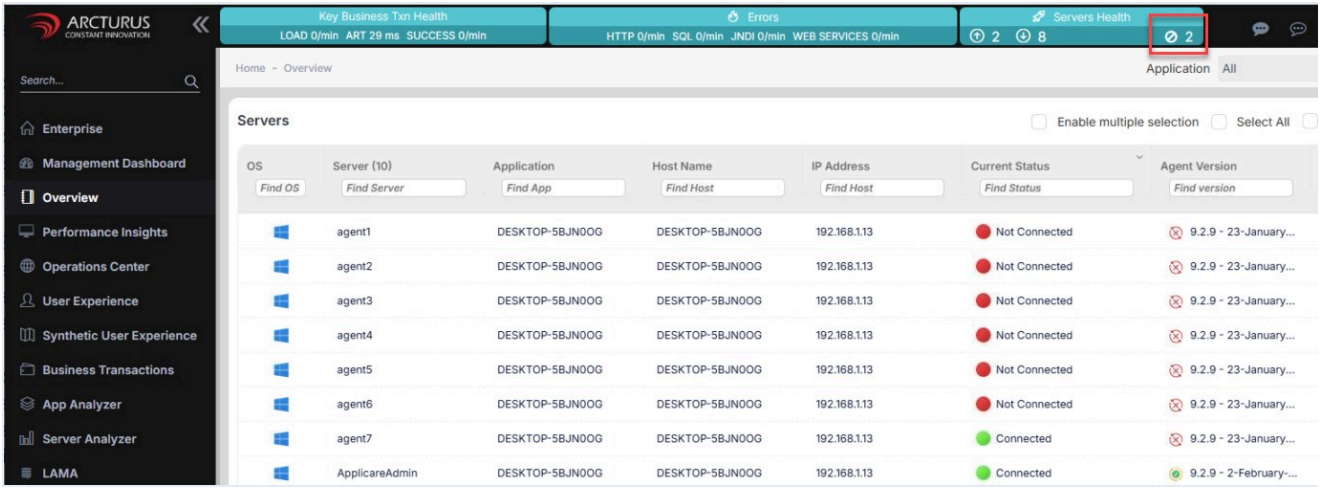
[Monitoring with Applicare]

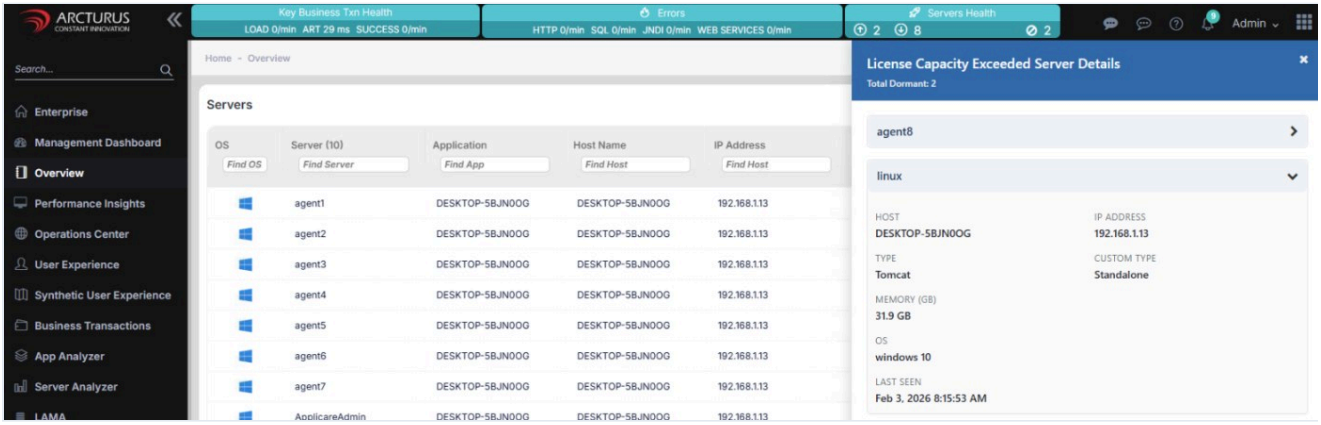
USING APPLICARE › DASHBOARDS

Unattached servers

Applicare 9.2 · Updated February 4, 2026 · **Pending technical review**

This feature displays the details of agents who attempt to add on the Applicare after reaching the allocated agent license limit. This helps administrators identify and monitor additional agent registration attempts when licenses are unavailable.





USING APPLICARE › APPLICATIONS

Log Analyzer

Applicare 9.2 · Updated January 21, 2026 · logs · **Pending technical review**

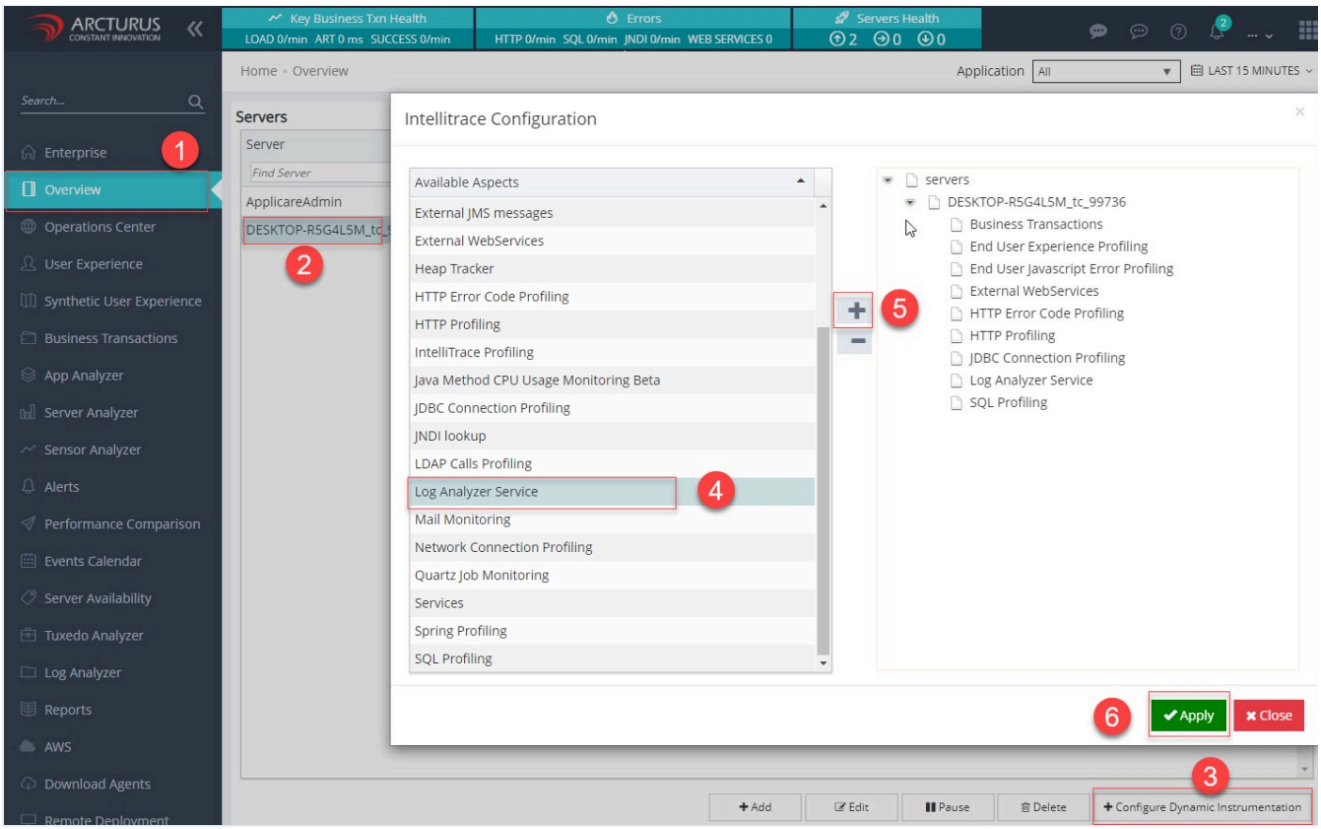
Applicare log monitoring allows you to search and find information spread across log files from applications and servers through a single, intuitive interface. Its seamless integration with IntelliSense and Business Transaction Monitoring allows you to quickly focus on relevant sections of logs files for any transaction with just a few clicks.

Configuration

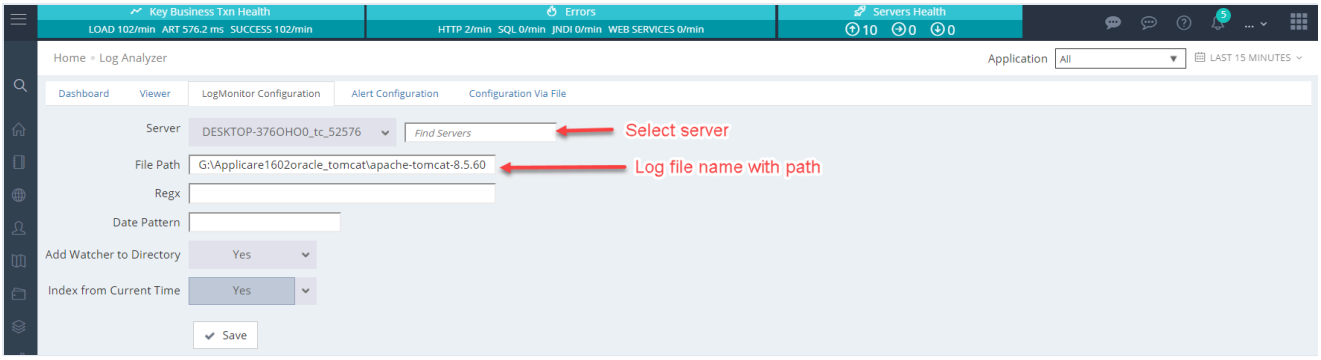
Applicare can monitor any log file accessible by a monitored server / agent. You can provide the absolute path to the log file or a directory containing log files. If the provided path is a directory, all log files found in the directory will be monitored.

By default, the **Log Analyzer Service** aspect is available for all agents. If it is not available, follow these Steps to add that Aspect.

Step 1: Go to **Overview Menu,** select the server, click the **Configure Dynamic Instrumentation** button and add the **Log Analyzer Service** to the server.



Step 2: Navigate to the **LogAnalyzer Menu** -> **LogMonitor Configuration** to add the logs.



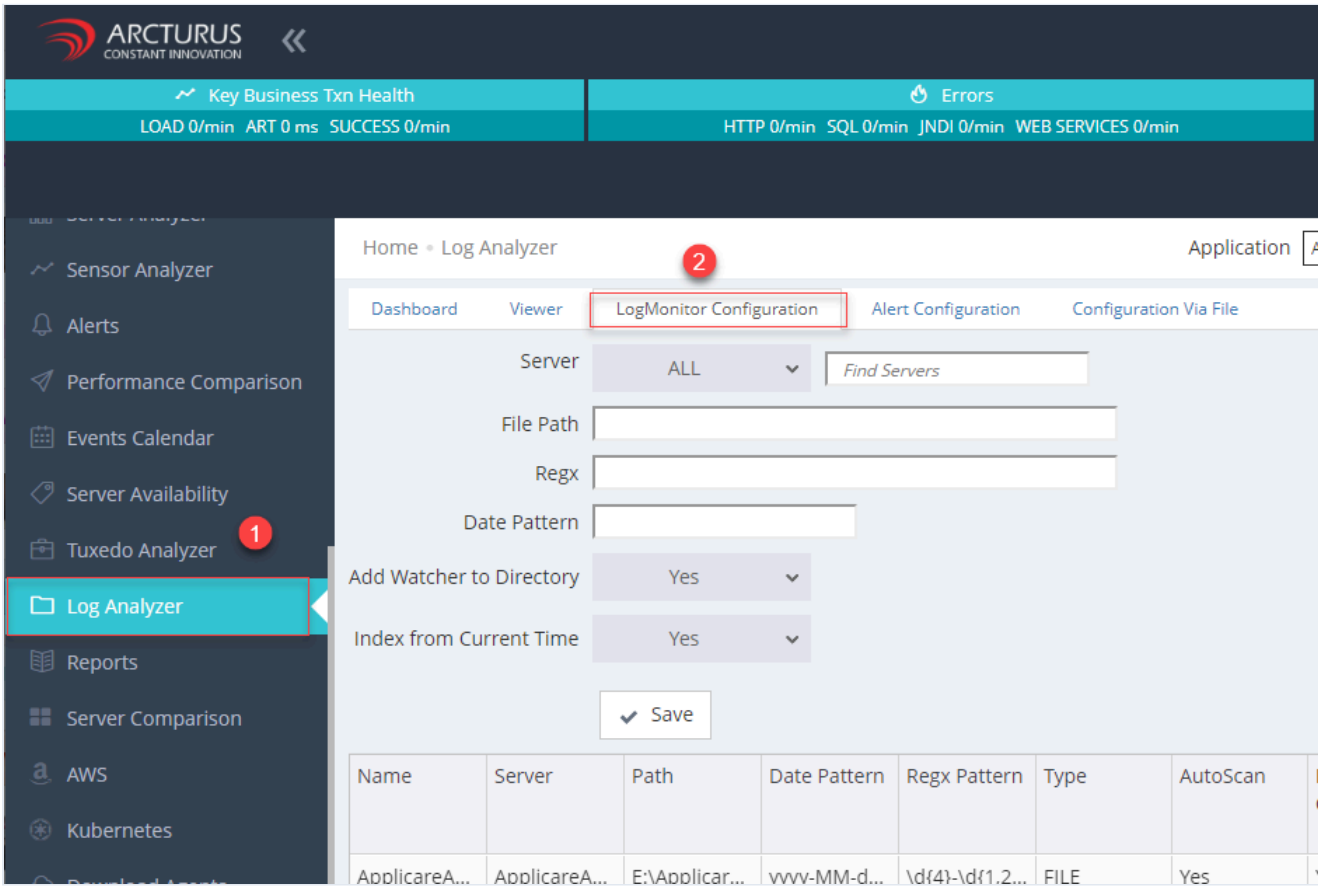
Example: Assume the Applicare Agent is installed in the **D:** drive and you want to add the Applicare Agent log for monitoring.

1. Select the **Applicare Agent name** from the **Server** dropdown.
2. Copy the Applicare Agent log path.
3. Paste the path into the **File Path** field.

Example log path: `D:\ApplicareSingleAgentWindows\logs\IIS.log`

Log Monitor Configuration

1. Go to **Log Analyzer - Log Monitor Configuration** menu.



2. Enter the details and click the **save** button to save the changes.

Server - Select the Server to which we are going to add the logs.

File Path - Either you can provide the directory path or file path.

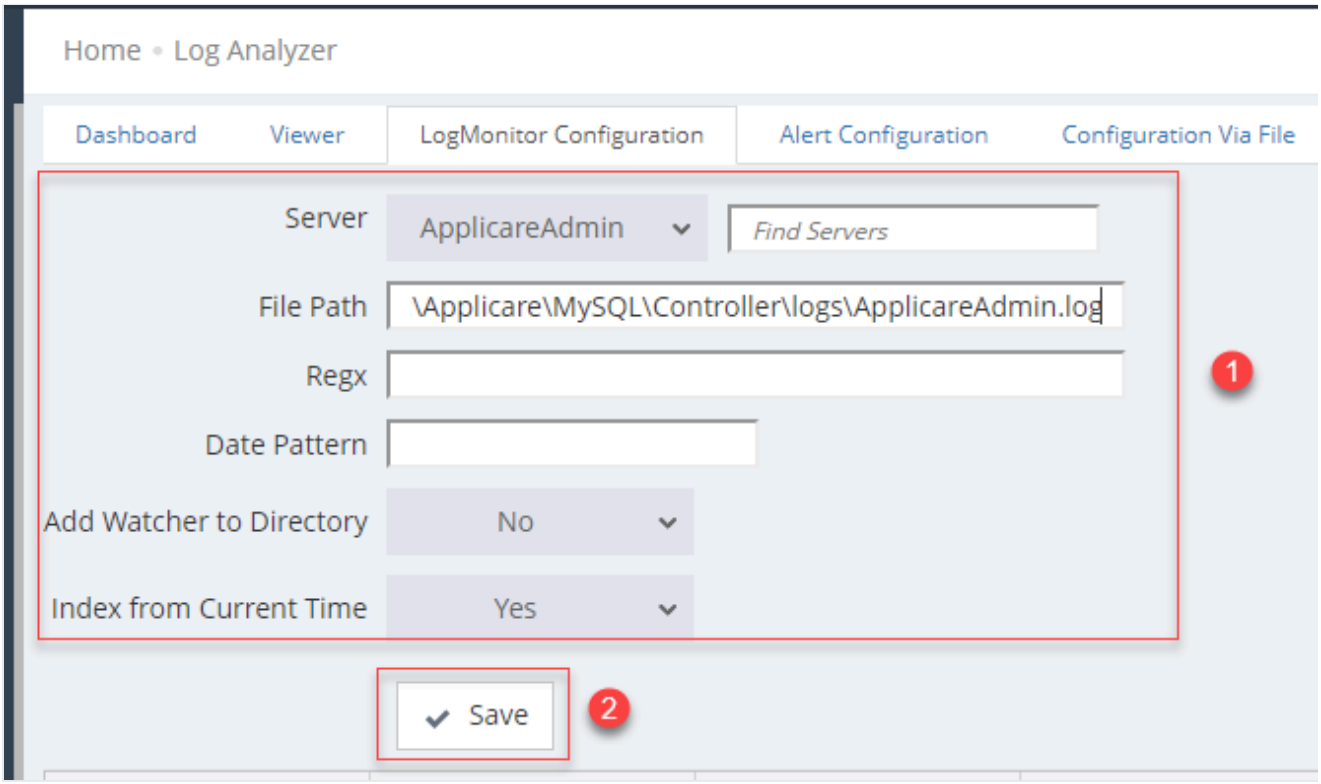
Regex Date Pattern - Leave empty.

Add watcher to Directory - Applicable for directories.

- **Yes** - The system will automatically monitor any new log files created in the directory.
- **No** - The system will not monitor new log files in the directory.

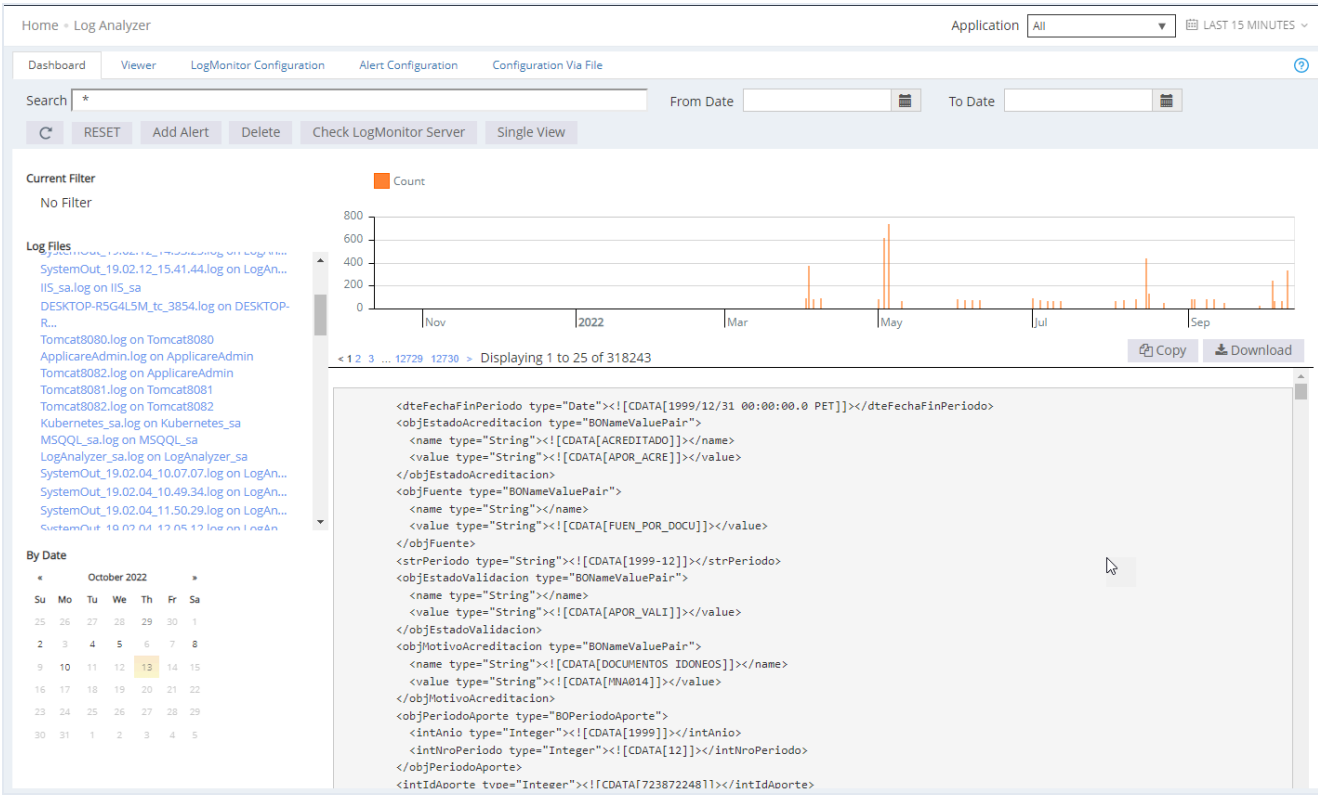
Index From Current Time - Choose whether to start monitoring logs from the current time.

- **Yes** - Monitoring will begin from the current time.
- **No** - Monitoring will include both new and existing records.

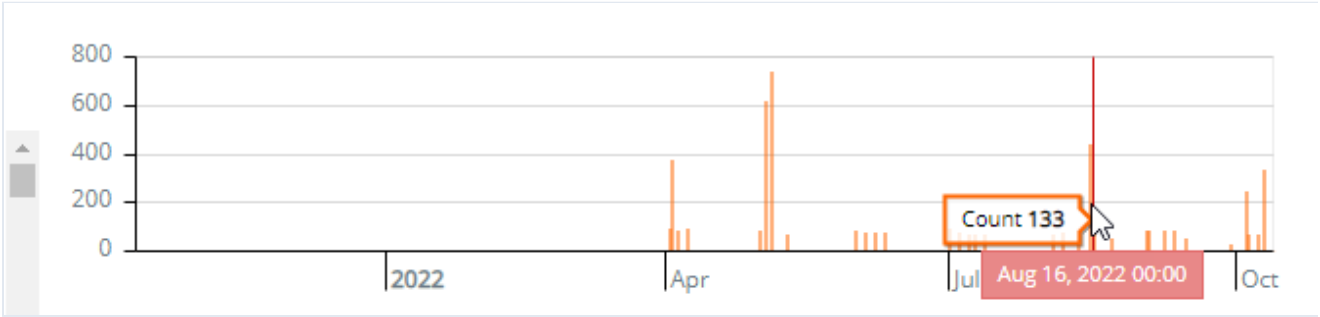


Dashboard

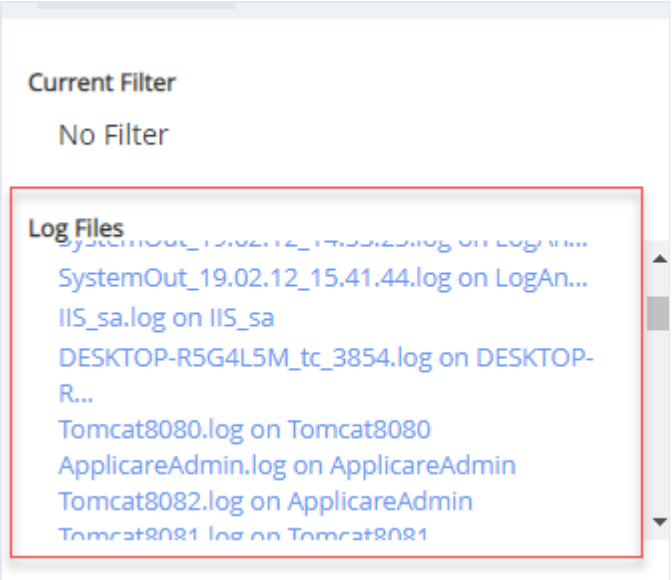
With the Log Analyzer, you can search through all your logs files from a single UI. You can refine search results by specifying a date/time range and further filter by log file or date using the dynamic filters on the left. By default, records are displayed in descending order based on the log date and time.



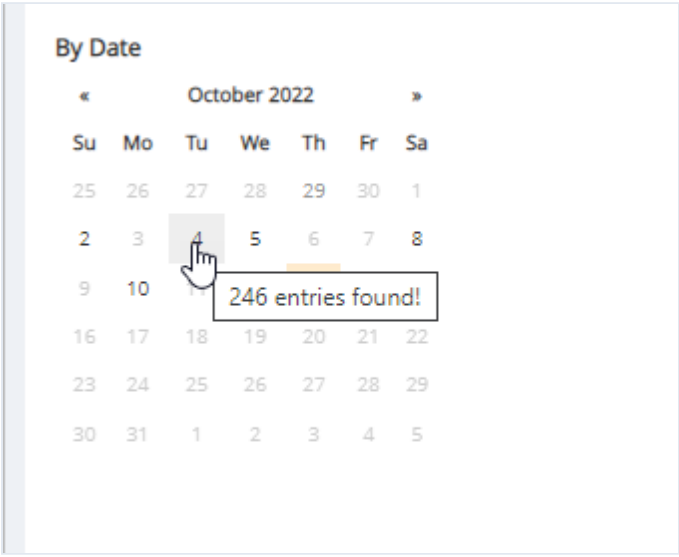
- **Graph** - Displays the count of records per day.



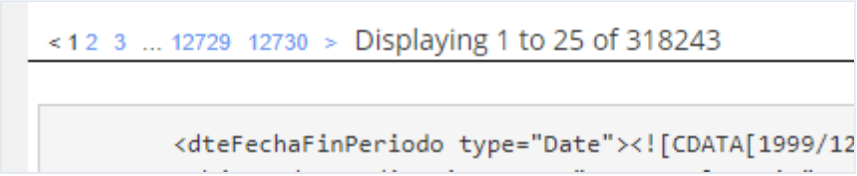
- **Log Files** - It will display the log file path and names based on the search result. By clicking on the Log file name it will filter the records available in that particular log files.



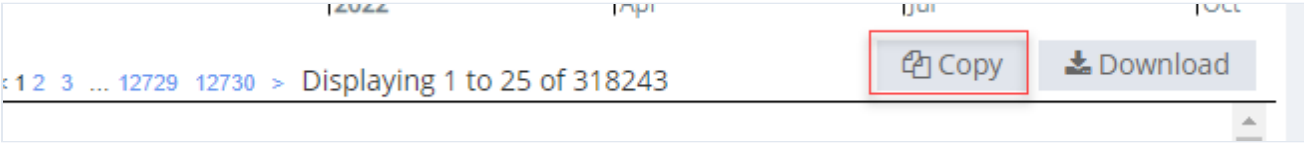
- **Calendar** - The calendar highlights days with log records in bold. Hovering over a highlighted day will display the count of log records for that day.



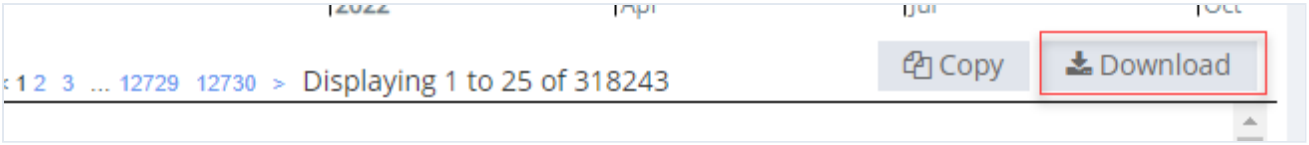
- **Pagination -** By default, 25 records are shown per page. Use the page numbers to navigate to the pages.



- **Copy button** - Copies the records currently displayed on the page.

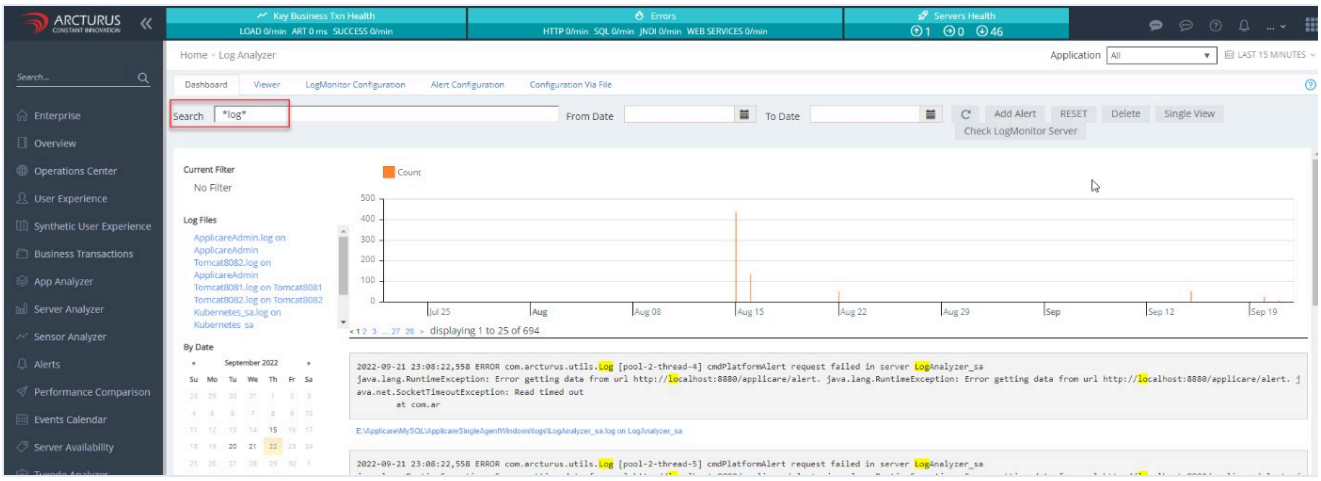


- **Download button** - Downloads the records currently displayed on the page as a text file.

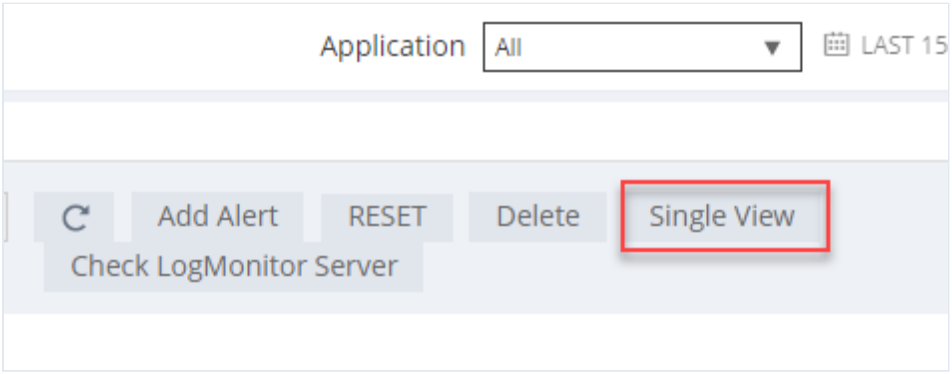


Single View

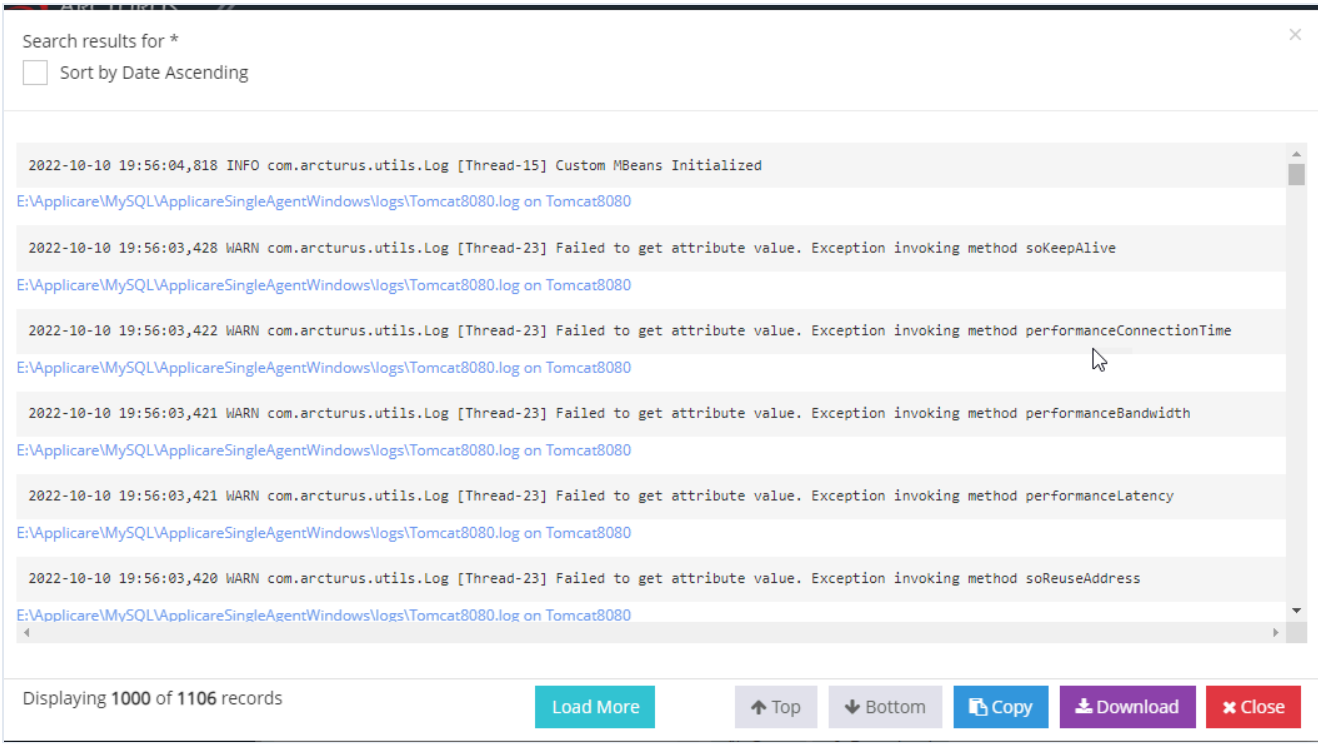
1. Enter your **search query** in the field and press Enter to search the records.



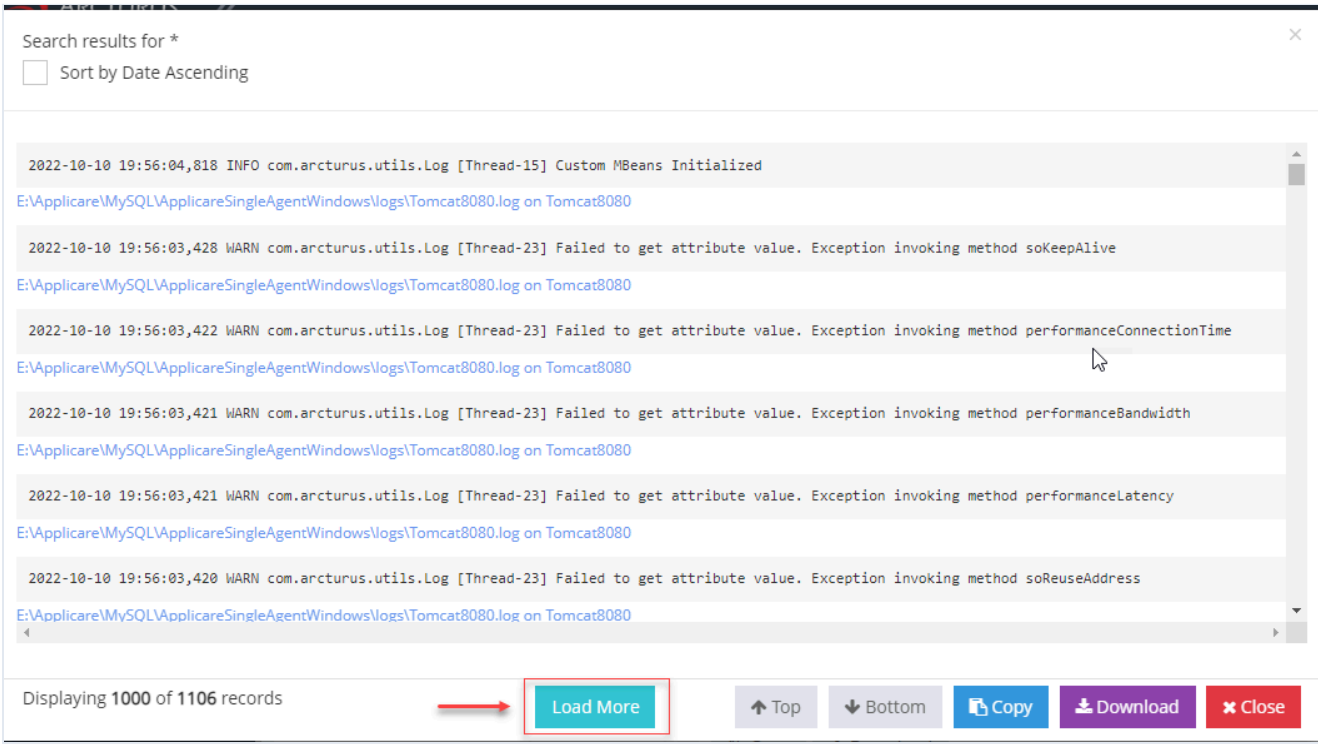
2. Click the **Single View** button.



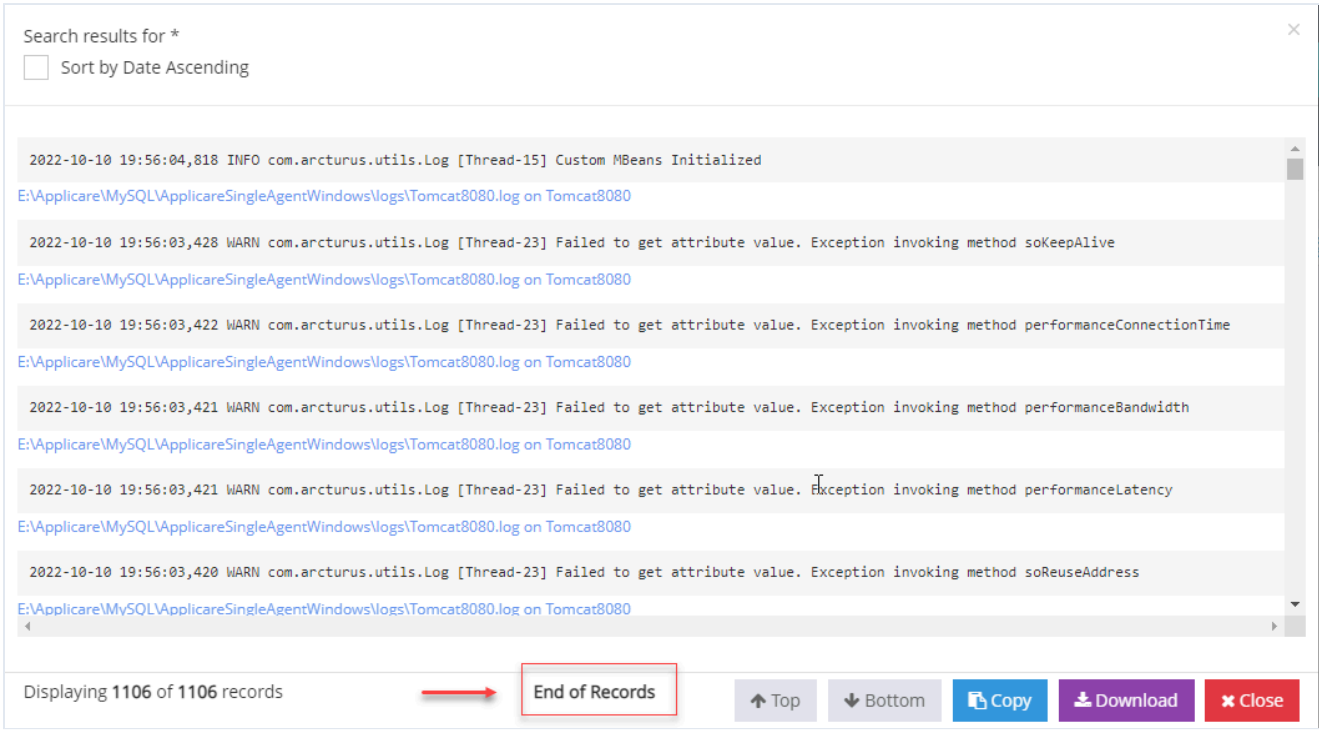
3. A **popup** will appear, Displaying up to **1000** records by default.



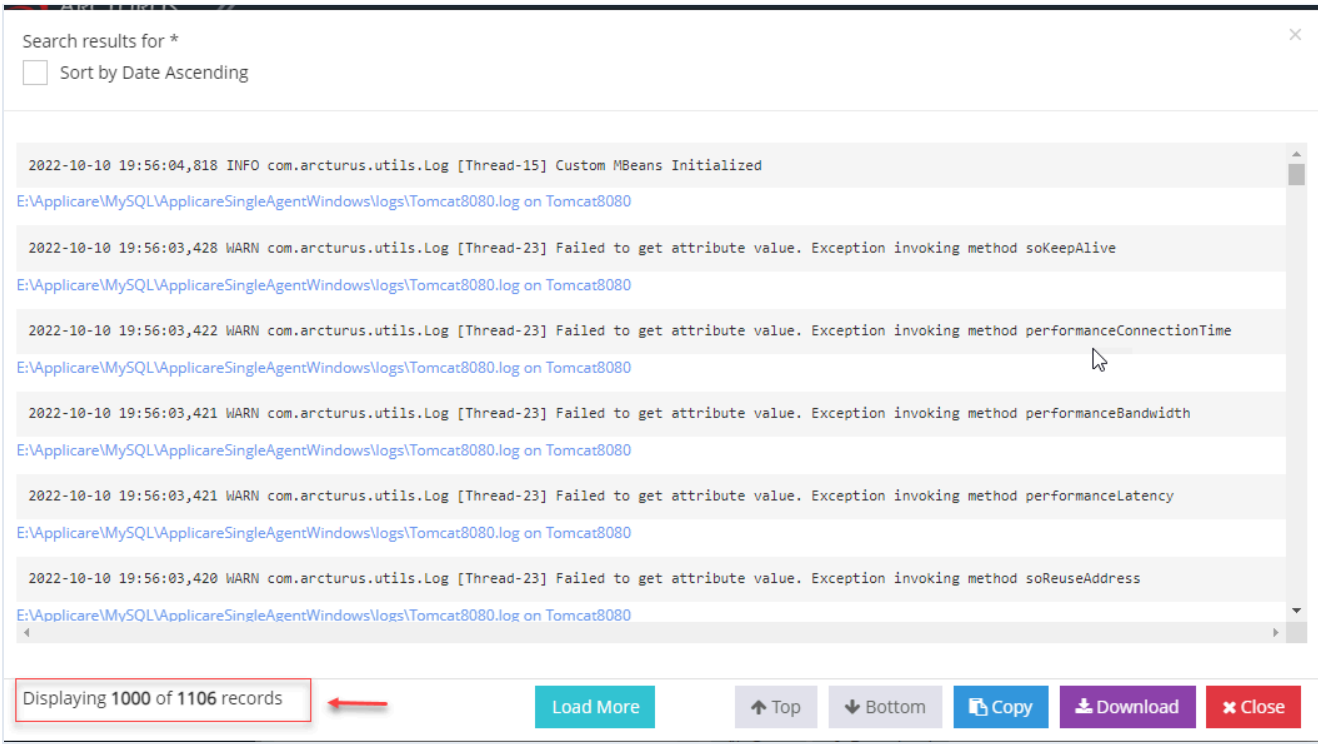
4. To load additional records, click the **Load More** button.



When all records have been loaded, you will see a message indicating **End of records**.

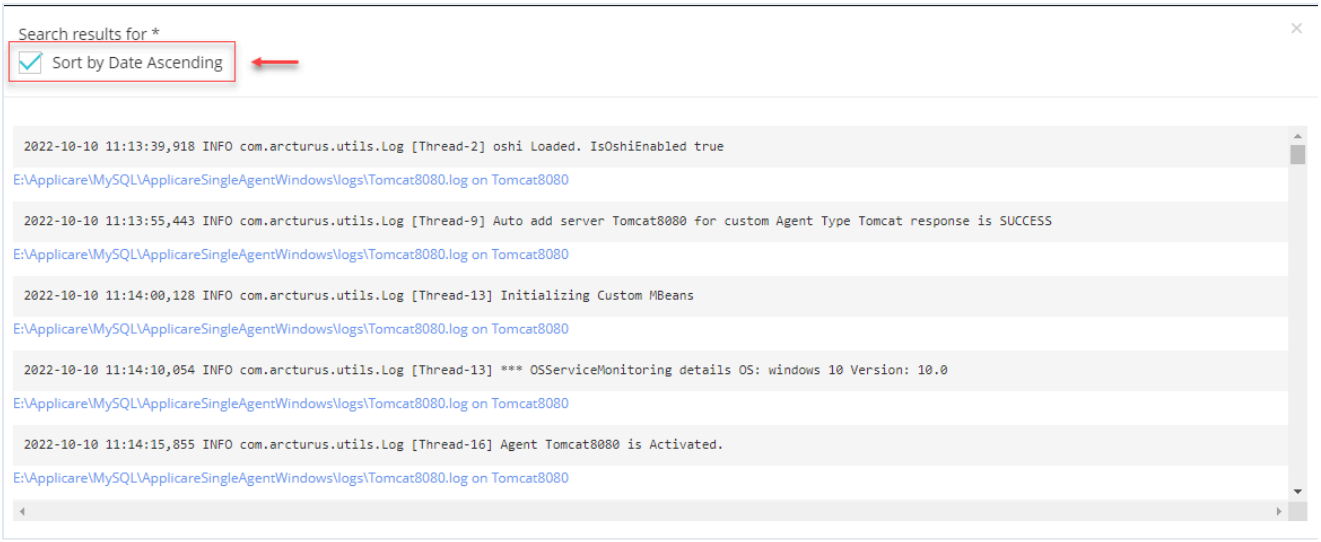


5. The **count** of loaded records will be displayed on the left-hand side.

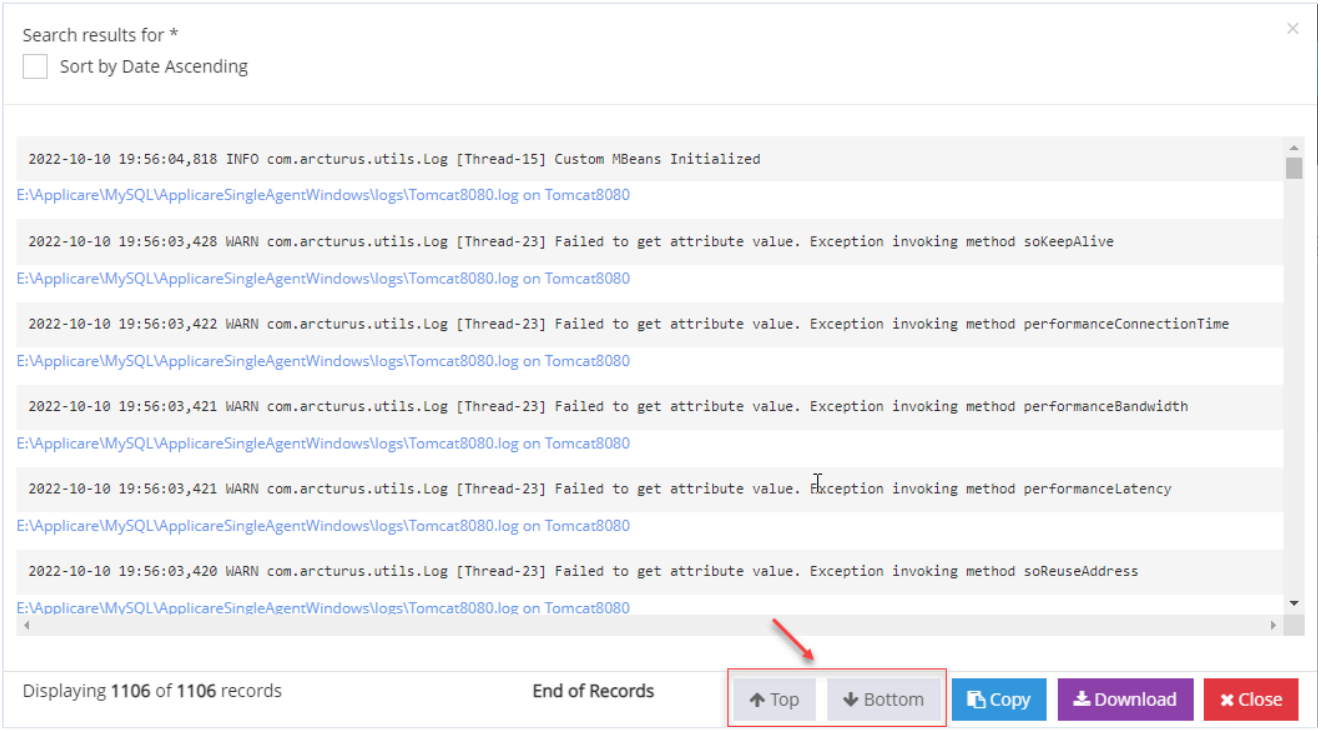


6. By default, records are loaded in descending order based on log date and time.

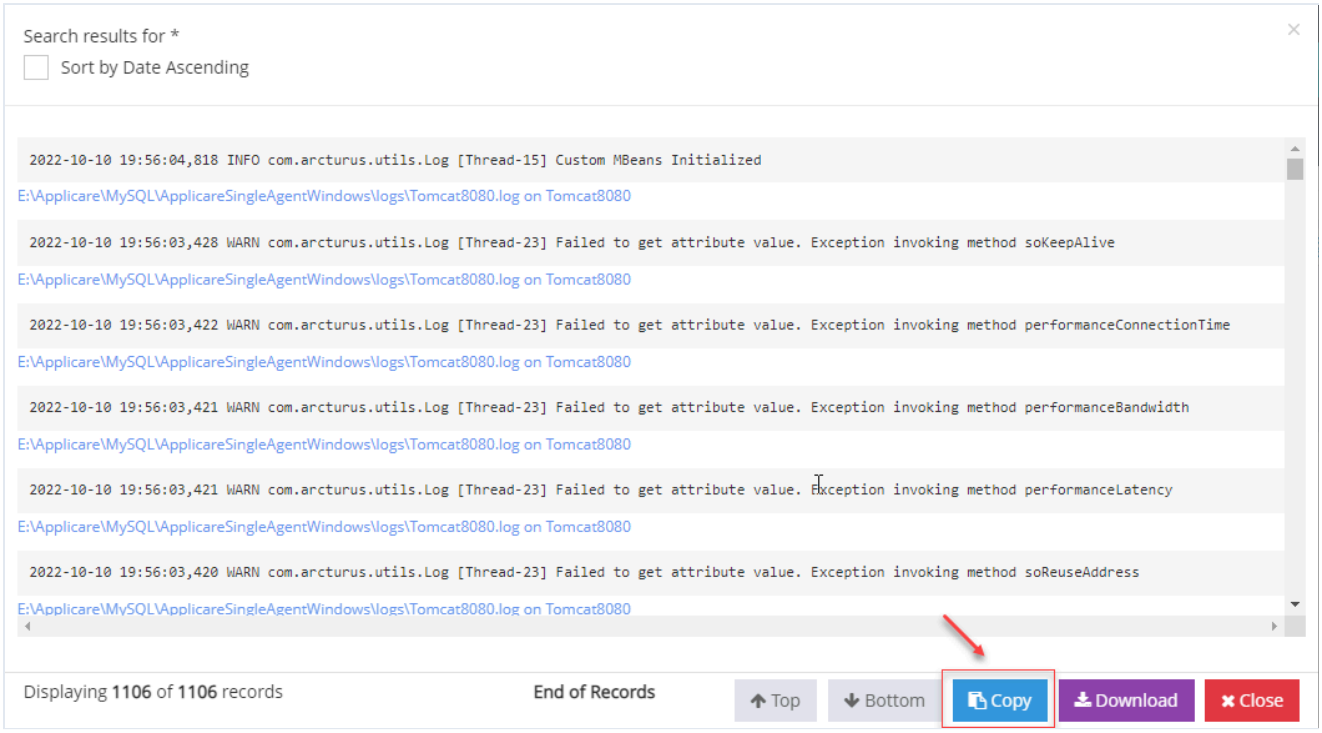
- To sort the records in ascending order, select the **Date ASC** check box.
- To sort the records in descending order, clear the **Date ASC** check box.



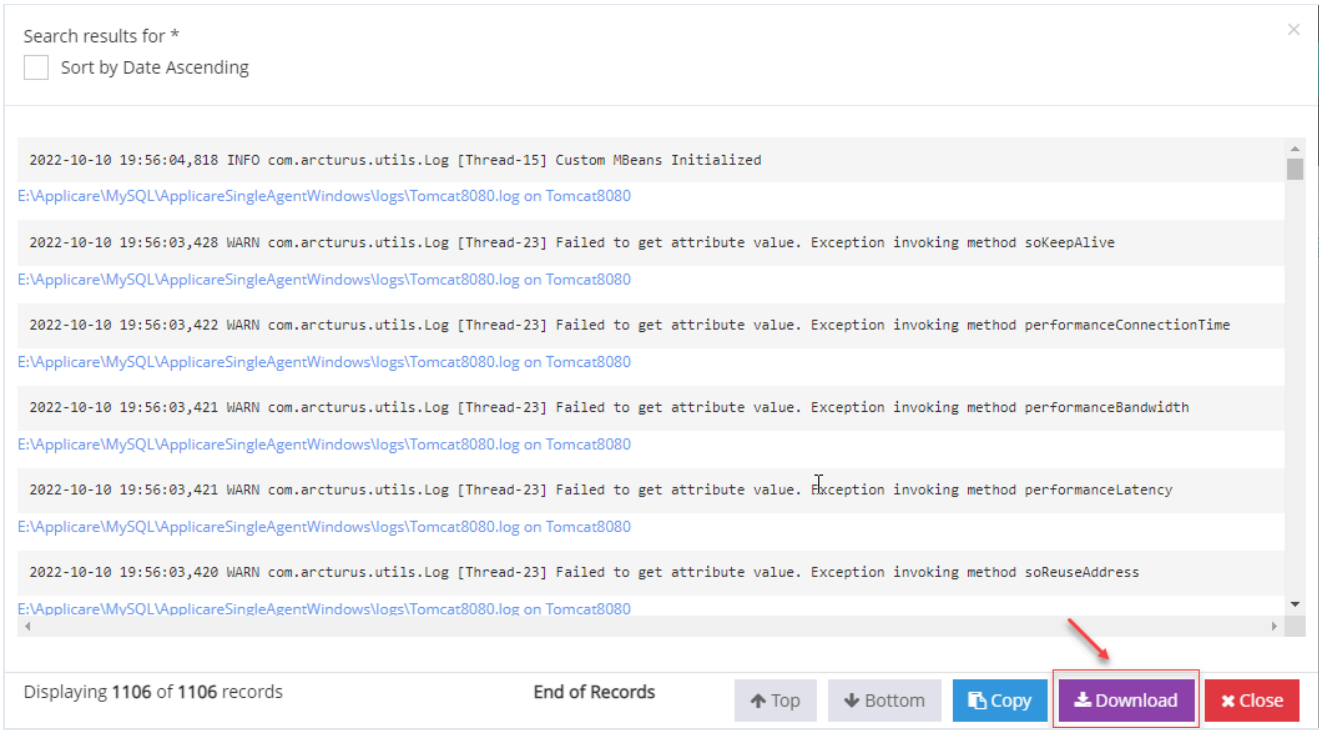
7. Click the **Top** and **Bottom** buttons to scroll to the top and bottom of the records. Clicking **Bottom** will not only scroll to the end but also load the next set of 1000 records.



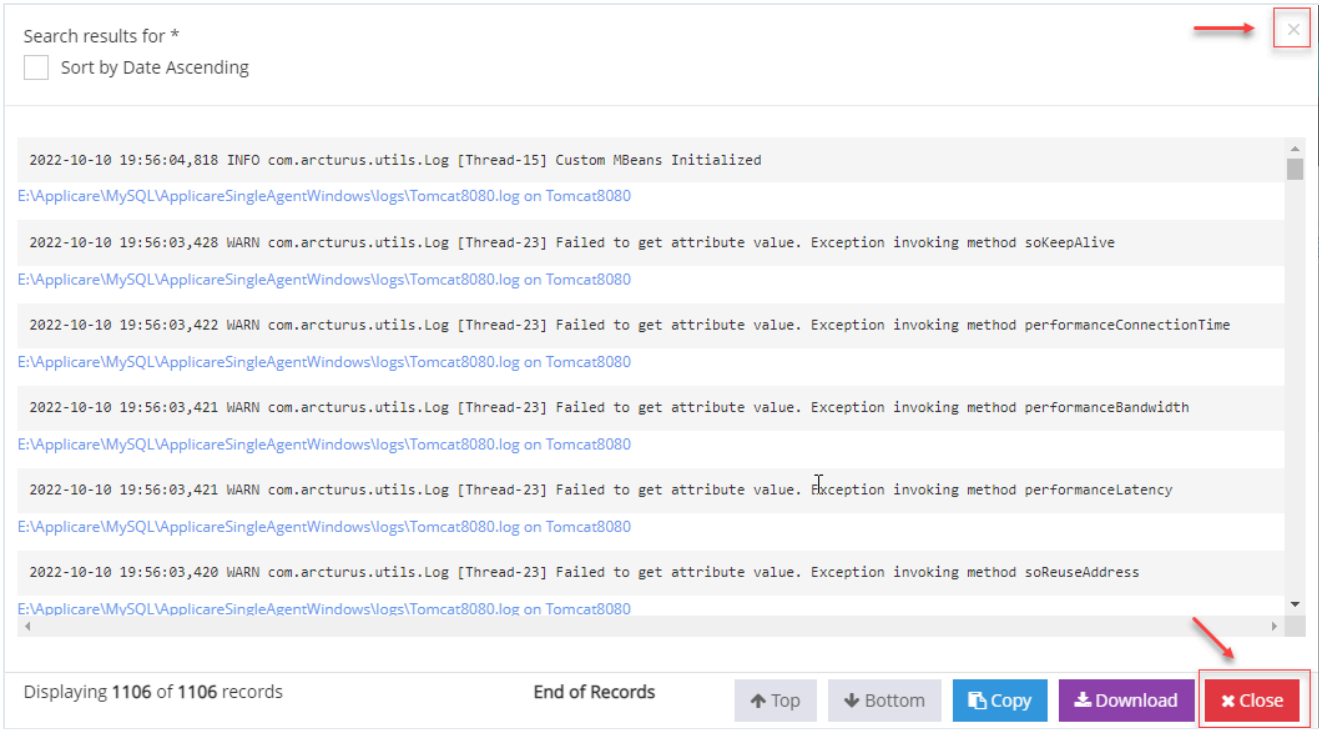
8. Click the **Copy** button to duplicate the currently loaded records.



9. Click the **Download** button to save the loaded records as a text file.



10. **Close Popup** - To close the popup, click either the **close button** at the bottom or the **close icon** at the top of the popup.



Log Analyzer configuration import via csv file

Date and Regx pattern format

Monitoring with Applicare

USING APPLICARE › APPLICATIONS

Alerts

Applicare 9.2 · Updated August 29, 2024 · alerts · **Pending technical review**

User Role and Alerts

Super Admin, Users with Admin rights or App Admin rights for an application can add/view alerts related to that application. User with read only access for an application can only view predefined alerts on the application.

Applicare automatically generate platform alerts for CPU, disk and memory utilization for the agents if it is configured in MySQL or Oracle database.

Configure Alerts

Applicare allows users to set alerts and receive email notifications based on custom conditions. It offers the following types of alerts:

Platform Alerts	**- Operating System and JVM metrics (CPU, Memory, Swap, Heap Usage)**
JMX Alerts	- Set up alerts based on MBean attributes
SNMP Alerts	- Create alerts for values on SNMP capable devices
Tuxedo Queue Alerts	- Set up alerts for Tuxedo Queue
Tuxedo Service Alerts	- Create alerts on Tuxedo Services
UEM Slow Transactions Alerts	- Create User Experience alerts for slow transactions
Slow Transactions	- Create alerts on slow transactions as detected by IntelliSense
Business Transactions - SLA Avg	- Create alerts on Business Transactions exceeding SLA Avg
Business Transactions - SLA Max	- Create alerts on Business Transactions exceeding SLA Max

Tools -> Manage Alerts enable you to create, edit, pause or activate alerts. It also allows you to assign a command to be executed when an alert is fired. Applicare allows any shell/command script execution on any location where Applicare agent is running. This is very valuable for doing diagnosis or taking remediation actions when a certain alert is generated.

For creating high CPU utilization alert select New and platform alert

> [Screenshot 1 could not be downloaded during import — see migration report]

Click on the browser icon next to the property and select CPU -> Usage.

The Alert Trigger Threshold * Alert Monitor Job Interval defines the time interval that condition should hold before generating the alerts. Alert Monitor Job Interval by default is 1 minute but can be adjusted by adding **applicare.alertMonitorJobInterval** (in minutes) property to Applicare.props file. In the example below alert will be generated if the CPU usage is higher than 90 for 3 consecutive polls by Applicare. Similarly in the example below auto clear threshold defines that CPU usage needs to be below 90 for 3 consecutive polls for generating clear alert.

> [Screenshot 2 could not be downloaded during import — see migration report]

Intellicheck Alerts Configuration

IntelliCheck alerts are paused for **intellicheckAlertsPauseTime** (in minutes) after # of alert reaches the **intellicheckAlertPauseThreshold** limit set in Applicare.properties. With the example values below 3 IntelliCheck alerts would come for JDBC resource not available and then alerts on the same JDBC resource would be paused for 30 minutes.

```
intellicheckAlertsPauseTime=30

intellicheckAlertPauseThreshold=3
```

By default, **intellicheckAlertsPauseTime** is set to 15 minutes alert and **intellicheckAlertPauseThreshold** is set to 5 alerts.

Email Notifications

To start the receiving email notifications for new alerts, you first need to configure the SMTP server details.

Navigate to **Applicare** > **Configuration** > **Configure Email/SMTP Configuration** from the top menu.

> [Screenshot 3 could not be downloaded during import — see migration report]

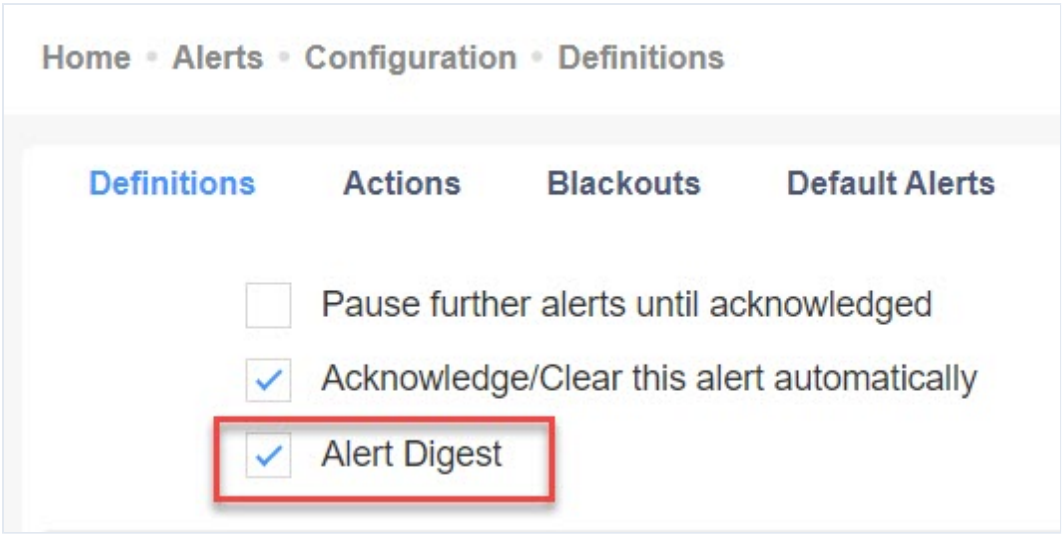
Mail To is the catch all email address. All Applicare alerts are sent to this email address.

To customize who gets what alerts select **Home** > **Alerts** (tab)> **Configuration** (wheel icon). Then select **Actions**.

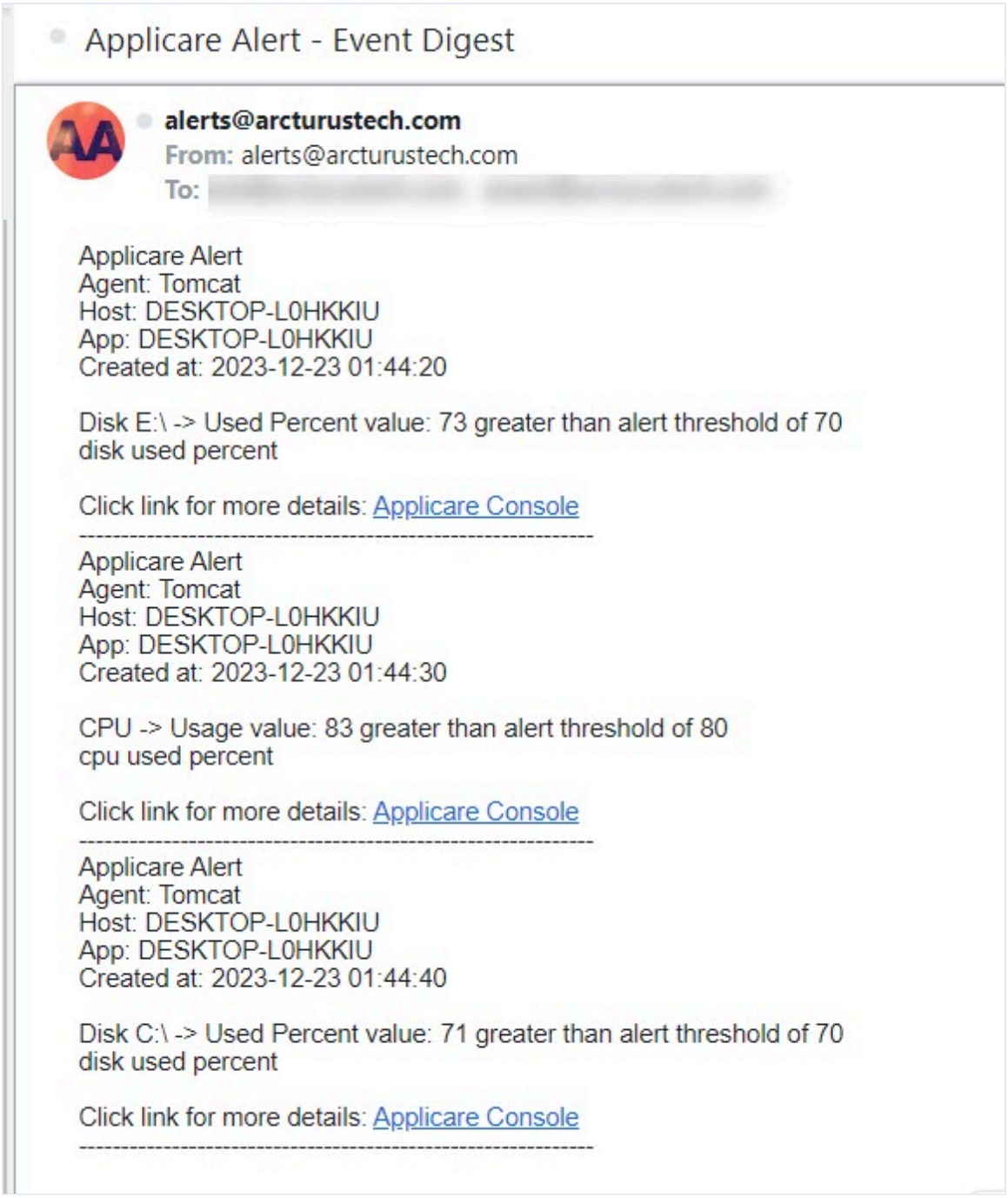
Under actions you can select users, add email addresses or email list addresses for receiving alerts on an application, a server or an individual alert. Applicare would send alert to emails/users defined for individual alert. If there are no specific emails/users defined for individual alerts, it would inherit the same from the server or the application it is part of e.g. If **portalLX** application alerts are sent to Larry and no other specifics are defined for servers that are a part of **portalLX** or individual alerts then all the alert emails for portalLX would be sent to Larry. Once a specific email address/s and users are defined for either server alerts or individual alerts then that overrides the application alert settings.

If you update the emails/users settings are changed for an application after defining setting for individual alerts or server alerts, they would be overwritten by the newly defined settings for the application.

Alert Digest



Applicare includes an alert digest feature that consolidates multiple alerts into a single email. This approach simplifies the management and readability of alerts, providing users with a more organized and user-friendly experience.



Advanced Options

From the configuration screen, you can enable or disable alerting.

Select **Applicare** > **Configuration** > **Advanced Options** from the top menu.

> [Screenshot 6 could not be downloaded during import — see migration report]

Using A Custom Alert Processing Class

You can create a Java class to which will be invoked for each alert created by Applicare agents by implementing the interface **com.arcturus.alerts.notifications.AlertAction** and the only method in the interface public void **processAlert(AlertQueue alert)**. You can get the details of the alert using the getter methods of AlertQueue object as shown in the example class below.

```
TEXT
package com.arcturus.alerts.notifications;

import java.text.SimpleDateFormat;

import com.arcturus.alerts.domain.AlertQueue;

public class SampleAction implements com.arcturus.alerts.notifications.AlertAction{

    public void processAlert(AlertQueue alert) {
        System.out.println("Server: " + alert.getServer()); //the server name where the alert
was generated
        System.out.println("Type: " + alert.getSubsystem()); //type of the alert (JMX_ALERT,
PLATFORM_ALERT, etc..)
        System.out.println("Priority: " + alert.getPriority()); //priority of the alert
        System.out.println("Message: " + alert.getMessage()); //user defined message in alert
definition (description)
        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        System.out.println("Created: " + dateFormat.format(alert.getCreated())); //time the
alert was created
    }
}
```

To deploy the class and any dependent classes, you can package it in a jar file and copy to **Applicare/server/ext** folder (create it if it doesn't exist) and restart Applicare server.

When compiling make sure to include the **Applicare/server/webapps/applicare/WEB-INF/classes** folder in your classpath.

After restarting the server, navigate to **Applicare** > **Configuration** > **Advanced Options.** Enter the fully qualified class name of your custom class in the "Custom Alert Processing Class:" field and click **Apply**.

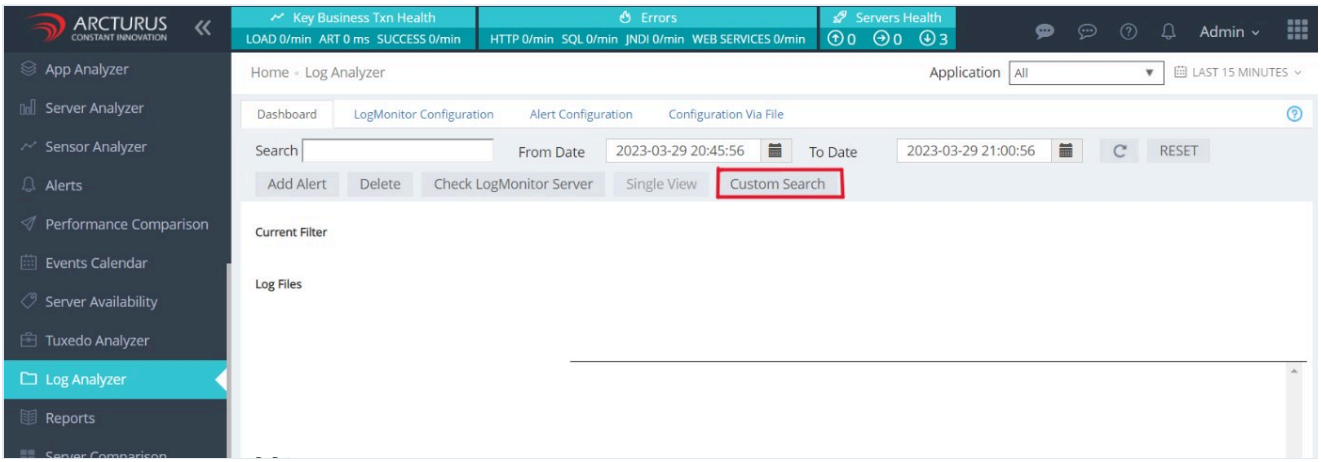
> [Screenshot 7 could not be downloaded during import — see migration report]

USING APPLICARE › APPLICATIONS

Custom Search in LogAnalyzer

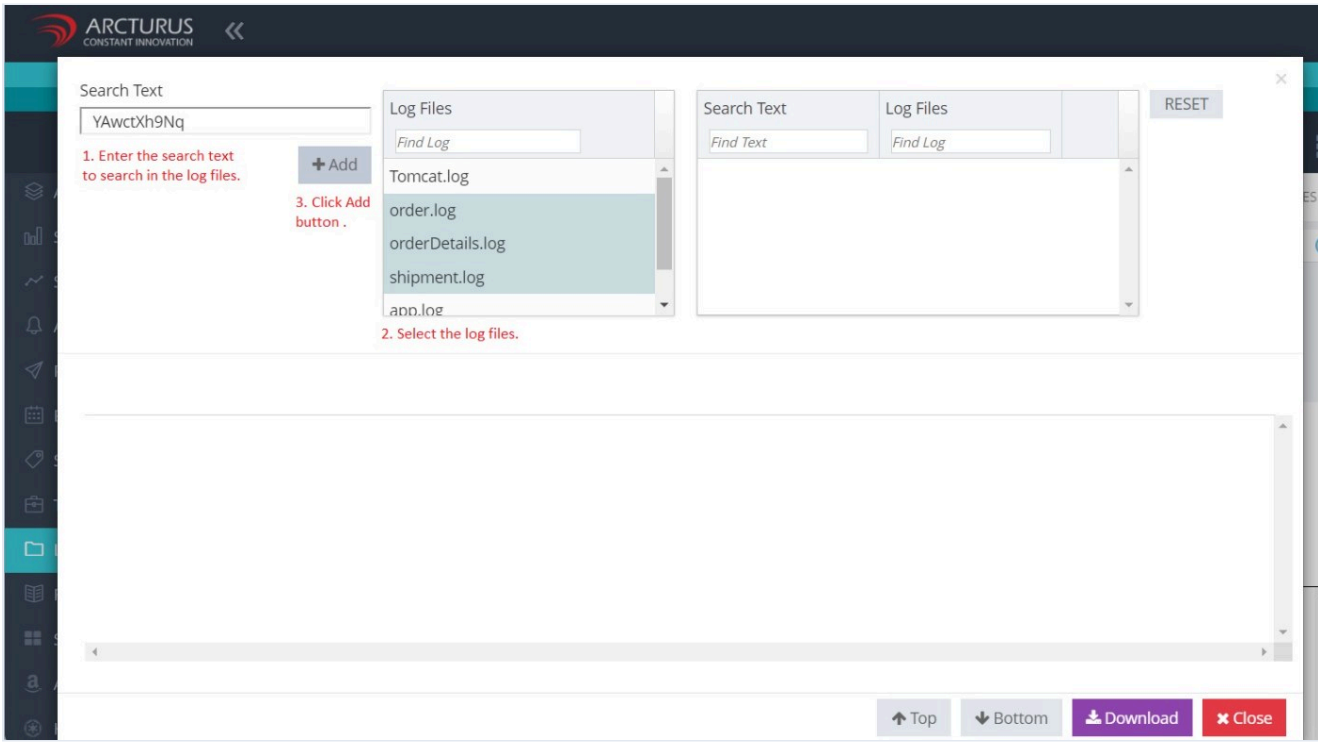
Applicare 9.2 · Updated September 13, 2024 · logs · **Pending technical review**

1. Go to **Log Analyzer** menu and click the **Custom Search** button.

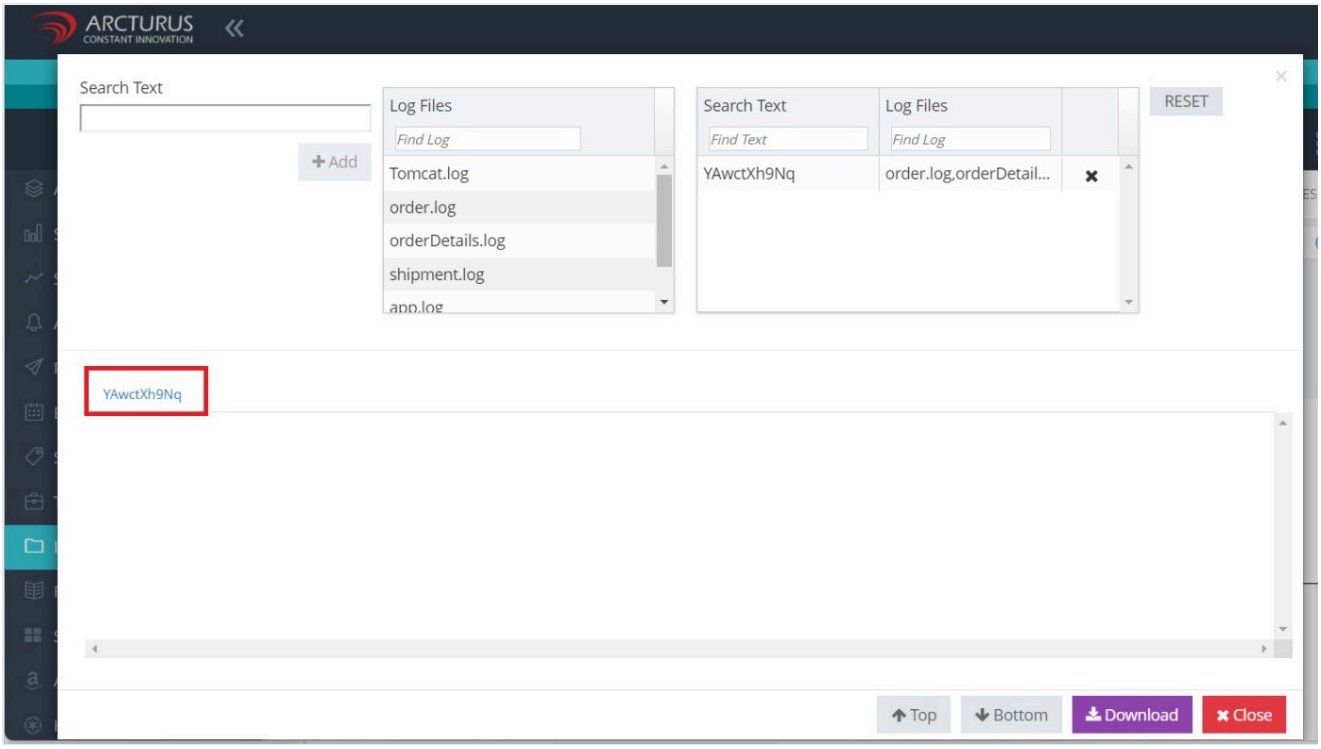


2. In the Custom Search popup

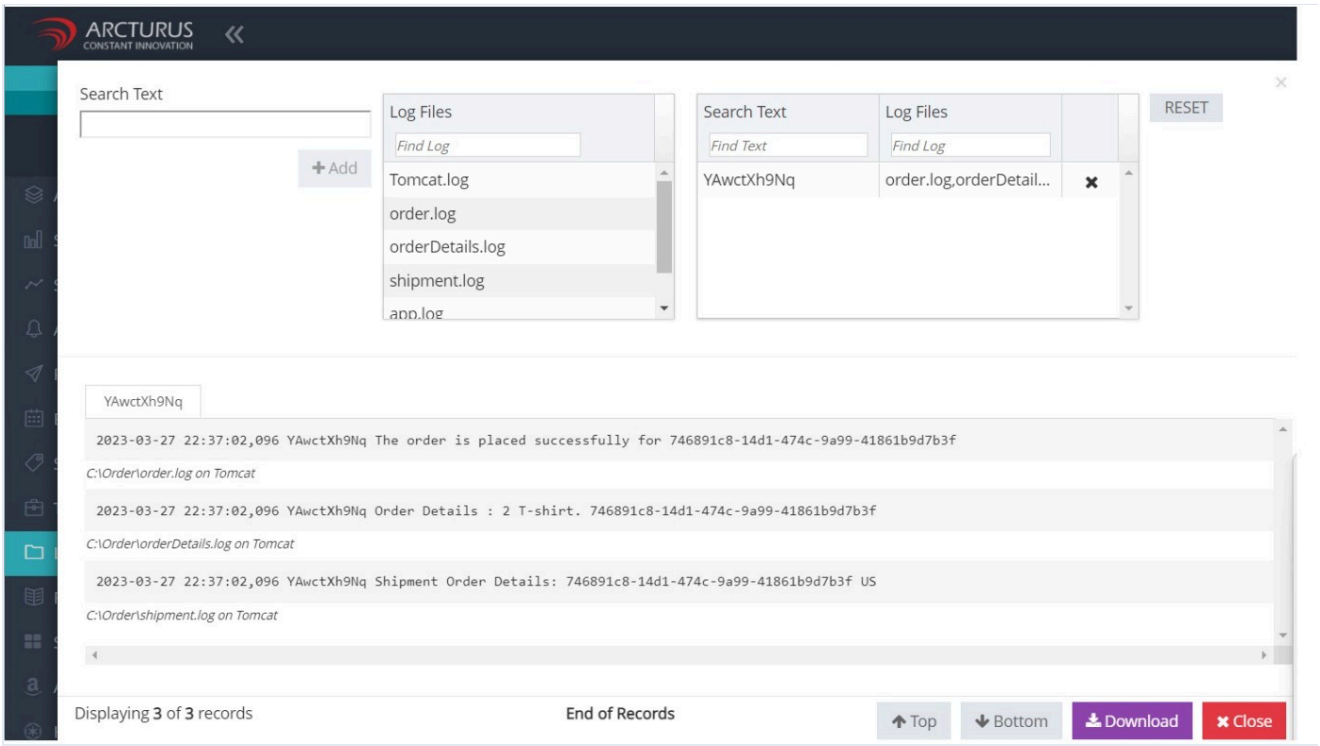
1. Enter the search text.
2. Select the log files.
3. Click the Add button.



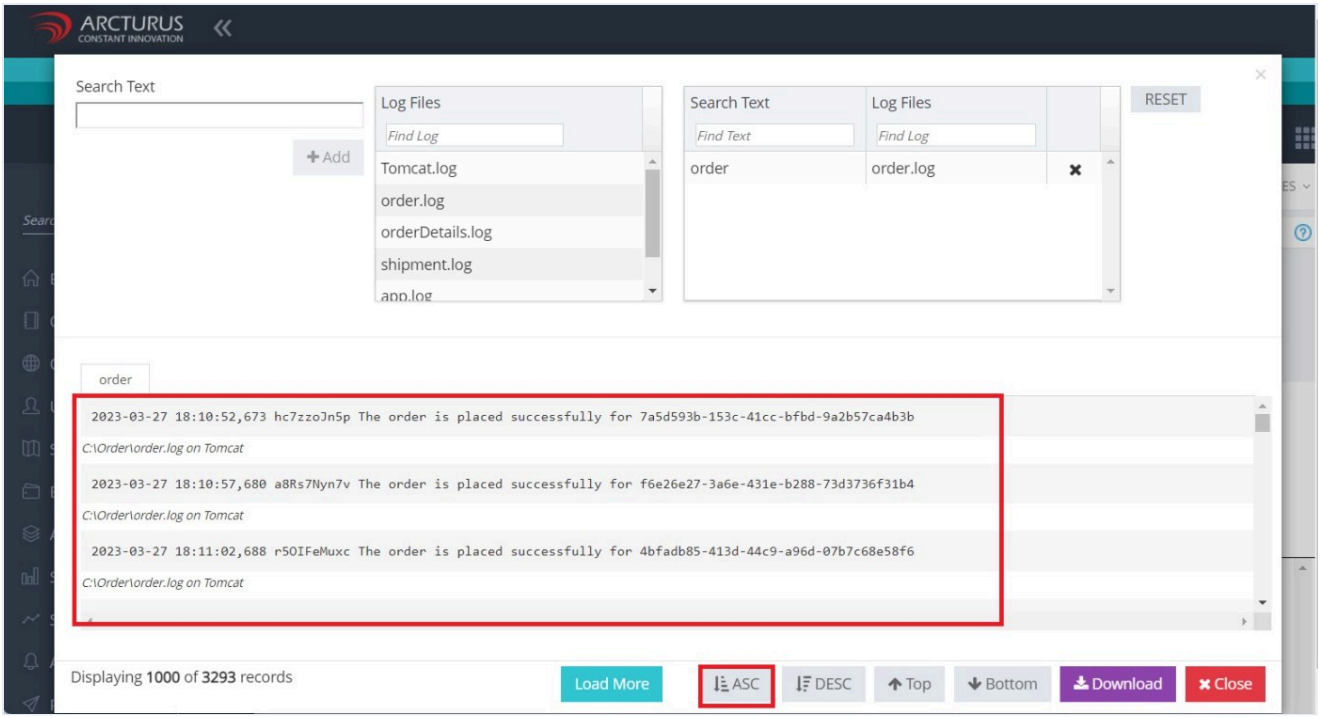
3. A tab will appear with the name of the search text. Click on the tab to view the results.



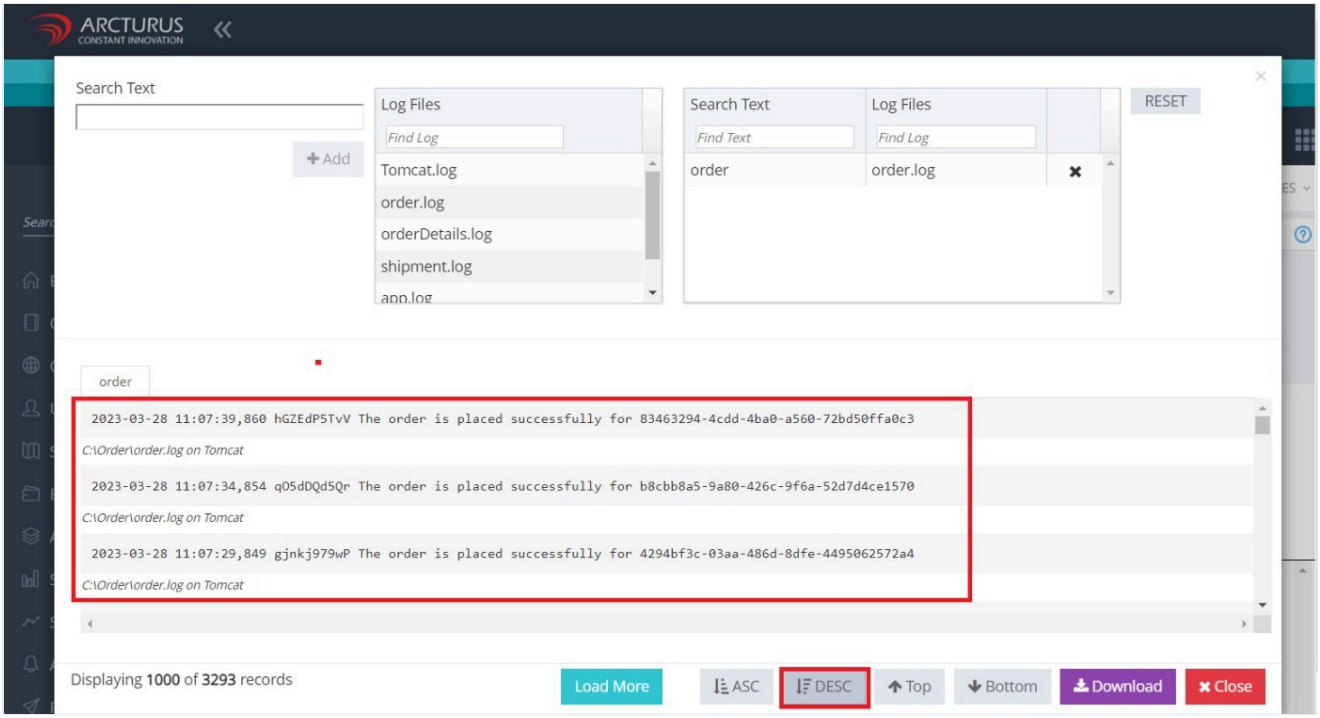
By default, it will display the **top 1000 records** in **descending order** based on the log time.



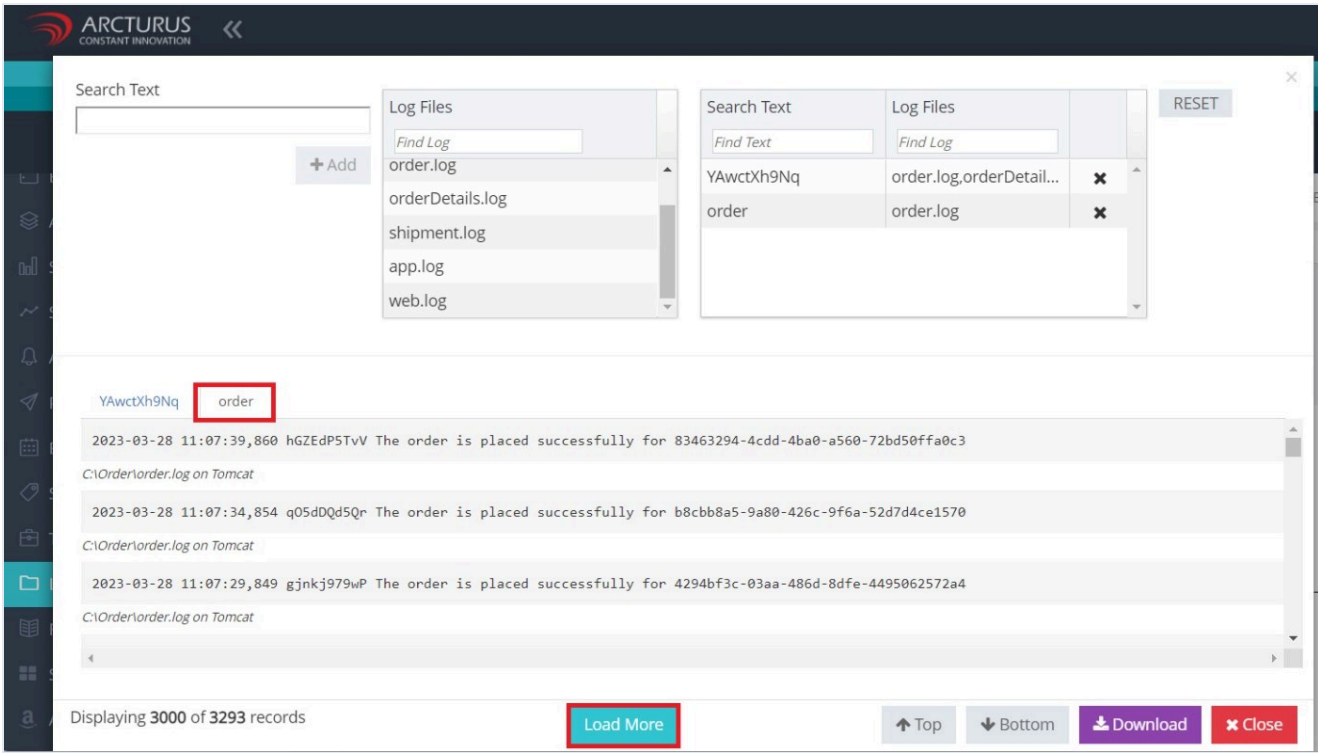
To view the records in the **descending order** based on date-time click the **DESC** button.

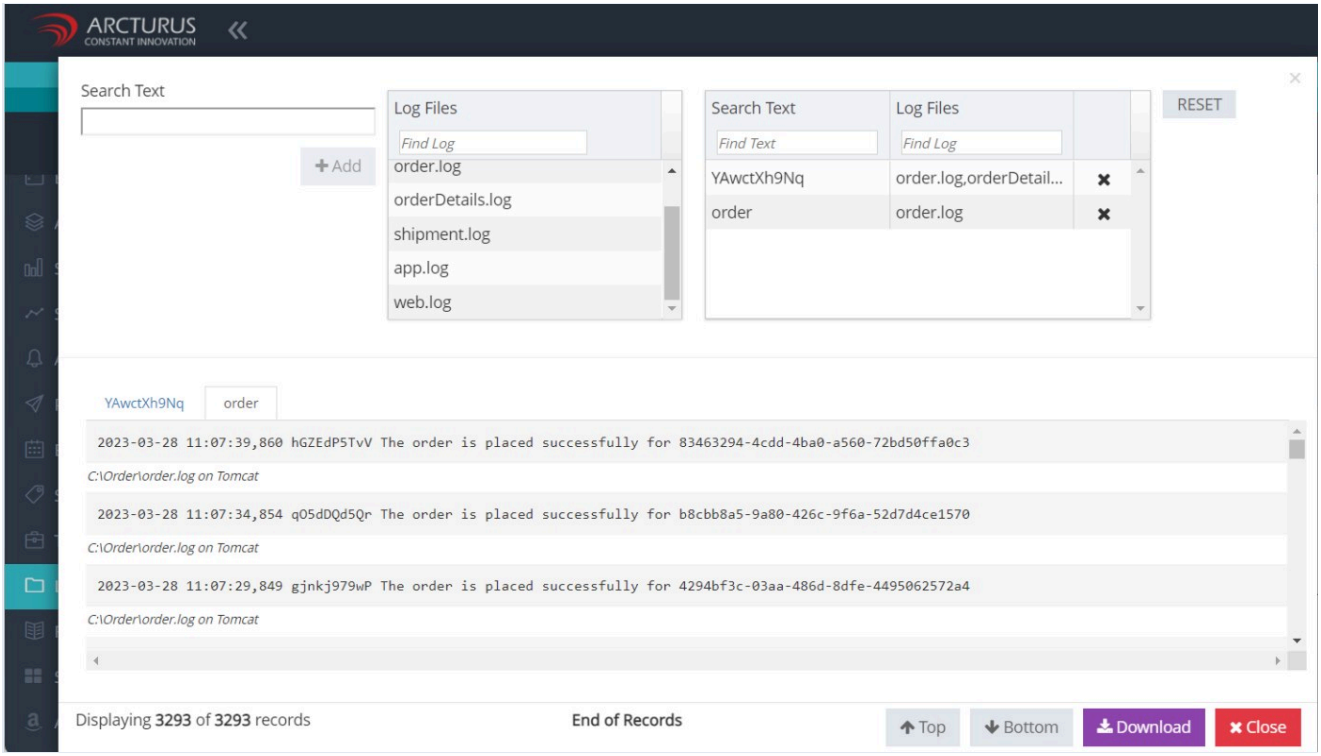


To load the records in the **descending order** based on datetime click the **DESC** button.

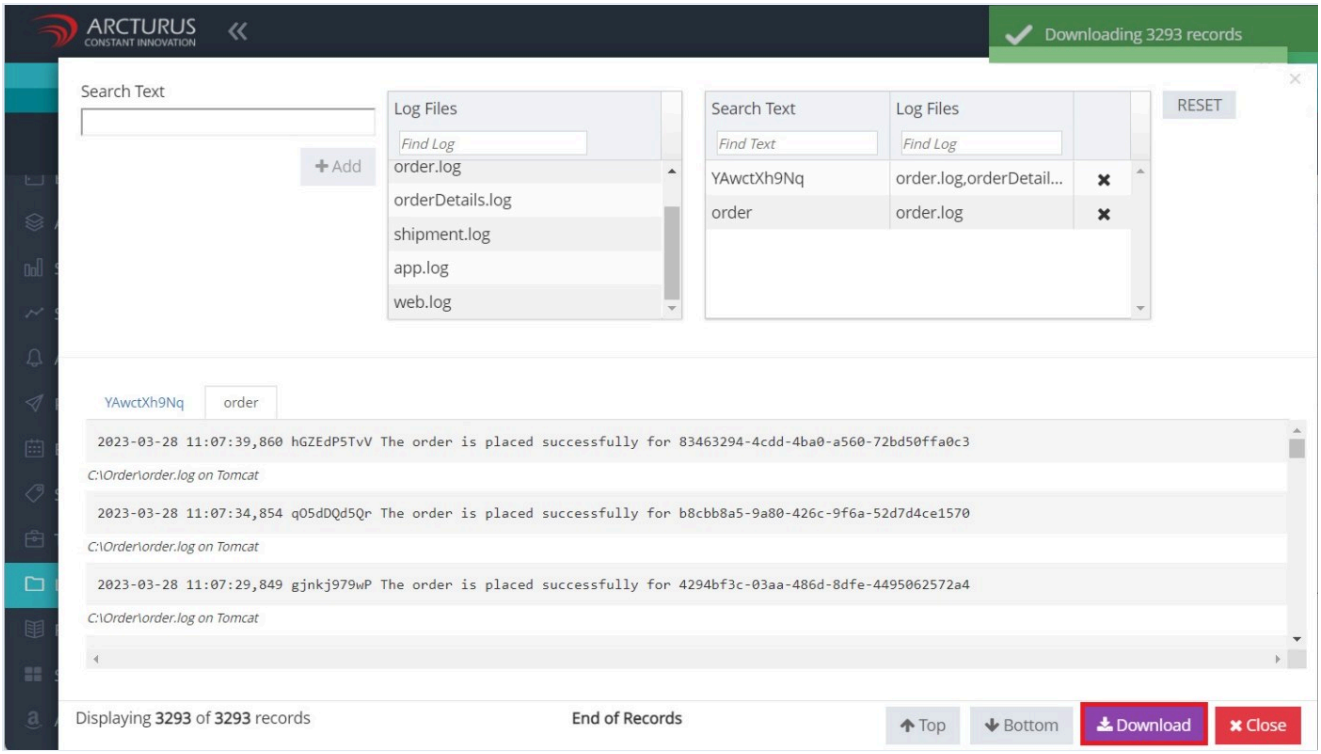


4. If there are more than 1000 records the **Load More** button will be displayed. Click the **Load More** button to load next set of 1000 records.





5. Click the **Download** button to download the records.



Monitoring with Applicare

USING APPLICARE › APPLICATIONS

Date and Regex pattern format

Applicare 9.2 · Updated April 21, 2024 · logs · **Pending technical review**

Sl.No.	Regx	Date Pattern
1	\d{4}-\d{2}-\d{1,2}T\d{1,2}:\d{2}:\d{2}.\d{3}+\d{4}	yyyy-MM-dd'T'HH:mm:ss.SSSZ
2	[A-Za-z]{3,}\s\d{1,2},\s\d{4},\s\d{1,2}:\d{2}:\d{2},\d{1,3}\s[A-Z]{2}\s[A-Z]{3}	MMMMM dd, yyyy, hh:mm:ss,SSS a z
3	[A-Za-z]{3}\s\d{1,2},\s\d{4},\s\d{1,2}:\d{2}:\d{2},\d{1,3}\s[A-Z]{2}\s[A-Z]{3}	MMM dd, yyyy, hh:mm:ss,SSS a z
4	[A-Za-z]{3,}\s[A-Za-z]{3,}\s\d{1,2}\s\d{1,2}:\d{2}:\d{2}\s[A-Za-z]{3,}\s\d{4}	EEE MMM d HH:mm:ss z yyyy
5	[A-Za-z]{3}\s[A-Za-z]{3}\s\d{1,2}\s\d{1,2}:\d{2}:\d{2}\s[A-Za-z]{3,}\s\d{4}	EEE MMM d HH:mm:ss z yyyy
6	\d{1,2}/\d{1,2}/\d{2}\s\d{1,2}:\d{2}:\d{2}:\d{1,3}\s[a-zA-Z]{3,}	MM/dd/yy HH:mm:ss:SSS z
7	\d{4}-\d{1,2}-\d{1,2}\s\d{1,2}:\d{2}:\d{2},\d{1,3}	yyyy-MM-dd HH:mm:ss,SSS
8	\d{4}-\d{1,2}-\d{1,2}\s\d{1,2}:\d{2}:\d{2}.\d{1,3}	yyyy-MM-dd HH:mm:ss.SSS
9	\d{1,2}/\d{1,2}/\d{2}\s\d{1,2}:\d{2}:\d{2}:\d{1,3}\s[A-Z]{3}	MM/dd/yy HH:mm:ss:SSS z
10	\d{1,2}/[A-Za-z]{3,}/\d{4}:\d{1,2}:\d{2}:\d{2}\s-\d{4}	dd/MMMMM/yyyy:HH:mm:ss Z
11	\d{1,2}/[A-Za-z]{3}/\d{4}:\d{1,2}:\d{2}:\d{2}\s-\d{4}	dd/MMM/yyyy:HH:mm:ss Z
12	\d{1,2}/\d{1,2}/\d{2}\s\d{1,2}:\d{2}:\d{2}:\d{1,3}	MM/dd/yy HH:mm:ss:SSS
13	\d{1,2}/\d{1,2}/\d{2}\s\d{1,2}:\d{2}:\d{2}	MM/dd/yy HH:mm:ss
14	\d{1,2}/[A-Za-z]{2,}/\d{4}:\d{1,2}:\d{2}:\d{2}	dd/MMMM/yyyy:HH:mm:ss
15	\d{14}	yyyyMMddHHmmss
16	\d{12}	yyyyMMddHHmm
17	\d{8}	yyyyMMdd
18	\d{8}\s\d{6}	yyyyMMdd HHmmss
19	\d{8}\s\d{4}	yyyyMMdd HHmm
20	\d{1,2}-\d{1,2}-\d{4}\s\d{1,2}:\d{2}:\d{2}	dd-MM-yyyy HH:mm:ss
21	\d{4}-\d{1,2}-\d{1,2}\s\d{1,2}:\d{2}:\d{2}	yyyy-MM-dd HH:mm:ss
22	\d{1,2}/\d{1,2}/\d{4}\s\d{1,2}:\d{2}:\d{2}	MM/dd/yyyy HH:mm:ss
23	\d{1,2}/\d{1,2}/\d{2}\s\d{1,2}:\d{2}:\d{2}	MM/dd/yy HH:mm:ss
24	\d{4}/\d{1,2}/\d{1,2}\s\d{1,2}:\d{2}:\d{2}	yyyy/MM/dd HH:mm:ss

Sl.No.	Regx	Date Pattern
25	\d{1,2}/\d{1,2}/\d{2}\s\d{1,2}:\d{2}:\d{2}	dd/MM/yy HH:mm:ss
26	\d{1,2}\s[a-z]{3}\s\d{4}\s\d{1,2}:\d{2}:\d{2}	dd MMM yyyy HH:mm:ss
27	\d{1,2}\s[a-z]{4,}\s\d{4}\s\d{1,2}:\d{2}:\d{2}	dd MMMM yyyy HH:mm:ss
28	\d{1,2}-\d{1,2}-\d{4}\s\d{1,2}:\d{2}	dd-MM-yyyy HH:mm
29	\d{4}-\d{1,2}-\d{1,2}\s\d{1,2}:\d{2}	yyyy-MM-dd HH:mm
30	\d{1,2}/\d{1,2}/\d{4}\s\d{1,2}:\d{2}	MM/dd/yyyy HH:mm
31	\d{4}/\d{1,2}/\d{1,2}\s\d{1,2}:\d{2}	yyyy/MM/dd HH:mm
32	\d{1,2}\s[a-z]{3}\s\d{4}\s\d{1,2}:\d{2}	dd MMM yyyy HH:mm
33	\d{1,2}\s[a-z]{4,}\s\d{4}\s\d{1,2}:\d{2}	dd MMMM yyyy HH:mm
34	\d{1,2}-\d{1,2}-\d{4}	dd-MM-yyyy
35	\d{4}-\d{1,2}-\d{1,2}	yyyy-MM-dd
36	\d{1,2}/\d{1,2}/\d{4}	MM/dd/yyyy
37	\d{4}/\d{1,2}/\d{1,2}	yyyy/MM/dd
38	\d{1,2}\s[a-z]{3}\s\d{4}	dd MMM yyyy
39	\d{1,2}\s[a-z]{4,}\s\d{4}	dd MMMM yyyy

USING APPLICARE › APPLICATIONS

LAMA

Applicare 9.2 · Updated April 9, 2026 · logs · **Pending technical review**

LAMA is an API based performance data gathering and early warning system that is designed to help brokerage firms identify and address potential problems before they cause an outage. It collects data from the broker's trading systems and uses machine learning to identify patterns that could indicate a problem. If LAMA detects a potential problem, it will generate an alert so that the brokerage firm and exchange can take preventive corrective action. It is a cost-effective way to prevent outages and reduce the risk of financial losses.

NSE India has adopted LAMA and LAMA reporting is a requirement for brokerage firms.

The data sent from Applicare to LAMA represents a consolidated summary of all servers within the application, rather than details for individual servers.

You can access individual server data directly through the Applicare console. Applicare computes the average of this data and posts the aggregated result to LAMA.

It is important to note that data posting to LAMA occurs only while the controller is actively running.

Median formulae for all the metrics:

Median Calculation Formula

Let:

- T be the sorted list of throughput values.
- n be the number of values in T.
- $T[i]$ represents the i^{th} element in the sorted list.

The median T_{med} is calculated as:

1. **If n is odd**:

$$T_{med} = T\left(\frac{n+1}{2} - 1\right)$$

(Select the middle element from the sorted list.)

2. **If n is even**:

$$T_{med} = \frac{T\left(\frac{n}{2} - 1\right) + T\left(\frac{n}{2}\right)}{2}$$

(Take the average of the two middle elements.)

Hardware Metrics

1. CPU (%) - Machine CPU usage. Available in Enterprise -> Summary.

CPU Average = (Sum of CPU usage across all configured applications) / (Total number of configured applications).

```yaml
Example:
Configured Applications : Nopcom, Petstore
Nopcom CPU values in last 5 minutes -> 10,24,52,34,12
Petstore CPU values in last 5 minutes -> 1,5,20,7,15
CPU Average = (SUM of configured applications CPU) / (Total no. of configured applications CPU)
CPU Average = (10 + 24 + 52 + 34 + 12 + 1 + 5 + 20 + 7 + 15)/ (10)
CPU Average = 18
```

2. Memory (%) - Machine memory usage. Available in Enterprise -> Summary.

Memory Average = (Sum of memory usage across all configured applications) / (Total number of configured applications)

```yaml
Example:
Configured Applications : Nopcom, Petstore
Nopcom Memory values in last 5 minutes -> 45,32,52,60,47
Petstore Memory values in last 5 minutes -> 51,53,35,20,55
Memory Average = (SUM of configured applications Memory) / (Total no. of configured applications Memory)
Memory Average = (45 + 32 + 52 + 60 + 47 + 51 + 53 + 35 + 20 + 55)/ (10)
Memory Average = 45
```

3. Disk (%) - Machine disk usage. Available in Server Analayzer -> OS.

4. Uptime (in minutes) - Availability of the server in the machine for 5 minutes interval. Available in the Enterprise -> Summary.

Exchange DOC

Metric Data

Field Description	Key Name	Data Type	Frequency	Expected Value
CPU Utilization (in %)	cpu	Object	Every 5 min	Float (between 0-100, upto 2 decimals)
Memory Utilization (in %)	memory	Object	Every 5 min	Float (between 0-100, upto 2 decimals)
Disk Utilization (in %)	disk	Object	Every 5 min	Float (between 0-100, upto 2 decimals)
Uptime (in minutes)	uptime	Object	Every 5 min	Long value

Application Metrics

1. Request per second - Indicates the server's throughput. This metric can be found in Server Analyzer -> App Server.

2. Response Time (milliseconds) - Average Response Time - Reflects the server's average response time. You can find this metric in Server Analyzer -> App Server.

3. Trading Failure API count - Failure count of the trading API's. Available in Key Business Transactions Load Distribution.

4. Client Authentication Failure Auth - Failure count of the client authentication API's. Available in Key Business Transactions Load Distribution.

5. History request per second - Historical throughput of the server.

Available in Server Analyzer -> App Server.

6. Historical Response Time (milliseconds) - Represents the average response time of the server.

Available in Server Analyzer -> App Server.

1. Logic to implement the Historical Throughput (HST_THRGHPT) and Historical Latency (HST_LTNCY)

a. Everyday at 11:30 PM the last 24 hours throughput and latency (min, max, med and avg) values will be calculated and save it in a table. So this table will have every day details.

b. On every start of the day the last 21 days average data for the (min, max, med and avg) values are calculated and send the details to LAMA. By Default all the Saturday's and Sunday's data will be ignored in this 21 days data.

Example:

Day	Historical Throughput				Historical Latency				LAMA Data (21 days data)
	Min	Max	Med	Avg	Min	Max	Med	Avg	
1	100	500	125	326	1000	5000	1245	3567	Average of Day1
2	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 2
3	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 3
4	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 4
5	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 5
6	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 6
7	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 7
8	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 8
9	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 9
10	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 10
11	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 11
12	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 12
13	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 13
14	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 14
15	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 15
16	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 16
17	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 17
18	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 18
19	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 19
20	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 20

21	100	500	125	326	1000	5000	1245	3567	Average of Day 1 to 21
22	100	500	125	326	1000	5000	1245	3567	Average of Day 2 to 22
23	100	500	125	326	1000	5000	1245	3567	Average of Day 3 to 23
24	100	500	125	326	1000	5000	1245	3567	Average of Day 4 to 24
25	100	500	125	326	1000	5000	1245	3567	Average of Day 5 to 25
26	100	500	125	326	1000	5000	1245	3567	Average of Day 6 to 26
27	100	500	125	326	1000	5000	1245	3567	Average of Day 7 to 27
28	100	500	125	326	1000	5000	1245	3567	Average of Day 8 to 28
29	100	500	125	326	1000	5000	1245	3567	Average of Day 9 to 29
30	100	500	125	326	1000	5000	1245	3567	Average of Day 10 to 30
31	100	500	125	326	1000	5000	1245	3567	Average of Day 11 to 31
32	100	500	125	326	1000	5000	1245	3567	Average of Day 12 to 32
33	100	500	125	326	1000	5000	1245	3567	Average of Day 13 to 33

7. Log Monitoring - Error codes configured in the LAMA of the configured Applications. Available in Errors screen.

Exchange DOC

Metric Data

Field Description	Key Name	Data Type	Frequency	Expected Value
Log Monitoring (log messages for log levels Error, Fatal and Warning)	Log	String (2000)	Every 5 min	Timestamp, Server Name, Source Name, Log ID, Log Level, Description
Request/Second - throughput of E2E system	throughput	Object	Every 5 min	Min, Max, Avg & Median values
Response Time (in microseconds) – round trip time of E2E system	latency	Object	Every 5 min	Min, Max, Avg & Median values
Trading API Failure Count	failureTradeApi	Numeric	Every 5 min	Long value
Client Authentication Failure Count	failureAuthentication	Numeric	Every 5 min	Long value
Historical Requests/Second	historicalThroughput	Object	Start of Day	Min, Max, Avg & Median values
Historical Response Time (in microseconds)	historicalLatency	Object	Start of Day	Min, Max, Avg & Median values

Network Metrics

1. Bandwidth (%) - It is the average of both the input and output bandwidth utilization of the interface in the router.

```yaml
Configured Mbps = 1000 // 1Gbps  Default value

Input rate in bits / sec – (IfInOctects * 0.8) / 60;
Output rate in bits / sec – (IfOutOctects * 0.8) / 60;

Input Utilization:
Input Utilization (%) = ((Input Rates in bits/sec) / (Configured Mbps * 1000000)) * 100;

Output Utilization:
Output Utilization (%) = ((Output Rates in bits/sec) / (Configured Mbps * 1000000)) * 100;

Bandwidth Utilization = (Input Utilization (%) + Output Utilization (%)) / 2;

Example:
Configured Mbps – 300Mbps
Input Utilization (%) = (47575000 / (300 * 1000000)) * 100;
Input Utilization (%) = 15.86

Output Utilization (%) = (157305000 / (300 * 1000000)) * 100;
Output Utilization (%) = 52.44

Bandwidth Utilization = (15.86 + 52.44) / 2
Bandwidth Utilization (%) = 34.15
```

Important:

The above calculation is the average for a minute. Likewise for LAMA it will be the average for 5 minutes.

2. Packet error count - Sum of current Input and output error count of the interface in the router via SNMP.

ifinErrors

ifOutErrors

```text
Packet error count = (ifinErrors+ ifOutErrors)
```

Important:

The above calculation is the average for a minute. Likewise for LAMA it will be the average for 5 minutes.

Exchange DOC

Metric Data

Field Description	Key Name	Frequency	Data Type	Expected Value
Bandwidth Utilization (in %)	bandwidth	Every 5 min	Object	Float (between 0-100, upto 2 decimals)
Network Latency (in microseconds)	latency	Every 5 min	Object	Long value
Packet Error Count	packetCount	Every 5 min	Numeric	Long value
DNS Lookup Failure Count	lookupCount	Every 5 min	Numeric	Long value

Important:

The Exchanges have decided to deprecate the below-listed parameters. **Network Latency DNS Lookup Count**

Database Metrics

ORACLE DB

1. Replication status - Refers to the availability of the replication process, not the availability of the database itself.

```
TEXT
select process from v$managed_standby where process='<PROCESS_NAME>' fetch first 1 row only
```

Example:

PROCESS_NAME - **MRP0**

Here **MRP0** is the process which is doing the replication. If the process is active, the replication status will be "Up." If the process is not running, the status will be "Down."

2. Replication Queue size - Pending log sync count in the replication.

```
TEXT
SELECT MAX(A.SEQUENCE#) FROM V$ARCHIVED_LOG A,v$archive_dest_status b where (a.dest_id='1' or a.applied ='YES') and a.dest_id=b.dest_id group by b.destination,b.ERROR, a.dest_id, thread# order by a.thread#
```

Here it will give the list of SEQUENCE#. The SEQUENCE# will be sorted in the descending order and the difference of the first two values will be calculated as a queue size.

Example:

```
TEXT
Intital Result
86773
86632
86775

Result after sorting
86775
86773
86632

Queue Size = Value1 – Value2
Queue Size = 86775 – 86773
Queue Size = 2
```

3. Replication Bandwidth (%) - Bandwidth Utilization of the Database.

```
TEXT
select AVG(round((blocks*block_size/1024/1024)/((next_time-first_time)*86400),2)) as
bytesinMbps from v$archived_log where ((next_time-first_time)*86400<>0) and first_time
between to_date('FROM_DATE','YYYY/MM/DD HH24:MI:SS') and to_date('TO_DATE','YYYY/MM/DD
HH24:MI:SS') and dest_id=1
```

FROM_DATE and **TO_DATE** for every 1 minute interval.

Example:

FROM_DATE - 2024/05/02 16:13:00

TO_DATE - 2024/05/02 16:14:00

Important:

The calculation provided reflects a one-minute average. For LAMA, the average is computed over a five-minute period.

4. Replication Latency (milliseconds) - Time delay occurred on the data transfer.

```
TEXT
SELECT VALUE FROM V$DATAGUARD_STATS WHERE UPPER(NAME)='APPLY LAG'
```

MySQL DB

1. Replication status - Refers to the availability of the replication process, not the availability of the database itself.

For MySQL < 8.0.22

```
TEXT
SHOW SLAVE STATUS;
```

For MySQL ≥ 8.0.22

```
TEXT
SHOW REPLICA STATUS;
```

Logic

MySQL < 8.0.22

- **DB Status is UP** when:
  - Slave_IO_Running = `Yes`
  - Slave_SQL_Running = `Yes`
- **DB Status is DOWN** when:
  - Either of the above is not `Yes`

MySQL ≥ 8.0.22

- **DB Status is UP** when:
  - Replica_IO_Running = `Yes`
  - Replica_SQL_Running = `Yes`
- **DB Status is DOWN** when:
  - Either of the above is not `Yes`

2. Replication Queue size - Pending log sync count in the replication.

Logic

```
TEXT
MySQL < 8.0.22
Queue Size = Read_Master_Log_Pos – Exec_Master_Log_Pos

MySQL ≥ 8.0.22
Queue Size = Read_Source_Log_Pos – Exec_Source_Log_Pos
```

3. Replication Bandwidth (%) - Bandwidth Utilization of the Database.

```
TEXT
SHOW GLOBAL STATUS WHERE Variable_name IN ('Bytes_received', 'Bytes_sent');
```

Logic

Step 1: Delta Calculation

```
TEXT
Delta Bytes = Current Value – Previous Value
```

Step 2: Convert to Bits/sec

```
TEXT
Bandwidth (bps) = (Delta Bytes / Time Interval) * 8
```

Step 3: Utilization % for 1 Gbps

```
TEXT
Utilization (%) = (Bandwidth / 1000000000) * 100
```

4. Replication Latency (microseconds) - Time delay occurred on the data transfer.

Logic

```
TEXT
MySQL < 8.0.22
Latency = Seconds_Behind_Master * 1000000

MySQL ≥ 8.0.22
Latency = Seconds_Behind_Source * 1000000
```

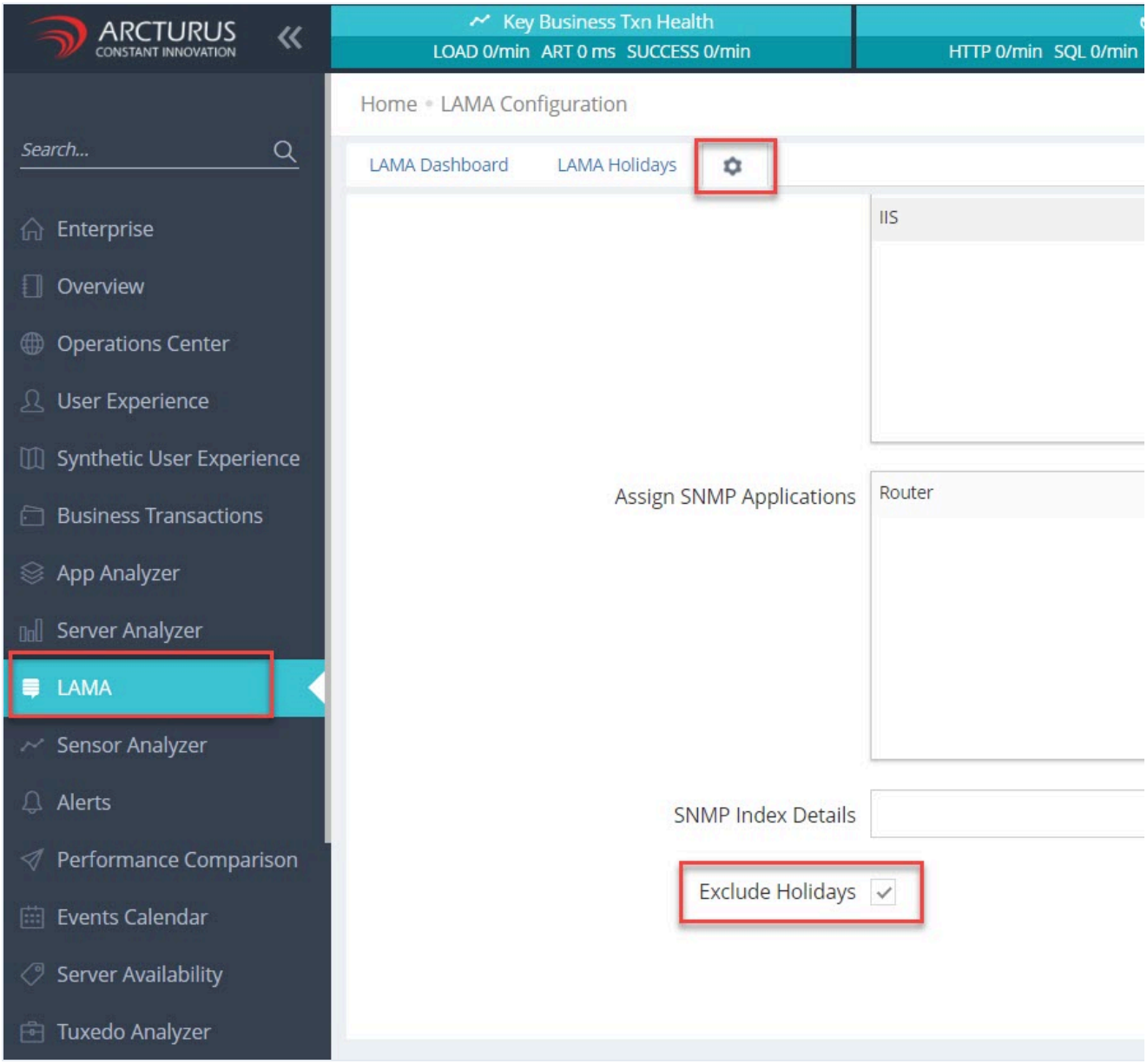
Exchange DOC

Metric Data

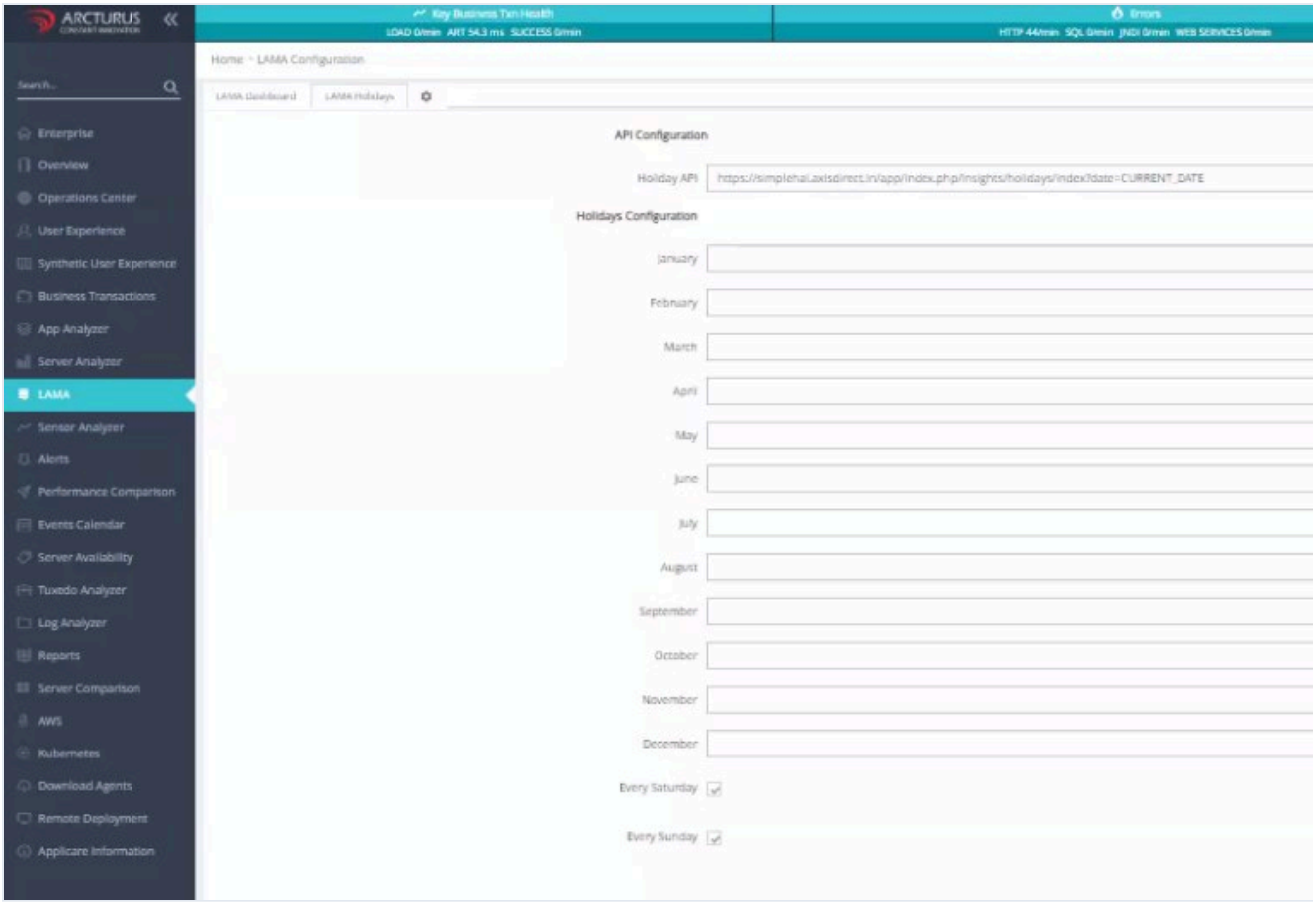
Field Description	Key Name	Frequency	Data Type	Expected Value
Replication Status (1=Up, 0= Down)	status	Every 5 min	Short	1
Replication Queue Size	qSize	Every 5 min	Object	Long value
Replication Bandwidth Utilization (in %)	bandwidth	Every 5 min	Object	Float (between 0-100, upto 2 decimals)
Replication Latency (in microseconds)	latency	Every 5 min	Object	Long value

LAMA Holiday Configuration

If the Exclude Holidays check box is checked in the **LAMA configuration** then both the data collection and data post will be skipped for LAMA based on the LAMA Holiday Configuration.



LAMA Holidays



Holiday API

The holiday will be checked in two conditions.

1. Holiday API is added in the LAMA Holidays screen

Every day after 12:00 AM the api will be triggered once and check whether that day is holiday or not. If yes then both the data collection and data post will be skipped on that day. Otherwise the data collection and data post will happen on that day.

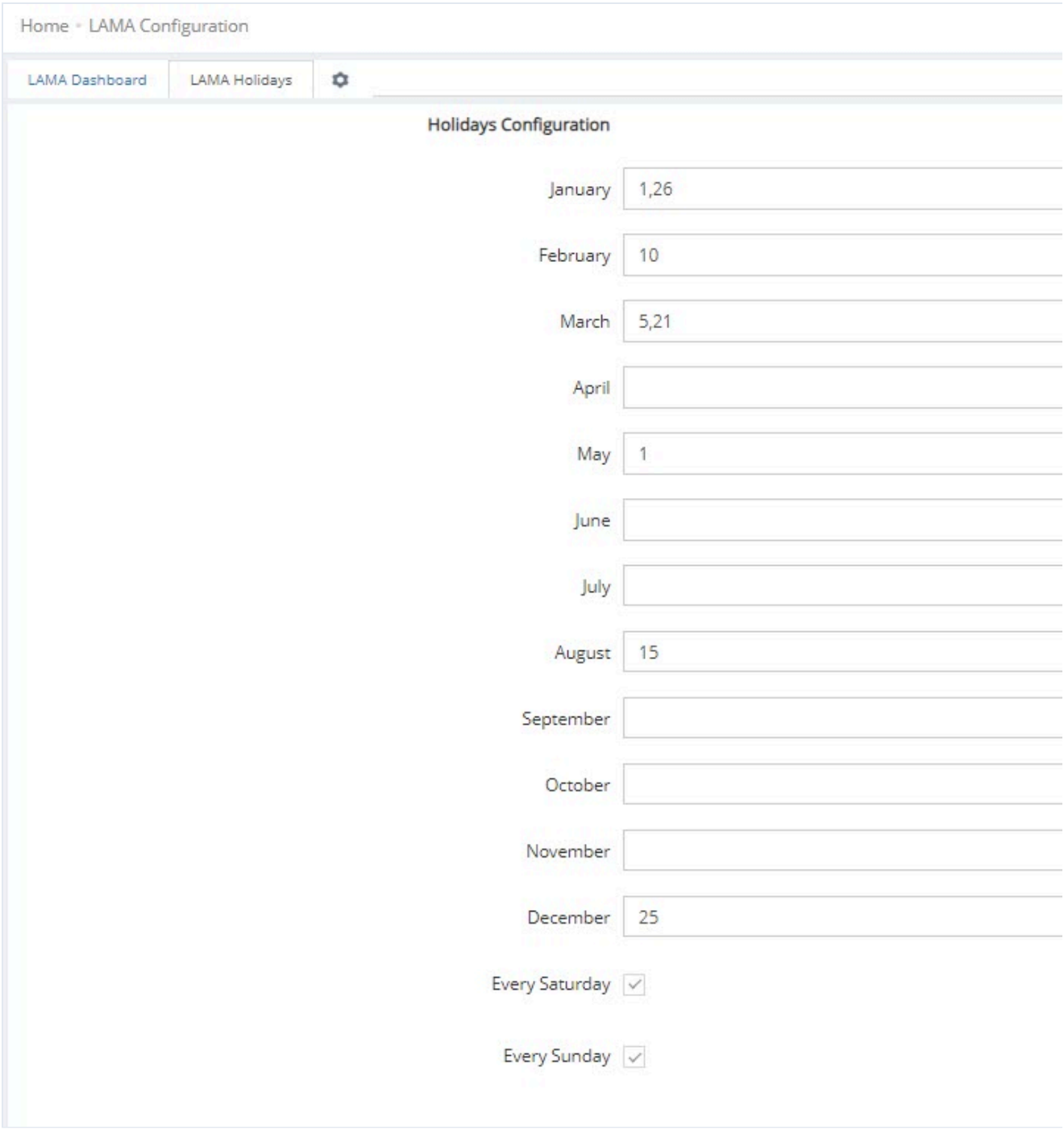


Note:

It is not mandatory to update the holidays in the LAMA Holidays screen. Because controller will automatically update the holiday details from that API. The holiday details will be updated only for that current day.

2. Holiday API is not added in the LAMA Holidays screen

The holiday condition will be fully evaluated based on the days configured in the LAMA Holidays screen.

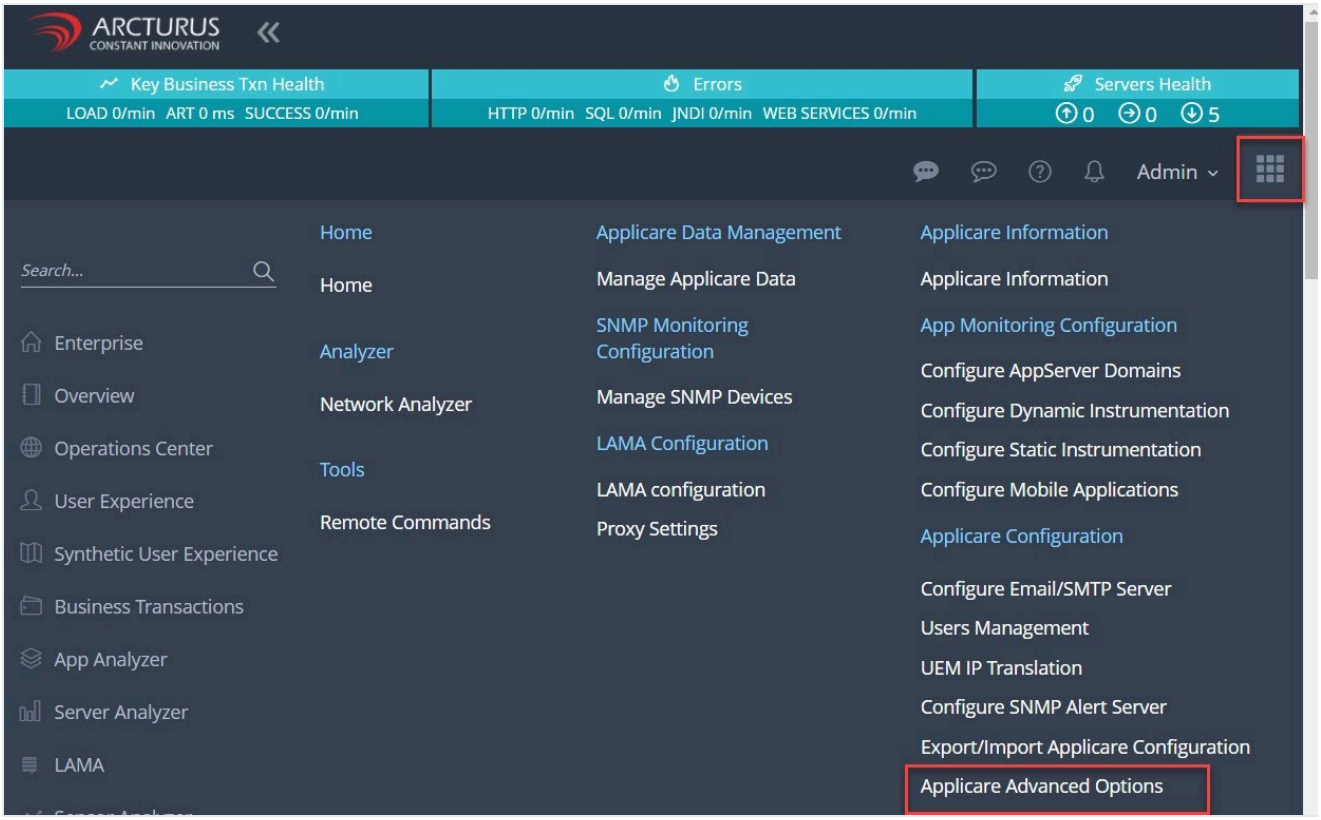


LAMA Logs

By default only **error logs** and the **API check log** for LAMA will be printed in the **ApplicareAdmin.log** file. The log file will be flushed once it reaches 10 MB.

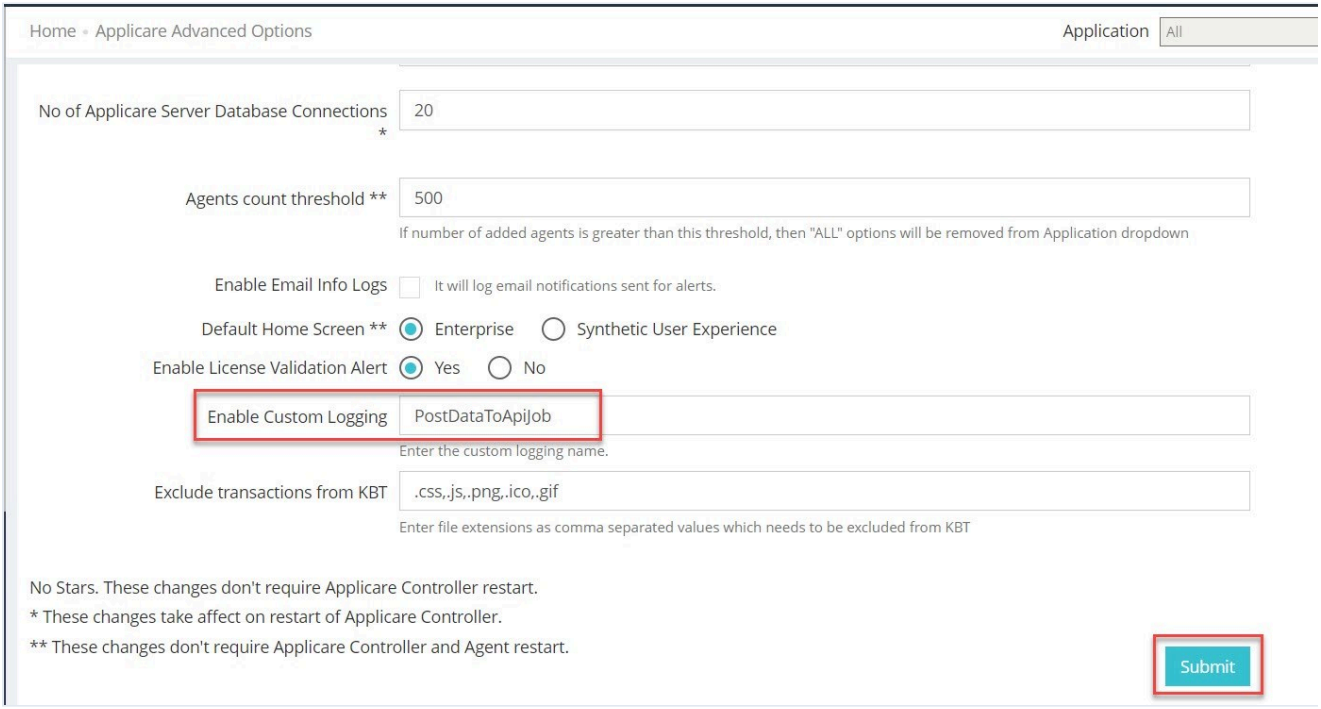
Steps to enable the success log for LAMA.

1. Go to **Applicare Advanced Options** menu.



2. Enter the value **PostDataToApiJob** in the **Enable Custom Logging** field.

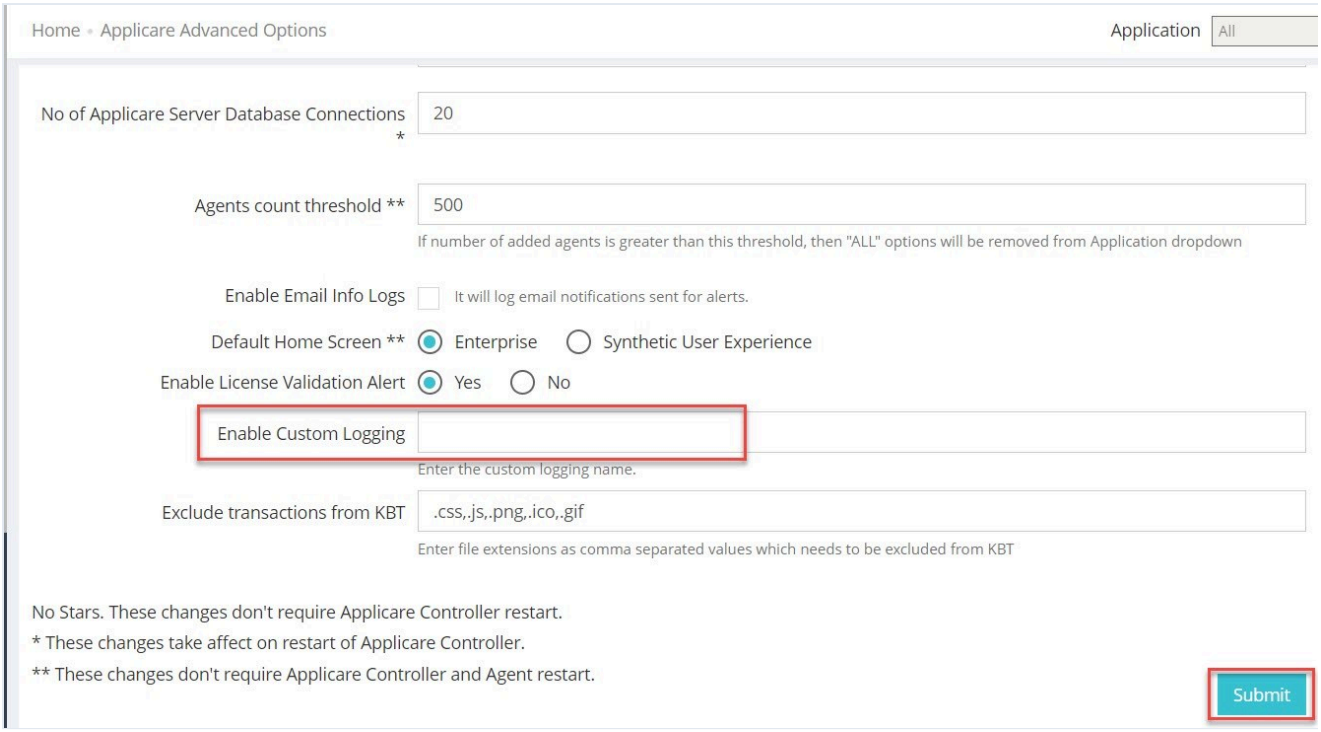
3. Click the **Submit** button to apply and save your changes.



Steps to disable the success log for LAMA.

1. Remove the **PostDataToApiJob** value from the **Enable Custom Logging** field.

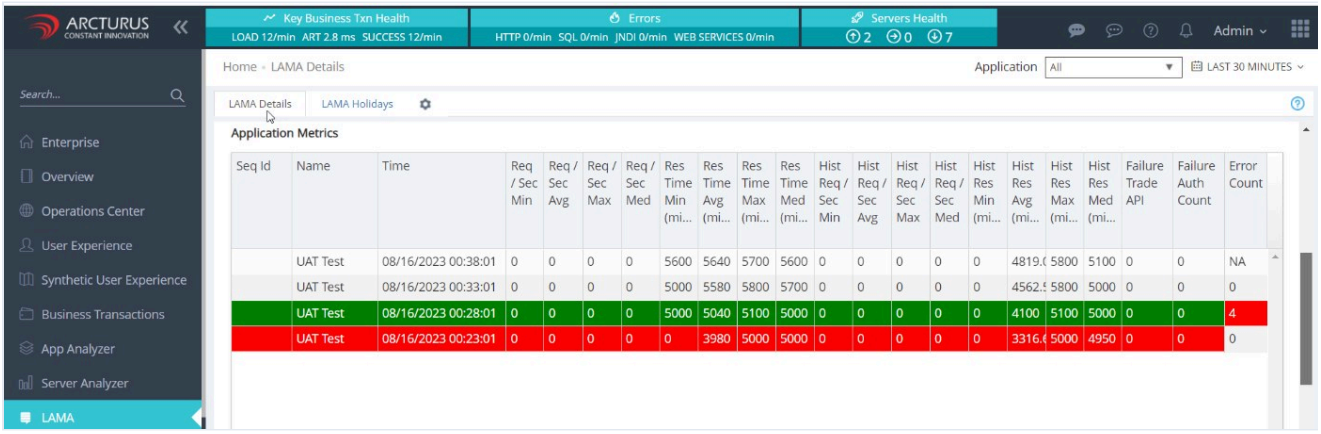
2. Click the **Submit** button to save the changes.



Important:

Controller restart is not required.

LAMA Details



Seq Id	Name	Time	Req / Sec Min	Req / Sec Avg	Req / Sec Max	Req / Sec Med	Res Time Min (mi...	Res Time Avg (mi...	Res Time Max (mi...	Res Time Med (mi...	Hist Req / Sec Min	Hist Req / Sec Avg	Hist Req / Sec Max	Hist Req / Sec Med	Hist Res Min (mi...	Hist Res Avg (mi...	Hist Res Max (mi...	Hist Res Med (mi...	Failure Trade API	Failure Auth Count	Error Count
	UAT Test	08/16/2023 00:38:01	0	0	0	0	5600	5640	5700	5600	0	0	0	0	0	4819.(	5800	5100	0	0	NA
	UAT Test	08/16/2023 00:33:01	0	0	0	0	5000	5580	5800	5700	0	0	0	0	0	4562.5	5800	5000	0	0	0
	UAT Test	08/16/2023 00:28:01	0	0	0	0	5000	5040	5100	5000	0	0	0	0	0	4100	5100	5000	0	0	4
	UAT Test	08/16/2023 00:23:01	0	0	0	0	0	3980	5000	5000	0	0	0	0	0	3316.(	5000	4950	0	0	0

Red color row - Data posting does not occur at regular intervals.

Green color row - Data posting begins occurring at regular intervals.

Error Count Column

1. **Red color** - If there was any errors posted to LAMA this column will be displayed in red color with the total error count. **Click that column to view the error details posted to LAMA**.
2. **No color** with **0** value - No errors have been posted to LAMA.
3. **No color** with **NA** value - No errors have been posted to LAMA.

Updated Metrics Details from the Exchange.

	A	B	C	D	E
1	Metric Type	Metric Key	Value Type	Metric Key Description	Expected Value
2	Application Sequence Mertric(ASM)	throughput(THRGHPT)	Request/Second - throughput of E2E system	It defines the number of requests passing through the application/system per second	Count
3		latency(LTNCY)	Response Time (in Milliseconds) – round trip time of E2E system	It defines the time taken for user to submit the request and receive the response of the request (New-Ack Latency)	Milliseconds
4		failureTradeApi(FLD_TRDAPI)	Trading API Failure Count	It refers to the number of failures while executing Trading API.	Count
5		failureAuthentication(FLD_ATH)	Client Authentication Failure Count	It refers to the number of authentication failures.	Count
6		historicalThroughput(HST_THRGHPT)	Historical Requests/Second	21 days summary of Throughput	Count
7		historicalLatency(HST_LTNCY)	Historical Response Time (in Milliseconds)	21 days summary of Response Time	Milliseconds
8	Hardware Sequence Metric(HSM)	cpu(CPU)	CPU Utilization (in %)	the percentage of available time spent is executing the tasks.	Percentage
9		memory(MEM)	Memory Utilization (in %)	the percent of available memory in use at a given moment.	Percentage
10		disk(DSK)	Disk Utilization (in %)	the percentage of available disk space in use at a given moment.	Percentage
11		uptime(UPTM)	Uptime (in minutes)	the time period for which the system has been active and running	Minutes
12	Database Sequence Metric(DSM)	status(status) (Threshold Not required, we only persist data)	Replication Status (1=Up, 0= Down)	Replication status determines the current state of an object being replicated. The replication status of a source object will return either 1=Up or 0=Down	Integer
13		qSize(QSIZE)	Replication Queue Size	It is the number of units/requests/packets waiting in a queue or present in a system.	Count
14		bandwidth(BNDWTH)	Replication Bandwidth Utilization (in %)	It refers to the amount of information that can be sent over a network connection at one time.	Percentage
15		latency(LTNCY)	Replication Latency (in Milliseconds)	Replication latency is the amount of time it takes for a transaction that occurs in the primary database to be applied to the replicate database.	Milliseconds
16	Network Sequence Metric(NSM)	bandwidth(BNDWTH)	Bandwidth Utilization (in %)	Bandwidth utilization refers to a computer network's capacity to transfer data between devices or over the internet in a specific amount of time.	Percentage
17		packetCount(PCKT_CT)	Packet Error Count	It defines the number of packets received with errors.	Count

Monitoring with Applicare

USING APPLICARE › TRANSACTIONS

Business Transactions User Guide

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Overview

Key Business Transactions enables you to capture performance statistics and set SLAs for Web or Java method calls. It also allows you to track individual transactions with detailed performance metrics and call graphs.

Web key business transactions are defined by specifying one or more URI patterns. Any request matching these defined patterns will be intercepted and statistic will be gathered under the given business transaction name.

Java key business transactions operate similarly, but instead of URIs, you define class and method name patterns.

The Profiling Agent must be enabled for business transactions to function on a server. For web based business transactions to work you need to deploy both HTTP Profiling and Business Transactions aspects. You can enable or deploy these on the Configuration -> Configure Dynamic Instrumentation page.

Configuring Business Transactions

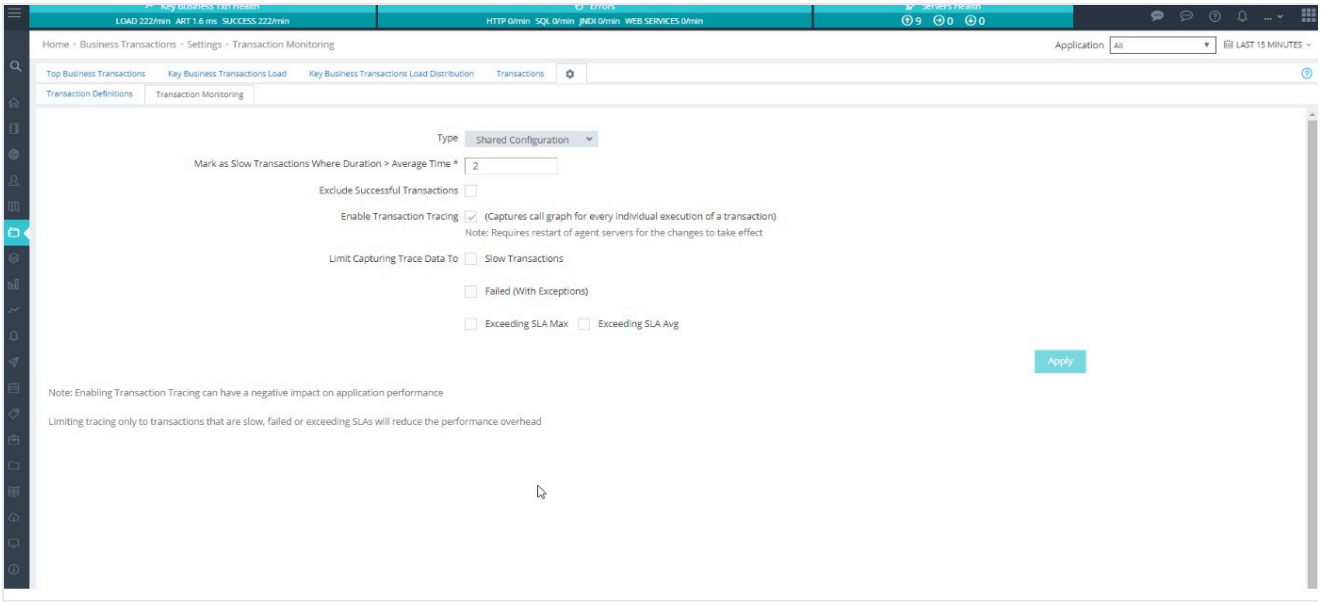
Prerequisites

Before Applicare 7.1.4 you must enable Standard Mode Instrumentation and activate the instrumentation agent on the servers where you want to track business transactions. See Configuring Dynamic Instrumentation on how to enable Standard Mode instrumentation on servers.

For Applicare 7.1.4 onwards At a minimum you must enable HTTP, SQL, JDBC Connection and Business Transactions profiling by deploying those aspects in Configuring Dynamic Instrumentation In Applicare 7.1.4 onwards, 'Enable Per-Transaction Monitoring' in Business Transactions configuration has no impact and BTM wouldn't start until 'Business Transactions' aspect is deployed along with other prerequisites in 'Configure Dynamic Instrumentation'.

Configuration

You can access the business transaction configuration page by navigating to Business Tansactions -> Configure (wheel button on right) Transactions Monitoring. On Transaction Monitoring tab you can configure the amount of details to be captured on configured transactions.



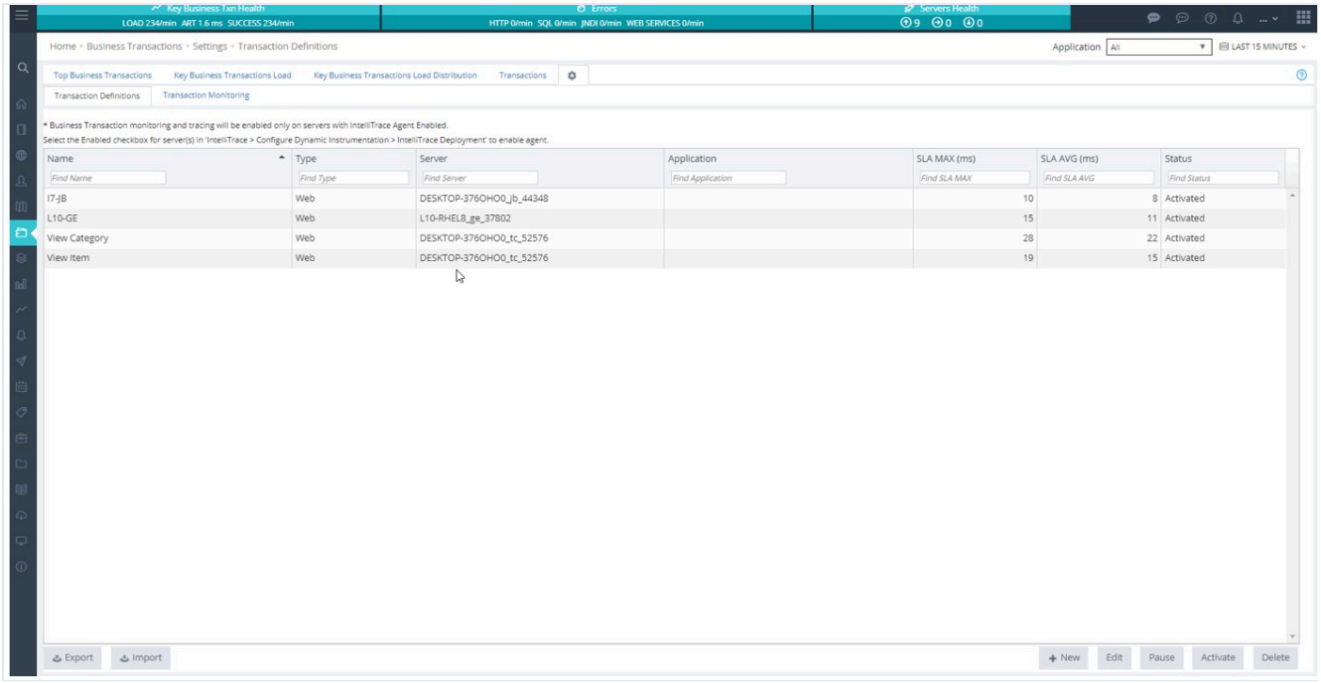
Exclude Successful Transactions allows you to exclude successful/good transactions from Busniss Transaction Monitoring. This helps concentrate on bad transactions which needs a fix or improvement.

Enabling Per-Transaction monitoring gives you the benefit of monitoring your business transactions performance over time as a whole and on a per server basis. Performance overhead of gathering this level of data is minimal and can be enabled even on production systems.

Enabling Transaction Tracing will capture execution graph of each transaction including Java method calls, SQLs and their parameters. Performance overhead of gathering this level of data might be noticable specially in production systems with very high number of business transaction executions per minute. You can reduce the overhead by limiting call graph tracing to transaction executions that are slow, failed or exceeding SLA limits.

Business Transaction Definition

To view the Business Transaction Definition page, navigate to: **Applicare Home -> Business Transactions -> Configure Business Transactions -> Transaction Definitions**.

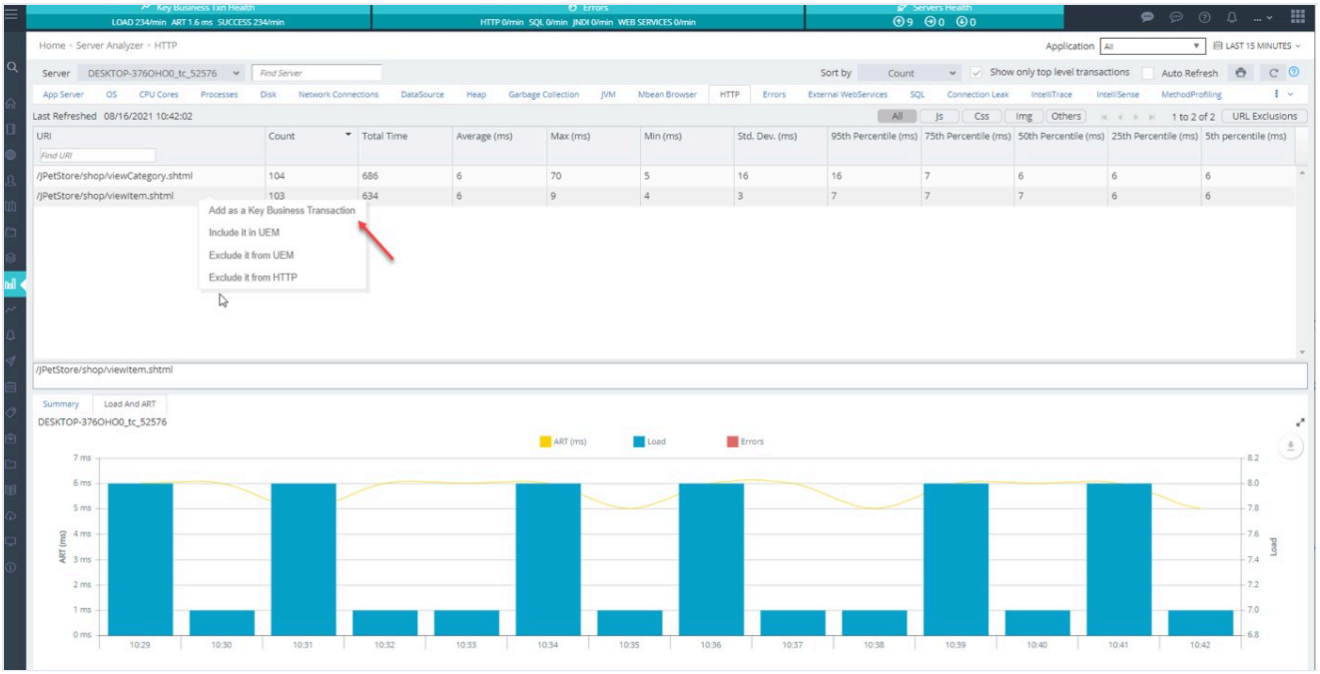


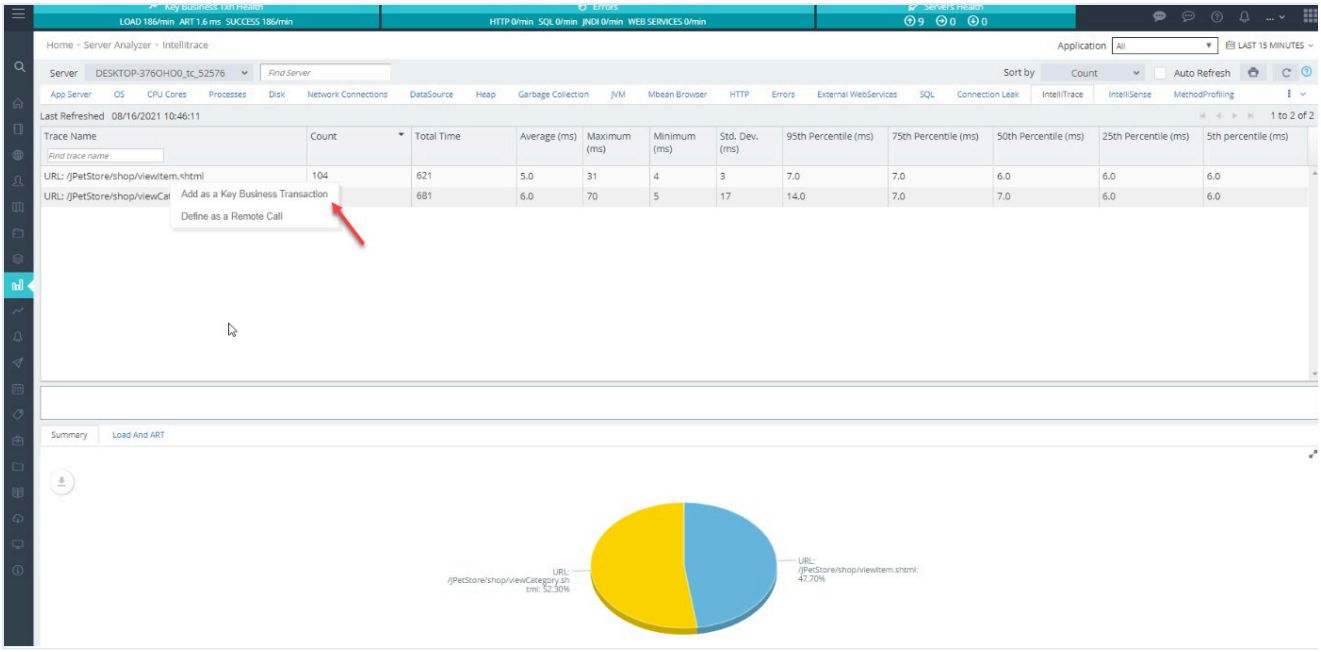
This page shows all defined Key Business Transactions and lets you create, edit, or delete them. Changes made here take effect immediately across the domain without requiring a server restart.

You can export definitions to an XML file or import them into another Applicare environment using the Export/Import buttons.

Defining Business Transactions

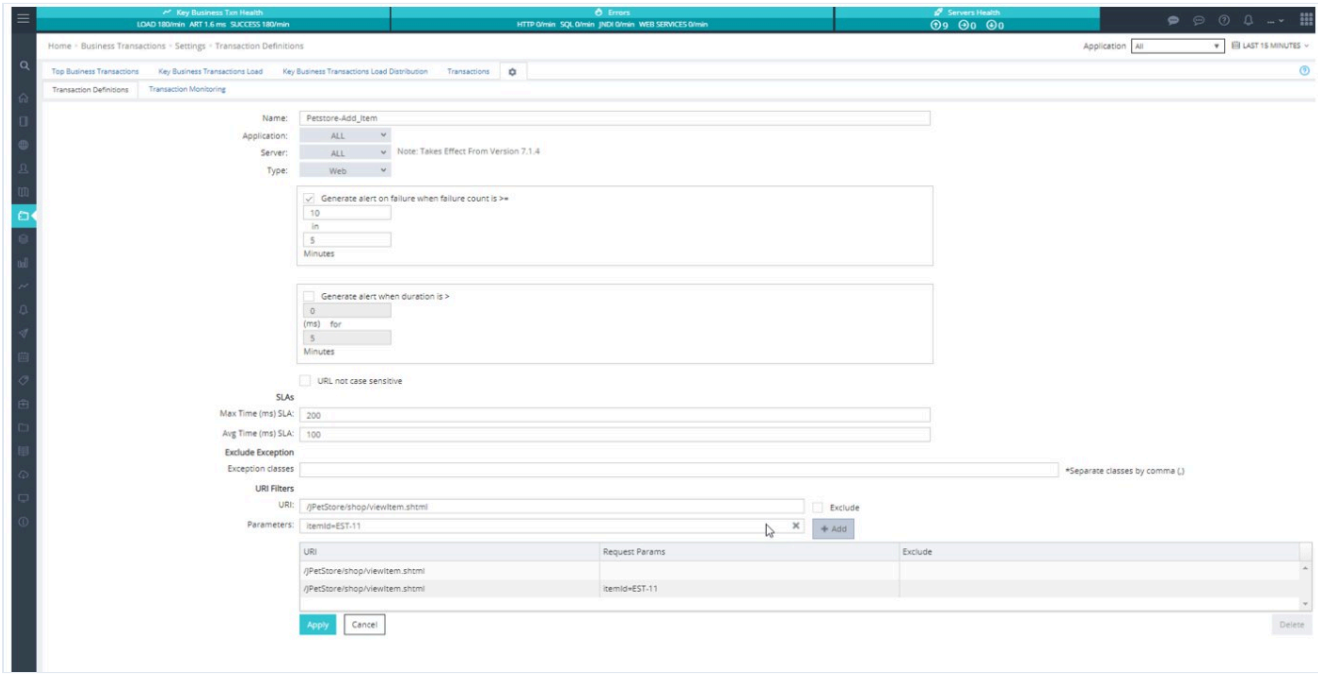
Business transactions can be defined either manually from this page or within the IntelliTrace Analyzer. If IntelliTrace Standard Mode or Advanced Mode with IntelliTrace Profiling is enabled, you can navigate to the HTTP or IntelliTrace tabs in IntelliTrace Analyzer. Right-click on any autodetected URL or Java method to define a Business Transaction for that URL or method.





Add a New Web Business Transaction

To manually add a new Web Business Transaction, click the **New** button on the Transaction Definition page.



1. Provide a **Name** for the Business Transaction

2. Select **Web** from the **Type** drop down box

3. Provide average and maximum SLA values. Leave them blank or set to zero if you don't want to enforce SLA values for this transaction definition

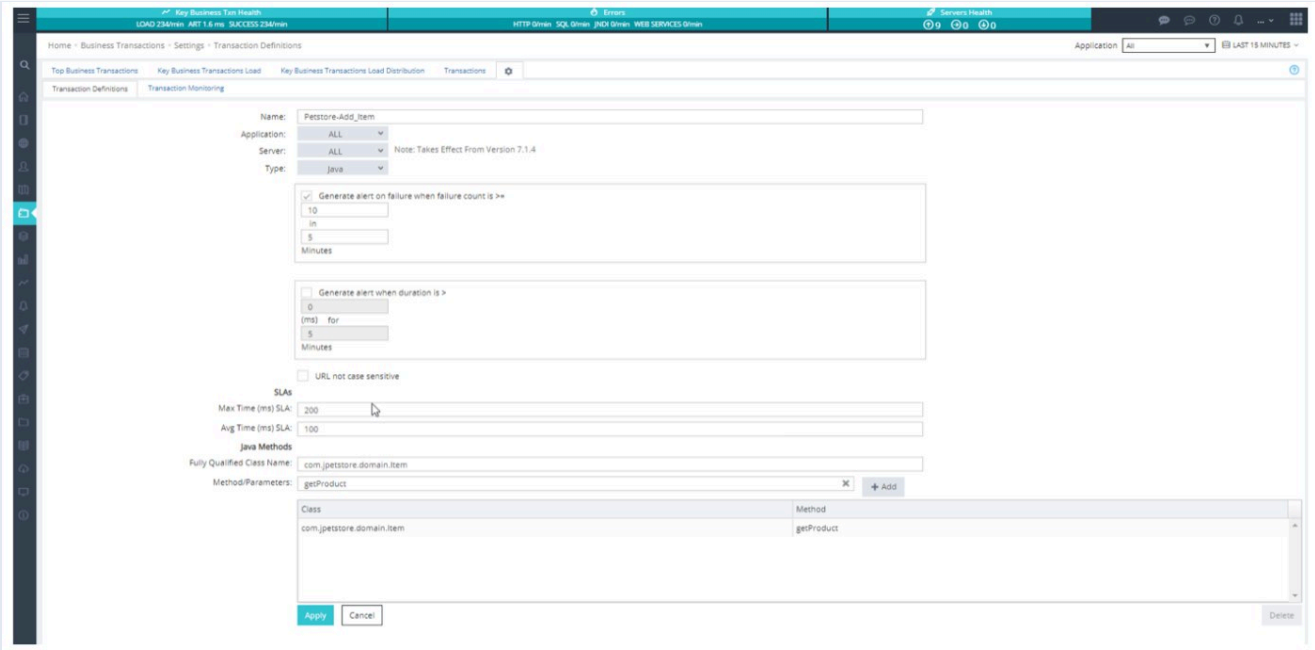
4. Provide a URI pattern (Note: You can use * character as a wild card here)

5. Optionally provide a HTTP parameter pattern if you want to filter further by HTTP parameters. This follows the normal HTTP convention of separating key and value with = sign and separating multiple parameters with & sign. (Note: You can use * character as a wild card for values)

6. Click **Add** button to add the URI pattern to definition. You must add at least one URI pattern. Multiple URI patterns can be added to a definition and HTTP requests matching at least one URI pattern will be included in the Business Transaction. 7. Once you have added all URI Patterns, click **Save** button to save the Business Transaction Definition.

Add New Java Business Transaction

To manually add a new Java Business Transaction, click the **New** button on the Transaction Definition page



1. Provide a **Name** for the Business Transaction

2. Select **Java** from the **Type** drop down box

3. Provide average and maximum SLA values. Leave them blank or set to zero if you don't want to enforce SLA values for this transaction definition

4. Provide fully qualified class name (Note: You can use * character as a wild card for the class name)

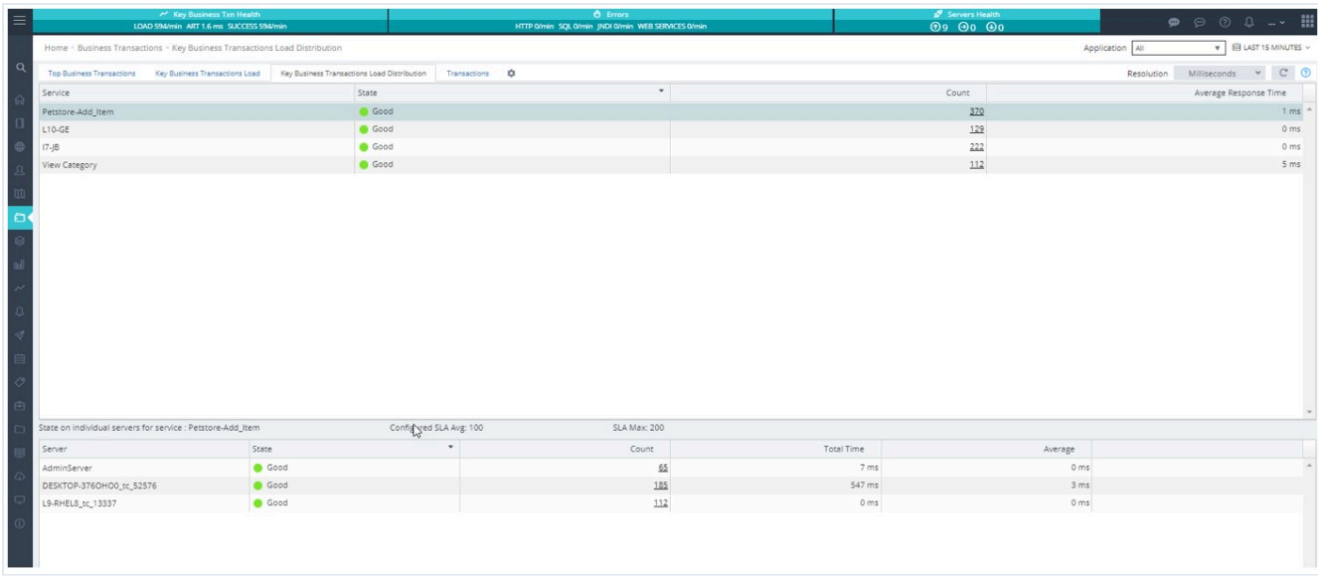
5. Provide a method name (Note: You can use * character as a wild card here)

6. Click **Add** button to add the class/method pattern to definition. You must add at least one class/method pattern. Multiple patterns can be added to a definition and method calls matching at least one pattern will be included in the Business Transaction.

7. Once you have added all patterns, click **Save** button to save the Business Transaction Definition.

Key Business Transactions Load Distribution View

This view displays the state of Key Business Transactions across all servers.



If a Transaction meets SLA values defined for it across all servers, it is shown as in **Good** state. If its not meeting SLA values across all servers, the state will be **Failed**. If its not meeting SLA values on some servers then the state will be **Partial Failure**.

Clicking on a Business Transaction will display the state of the transaction on individual servers in the lower table.

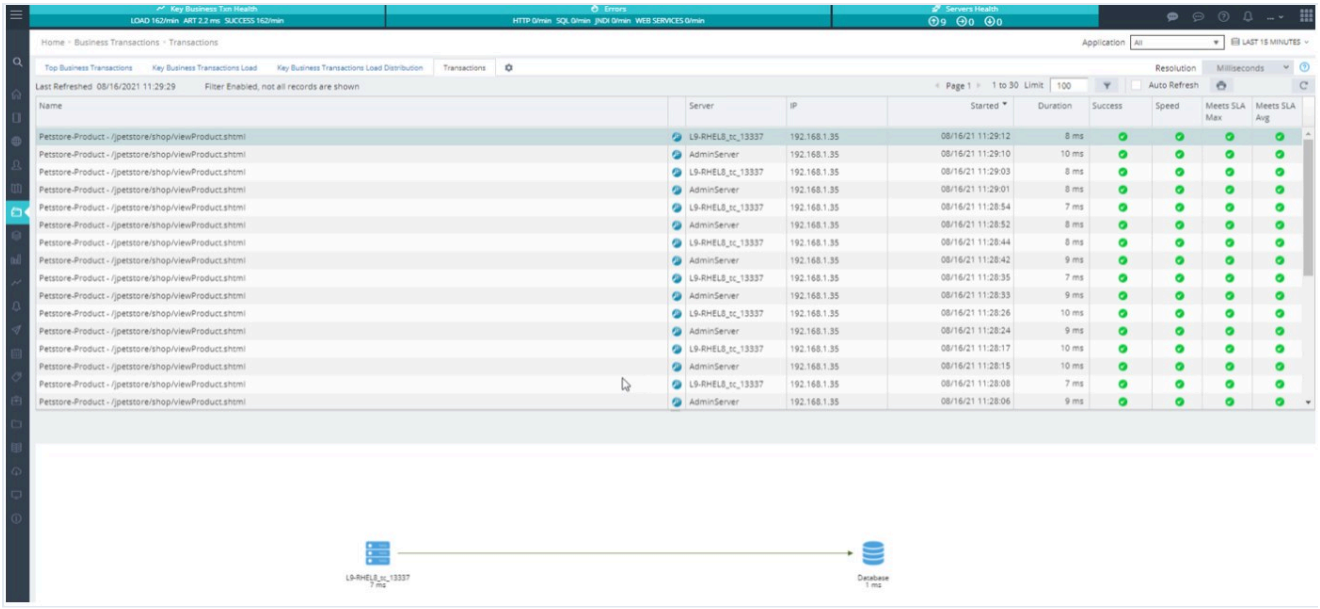
Double clicking on a Transaction in the top table will take you to the Transactions tab and will set the search filter to the selected transaction.

Double clicking on a Transaction in the bottom table will take you to the Transactions tab and will set the search filter to the selected transaction and server.

The states displayed on Summary View are calculated periodically for performance reasons and are not updated real time.

Transactions View

Transaction view displays transaction execution stats if Per-Transaction Monitoring is enabled. Double clicking on any transaction displayed here will take you to the detailed view where you can see the in parameters, exceptions if any and trace details if transaction tracing is enabled.

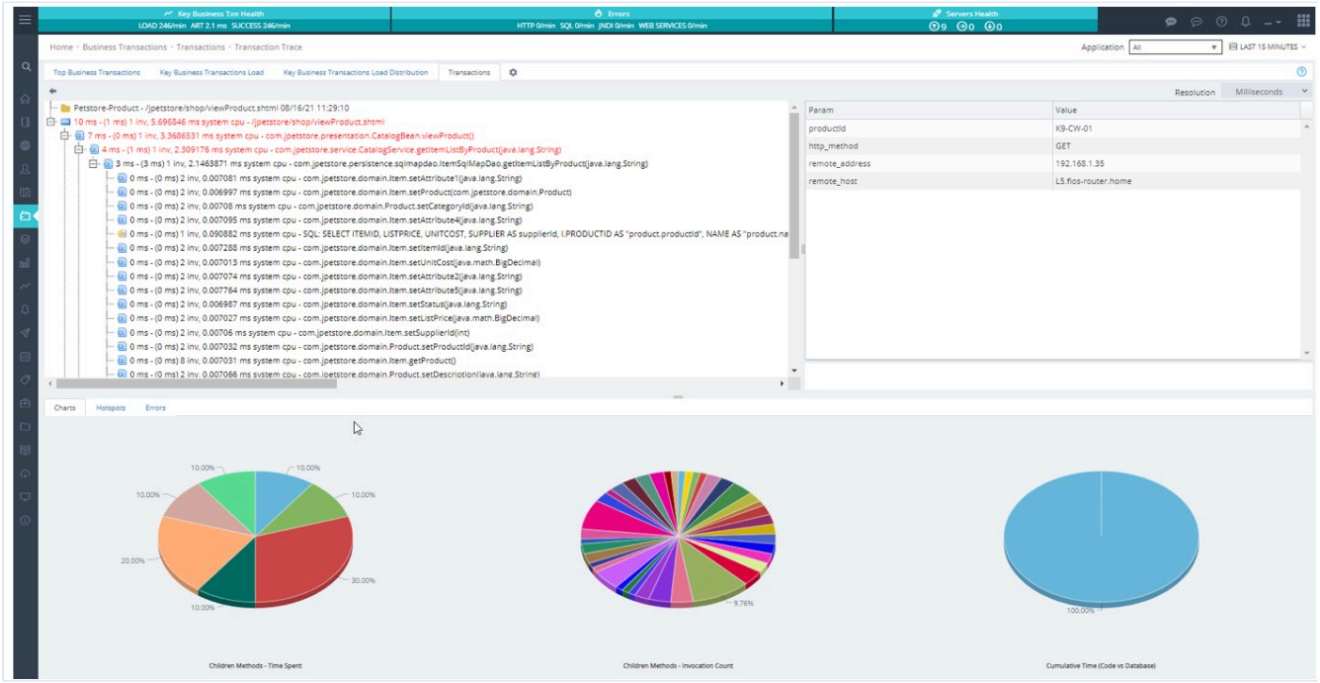


Success - If transaction was completed normally without an exception thrown from the method (applies only to Java transactions).

Speed - If transaction duration was less than the average * (User defined value in Business Transaction Configuration).

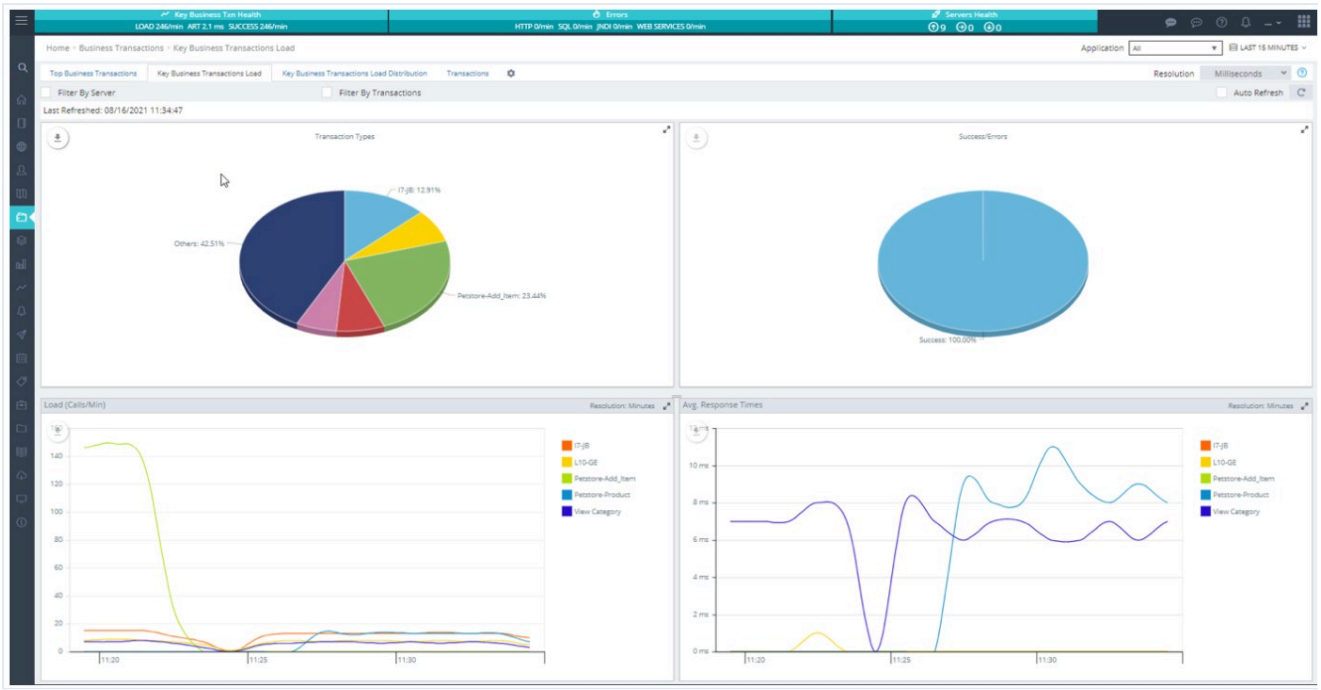
Meets SLA Max - Indicates transaction duration met the configured SLA Max value.

Meets SLA Avg - Indicates transaction duration met the configured SLA Average value.



Key Business Transactions Load View

Using this tab you can view the performance stats of key business transactions across the environment as well as on a per server basis. You must enable Business Transaction Profiling in order to gather data to be displayed on the dashboard.



Configuring Distributed Transaction Tracing

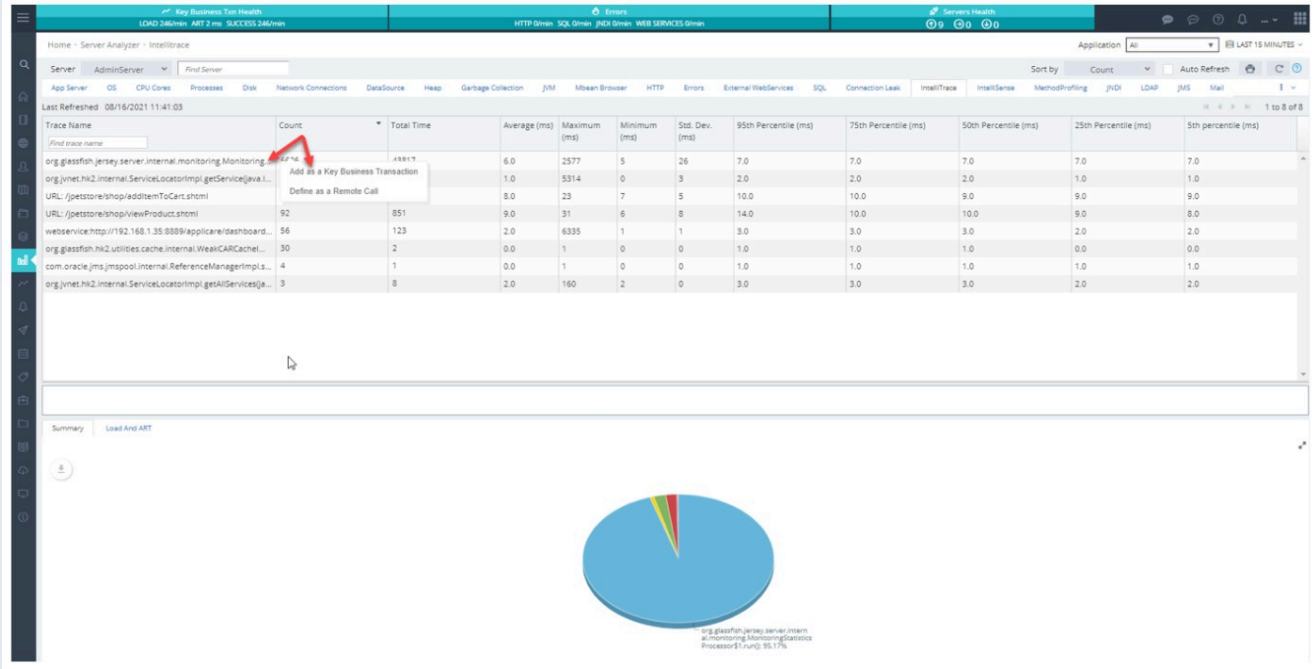
Applicare BTM can track transactions that span multiple servers (JVMs) and display the consolidated call graph consisting of executions on multiple servers in a single UI.

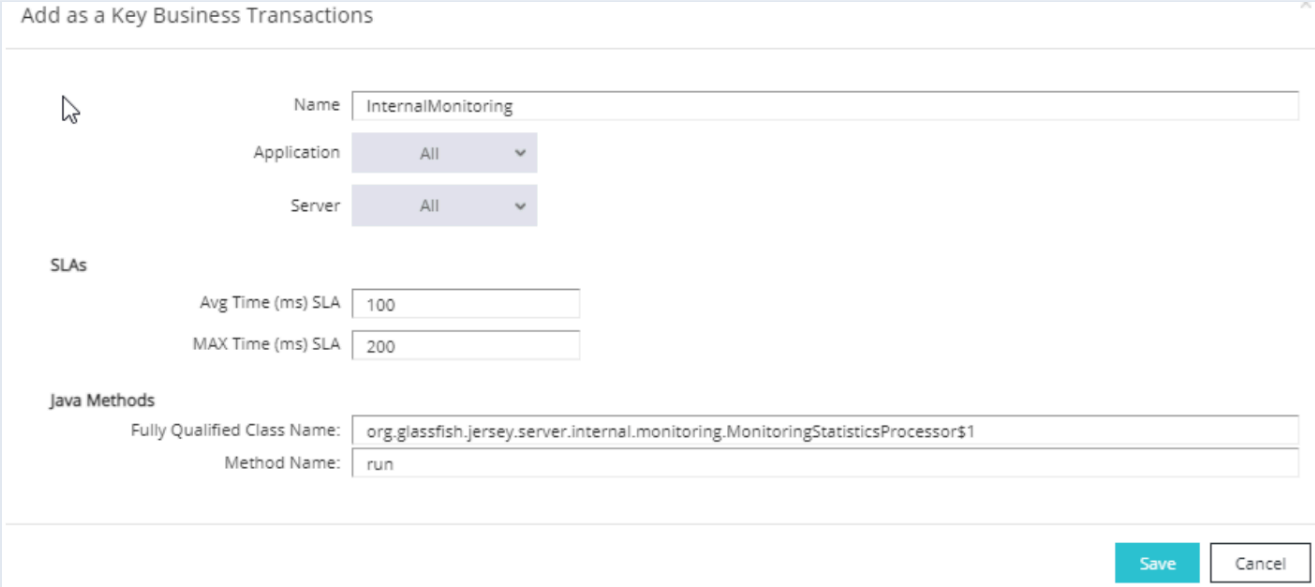
Note: All servers that participates in distributed transactions must have their system clocks synchronized (E.g. Using NTP) for remote transaction tracing to work properly.

There are two ways to enable transaction tracking between two servers.

First Method

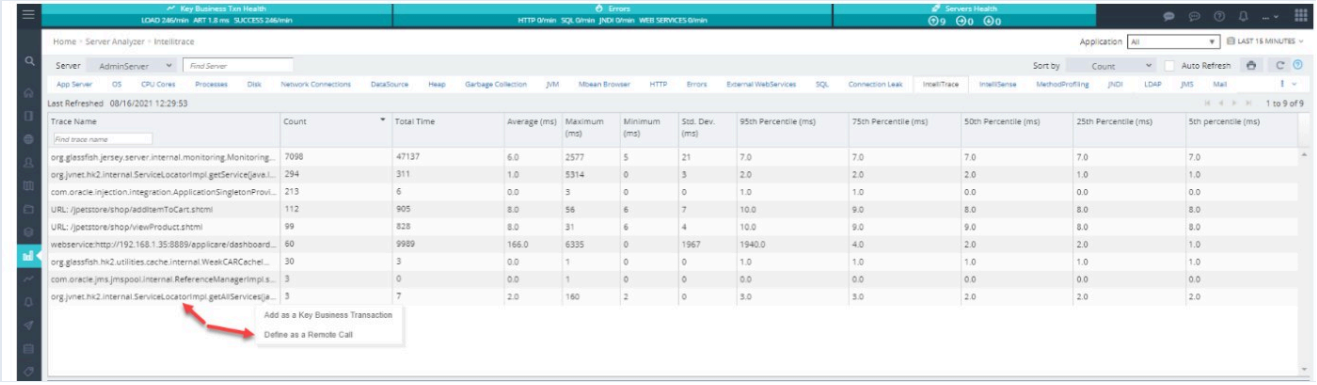
First you must define the remote service as a Java business transaction. You can right click your service method which accepts the remote parameters in IntelliTrace window and select "Define as a Business Transaction" option.

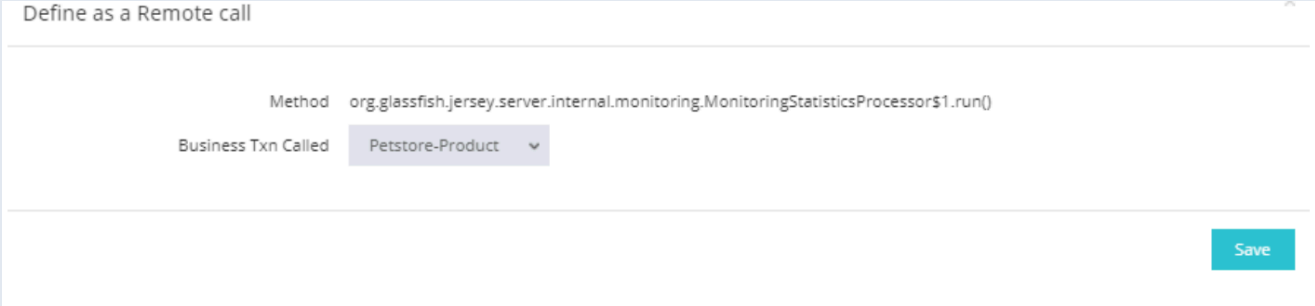




In the above example we are defining the org.glassfish.jersey.server.internal.monitoring.MonitoringStatisticsProcessor$1.run() method which is the service method that parameter as a Business Transaction named "**Petstore-Product**".

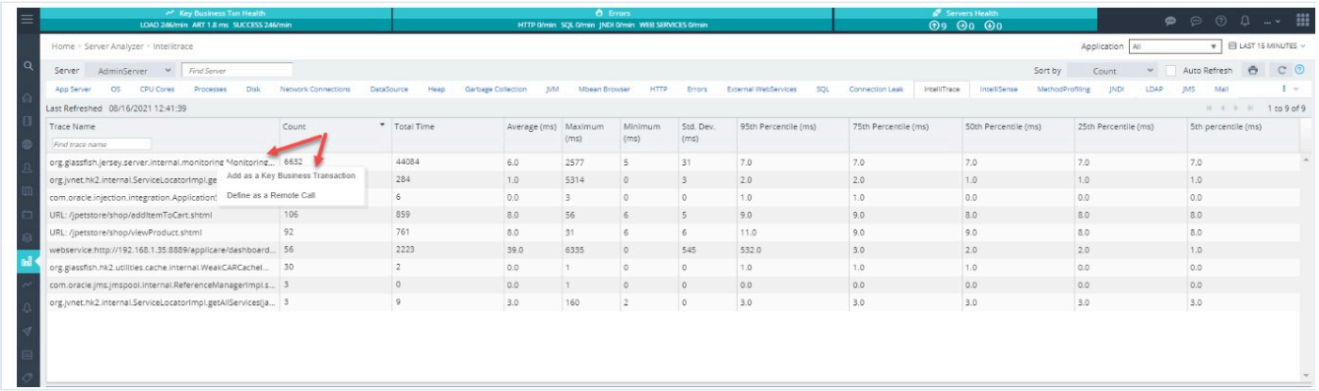
Next we define the method that calls the remote service as a remote method by right clicking the method name in IntelliTrace or in the BTM trace window and selecting "**Define as a Remote Call**" option. Be sure to choose the method that accepts the parameters which are sent to the remote service.

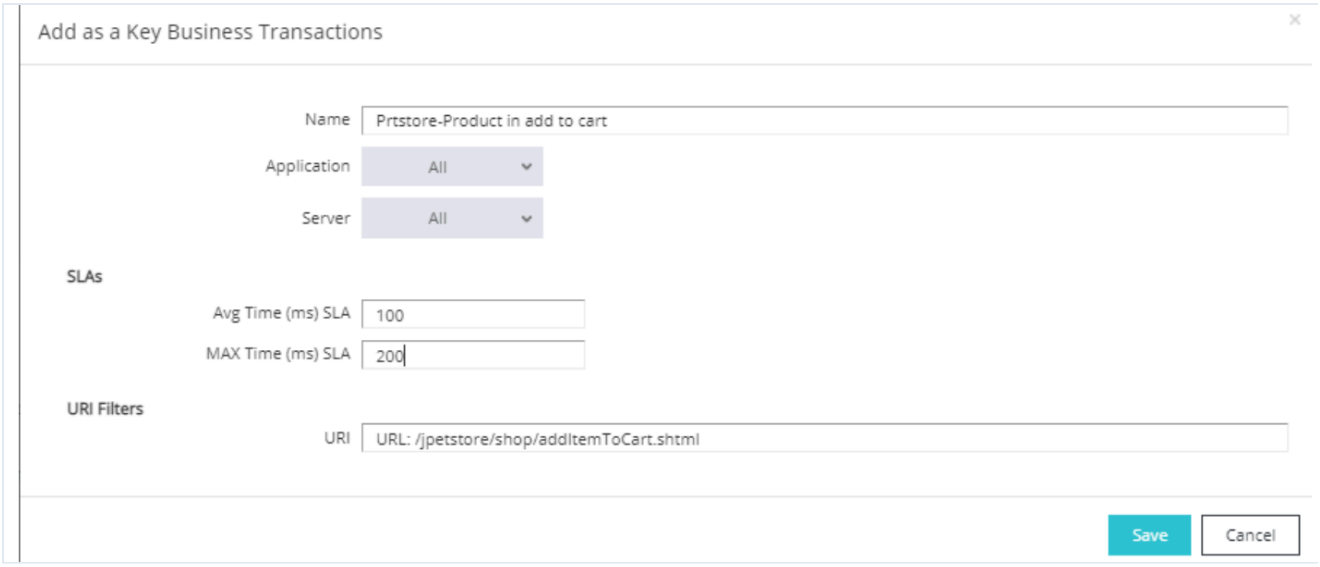




In the example above we define the method jpetstore/shop/addItemToCart.shtml as a remote method call and we select the previously defined **Pet store-Product in add to cart** service as the Business Transaction called by this method.

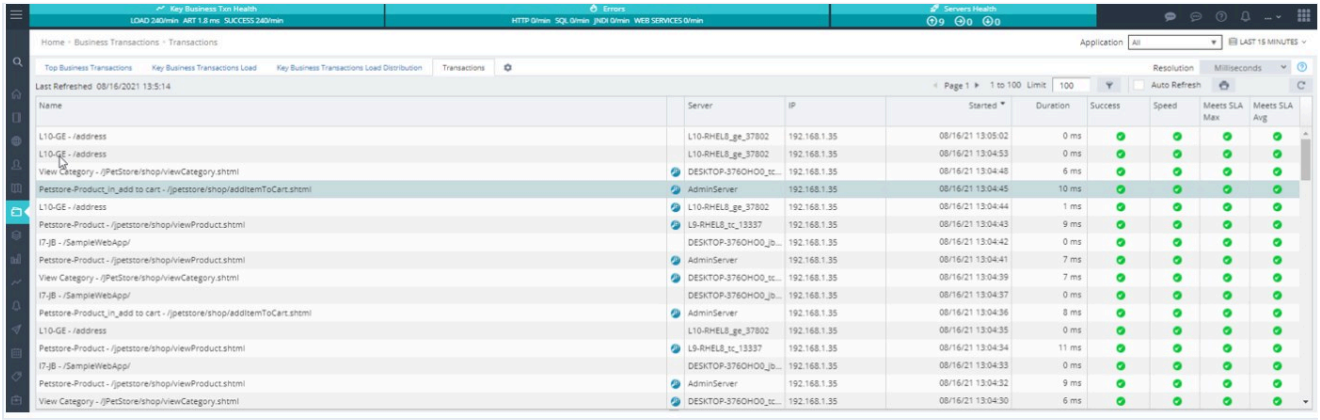
Next we will define the HTTP call which makes the remote method call as a business transaction too

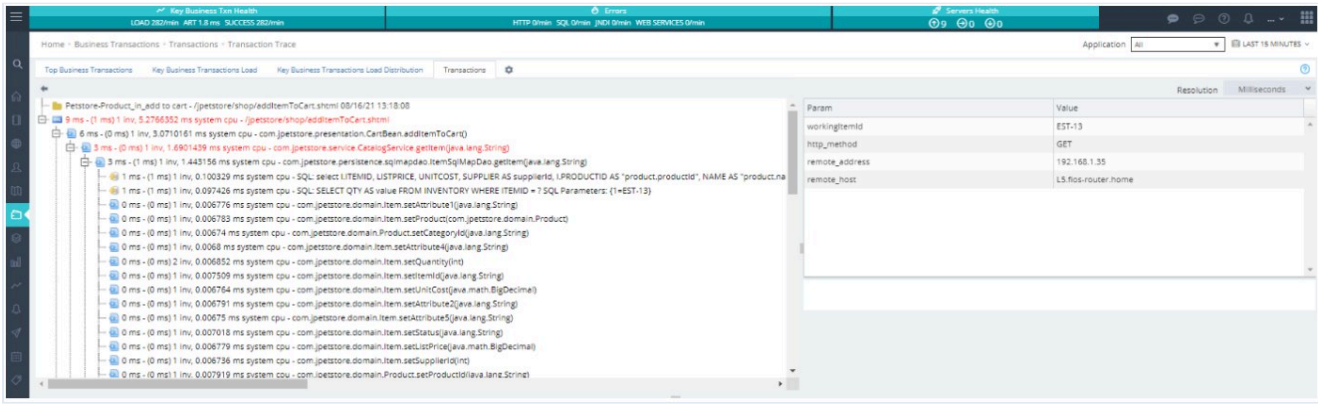




Now that both transactions and remote method calls are defined, you can monitor the transactions as they are invoked on the server and track their flow across different servers.

The following two images illustrate a **Validate CC** transaction invoked on the AdminServer. By double-clicking, you can view the call graph for the AdminServer and the call graph for the remote service on the Tomcat server, which was invoked as part of the transaction on the AdminServer.



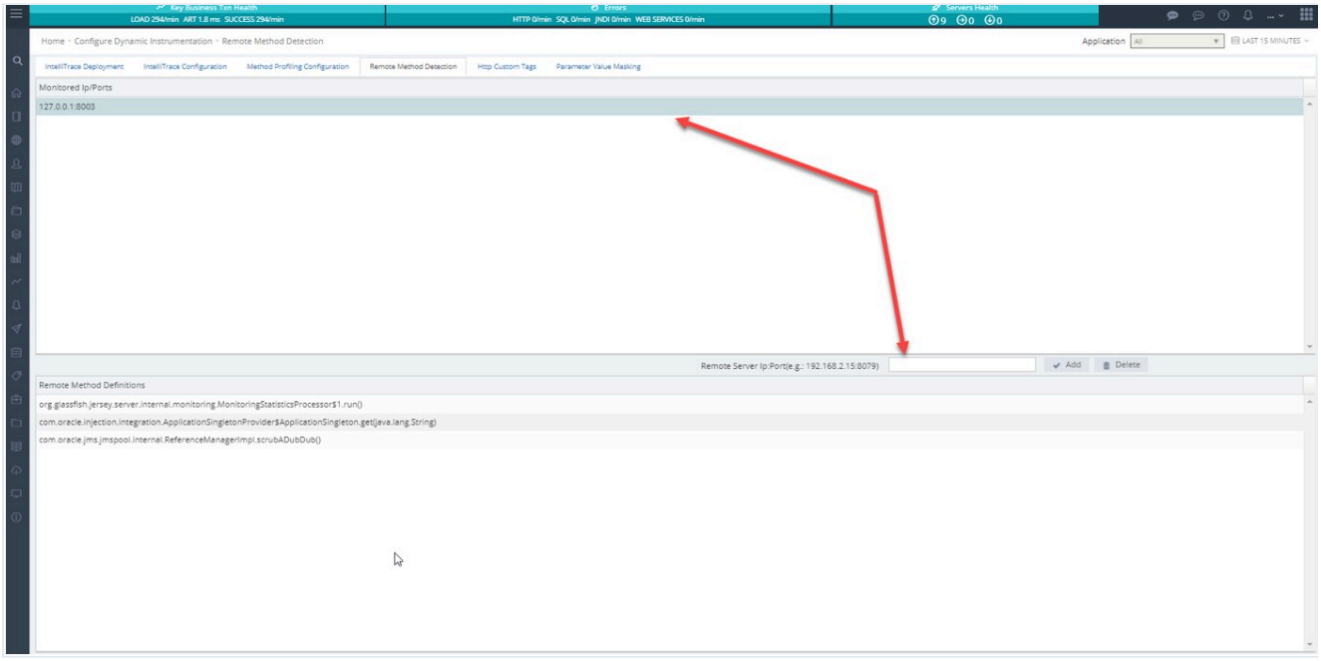


Second Method

Second approach is to use Network Connection Profiling to track transactions and as in the first method, you will have to define the calling and the called business transactions. You will not need to define the method that calls the remote transaction as a Remote Method.

For this approach to work, Network Connection Profiling must be enabled on all participating servers. You can do this by going to **Profiling -> Configure Dynamic Instrumentation** page and deploying Network Connection Profiling in Advanced mode.

Once you enable Network Connection Profiling, go to the Remote Method Definition tab on the same page and add the server IP and port where the remote transactions are running on.



Monitoring with Applicare

USING APPLICARE › TRANSACTIONS

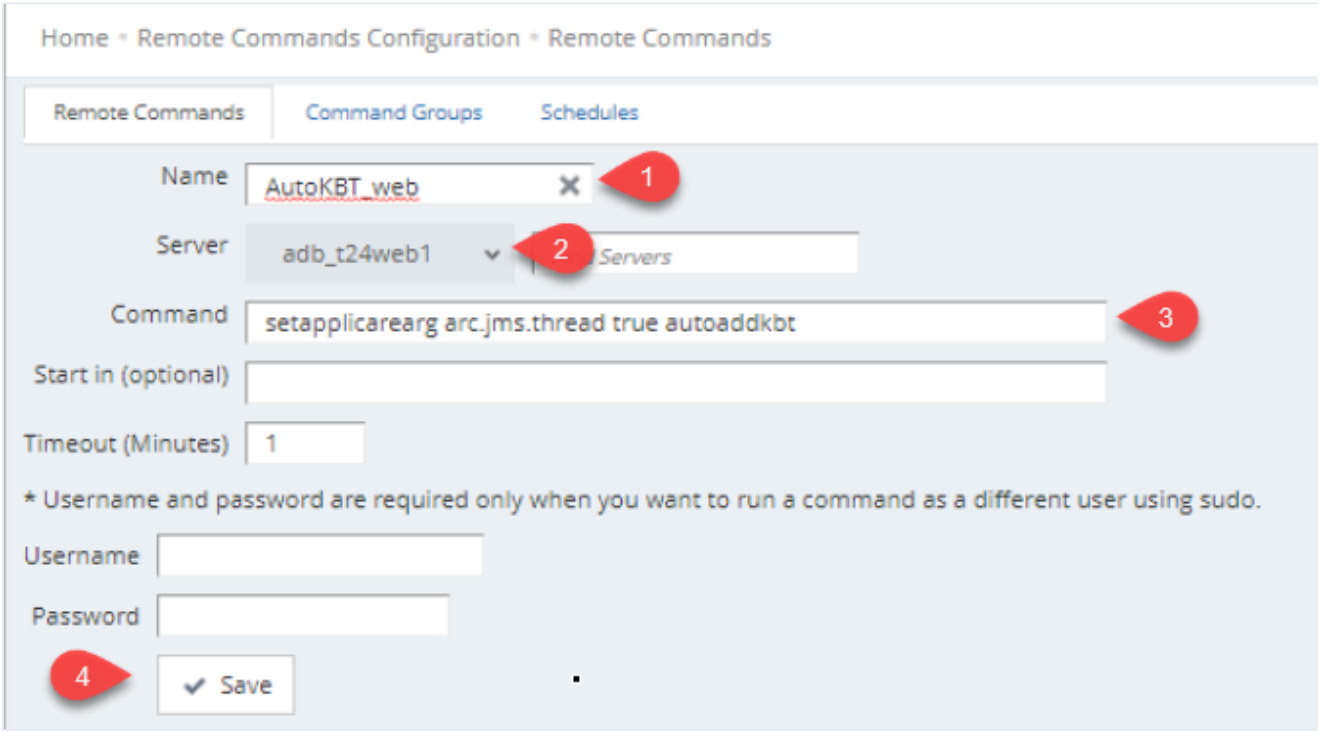
Automatically add KBT configuration

Applicare 9.2 · Updated September 9, 2024 · **Pending technical review**

Step 1: Log in to Applicare console, navigate to the **Remote Commands** screen, and then click the **setting** button.

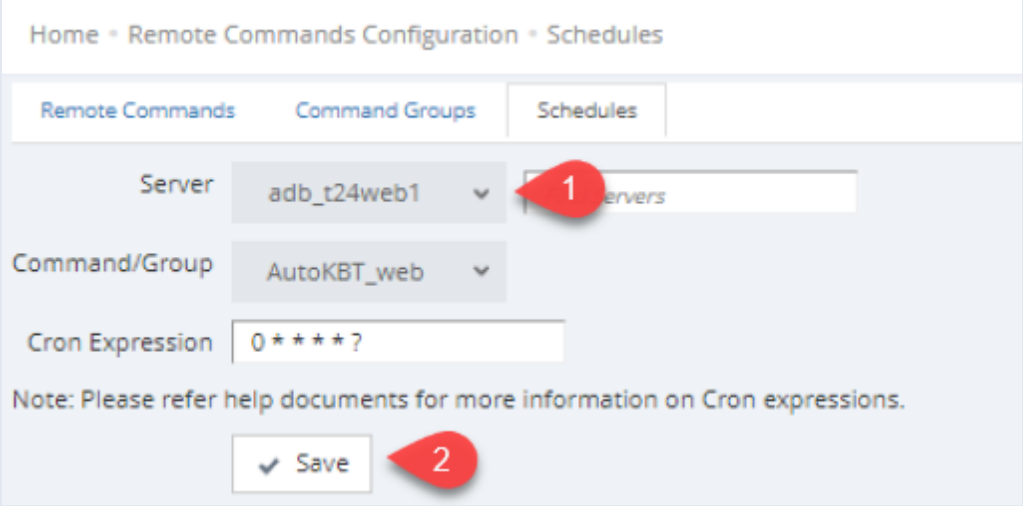
Step 2: Enter a name in the **Name** field, select a server with either **sender or receiver** application from the **Server dropdown**, paste the command into the **command field** and then click **save** button.

```text
setapplicarearg arc.jms.thread true autoaddkbt
```



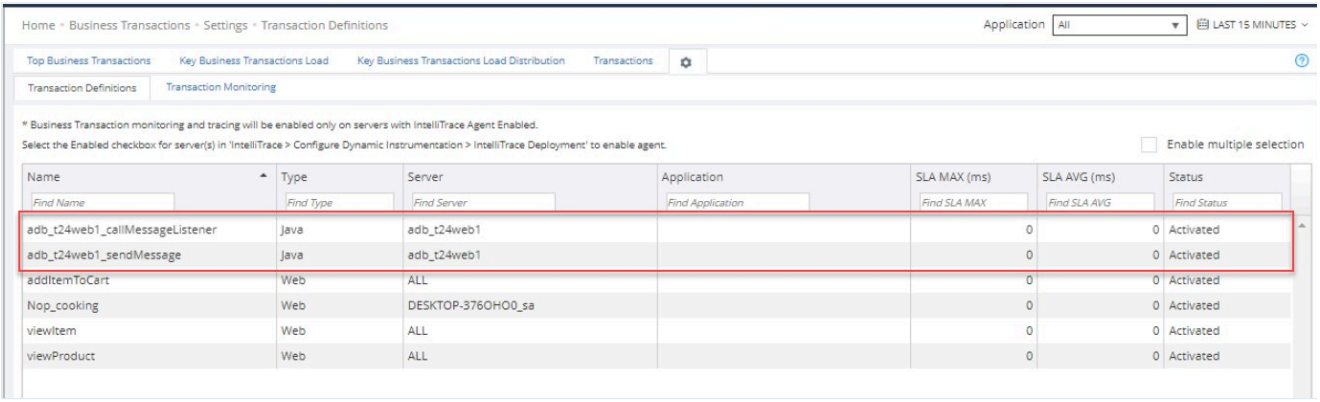
Step 3: On the same screen, click on the third tab **schedules**.

Step 4: Choose the server configured for the remote command from the **server dropdown**, then click the **save** button.



Step 5: After 2-3 minutes, navigate to **Business Transactions-> Settings-> Transaction** Definitions screen to verify that the KBT has been created.

KBT creation only occurs when a JMS transaction is executed.



If you prefer not to auto add KBT and only view KBT trace details in remote commands, remove the "**autoaddkbt**" keyword from the command.

```
TEXT
setapplicarearg arc.jms.thread true
```

USING APPLICARE › TRANSACTIONS

Download IntelliTrace report

Applicare 9.2 · Updated January 8, 2025 · **Pending technical review**

IntelliTrace report for Http:

Step 1: Go to the report screen in the left-hand menu.

Step 2: Below is a list of which options to select from which dropdown.

a.) **Select Report:** Select the "**Intellitrace report"** option in this dropdown.

b.) **Data:** Select the "**http"** option in this dropdown.

c.) **Order by:** Select the "**MaxAvg.Time"** option in this dropdown.

d.) **Server:** In this dropdown, select the "**server"** for which you want intellitrace data.

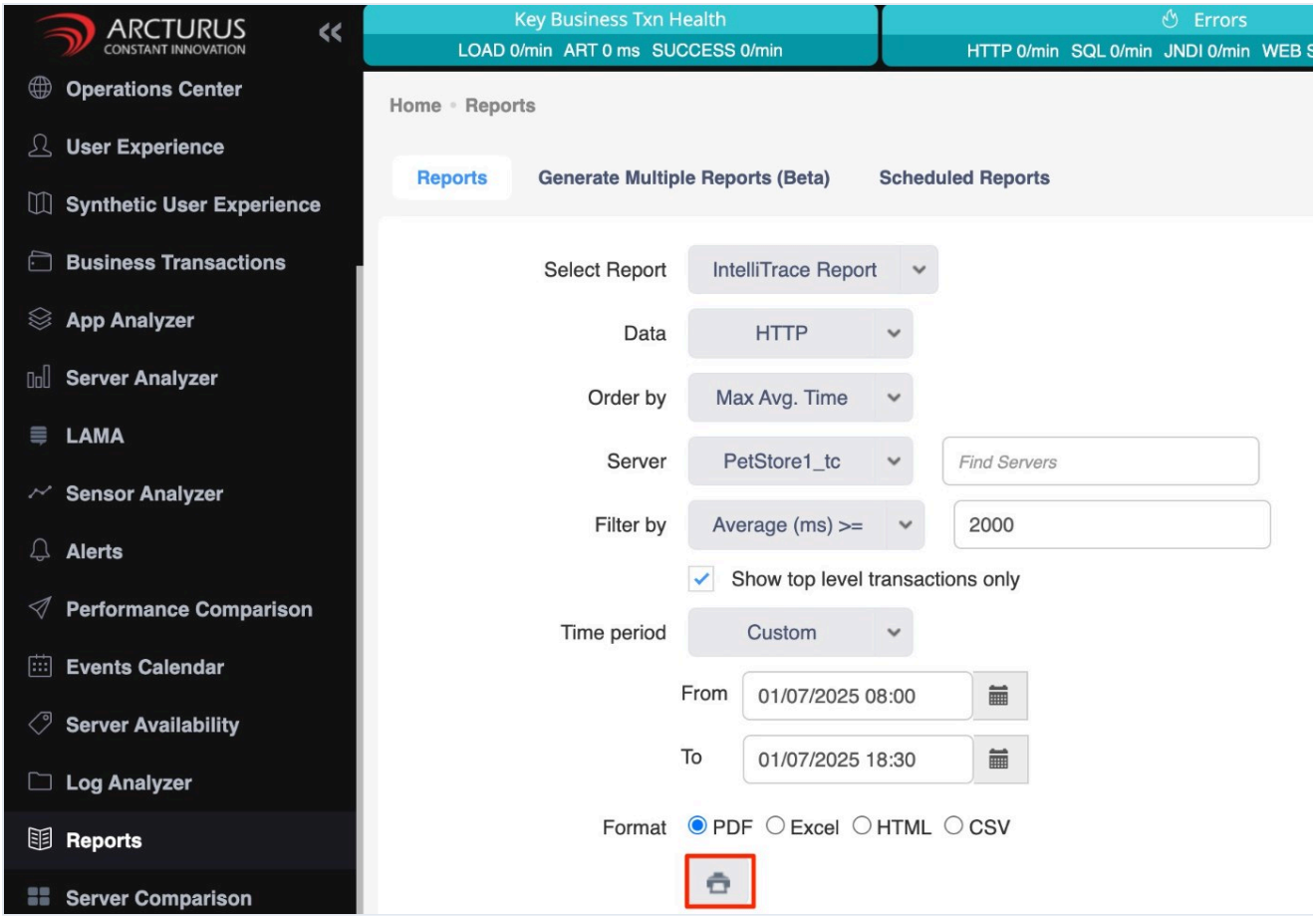
e.) **Filter by:** In this dropdown, select the "**Average>="** option and enter "**2000"** in the filter value text box.

f.) Enable the Show top level transactions only checkbox.

g.) **Time period:** Select "**Custom**" from this dropdown menu, then specify the desired data time period using the "**From**" and "**To**" fields.

h.) **Format:** Choose your preferred output format (**PDF, Excel, HTML,** or **CSV**).

Step 3: Enter the values as shown above, then click the **print icon**.



IntelliTrace report for SQL:

Step 1: Go to the report screen in the left-hand menu.

Step 2: Below is a list of which options to select from which dropdown.

a.) **Select Report:** Select the "**Intellitrace report"** option in this dropdown.

b.) **Data:** Select the "**SQL"** option in this dropdown.

c.) **Order by:** Select the "**MaxAvg.Time"** option in this dropdown.

d.) **Server:** In this dropdown, select the "**server"** for which you want intellitrace data.

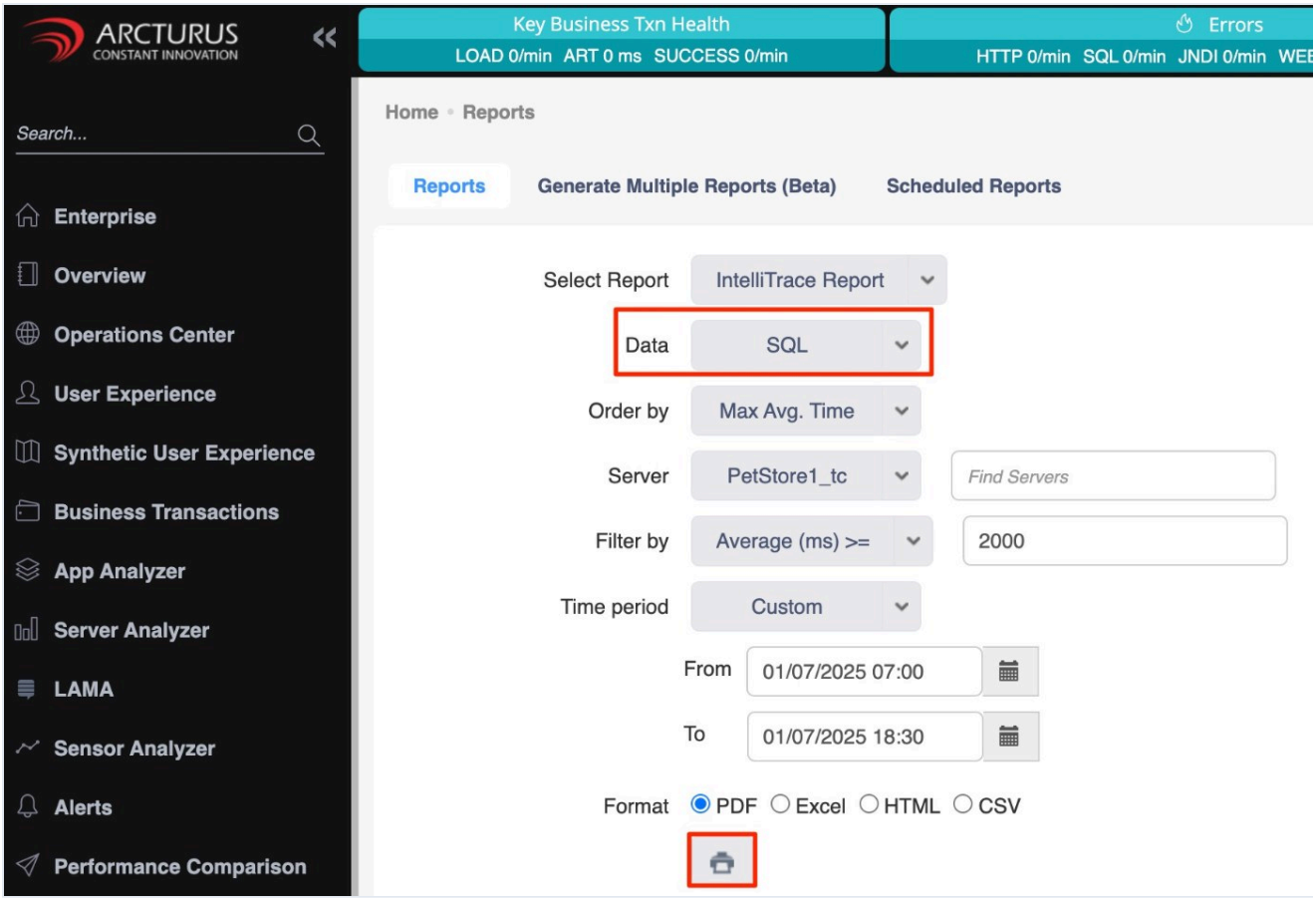
e.) **Filter by:** In this dropdown, select the "**Average>="** option and enter "**2000"** in the filter value text box.

f.) Enable the Show top level transactions only checkbox.

g.) **Time period:** Select "**Custom**" from this dropdown menu, then specify the desired data time period using the "**From**" and "**To**" fields.

h.) **Format:** Choose your preferred output format (**PDF, Excel, HTML,** or **CSV**).

Step 3: Enter the values as shown above, then click the **print icon**.


ARCTURUS
CONSTANT INNOVATION
Search...
Enterprise
Overview
Operations Center
User Experience
Synthetic User Experience
Business Transactions
App Analyzer
Server Analyzer
LAMA
Sensor Analyzer
Alerts
Performance Comparison
Key Business Txn Health
LOAD 0/min ART 0 ms SUCCESS 0/min
Errors
HTTP 0/min SQL 0/min JNDI 0/min
Home • Reports
Reports
Generate Multiple Reports (Beta)
Scheduled Reports
Select Report
IntelliTrace Report
Data
SQL
Order by
Max Avg. Time
Server
PetStore1_tc
Find Servers
Filter by
Average (ms) >=
2000
Time period
Custom
From
01/07/2025 07:00
To
01/07/2025 18:30
Format
PDF
Excel
HTML
CSV

USING APPLICARE › TRANSACTIONS

Configuration for KBT with parameter

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

If you need to capture only a specific URL in KBT, then follow these steps:

Step 1: Navigate to **Business Transactions-> Settings-> Transaction Definitions** in Applicare, and then click the **New** button at the bottom right.

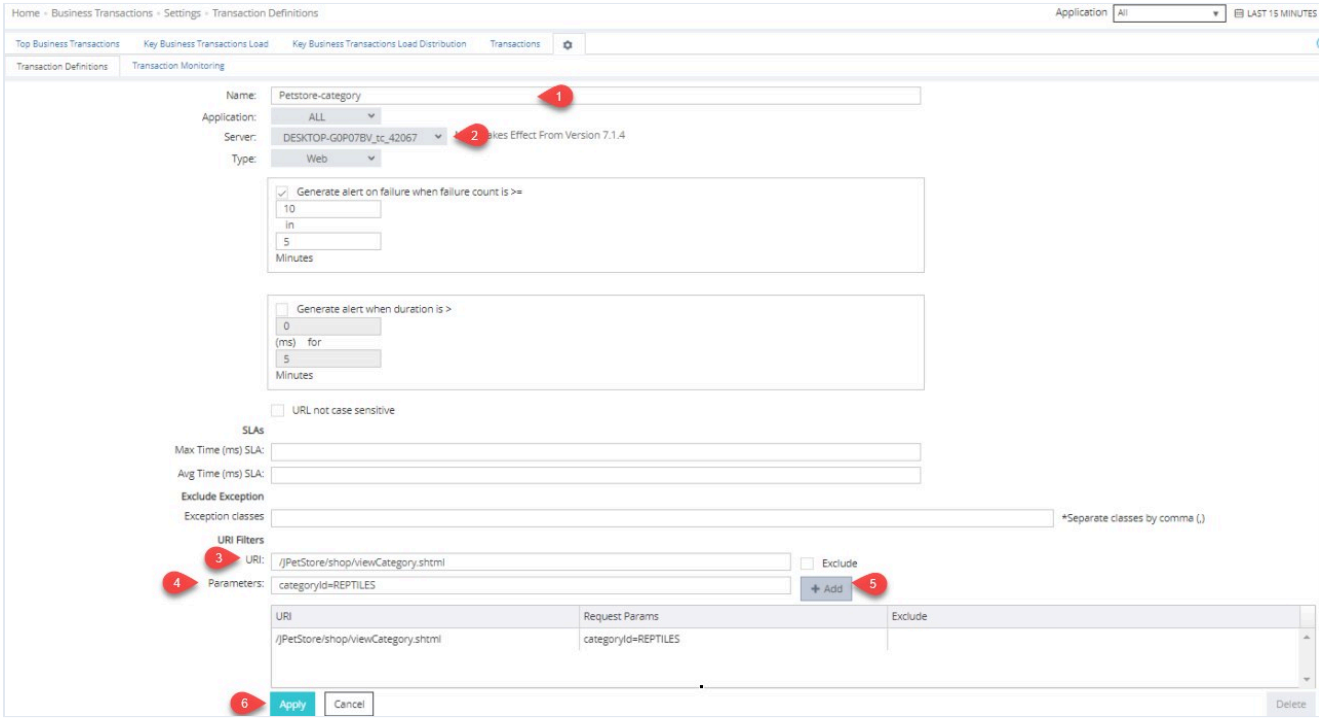
Step 2: Enter the **KBT** name in the Name text box, then click the **server dropdown** and select the appropriate **server**.

Step 3: Enter the application URL in the **URL** text box, then input the parameter you want to add in the **parameter** text box. Click on **Add** button, and then click **Apply**.

Example:

URL = /JPetStore/shop/viewCategory.shtml

Parameter = categoryId=REPTILES



If you need to fetch a single parameter with different values for a specific URL in KBT, follow these steps:

Step 1: Navigate to **Business Transactions-> Settings-> Transaction Definitions** screen in Applicare and click the **New** button at the bottom right.

Step 2: Enter the **KBT** name in the **Name** text box, then click the **server dropdown** and select the desired **server**.

Step 3: Enter the **application URL** in the URL text box, but this time parameter with a different value. After adding the parameter, click the **Apply** button.

Step 4: Repeat the steps in **step 3,** to add another parameter, then click the **Apply** button.

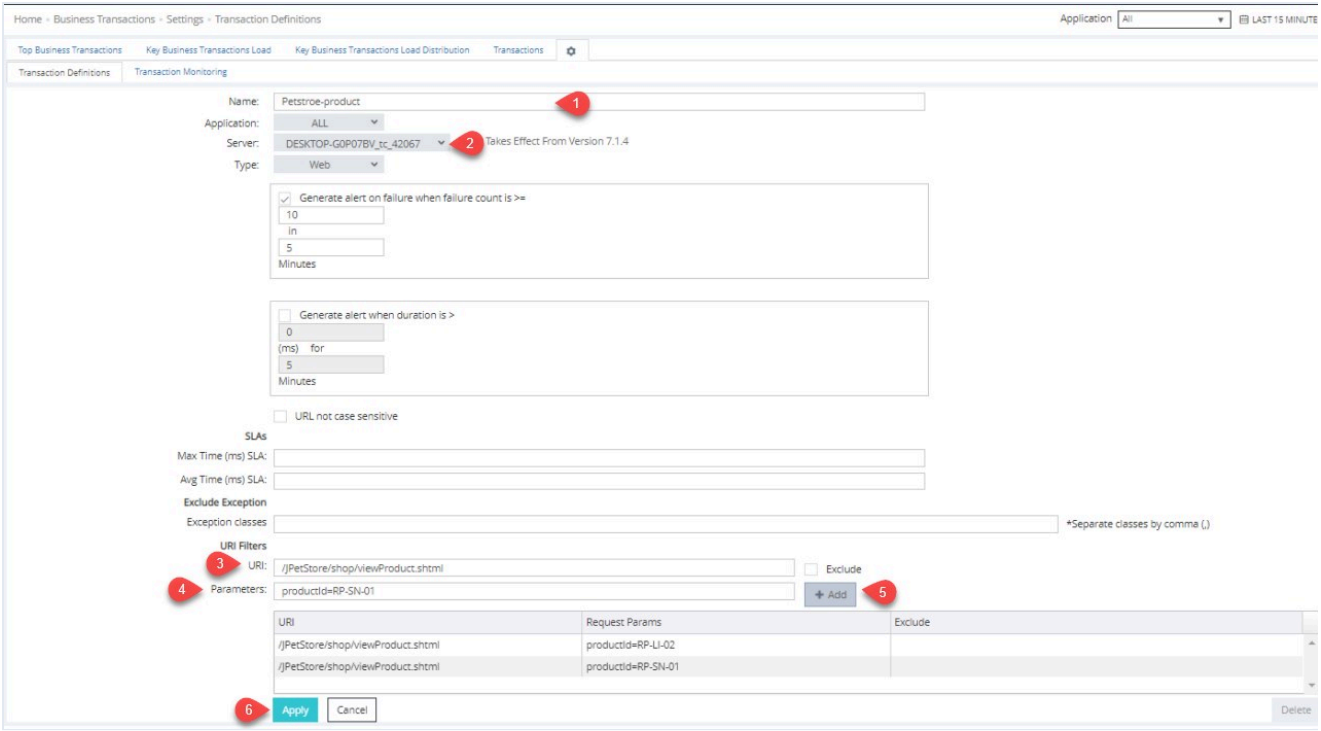
e.g.

URL = /JPetStore/shop/viewProduct.shtml

Parameter = productId=RP-LI-02

URL = /JPetStore/shop/viewProduct.shtml

Parameter = productId=RP-SN-01



If you need to capture only different parameters for a specific URL in KBT. Follow these steps:

Step 1: Navigate to **Business Transactions-> Settings-> Transaction Definitions** in Applicare, and click the **New** button at the bottom right.

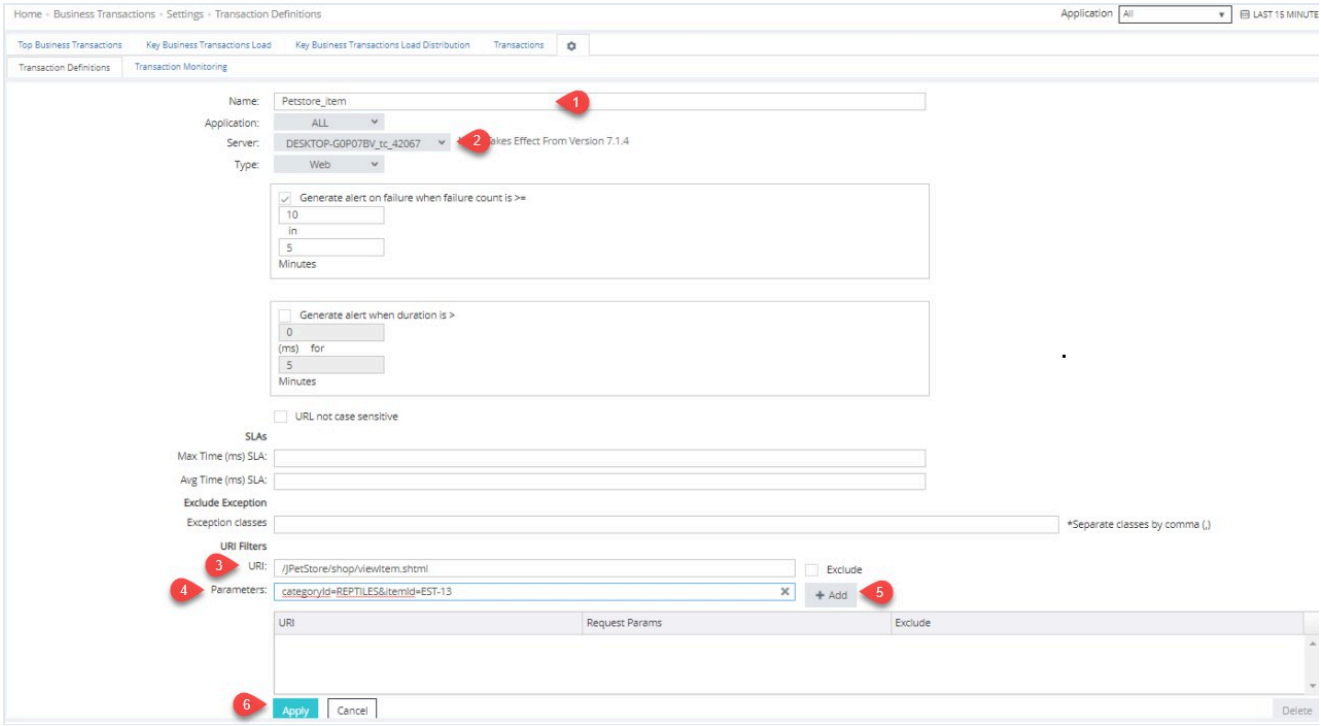
Step 2: Enter the **KBT** name in the **Name** text box, then click the **server dropdown** and select the appropriate **server**.

Step 3: Enter the application URL in the **URL** text box, then input the different parameter you want to add in the **Parameter** text box, click the **Add** button.

e.g.

URL = /JPetStore/shop/viewItem.shtml

Parameter = categoryId=REPTILES&itemId=EST-13



If want to capture all request from a specific parameter in KBT, follow these steps:

Step 1: Navigate to **Business Transactions-> Settings-> Transaction Definitions** in Applicare and click the **New** button at the bottom right.

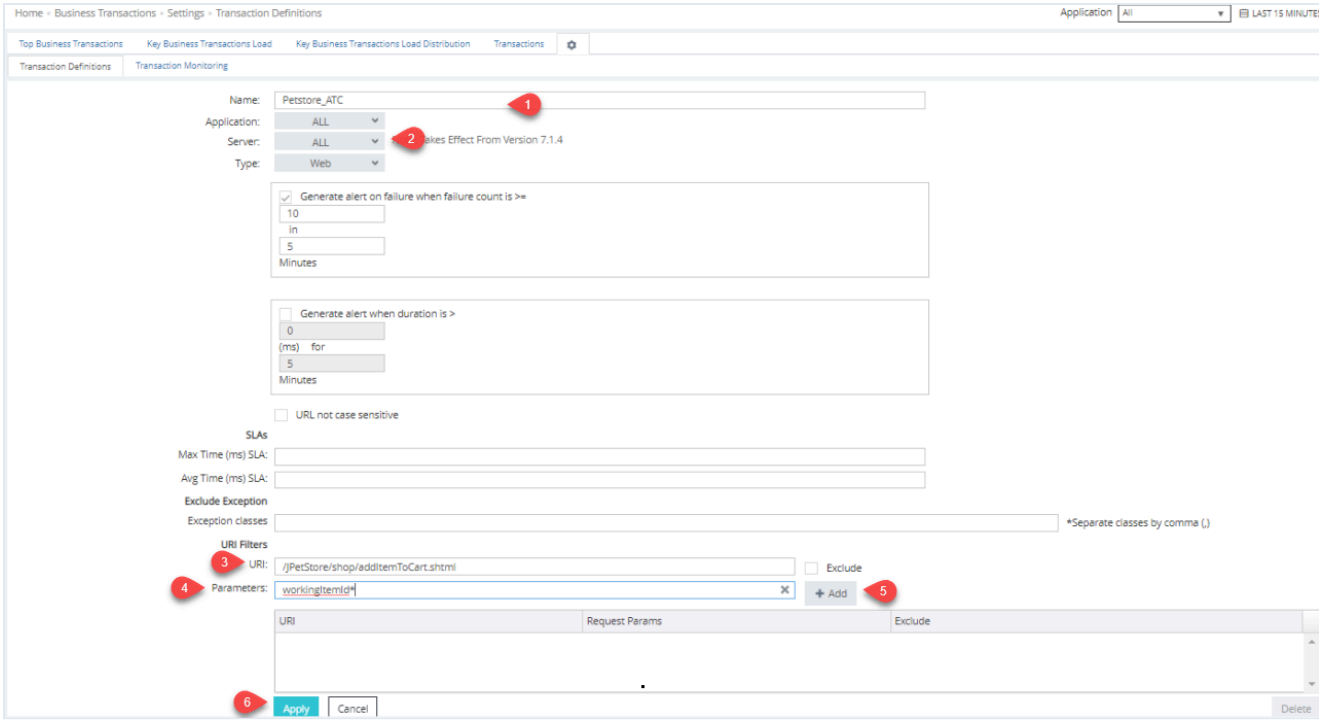
Step 2: Enter the **KBT** name in the **Name** text box, then click the **server dropdown** and select the desired **server**.

Step 3: Enter the application URL in the **URL** text box, then input the **parameter** you want to include in the Parameter **text box,** click the **Add** button.

e.g.

URL = /JPetStore/shop/addItemToCart.shtml

Parameter =workingItemId*



Monitoring with Applicare

USING APPLICARE › INFRASTRUCTURE

Server Analyzer

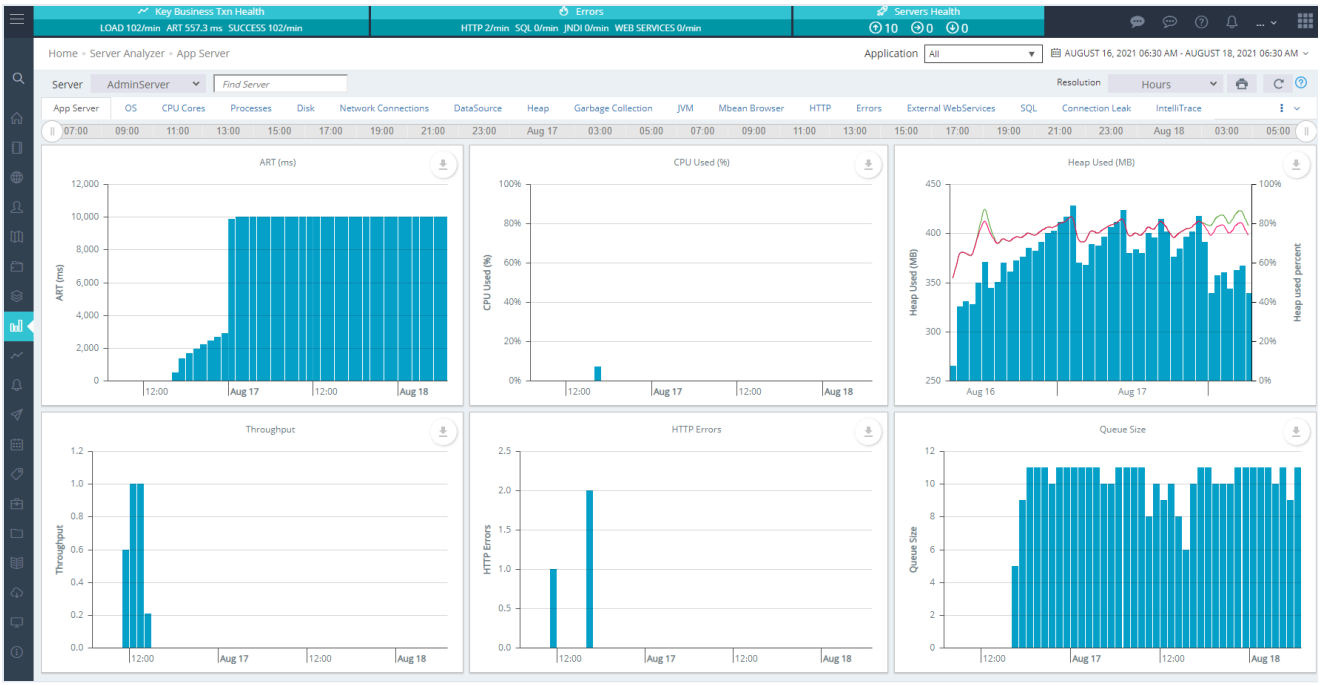
Applicare 9.2 · Updated April 5, 2025 · **Pending technical review**

Overview

The Server Analyzer gives you a quick look at your server's health and activity. Select a time frame at the top to view historical data. The server list will show only those related to chosen application. To zoom in, click and drag over the area you want to explore.

App Server Analyzer

The AppServer tab offers a quick overview of application server performance. It helps identify potential bottleneck, such as queue buildup, which indicates the server is struggling to process incoming request. Issues like garbage collection or high CPU can also be mentioned here. The AppServer Analyzer provides a correlated view to pin point the effects of change in performance metrics.



If a time period is selected on top of the home dash then history view is visible. Average Response Time (ART) is a great indicator of a server's performance. A significant increase in ART means things are getting stuck somewhere in the app server. Similarly heap used, throughput, queue size, http errors and CPU used are available.

Throughput: Server throughput is the number of requests served per second by the server. **Busy Threads**: The number of Server threads thats are busy serving requests at any given time.

OS Analyzer

The OS Internals Analyzer provides a fast and convenient window into your physical server's/VM's performance including # of processes, tcp connections, CPU, memory and storage performance.

Processes: Number of processes running on the machine. Running too many processes may cause memory contention (paging), I/O contention or excessive context switching. This contention can reduce system throughput if parallel execution were not used.

CPU %- Cpu usage in percent. High CPU usage is not a significant cause for concern as long as it doesn't occur over the long-term. But if the cpu usage remains high for long term then they system may freeze or will respond too slow.

Process Queue Length- The Processor Queue Length is the number of threads that are ready but currently unable to run on the processor due to another active thread. This count shows ready threads only, not threads that are running. A bottleneck on the processor may be thought to occur where the number of threads in the queue is more than 2 times the number of processor cores over a continuous period.

Memory Free %- Percentage of free memory available in physical memory. Memory can have a huge impact on a system is performance. As physical free space increases, the machineï¿½s dependency on virtual memory decreases. The less a computer has to depend on virtual memory, the more efficiently that machine will run.

Memory Free- Free memory available (in Mega Bytes) in physical memory.

Page File Usage %- Displays the percentage of the paging file that is currently in use.

Disk Transfers/sec- A high Disk Transfers/sec value may occur due to a burst of disk transfer requests by either an operating system or application.

Context Switches- This attribute reports the combined rate at which all processors on the computer are switched from one thread to another. High context switches, shows that threads have less time to do their work and the system performance might go down.

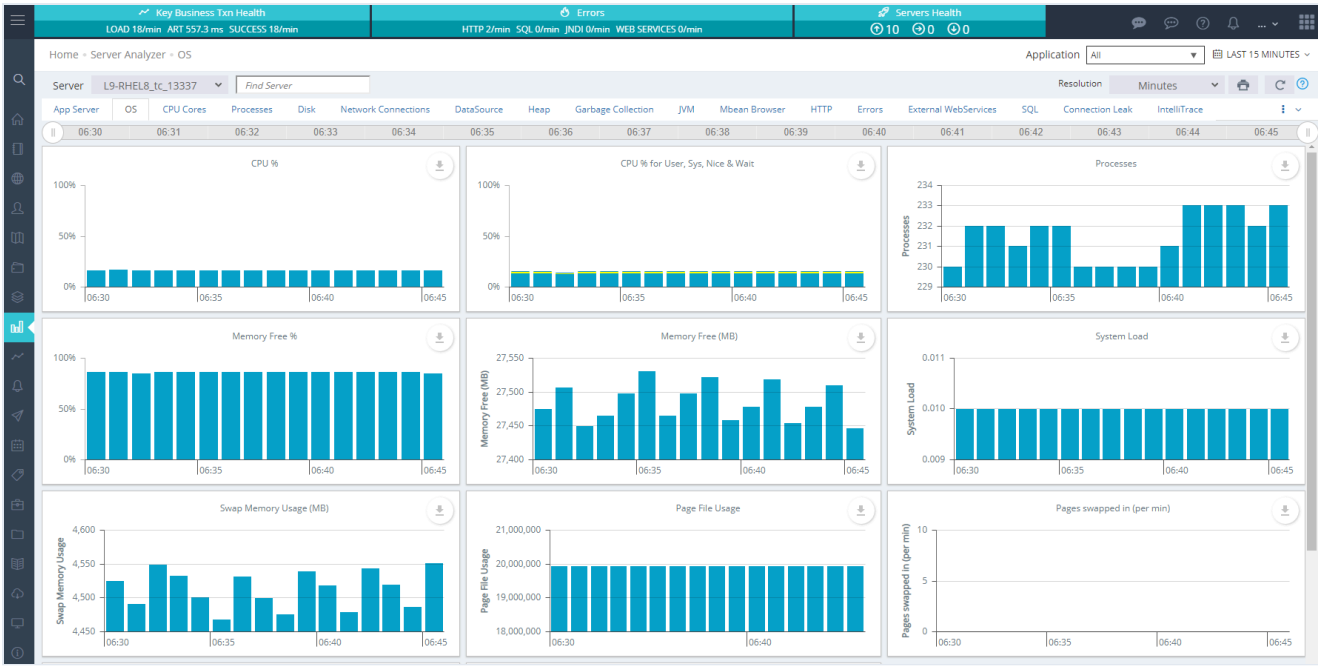
Swap memory usage- Swap usage refers to the amount of virtual memory that is currently being used to temporarily store inactive pages from the main physical memory. Swap space is your safety net for when you run out of RAM. While swapping, processes may get slower, data is still nevertheless processed. However, when you run out of virtual memory, processes are queued and stalled until some memory is freed up.

Pages Swapped In- Number of pages read(in last one minute) from paging/swap file(s) to resolve hard page faults. (Hard page faults occur when a process requires code or data that is not in physical memory, and must be retrieved from disk.)

Pages Swapped Out- Number of pages written(in last one minute) to paging/swap file(s) to free up space in physical memory. Pages are written back to disk only if they are changed in physical memory, so they are likely to hold data, not code. A high rate of pages output might indicate a memory shortage. The operating system writes more pages back to disk to free up space when physical memory is in short supply. Even when swap space is not particularly full, continuously high swap-out rates indicate that the pages that are swapped are actively processed. This means that your system requires more fast physical memory for your workloads, and swap is not helping.

TCP Connections- Total TCP connections count.

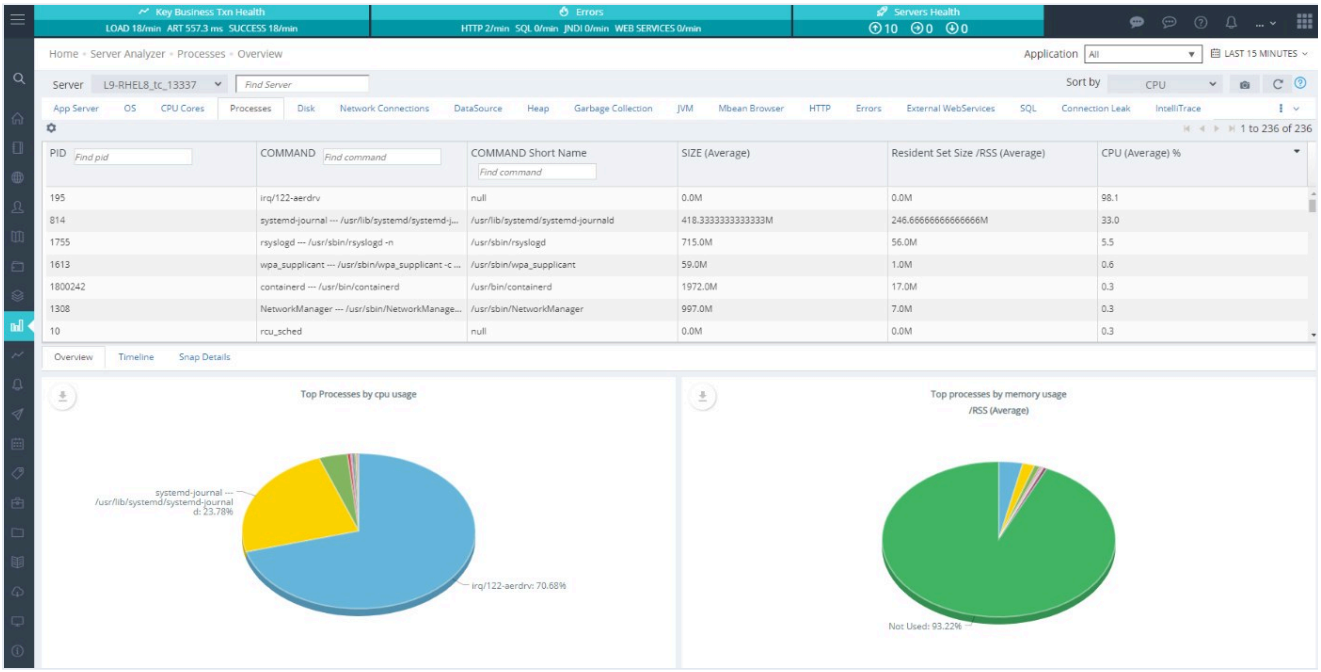
Here, you can access the OS performance history for the selected time period.



Process Analyzer

This view displays the performance stats for CPU, memory and running processes organized in somewhat similar manner to the output of **Top** utility on Unix based systems.

snapshots of various processes running at any point of time and what kind of resources they were consuming at that point of time is available here. Applicare by default would take a snapshot of this info every 30 minutes but this can be configured by changing the snaps interval.



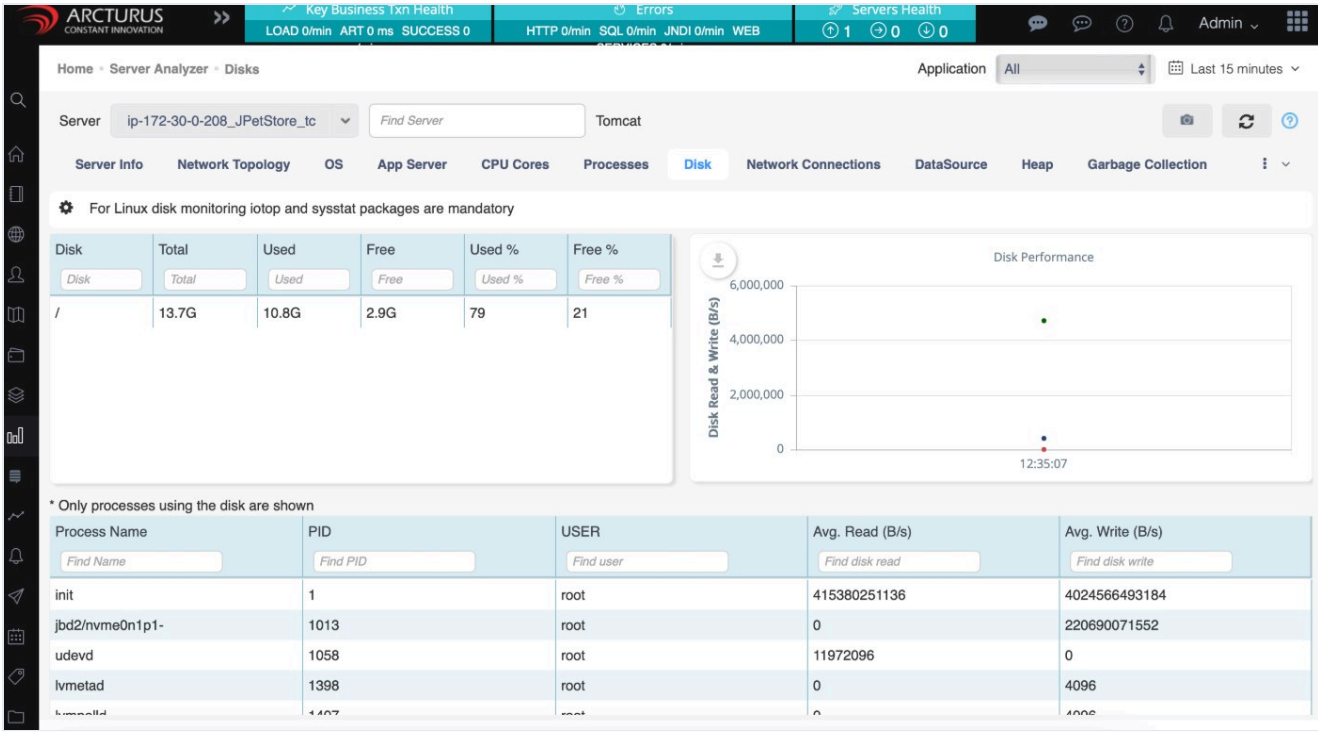
Column Descriptions:

PID :	**The process ID of each task.**
USER :	The user name of the task's owner.

PID :	**The process ID of each task.**
STIME :	Time the process was started.
SIZE :	The size of the task's code plus data plus stack space, in megabytes, is shown here.
RSS :	The total amount of physical memory used by the task, in megabytes, is shown here.
SHARE :	The amount of shared memory used by the task is shown in this column.
STATE :	The state of the task is shown here. The state is either S for sleeping, D for uninterruptible sleep, R for running, Z for zombies, or T for stopped or traced.
TIME :	Total CPU time the task has used since it started in MM:SS format.
CPU :	The task's current usage of CPU time as a percentage.

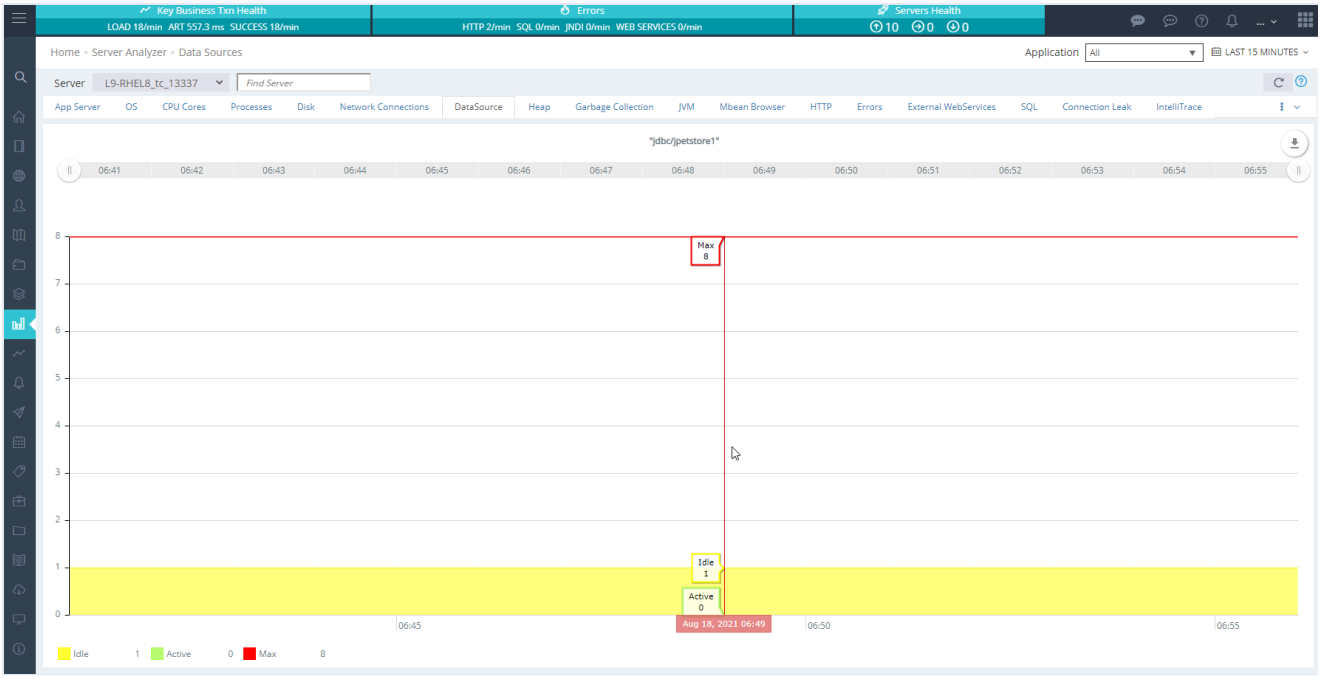
DISK

On the left side, view the used and available disk space. On the right side, monitor disk performance, with only disk-using processes shown.



Data Source Analyzer

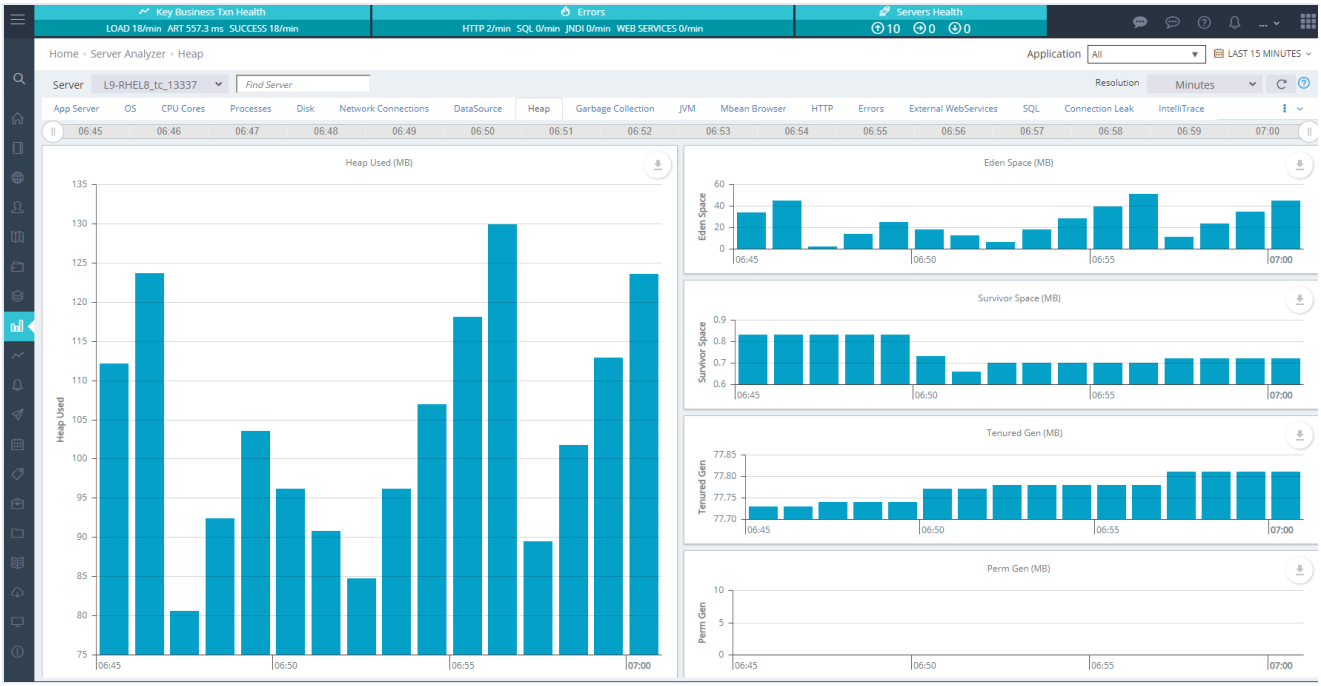
The DataSource Analyzer provides a fast and convenient window into various data source's usage. Datasource utilization is very useful in finding out issues related to connections running out and tuning of connection pools.



Heap Analyzer

The Heap Analyzer provides a convenient window into the history of your server's garbage collection and memory allocation behavior. You can zoom in and out, choose the range of days from the drop down, and select which server to display.

You can easily spot slow memory leaks with the Heap Analyzer. Zoom out to show the data for as long as the server has been up, and you can see how each generation is growing over time. If after major garbage collections the heap doesn't settle to prior levels, but instead keeps increasing, you may have a memory leak.



Heap Dump

You can generate a heap dump of the selected server by clicking on Heap Dump button. Please note this will make the server unresponsive until the heap dump is written to a file on the server. Generating heap dumps on demand is availble on

- Sun JVM 6 and above. - Oracle JRockit R28 and above.

Also you can enable automatic heap dumps on supported JVMs when available free memory reaches a certain threshold by setting 2 startup arguments.

e.g. Using startup arguments *-Dapplicare.dumpheap.perc=10 -Dapplicare.dumpheap.maxcount=2* will generate heap dumps when the server's available free heap size reaches 10% of the max heap size and will create maximum of 2 heap dumps during the lifetime of the JVM.

You can use the generated heap dumps to detect memory leaks and analyze the objects and their relationships by loading the heap dump in Arcturus Memory Analyzer.

Garbage Collection Analyzer

The GC Analyzer provides a fast and convenient window into your server's garbage collector behavior. This gives you the ability to quickly spot garbage collection issues and tweak GC parameters to improve performance and reduce latency of your applications.

GC Analyzer is available on Oracle Sun JVMs, JRockit and IBM JVMs. It requires verbose GC enabled on the JVM.

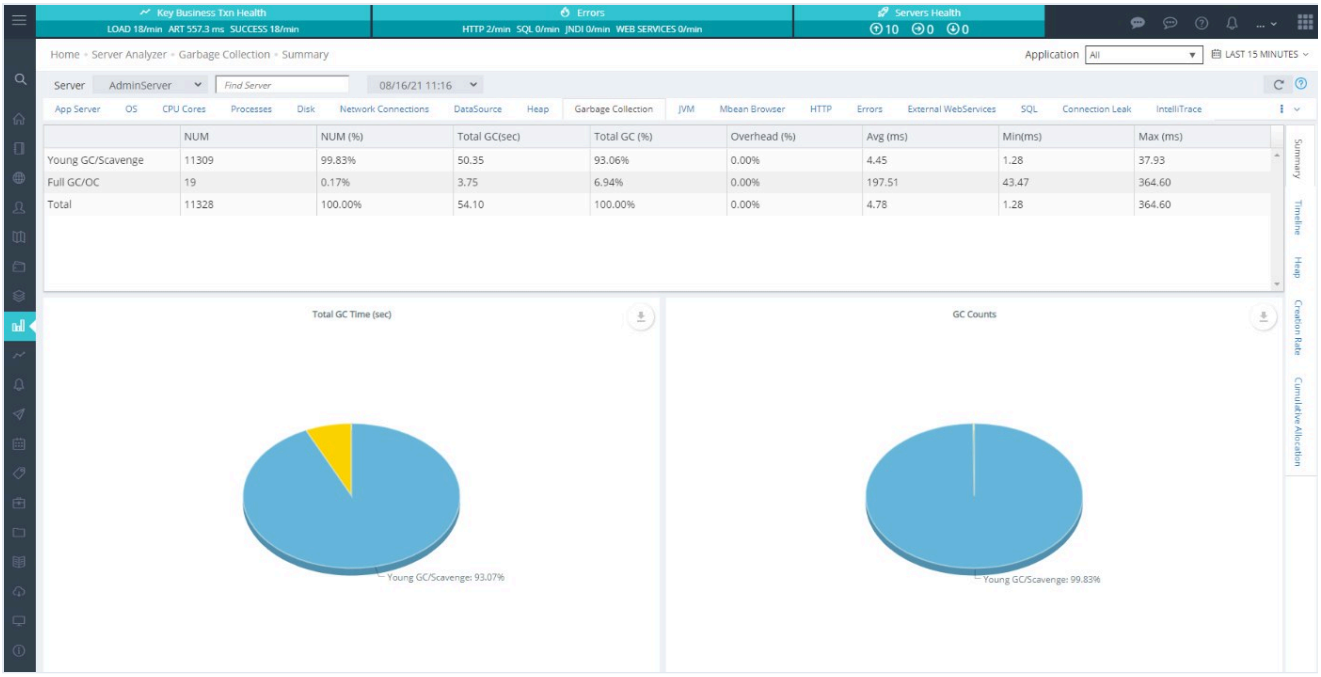
e.g. following arguments are automatically added by Applicare to enable verbose GC on Oracle Sun JVM.

-verbosegc -Xloggc:%SERVER_NAME%.gc

Also the use of **-XX:+PrintTenuringDistribution** flag is not supported at the moment and the use of it will prevent Applicare from reading the GC log file.

GC Summary

The summary tab gives you an overview of Full and Young GC cycles that occured over the lifetime of JVM. An important value you should take note of is the Overhead (%) in the table.

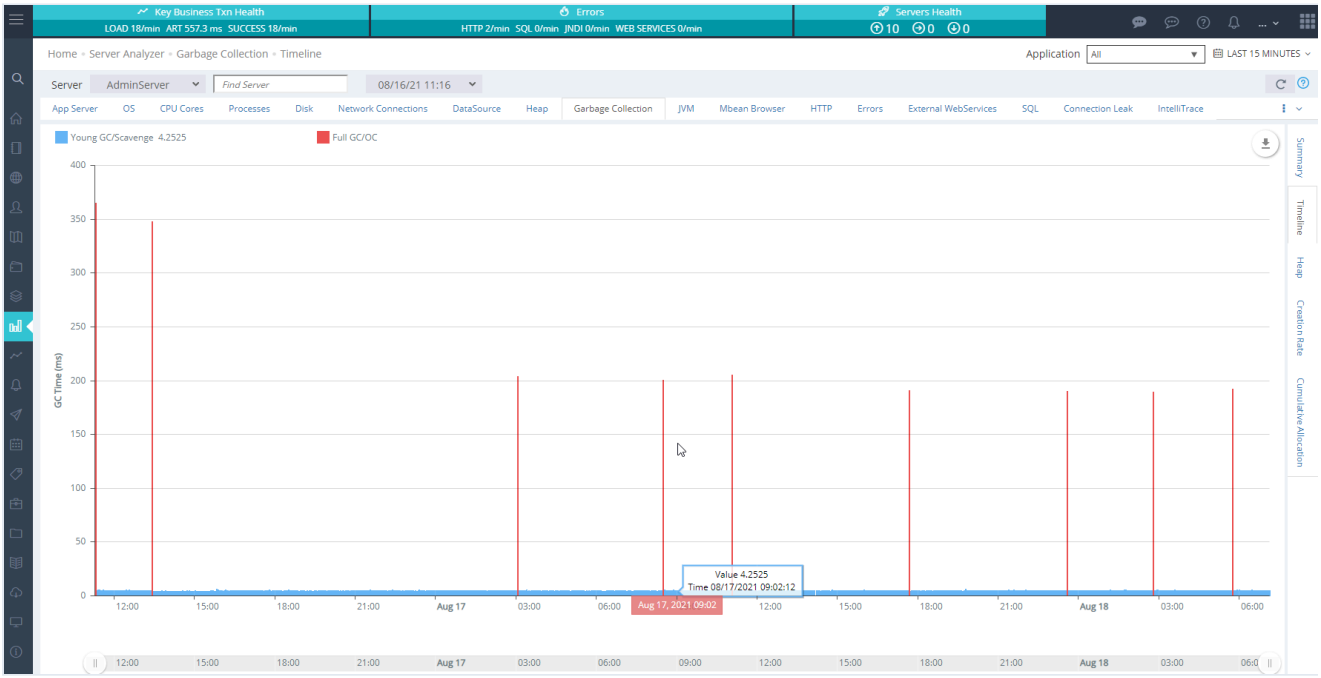


Detailed Charts

The GC Analyzer feature charts that visually represent various aspects of Garbage Collection cycles.

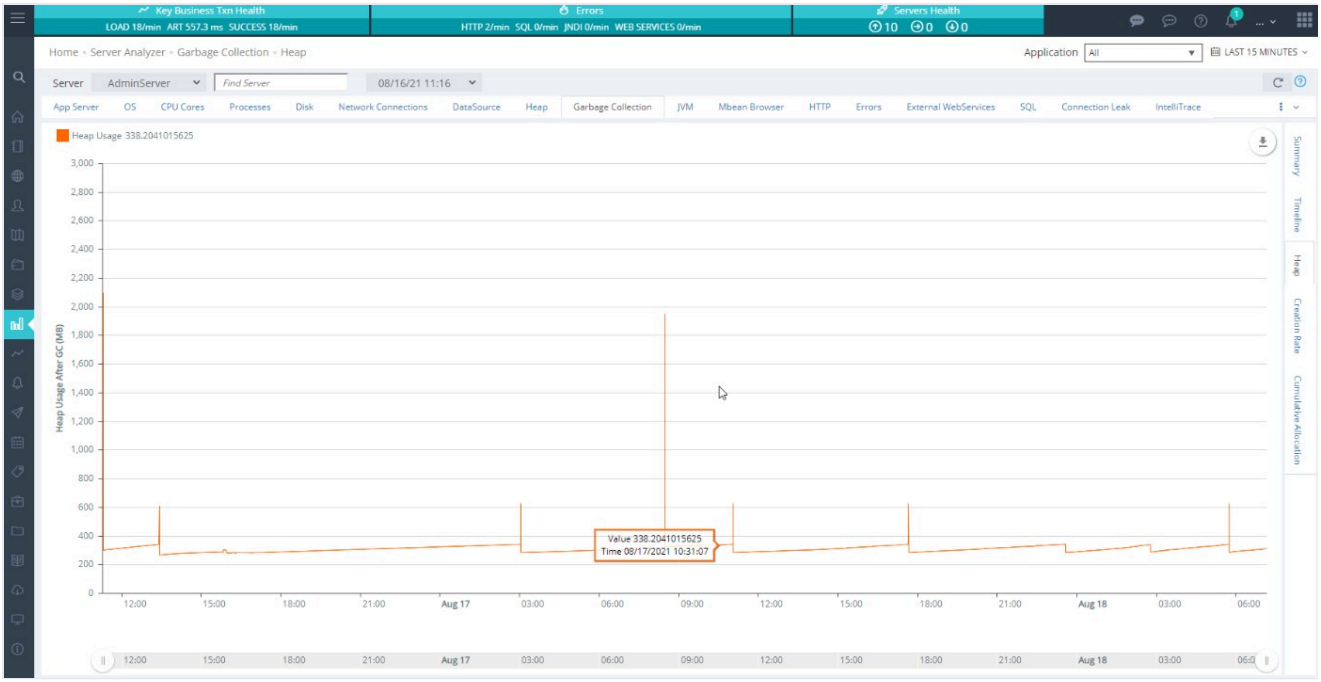
GC Timeline

Garbage Collection shows each GC cycle over time and provides a clear picture of the type of GC cycle and time taken by each.



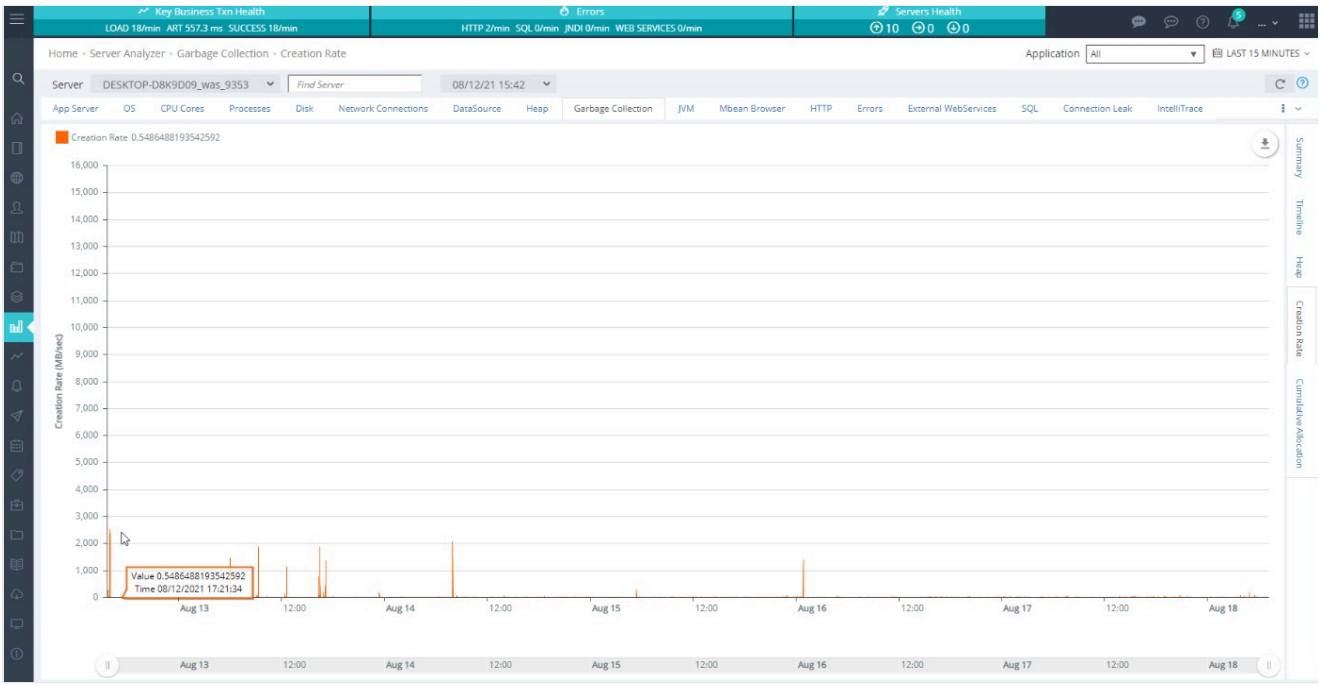
Heap

This chart plots the Heap size before and at the end of each GC cycle over time.



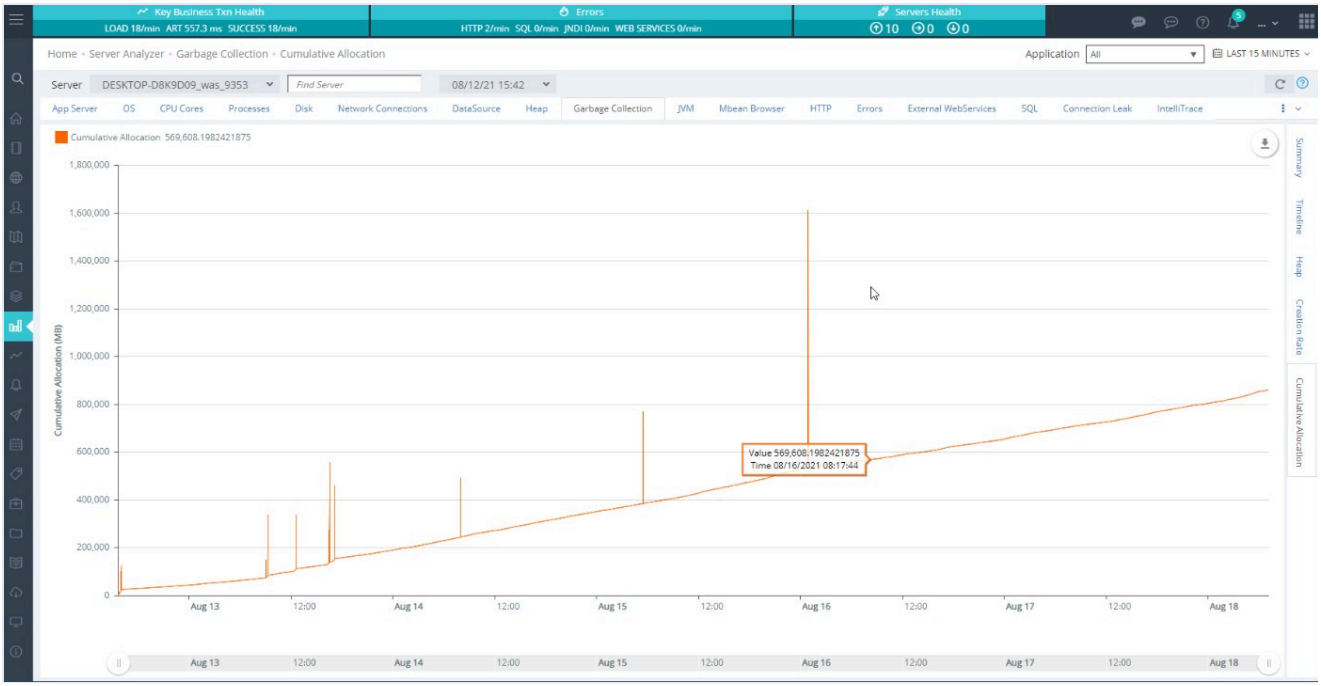
Creation Rate

This analyzer shows the total size of objects created between each GC cycle.



Cumulative Allocation

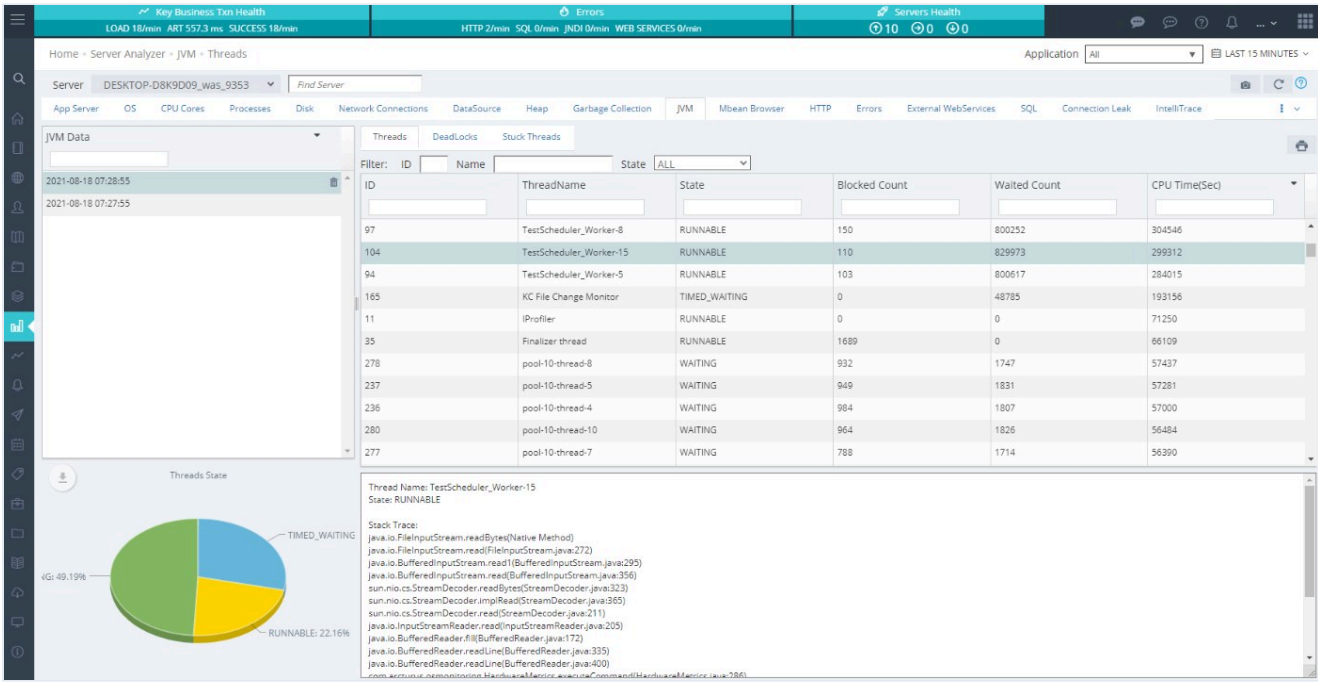
This analyzer shows total size of objects created over time.



JVM Analyzer

The JVM Analyzer provides a very detailed view into a server's thread behavior, detects Deadlocks, detects stuck threads, and provides an interface for viewing stack traces.

This view allows you to save a thread dump of a running server and analyze the saved dumps at a later time. The saved snap shots contain all thread information displayed on the real time view and any deadlock or stuck threads detected at the time of creating the thread dump.



Thread Analysis

Deadlock Detection

A ***Deadlock*** is a situation where two or more threads are blocked forever, waiting on each other to finish. You can find a deadlock by taking a thread dump to display the current threads and look for the BLOCKED state, or you can use the **JVM Analyzer** and click on the **Deadlocks tab.** This tab shows all available information about the deadlock situation. When a deadlock occurs, the tab will turn red to alert you that a deadlock exists. Then you can capture the data and send it to development so they can analyze the code.

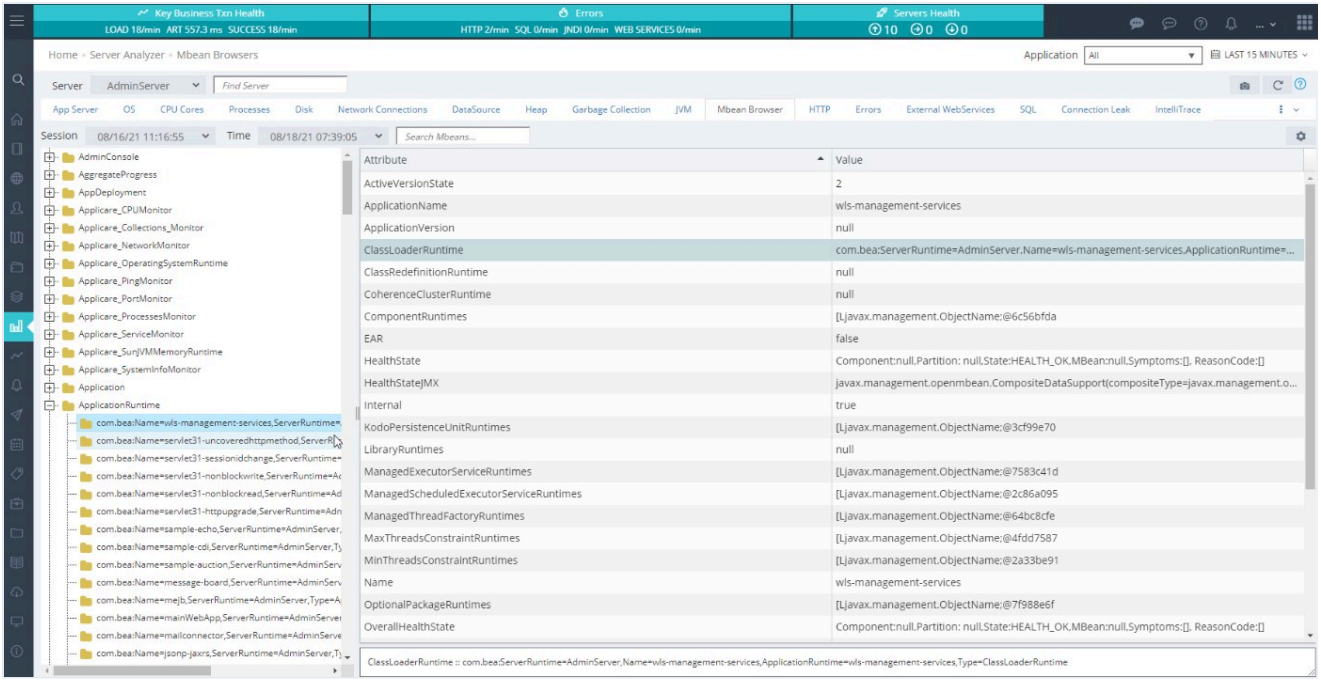
good course of action to follow when you detect a deadlock is to assign a portion of your team to concentrate on finding out how to reproduce the deadlock, and the other portion to look at the stack traces to figure out why the code is deadlocking

Stuck Thread Detection

A ***stuck thread*** is a thread that is blocked and can't return to the thread pool within a defined period of time. When an application thread is blocked, it can't quickly finish its job and be reused. In most situations, the cause of stuck threads is also the cause of poor system performance. This is because the stuck thread interferes with regular task execution. The “Stuck Threads” tab will turn red and display the diagnostic information to help you identify the root cause of the stuck thread. Stuck Threads are often caused by the code opening a network connection without specifying a network connect or read timeout. When a timeout isn’t configured for each method call that involves networking, it will potentially create a block because it will wait infinitely for a response. The fallback timeout in this case comes from the OS which is by default set very long (e.g. 5-10 minutes). Especially in the case of SOA and Web Services where there are many network calls, timeouts need to be defined properly.

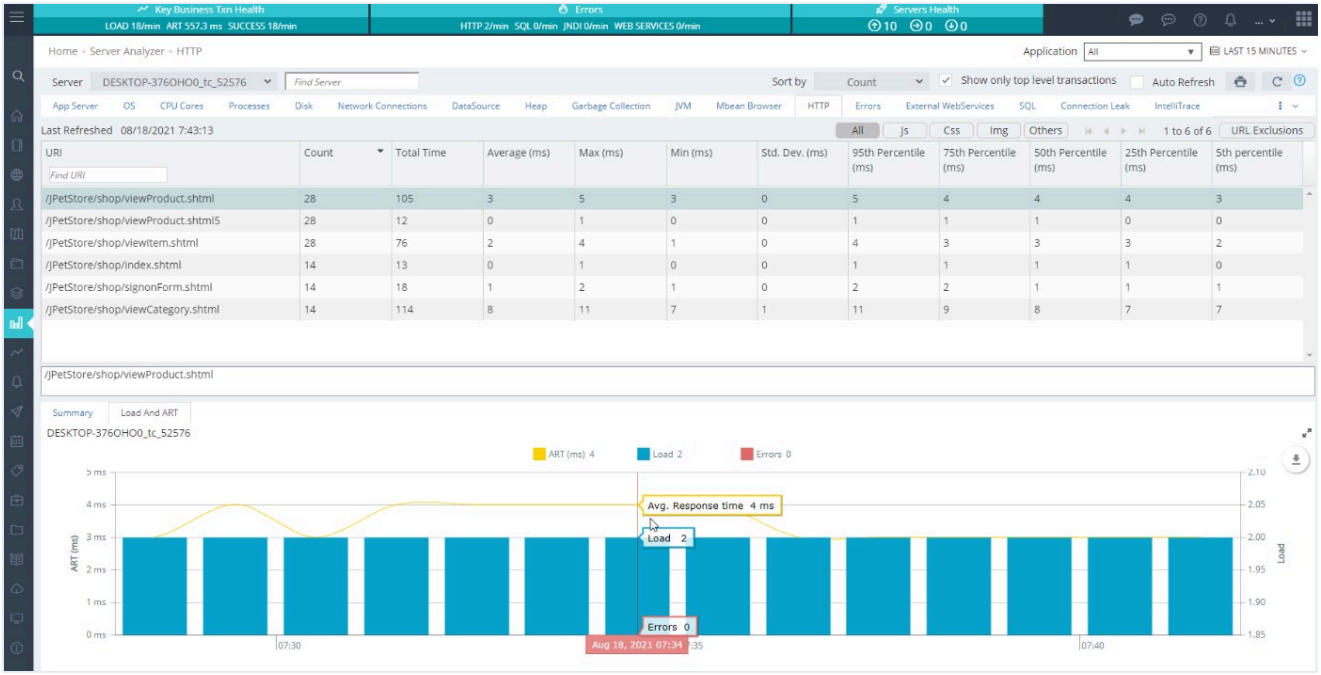
MBean Browser

The MBean Browser enables you to explore MBean properties and their values.



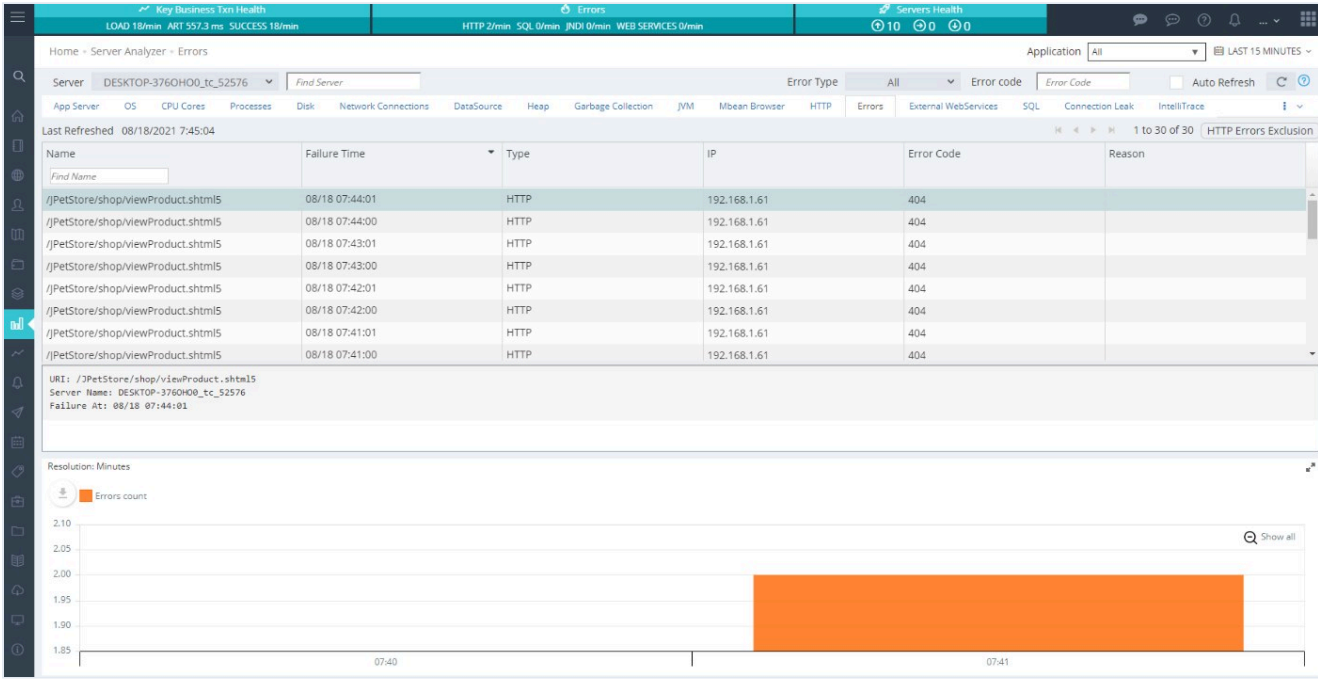
HTTP

The HTTP Analyzer offers performance statistics for HTTP requests on the selected server. The real-time view displays HTTP stats since the JVM started, while the historical view provides data for the selected time period. The real-time view also indicates whether HTTP Profiling is enabled and, if so, how long it has been active.



HTTP Errors

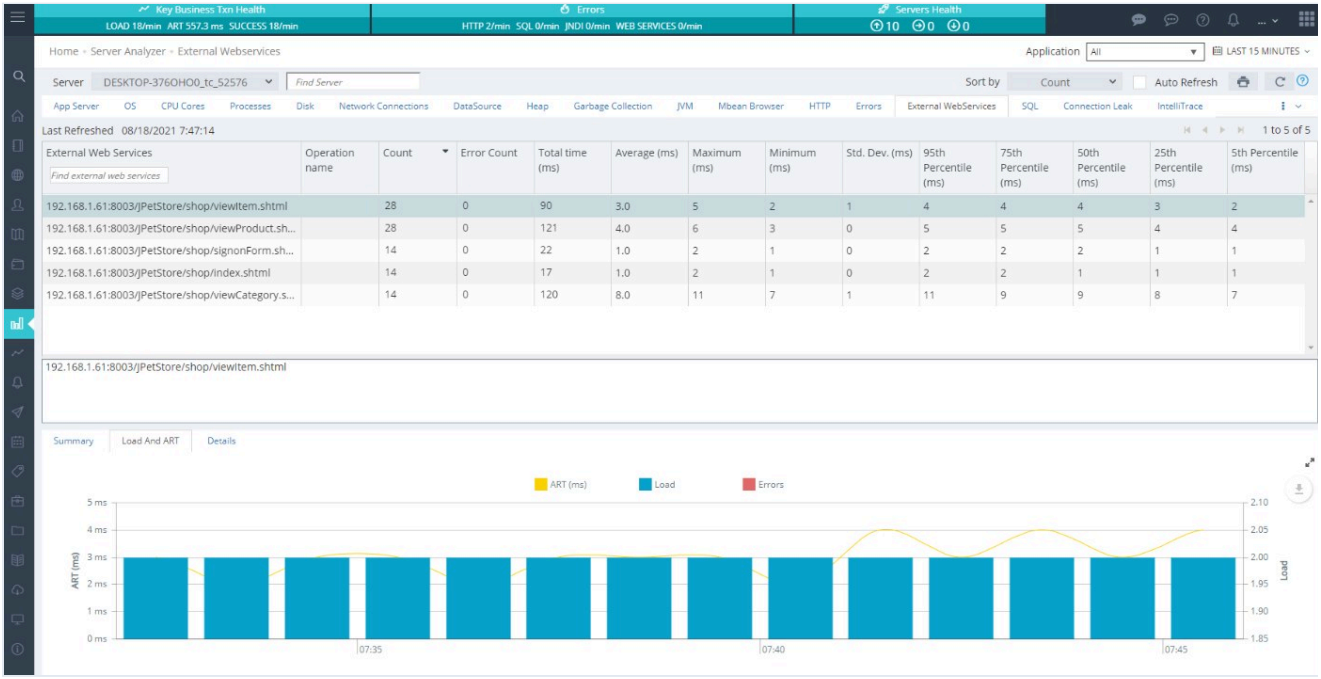
HTTP Errors shows URI, failure time, http error code, exception and IP address where the call came from.



External WebServices

Server Analyzer External WebServices shows all the web services that are called from the monitored JVM/App Server.

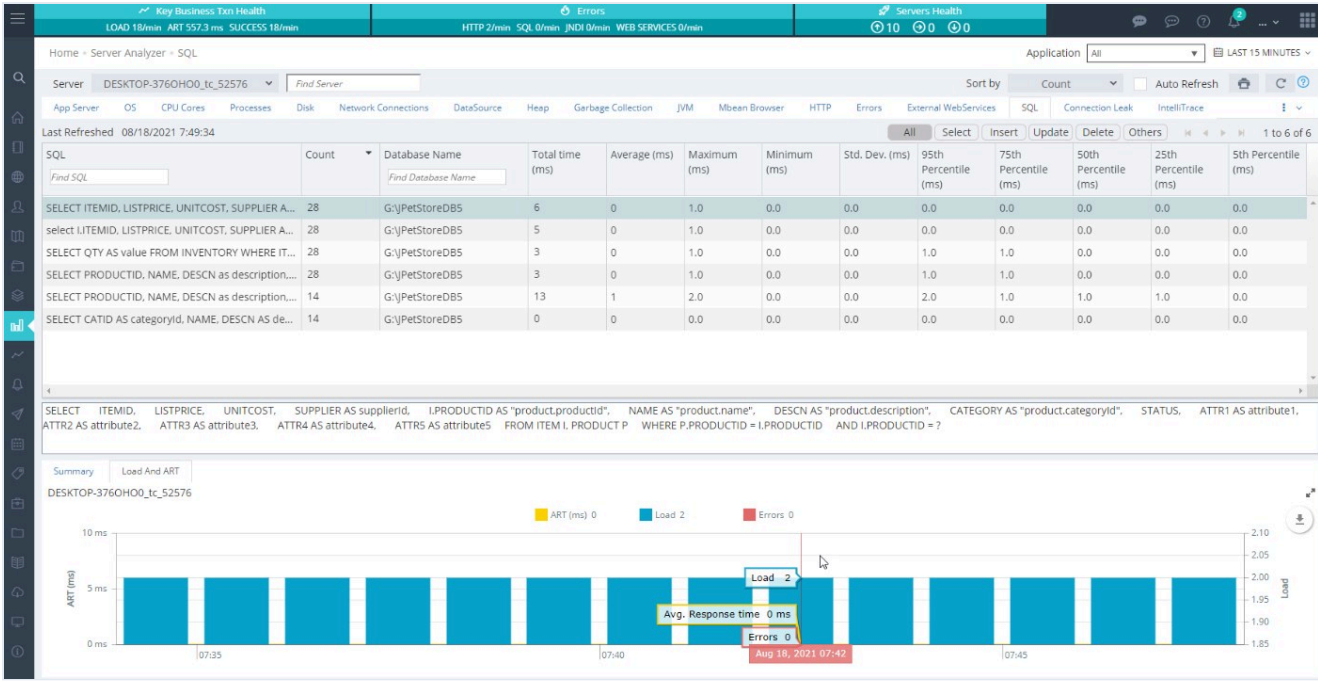
External WebServices monitors all calls to externally hosted webservices and provides performance stats, server ip and port where the calls are going for easily tracking the issues that come with external webservices response time going up.



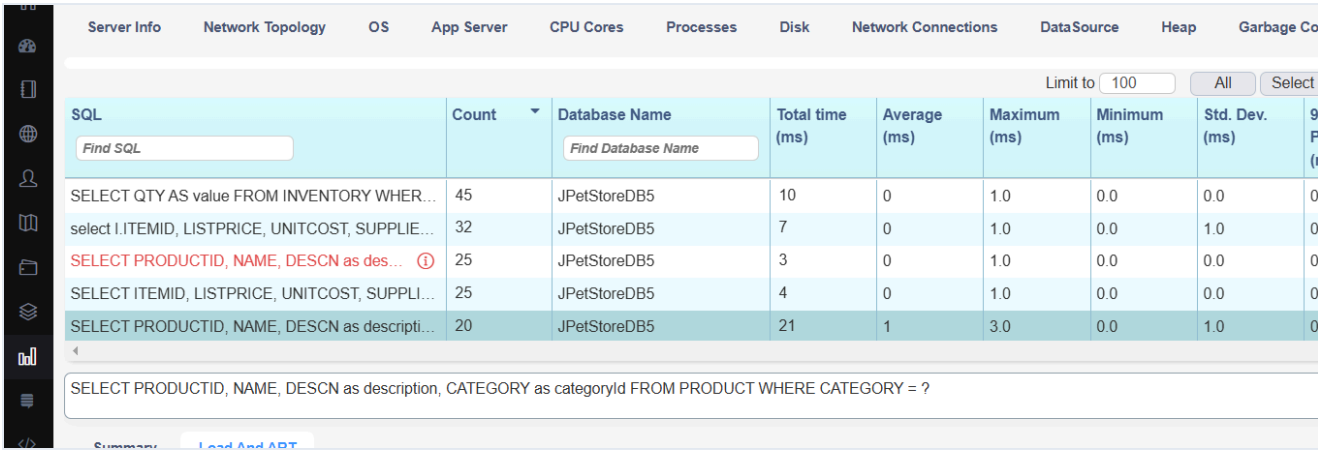
SQL

IntelliTrace Analyzer contains separate views to display data collected by IntelliTrace aspects. SQL Statement Profiling provides real-time performance statistics of database requests. You can view what statements are taking the most time and quickly diagnose performance impacting SQL/s.

Maximum (ms) and Minimum (ms) data will not be available for database monitoring agents.

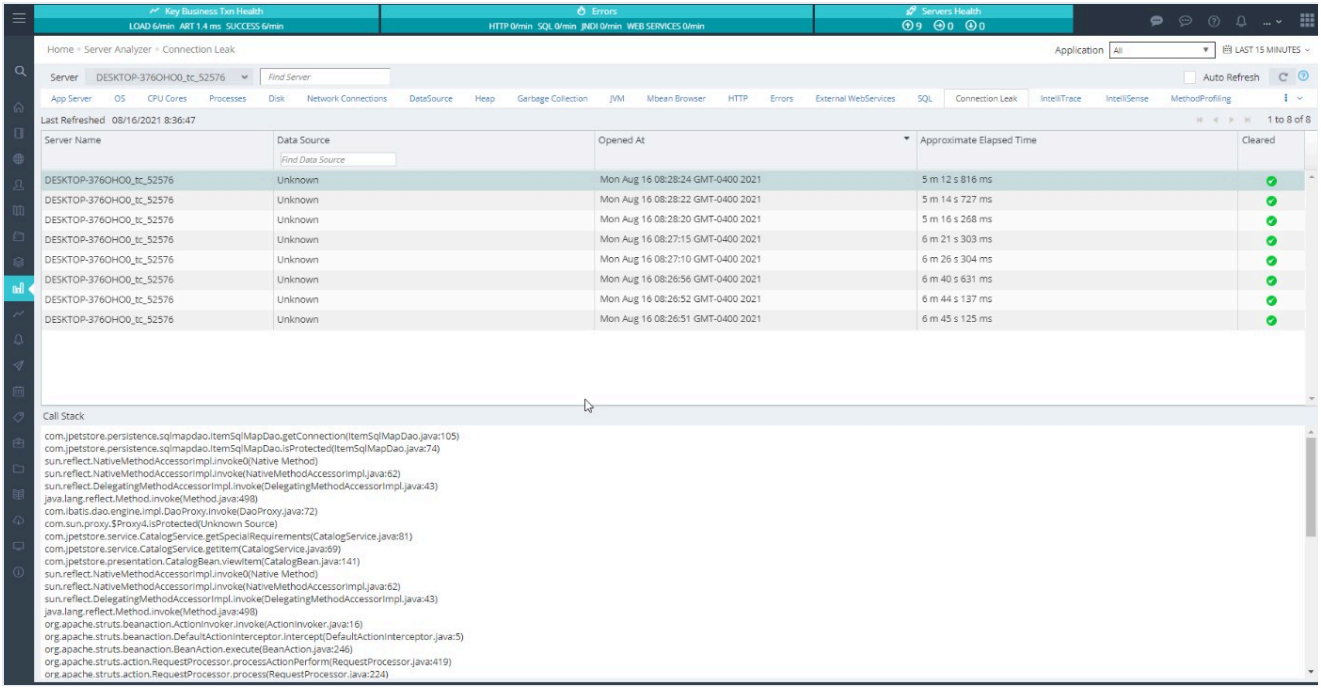


This screen highlights SQL queries vulnerable to injection attacks in red color.



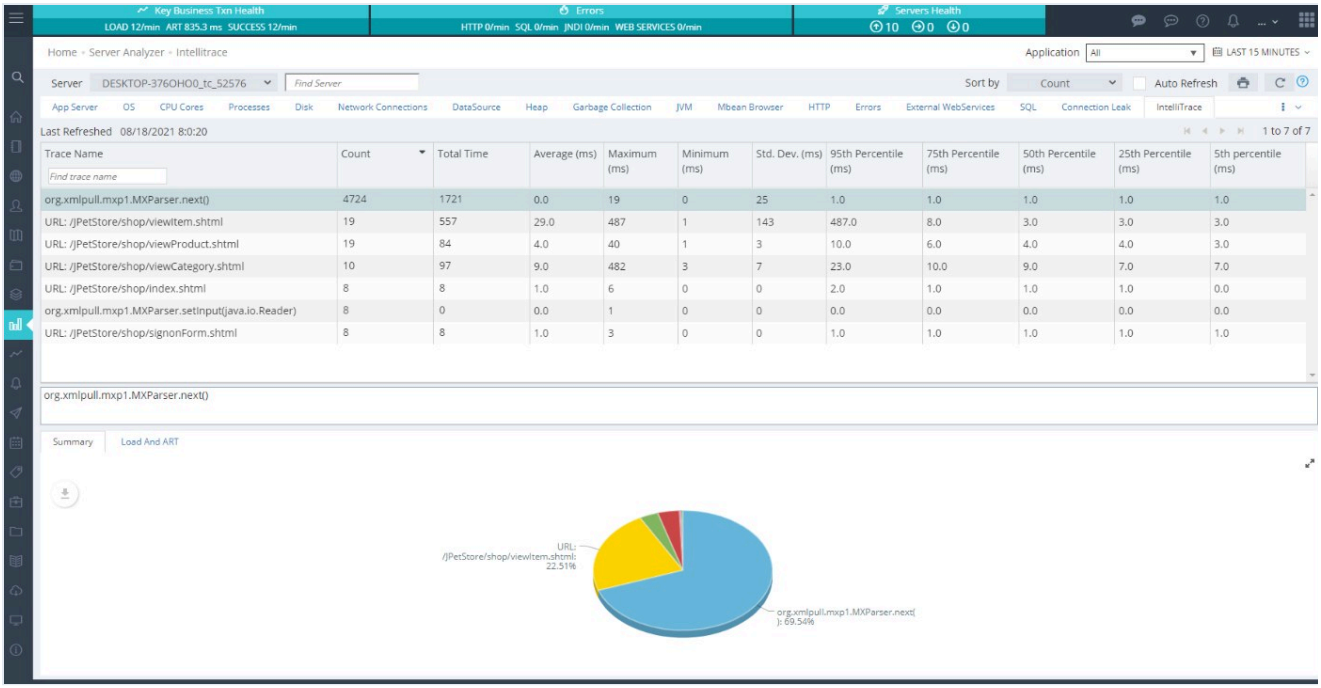
Connection Leak

JDBC Connection Profiling feature allows you to detect leaked JDBC connections, connections held open for long periods of time and code responsible for connection leaks. You can view the connections that are currently in open state and clicking on an open connection will display the call stack which requested the connection. Statistics displayed are for the selected server in the UI.



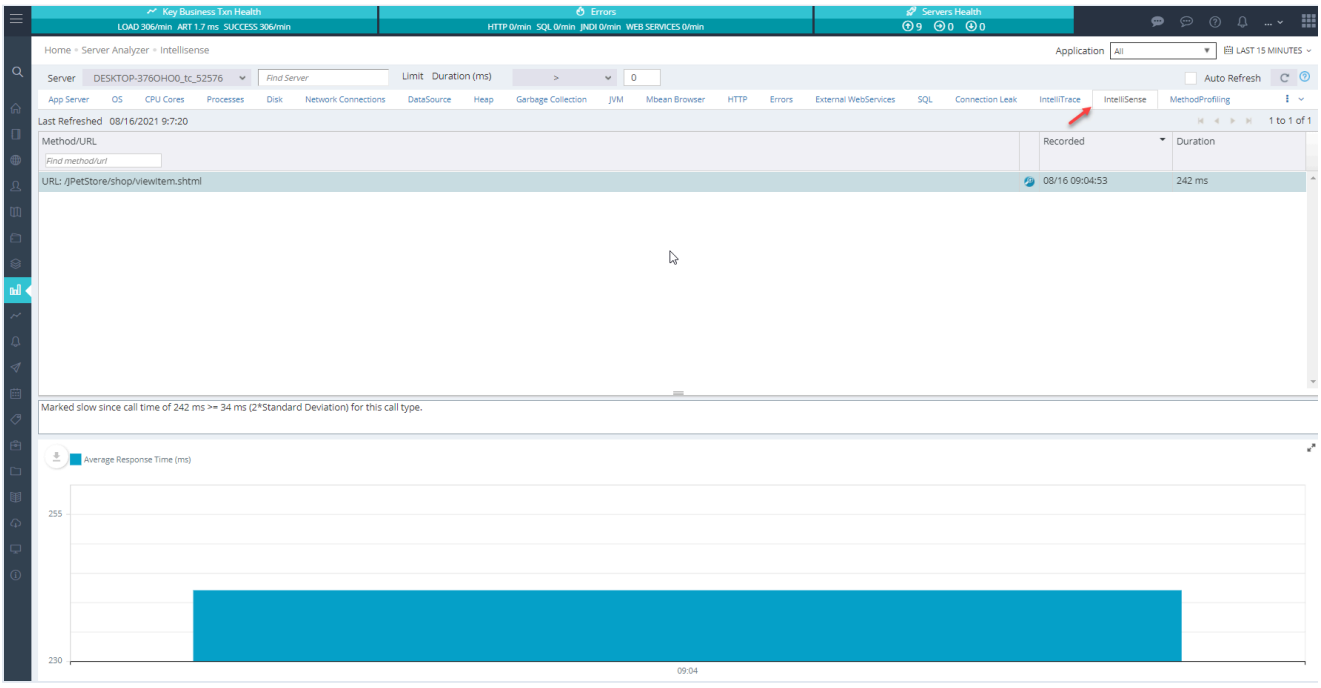
IntelliTrace

IntelliTrace decides what methods to track dynamically and decides the depth of the call stack based on invocation counts and time spent on each individual method to minimize the overhead of tracing. IntelliTrace does what the real experts do with the profiling data - illuminate data that has little impact but it does it in real time. This way no time/resources are spent on gathering data that doesn't have meaningful impact on performance.



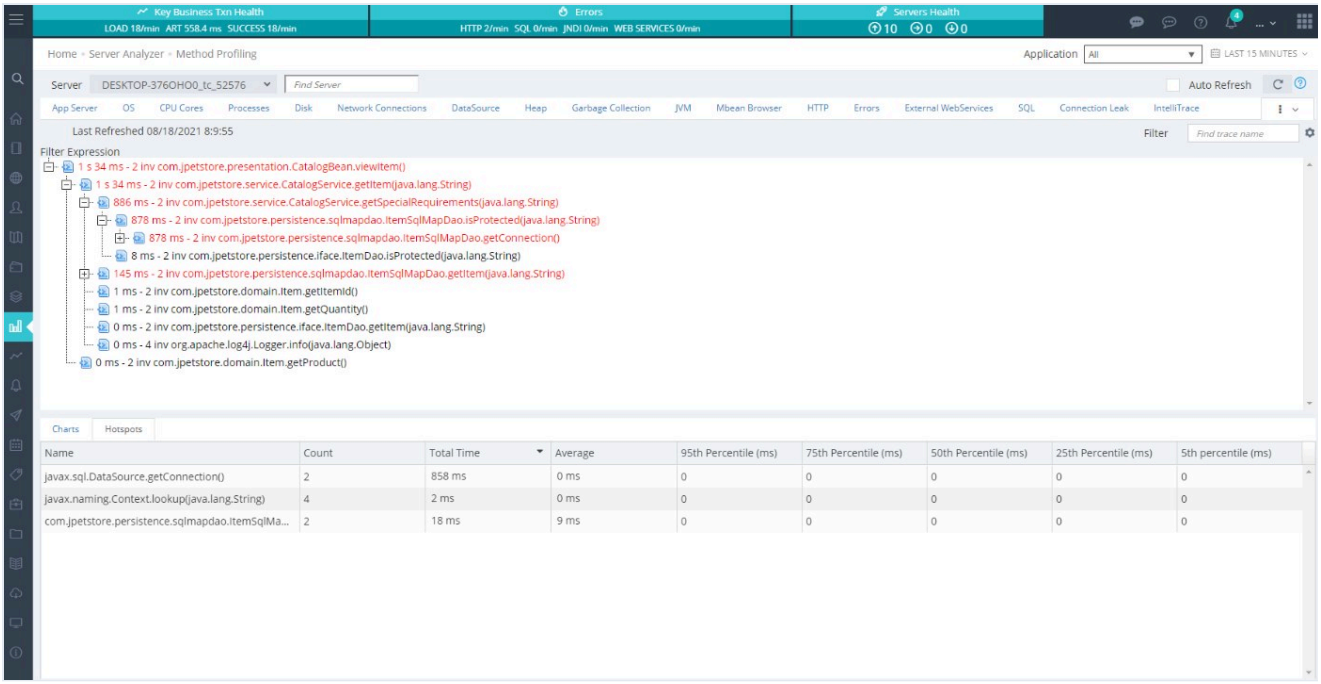
IntelliSense

IntelliSense is a feature that works when IntelliTrace is enabled to automatically detect transaction executions that deviate considerably from the normal execution times and saves a complete call graph of the execution for later analysis.



Method profiling

Method Profiling feature allows you to easily view the performance statistics of real-time method executions on-the-fly. This Profiling Aspect needs to be configured and deployed before you can view performance data. See Method Profiling Configuration for more information on configuration.



JNDI

JNDI calls monitoring tracks JNDI performance and helps identify issues related to lookup delays.

JMS

JMS calls monitoring delivers performance statistics for JMS queues and topics, including metrics such as the number of calls, total time, average duration, and various percentiles.

USING APPLICARE › INFRASTRUCTURE

Applicare Docker Monitoring

Applicare 9.2 · Updated September 13, 2024 · docker · **Pending technical review**

The Docker container must have the following packages installed for monitoring.

procps, net-tools, iotop and sysstat

To install packages inside a Docker container

Enter the Docker containers and install the packages one at a time.

If you encounter an **Unable to locate package** error, run the command **apt-get update** for Debian /Ubuntu or **yum update** for CentOS or RHEL. After updating, install the packages individually.

On Debian, Ubuntu, and Linux Mint

apt-get install procps apt-get install net-tools apt-get install iotop apt-get install sysstat

On RedHat, CentOS and Fedora yum install procps yum install net-tools yum install iotop yum install sysstat

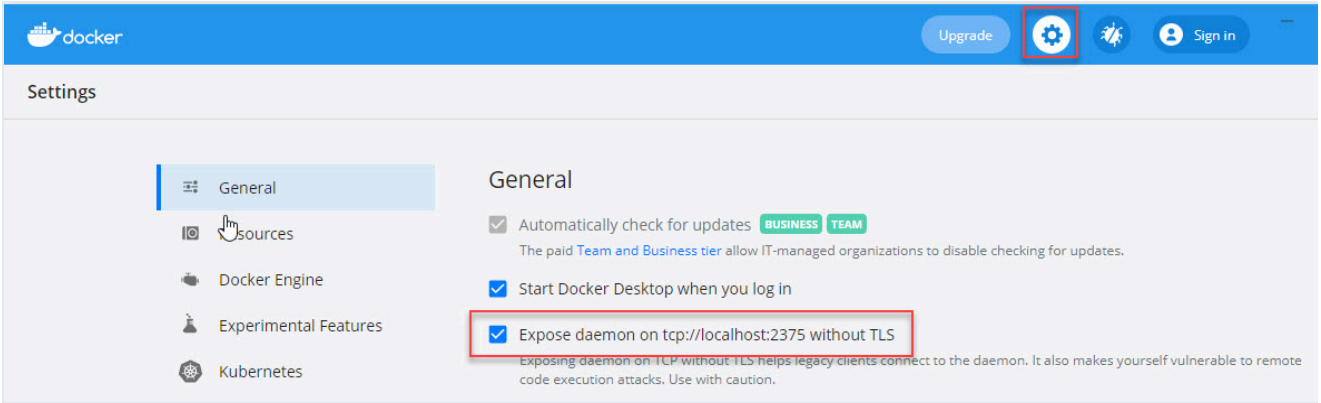
On Fedora 22+ Releases

dnf install procps dnf install net-tools dnf install iotop dnf install sysstat

Windows

For HTTP Monitoring

1. Open the Docker desktop, and go to **settings,** and enable the "**Expose daemon on tcp://localhost:2375 without TLS**" option and restart the docker desktop.



Linux

For HTTP Monitoring

1. Open the "**docker.service**" file and change the "**ExecStart=**" line to the below one

Syntax

```
TEXT
ExecStart=<dockerdpath> –H fd:// –H tcp://<docker_installed_ip>:<docker_running_port>
```

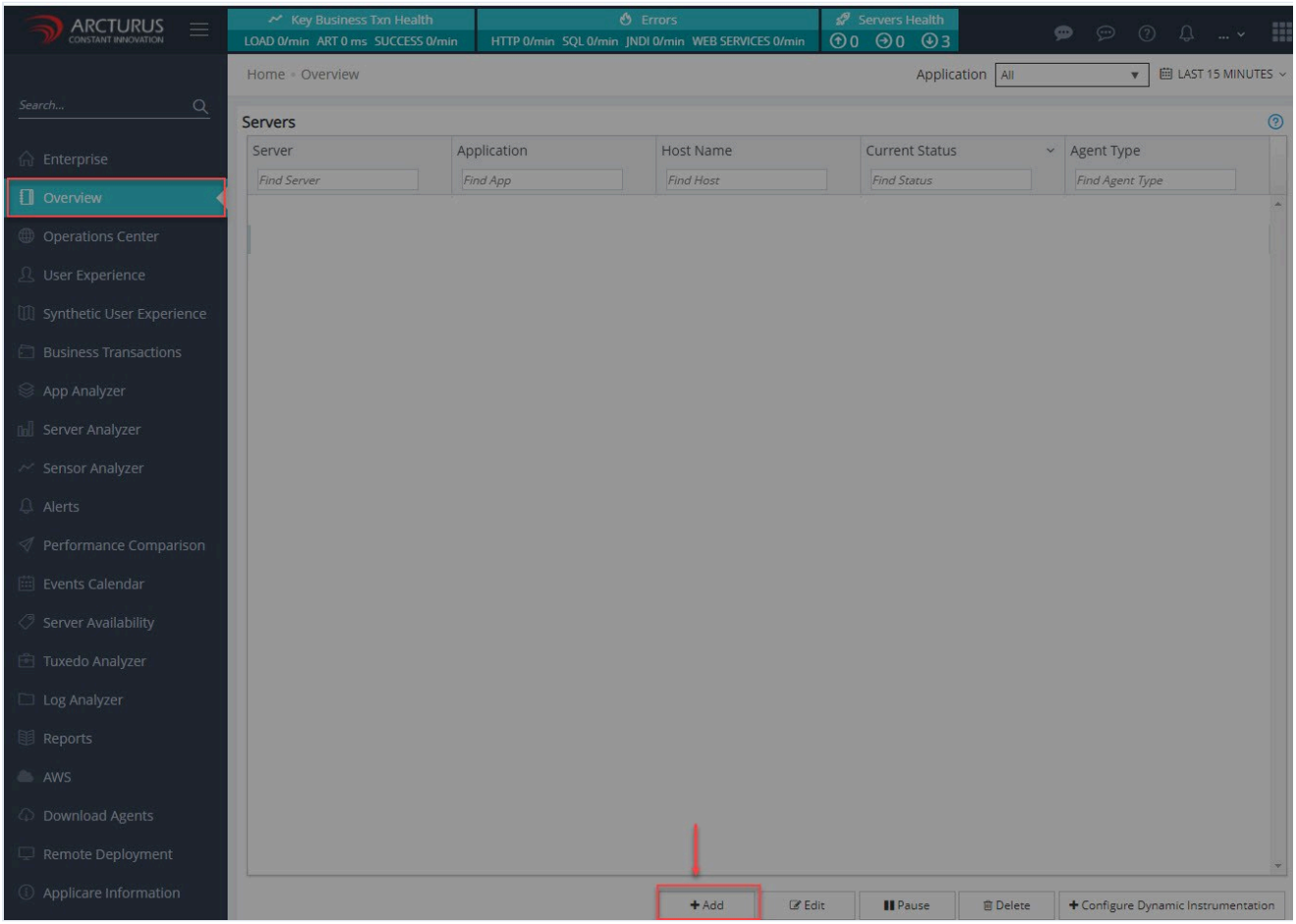
e.g.

ExecStart=/usr/bin/dockerd -H fd:// -H tcp://192.168.0.1:2375

2. Run the command "**systemctl daemon-reload**" and then restart the "**docker.service**".

Applicare Controller

1. Log in to **Applicare Console** and navigate to the **Overview** screen, and click the **Add** button.



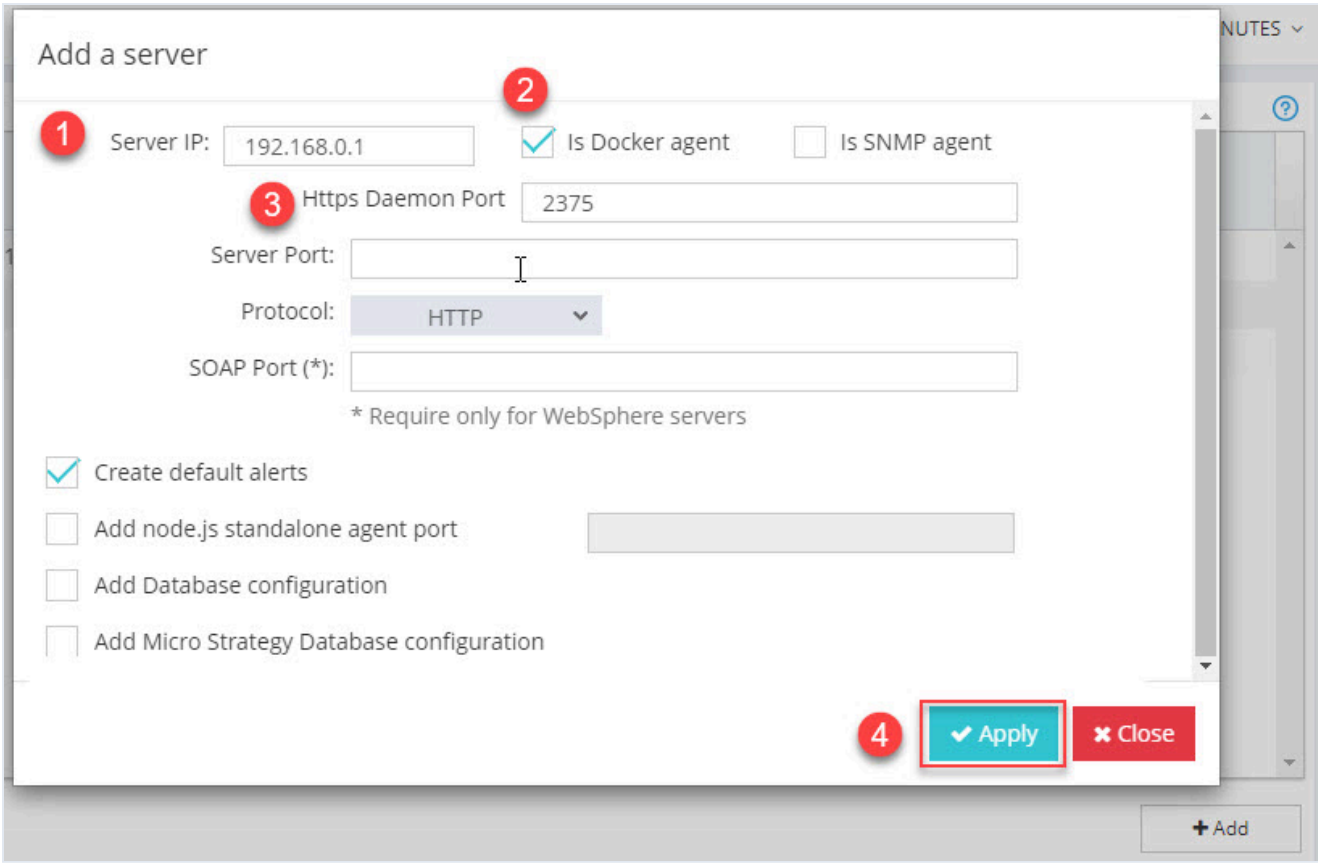
2. Enter the required details and save.

a. Input the **Server IP**

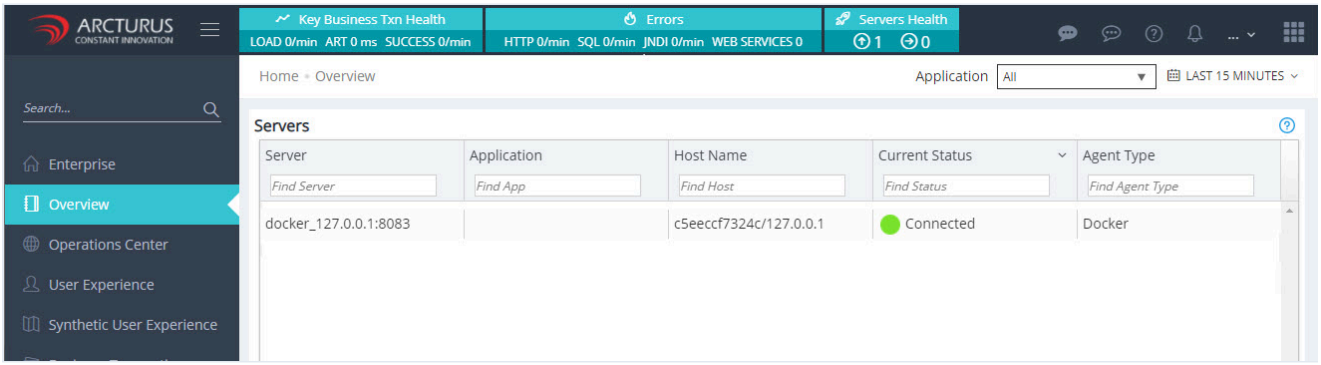
b. Enable the **Is Docker agent** option.

c. Enter the **HTTP Daemon Port**

d. Click the "**Apply**" button.



3. All running containers will be added as agents, any new containers that start will be automatically added as new agent.



4. After adding the containers, navigate to the Server Analyzer screen and check the metrics.

Monitoring with Applicare

USING APPLICARE › INFRASTRUCTURE

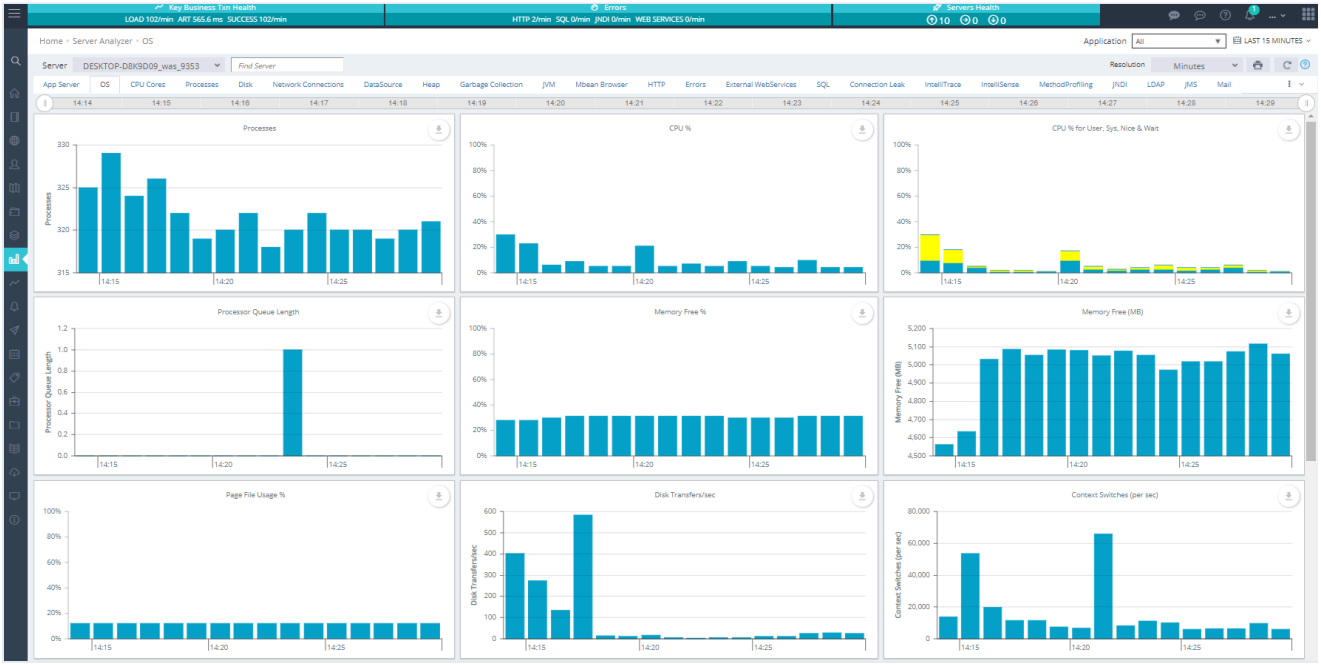
Operating System Analyzer

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

The OS Internals Analyzer offers a quick and convenient view into your physical server's processes, network activity, CPU usage, memory and storage performance.

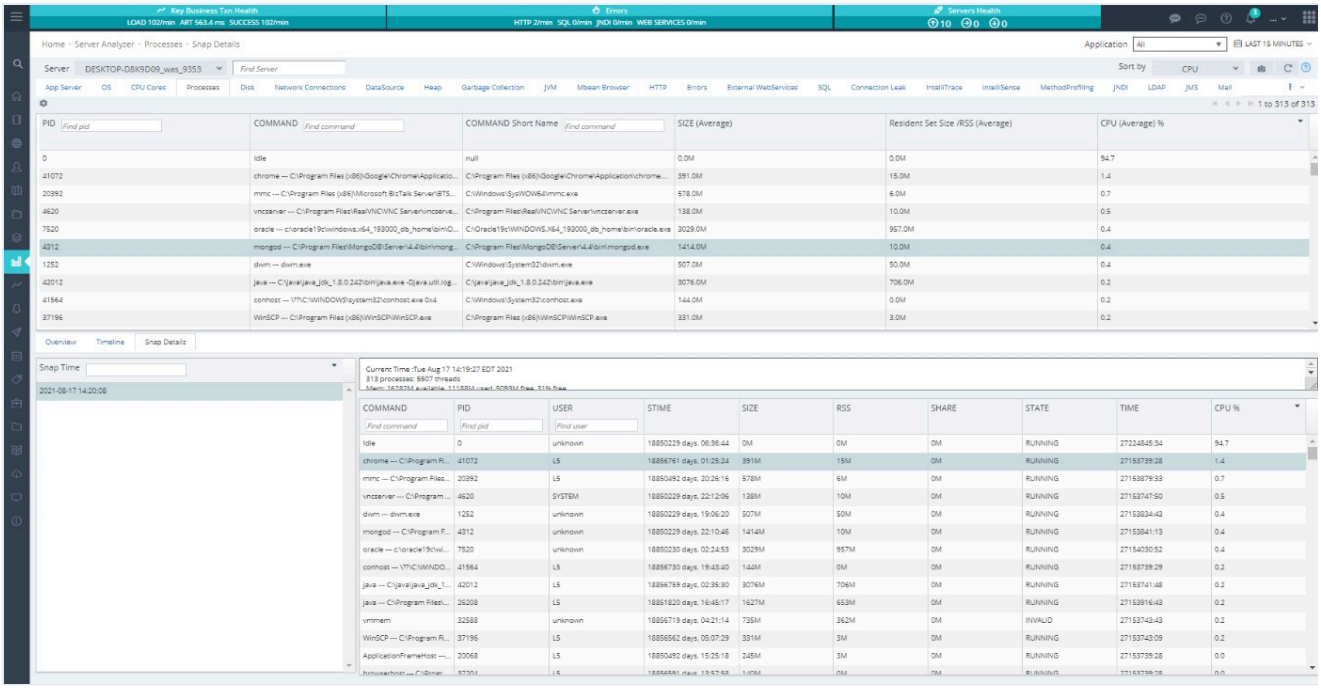
Charts

This tab allows you to monitor overall system performance visually through graphical representations. You can monitor your server's memory usage, system load, CPU utilization, context switching, and network data.



Processes

This view displays performance stats for CPU, memory and running processes, organized similarly to the output of the **Top** utility on Unix-based systems.



Column Descriptions:

PID :	**The process ID assigned to each task.**
USER :	The username of the task's owner.
STIME :	The time when the process was initiated.
SIZE :	The size of the task's code, data and stack space, displayed in megabytes, is shown here.
RSS :	The total amount of physical memory used by the task, measured in megabytes, is shown here.
SHARE :	This column displays the amount of shared memory used by the task.
STATE :	This column indicates the task's state, which can be S for sleeping, D for uninterruptible sleep, R for running, Z for zombies, or T for stopped or traced.
TIME :	The total CPU time utilized by the task since its start, displayed in MM.
CPU :	The task's current CPU time usage, expressed as a percentage.

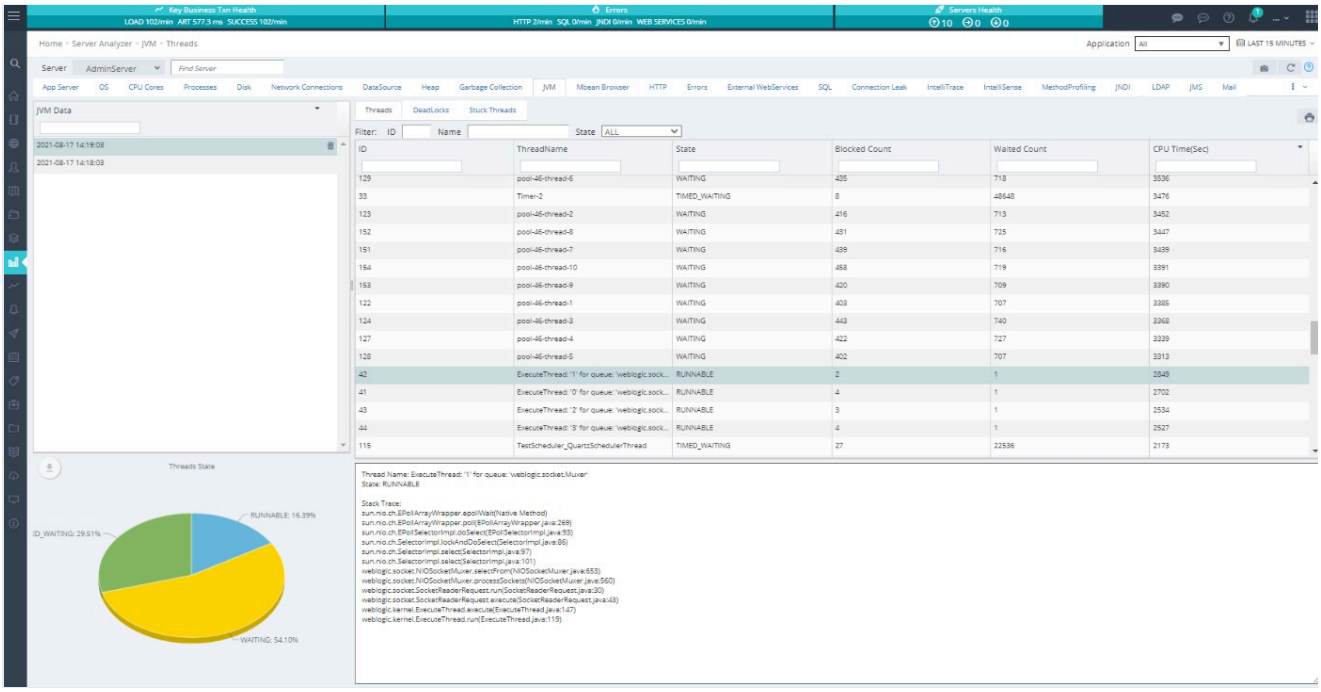
USING APPLICARE › INFRASTRUCTURE

JVM Analyzer

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Thread Dumps

This view enables you to capture a thread dump of a live server, which you can analyze the saved later. The saved snapshots include all the thread details visible in the real-time view and as well as any deadlock or stuck threads detected when the dump was created.



[Monitoring with Applicare]

USING APPLICARE › INFRASTRUCTURE

GC Analyzer

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

The GC Analyzer offers a quick and easy way to examine your server's garbage collector performance. This tool enables you to swiftly identify garbage collection issues and adjust GC parameters, enhancing performance and reducing application latency.

The GC Analyzer is available exclusively on Sun JVMs and requires that verbose GC logging be enabled on the JVM.

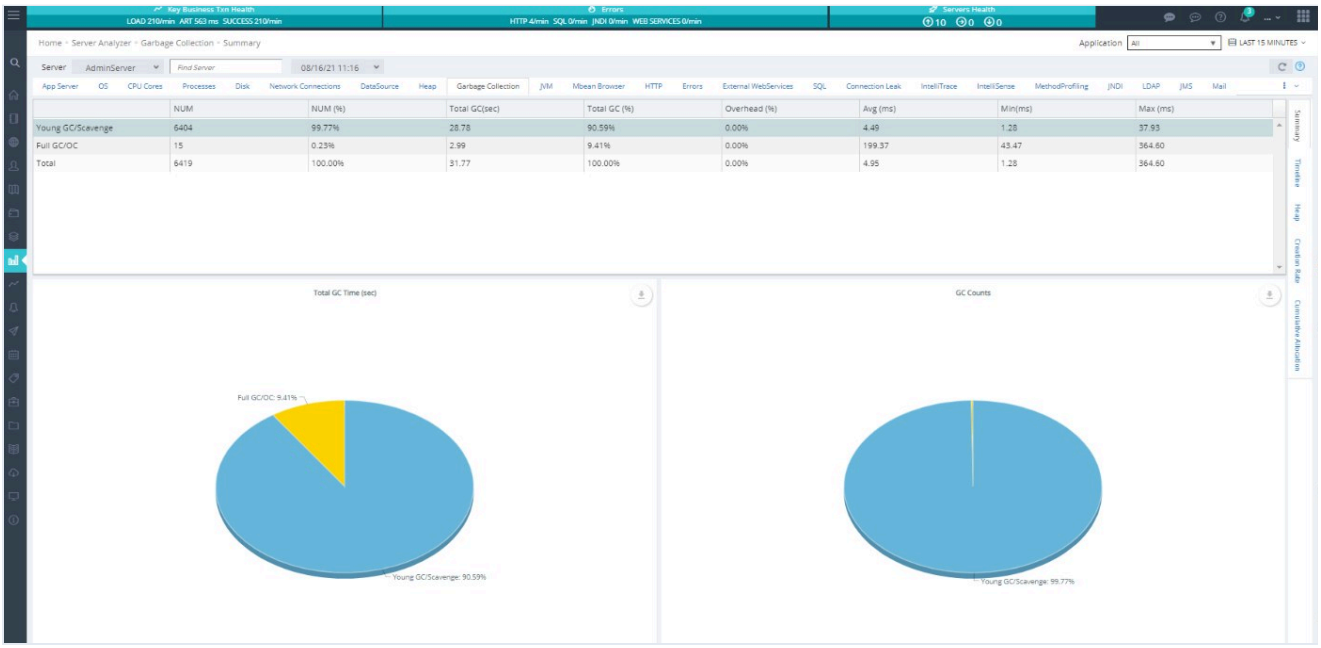
Applicare automatically adds the following argument to enable verbose GC:

-verbosegc -Xloggc:%SERVER_NAME%.gc

Currently, the use of the **-XX:+PrintTenuringDistribution** flag is not supported and will prevent Applicare from accessing the GC log file.

Summary View

The summary tab provides an overview of Full and Young GC cycles that have occurred throughout the JVM's lifetime. A key value to watch is the Overhead (%) displayed in the table.



Detailed Charts

The GC Analyzer features various charts that visually represent different aspects of Garbage collection cycles.

Timeline Tab - Provides a visual representation of each GC cycle over time, offering a clear view of the type of GC cycle and then duration it took.

Heap Tab - Displays a graph of heap size at the end of each GC cycle, tracking changes over time.

Object Creation Rate Tab - This graph shows the total size of objects created between each GC cycle.

Cumulative Allocation Tab - Displays the total cumulative size of objects created over time.

Monitoring with Applicare

USING APPLICARE › INFRASTRUCTURE

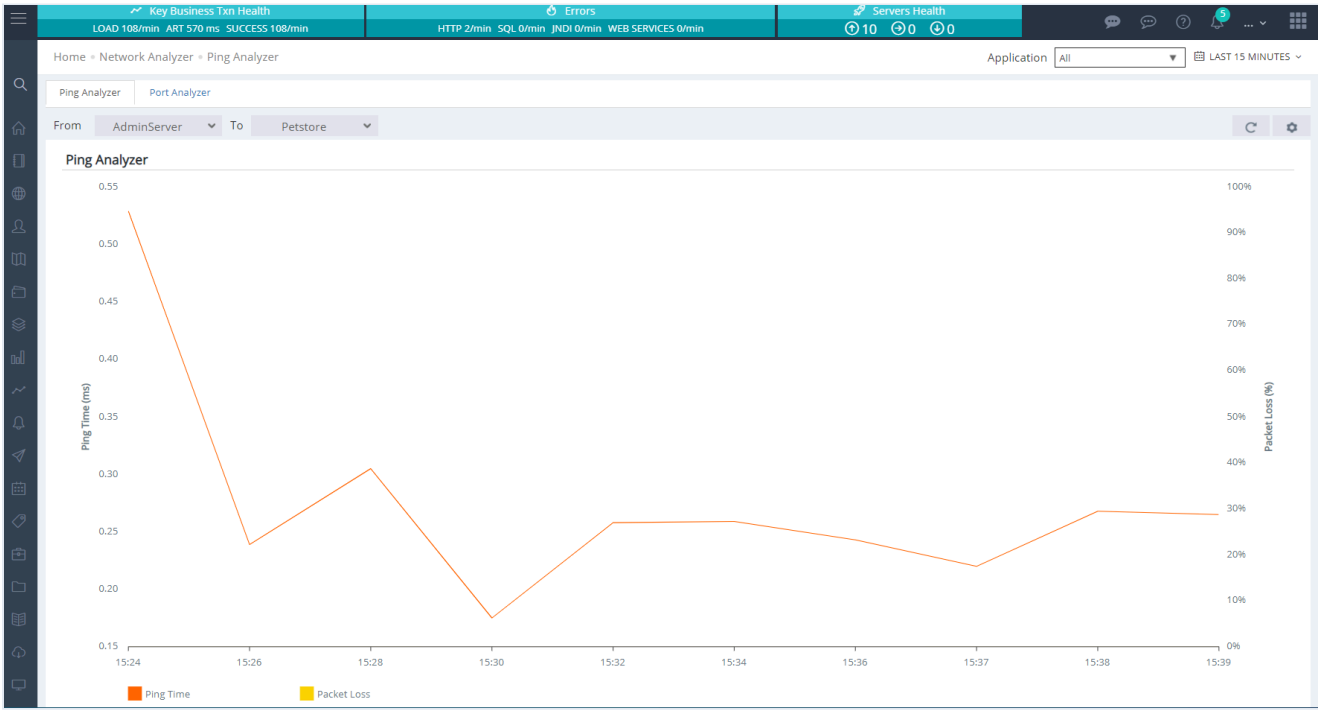
Network Analyzer

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

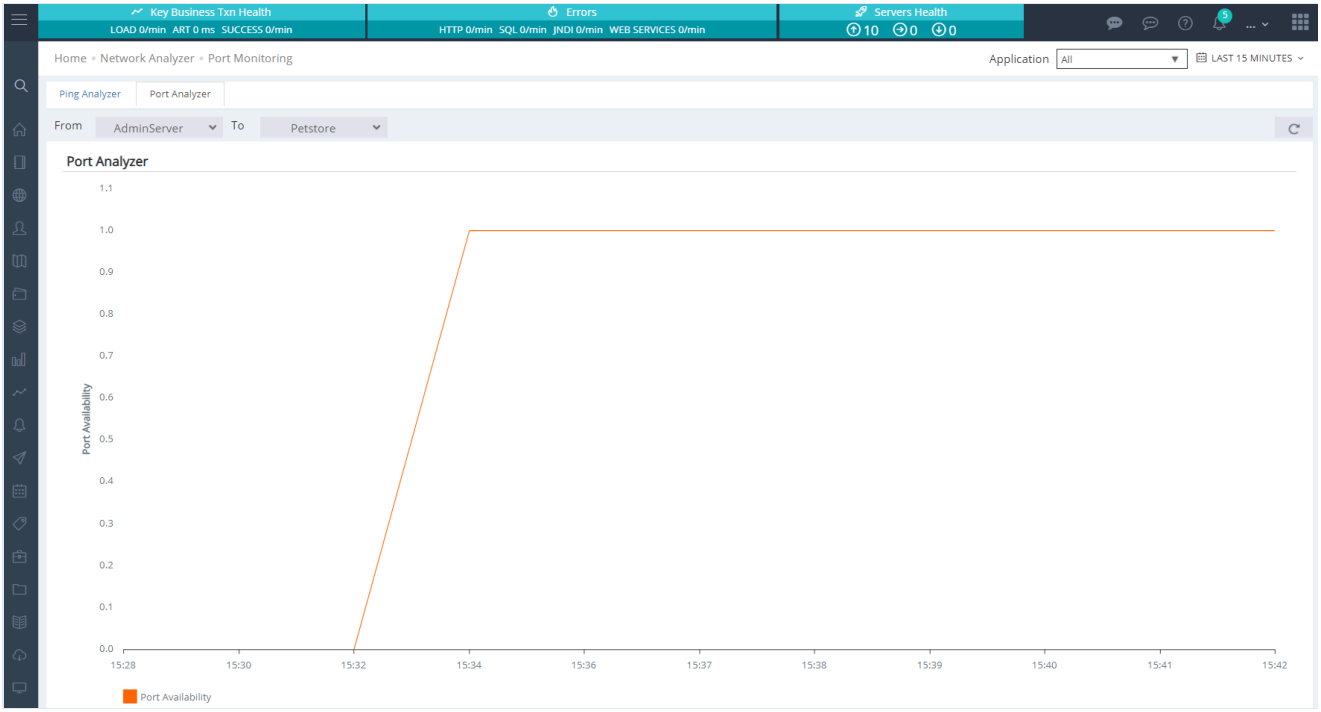
Ping Analyzer and Port Monitor

The Ping Analyzer measure the latency and packet loss when pinging a server from either the Applicare server or Applicare agents. This utility is essential for diagnosing network issues. You can configure alerts based on these metrics through JMX Alerts by navigating to Applicare Ping Monitor MBean and adjusting the Ping Time and Packet Loss properties.

Ping Analyzer

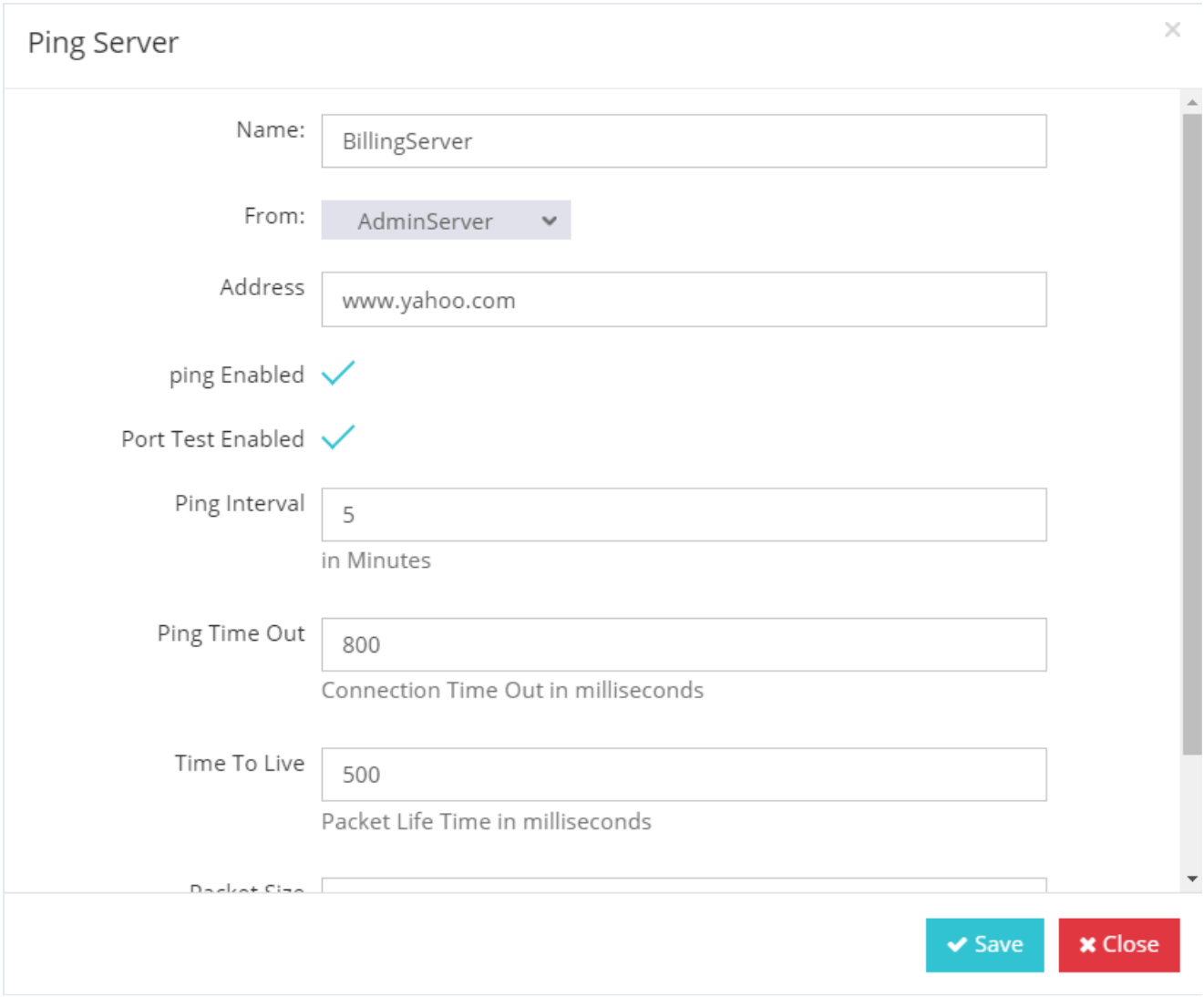


Port Analyzer



Parameters

Time to Live(TTL): Duration for which a sent packet remains active before being discarded. **Packet size:**The Size of the packet being sent. **Time Out:** Duration after which the connection will be terminated if no response is received.



Monitoring with Applicare

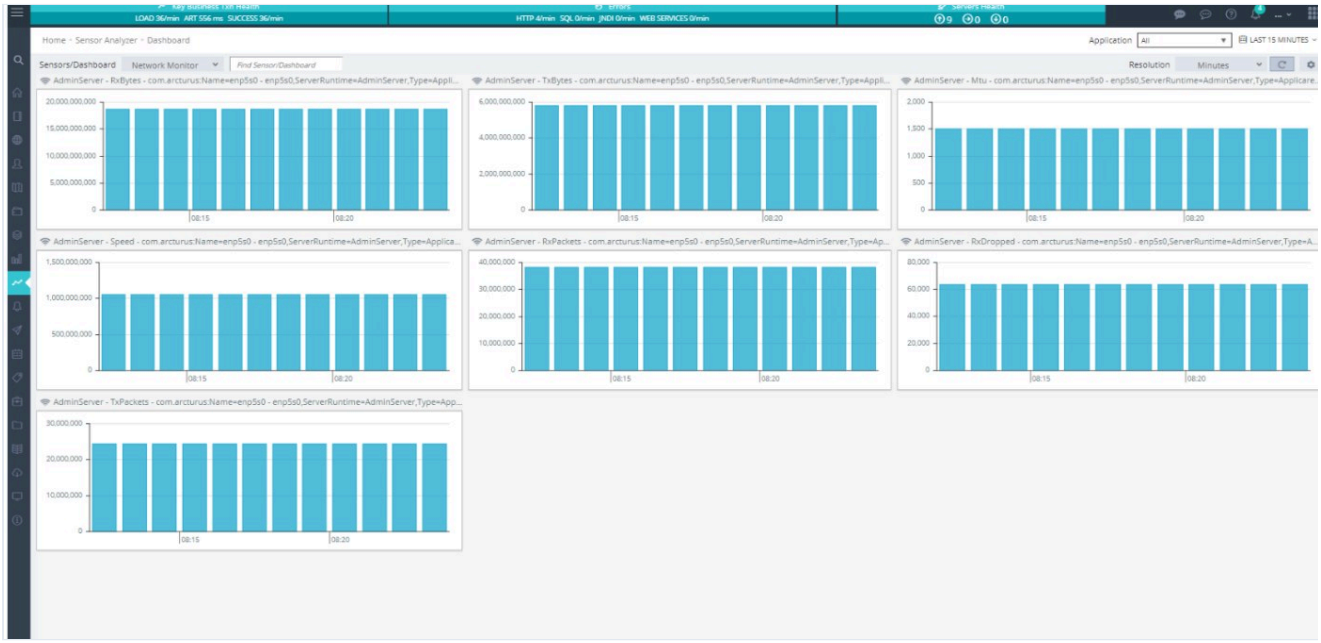
USING APPLICARE › INFRASTRUCTURE

Sensor Analyzer

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

The Sensors Analyzer offers an accessible view of data gathered by JMX Sensors over time. For guidance on setting up JMX Sensors, refer to the instructions under Create Sensors.

You can customize your view by adding or removing charts. Simply click the **Sensors** button and choose the charts you wish to include or exclude.



Monitoring with Applicare

USING APPLICARE › INFRASTRUCTURE

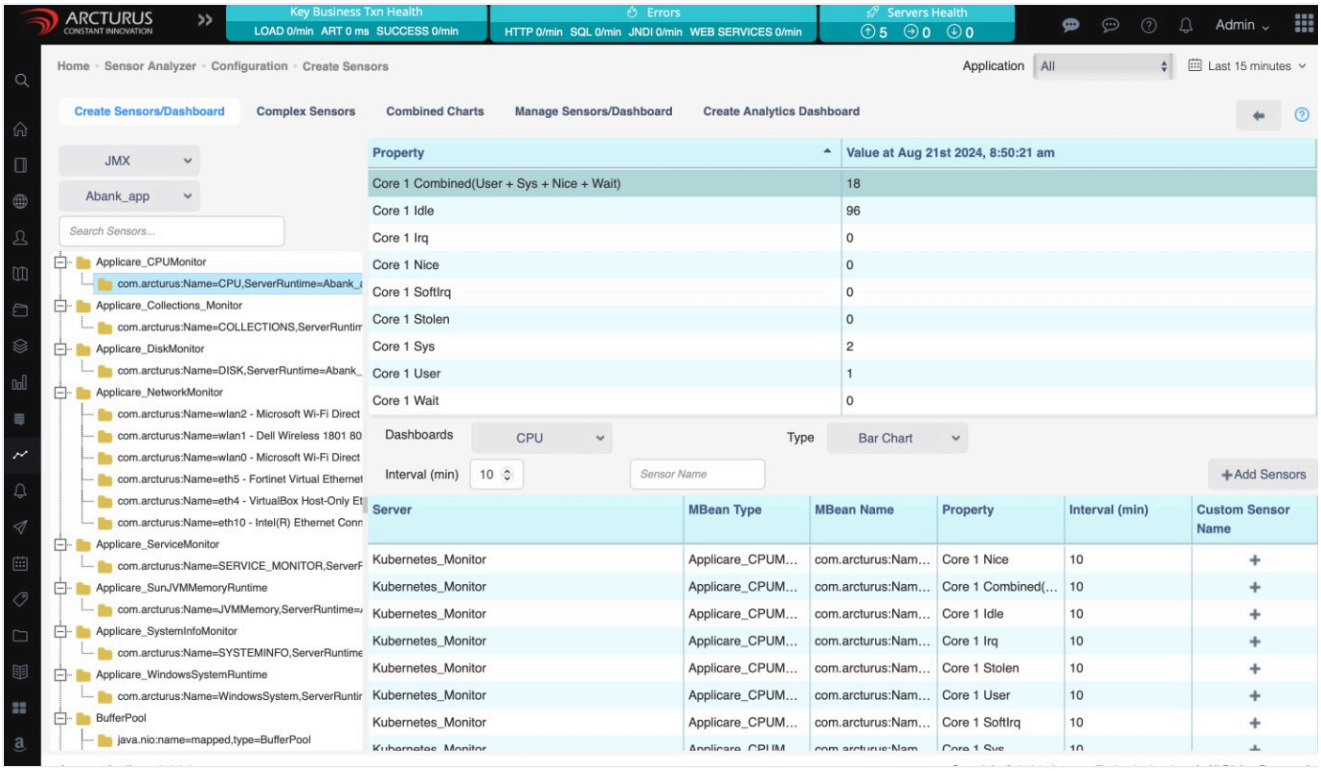
Applicare Custom JMX Analysis: Create a Sensor

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

An Applicare JMX Sensor is a JMX property whose value is stored by Applicare for analysis at any time. The server instance does not need to be active during the analysis. Applicare uses an efficient mechanism to persist sensor data with minimal overhead.

Create Sensors

To access the "Creating JMX Sensors" screen, select **Applicare Home > Sensors Analyzer > Configure Sensors** button on right (wheel). The tree structure on the **Create Sensors** tab can be expanded to display the MBeans and its properties. You can select the Properties to create sensors and click **Save Sensors** button to save. This is illustrated below:

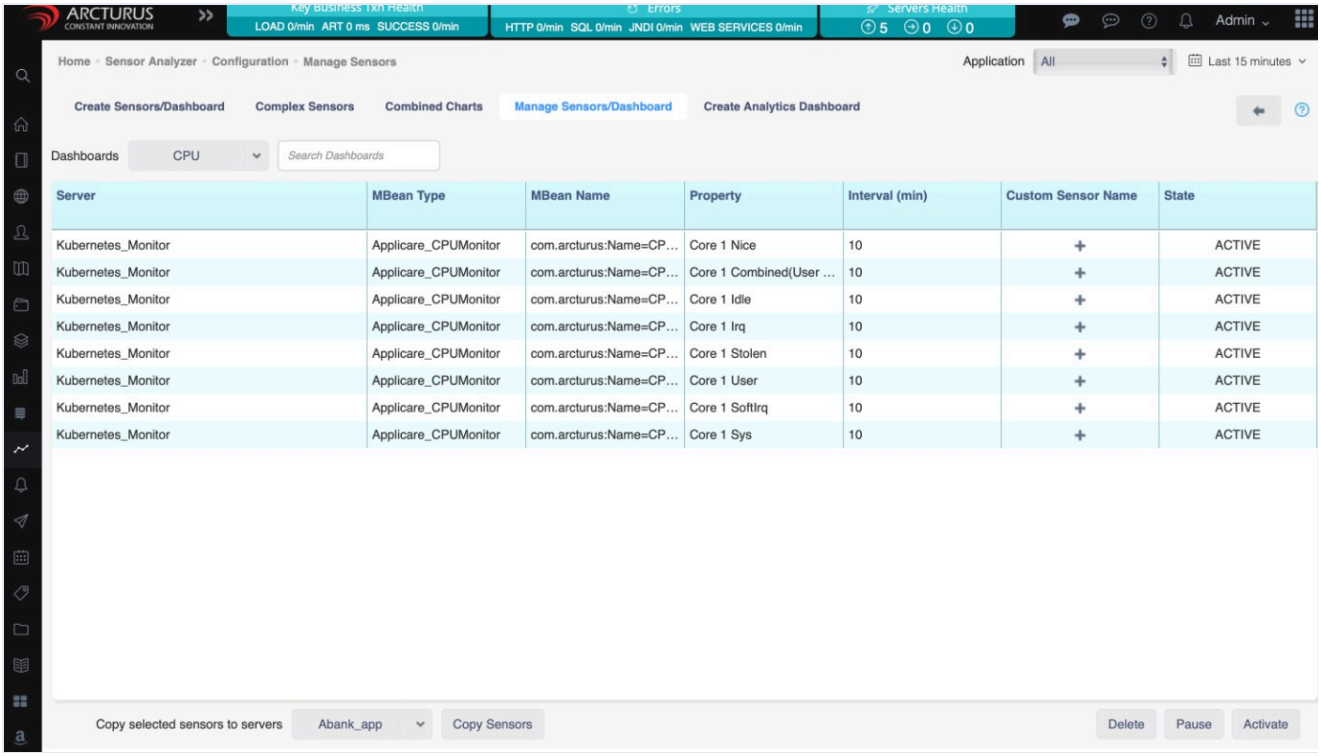


Click here to configure DB Query Sensor

Manage Sensors

Manage Sensors tab can be used to view all created sensors, delete and copy sensors from one server to another.

To copy sensors, select sensors you want to copy and select the target server from server drop down at the bottom. Click **Copy Sensors** button to perform the copy. MBean names should exist on the target server.



Monitoring with Applicare

USING APPLICARE › INFRASTRUCTURE

Monitoring MSSQL Metrics with DB Query Sensors

Applicare 9.2 · Updated August 29, 2024 · **Pending technical review**

The user must have enough permissions to access these metrics.

For details on user Permissions and DB Configuration Click here

Using DB Query Sensors to Retrieve MSSQL Metrics

Private Sensor Name: Total Connection **SQL:** select sum(cntr_value), '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name in ('User Connections','Logical Connections');

Private Sensor Name: User Connections **SQL:** select cntr_value,'~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='User Connections';

Private Sensor Name: Logical Connections **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Logical Connections';

Private Sensor Name: Active Temp Tables **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Active Temp Tables';

Private Sensor Name: Lock Blocks Allocated **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Lock Blocks Allocated';

Private Sensor Name: Lock Owner Blocks Allocated **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Lock Owner Blocks Allocated';

Private Sensor Name: Connection Memory **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Connection Memory (KB)';

Private Sensor Name: Process blocked **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Processes blocked';

Private Sensor Name: Transactions **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Transactions' and object_name like '%Transactions%';

Private Sensor Name: Active Transactions **SQL:** select sum(cntr_value), '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Active Transactions';

Private Sensor Name: Total Database file size (KB) **SQL:** select sum(cntr_value), '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Data File(s) Size (KB)';

Private Sensor Name: Total Database Log file size (KB) **SQL:** select sum(cntr_value), '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Log File(s) Size (KB)';

Private Sensor Name: Buffer cache hit ratio **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Buffer cache hit ratio';

Private Sensor Name: Database pages **SQL:** select cntr_value, '~' AS Applicare_Internal from sys.dm_os_performance_counters where counter_name='Database pages' and object_name like '%Transactions%';

USING APPLICARE › INFRASTRUCTURE

Server Availability Monitoring

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Overview

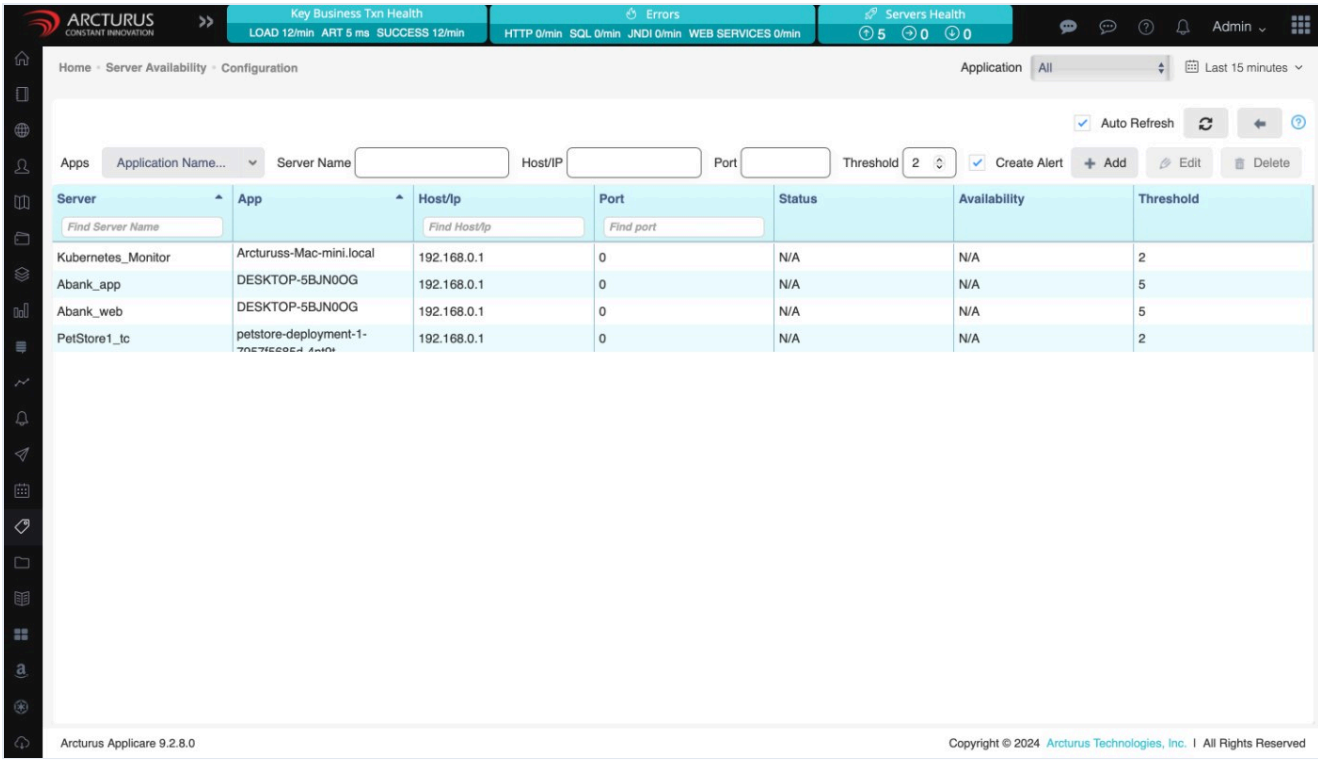
Server availability monitoring tracks changes in server status and generates reports on uptime and performance.

Server Availability

The Server Availability chart displays the status of the servers as available, not available, not monitored, or under blackout. A blackout status indicates that a blackout was or is currently in effect during that time period.



Server Availability Monitoring Configuration



Monitoring with Applicare

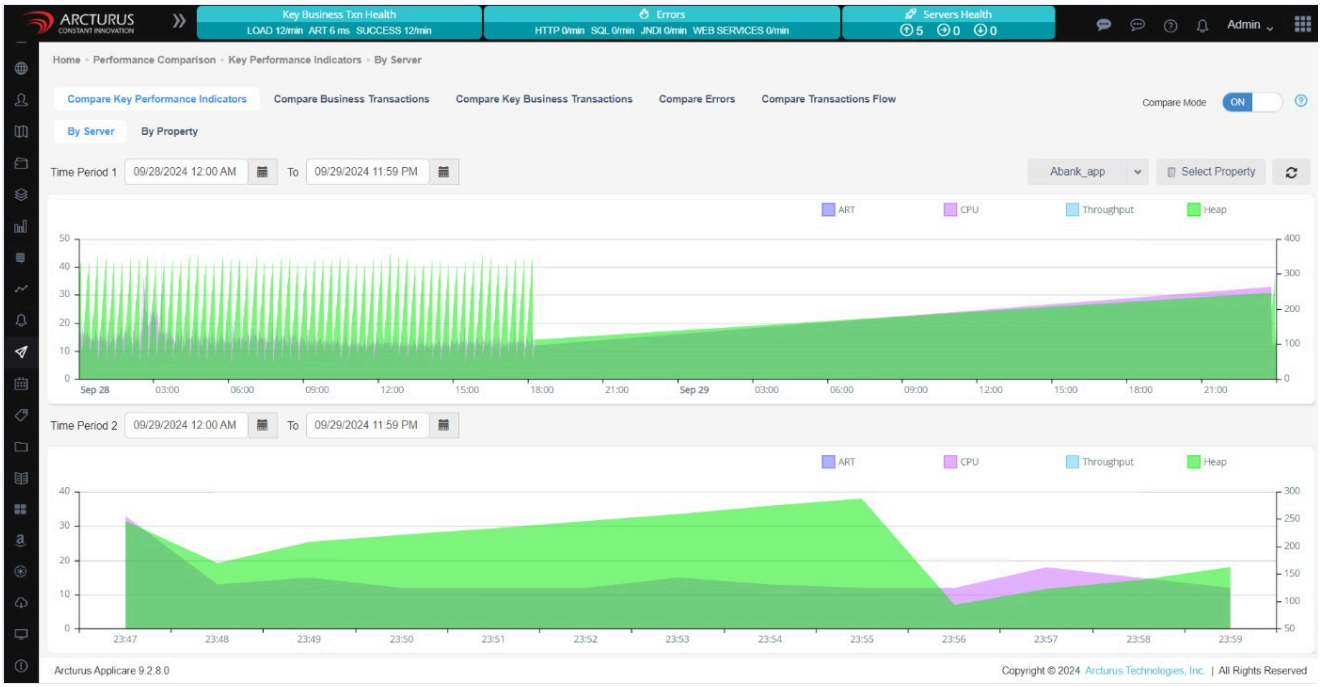
USING APPLICARE › INFRASTRUCTURE

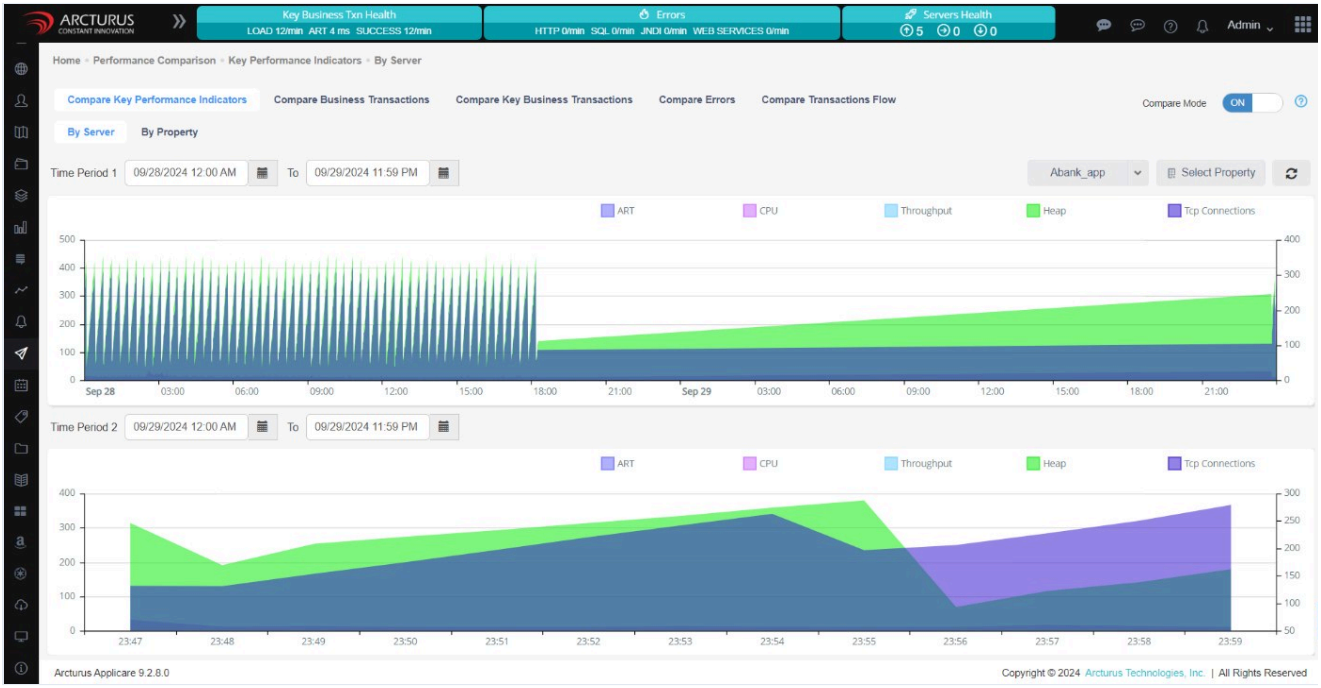
User Guide for Performance Comparison

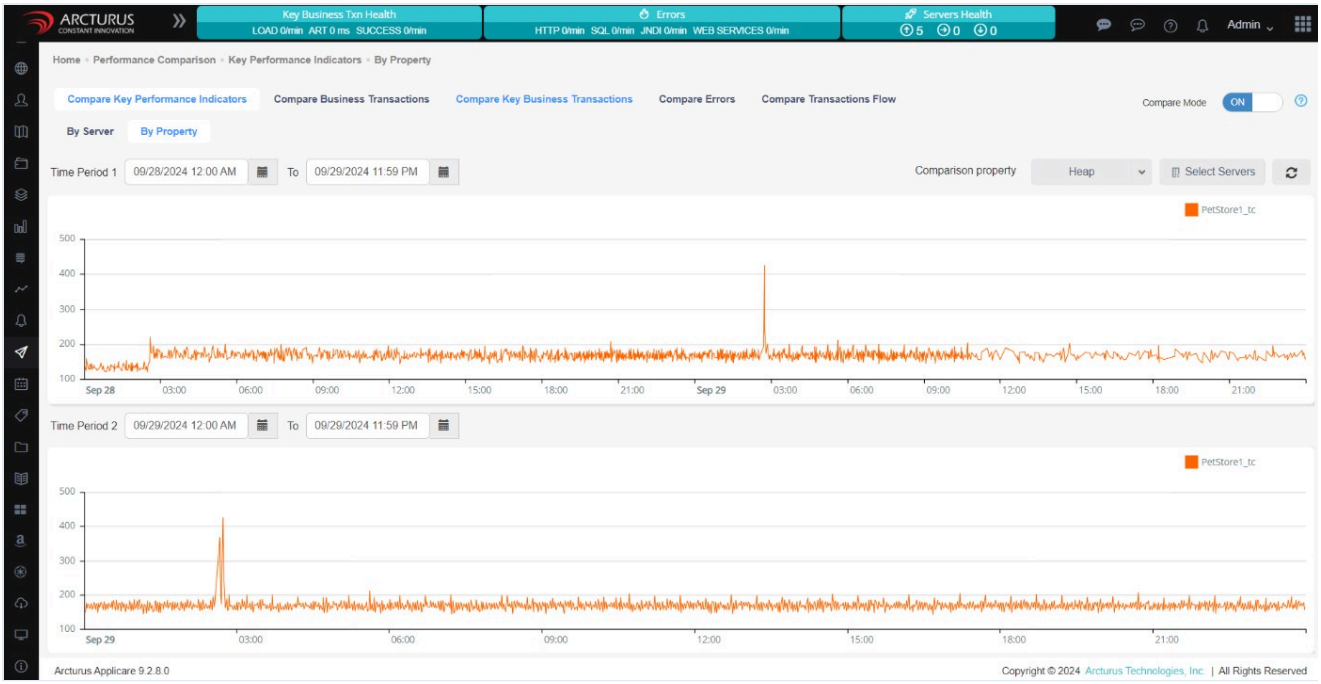
Applicare 9.2 · Updated September 30, 2024 · **Pending technical review**

Key Performance Indicators

Performance Comparison allows of various performance metrics such as (Heap, CPU, Throughput, Average Response Time etc.) between two time periods. It is extremely simple - select the time periods, select servers to compare, choose property to compare e.g. CPU on NodeServer16.

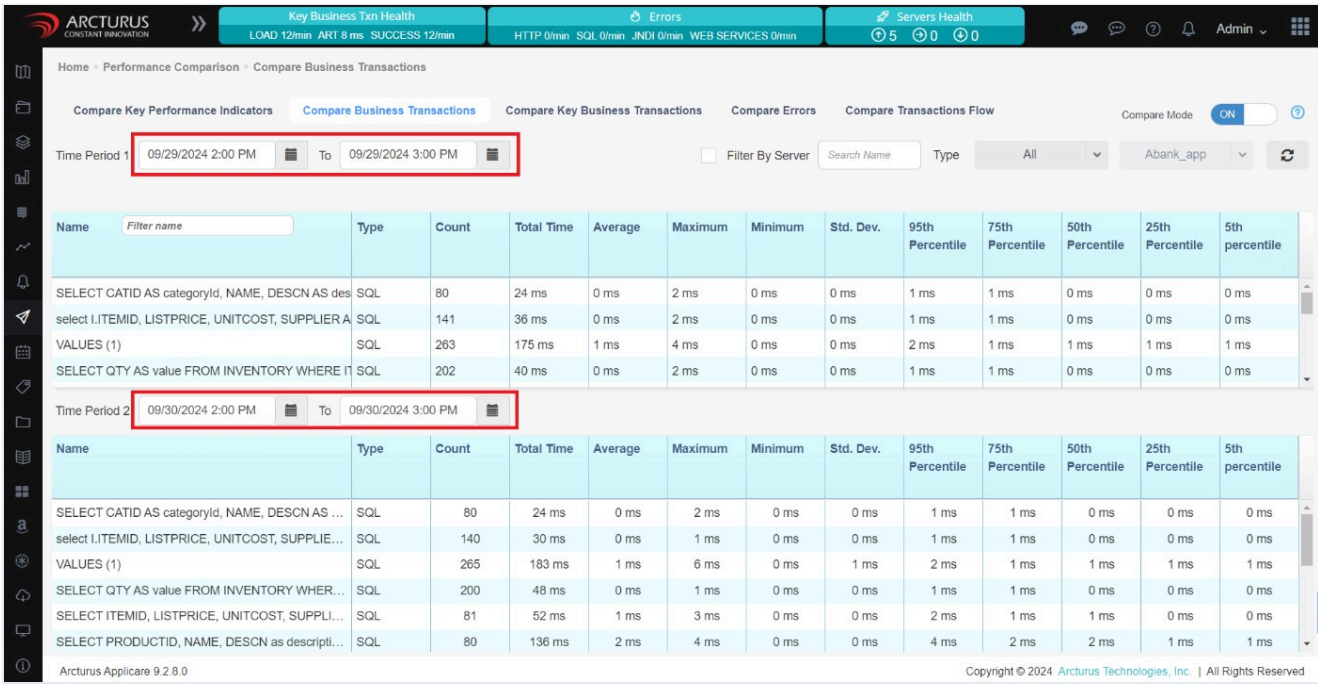






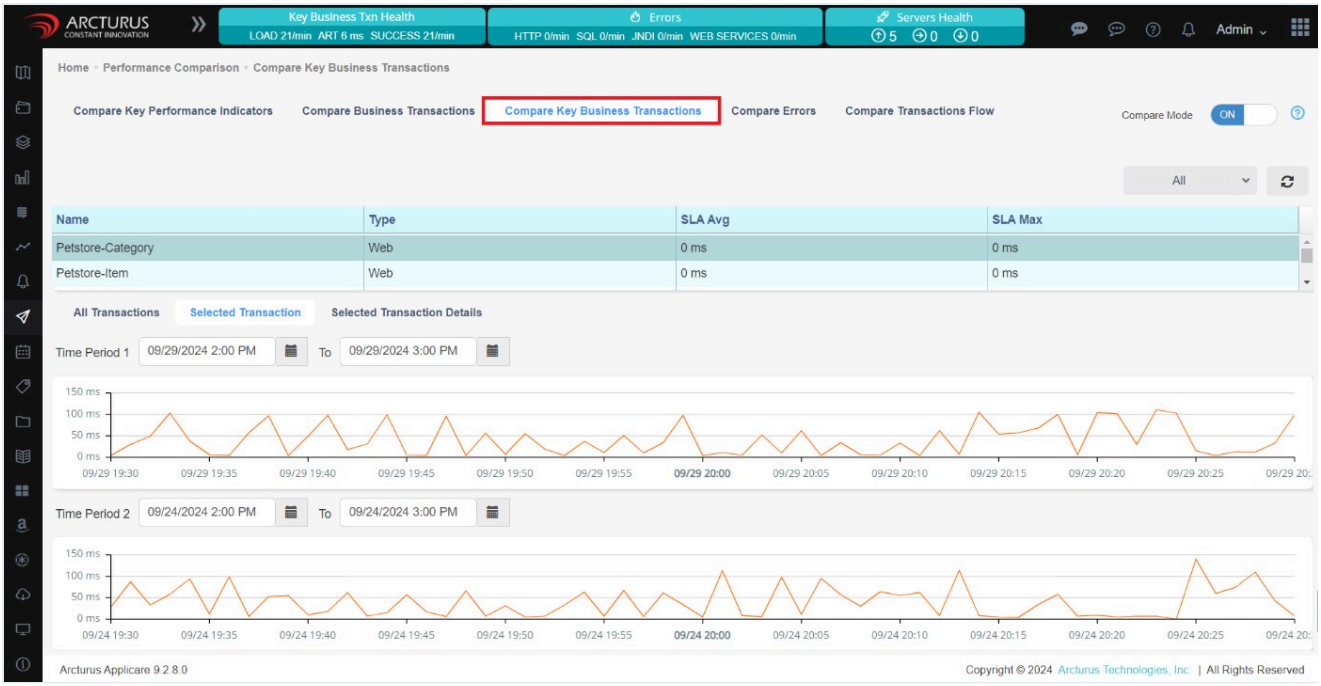
Compare Business Transactions

Displays the HTTP data for all servers in the Applicare Controller based on the specified time period.



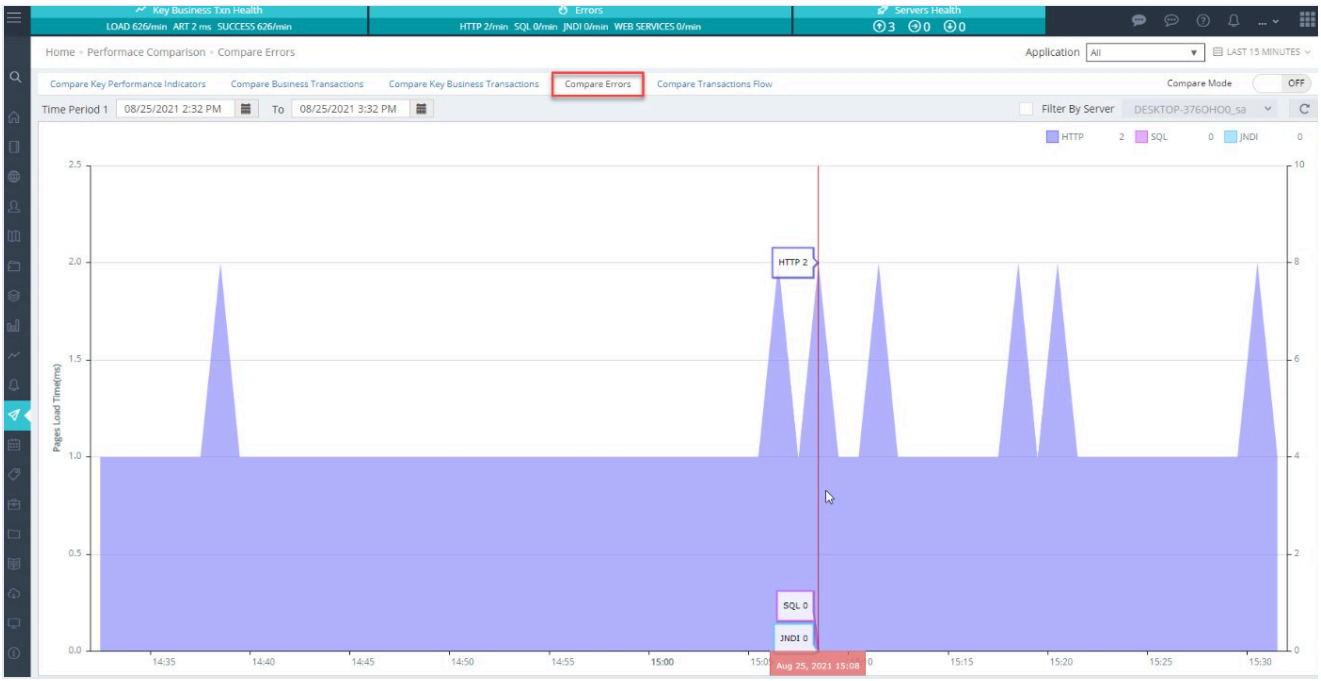
Compare Key Business Transactions

If KBT is added to the server in the Applicare controller then all the KBT will be displayed according to the time period we have given.

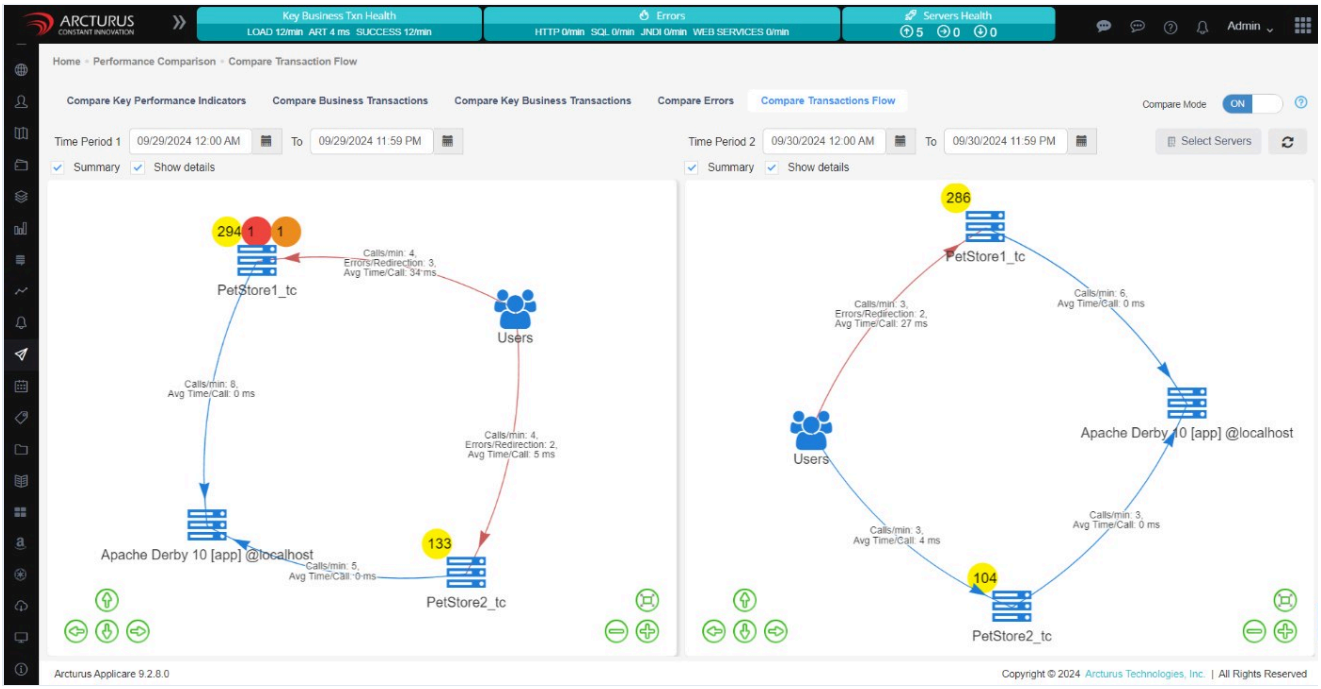


Compare Errors

Display any errors like HTTP errors, SQL errors, or JNDI issues, encountered on the server within the specified time period in the Applicare controller.



Compare Transaction Flow

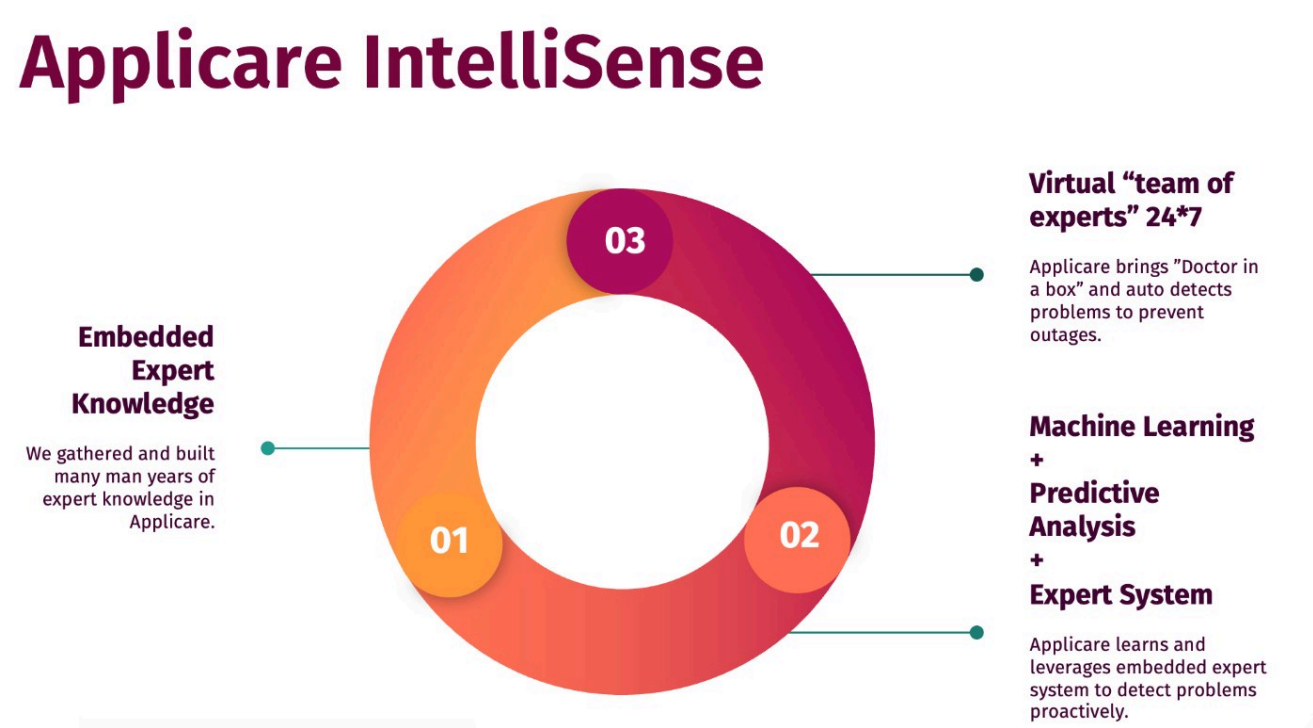


Monitoring with Applicare

USING APPLICARE › AIOPS

Applicare IntelliSense

Applicare 9.2 · Updated September 13, 2024 · aiops · **Pending technical review**



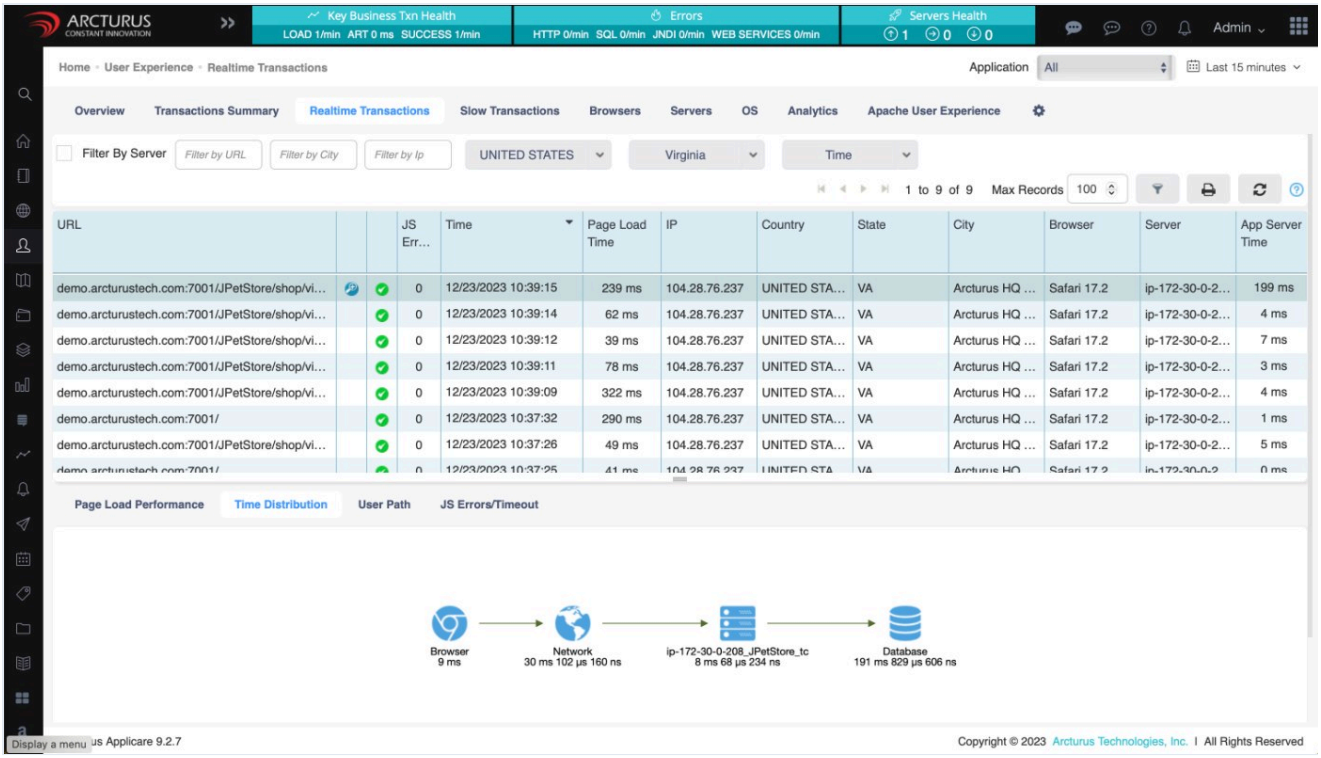
Applicare IntelliSense leverages AI to generate

- Machine Learning for Anomaly Detection
- Predictive Forecasting
- Problematic Pattern Recognition and Alerting
- Generative AI

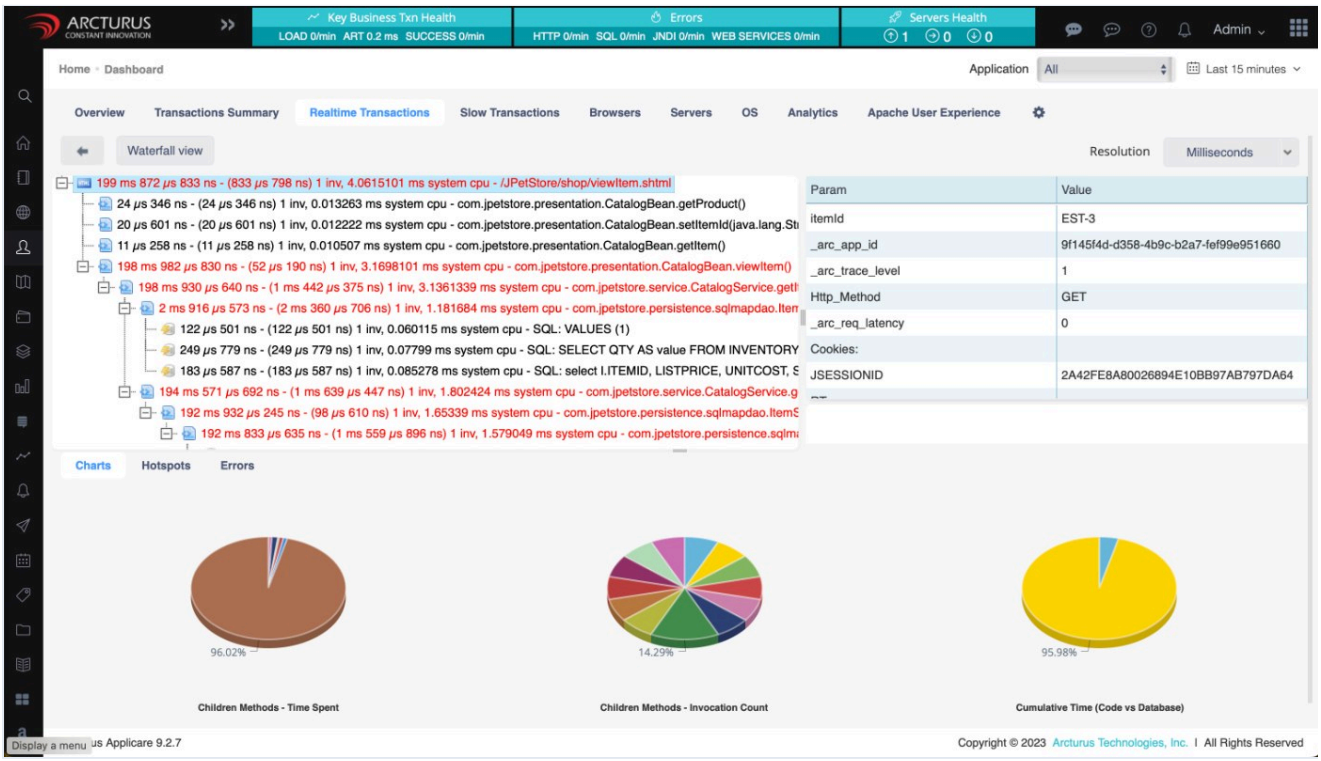
Machine Learning for Anomaly Detection

Applicare IntelliSense learns in real time from each transaction that passes through Applicare. Using machine learning Applicare automatically builds baseline and topology for each transaction type and each method with the transaction. IntelliSense leverages this machine learnt data for anomaly detections and is able to identify transactions that are slow and pin point the cause of the slowness within that transaction. IntelliSense using its machine learning does this root cause analysis down to the line of code that is causing the issue in the very first occurrence of the problem without requiring any human intervention.

Applicare auto detects slow transaction/s, identifies slowness is coming from database tier



and captures the complete stack trace (only for the slow transactions).



Predictive Forecasting

Applicare offers predictive forecasting with auto-alerts for when key metrics like Average Response Time (ART) and CPU usage approach critical thresholds.

Problematic Pattern Recognition and Alerting

Applicare has many man years for expert knowledge that is constantly working without any breaks in customer's environment. Applicare InstelliSense leveraging this knowledge auto identifies complex problematic patterns that are causing issues in the the environment e.g. detecting packets getting dropped from an ethernet card and that causing slowness in transactions due to resubmission.

We are constantly gathering these patterns from around the globe, validating and adding them to Applicare via updates. This way Applicare is getting smarter on a daily basis with a strong team of experts behind it.

Generative AI

Our team is working some of the coolest use cases of generational AI as we speak and we can't wait to show you those in our upcoming releases. Hang tight!

Monitoring with Applicare

USING APPLICARE › AIOPS

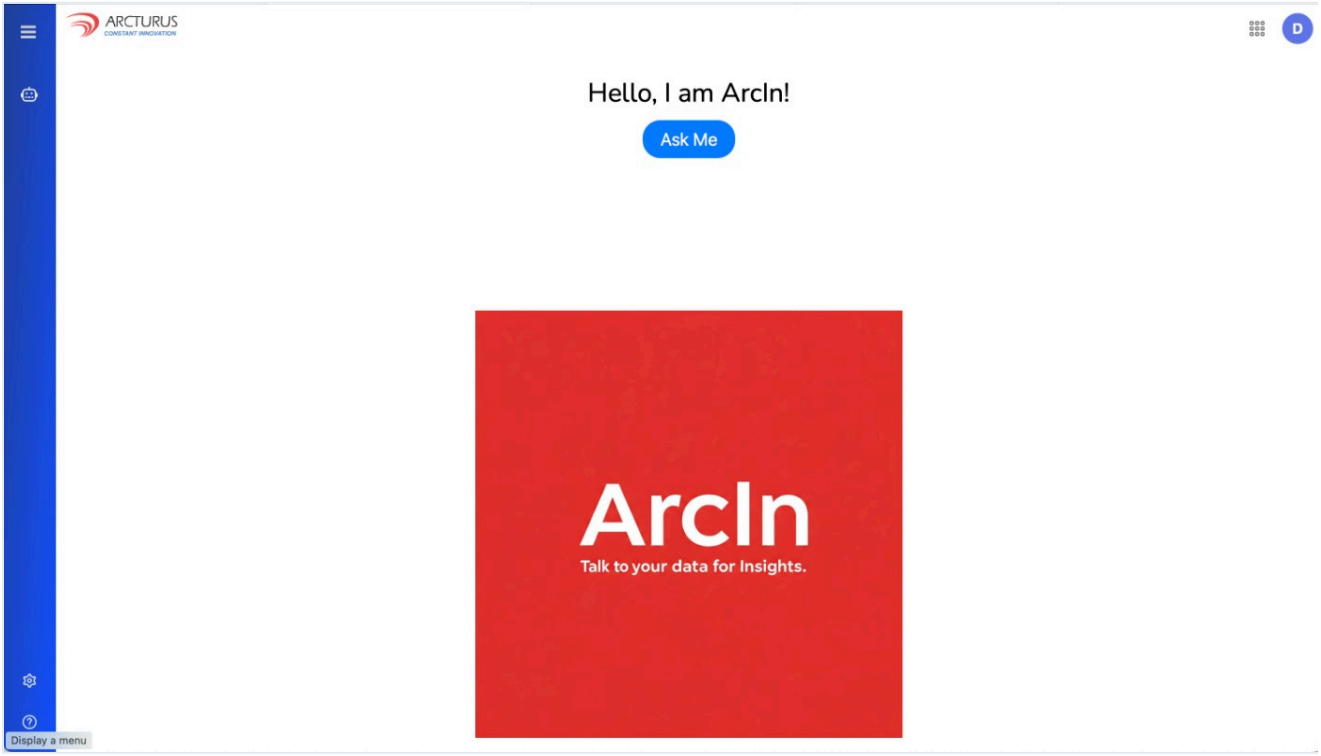
Arcturus Insights - ArcIn

Applicare 9.2 · Updated April 1, 2025 · aiops · **Pending technical review**

Talk to your data for Insights. Simply login to ArcIn, configure your database and start asking questions.

Login at

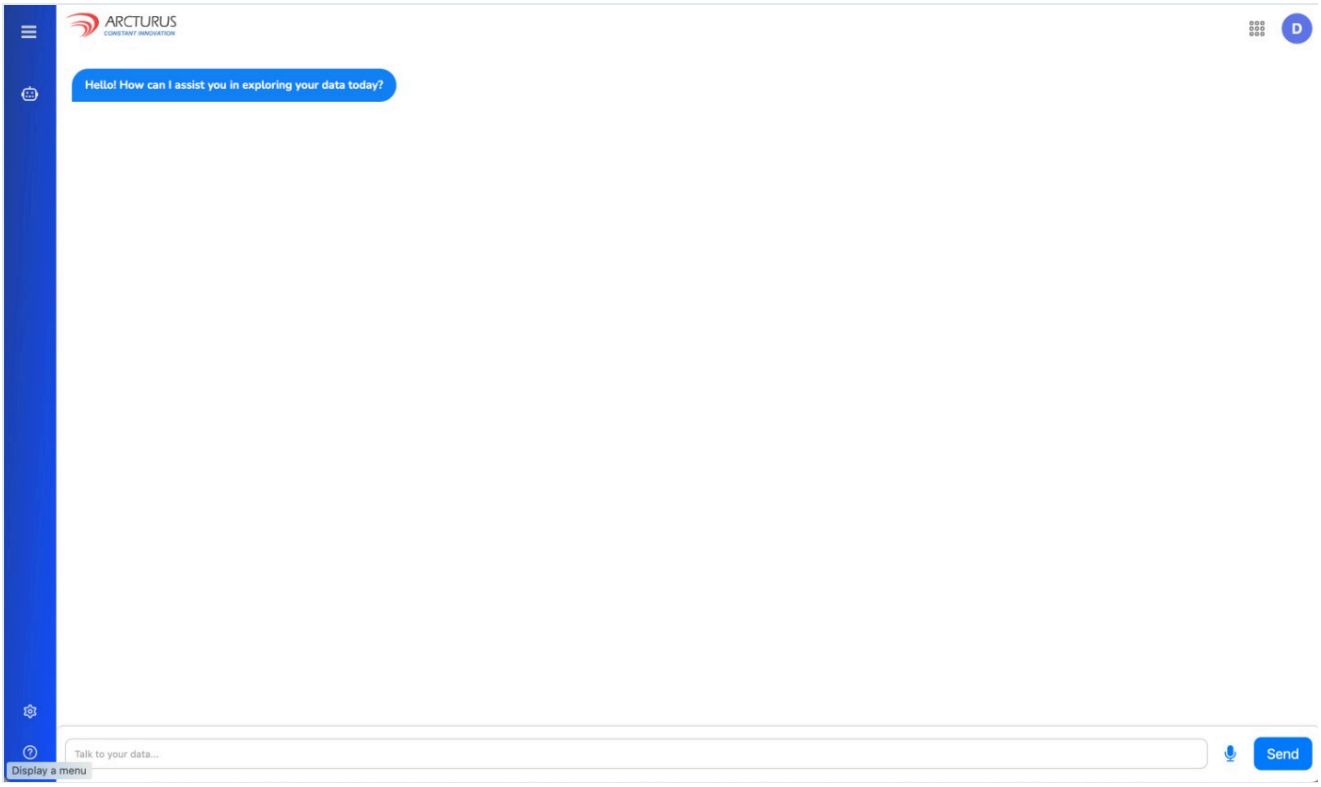
```
TEXT
https://arcin.applicare.io/
```



Configure database to talk to and get insights.

https://helpdesk.arcturustech.com/hc/en-us/articles/35357028029581-ArcIn-Database-Configuration

Simply ask questions about your data and get instant answers!


ARCTURUS
D
Hello! How can I assist you in exploring your data today?
Talk to your data...
Send
Display a menu

USING APPLICARE › AIOPS

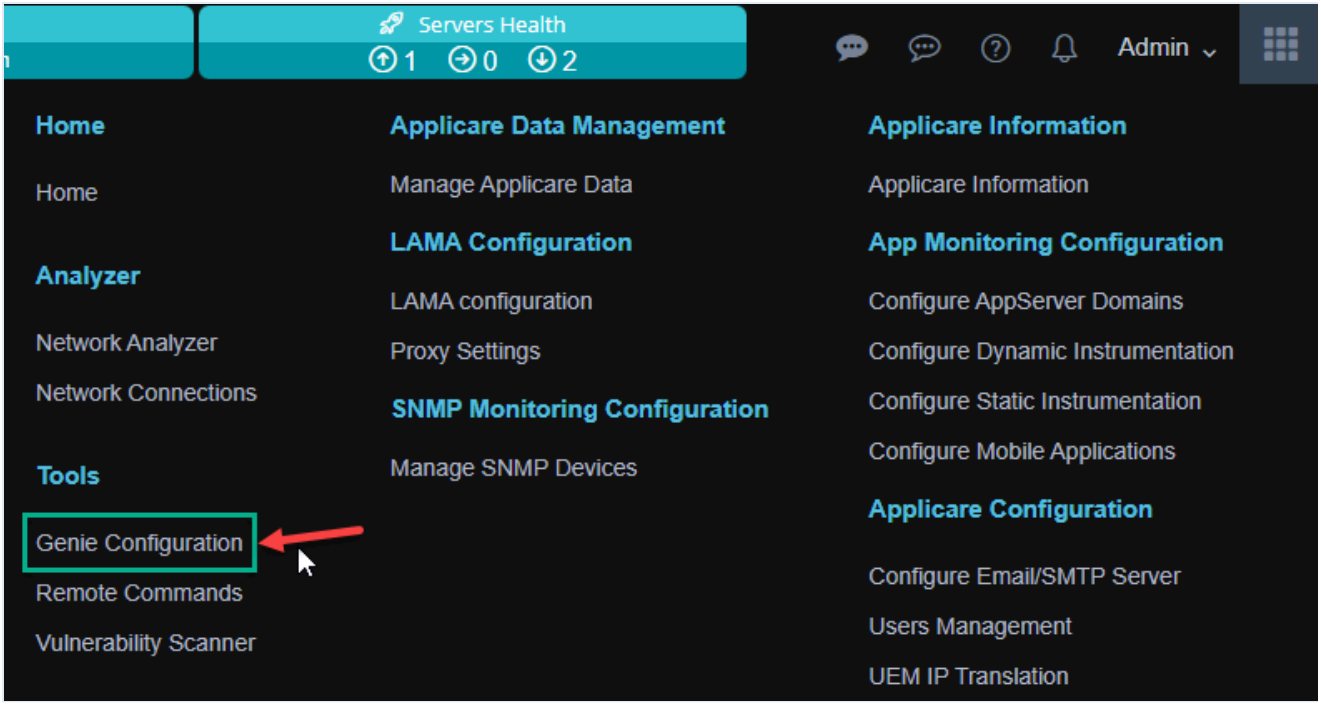
Applicare Genie

Applicare 9.2 · Updated October 23, 2024 · aiops · **Pending technical review**

One of the standout features in our Applicare is the **Genie Decision Support System**, an intelligent AI-powered assistant designed to revolutionize server management and monitoring. With Genie, users can effortlessly inquire about real-time server status, receive detailed alerts, and access critical performance data through a conversational interface. This seamless interaction allows for quick retrieval of essential information, whether tracking server health or investigating system alerts, significantly enhancing efficiency and oversight.

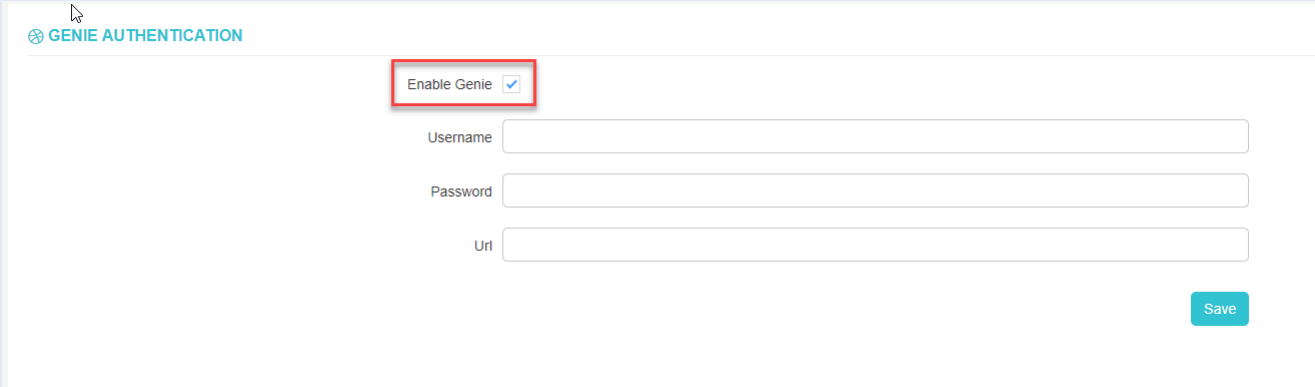
Step 1: Open Genie

1. Navigate to the Right side menu
2. Find the option labeled **Genie Configuration**. Clicking on it will take you to the configuration page where you can set up Genie.



Step 2: Enable Genie

1. Once in the configuration page, you will see a checkbox to **Enable Genie**.
2. Switch it on to activate the chat bot feature.



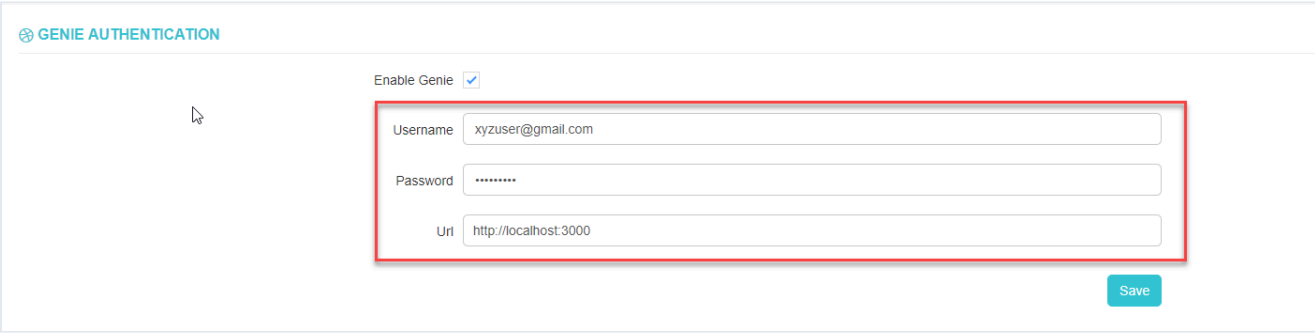
Step 3: Provide the Required Credentials

1. **Enter the URL**: You will need to provide the URL where Genie will connect. Enter the correct Genie server URL (e.g., `http://localhost:3000`) or (e.g., `https://localhost:3000`) in the field labeled **Genie Chat URL**.

   Localhost refers to the **IP address** `127.0.0.1`, which is the loopback address, commonly referred to as "localhost."

   3000 is the **port number** where the server or service is listening for incoming connections.

   2. Username and Password: In the respective fields, enter the **Genie Username** and **Genie Password**. These credentials are necessary for authenticating and establishing a secure connection with the Genie Chat Bot.



Step 4: Save the Details

1. After entering the username, password and URL, click on the **Save** button to proceed.
2. This will save the credentials and URL, ensuring that Genie is properly configured and ready to interact with your system.

Step 5: Start Using Genie

1. Once the setup is complete and saved, Genie will be ready to use. You can now access the chat bot.

USING APPLICARE › AIOPS

ArcIn Database Configuration

Applicare 9.2 · Updated March 27, 2026 · aiops · **Pending technical review**

Prerequisites: Allow the **target database** to accept connections from the IP address 20.57.1.113.

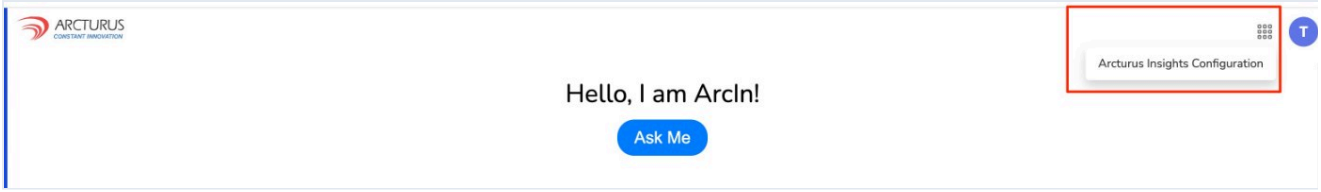
Step 1: If you don't have an Arcturus account, register here

```text
https://support.arcturustech.com/register
```

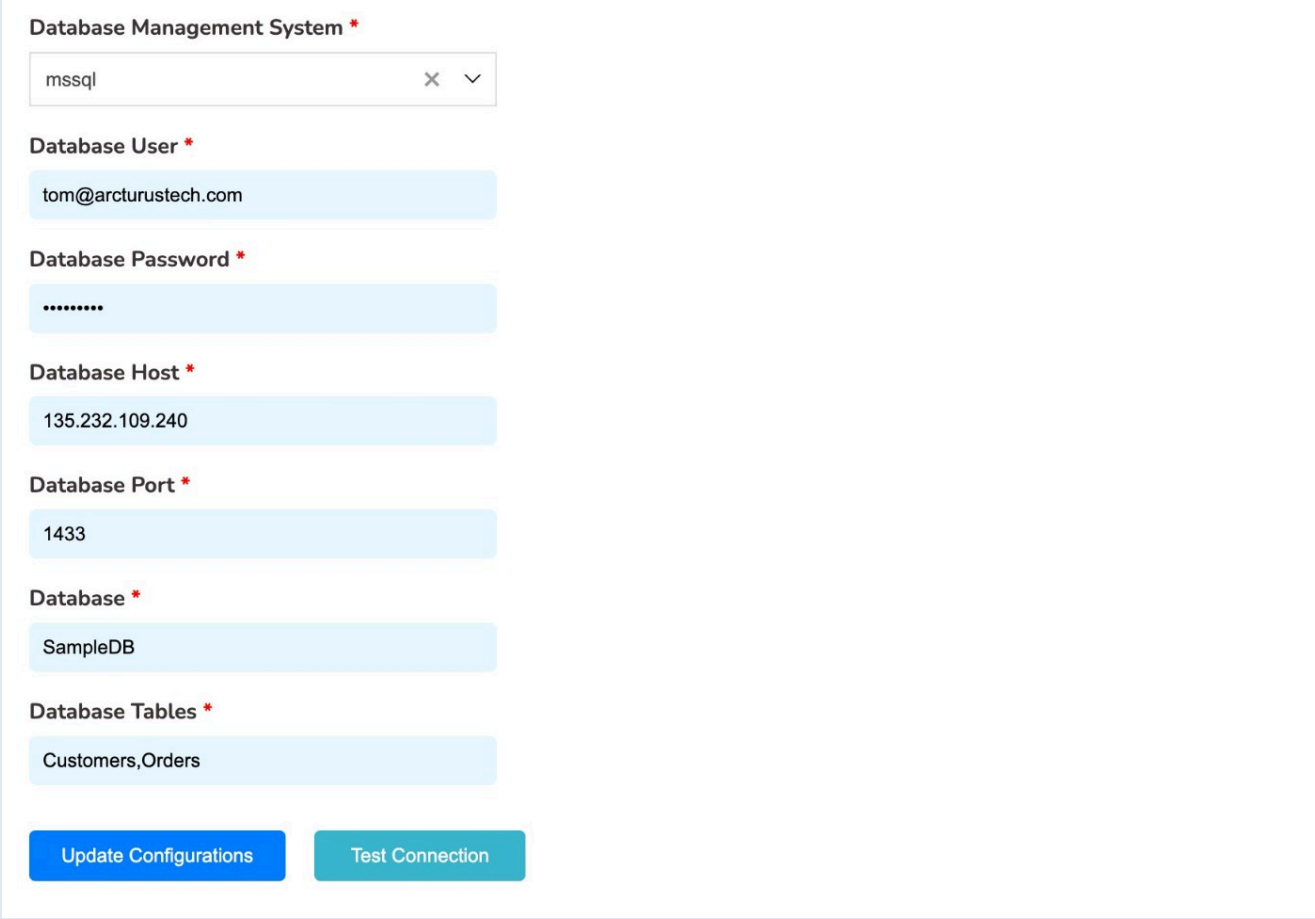
Step 2: Login our ArcIn site using the registered support site credentials.

```text
https://arcin.applicare.io/
```

Step 3: Navigate to Arcturus Insights Configuration page.



Step 4: Select your database type from the **"Database Management System"** dropdown, enter the required database details, and click **"Update configuration"** to save your settings.



Test Connection and ensure test is successful. Once you have successfully tested database connectivity, you can start asking questions. Its that simple.

USING APPLICARE › AIOPS

Set Up Predictive Analysis in Applicare on Linux

Applicare 9.2 · Updated August 22, 2024 · aiops, linux · **Pending technical review**

Step 1: Open up a command prompt or terminal then navigate to the Applicare home directory.

Step 2: Create a directory named **predictive_analysis** in the Applicare home path and set the appropriate permission for this directory.

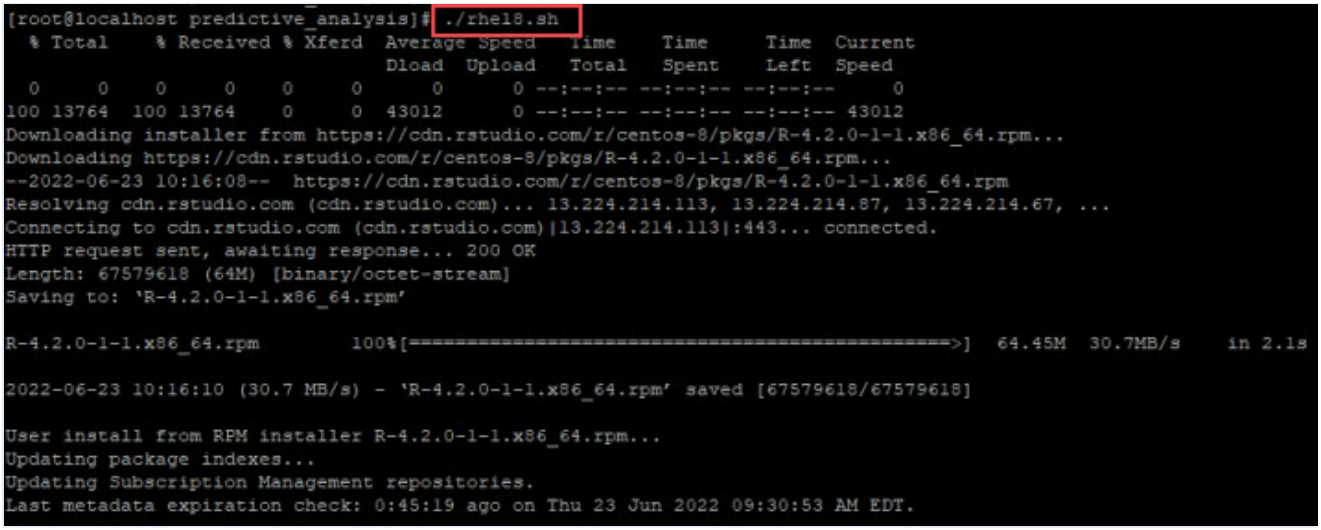
```bash
chmod -R 777 <predictive_analysis>
```

Step 3: Navigate to the **predictive_analysis** directory and create a file named **rhel8.sh.** Paste the following contents into that file and save it. Then set the appropriate permission for this file.

```bash
#!/bin/bash
export R_VERSION=4.2.0
bash -c "$(curl -L https://rstd.io/r-install)"
ln -s /opt/R/${R_VERSION}/bin/R /usr/local/bin/R
ln -s /opt/R/${R_VERSION}/bin/Rscript /usr/local/bin/Rscript
chmod -R 777 pred.R
```

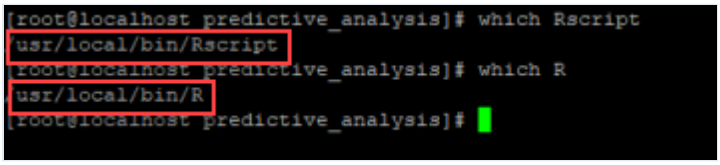
```bash
chmod +x rhel8.sh
```

Step 4: After executing the **basch script.** It should produce output similar to the example below.



Step 5: Ensure that **R** is correctly installed and located. Execute the below command it.

```text
which Rscript
```



The result should be **"usr/local/bin/Rscript"**

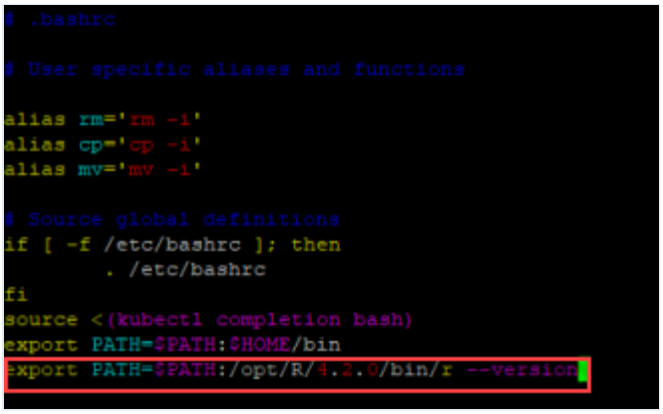
Apply step 6 ,7 and 8 only if you didn't get results on executing which R command.

Step 6: If the results differ from the expected output, modify your .bashrc file to set the correct path for **R**:

```text
vim ~/.bashrc   //OR THE PATH TO YOUR BASHRC FILE
```

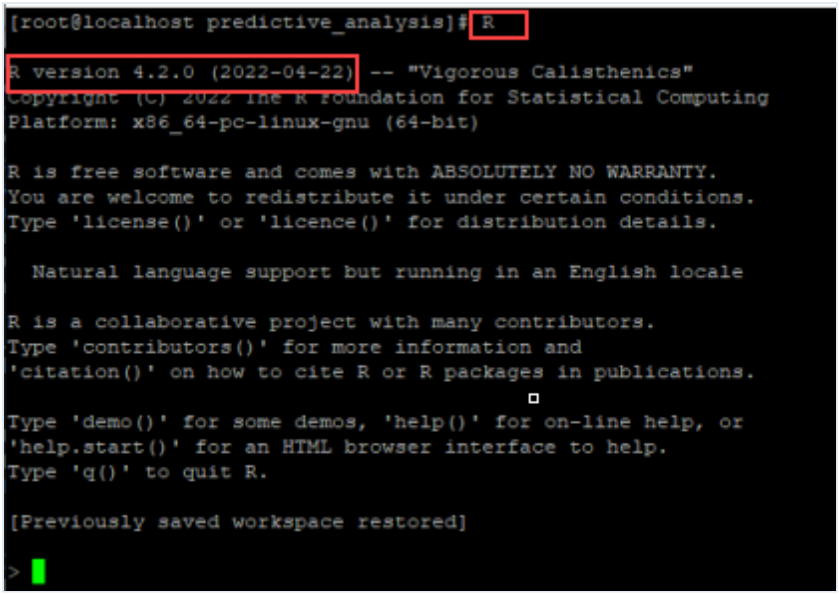
Step 7: Once you are in the bashrc file, add the following line to your file:

```bash
export PATH=$PATH:/opt/R/4.2.0/bin/r --version
```



Step 8: Save and exit the file

Step 9: To verify that **R** was properly installed, type **R** in the terminal. You should see the **R** interface appear in the terminal or command prompt.




```
TEXT
R should now be successfully installed and the Predictive Analysis program should be running in the background.
```

Step 10: Go to <**Applicare_home_directory**>\server\bin and edit the **catalina.sh** file.

```
TEXT
-Dapplicare.rHomePathDir=<R_installed_path>
```

Enter the directory displayed after executing **R** in **step 5,** replacing the **<R_installed_path>** placeholder with this directory path .

Example

```
TEXT
-Dapplicare.rHomePathDir=/usr/local/bin/Rscript
```

USING APPLICARE › USER EXPERIENCE MONITORING

End User Experience Monitoring (EUM)/User Experience Monitoring (UEM) Analyzer

Applicare 9.2 · Updated August 28, 2024 · **Pending technical review**

The End User Experience Analyzer delivers detailed performance insights into web applications as experienced by actual end users. Applicare agent automatically injects Java script code to JSP pages and that reports back the total time it took to load the page from the time a user clicked the link, network time, server processing and browser rendering times, browser type and user location.

Configuration

By default, Applicare scripts for collecting EUM data are served by the Applicare Controller. However, they can also be served by any Applicare Collector instance. Following JVM arguments need to be set in case a collector is serving these scripts and persisting returned data

Example URLs below assumes that '192.168.1.10' is the IP address of the Applicare Collector and '8080' is the port it is listens on. Please update the IP address and Port as needed adding these arguments and adding them to agent start.

VM Argument	Example & Description
arceum.js	-Darceum.js=`https://192.168.1.10:8080/applicare/eum/arc-eum.js`
	Utilize this parameter to specify from where the Applicare should load arc-eum.js This should be URL to applicare_agent application.
arceumprocess.js	-Darceumprocess.js=`https://192.168.1.10:8080/applicare/eum/arc-eum-process.js`
	Utilize this parameter to specify from where the Applicare should load arc-eum-procss.js This should be URL to applicare_agent application.
eum.beacon	-Deum.beacon=`https://192.168.1.10:8080/applicare/eum/beacon`
	Utilize this parameter to specify on which URL Applicare is listening for UEM data. This should be URL to applicare_agent application.
arc-nav.js	-Darc-nav.js=https://`192.168.1.10:8080/applicare`/eum/arc-nav.js
	If your web application is using servlets and you want to monitor them under UEM then you can utilize this parameter. By default Applicare does.
uem.servlets	-Duem.servlets=true
	If your web application is using servlets and you want to monitor them under UEM then you can utilize this parameter. By default Applicare does

The most crucial step is ensuring that these scripts and beacon URLs are accessible to end users. You might need to adjust firewall rules to permit external access to these URLs.

By setting the parameters mentioned, Applicare will inject JavaScript like the following into pages generated by your application's JSP/servlet when accessed by end users.

```xml
<script src="https://192.168.1.10:8080/applicare_agent/eum/arc-eum.js"
type="text/javascript"></script>

<script type="text/javascript">

BOOMR.addVar({ "server": "AdminServer","t_backend": "4","b_name":
BrowserDetect.browser,"b_ver": BrowserDetect.version,"b_os": BrowserDetect.OS });

BOOMR.init({user_ip: "192.168.1.16",beacon_url:
"https://192.168.1.10:8080/applicare_agent/eum/beacon"});

</script>
```

Next, enable the End User Experience Profiling aspect on your servers to automatically inject JavaScript into JSP pages. Follow these steps:

- Navigate to **Configuration > Configure Dynamic Instrumentation** page in Applicare console.
- On IntelliTrace Deployment tab, select Advanced Mode and click Configure button next to it.
- Move End User Experience Profiling aspect from left side list to Deployed Aspect list by selecting the aspect and clicking Add button.
- Click Apply button and make sure Agent is enabled for servers in IntelliTrace deployment tab.

Programmatic JavaScript Injection

If auto-instrumentation is not available or feasible, you can still implement end user monitoring by manually injecting the necessary scripts into your pages.

Applicare provides an API that allows you to insert the required JavaScript just before the "</body>" tag of your pages.

For instance, you can adjust your application's template to include the following for manual JavaScript injection:

```
... existing template code ...<%= com.arcturus.aop.shared.RUMManager.injectJS() %></body>
```

Additionally, ensure that "<Applicare Installation Dir>/applicare-aspects.jar" is included in your class path during development.

Manual JavaScript Injection for Static Pages

For end user experience monitoring of static pages, Applicare EUM scripts can be manually injected.

URL Exclusion & Inclusion

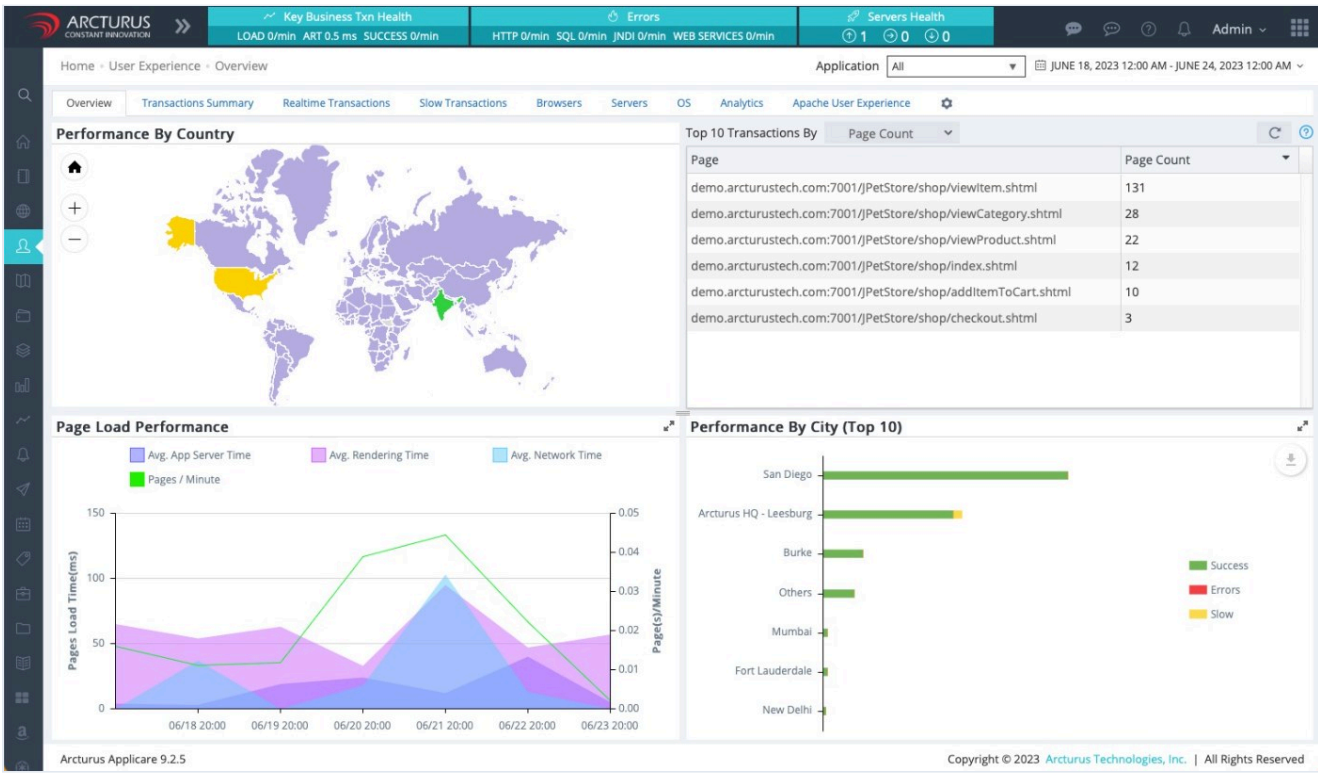
To manage URL monitoring in UEM: Add a URL from the HTTP Analyzer to the UEMURLExclusions.txt file to exclude it from monitoring.

Add a URL from the HTTP Analyzer to the UEMURLInclusions.txt file to include it in monitoring.

Analyzer

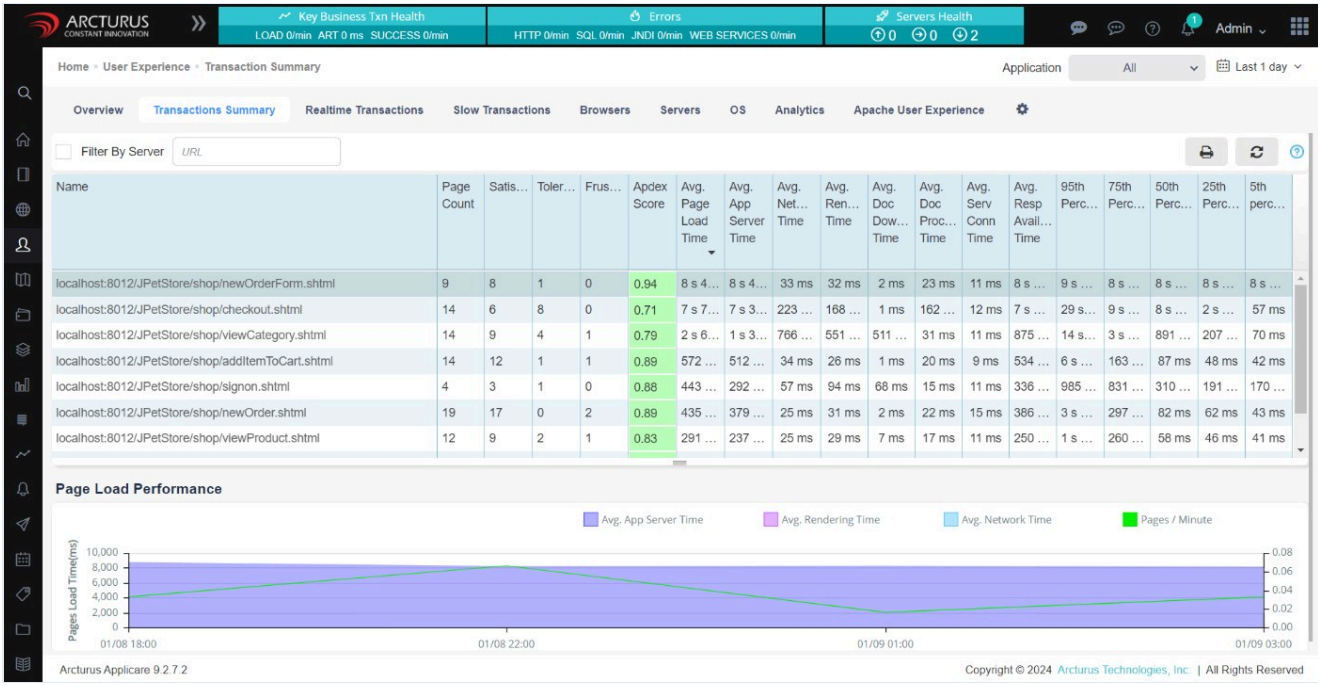
The End User Experience Analyzer presents historical performance data across various aspects in separate tabs. The Real-time Transactions tab updates immediately as users interact with your application, while data on other tabs may take a few minutes to appear.

Overview



The world map on the dashboard shows the performance breakdown for each country. Mouse over a country to see performance summary data for the country. On the right you can view top transactions by different criteria such as page count, average page load time etc... The chart below shows the performance data over the selected time period and if you click on a country in the map, the chart will display the performance data for the country as shown in the image below.

Transaction Summary



The Transaction Summary view shows performance statistics for transactions over a selected time period, with a timeline chart at the bottom. Clicking on a transaction will refresh the chart to display performance data for the selected transaction.

The **Apdex score** quantifies user satisfaction with web application performance on a scale from 0 to 1, where 1 signifies perfect satisfaction and 0 denotes complete dissatisfaction. It condenses various response times into a single metric.

By default, Applicare will consider the average response time of any transactions as its **Desired response time(T).**

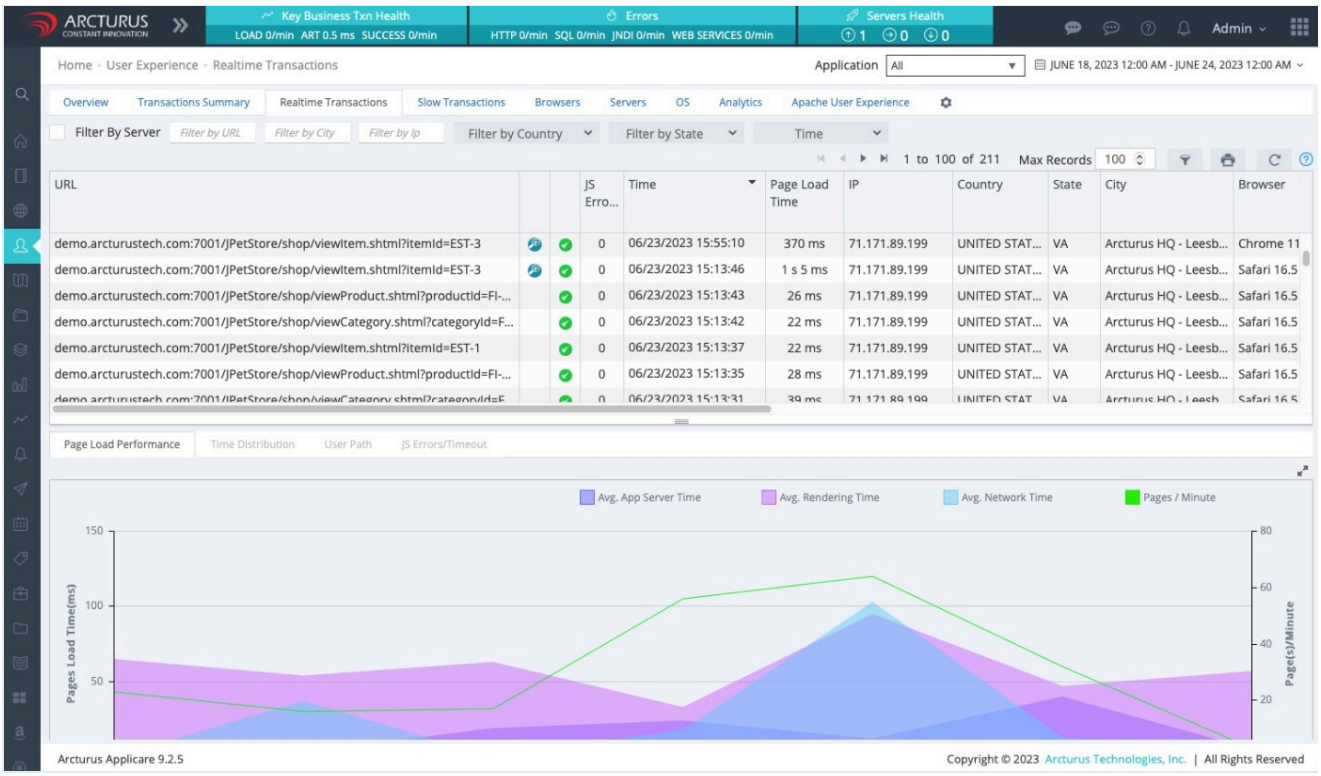
Transactions will be classified as follows:

- **Satisfied**: Response times ≤ T.
- **Tolerating**: Response times between T and 4T.
- **Frustrated**: Response times > 4T.

Apdex Score is calculated with the formula:

Apdex Score = (Satisfied Count + (Tolerating Count / 2)) / Total Count

Realtime Transactions



The Realtime view displays the last n number individual transactions recorded. Along with performance stats you can view location and browser information of users accessing your application. This view doesn't show any transactions older than a day.

In case the Applicare Controller/Collector collecting EUM data is behind a proxy, firewall or load balancer, make sure that the proxy, firewall or load balancing server forwards the client requests with "X-Forwarded-For" header parameter. If this parameter is not included in the HTTP request then Applicare may show the IP address of proxy, firewall or load balancing server in End User Experience Analyzer.

Below are the details on monitored data shown for each transaction

- **Resource Fetch Time:** The time it takes for a web browser to fetch a resource, such as an image, CSS file, or JavaScript file. This includes the time it takes to:
  - **DNS lookup:** The time it takes to resolve the domain name of the resource.
  - **TCP connection establishment:** The time it takes to establish a connection to the server that hosts the resource.
  - **TLS handshake:** The time it takes to establish a secure connection to the server, if the resource is hosted over HTTPS.
  - **Request:** The time it takes to send the request for the resource to the server.
  - **Response:** The time it takes to receive the response from the server, including the time it takes to download the resource.

    Resource Fetch Time is an important metric for measuring the performance of a web page. A short Resource Fetch Time means that the page will load more quickly, which can improve the user experience.

There are a number of factors that can affect Resource Fetch Time, including:

- **The size of the resource:** Larger resources will take longer to fetch.
- **The speed of the user's internet connection:** A slower internet connection will result in a longer Resource Fetch Time.
- **The performance of the browser:** A faster browser will be able to fetch resources more quickly.
- **The location of the server:** The closer the server is to the user, the shorter the Resource Fetch Time will be.

Here are some tips for improving Resource Fetch Time:

- **Use minified and compressed resources:** Minified and compressed resources are smaller, which means they will take less time to fetch.
- **Use a CDN:** A CDN (Content Delivery Network) can deliver resources from servers that are closer to the user, which can improve Resource Fetch Time.
- **Use a caching mechanism:** A caching mechanism can store resources that have already been fetched, which can reduce the need to fetch them again.

- **Document Download Time:** The time it takes to download a document from a web server to a user's device. This includes the time it takes to:
  - **DNS lookup:** The time it takes to resolve the domain name of the web server.
  - **TCP connection establishment:** The time it takes to establish a connection to the web server.
  - **TLS handshake:** The time it takes to establish a secure connection to the web server, if the request is made over HTTPS.
  - **Request:** The time it takes to send the request for the document to the web server.
  - **Response:** The time it takes to receive the response from the web server, including the time it takes to download the document.
- **Document Processing Time:** The time it takes for the browser to finish parsing the HTML code of a web page and construct the Document Object Model (DOM). The DOM is a hierarchical representation of the page's elements, which can be manipulated by JavaScript.

  Factors Affecting Document Processing Time:
  - **HTML Code Size**: Larger HTML files take longer to parse.
  - **HTML Code Complexity**: More complex HTML structures require additional parsing time.
  - **Browser Performance**: Faster browsers can parse HTML more quickly.
  - **JavaScript Presence**: JavaScript files can increase document processing time.

  Tips for Improving Document Processing Time:
  - **Use minified and compressed HTML code-** Minified and compressed HTML code is smaller, which means it will take less time to parse.
  - **Use a CDN-** A CDN (Content Delivery Network) can deliver HTML code from servers that are closer to the user, which can improve document processing time.
  - **Use a caching mechanism-** A caching mechanism can store HTML code that has already been parsed, which can reduce the need to parse it again.

    - **Optimize the JavaScript files-** Optimizing JavaScript files can improve their performance and reduce the document processing time.

- **Time to First Byte (TTFB)-** A metric that measures the time it takes for a web browser to receive the first byte of a response from a web server. This includes the time it takes to:
    - **DNS lookup-** The time it takes to resolve the domain name of the web server.
    - **TCP connection establishment-** The time it takes to establish a connection to the web server.
    - **TLS handshake-** The time it takes to establish a secure connection to the web server, if the request is made over HTTPS.
    - **Request-** The time it takes to send the request for the resource to the web server.
- **DOM Ready Time:**

  The duration it takes for a web browser to parse the HTML code and build the Document Object Model (DOM) of a web page. The DOM represents the page's elements in a hierarchical structure, which can be manipulated by JavaScript. The DOM Ready Time event is triggered when the DOM is ready, though the page may still be loading images, CSS, and other resources. This metric indicates how quickly a page becomes interactive.

  Factors affecting DOM Ready Time include HTML code size and complexity, the user's internet speed, and browser performance.

  Improving DOM Ready Time can enhance user experience by making the page interactive more quickly. Here are some tips:

    - Minify and compress HTML, CSS, and JavaScript files.
    - Reduce the number of large images or resources.
    - Utilize a Content Delivery Network (CDN) for static resources.
    - Implement caching to store frequently accessed resources.
- **Network Time-** The duration for a web browser to retrieve all resources required to load a web page, including:
    - **DNS lookup-** Time to resolve the domain name of the resources.
    - **TCP connection establishment-** Time to set up a connection with the resource servers.
    - **TLS handshake-** Time to establish a secure connection for HTTPS resources.
    - **Request-** Time to send requests to the servers for the resources.
    - **Response-** Time to receive responses and download the resources.
- **Rendering Time-** The time needed for a web browser to display a web page, including:
    - **Parse the HTML code:** Time to parse HTML code and build the DOM.
    - **Load the CSS and JavaScript files:** Time to load CSS and JavaScript files for styling and interaction.
    - **Render the images and other resources:** Time to render images and other resources used on the page.

There are a several reasons why DOM ready time might be high while document processing time can remains low.

- **The HTML code is large and complex.** A large and complex HTML code can take longer to parse, even if it does not contain any JavaScript. This is because the browser has to process more information to create the DOM.
- **The web page uses a lot of third-party resources.** Third-party resources, such as images, CSS, and JavaScript files, can also add to the DOM ready time. This is because the browser has to wait for these resources to load before it can create the DOM.
- **The web page is using a slow web server.** If the web server is slow, it can take longer for the browser to download the HTML code and third-party resources. This can also increase the DOM ready time.

If your DOM ready time is high, but your document processing time is low, then you may need to optimize your HTML code to make it smaller and easier to parse.

If you access Realtime Transaction and double click on any URL you may encounter a message "Trace not available..."

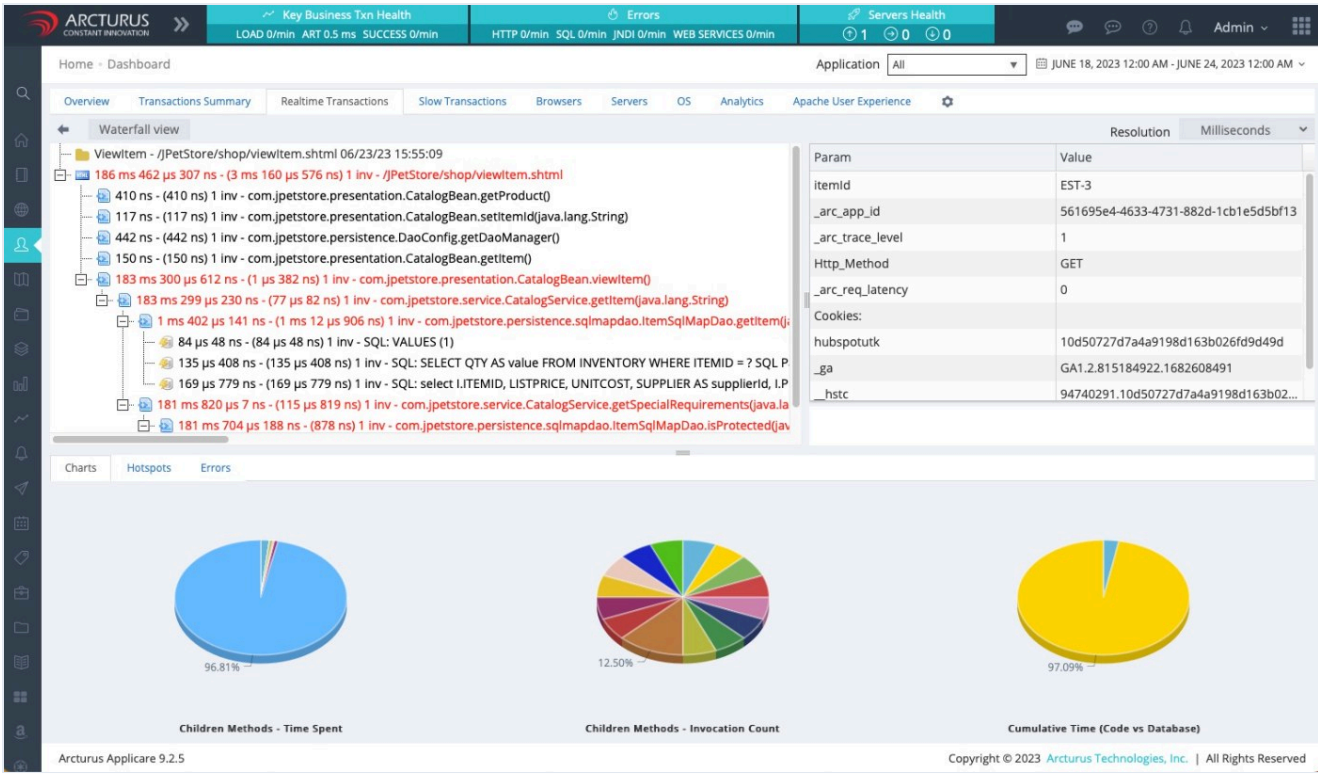
Activating tracing on End User Experience

1. Navigate to **Server Analyzer > IntelliTrace > HTTP**
2. Right-click the URL you want to trace in detail tracing and select "Add as a Key Business Transaction".
3. Add the URL to Key Business Transaction. Alternatively, go to directly visit **Business Transactions > Configure Key Business Transactions** and include the URL to Business Transaction Monitoring.
4. Go to **Profiling > Configure Business Transactions > Transaction Monitoring**.
5. Activate Per-Transaction Monitoring.
6. Activate Transaction Tracing.

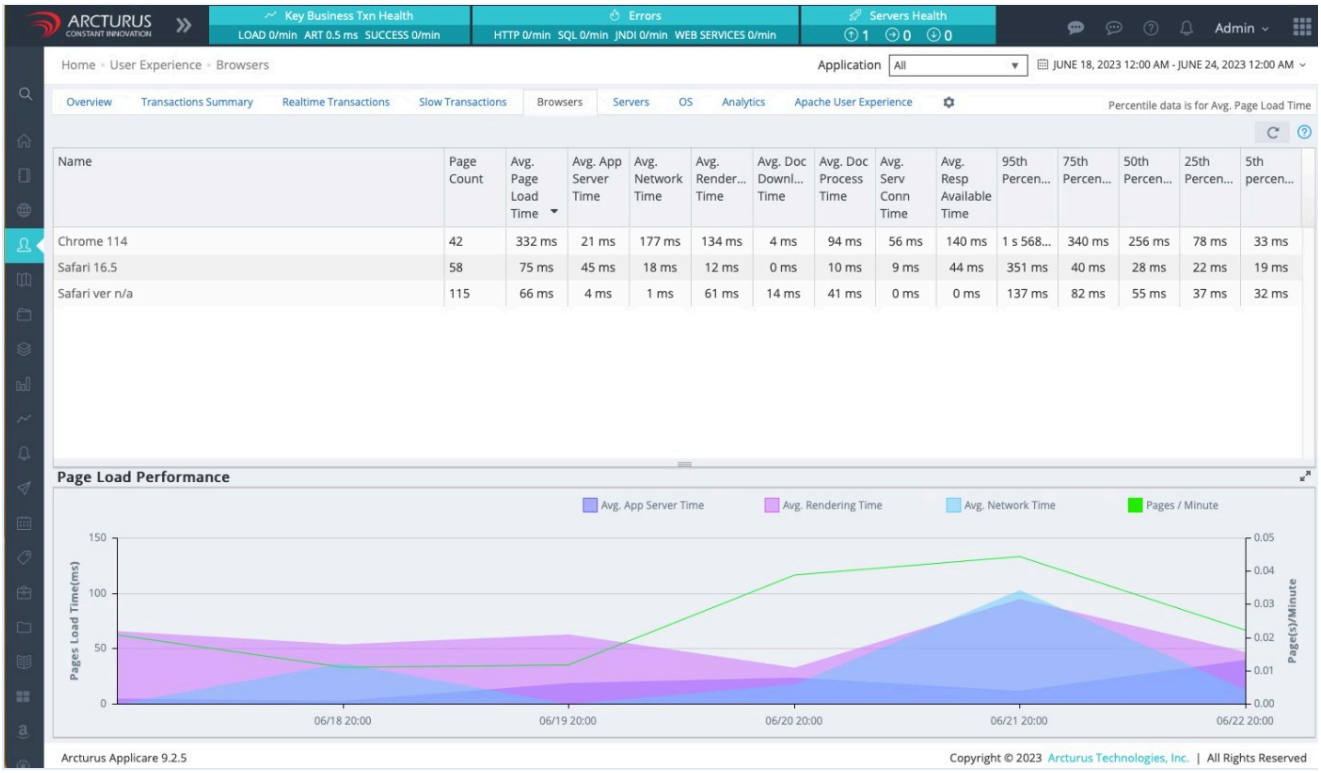
Once you have followed the above steps, restart the agent jvm.

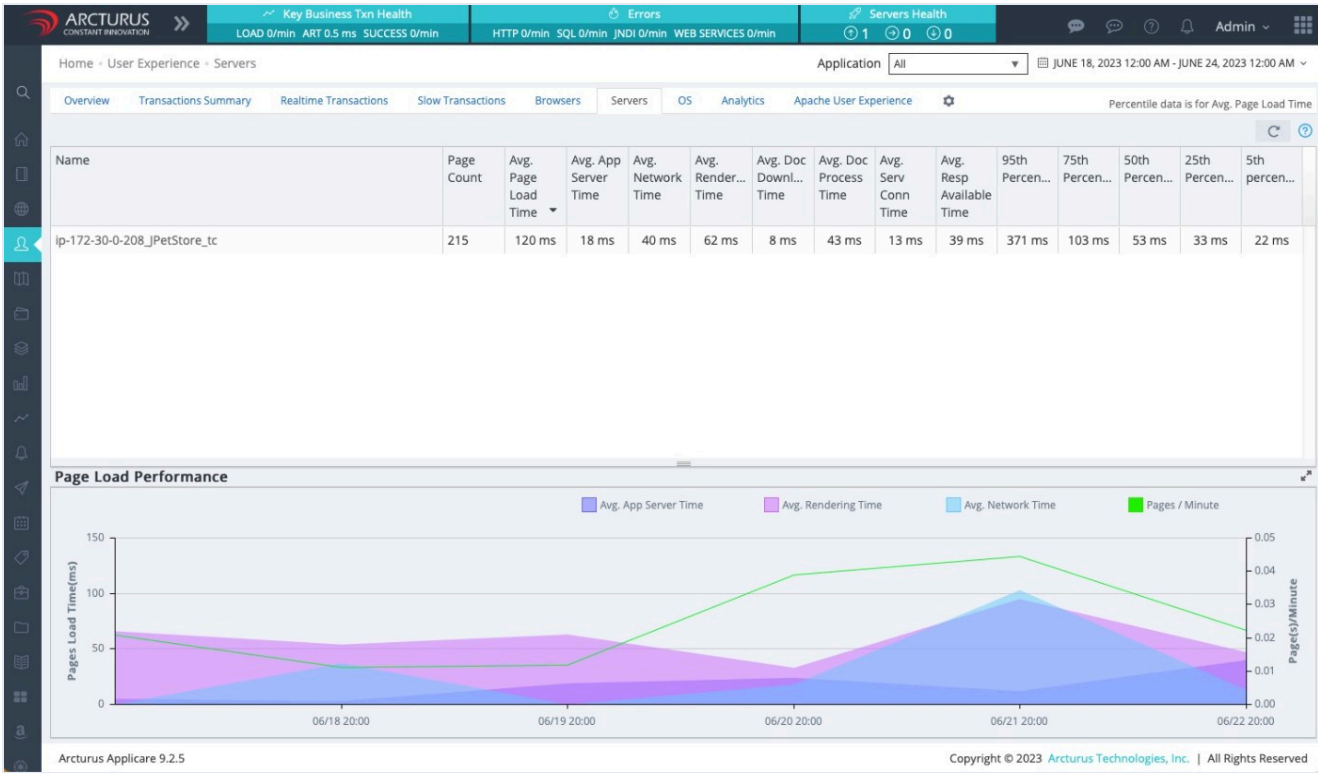
Now you can visit the UEM > End User Experience Analyzer > Realtime Transactions and double click on the URL.

If you have added the URL as KBT, detailed trace will be displayed as shown in the following image.



Browser and Backend Server Views





The Browsers and Servers tabs are similar to the Transaction Summary view but provide performance data segmented by client browser type and backend servers handling transactions.

End Client Processes Monitoring

End Client Process Monitoring tracks the running processes and resources utilized by each process on the end client machine where browser is running. This helps identify if the slowness is caused by some process running on End User's computer. End Client Process Monitoring is only supported on Windows and Internet Explorer (IE) end clients. End Client Process Monitoring requires permissions from the end user (or centrally configured by administrators) to allow running 'Microsoft Script Runtime'. If these permissions are not centrally configured then users are prompted on first visit to the pages with end client process monitoring enabled.

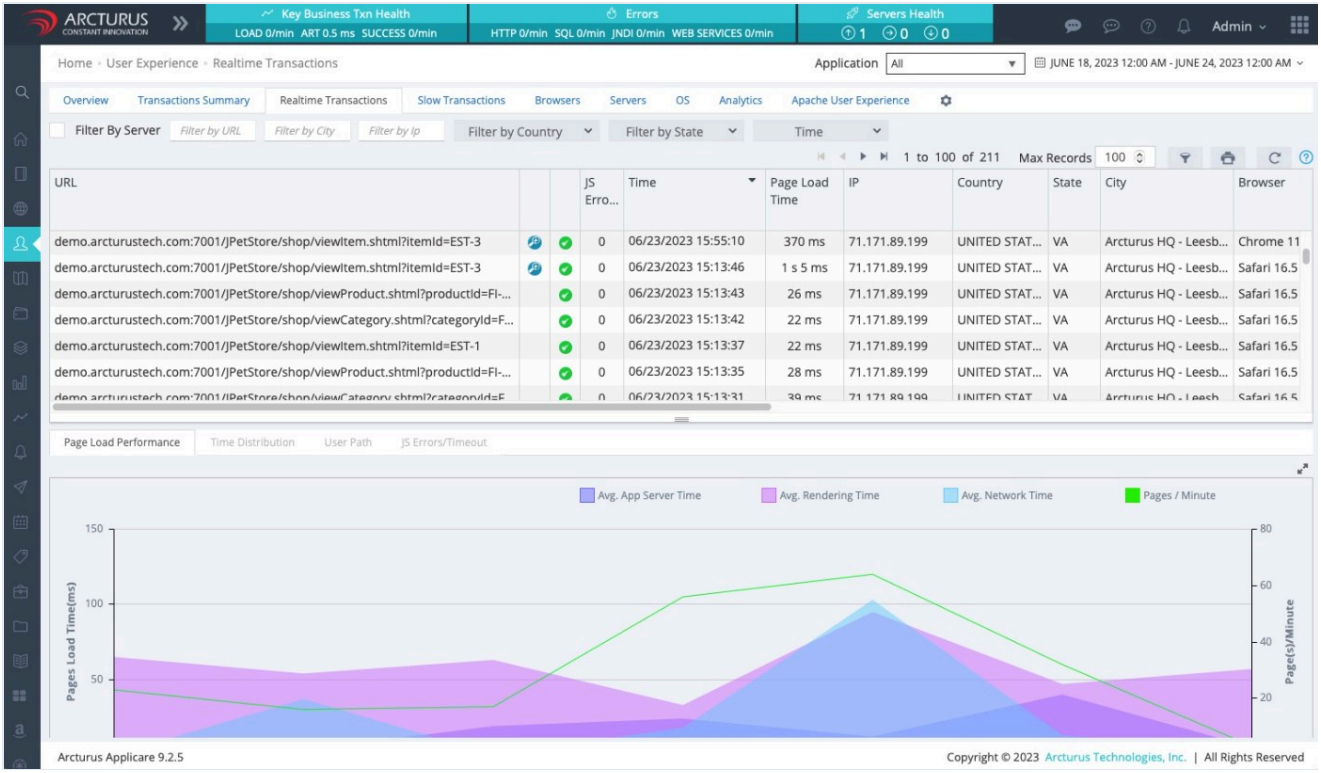
End Client Process Monitoring can be enabled for slow calls or all calls.

USING APPLICARE › USER EXPERIENCE MONITORING

EUM Real-Time Transaction Monitoring

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

The real-time view shows the last n number of individual transaction recorded. Along with performance stats you can view location and browser information of users accessing your application. This view doesn't show any transaction older than a day.



If the Applicare Controller/Collector gathering EUM data is situated a proxy, firewall or load balancer, ensure that these intermediaries forward client requests with the "X-Forwarded-For" header parameter. If this parameter is not included in the HTTP request then Applicare may show the IP address of proxy, firewall or load balancing server in End User Experience Analyzer.

Below are the details on monitored data shown for each transaction

- **Resource Fetch Time:** The duration required for a web browser to retrieve a resource, such as an image, CSS file, or JavaScript file. This includes the time it takes to:
  - **DNS lookup:** The time required to resolve the domain name of the resource.
  - **TCP connection establishment:** The time required to establish a connection to the server hosting the resource.
  - **TLS handshake:** The time required to establish a secure connection with the server, if the resource is hosted over HTTPS.
  - **Request:** The time required to send the request for the resource to the server.

- **Response:** The time it takes to receive a response from the server, including duration required to download the resource is known as Resource Fetch Time. This metric is crucial for evaluating web page performance, as a shorter Resource Fetch Time leads to faster page loading and enhance user experience.

  Several factors can influence Resource Fetch Time, including:

  - **Resource Size:** Larger resources will take longer to fetch.
  - **Internet Connection Speed:** A slower internet connection will lead to a longer Resource Fetch Time.
  - **Browser Performance:** A faster browser can retrieve resources more quickly.
  - **Server Location:** The closer the server is to the user, the shorter the Resource Fetch Time will be.

  Here are some tips for improving Resource Fetch Time:

  - **Use Minified and Compressed Resources:** Minified and compressed resources are smaller, resulting in fast fetch time.
  - **Use a CDN:** A CDN (Content Delivery Network) can deliver resources from servers that are closer to the user, which can improve Resource Fetch Time.
  - **Implement Caching:** A caching mechanism can store resources that have already been retrieved, reducing need to fetch them again.

- **Document Download Time:** The duration required to download a document from a web server to a user's device. This includes the time it takes to:

  - **DNS Lookup:** The time needed to resolve the domain name of the web server.
  - **TCP Connection Establishment:** The time required to establish a connection to the web server.
  - **TLS Handshake:** The time required to establish a secure connection with the web server, if the request is made over HTTPS.
  - **Request:** The time required to send the request for the document to the web server.
  - **Response:** The time needed to receive the response from the web server, including the time required to download the document.

- **Document Processing Time:** The time it takes for the browser to finish parsing the HTML code of a web page and construct the Document Object Model (DOM). The DOM is a hierarchical representation of the page's elements, which can be manipulated by JavaScript.

  Document processing time can be affected by a number of factors, including:

  - **Size of the HTML Code:** Larger HTML files will take longer to parse.
  - **Complexity of the HTML Code:** More complex HTML files will take longer to parse.
  - **Browser Performance:** Faster browsers will be able to parse HTML code more efficiently.
  - **The presence of JavaScript:** JavaScript files can increase the document processing time.

  Here are some tips for improving document processing time:

  - **Use Minified and Compressed HTML Code:** Minified and compressed HTML code is smaller, which means it will take less time to parse.

- **Use a CDN:** A CDN (Content Delivery Network) can deliver HTML code from servers that are closer to the user, which can improve document processing time.
- **Implement Caching:** A caching mechanism can store HTML code that has already been parsed, which can reduce the need to parse it again.
- **Optimize JavaScript files:** Optimizing JavaScript files can improve their performance and reduce the document processing time.

- **Time to First Byte (TTFB):** A metric that measures the time it takes for a web browser to receive the first byte of a response from a web server. This includes the time it takes to:
  - **DNS lookup:** The time it takes to resolve the domain name of the web server.
  - **TCP connection establishment:** The time it takes to establish a connection to the web server.
  - **TLS handshake:** The time it takes to establish a secure connection to the web server, if the request is made over HTTPS.
  - **Request:** The time it takes to send the request for the resource to the web server.
- **DOM Ready Time:** The time it takes for a web browser to finish parsing the HTML code of a web page and construct the Document Object Model (DOM). The DOM is a hierarchical representation of the page's elements, which can be manipulated by Javascript.

  The DOM Ready Time event is fired when the DOM is ready, but the page is not necessarily fully loaded yet. This means that images, CSS, and other resources may still be loading. However, the DOM Ready Time event is a good indication of how long it takes for a web page to become interactive.

  There are a number of factors that can affect DOM Ready Time, including the size and complexity of the HTML code, the speed of the user's internet connection, and the performance of the browser.

  DOM Ready Time is an important metric for measuring the performance of a web page. A short DOM Ready Time means that the page will be interactive more quickly, which can improve the user experience.

  Here are some tips for improving DOM Ready Time:
  - Use minified and compressed HTML, CSS, and JavaScript files.
  - Avoid using too many images or other large resources.
  - Use a CDN to deliver static resources.
  - Use a caching mechanism to store frequently accessed resources.
- **Network Time:** The time it takes for a web browser to fetch all of the resources that are needed to load a web page. This includes the time it takes to:
  - **DNS lookup:** The time it takes to resolve the domain name of the resources.
  - **TCP Connection Establishment:** The time it takes to establish a connection to the servers that host the resources.
  - **TLS Handshake:** The time required to establish a secure connection to the servers, if the resources are hosted over HTTPS.
  - **Request:** The time it takes to send the requests for the resources to the servers.
  - **Response:** The time it takes to receive the responses from the servers, including the time it takes to download the resources.

- **Rendering Time:** The time it takes for a web browser to render a web page. This includes the time it takes to:
  - **Parsing the HTML Code:** The time it takes to parse the HTML code of the web page and construct the DOM.
  - **Loading CSS and JavaScript Files:** The time it takes to load the CSS and JavaScript files that are needed to style and interact with the web page.
  - **Rendering Images and Other Resources:** The time required to render the images and other resources that are used on the web page.

Several factors can contribute to high DOM ready time despite a low document processing time.

- **Large and Complex HTML Code:** A large and complex HTML code can take longer to parse, even if it does not contain any JavaScript. This is because the browser has to process more information to create the DOM.
- **Heavy Use of Third-Party Resources:** Third-party resources, such as images, CSS, and JavaScript files, can also add to the DOM ready time. This is because the browser has to wait for these resources to load before it can create the DOM.
- **Slow Web Server:** If the web server is slow, it can take longer for the browser to download the HTML code and third-party resources. This can also increase the DOM ready time.

If you notice a high DOM ready but a low document processing time, consider optimizing your HTML code to reduce its size and complexity, making it easier and quick to parse.

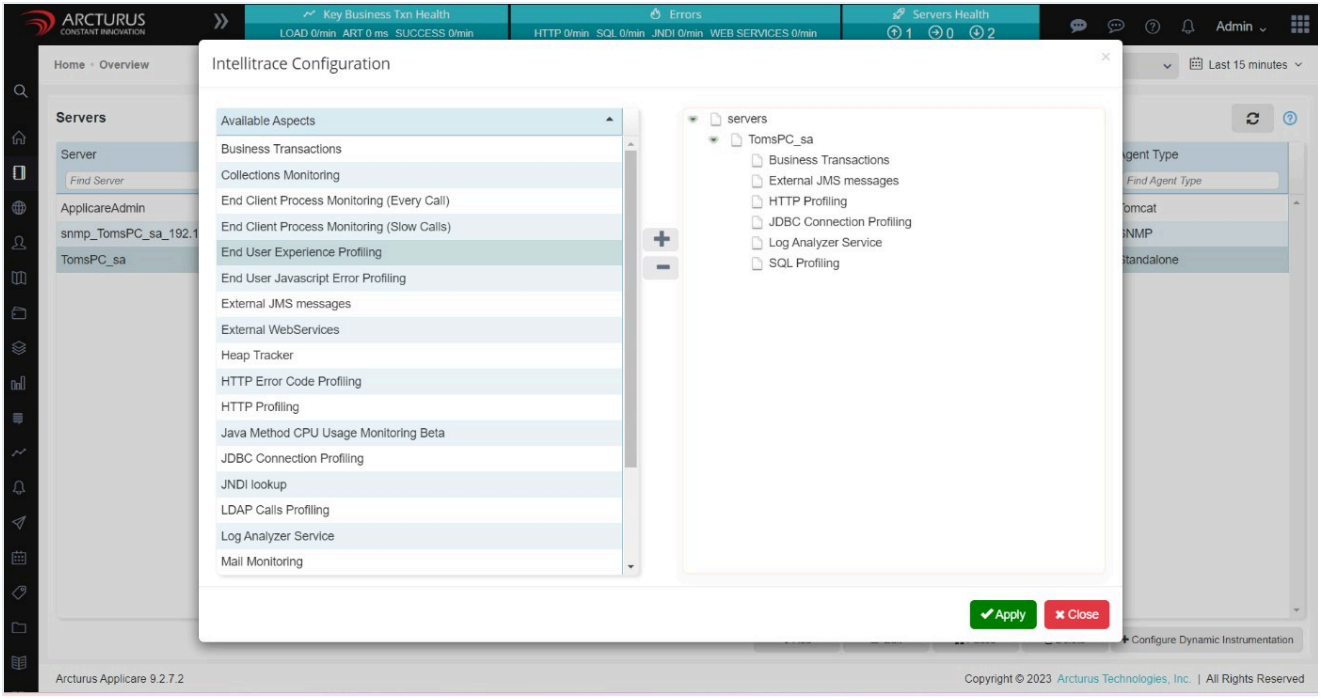
Monitoring with Applicare

USING APPLICARE › USER EXPERIENCE MONITORING

Applicare RUM/EUM Integration Script

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

In Applicare, you can activate end user monitoring by turning on the "End User Experience Profiling" feature from the overview screen. Once this feature is enabled, Applicare will automatically inject its RUM script into the web pages of the monitored applications behind the scenes.



Alternatively, you can manually inject this script into your application's web pages and redeploy the code to collect EUM data. Refer this link for Manual Applicare EUM script injection.

Applicare will inject two JavaScripts files named **arc-eum.js** and **arc-nav.js**.

The **arc-nav.js** script utilizes the Navigation API to measure various parameters including Document download time, Document processing time, Server connection time, Response availability time, Rendering time, DOM readiness time and Resource fetch time.

The **arc-eum.js** script employs BoomerangJS, an open-source middleware, to measure Page load time, Network time and browser details.

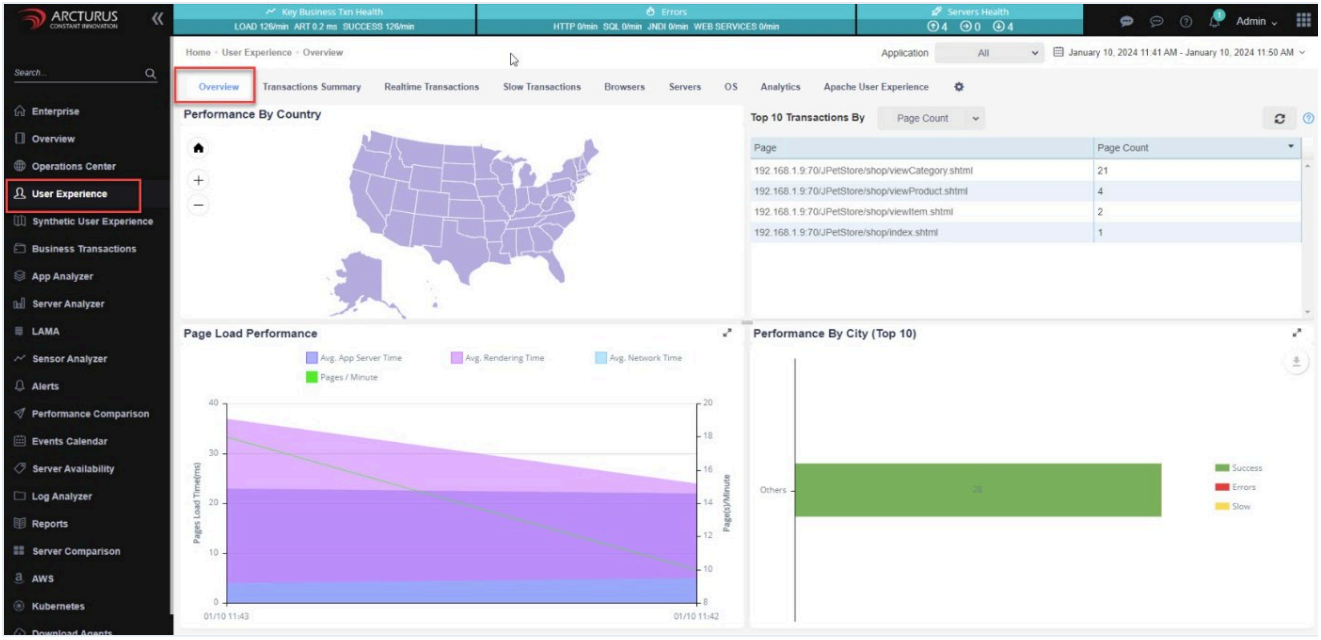
Monitoring with Applicare

USING APPLICARE › USER EXPERIENCE MONITORING

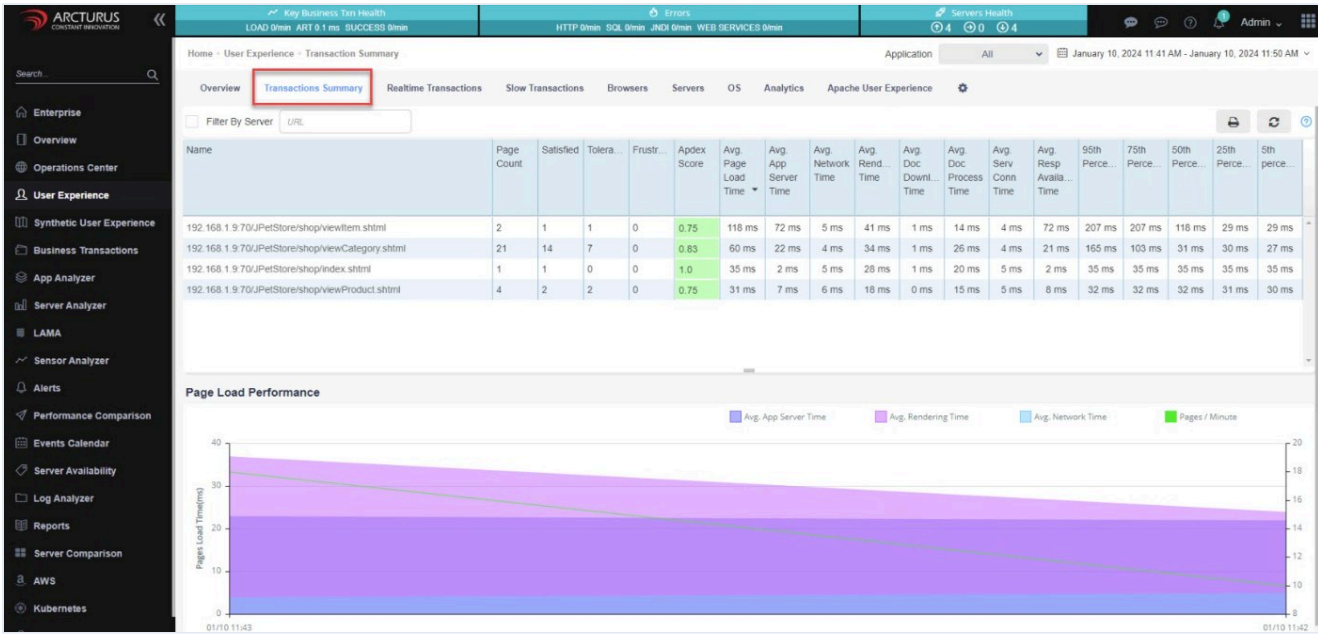
UEM Injection through the Applicare Reverse Proxy Server

Applicare 9.2 · Updated April 1, 2025 · **Pending technical review**

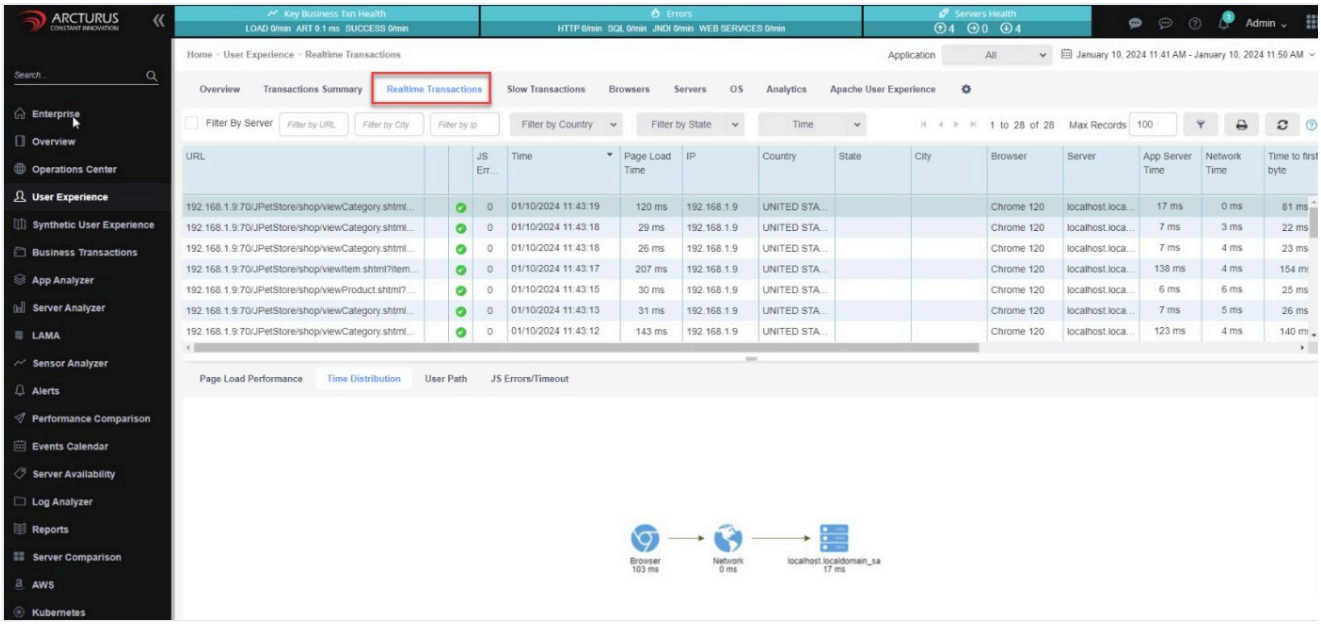
User Experience: An Overview



User Experience: Transaction Overview



User Experience: Real-Time Transactions



Steps to configure the Applicare reverse proxy server

Step 1: Download the Applicare Reverse Proxy Server from this link **download here.**

Step 2: Navigate to the path **<applicare_reverseProxy>/conf** and modify the **reverseproxy.conf** file.

Step 3: Change the **listen port** and **server_name** value inside the **server {** block.

listen -> Applicare reverse proxy server PORT

server_name -> Applicare reverse proxy server IP

```
server {
    listen       80;
    server_name  localhost;

    #charset koi8-r;

    #access_log  logs/host.access.log  main;

    location / {
        root   html;
        index  index.html index.htm;
    }
```

e.g.



Step 4: Add the **upstream** configuration before the **server {** block

Syntax:

```
TEXT
upstream UPSTREAM_REDIRECTOR_NAME {
    ip_hash;
    server REDIRECTOR_APPLICATION_IP:REDIRECTOR_APPLICATION_PORT;
}
```

Example:

```
TEXT
upstream petstoreredirect {
    ip_hash;
    server 192.168.0.165:8881;
}
```

Step 5: Add the new location block after the **location / {** block

Syntax:

```
TEXT
location /REDIRECTOR_APPLICATION_BASE_URL {
    proxy_pass REDIRECTOR_APPLICATION_PROTCOL://UPSTREAM_REDIRECTOR_NAME;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    sub_filter '</body>' '</body>new_string';
    sub_filter_once off;
}
```

Step 6: Replace the **REDIRECTOR_APPLICATION_BASE_URL** with the application base url.

REDIRECTOR_APPLICATION_BASE_URL -> JPetStore

```
TEXT
location /JPetStore {
```

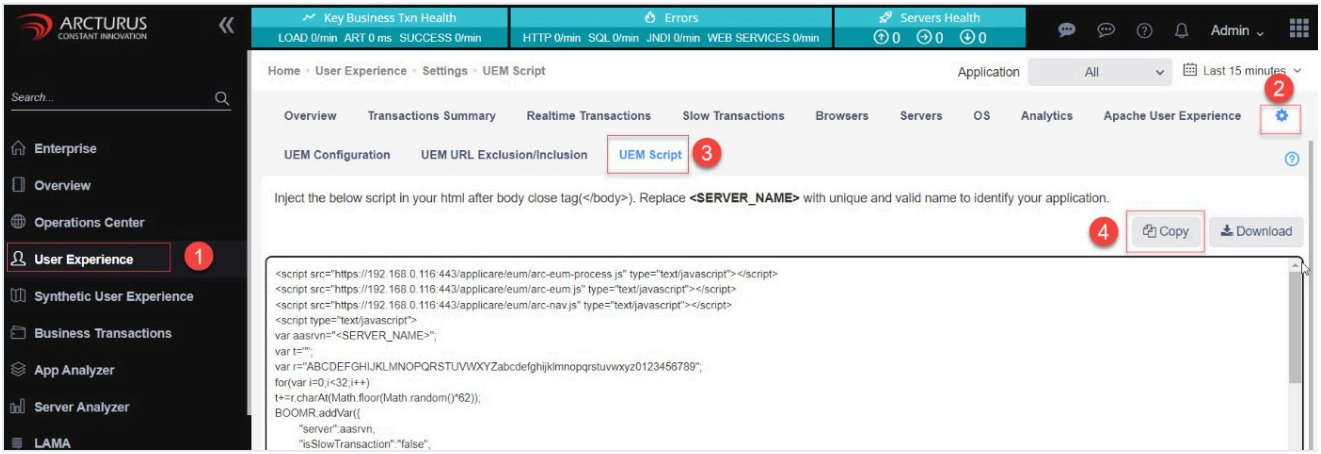
Step 7: Replace the **REDIRECTOR_APPLICATION_PROTCOL** and **UPSTREAM_REDIRECTOR_NAME**.

REDIRECTOR_APPLICATION_PROTCOL -> http

UPSTREAM_REDIRECTOR_NAME -> petstoreredirect

```
TEXT
proxy_pass http://petstoreredirect;
```

Step 8: Login to Applicare console and go to **User Experience -> Settings -> UEM Script** screen and **copy** the script and replace it in the place of **new_string** keyword in the sub_filter in the Nginx sever conf file.



The **conf file** will appear as shown below.

```
        location /JPetStore {
            proxy_pass http://petstoreredirect;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            sub_filter '</body>' '</body><script src="https://192.168.0.116:443/applicare/eum/arc-eum-process.js" type="text/javascript"></script>
<script src="https://192.168.0.116:443/applicare/eum/arc-eum.js" type="text/javascript"></script>
<script src="https://192.168.0.116:443/applicare/eum/arc-nav.js" type="text/javascript"></script>
<script type="text/javascript">
var aasrvn="<SERVER_NAME>";
var t="";
var r="ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
for(var i=0;i<32;i++)
t+=r.charAt(Math.floor(Math.random()*62));
BOOMR.addVar({
    "server":aasrvn,
    "isSlowTransaction":"false",
    "avgTime":"0.0",
    "collectProcessListFor":"all",
    "percentile":"0.0",
    "traceId":t,
    "t_backend":"$request_time" * 1000,
    "b_name":BrowserDetect.browser,
    "b_ver":BrowserDetect.version,
    "clientId":0,
    "b_os":BrowserDetect.OS,
    "code":"200",
    "manual":"true",
    "eumId":t});
BOOMR.init({user_ip:"127.0.0.1",beacon_url:"https://192.168.0.116:443/applicare/eum/beacon"});
BOOMR.plugins.RT.startTimer("t_page");
if(!(window.arcPops && window.arcPops.errorInitiated)){
var oldOnError=window.onerror;
window.arcPops={errorInitiated:true};
var xhttp = new XMLHttpRequest();
window.onerror = function(msg, url, lineNo, columnNo, error){
if(oldOnError)oldOnError.apply(this, arguments);
var url="https://192.168.0.116:443/applicare/eum/beacon?serverName="+aasrvn+"&msg="+msg+"&lineNo="+lineNo+"&columnNo="+columnNo+"&eumId="+t+"&jser
navigator.sendBeacon(url);return false;}}
var originalSetRequestHeader = XMLHttpRequest.prototype.setRequestHeader;
XMLHttpRequest.prototype.setRequestHeader = function(header, value) {
originalSetRequestHeader.apply(this, arguments);
};
(function(open){XMLHttpRequest.prototype.open = function(method, url, async, user, password){function handleEvent(e) {
var beaconURL = "https://192.168.0.116:443/applicare/eum/beacon?serverName="+aasrvn+"&msg=Request Timeout&lineNo=&columnNo=&eumId="+t+"&jserror=tr
navigator.sendBeacon(beaconURL);
}function handleHeaderEvent(e) {
if (this.readyState === 1) {
this.setRequestHeader("_arc_req_st_time", Date.now());
}
}
this.addEventListener("abort", handleEvent);this.addEventListener("timeout", handleEvent);this.addEventListener("readystatechange", handleHeaderEve
open.apply(this, arguments);};})(XMLHttpRequest.prototype.open)
</script>';
            sub_filter_once off;
```

Step 9: Substitute the following values.

a. Replace **<SERVER_NAME>** with a unique identifier for your application.

b. Set the value from **"true"** to **"false"**.

c. In the **"t_backend"** section, update the value from **"0"** to **"$request_time" * 1000**



Step 10: Navigate to the <**applicare_reverseProxy>** home directory and run the following command to start.

```bat
startreverseproxy.bat
```

Step 11: Access the application through the Reverse Proxy server.

Syntax:

```xml
<REVERSE_PROXY_PROTOCOL>://<REVERSE_PROXY_IP>:
<REVERSE_PROXY_PORT>/REDIRECTOR_APPLICATION_BASE_URL
```

e.g.

```text
http://192.168.0.113:83/JPetStore
```

Step 12: Navigate to the <**applicare_reverseProxy>** home directory and run the following command to stop the Reverse Proxy Server.

```bat
stopreverseproxy.bat
```

USING APPLICARE › USER EXPERIENCE MONITORING

Apache User Experience

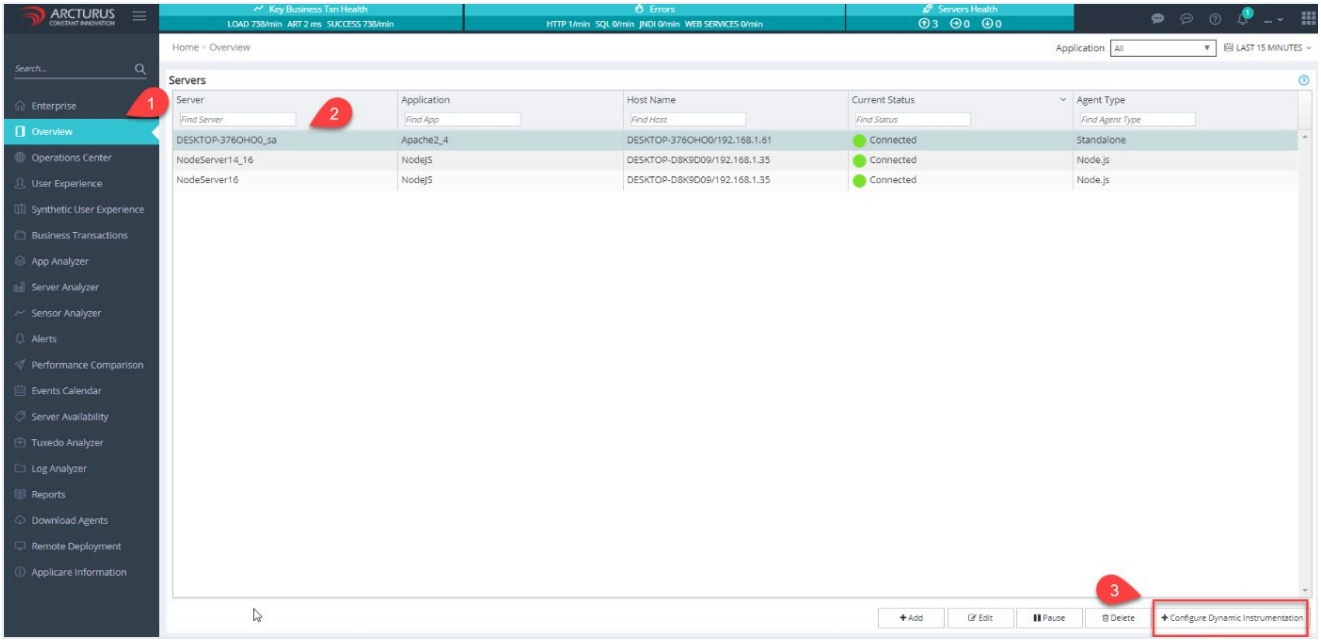
Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Enabling Apache User Experience Analyzer

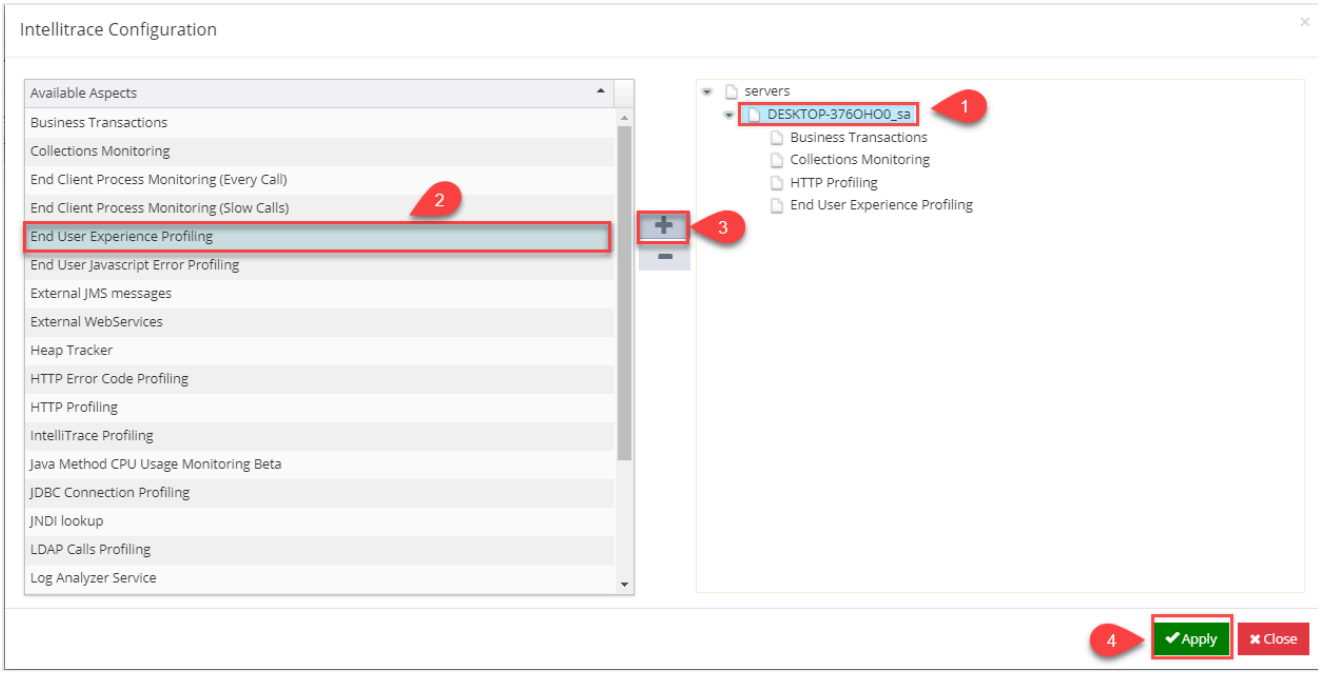
Enabling this feature will log each individual transaction separately into the database, resulting in a large volume of data being recorded.

To set up the Apache User Experience Analyzer, follow these steps:

1. Navigate to the **Overview** screen, select the **server,** and click on the **Configure Dynamic Instrumentation**.

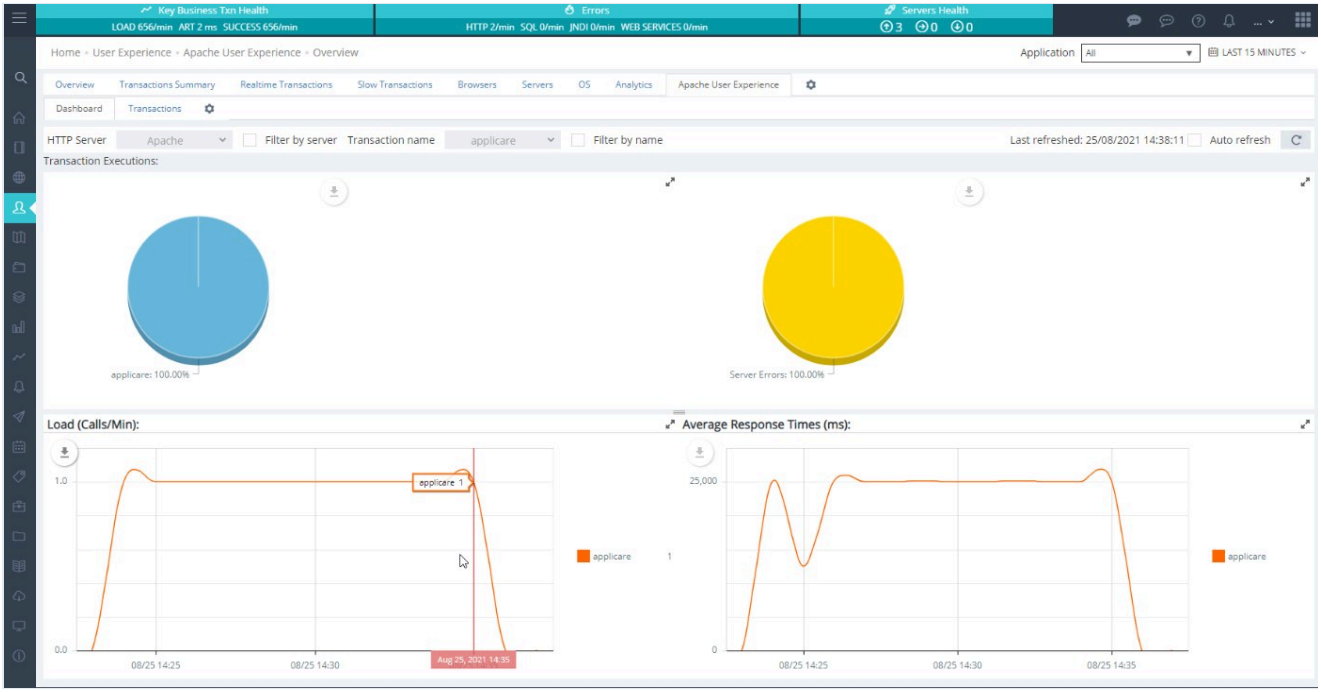


2. Add the "**End User Experience Profiling**" aspect to the server.

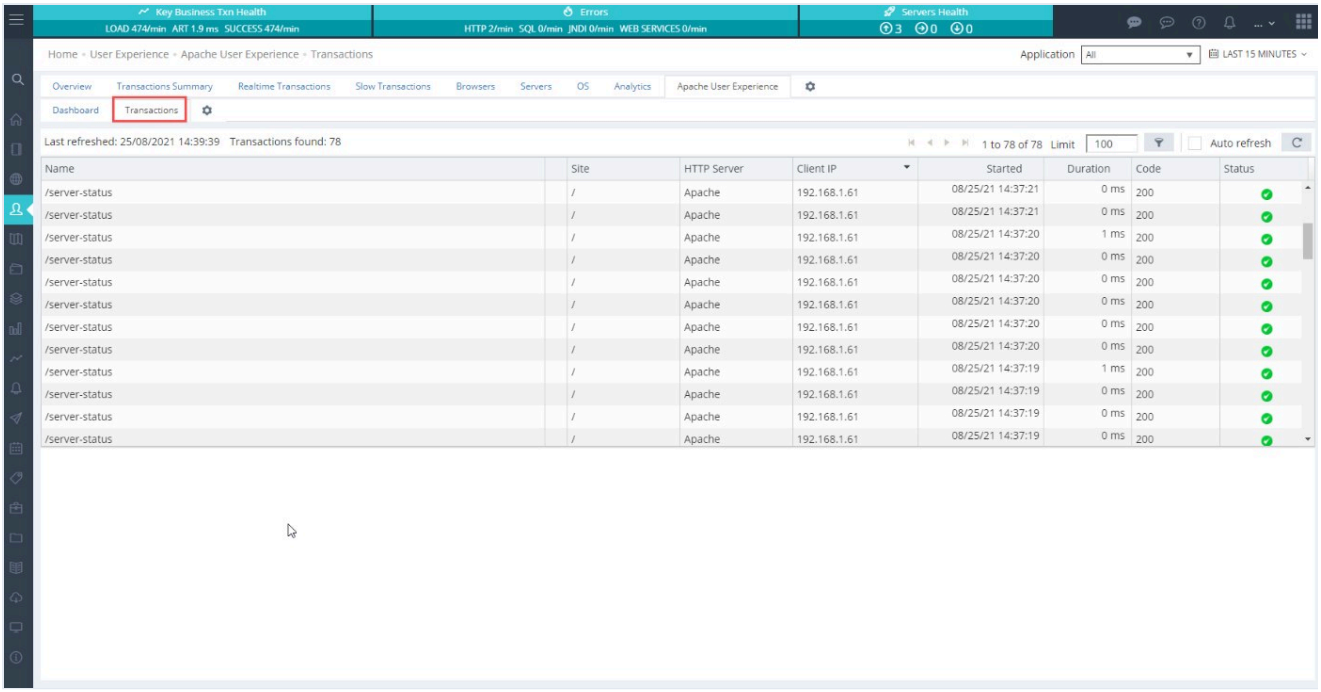


3. Within a few minutes, the data will appear in the "**Apache User Experience Analyzer**".

Dashboard

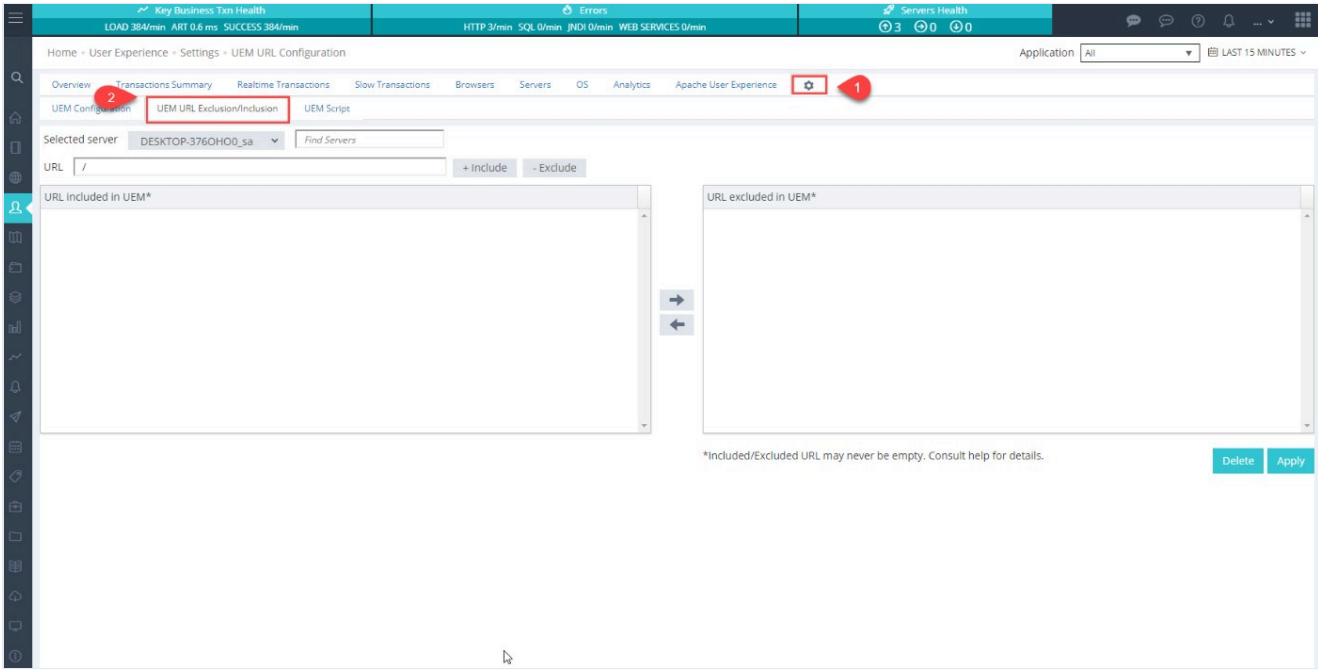


Transactions



To Restrict logging to specific URLs in the Apache User Experience Analyzer

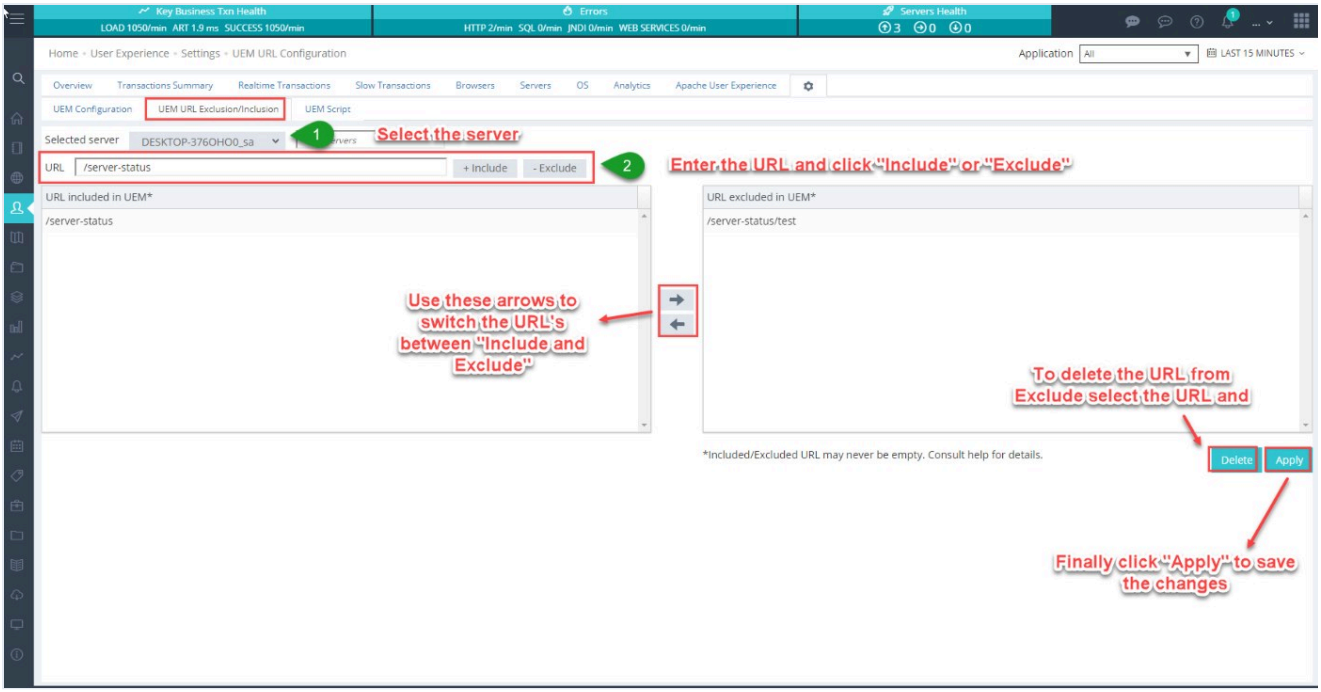
1. Navigate to **User Experience --> Settings (icon) --> UEM URL Exclusion/Inclusion**



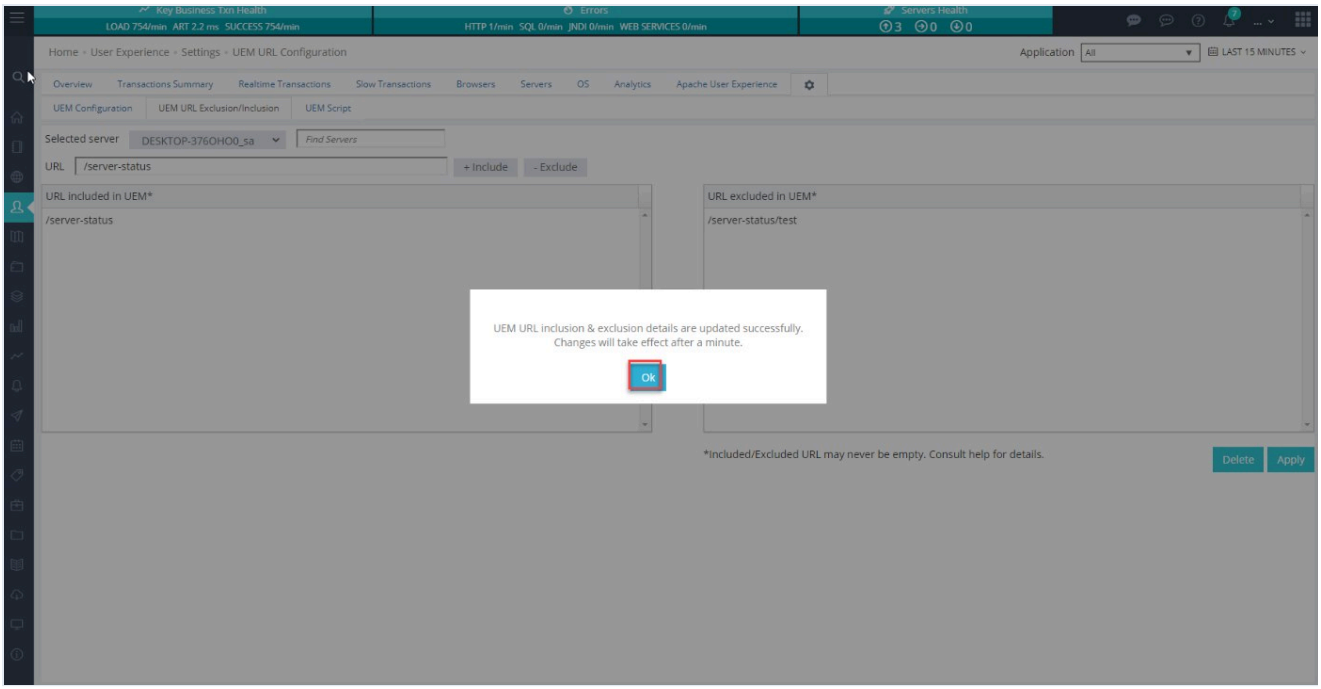
2. Add the URL's to the UEM Include and Exclude lists.

UEM Include & Exclude

If the UEM Include list is populated, Only the URLs specified in that list will be logged.



Click "OK"



After applying the changes, a restart of SingleAgent is required. Ensure you add all necessary URL's to the UEM Include or Exclude list, apply the changes, and then restart the SingleAgent.

Monitoring with Applicare

USING APPLICARE › USER EXPERIENCE MONITORING

UEM IP Address Translation

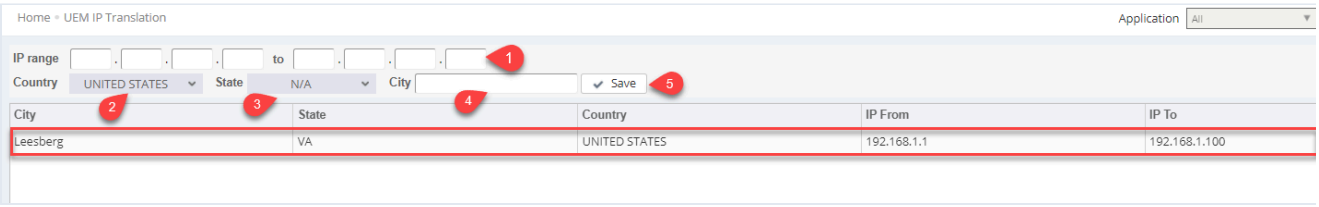
Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Overview

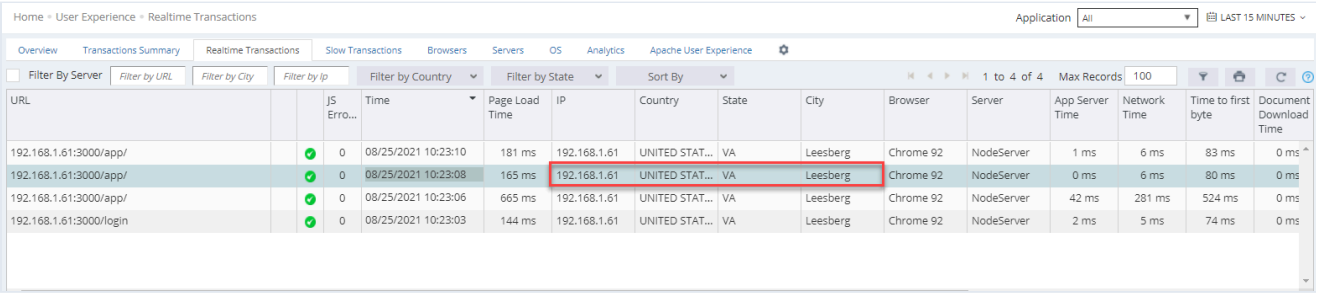
UEM IP translation configures the source of request that reach the application on the app server.

Configuration

Specify on the IP range, select country and state, enter the city name, and then click the save button to apply the configuration.



After configuring and saving, navigate to **User Experience-> Realtime Transactions** to view the transaction data associated country, state, and city.



Monitoring with Applicare

USING APPLICARE › USER EXPERIENCE MONITORING

iOS Mobile App Performance Monitoring

Applicare 9.2 · Updated August 23, 2024 · **Pending technical review**

The Applicare Mobile Application Tracker provides real-time monitoring of your mobile applications, including the ability to track business transactions. This powerful service offers features such as real-time request tracking, activity path tracking, geo-tracking, error reporting, and much more.

One of the advantages of mobile app development is the ability to quickly distribute new versions and ensure that all users are using the latest app. If you haven't been monitoring your code previously, you can easily implement Applicare Tracker, push the update, and start collecting metrics within minutes.

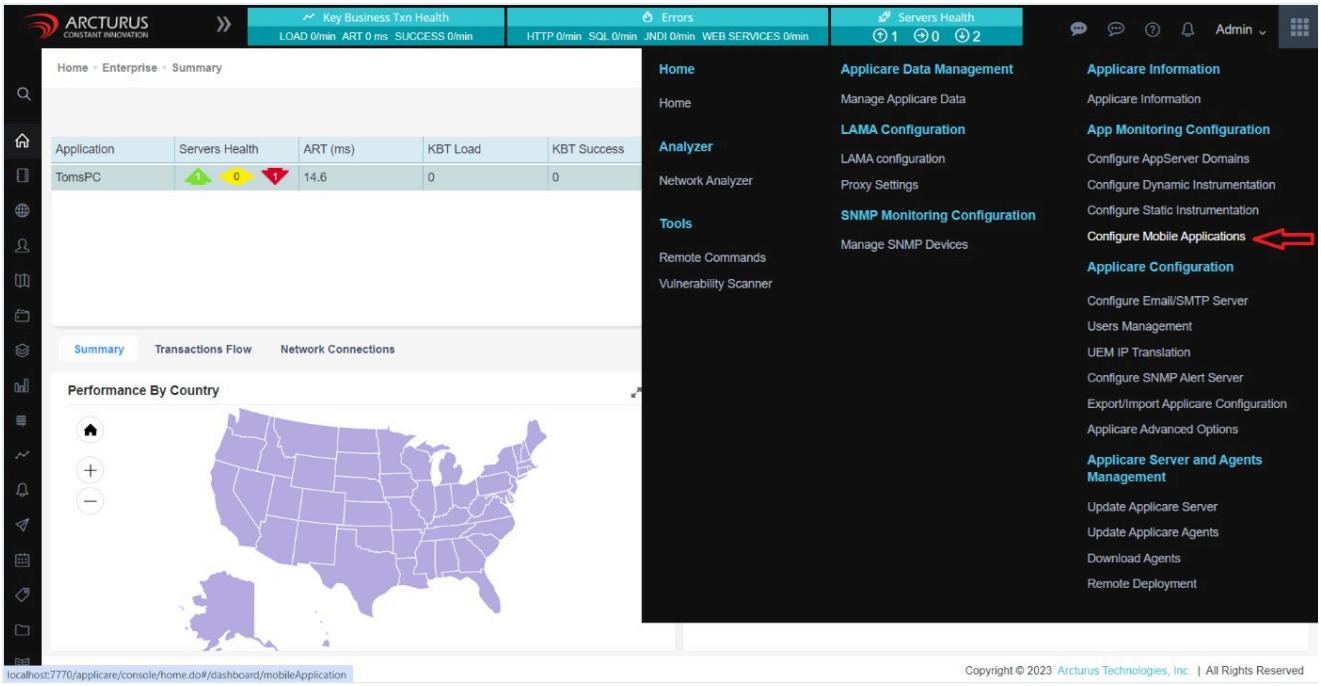
Why Use Applicare Mobile Application Tracker?

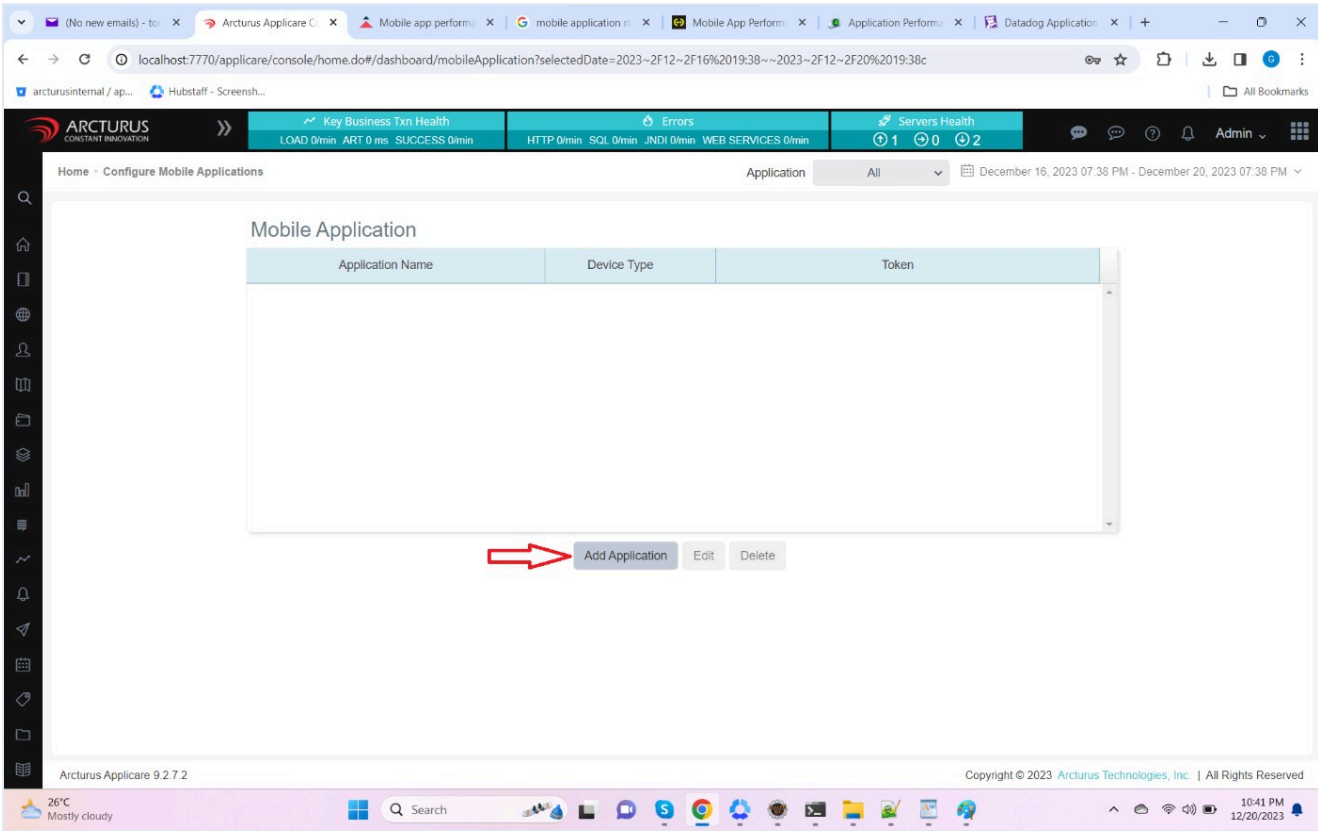
- Quick and straightforward setup process.
- Seamlessly integrates with any Android application.
- Provides insights into real user interactions.
- Capture performance metrics, set SLAs for Android method calls, and track individual transactions with performance statistics and call graphs.
- Gain insights into user characteristics.
- Visualize user locations on a map.
- Detect performance issues and server failures automatically.
- Receive regular reports to stay informed about performance issues.
- Examine detailed historical server data, including conditions before crashes.

Steps to Integrate the Applicare Mobile Tracking Agent

Register your Mobile App and Generate a Token

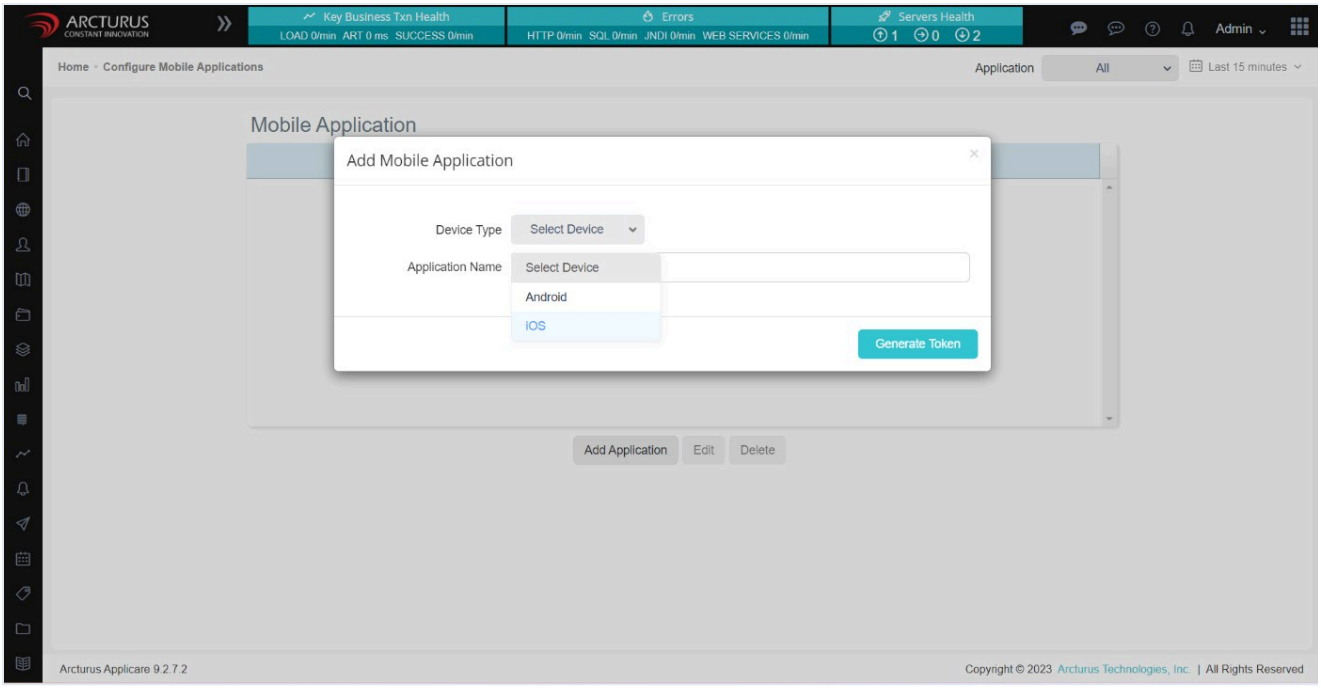
- Sign in to your Applicare Admin Console.
- Access the Menu in top right corner and navigate to the "Configure Mobile Application" screen.

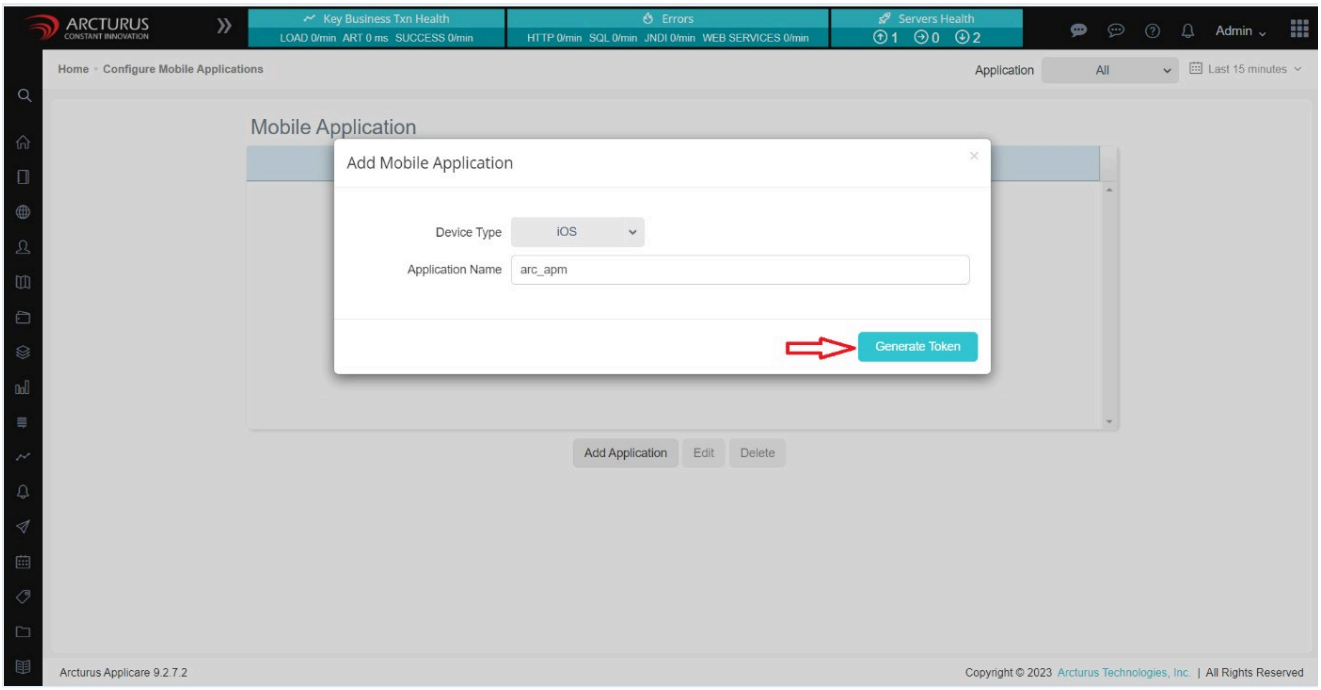


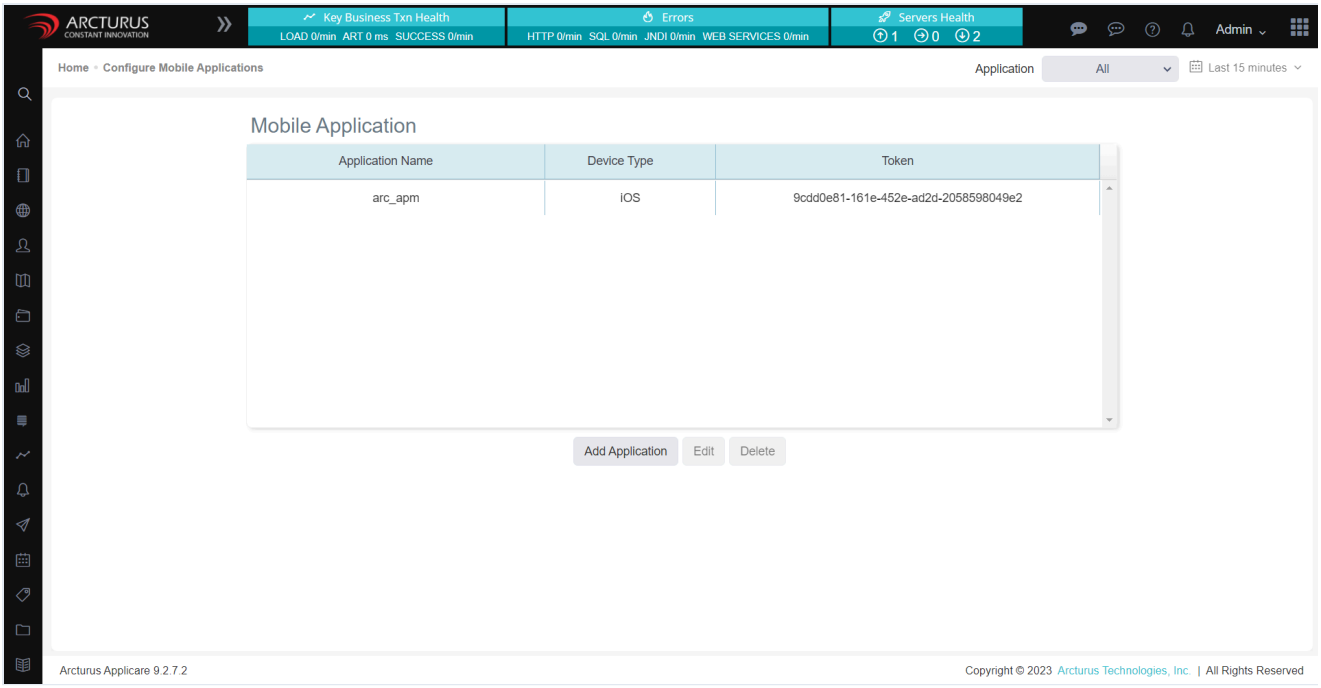


- On the "Configure Mobile Application" screen, select "iOS" from the dropdown menu. Enter your application's name and click the “Generate Token” button. A token will be generated; copy and save this token for future use.

  .


ARCTURUS
CONSTANT INNOVATION
Key Business Txn Health
LOAD 0/min ART 0 ms SUCCESS 0/min
Errors
HTTP 0/min SQL 0/min JNDI 0/min WEB SERVICES 0/min
Servers Health
1 0 2
Admin
Home • Configure Mobile Applications
Application All
Last 15 minutes
Mobile Application
Add Mobile Application
Device Type
Select Device
Application Name
Select Device
Android
iOS
Generate Token
Add Application
Edit
Delete
Arcturus Applicare 9.2.7.2
Copyright © 2023 Arcturus Technologies, Inc. | All Rights Reserved


ARCTURUS
CONSTANT INNOVATION
Key Business Txn Health
LOAD 0/min ART 0 ms SUCCESS 0/min
Errors
HTTP 0/min SQL 0/min JNDI 0/min WEB SERVICES 0/min
Servers Health
1 0 2
Admin
Home • Configure Mobile Applications
Application All
Last 15 minutes
Mobile Application
Add Mobile Application
Device Type
iOS
Application Name
arc_apm
Generate Token
Add Application
Edit
Delete
Arcturus Applicare 9.2.7.2
Copyright © 2023 Arcturus Technologies, Inc. | All Rights Reserved



Adding Applicare Agent to your app

Add the framework applicare_agent.framework file into your Objective-C or Swift project.

For Objective-C

- Inside the AppDelegate.m file import the header by adding the following line.

```
TEXT
#import <applicare_agent/ApplicationTracker.h>
```

- Now inside the didFinishLaunchingWithOptions method, add the following lines

If your app is using server add the below code.

```
TEXT

NSString *serverPort = YourServerPort;
NSString *serverIp = YourServerIp;
NSString *token = YourToken;
NSString *agentType = @"agent";
```

If your app is not using server, add the below code.

```
TEXT
NSString *serverPort = ApplicareServerPort;
NSString *serverIp = ApplicareServerIp;
NSString *token = YourToken;
NSString *agentType = @"none";
```

- Now initialize the framework by adding the following line.

```
TEXT
[ApplicationTracker initialize:serverPort arg2: serverIp arg3: token arg4: agentType];
```

- It should finally looks like below for apps with server.

```
18
19 - (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSD
20     // Override point for customization after application launch.
21     NSString *serverPort = @"8080";
22     NSString *serverIp = @"192.168.2.100";
23     NSString *token = @"4db94134-bc29-4eba-b972-038d0145ce87";
24     NSString *agentType = @"agent";
25     [ApplicationTracker initialize:serverPort arg2: serverIp arg3: token arg4: agent
26     return YES;
27 }
28
```

For Swift : Here are steps to add an Objective-C frame to a Swift project.

- **Import the Framework**: Drag and drop the Objective-C framework into your Xcode Swift project.
- **Create a New Objective-C File**: Go to File -> New -> File, and select Objective-C for iOS.
- **Create Bridging Header**: Accept the prompt to create a bridging header file between Objective-C and Swift.
- **Delete the Objective-C File**: You can now delete the newly created Objective-C file, but keep the bridging header file ${YOURPROJ}-Bridging-Header.h.
- **Import the Framework**: In the bridging header file, use the standard Objective-C import syntax to include your framework.

```
TEXT
#import <applicare_agent/ApplicationTracker.h>
```

- This approach eliminates the need to perform an ***import ApplicationTracker.h*** directly in your ***AppDelegate.swift*** file. By using the bridging header, you can access the framework across all Swift files in your project.
- Inside the didFinishLaunchingWithOptions method in the AppDelegate.swift file add the below lines.

If your app is not using server add the below lines.

```
TEXT
let serverPort = ApplicareServerPort
let serverIp = ApplicareServerIp
let token = YourToken
let agentType = "none"
```

If your app is using server then add the below lines.

```
TEXT
let serverPort = YourServerPort
let serverIp = YourServerIp
let token = YourToken
let agentType = "agent"
```

- Now initialize the framework by adding the following lines.

```
TEXT
ApplicationTracker.initialize(serverPort, arg2: serverIp, arg3: token, arg4: agentType)
```

- It should finally looks like below for apps without server.

```
func application(application: UIApplication, didFinishLaunchingWithOptions launchOptions: [NSObject
    // Override point for customization after application launch.
    let serverPort = "8880"
    let serverIp = "192.168.2.100"
    let token = "4db94134-bc29-4eba-b972-038d0145ce87"
    let agentType = "none"
    ApplicationTracker.initialize(serverPort, arg2: serverIp, arg3: token, arg4: agentType)
    return true;
}
```

USING APPLICARE › USER EXPERIENCE MONITORING

Monitoring performance for Android mobile apps

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

Applicare Mobile Application Tracker offers a solution for real-time monitoring of your mobile applications, with a distinctive feature for tracking live business transactions. The Applicare tracker is a robust service equipped with features like real-time request tracking, activity path tracking, geo-tracking and much more.

One of the advantage developing a mobile app is the ability to quickly distribute updates, ensuring that your entire user base is always using the latest version of your app. If you weren't previously monitoring your code, you can effortlessly implement Applicare tracker, deploy it, and begin collecting metrics within minutes.

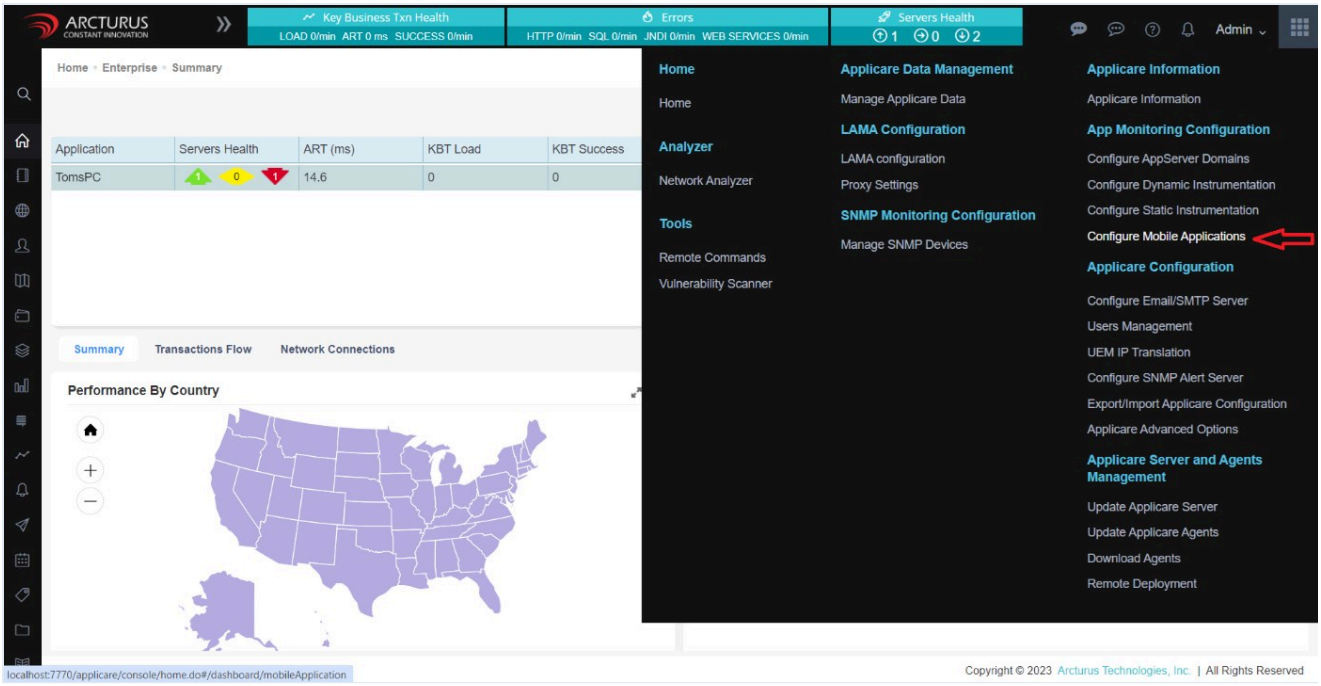
Why choose Applicare Mobile Application Tracker?

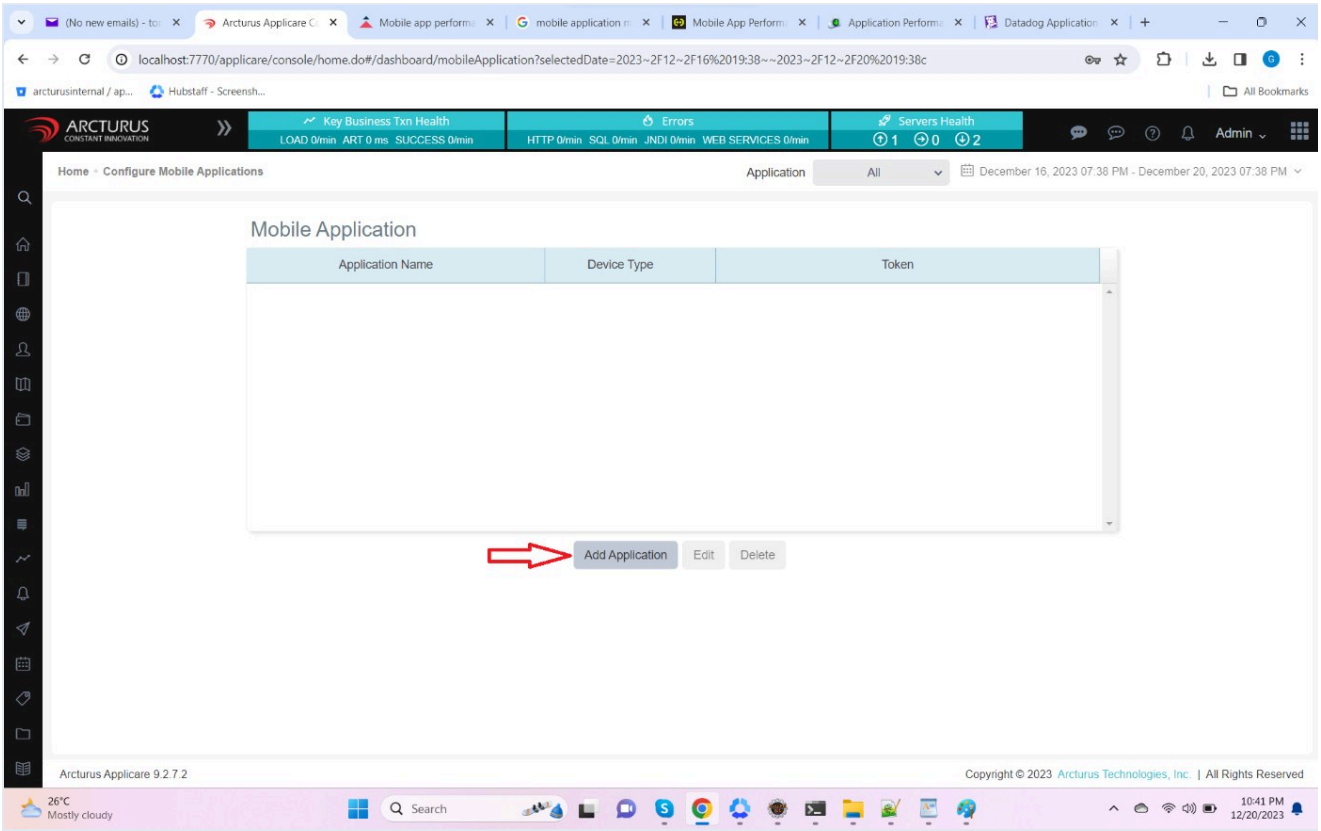
- Ease of installation.
- Easily integrates with any Android application.
- Track real user interactions for comprehensive insights.
- Business Transactions features allows you to capture performance statistics and optionally set SLAs for Android methods calls and track individual execution of transactions with performance stats and call graphs.
- Demographic characteristics of users.
- Visualize user locations.
- Auto detect the root cause of performance degradation and server failure.
- Generate scheduled reports to keep you aware of performance issues.
- Analyze detailed historical server information, even pre-crash conditions.

Steps to Integrate Applicare mobile tracking agent

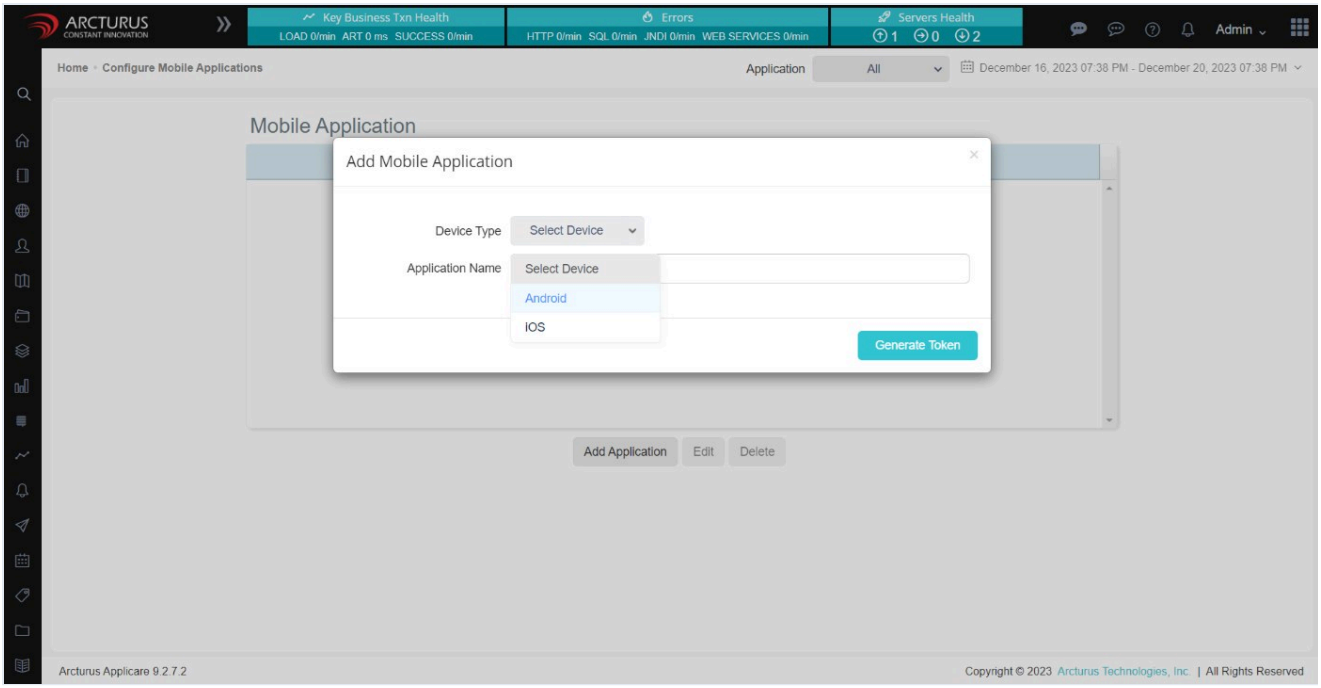
Generate a Token and Download the Applicare Android SDK Agent

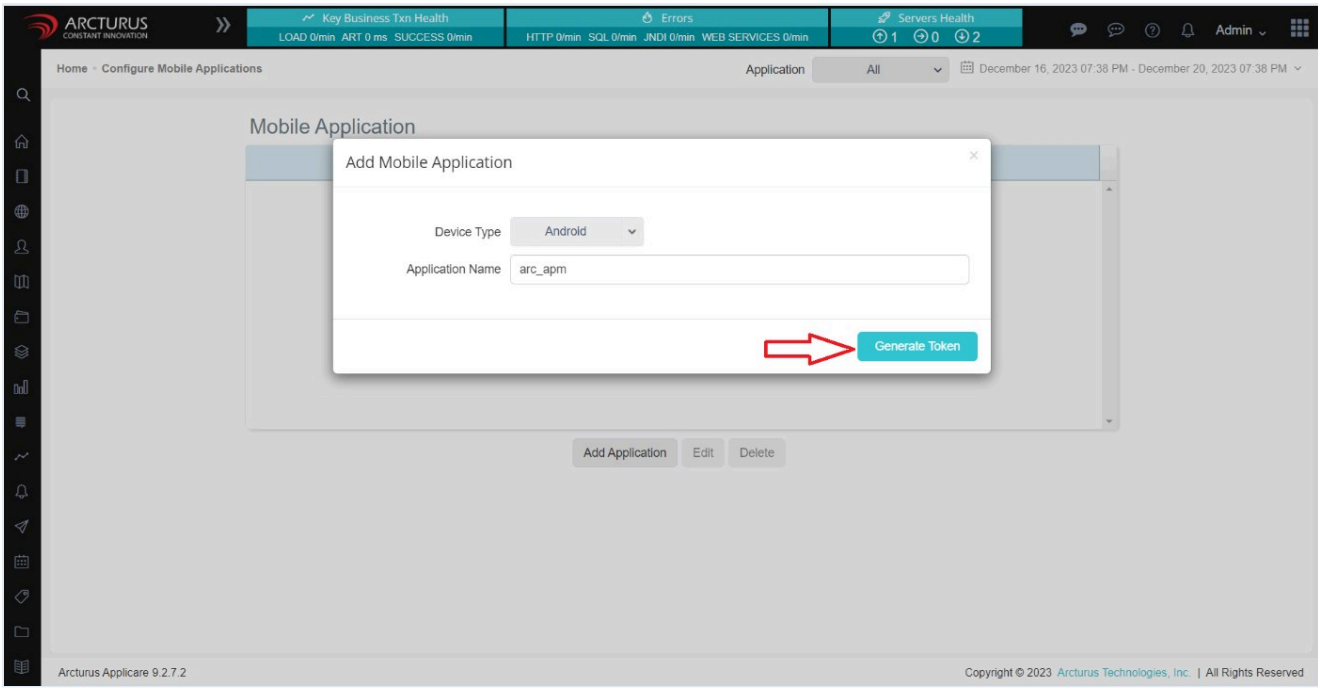
- Log in to your Applicare Admin Console.
- Access the menu in the top right corner and navigate to the "Configure Mobile Application" screen.

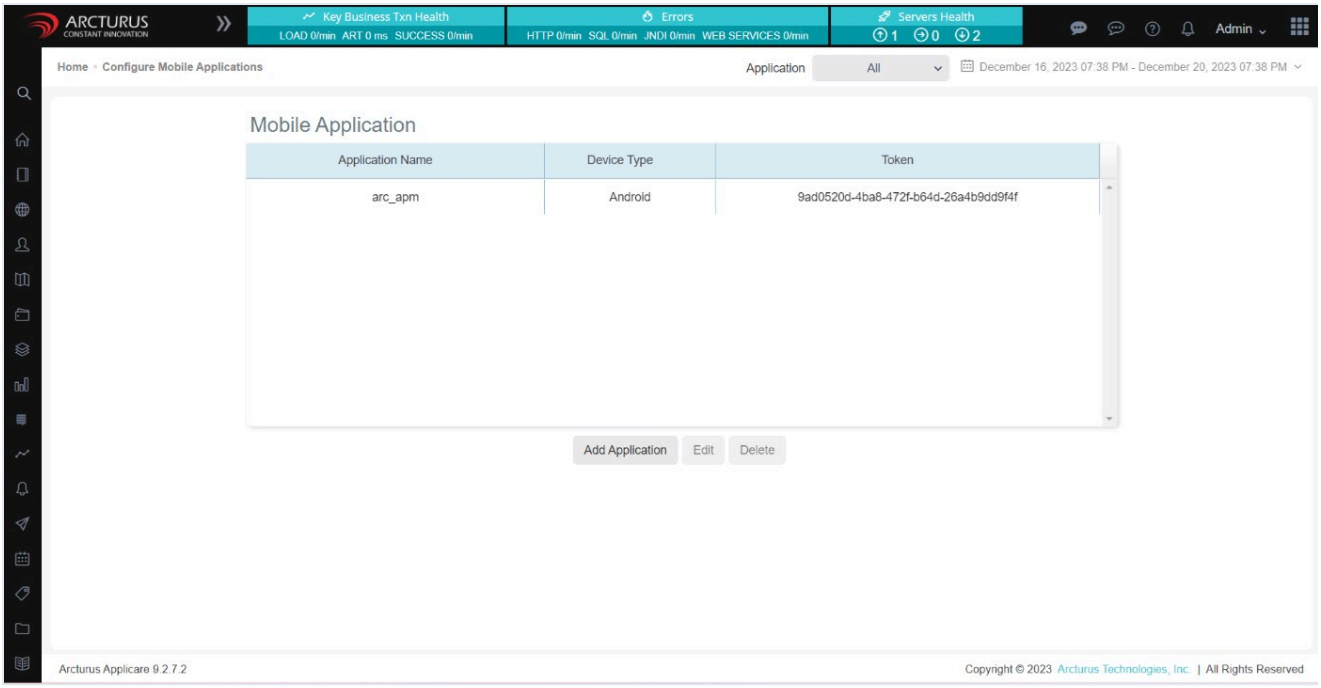




- On the "Configure Mobile Application" screen, select "Android" from the dropdown menu. Enter the application name and click the “Generate Token” button. A token will be generated; be sure to copy and save it for future use.


ARCTURUS
CONSTANT INNOVATION
Key Business Txn Health
LOAD 0/min ART 0 ms SUCCESS 0/min
Errors
HTTP 0/min SQL 0/min JNDI 0/min WEB SERVICES 0/min
Servers Health
1 0 2
Admin
Home • Configure Mobile Applications
Application All
December 16, 2023 07:38 PM - December 20, 2023 07:38 PM
Mobile Application
Add Mobile Application
Device Type Select Device
Application Name Select Device
Android
IOS
Generate Token
Add Application Edit Delete
Arcturus Applicare 9.2.7.2
Copyright © 2023 Arcturus Technologies, Inc. | All Rights Reserved


ARCTURUS
CONSTANT INNOVATION
Key Business Txn Health
LOAD 0/min ART 0 ms SUCCESS 0/min
Errors
HTTP 0/min SQL 0/min JNDI 0/min WEB SERVICES 0/min
Servers Health
1 0 2
Admin
Home • Configure Mobile Applications
Application All
December 16, 2023 07:38 PM - December 20, 2023 07:38 PM
Mobile Application
Add Mobile Application
Device Type Android
Application Name arc_apm
Generate Token
Add Application Edit Delete
Arcturus Applicare 9.2.7.2
Copyright © 2023 Arcturus Technologies, Inc. | All Rights Reserved



- Download the Applicare Android SDK agent.

Integrating the Applicare Agent into your Android Project

- Place the ApplicareMobileTracker.aar file into the **libs** folder under the **library** directory of your Andriod app project.
- Open the build.gradle file located in the **app** folder and add the necessary dependency.

```
TEXT
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'ApplicareMobileTracker.aar'
}
```

- Open the activity class where you want to track load time. In the onCreate() method, create an instance of the TesFactoryMethod Class.

```
TEXT
public class MyApplication extends Application {
    @Override
    public void onCreate() {
        super.onCreate();
        TestFactoryMethod.passtoken("XXXXXXXXXXXXXXXXXXX”);
        //call applicare method
        TestFactoryMethod.getStartTime();
    }

    @Override
    public void onResume() {
        super.onResume();
        TestFactoryMethod.getEndTime();
        TestFactoryMethod.sendDetails(MyApplication.this);
    }
}
```

- Build and install the updated Application.
- Log into the admin console to monitor your Android app's performance.

Monitoring with Applicare

USING APPLICARE › USER EXPERIENCE MONITORING

Measuring User Satisfaction - Apdex

Applicare 9.2 · Updated September 13, 2024 · **Pending technical review**

The **Application Performance Index (Apdex) score** quantifies user satisfaction with the performance of web applications and services. It converts various response times into a single score ranging from 0 to 1, with 1 indicating perfect satisfaction and 0 representing complete dissatisfaction. By default, Applicare uses the average response time of transactions as its **Desired response time(T).**

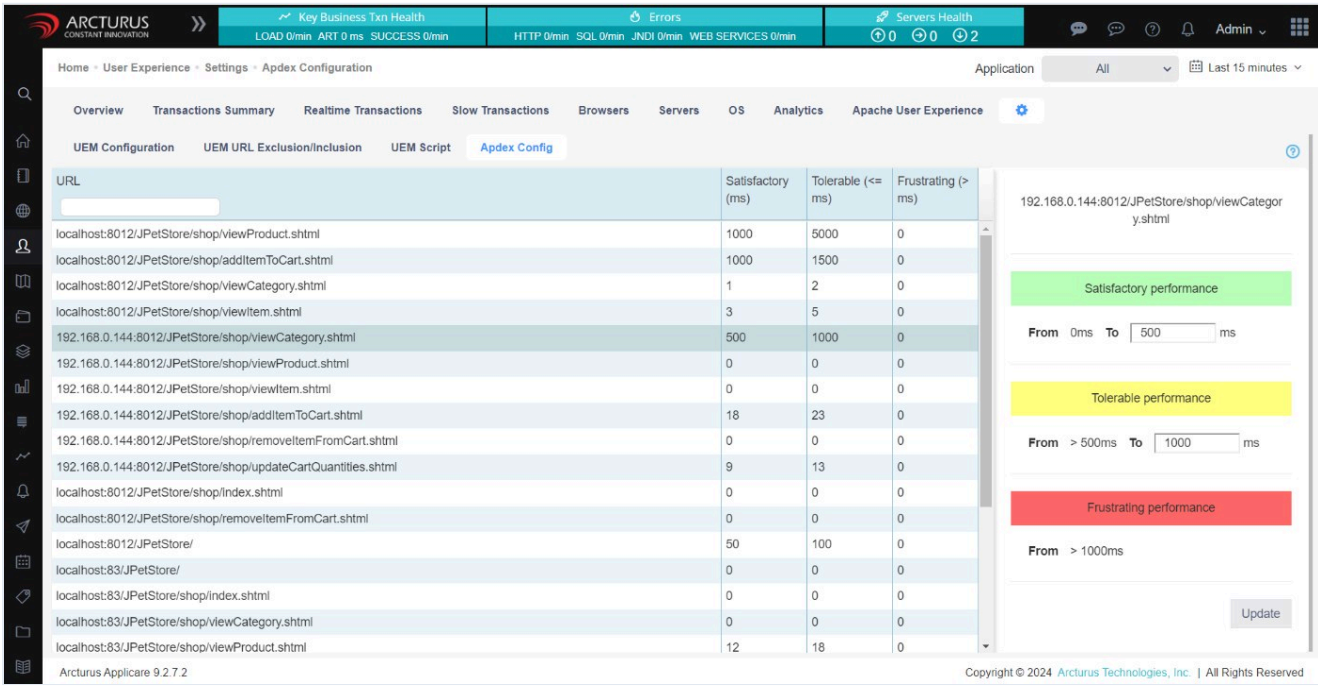
Each transactions is classified as follows:

- **Satisfied:** Response times less than or equal to T.
- **Tolerating:** Response times between T and 4T.
- **Frustrated:** Response times greater than 4T.

The Apdex Score is calculated using the formula:

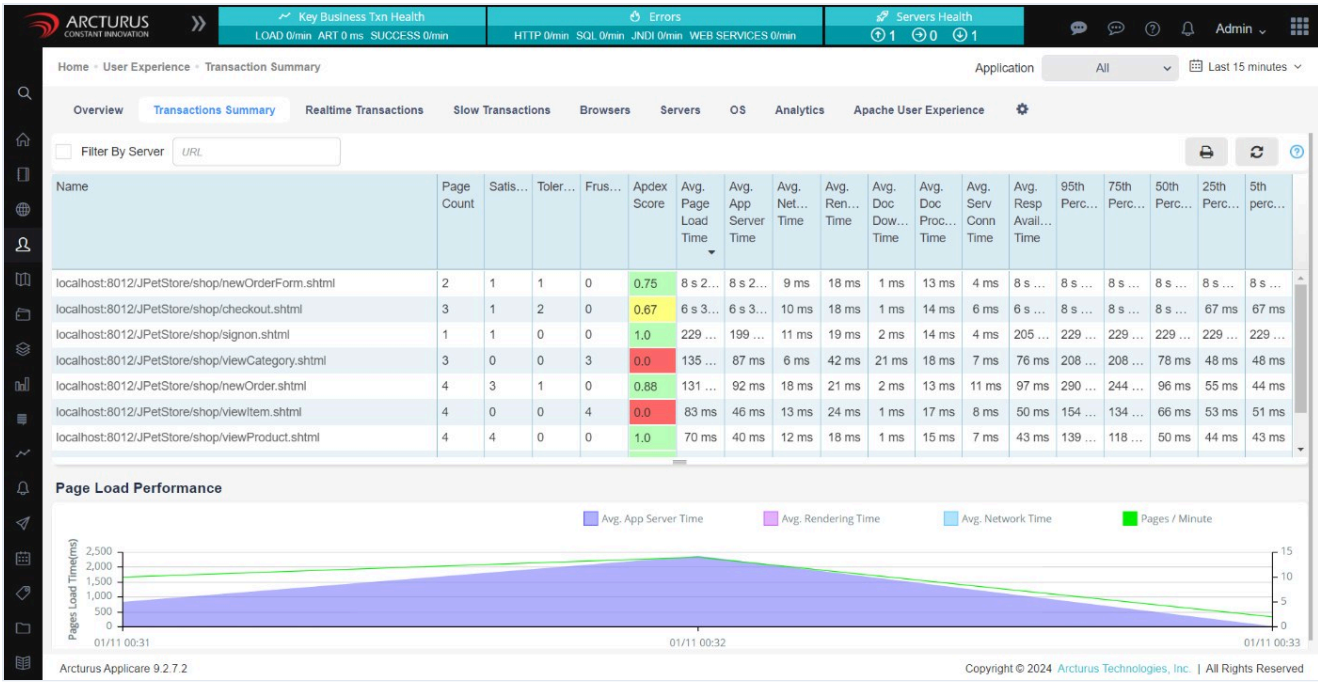
Apdex Score = (Satisfied Count + (Tolerating Count / 2)) / Total Count

You can configure the Apdex threshold for individual transactions in Applicare console by **navigating to Home-> User Experience -> Settings -> Apdex Configuration** screen.



The Apdex score for all the transactions can be viewed under **User Experience -> Transaction summary.**

If no Apdex threshold is set for a transaction, then average response time of that transaction will be used as the default threshold.



Monitoring with Applicare