

Monitoring the Robotshop Application using Instana



Estimated Time: 90 minutes

Getting started:

In this lab, you will delve into the process of connecting the Robotshop application with Instana for efficient monitoring and analysis. Using Docker commands in the terminal, you will establish a seamless connection between the Instana dashboard and the Robotshop application. This hands-on practice will equip you with the knowledge and skills to monitor your application's performance effectively and make data-driven decisions to optimize its efficiency. Let's embark on this journey of seamless integration and exploration of monitoring capabilities with Instana and Docker!

Learning Objectives

On completion of this lab, you will learn to:

- Connect Robotshop Application to Instana via Docker
- Explore Robotshop Application on Instana
- Perform load generation and verification on Instana
- Construct Widgets and Alerts on Instana Dashboard

Prerequisites

- You need to have an Instana account to complete this lab. Instana provides a 14-day free trial. If you already have an Instana trial account as a part of the previous lab, then you may use the same, else you may click the link [Setting up an Instana account](#) to do the same.
- Basic understanding of Docker and docker commands.

Exercise 1: Connecting Robotshop application to Instana via Docker

1. Fork the Git repository: To obtain the robot-shop code, you'll need to fork the project repository available at this [link](#). Forking creates a copy of the repository in your GitHub account, allowing you to work on it independently.
2. In order to fork a repository, you'll need to have your own GitHub account. If you don't have one yet, you can sign up for a free GitHub account on their website. Click Fork, as shown in the image below.

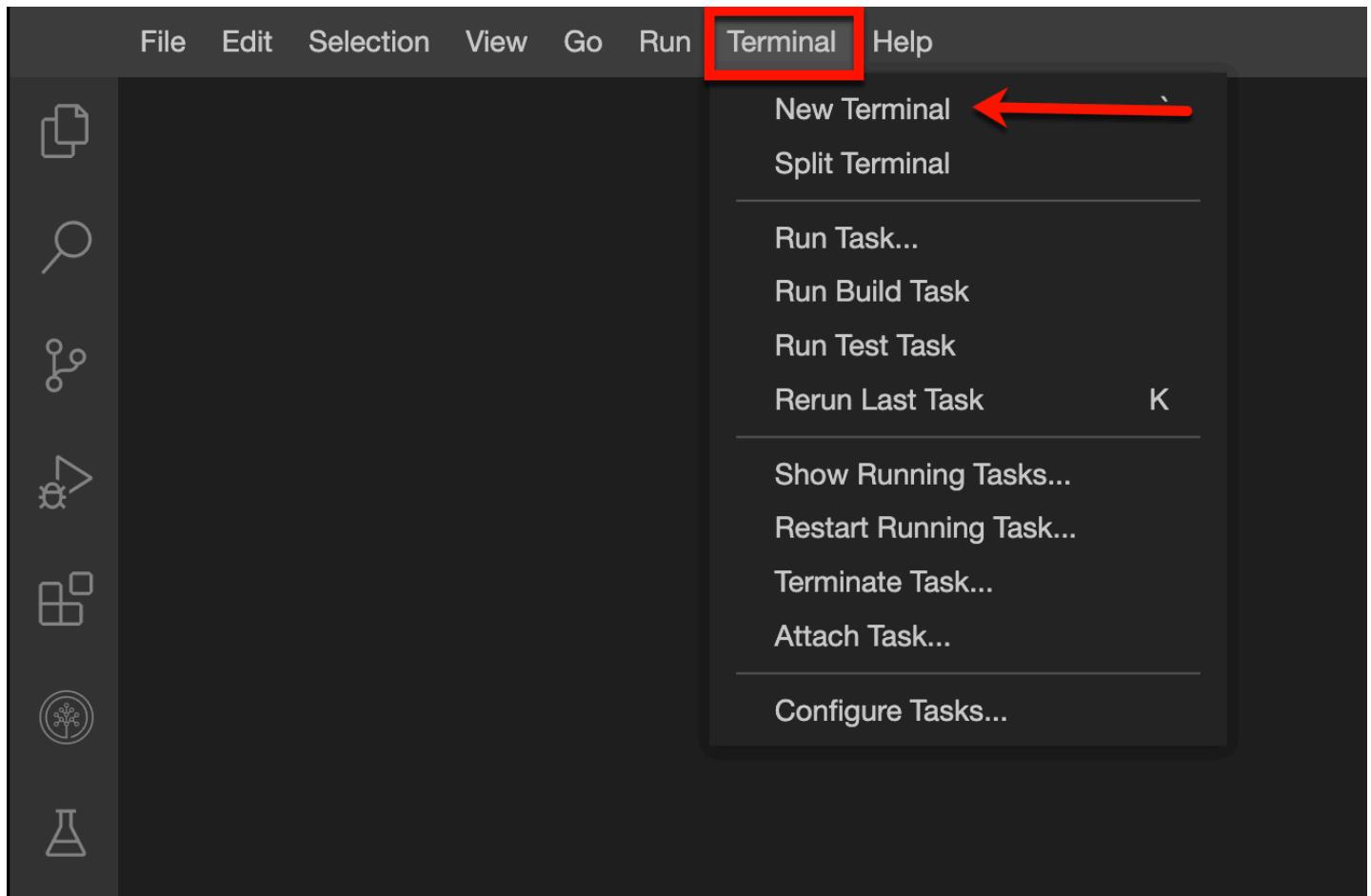
The screenshot shows the GitHub repository page for 'instana / robot-shop'. At the top, there are navigation links for Code, Issues (6), Pull requests (9), Actions, Projects, Security, and Insights. Below the header, the repository name 'robot-shop' is shown as public. On the right, there is a 'Watch' button (82) and a 'Fork' button (1.1k), which is highlighted with a red box. The main area displays the repository's master branch with 8 branches and 0 tags. A list of recent commits is shown, all made by 'bincydan'. The commits include: 'Merge pull request #134 from instana/Remove_Trailing_Slash ...', '.github/workflows limit image builds', 'DCOS reduced cpu requirements; added deploy script', 'K8s Enhance docs to demo affinities', 'OpenShift Update README.md', 'Swarm added docker swarm', 'cart update all versions', 'catalogue update all versions', and 'dispatch update all versions'. On the right side, there is an 'About' section with a sample microservice description, links to instana.com, and various GitHub statistics like 700 stars, 82 watching, and 1.1k forks.

Now, you have the robot-shop code available in your own forked repository, and you can start exploring and working on the project.

3. Navigate to your GitHub forked repository and copy the clone URL.

The screenshot shows the GitHub forked repository page for 'robot-shop'. At the top, it says 'forked from instana/robot-shop'. The repository name is 'robot-shop' (Public). The 'Code' dropdown menu is open, with the 'Clone' option selected. The URL 'https://github.com/[REDACTED]/robot-shop.git' is highlighted with a red box. Below the URL, there are options for 'Local' and 'Codespaces (New)'. The 'Local' tab is selected. Under 'Clone', there are links for 'HTTPS', 'SSH', and 'GitHub CLI'. A note says 'Use Git or checkout with SVN using the web URL.' Below the clone options, there are buttons for 'Open with GitHub Desktop' and 'Download ZIP'. The repository's master branch has 1 branch and 0 tags. A message says 'This branch is up to date with instana/robot-shop:master.' A list of recent commits is shown, identical to the original repository.

4. Click Terminal and then New Terminal. This will open a new terminal window where you can run your code.



5. To clone the repository, execute the following command in your terminal:

```
git clone <your_repo_name>
```

```
theia@theiadocker-:~/home/project$ git clone https://github.com/robotshop/robot-shop.git
Cloning into 'robot-shop'...
remote: Enumerating objects: 2571, done.
remote: Total 2571 (delta 0), reused 0 (delta 0), pack-reused 2571
Receiving objects: 100% (2571/2571), 81.41 MiB | 38.12 MiB/s, done.
Resolving deltas: 100% (1399/1399), done.
theia@theiadocker-:~/home/project$
```

6. Next, navigate to the "robot-shop" directory by using the following command in the terminal:

```
cd robot-shop
```

7. Before initiating the build and downloading the tracing module for Nginx, ensure you have a valid Instana agent key. Set this key in the environment by running the export command as follows:

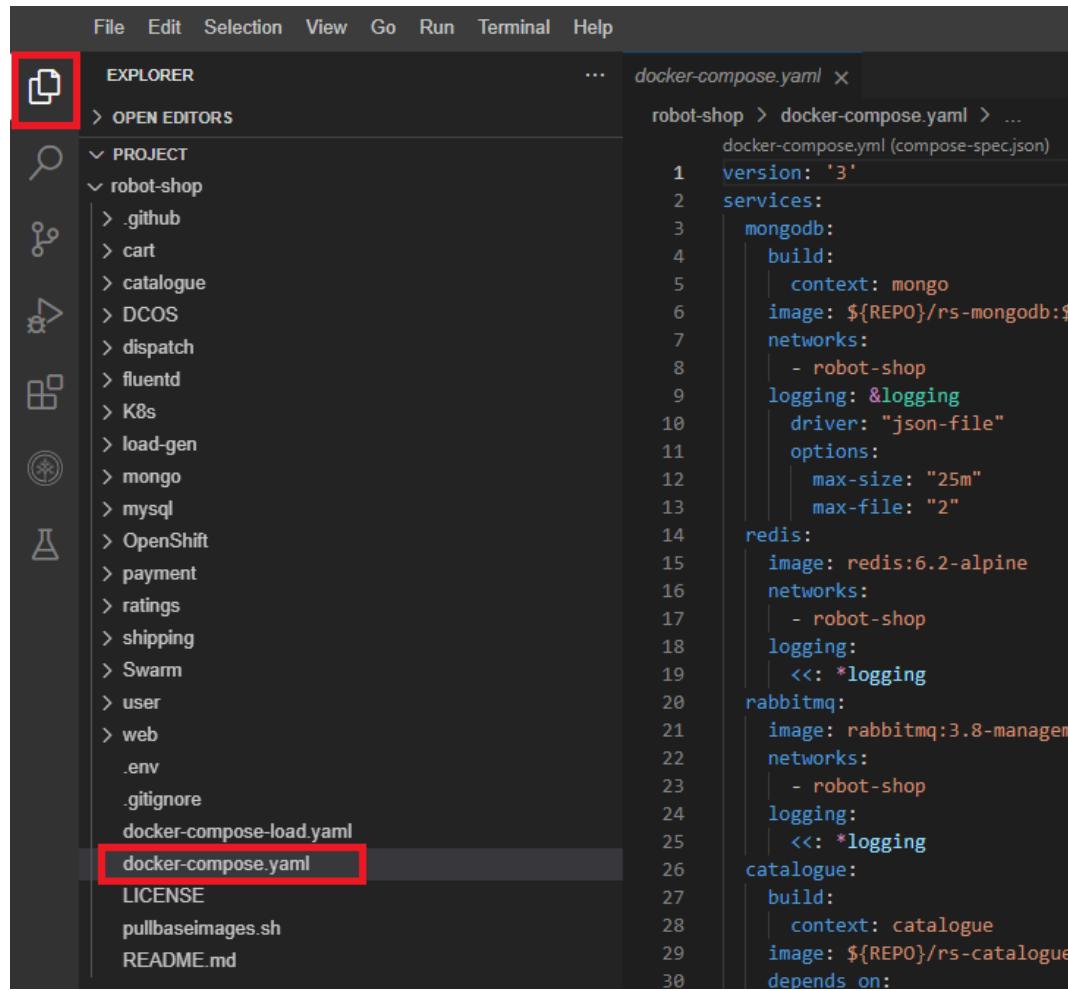
```
export INSTANA_AGENT_KEY=<your agent key>
```

Note: In case you have not saved the Instana agent key as a part of the previous lab, you may log in to your Instana account to retrieve the same. For more details, you can refer to [Setting up an Instana account](#).

```
theia@theiadocker-:~/home/project$ cd robot-shop/
theia@theiadocker-:~/home/project/robot-shop$ export INSTANA_AGENT_KEY=your_instana_agent_key
theia@theiadocker-:~/home/project/robot-shop$
```

8. To configure the web image within the docker-compose.yaml file, you need to set the environment variables `INSTANA_EUM_KEY` and `INSTANA_EUM_REPORTING_URL`.

In the File Explorer, navigate to the robot-shop repository and locate the "docker-compose.yaml" file. Double-click the "docker-compose.yaml" file to open it with a text editor to set the environment variables.



```
version: '3'
services:
  mongodb:
    build:
      context: mongo
    image: ${REPO}/rs-mongodb:$VERSION
    networks:
      - robot-shop
    logging:
      &logging
        driver: "json-file"
        options:
          max-size: "25m"
          max-file: "2"
  redis:
    image: redis:6.2-alpine
    networks:
      - robot-shop
    logging:
      <<: *logging
  rabbitmq:
    image: rabbitmq:3.8-management
    networks:
      - robot-shop
    logging:
      <<: *logging
  catalogue:
    build:
      context: catalogue
    image: ${REPO}/rs-catalogue
    depends_on:
```

In the docker-compose.yaml file, scroll down to the bottom until you find lines 164 to 166. Remove the hash '#' sign from the beginning of each line to enable Instana EUM (End User Monitoring).

```
docker-compose.yaml x

robot-shop > docker-compose.yaml > {} services > {} web > ...
144     |     context: web
145     |     args:
146         |         # agent key to download tracing libs
147         |         KEY: ${INSTANA_AGENT_KEY}
148     image: ${REPO}/rs-web:${TAG}
149     depends_on:
150         - catalogue
151         - user
152         - shipping
153         - payment
154     ports:
155         - "8080:8080"
156     networks:
157         - robot-shop
158     healthcheck:
159         test: [ "CMD", "curl", "-H", "X-INSTANA-SYNTHETIC: 1", "-f", "http://localhost:8080" ]
160         interval: 10s
161         timeout: 10s
162         retries: 3
163     # Uncomment to enable Instana EUM
164     # environment:
165         # INSTANA_EUM_KEY: <your eum key>
166         # INSTANA_EUM_REPORTING_URL: <your reporting url>
167     logging:
168         <<: *logging
169
170 networks:
171     robot-shop:
```

Replace your `INSTANA_EUM_KEY` and `INSTANA_EUM_REPORTING_URL` with the actual values provided by Instana for your application. You may refer to [Setting up an Instana account](#) for the same in case you missed saving these as a part of the previous lab.

```
File Edit Selection View Go Run Terminal Help

docker-compose.yaml x

robot-shop > docker-compose.yaml > {} services > {} web > {} environment > ⚡ INSTANA_EUM_KEY
144     |     context: web
145     |     args:
146         |         # agent key to download tracing libs
147         |         KEY: ${INSTANA_AGENT_KEY}
148     image: ${REPO}/rs-web:${TAG}
149     depends_on:
150         - catalogue
151         - user
152         - shipping
153         - payment
154     ports:
155         - "8080:8080"
156     networks:
157         - robot-shop
158     healthcheck:
159         test: [ "CMD", "curl", "-H", "X-INSTANA-SYNTHETIC: 1", "-f", "http://localhost:8080" ]
160         interval: 10s
161         timeout: 10s
162         retries: 3
163     # Uncomment to enable Instana EUM
164     environment:
165         INSTANA_EUM_KEY: _____
166         INSTANA_EUM_REPORTING_URL: _____
167     logging:
168         <<: *logging
169
```

Now, locate the `index.html` in `robot-shop/web/static/index.html`. Once you've located it, you can paste the script that you previously copied from the robot-shop website into this file.

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  <script>
5      (function(s,t,a,n){s[t]||(s[t]=a,n=s[a]=function(){n.q.push(arguments)}},
6      n.q=[],n.v=2,n.l=1*new Date())})(window,"InstanaEumObject","ineum");
7
8      ineum('reportingUrl', 'https://REDACTED.instana.io');
9      ineum('key', 'REDACTED');
10     ineum('trackSessions');
11 </script>
12 <script defer crossorigin="anonymous" src="REDACTED"></script>
13 <!--#include file="com.instana.js"-->
14 <title>Stan's Robot Shop</title>
15 <link rel="shortcut icon" type="image/png" href="/media/instana_icon_square.png"/>
16 <style type="text/css">
17     @import url('https://fonts.googleapis.com/css?family=Orbitron');
18     @import url('/css/style.css');
19     @import url('/css/auto-complete.css');
20 </style>
21 <meta http-equiv="Cache-Control" content="no-Cache">
22 <meta http-equiv="Pragma" content="no-cache">
23 </head>
24
25 <body ng-app="robotshop">
<div ng-controller="shopForm">

```

Note: Ensure you are using your credentials retrieved from your Instana application accurately. Accurate usage of your Instana credentials is crucial for seamless integration and proper functioning of the monitoring features.

9. To begin, pull down the required images using the following command:

```
docker-compose pull
```

Executing this command will pull all the necessary Docker images specified in the docker-compose.yaml file. Once the images are downloaded, you'll be all set to proceed with setting up and running your application using Docker Compose.

Note: This command can take up to 3 minutes to run. You can explore the docker-compose.yaml file in the meantime.

10. Start the Robotshop application by running the following command:

```
docker-compose up -d
```

```

theia@theia-docker-pallavir:/home/project/robot-shop$ docker-compose up -d
[+] Running 13/13
## Network robot-shop_robot-shop      Created
## Container robot-shop-mysql-1       Started
## Container robot-shop-mongodb-1      Started
## Container robot-shop-rabbitmq-1     Started
## Container robot-shop-redis-1        Started
## Container robot-shop-dispatch-1     Started
## Container robot-shop-payment-1      Started
## Container robot-shop-ratings-1      Started
## Container robot-shop-cart-1         Started
## Container robot-shop-shipping-1     Started
## Container robot-shop-user-1         Started
## Container robot-shop-catalogue-1    Started
## Container robot-shop-web-1          Started

```

This command will initiate the Docker Compose process, which will set up and run the Robotshop application using the configurations specified in the docker-compose.yaml file.

11. Run the below command for docker login:

```
docker login containers.instana.io -u _ -p $INSTANA_AGENT_KEY
```

```
theia@theiadocker-pallavir:/home/project/robot-shop$ docker login containers.instana.io -u _ -p $INSTANA_AGENT_KEY
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /home/theia/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
```

Login Succeeded

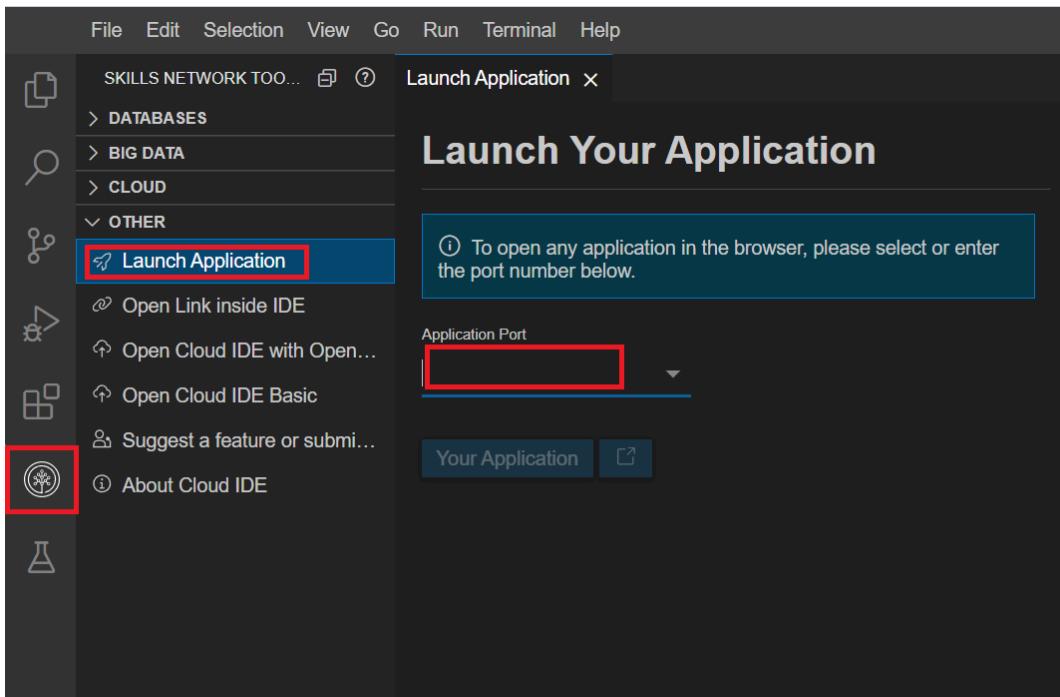
12. To run the Docker container and launch the application, use the docker run command as follows:

```
docker run \
--name robot-shop \
-e INSTANA_AGENT_KEY='your_instana_agent_key' \
-e INSTANA_AGENT_ENDPOINT='your_instana_agent_endpoint' \
-e INSTANA_AGENT_ENDPOINT_PORT=443 \
-v /var/run/docker.sock:/var/run/docker.sock \
containers.instana.io/instana/release/agent/dynamic:latest
```

```
theia@theiadocker-pallavir:/home/project/robot-shop$ docker run \
> --name robot-shop \
> -e INSTANA_AGENT_KEY='[REDACTED]' \
> -e INSTANA_AGENT_ENDPOINT='[REDACTED]' \
> -e INSTANA_AGENT_ENDPOINT_PORT=443 \
> -v /var/run/docker.sock:/var/run/docker.sock \
> containers.instana.io/instana/release/agent/dynamic:latest
Unable to find image 'containers.instana.io/instana/release/agent/dynamic:latest' locally
latest: Pulling from instana/release/agent/dynamic
5329d7039f29: Pull complete
1a684b2ce2a2: Pull complete
19f3c0601a49: Pull complete
a276944861ec: Pull complete
75e0c1588f80: Pull complete
4f4fb700ef54: Pull complete
0269f73e750a: Pull complete
44e398a0a555: Pull complete
17481c4854ab: Pull complete
5a0905996848: Pull complete
888cf1ba0306: Pull complete
c0d7df430776: Pull complete
5299fb955772: Pull complete
82b6266dcde8: Pull complete
099ccc02ee6a: Pull complete
17481c4854ab: Pull complete
5a0905996848: Pull complete
888cf1ba0306: Pull complete
c0d7df430776: Pull complete
5299fb955772: Pull complete
82b6266dcde8: Pull complete
099ccc02ee6a: Pull complete
17481c4854ab: Pull complete
5a0905996848: Pull complete
888cf1ba0306: Pull complete
c0d7df430776: Pull complete
5299fb955772: Pull complete
82b6266dcde8: Pull complete
099ccc02ee6a: Pull complete
```

Note: Ensure that you replace all the environment variables in the configuration with your actual credentials instead of using the ones shown in the example image.

13. To access the application, click "Skills Network", located on the left side. This will open the Skills Network Toolbox. Within the Toolbox, click Other, and then select Launch Application. Enter the port number "8080" in the box and click "Launch Application". By following these steps, you'll be able to access and interact with the application on port 8080.



14. By opening the link in your web browser, you should be able to access and view the Stan's Robotshop application. This step ensures that the application is running correctly and allows you to interact with its features.

Welcome to Stan's Robot Shop

Here you will find all of Stan's friends. Have a browse around and see who is here.

This is a simple example microservices ecommerce application. It has been built using various technologies:

- AngularJS (1.x)
- Nginx
- NodeJS
- Java
- Python
- Golang
- PHP (Apache)
- MongoDB
- Redis
- MySQL

When deployed into an environment monitored by Instana, these technology stacks will be automatically detected and monitored, all with minimum configuration. Every request will be traced end to end. Stan will keep an eye on all those metrics, events and traces and let you know what needs your attention.

To find out more visit the [Instana site](#).

All the code is available on [Github](#).

anonymous-1

Exercise 2: Exploring Robotshop application on Instana

Now that your application is fully prepared, it's time to initiate monitoring using Instana.

1. To access the Infrastructure dashboard, simply click the corresponding option from the left sidebar. By selecting the Infrastructure option, you'll be directed to the dashboard that provides insights into the infrastructure components of your application.

From the dashboard, you can explore various metrics, visualize resource utilization, and analyze the health of your infrastructure components to ensure they are operating optimally.

Note: This dashboard will become available after running the Docker command from the terminal and successfully connecting the Robotshop application with Instana.

The screenshot shows the Instana Infrastructure dashboard. On the left, there is a sidebar with various monitoring options like APM, Java Runtime, and Process ID. The main area features a 3D map where a host named "Instana Agent @ 08e0950e14ae" is highlighted with a red box. A tooltip for this host says "Open Dashboard". Below the map, there is a "Monitoring Issues" section with one entry: "63caf3e58811 ebpf_not_supported". The "CPU Load" and "JVM Memory" sections are also visible at the bottom.

2. Now, click Open Dashboard as shown in the below image:

The screenshot shows the Instana Agent dashboard for the host "Instana Agent @ 63caf3e58811". The left sidebar lists system details such as Java Runtime, Java Version, User, Started At, Executable, Max Open Files, Process ID, Arguments, and Environment Variables. The main area has three main sections: "Monitoring Issues" (listing the ebpf_not_supported error), "CPU Load" (a line chart showing spikes in User and System load over time), and "JVM Memory" (a chart showing Heap, Direct Buffers, and Off Heap memory usage).

Key features and functionalities of the Infrastructure module in Instana include:

- Real-time Monitoring:** Instana continuously collects and analyzes metrics, logs, and traces from your infrastructure. It provides real-time monitoring of key performance indicators, such as CPU usage, memory utilization, network traffic, disk I/O, and more.
 - Health Monitoring:** Instana evaluates the health of your infrastructure components and alerts you about any anomalies or potential issues. It uses built-in health rules and anomaly detection algorithms to proactively identify and notify you about any performance degradation or abnormalities.
 - Resource Utilization:** Instana provides insights into the resource utilization of your infrastructure, allowing you to identify bottlenecks and optimize resource allocation. It tracks metrics such as CPU, memory, disk, and network usage to help you understand resource consumption patterns.
3. To obtain detailed information about crucial metrics such as CPU load, CPU usage, and more, navigate to the Infrastructure dashboard and click the corresponding Host that corresponds to the Robotshop application.

No Issues

Monitoring Issues

On	Code	Description
63caf3e58811	ebpf_not_supported	The Operating System of this host does not seem to offer the extended Berkeley Packet Filter (eBPF) functionality that is required by the Process Abnormal Termination detection.

CPU Load

User and System show load in percent where 100% equal a single core at full load. Load is computed in the JVM related to the combined processing power and normalized to 1.

JVM Memory

Heap: 150 MiB

No Issues

Monitoring issue: ebpf not supported

The Operating System of this host does not seem to offer the extended Berkeley Packet Filter (eBPF) functionality that is required by the Process Abnormal Termination detection.

CPU Usage

30%

Memory Usage

10%

CPU Load

5.34

CPU Usage

100%

Context Switches

76.000

CPU Load

8.19

4. Navigate to the Instana option in the left menu. Upon accessing the main page, you will find the Robotshop website running. Click the Robotshop website to open and explore its details.

robot-shop

No Issues

Analyze Page Loads

Filters Browser OS Country Subdivision Meta Window Width

Summary Speed Resources HTTP Requests JS Errors Geography Custom Events Pages Smart Alerts Configuration

Page Loads: 1

Page Transitions: 1

onLoad Time (mean): 1.59s

onLoad Time (90th): 1.59s

onLoad Time (95th): 1.59s

Page Views

JS Errors

onLoad Time

By following these steps, you'll be able to access and explore the monitoring details of the Robotshop website in the Instana dashboard. This allows you to gain valuable insights into the website's performance, health, and user interactions.

5. Take a closer look at the Robotshop website by scrolling down to explore it further. On this page, you'll find information about the top pages of the website. Currently, you have access to two pages, which you can examine in detail.

Top Pages

Page Views

onLoad Time

Errors

splash.html

1

splash

0

[View all pages](#)



[ADD SMART ALERT](#)

6. Now, go back to your Stan's Robotshop Application on your browser and perform some actions like Register and log in.

The screenshot shows the Stan's Robot Shop application interface. On the left, there is a sidebar with navigation links: 'Login / Register', 'Cart' (which is currently 'Empty'), and 'Categories' (listing 'Artificial Intelligence' and 'Robot'). The main content area has two forms. The top form is for 'Login' and contains fields for 'Name' and 'Password', with a 'Login' button. The bottom form is for 'Register' and contains fields for 'Name', 'Email', 'Password', and 'Confirm Password', with a 'Register' button. At the bottom left of the main area, it says 'anonymous-2'.

7. Once you have successfully logged into the application, enhance your experience by adding a product from the catalog to your shopping. Proceed to the cart by selecting Cart, then click Checkout. Within the checkout process, choose any desired Country and Location from the provided drop-down menus. Next, click Calculate to determine the total cost, review the order details, and confirm your selections. To finalize the transaction, click Pay now.

8. To observe the occurrence of various pages, including cart, shipping, and payments, in the Robotshop application, simply follow these steps:

- Now, return to the Instana Dashboard, which serves as the monitoring tool to track the performance of your application.
- After navigating back to the dashboard, ensure you have the most up-to-date data by refreshing the page.
- After refreshing the Instana Dashboard, you'll notice that all pages, including cart, shipping, payments, and others, are now visible in the "Top Pages" section.
- With each order placed in the Robotshop application, Instana diligently captures and displays the occurrences of these pages in the "Top Pages" section. This data becomes a valuable resource for gaining insights into user interactions and usage patterns within the application.

Top Pages	Page Views	onLoad Time	Errors
splash	1		
cart.html	1		
shipping.html	1		
splash.html	1		
login.html	1		

[View all pages](#)

 [ADD SMART ALERT](#)

Exercise 3: Load Generation and Verification on Instana

Now that we have gained an understanding of monitoring applications on Instana, it's an opportune time to delve deeper into the monitoring capabilities by generating more load.

By generating additional load on the application, you can simulate increased user interactions and stress test its performance under heavier conditions. This process allows you to observe how the application handles higher traffic and identify any potential bottlenecks or performance issues that might arise.

1. Go back to the terminal. Click Terminal and then New Terminal from the top menu, and start a new terminal.

Note: Please avoid closing the terminal where the Docker run command is currently executing. Keeping this terminal open is essential as it ensures the Docker container remains running and active, allowing continuous monitoring and seamless interaction with the application.

2. In the new terminal, navigate to the `robot-shop` directory.

3. To view the number of items you have added to the cart, use the `curl` command in the terminal as given below:

```
curl http://localhost:8080/api/cart/metrics
```

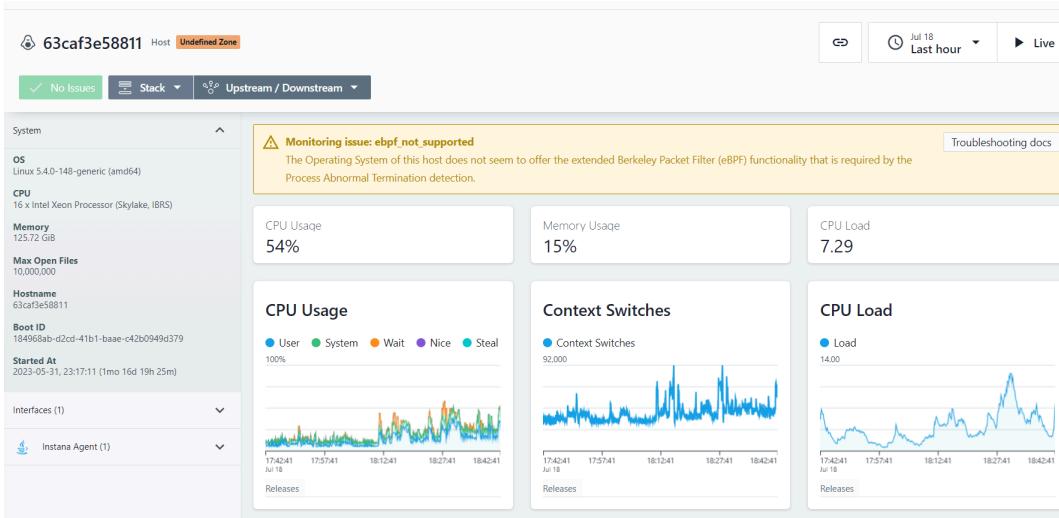
```
Problems      theia@theiadocker-pallavir: /home/project/robot-shop      theia@theiadocker-pallavir: /home/project/robot-shop ×
theia@theiadocker-pallavir: /home/project$ cd robot-shop/
theia@theiadocker-pallavir: /home/project/robot-shop$ curl http://localhost:8080/api/cart/metrics
# HELP items_added running count of items added to cart
# TYPE items_added counter
items_added 1
theia@theiadocker-pallavir: /home/project/robot-shop$ □
```

4. To increase the load on the application and observe the CPU load on Instana, execute the following command:

```
docker-compose -f docker-compose.yaml -f docker-compose-load.yaml up
```

```
theia@theiadocker-pallavir:/home/project/robot-shop$ docker-compose -f docker-compose.yaml -f docker-compose-load.yaml up
WARN[0000] The "INSTANA_AGENT_KEY" variable is not set. Defaulting to a blank string.
[+] Running 11/15
  ⋮ load Pulling
    # 4c25b3090c26 Already exists          4.7s
    # 1acf565088aa Already exists          0.0s
    # b95c0dd0dc0d Already exists          0.0s
    # 5cf06daf6561 Already exists          0.0s
    # 942374d5c114 Already exists          0.0s
    # 64c0f10e4cfa Already exists          0.0s
    # 76571888410b Already exists          0.0s
    # 5e88ca15437b Already exists          0.0s
    # 0ab5ec771994 Already exists          0.0s
    # b74c678628e5 Pull complete           0.3s
    # 29c5f6ec7ca7 Pull complete           0.6s
  ⋮ 186e191791bf Extracting [=====>] 27.0...
  ⋮ 290f4940ff67 Download complete      4.4s
  ⋮ 7bab3e563223 Download complete      4.4s
```

5. Return to the Instana page and click refresh. After refreshing the Instana page, navigate to the Infrastructure option from the left menu. Access the corresponding dashboard, and you will notice a significant increase in CPU usage and load metrics. This sudden spike in data is indicative of a surge in activity within the application.



Exercise 4: Creating Widgets and Alerts

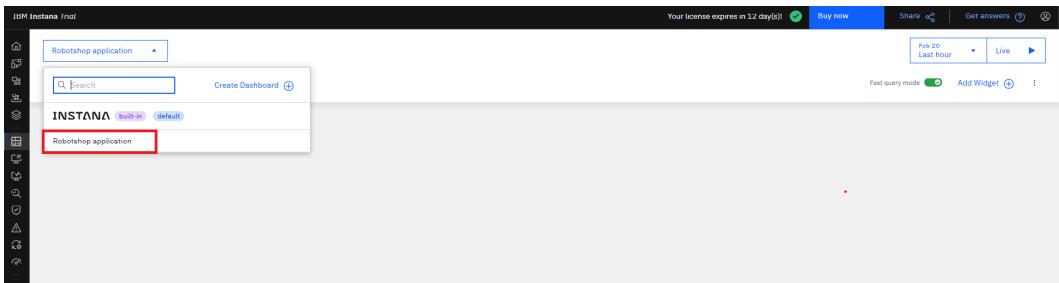
In this exercise, you will explore the process of creating various widgets and alerts on the Instana dashboard. By doing so, you'll learn how to customize and visualize monitoring data effectively, enabling you to gain valuable insights into your application's performance and health.

Widgets

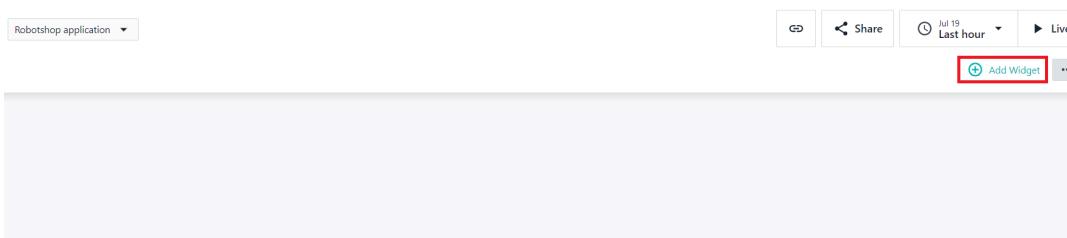
You will be creating a **Widget** on the Robotshop application dashboard. Widgets provide valuable visual representations of various metrics and data, enhancing the monitoring and analysis of the application's performance and health.

Widget 1: This widget will display the total count of items that have been added to the cart in the Robotshop application. By displaying this information in a visual format, you can easily monitor and track the cart's usage, enabling you to assess the popularity of various products and the overall engagement of users with the application's shopping feature.

- On the Instana main page, simply click to open the dashboard that was created during the setup lab.



- Now, proceed by clicking **Add Widget** to add a new widget to the dashboard.



- Choose the Chart: Time Series option and proceed by clicking Next.

⊕ Add Widget

Apdex: Chart BETA

Big Number

Chart: Pie

Chart: Time Series

Markdown

SLO

Time Zone

Top List BETA

Top 5 slowest services

catalogue cart ratings discount shipping

12

12:17:20 12:17:30 12:17:40 12:17:50 12:18:00 12:18:10 12:18:20

Jun 29

Cancel Next

- To create a widget, use the parameters provided below:

- Data source: Websites(Beacons)
- Beacon Type: Page Transition
- Metric: Page Transitions
- Filter: Add 1st filter, from the dropdown select Configuration>Name and in the value box write robot-shop. Add 2nd filter Location>PageName and write cart.html into the value box.
- Group: Add group Location>Path

⊕ Add Widget – Chart: Time Series

Y1.1 Page Transition - Page Transitions

Data Source Websites (Beacons)

Beacon Type Page Transition

Metric Page Transitions

Aggregation sum

Filter Configuration > Name = robot-shop AND Location > Page Name = cart.html X Add filter

Group Optional Location > Path X

Top 5 ▼ Show remaining groups aggregated as "Other"

Time Shift Apply a time shift to this data series

- Axis Configuration:
 - Select Chart:Bar Formatter:Number, eg 42
 - Widget Name: Items in cart
 - Click Create

Summary + Data

Chart Bar

Formatter Number, e.g. 42

Min Auto Max Auto

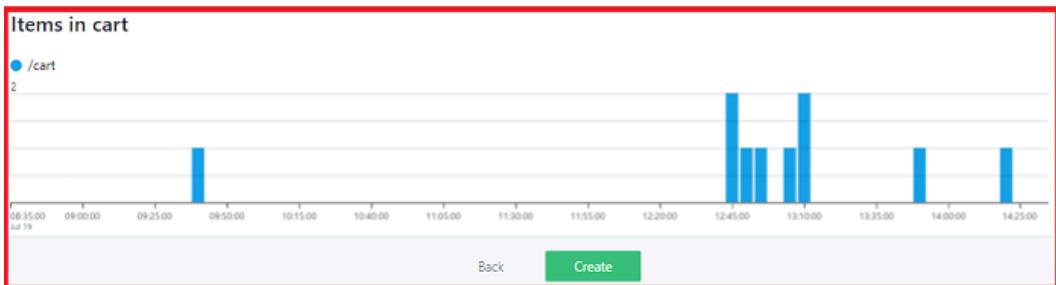
Datasets

Y1 Page Transition - Page Transitions

Widget Name

Name Items in cart

review



In the Robotshop application dashboard, you can observe the number of items added to the cart depicted through bars on a graph. The graph visually represents the quantity of items in the cart at specific points in time, offering a clear visualization of how the cart content evolves over time. By observing this graph, you can easily track fluctuations in cart contents, identify popular shopping periods, and understand user behavior patterns in relation to cart usage.

- Click "Save Changes" on the dashboard to preserve and store the newly added widget.

Widget 2: This widget will display the count of unique users who have logged into the Stan's Robotshop application, along with the respective occurrence frequency.

With this widget, you can visually track the number of distinct users accessing the application, allowing you to gauge user engagement and assess the overall popularity of the Robotshop. By monitoring the occurrences of unique logins, you gain valuable insights into user activity patterns and the level of interest in your application.

Unique Users: First time user

- Click Add Widget to proceed with adding a new widget to the dashboard.

Robotshop application ▼

GO Share Jul 19 Last hour Live

+ Add Widget ..

- Choose the Chart: Time Series option and proceed by clicking Next.

⊕ Add Widget

Apdex: Chart BETA

Big Number

Chart: Pie

Chart: Time Series BETA

Markdown

SLO

Time Zone

Top List BETA

Top 5 slowest services

catalogue cart ratings discount shipping

Cancel Next

- To create a widget, use the parameters provided below:

- Data source: 'Websites(Beacons)'
- Beacon Type: Custom Event
- Metric: Occurrences

⊕ Add Widget – Chart: Time Series

Datasets

Y1.1 Custom Event - Occurrences

Data Source	Websites (Beacons)
Beacon Type	Custom Event
Metric	Occurrences
Aggregation	sum
Filter	+ Add filter
Group	+ Add group
Time Shift	<input checked="" type="checkbox"/> Apply a time shift to this data series
Name	Custom Event - Occurrences

[Add dataset](#)

- Axis Configuration:

- Select Chart:Bar Formatter:Number, eg 42
- Widget Name: Unique users occurrences
- Click Create

Chart ▾

Formatter ▾

Min Max

Datasets

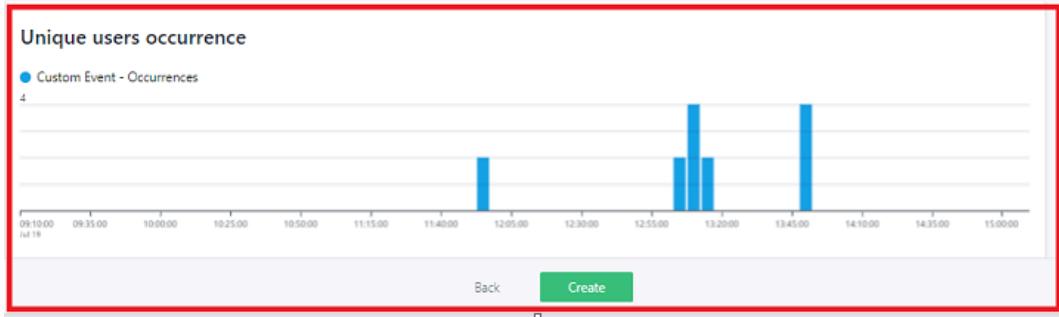
 Custom Event - Occurrences

Widget Name

Name ?

Optional

Preview



In the widget, you'll notice a graphical representation displaying the number of unique users, represented by distinct bars on the graph. Each bar corresponds to one user, and the occurrences are visually depicted for all the users. This graphical representation offers a clear and concise view of the individual user counts and their respective occurrences within the Stan's Robotshop application.

- Click Save changes on Dashboard to save the newly created widget.

Robotshop application ▾

Save changes

X Discard Changes

Items in cart

● /cart

2



Releases

Unique users occurrence

● Custom Event - Occurrences

4



Releases

You've successfully created two widgets on your dashboard, and each of them features an informative graph displaying essential information.

Smart Alerts

Now, you will add a smart alert whenever the e-commerce solution gets transactions from Canada.

The smart alert will be designed to monitor transaction activities in real-time and instantly trigger a notification whenever a transaction is detected from Canada. This proactive approach ensures that you are promptly informed about any transactions originating from that specific region.

1. Navigate to the **Settings** option in the left menu, and then select **Alert Channels**. Click **Add Alert Channel** and select **Email** from the drop-down list on the alert channel page. This will allow you to set up an email-based alert channel for receiving notifications and alerts from Instana.

The screenshot shows the 'Alert Channels' page under the 'Settings' menu. On the left sidebar, 'Alert Channels' is highlighted with a red box. The main area displays a table with one row, showing a bar chart icon and the message 'No data available'. A red box highlights the '+ Add Alert Channel' button in the top right corner.

Name	Properties
[Empty]	No data available

+ Add Alert Channel

2. Now, let's proceed to name the alert channel as **Transaction from Canada** and set up the **email** address where you wish to receive the alert notifications.

Create Email Alert Channel

Name
Transaction from canada

Emails
[REDACTED]@gmail.com

Test Channel

Cancel Create

3. In the Settings page, locate and click the Alert option, as indicated in the image below:

The screenshot shows the 'Settings' page with a sidebar on the left containing various icons and sections: Team Settings (selected), User Settings, Authentication, Account & Billing, ACCESS CONTROL (Users, Pending Invitations, Groups, API Tokens), EVENTS & ALERTS (Alerts Hub, Events, Alerts (selected)), LOG MANAGEMENT (Alert Channels, Maintenance Windows BETA, Custom Payload). The main content area is titled 'Alerts' and displays a table with columns 'Name' and 'Additional Scope'. A message at the bottom states 'No alert configured.' with a bar chart icon.

4. Click New Alert on the alert page. On the Create New Alert page:

- o Set the Name: Transaction from canada
- o In Events: Select Alerts on Event Type(s), turn the warning issues to green
- o In scope: All available entities
- o Alerting: Click Add Alert channel, select the Transaction from canada, and click Add 1 channel
- o Click Create

Users

Pending Invitations

Groups

API Tokens

EVENTS & ALERTS

Alerts Hub

Events

Alerts

Alert Channels

Maintenance Windows BETA

Custom Payload

LOG MANAGEMENT

Coralogix

ELK

3. Scope

Apply on (required)

All available entities

Caution! All events that match the event types will enter the notification stream.

Your selection matches 4 events over the past 2 weeks.

4. Alerting

Alert Channels (1)		+ Add Alert Channels	Filter...	
Name	Properties			

Transaction from canada
Email

EMAILS
pallaviraj7597@gmail.com

5. Custom Payloads (Optional)

Each key/value pair will be included as additional payload to each issue or incident alert notification.

Global Custom Payloads (0)	
----------------------------	--

5. To add a smart alert for the Robotshop website, simply open the website and locate 'Add smart alert' positioned at the bottom right corner.

- Step 1. Select Alert: Select `HTTP Status` code from the left menu and status code `202(Accepted)` from drop-down menu. Click `Next`.

Create Smart Alert

Step 1: Select Alert Step 2: Select Scope Step 3: Select Alert Channels [Switch to Advanced Mode](#)

What do you want to be alerted on?

Slowness	Automatic Alerts for HTTP Status Codes
JS Errors	Receive an alert every time when matching HTTP Status Codes occur more often than usual.
HTTP Status Codes	Status Code <code>202 (Accepted)</code>
Unexpectedly Low Number of Page Loads	
Unexpectedly High Number of Page Loads	
Custom Events	

- Step 2: Select Scope: Add filters:

- Click `+Add Filter` and Select `URL` from the `Location` section: Paste the Stan's Robotshop application page URL into the value box.
- Again, add another filter and select `Country` and write `Canada` in value box.
- Add another filter and `Page Name` and write `payment.html` into the value box.
- Click `Next`

- Step 3: Select Alert Channel: Click `Select Alert Channel`, select `Transaction from canada`, and click `Add 1 Channel`

- Click `Create`

6. In the robot-shop website, click `Smart Alert` option and click the 202 alert created.

Alert Configuration

Details

Threshold Type: Static Threshold Metric: Status Code Count \geq 5 Scope: Per Website

Trigger: Last 24 hours Last 7 days

Alerts Created (0)

No Smart Alerts created

7. Click edit, and then decrease the Threshold to 0. Select the Time Threshold option to Everytime a condition triggers a.... and decrease the time to 5 min and save.

Edit Smart Alert

Threshold: Status Code Count \geq 0 Use suggested value 5

Last 24 hours Last 7 days

Status Code Count Threshold Violations

No Alerts

Time Threshold: When do you want to be alerted?

When the condition persists over a specified amount of time

Every time the condition triggers a specified amount of times in a defined time frame

Evaluation Granularity: 5 min 10 min 15 min 20 min 30 min

Cancel Save

8. To verify your alert setup, return to the Stan's Robotshop application on your browser. Once there, go ahead and make a purchase by selecting a robot from the catalog and completing the checkout process, choosing the Canada location as your shipping destination.

By completing this transaction from Canada, we can observe how the application behaves and ensure that the smart alert is triggered correctly when transactions from Canada are detected.

9. After making a purchase from the Canada location in Stan's Robotshop application, you can expect to receive an email alert within the next 5 minutes. This alert will be triggered by the smart alert configuration we set up to monitor transactions from Canada.

Alert Configuration

Details

Threshold Type: Static Threshold Metric: Status Code Count \geq 0 Scope: Per Website

Trigger: Last 24 hours Last 7 days

Alerts Created (1)

2023-07-18, 20:56:02 (active)

Practice exercise

1. Create a widget that displays the number of times payments have been done by each individual user.

▼ Click here for Hint

Choose the below parameters for filters:

- Filter: Add 1st filter, from the dropdown select Configuration>Name and in the value box write robot-shop. Add 2nd filter Location>PageName and write cart.html into the value box.
- Group: Add group Location>Path

2. Create an Alert whenever the e-commerce solution gets transactions from **Nigeria** or **Nambia**.

▼ Click here for Hint

In Select Scope page, add filters for both **Nigeria** and **Nambia**.

Summary

Congratulations! You've successfully developed a Robotshop application and integrated it with Instana using Docker.

During the lab, you explored the functionalities of Instana's dashboards, learning how to create widgets and set up alerts. This hands-on experience has given you valuable insights into monitoring and managing your Robotshop application effectively through the Instana platform.

Author(s)

Pallavi Rai

Contributor(s)

Anamika Agarwal

© IBM Corporation. All rights reserved.