



Hands-On Lab:

Introduction to our Lab Environment

High-Level Overview of our Lab



Your laptop



Cloud Automation Attendee Host

Login with dtu_training / @dtulabs2021
Used to execute keptn CLI commands

Access through web ui, Keptn CLI or API

Connected

<https://hci34192.live.dynatrace.com/>

Dynatrace Software Intelligence Platform

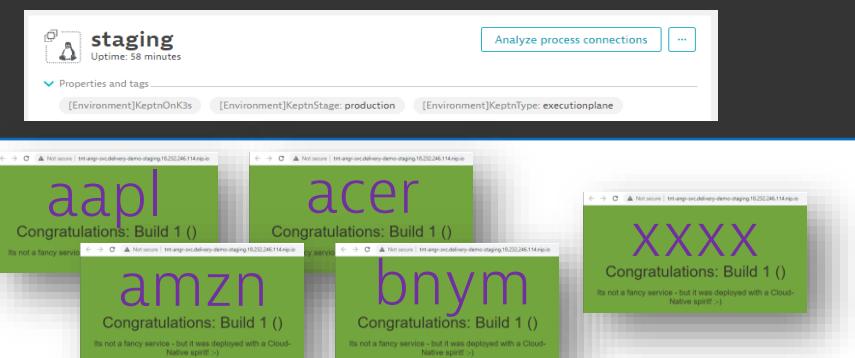
- Monitors our k3s clusters
- Synthetic tests against our services



kubernetes

https://*.delivery-demo-staging.claus-ws-staging.keptn.sh

Execution Plane for STAGING
+ Helm, Monaco, Generic Executor

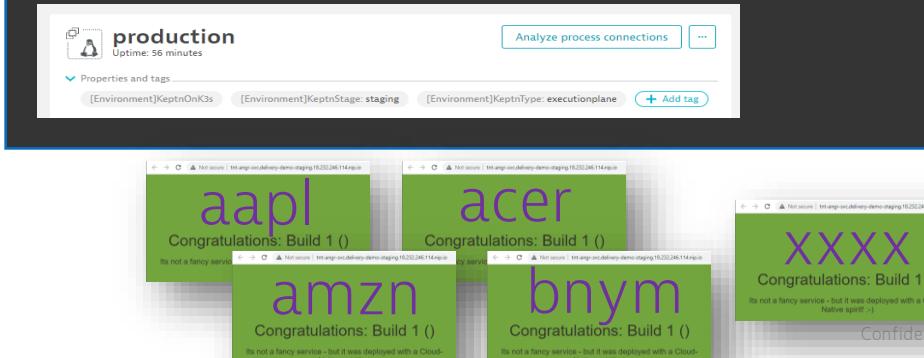


Registered to execute deploy, test, configure, ...

FullStack Monitored with OneAgent

https://*.delivery-demo-production.claus-ws-production.keptn.sh

Execution Plane for PRODUCTION
+ Helm, Monaco, Generic Executor



Our sample app: every attendee has its own application instance in staging & production

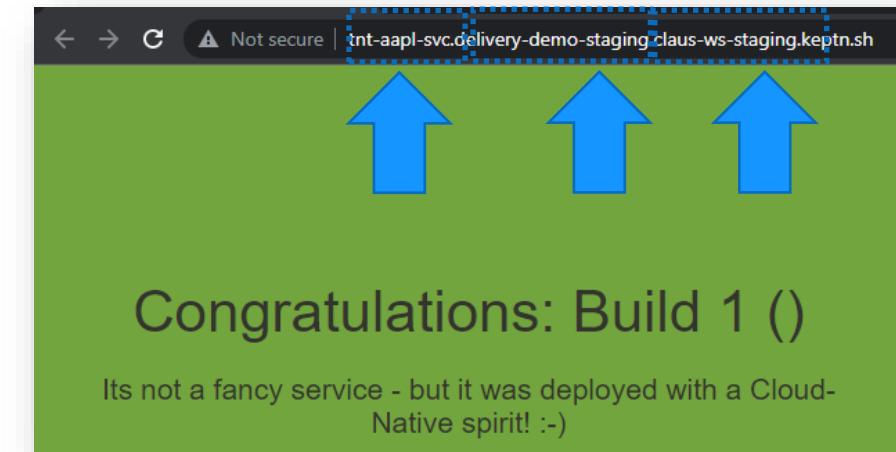
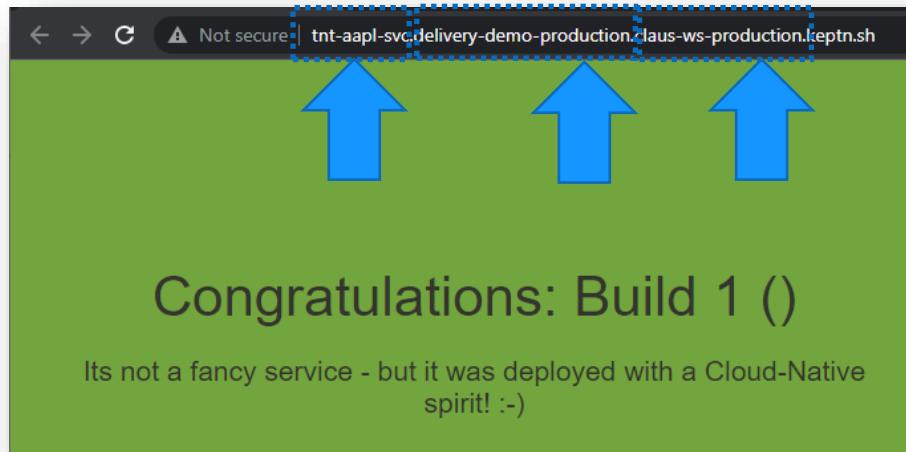
- Deployed through Cloud Automation on our two k8s clusters

The screenshot shows the Cloud Automation interface with a teal header bar containing the text "Cloud Automation / delivery-demo". Below it is a sidebar with icons for Services, Configurations, and Metrics. The main area displays "29 Services" and a list item "tnt-angr-svc" with a yellow bar indicating version 1.0.0. There are tabs for "staging" and "production".

Try to access YOUR app!

- With unique URLs including your own „TenantID“ and environment

- <http://tnt-aapl-svc.delivery-demo-staging.claus-ws-staging.keptn.sh/>
- <http://tnt-aapl-svc.delivery-demo-production.claus-ws-production.keptn.sh/>



Our sample app is automatically monitored with Dynatrace

- Including automatic created Synthetic Checks in Staging and Production
- Proper automated tagging and naming

Tagged synthetic monitor ↗

Filtered by:

		Creation date	Type
<input type="checkbox"/>	Synthetic monitor ▲		
<input type="checkbox"/>	Browser Check - delivery-demo.tnt-aapl-svc.production	Today	Browser clickpath
<input type="checkbox"/>	HTTP Check - delivery-demo.tnt-aapl-svc.production	Today	HTTP
<input type="checkbox"/>	HTTP Check - delivery-demo.tnt-aapl-svc.staging	Today	HTTP

tnt-aapl-svc-*.delivery-demo-staging

Seen recently

Properties and tags

[Environment]DT_APPLICATION_ENVIRONMENT: staging [Environment]DT_APPLICATION_NAME: delivery-demo
[Environment]DT_APPLICATION_RELEASE_VERSION: 1.0.0 [Environment]WorkshopTenant: aapl [Kubernetes]app: tnt-aapl-svc
[Kubernetes]app.kubernetes.io/component: api [Kubernetes]app.kubernetes.io/instance: delivery-demo-staging-tnt-aapl-svc
[Kubernetes]app.kubernetes.io/managed-by: Keptn [Kubernetes]app.kubernetes.io/name: tnt-aapl-svc
[Kubernetes]app.kubernetes.io/part-of: delivery-demo [Kubernetes]app.kubernetes.io/version: 1.0.0 [Kubernetes]helm.sh/chart: simplenode-0.1.0
keptn_deployment: user-managed keptn_project: delivery-demo keptn_service: tnt-aapl-svc keptn_stage: staging + Add tag

Find your app in the Dynatrace Environment

The screenshot displays the Dynatrace environment with two main panels. The top panel shows the 'Services' dashboard with a list of 2 services: 'tnt-aapi-svc-*.delivery-demo-production' and 'tnt-aapi-svc-*.delivery-demo-staging'. The bottom panel shows the 'Release monitoring' overview with sections for 'Release inventory' and 'Release events'.

Find your tenant via the management zone

Or via the WorkshopTenant tag filter

See all deployments in the release overview

Release inventory

Name	Version	Stage	Product	Instances	Throughput
tnt-amzn-svc-*delivery-de...	1.0.0	production	delivery-demo	1	-
tnt-amzn-svc-*delivery-de...	1.0.0	staging	delivery-demo	1	-
tnt-bnmy-svc-*delivery-de...	1.0.0	production	delivery-demo	1	-
tnt-bnmy-svc-*delivery-de...	1.0.0	staging	delivery-demo	1	-
tnt-bofa-svc-*delivery-de...	1.0.0	production	delivery-demo	1	-
tnt-bofa-svc-*delivery-de...	1.0.0	staging	delivery-demo	1	-
tnt-cola-svc-*delivery-de...	1.0.0	production	delivery-demo	1	-
tnt-cola-svc-*delivery-de...	1.0.0	staging	delivery-demo	1	-
tnt-comp-svc-*delivery-de...	1.0.0	production	delivery-demo	1	-
tnt-acr-svc-*delivery-de...	1.0.0	staging	delivery-demo	1	-
tnt-acr-svc-*delivery-de...	1.0.0	production	delivery-demo	1	-
tnt-aapi-svc-*delivery-de...	1.0.0	staging	delivery-demo	1	-
tnt-aapi-svc-*delivery-de...	1.0.0	production	delivery-demo	1	-
helm-service-keptn	0.8.7	keptn	-	1	-

Release events

52 events match your query and filtering

Event Type	Time	Details
Evaluation result: fail	today, 12:27	▼
Evaluation result: fail	today, 12:27	▼
Evaluation result: warning	today, 12:26	▼
Evaluation result: warning	today, 12:26	▼
Evaluation result: warning	today, 12:26	▼

Dynatrace Cloud Automation Environment

- Login with the same Dynatrace user provided

The screenshot shows the Dynatrace Cloud Automation dashboard at the URL `fvk03152.cloudautomation.live.dynatrace.com/bridge/dashboard`. The title bar says "Cloud Automation / Choose project". Below it, there are two project cards:

- delivery-demo**
2 Stages, 30 Services
Shipyard version: 0.2.0
Info: No Git upstream configured. [Set Git upstream](#)
- dynatrace**
1 Stages, 0 Services
Shipyard version: 0.2.0
Info: No Git upstream configured. [Set Git upstream](#)

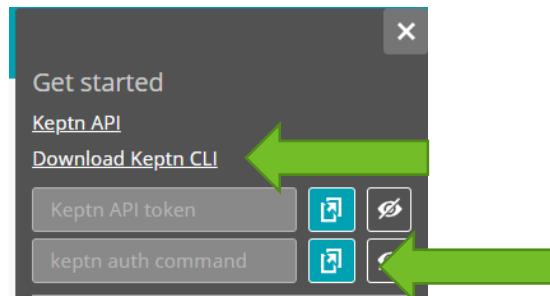
Each project card has a "quality-gate" button at the bottom.

(OPTIONAL) Access Keptn CLI – via Bastion Host or locally on your machine

- Via Bastion Host
 - Connect via Putty (or equivalent SSH client): IP Address is in Excel File
 - Lets run a „*keptn status*“ command: should say its authenticated against fvk03152.xxx

```
[dtu_training@ip-10-0-1-101 ~]$ keptn status
Starting to authenticate
Successfully authenticated against the Keptn cluster https://fvk03152.cloudautomation.live.dynatrace.com/api
Using a file-based storage for the key because the password-store seems to be not set up.
[dtu_training@ip-10-0-1-101 ~]$
```

- Locally on your laptop
 - First install the Keptn CLI
 - Then authenticate it by copying keptn auth from the UI



Quick Status Check: are we all good with accessing our environments?

- Please mark your tasks accordingly in the Excel file

Attendee	How familiar are you with Dynatrace? use it daily; occasionally; never	Workshop Tenant ID	OK to record session?	Claim your ID	Validate successful access of environments		
					Dynatrace Environment? Login with username / pwd given	Cloud Automation SaaS? (login with username / pwd given)	Access YOUR sample tenant app in production and staging?
Andreas Grabner		aapl		Done			

Lab 1: Production Reliability

Creating SLOs and putting them on a dashboard

DevOps & SRE Language: SLIs drive SLOs which inform SLAs!!!

Service Level Indicators (SLIs)

Percentage of an important metric against a criteria

Example: Service Response Time p95 < 400ms

Service Level Objectives (SLOs)

Success-% SLI over a timeframe

Example: p95 < 400ms in 90% of the time over 30 days

Error Budget

How much more impact can we afford before violating SLO?

Service Level Agreements (SLAs)

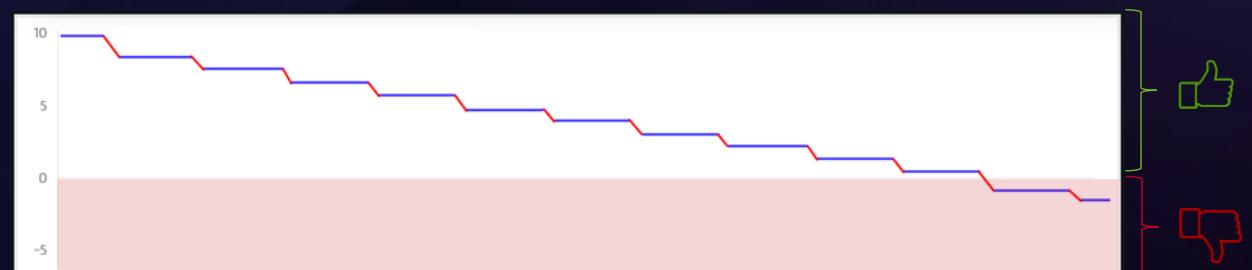
What happens IF SLO is breached

Example: Paying penalties, losing customers ...

Criteria

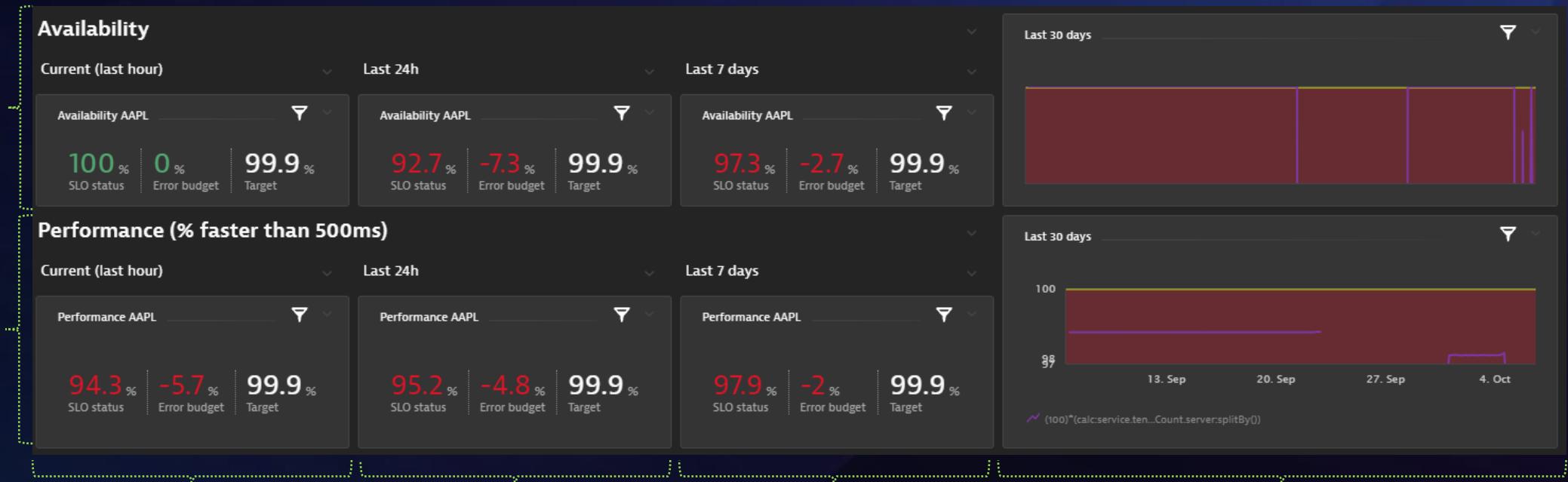



Error Budget: how much budget is left?



A best practice SLO dashboard to start with

% of Time
System is
available



% of Requests
meeting
Performance
Goal

Last 1 hour

Any current issues?

Last 24 hour

Any long running issues?

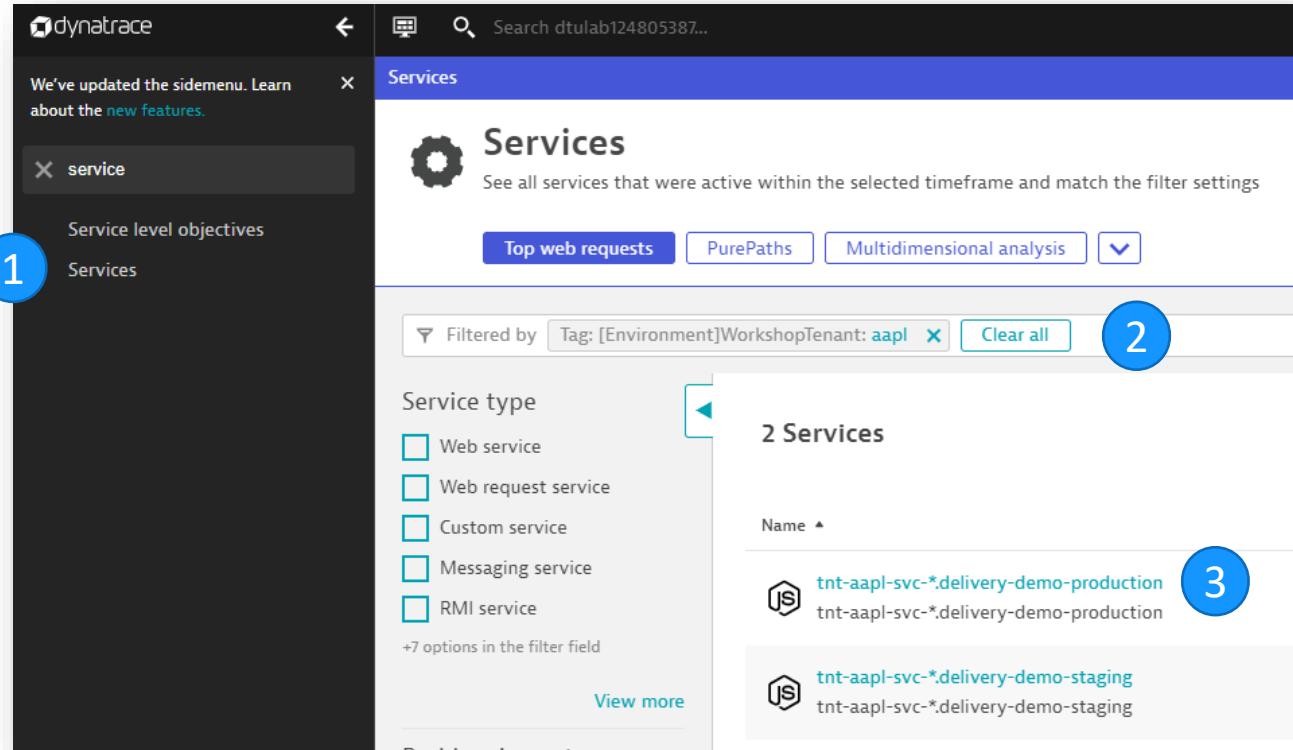
Last 7 days

Any permanent issues?

Last 30 days

Any repeating issues? Will we meet our SLO?

Lets quickly explore our responsible service in production

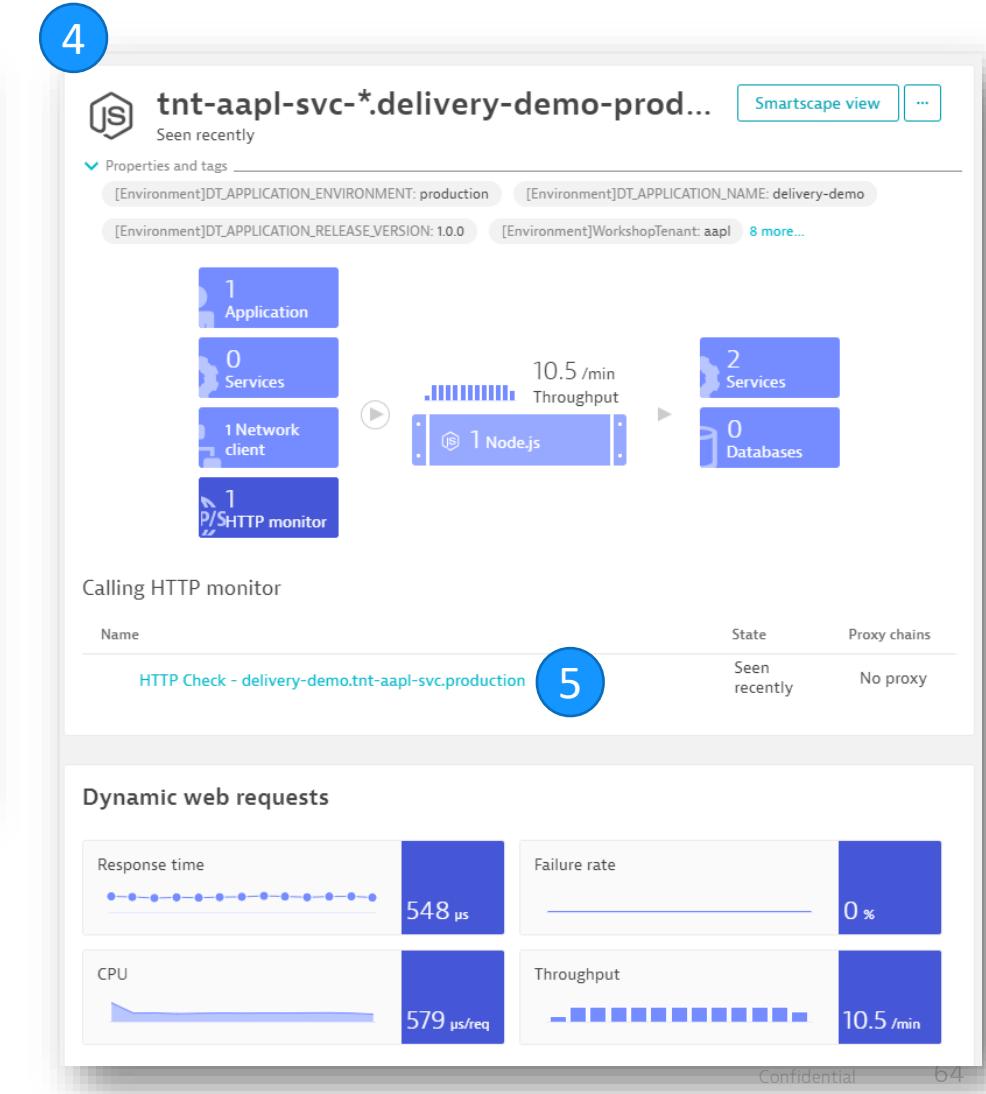


1 We've updated the sidemenu. Learn about the [new features](#).

2 Filtered by Tag: [Environment]WorkshopTenant: aapl

3 Services

- tnt-aapl-svc-*.delivery-demo-production
- tnt-aapl-svc-*.delivery-demo-staging



4 tnt-aapl-svc-*.delivery-demo-production

Properties and tags

- [Environment]DT_APPLICATION_ENVIRONMENT: production
- [Environment]DT_APPLICATION_RELEASE_VERSION: 1.0.0
- [Environment]WorkshopTenant: aapl

Metrics:

- Application: 1
- Services: 0
- Network client: 1
- Node.js: 1
- P/S HTTP monitor: 1

Throughput: 10.5 /min

Calling HTTP monitor

Name	State	Proxy chains
HTTP Check - delivery-demo.tnt-aapl-svc.production	Seen recently	No proxy

5

Dynamic web requests

Response time	Failure rate
548 µs	0 %

CPU	Throughput
579 µs/req	10.5 /min

Step #1: Lets create an SLO for production availability

- Lets create our own SLO based on your Synthetic Test Availability of our tenant in Production

The screenshot shows the Dynatrace interface for creating a new Service-level objective (SLO). The process is divided into five steps:

- Step 1:** The main dashboard shows a summary of 60 service-level objectives. A blue circle labeled "1" points to the "Service level objectives" button in the sidebar.
- Step 2:** The "Service-level objectives" page is displayed, featuring a "Service-level objectives" section with a brief description and a "Add new SLO" button. A blue circle labeled "2" points to this button.
- Step 3:** A modal window titled "Monitor a client-side objective" is open. It contains a "User experience" section with a magnifying glass icon, a "Mobile crash-free users" section, and a "Synthetic availability" section (which is highlighted with a blue border). A blue circle labeled "3" points to the "Synthetic availability" section. Below it, there's a text input field with a placeholder and a code snippet example: "Click on one of the templates above or enter a metric expression including math operations to" followed by "1 (builtin:synthetic.browser.availability.location.total:splitBy())".
- Step 4:** An "Entity selector" dialog is shown, containing a text input field with the query "type("SYNTHETIC_TEST"),tag("tnt-xxxx-svc"),tag("production")" and a "Preview" button. A blue circle labeled "4" points to this dialog.
- Step 5:** A "Define the SLO target" section is shown, featuring a horizontal bar chart with three segments: "Failure" (red), "Warning" (yellow), and "Good" (green). Below the chart, there are sections for "Target" (set at 99.98%), "Warning" (set at 99.99%), and "Timeframe" (set to "-1w"). A blue circle labeled "5" points to this section.

Hint: Copy queries and tags from slide notes!

Step #2: Lets create an SLO for production performance

- Let's create an SLO that measures the % of requests faster than 500ms on our tenant
 - Select „Service-level Availability“ and replace nominator metric with `calc:service.tenant.responsetime.count.faster500ms`

1 **Add new SLO**

2 **Select your indicators**

Monitor a service-level availability objective
Define and measure an objective for the availability of a service or a group of services.

Service-level availability **Service-method availability**

Click on one of the templates above or enter a metric expression including math operations to express your 0-100% normalized SLI. Possible metric keys
`1 (100)*(calc:service.tenant.responsetime.count.faster500ms:splitBy())/(builtin:service.requestCount.server:splitBy())`

Name this service-level availability SLO
Performance SLO for xxxx

3 **Entity selector**

```
type("SERVICE"),tag("[Environment]WorkshopTenant:xxxx"),
tag("[Environment]DT_APPLICATION_ENVIRONMENT:production")
```

Matching entities

Entity ID	Display Name
SERVICE-64B291A615754F6B	tnt-aapl-svc-*.delivery-demo-production

4 **Define the SLO target**

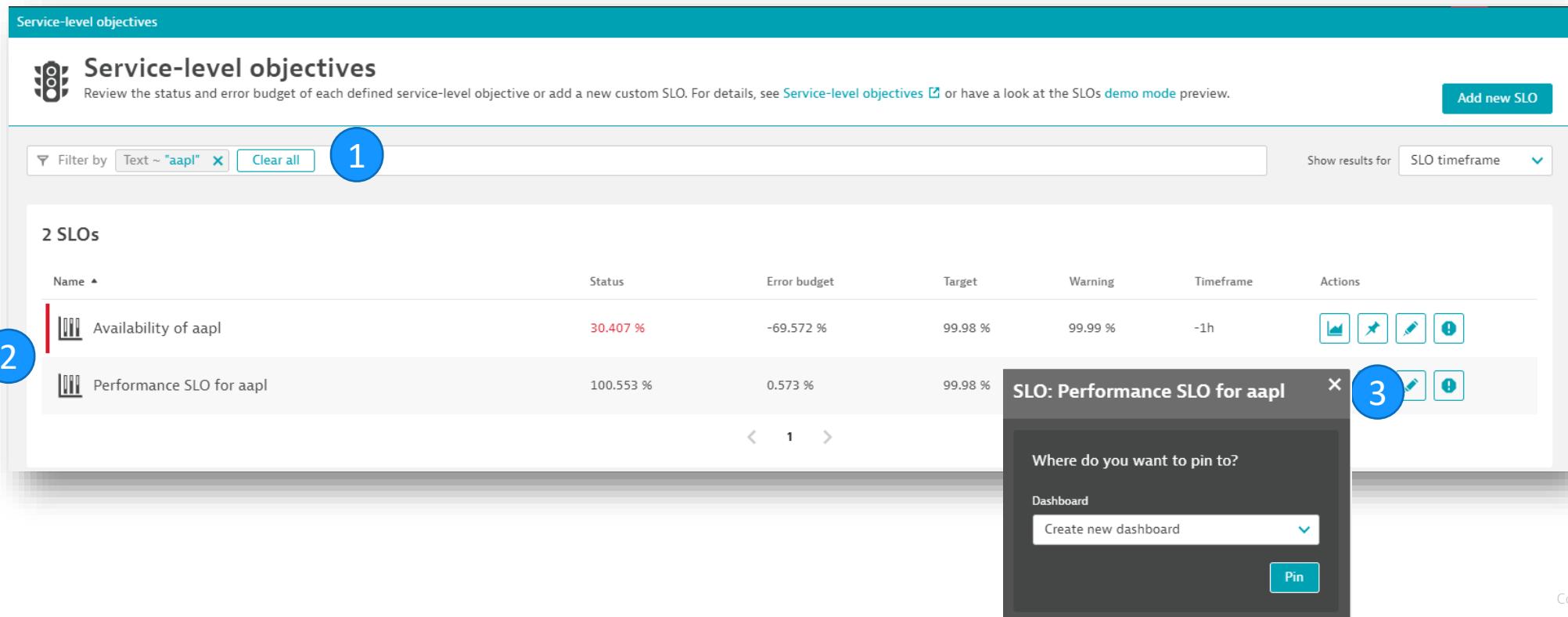
Failure **Target** 99.98 % Warning 99.99 % Good 100%

Timeframe -1h

Hint: Copy queries and tags from slide notes!

Step #3: Validate our SLOs are created and delivery data

1. Filter by name, e.g: xxxx to find your SLOs
2. Validate you see your SLOs
3. Start creating a new dashboard



Service-level objectives

Service-level objectives

Review the status and error budget of each defined service-level objective or add a new custom SLO. For details, see [Service-level objectives](#) or have a look at the SLOs [demo mode](#) preview.

Add new SLO

Filter by Text ~ "aapl" Clear all Show results for SLO timeframe

2 SLOs

Name	Status	Error budget	Target	Warning	Timeframe	Actions
Availability of aapl	30.407 %	-69.572 %	99.98 %	99.99 %	-1h	   
Performance SLO for aapl	100.553 %	0.573 %	99.98 %			   

Where do you want to pin to?

Dashboard Create new dashboard

Pin

Step #4 – Clone the existing dashboard template

Dashboards

Dashboards

Overview of all dashboards you are permitted to view or edit.

Please provide feedback and find planned enhancements at [Dynatrace answers](#).

Show all tenant dashboards (for admin users only)

Filter by Name ~ "xxxx"

Ownership

Any
 Mine
 Shared with me

Favorite

Any
 Yes

3 Dashboards

Favorite	Name	Modified at	Owner	Action
★	SLO Dashboard tnt-xxxx-svc Preset	Oct 08 07:54	andreas.grabner@dynatrace.com	 
★	KQG;project=deli demo;stage=staging;service=tnt-xxxx-svc Preset	Oct 08 07:54	andreas.grabner@dynatrace.com	
	SLO Dashboard tnt-xxxx-svc Preset	Oct 12 07:44	andreas.grabner@dynatrace.com	

CLONE 

SLO Dashboard tnt-xxxx-svc

Step #5: Customize it for your tenant

- Edit the settings necessary to pick up the data from your tenant

The screenshot shows the 'Edit dashboard' interface for a 'SLO Dashboard tnt-xxxx-svc'. The dashboard itself displays Service Level Objectives (SLOs) for the 'AAPL' service across different timeframes: Current (last hour), Last 24h, Last 7 days, and Last 30 days. The availability SLOs show 100% status, 0% error budget, and 99.9% target. The performance SLOs show 94.2% status, -5.8% error budget, and 99.9% target.

Give it a good name (green arrow pointing to the dashboard title)

Select your Availability SLOs and select correct timeframe (green arrow pointing to the availability section)

Select your Performance SLOs and select correct timeframe (green arrow pointing to the performance section)

Select your management zone (green arrow pointing to the 'Tenant' dropdown in the sidebar)

Additional links: Release management, Release validation

One tag applied

Service Level Objectives for tnt-xxxx-svc

Availability

Performance (% faster than 500ms)

Default filters

Tiles **Settings**

Below are a few quick configuration options for your dashboard.

Advanced settings

Default timeframe

Default management zone

Tenant: tnt-aapl-svc

Reports

Subscribers will receive a link to view this dashboard without signing in.

View documentation

Enable reports

From Manual to Automated Creation of SLOs & Dashboard with Monaco

- Monaco (Monitoring as Code): <https://dynatrace-oss.github.io/dynatrace-monitoring-as-code/>

```
export OWNER=youremail@domain.com
export TENANT=aapl
monaco -e environment.yaml -p lab1 projects
```

```
! slo.yaml  ! slo.json  x
keptn-on-k3s > cloudautomation > monaco > projects > lab1 > slo > ! slo.json > ...
1  {
2      "enabled": true,
3      "name": "{{ .name }}",
4      "description": "{{ .description }}",
5      "useRateMetric": true,
6      "metricExpression": "{{ .expression }}",
7      "metricRate": "",
8      "metricDenominator": "",
9      "metricNumerator": "",
10     "evaluationType": "AGGREGATE",
11     "filter": "{{ .filter }}",
12     "target": "{{ .target }}",
13     "warning": "{{ .warning }}",
14     "timeframe": "{{ .timeframe }}"
15 }
```

```
! slo.yaml  x  ! slo.json
keptn-on-k3s > cloudautomation > monaco > projects > lab1 > slo > ! slo.yaml > [ ] performanceSlo > timeframe
1 config:
2   - availabilitySlo: "slo.json"
3   - performanceSlo: "slo.json"
4
5 availabilitySlo:
6   - name: "Availability of {{ .Env.TENANT_ID }}"
7   - description: "% of time {{ .Env.TENANT_ID }} service is available based on synthetic test"
8   - expression: "(builtin:synthetic.browser.availability.location.total:splitBy())"
9   - filter: "mzName('{{ Tenant: tnt-{{ .Env.TENANT_ID }}-svc }}'),type('{{ 'SYNTHETIC_TEST' }}')"
10  - target: "95.00"
11  - warning: "99.00"
12  - timeframe: "-1d"
13
14 performanceSlo:
15   - name: "Performance SLO of {{ .Env.TENANT_ID }}"
16   - description: "% of requests handled by {{ .Env.TENANT_ID }} service faster than 500ms"
17   - expression: "(100)*(calc:service.tenant.responseTime.count.faster500ms:splitBy())/(builtin:service.responseTime.count)"
18   - filter: "mzName('{{ Tenant: tnt-{{ .Env.TENANT_ID }}-svc }}'),type('{{ 'SERVICE' }}'),tag('{{ Env.TENANT_ID }}')"
19   - target: "85.00"
20   - warning: "90.00"
21   - timeframe: "-1d"
```

Quick Status Check: are we all good with accessing our environments?

- Please mark your tasks accordingly in the Excel file

Production Reliability		
Created the SLO based on Synthetic Availability	Created SLO based on Response Time	Created an SLO dashboard

Lab 2: Release Validation

Automating release validation through an SLO Dashboard

Reminder: Release monitoring overview based on meta data

Releases

Release monitoring

Overview of deployed component versions and release events. For details, see [Release monitoring](#) or [activate demo mode](#) to view sample data.

Filter by

Release inventory

66 Releases

Name	Release version	Build version	Stage	Product	Instances	Throughput
simplenode-*demo-delivery...	3.0.1	-	dev	demo-delivery	1	-
simplenode-*demo-delivery...	3.0.1	-	staging	demo-delivery	1	-
simplenode-*demo-twostage...	3.0.0	-	production	demo-twostage-...	1	-
simplenode-*demo-twostage...	3.0.0	-	staging	demo-twostage-...	1	-
simplenode-primary-*demo...	3.0.1	-	production	demo-delivery	1	-
simplenode-prod-*demo-rol...	1.0.0	-	prod	demo-rollout	5	-
simplenode-staging-*demo-...	1.0.0	-	staging	demo-rollout	1	-
tnt-aapl-svc-*delivery-dem...	1.0.1	-	production	delivery-demo	2	-
tnt-aapl-svc-*delivery-dem...	1.0.1	-	staging	delivery-demo	2	-
tnt-acer-svc-*delivery-dem...	1.0.1	-	production	delivery-demo	2	-
tnt-acer-svc-*delivery-dem...	1.0.1	-	staging	delivery-demo	2	-
tnt-amzn-svc-*delivery-de...	1.0.1	-	production	delivery-demo	2	-
tnt-amzn-svc-*delivery-de...	1.0.1	-	staging	delivery-demo	2	-
tnt-bnym-svc-*delivery-de...	1.0.1	-	production	delivery-demo	2	-

Release events

193 events match your query and filtering

122 Custom infos 71 Deployments

Events	Time	Details
Deploy tnt-yumb-svc 1.0.1 with strategy user_ma...	today, 11:29	▲
Entity	tnt-yumb-svc-*.delivery-demo-production (tnt-yumb-svc-7ff4bfc766-d5t2x)	
Time	today, 11:29	
CI		☒
deploymentURI[Public]	http://tnt-yumb-svc.delivery-demo-production.claus-ws...	
DtCreds		
Image	grabnerandi/simplenodeservice	
Keptn Service	helm-service	
Keptn Context	e1e2e48b-e226-42b2-87c6-fd003d123823	

„To Keep or Not?“ - Automate Release Validation

Automated Release Validation based on Production SLOs, Dynatrace detected problems and leading indicators

This dashboard will automatically be analyzed as part of your production deployment automation. To see what's currently deployed check the [Releases Overview](#). For all individual check out [Cloud Automation Heatmaps](#)

TODO: Clone dashboard, select your Management Zone, Add your relevant SLO, replace XXXX with your tenant name

Release Validation Criteria: KQG.Total.Pass=90%;KQG.Total.Warning=70%;KQG.Compare.WithScore=pass;KQG.Compare.Results=1;KQG.Compare.Function=avg

Availability AAPL

96.486 % | -3.494 % | 99.98 %

SLO status | Error budget | Target

Good (Green) | Warning (Yellow) | Bad (Red) | No Data (Grey)

Performance AAPL

97.528 % | -2.452 % | 99.98 %

SLO status | Error budget | Target

Good (Green) | Warning (Yellow) | Bad (Red) | No Data (Grey)

Problems

2 / 8

Additional important SLIs & SLOs to validate a healthy production deployment of your app

Service Performance (SLI/SLO)	Service Errors & Throughput (SLI/SLO)	Process Metrics (SLI)
Response time (P95);sli=svc_rt_p95;pass=<+10%,<600 890µs Response time (95 th percentile)	Failure Rate (Avg);sli=svc_fr;pass=<+10%,<2 0% Failure rate (server side errors) (Average)	Process CPU;sli=proc 0.01% Process CPU usag
Response time (P90);sli=svc_rt_p90;pass=<+10%,<550 641µs Response time (90 th percentile)	Throughput (per min);sli=svc_tp_min;pass=<+10%,<200 12.4/min Request count (Sum)	Process Memory;sli=p 85.7 MB Process memory

evaluation

Labels: [Dashboard Link](#) DtCreds: dynatrace

2021-10-06 15:53

[Heatmap](#) [Chart](#)

Score	Availability_AAPL	host_cpu	host_disk_queue	host_mem	Performance_AAPL	problems	proc_count	process_cpu	process_memory	svc_fr	svc_rt_p50	svc_rt_p90	svc_rt_p95	svc_tp_min	svc2svc_calls
Availability_AA	Red	Red	Red	Red	Red	Green	Green	Green	Green	Red	Red	Red	Red	Red	Red
host_cp	Red	Red	Red	Red	Red	Green	Green	Green	Green	Red	Red	Red	Red	Red	Red
host_dq	Red	Red	Red	Red	Red	Green	Green	Green	Green	Red	Red	Red	Red	Red	Red
host_m	Red	Red	Red	Red	Red	Green	Green	Green	Green	Red	Red	Red	Red	Red	Red
Perf_AA	Red	Red	Red	Red	Red	Green	Green	Green	Green	Red	Red	Red	Red	Red	Red
prob	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green
proc_c	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green
proc_m	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green	Green
svc_f	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red
svc_r_p50	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red
svc_r_p90	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red
svc_r_p95	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red
svc_tp_m	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red
svc2sv	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red	Red

[Collapse to 10 SLIs](#)

After you login to Cloud Automation you see several projects

Cloud Automation / Choose project ▾

4 Projects

delivery-demo
2 Stages, 20 Services
Shipyard version: 0.2.0
i No Git upstream configured. [Set Git upstream](#)

staging production

Recent sequences:

- ☒ [delivery](#) of [tnt-nike-svc](#) failed today at 13:58
- ☒ [delivery](#) of [tnt-nflx-svc](#) failed today at 13:57
- ☒ [delivery](#) of [tnt-msft-svc](#) failed today at 13:57
- ☒ [delivery](#) of [tnt-intc-svc](#) failed today at 13:57
- ☒ [delivery](#) of [tnt-goog-svc](#) failed today at 13:56

devopstools
1 Stages, 2 Services
Shipyard version: 0.2.0
i No Git upstream configured. [Set Git upstream](#)

production

Recent sequences:

- ☒ [delivery](#) of [keptnwebservice](#) succeeded last Thursday at 22:53
- ☒ [delivery](#) of [keptnwebservice](#) failed last Thursday at 22:42
- ☒ [delivery](#) of [keptnwebservice](#) succeeded last Thursday at 15:47

dynatrace
1 Stages, 0 Services
Shipyard version: 0.2.0
i No Git upstream configured. [Set Git upstream](#)

quality-gate

Recent sequences:

- i No sequences have been executed yet.

release-validation
1 Stages, 20 Services
Shipyard version: 0.2.0
i No Git upstream configured. [Set Git upstream](#)

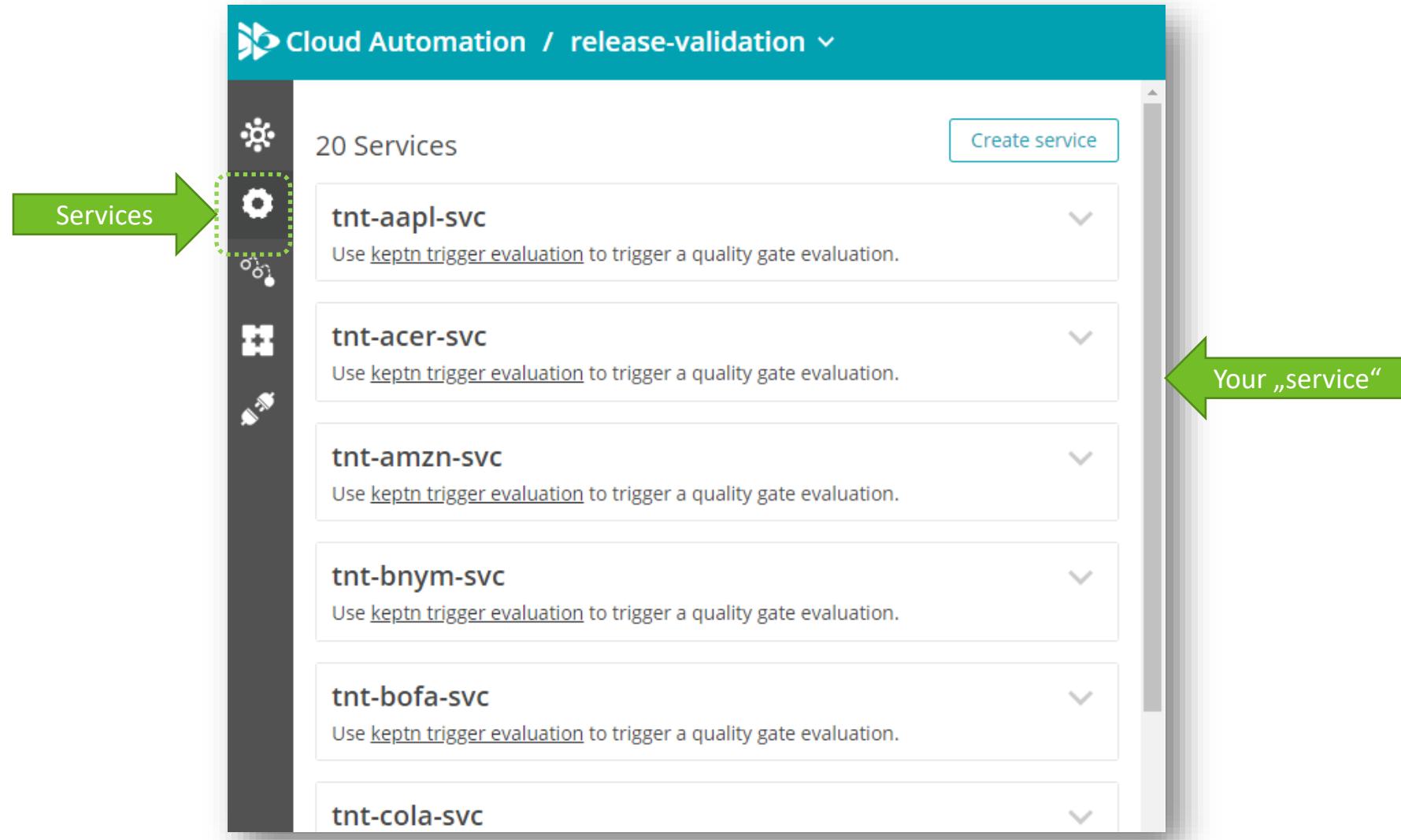
production

Recent sequences:

- i No sequences have been executed yet.

I have created one we can use for „release-validation“

I have created „services“ already for all our tnt-XXXX-svc



Cloud Automation / release-validation

20 Services

Create service

Services

tnt-aapl-svc
Use [keptn trigger evaluation](#) to trigger a quality gate evaluation.

tnt-acer-svc
Use [keptn trigger evaluation](#) to trigger a quality gate evaluation.

tnt-amzn-svc
Use [keptn trigger evaluation](#) to trigger a quality gate evaluation.

tnt-bnym-svc
Use [keptn trigger evaluation](#) to trigger a quality gate evaluation.

tnt-bofa-svc
Use [keptn trigger evaluation](#) to trigger a quality gate evaluation.

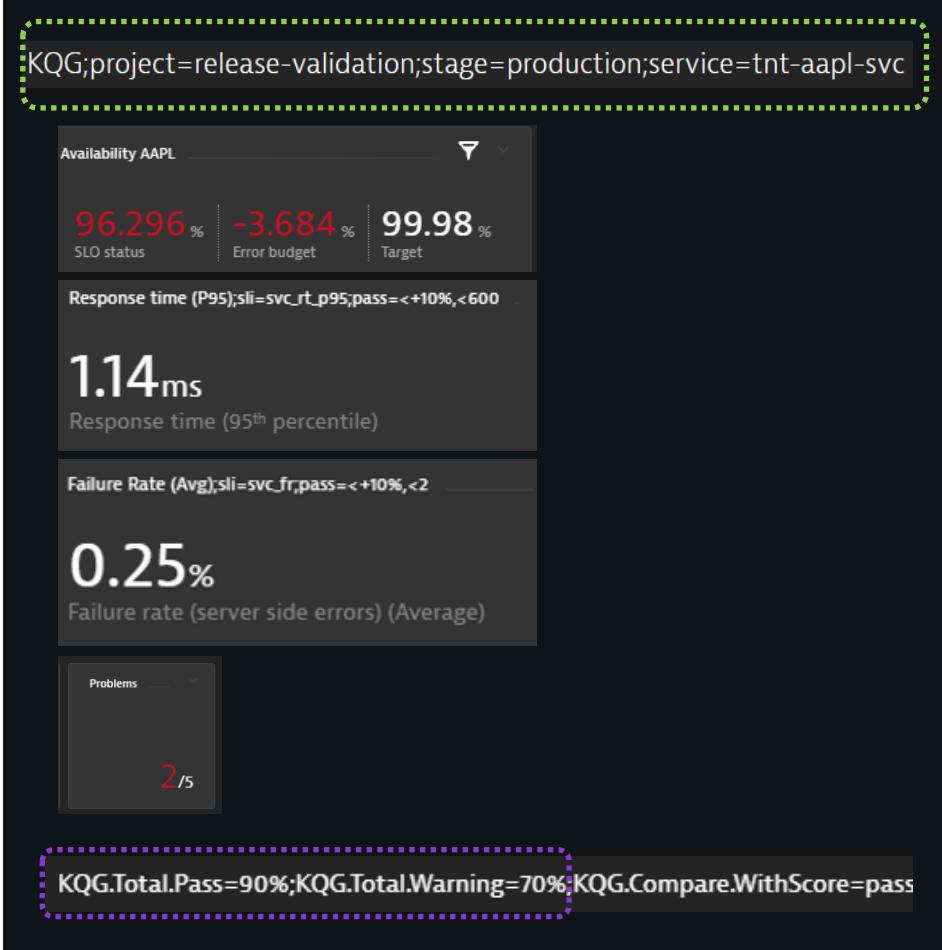
tnt-cola-svc

Your „service“

How does dashboards get analyzed when used in automation?

Dashboard name links to Cloud Automation

KQG;project=xxx,stage=xxx,service=xxx



Pulled Value via API
and compared against

Points
Normalized to 100

Availability: 96.296
<=99.98

0

Response Time: 1.14
<600 and <=+10% increase

25

Failure Rate: 0.25
< 2 and <=+10% increase

25

Open Problems: 2 🔑
Always compared against 0

0

Total Score
Compared based on pass/warn

50

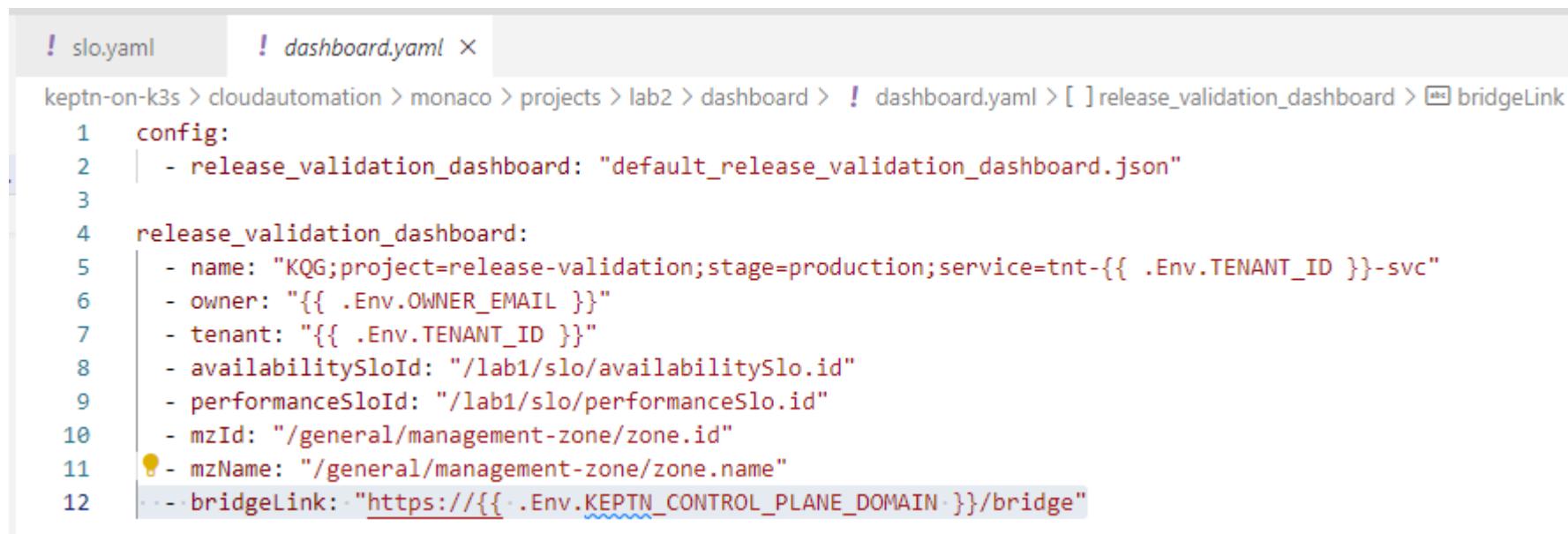
Now lets create a dashboard that we can use for release validation automation

- The automation will look for a Dynatrace dashboard with the following naming schema:
 - KQG;project=<PROJECT>;stage=<STAGE>;service=<SERVICE>
- In our case *project=release-validation*, *stage=production* and *service=tnt-xxxx-svc*
- We could start from scratch, or
 - start from a template 😊
 - Use Monaco 😊

Using Monaco to create release validation dashboard

- Monaco (Monitoring as Code): <https://dynatrace-oss.github.io/dynatrace-monitoring-as-code/>

```
export OWNER=youremail@domain.com
export TENANT=aapl
monaco -e environment.yaml -p lab2 projects
```



```
! slo.yaml ! dashboard.yaml ×

keptn-on-k3s > cloudatautomation > monaco > projects > lab2 > dashboard > ! dashboard.yaml > [ ]release_validation_dashboard > bridgeLink

1 config:
2   - release_validation_dashboard: "default_release_validation_dashboard.json"
3
4 release_validation_dashboard:
5   - name: "KQG;project=release-validation;stage=production;service=tnt-{{ .Env.TENANT_ID }}-svc"
6   - owner: "{{ .Env.OWNER_EMAIL }}"
7   - tenant: "{{ .Env.TENANT_ID }}"
8   - availabilitySloId: "/lab1/slo/availabilitySlo.id"
9   - performanceSloId: "/lab1/slo/performanceSlo.id"
10  - mzId: "/general/management-zone/zone.id"
11  - mzName: "/general/management-zone/zone.name"
12  - bridgeLink: "https://{{ .Env.KEPTN_CONTROL_PLANE_DOMAIN }}/bridge"
```

Step 1 – Clone the existing dashboard template

Dashboards

Dashboards

Overview of all dashboards you are permitted to view or edit.

Please provide feedback and find planned enhancements at [Dynatrace answers](#).

Show all tenant dashboards (for admin users only)

Filter by Name ~ "xxxx"

Search for xxxx in all tenant dashboards

Ownership

- Any
- Mine
- Shared with me

Favorite

- Any
- Yes
- No

Owner

- Any

3 Dashboards

Favorite	Name	Modified at	Owner
★	SLO Dashboard for Tenant xxxx	Nov 18 15:59	andreas.grabner@dynatra
★	KQG;project=delivery-demo;stage=staging;service=tnt-xxxx-svc	Nov 18 15:59	andreas.grabner@dynatra
★	KQG;project=release-validation;stage=production;service=tnt-xxxx-svc	Nov 22 16:08	

Import dashboard Create dashboard

CLONE

KQG;project=release-validation;stage=production;service=tnt-xxxx-svc

Step 2 – Give it proper name, change MZs and add your SLOs

The screenshot shows the Dynatrace dashboard editor interface. On the left is the dashboard preview with various tiles and filters. On the right is the configuration panel.

- Replace xxxx with your service:** A green arrow points to the URL bar where 'xxxx' is highlighted, with the instruction to replace it with your service name.
- Make sure your SLOs are used:** A green arrow points to the 'Release Validation Criteria' section, which displays two SLO cards for 'Availability AAPL' and 'Performance AAPL'. It highlights the 'SLO status' and 'Error budget' columns.
- Select your MZ:** A green arrow points to the 'Tenant' dropdown in the configuration panel, with the instruction to select your Management Zone.

Dashboard Preview (Left):

- URL:** KQG;project=release-validation;stage=production;service=tnt-aapl-svc
- Tags:** tnt-aapl-svc, validation, + Add tag
- Title:** Automated Release Validation based on Production SLOs, Dynatrace detected problems and leading indicators
- Description:** This dashboard will automatically be analyzed as part of your production deployment automation. To see what's currently deployed check the [Releases Overview](#). For all individual check out [Logs](#).
- TODO:** Clone dashboard, select your Management Zone, Add your relevant SLO, replace XXXX with your tenant name.
- Release Validation Criteria:** KQG.Total.Pass=90%;KQG.Total.Warning=70%;KQG.Compare.WithScore=pass;KQG.Compare.Results=1;KQG.Compare.Function=avg
- Tiles:** Availability AAPL (96.291%, -3.689%, 99.98%), Performance AAPL (97.431%, -2.549%, 99.98%), and a third tile showing 0.
- Additional important SLIs & SLOs:** Service Performance (SLI/SLO) showing 3.95ms (Response time (95th percentile)), Service Errors & Throughput (SLI/SLO) showing 0.25% (Failure rate (server side errors) (Average)), Process Metrics (SLI/SLO) showing 0.01% (Process CPU usage (Average)), and Host-based showing 0.9% (CPU usage).

Configuration Panel (Right):

- Edit dashboard:** Done
- Tiles** (selected)
- Settings**
- Below are a few quick configuration options for your dashboard.**
- Advanced settings**
- Default filters:**
 - Default timeframe
 - Default management zone
- Tenant:** tnt-aapl-svc
- Reports:**
 - Enable reports
- Subscribers:** Subscribers will receive a link to view this dashboard without signing in.
[View documentation](#)

Step 3 – Save the dashboard!

Search Dynatrace - hci34192: hci34192...

Tenant: tnt-aapl-svc

KQG;project=release-validation;stage=production;service=tnt-aapl-svc

2 tags applied

tnt-aapl-svc validation

Edit ...

Automated Release Validation based on Production SLOs, Dynatrace detected problems and leading indicators

This dashboard will automatically be analyzed as part of your production deployment automation. To see what's currently deployed check the [Releases Overview](#). For all individual check out [Cloud Automation Heatmaps](#)

TODO: Clone dashboard, select your Management Zone, Add your relevant SLO, replace XXXX with your tenant name

Release Validation Criteria: KQG.Total.Pass=90%;KQG.Total.Warning=70%;KQG.Compare.WithScore=pass;KQG.Compare.Results=1;KQG.Compare.Function=avg

Availability AAPL
96.291% | -3.689% | 99.98%
SLO status | Error budget | Target
Good Warning Bad No Data

Performance AAPL
97.431% | -2.549% | 99.98%
SLO status | Error budget | Target
Good Warning Bad No Data

Problems
0

Additional important SLIs & SLOs to validate a healthy production deployment of your app

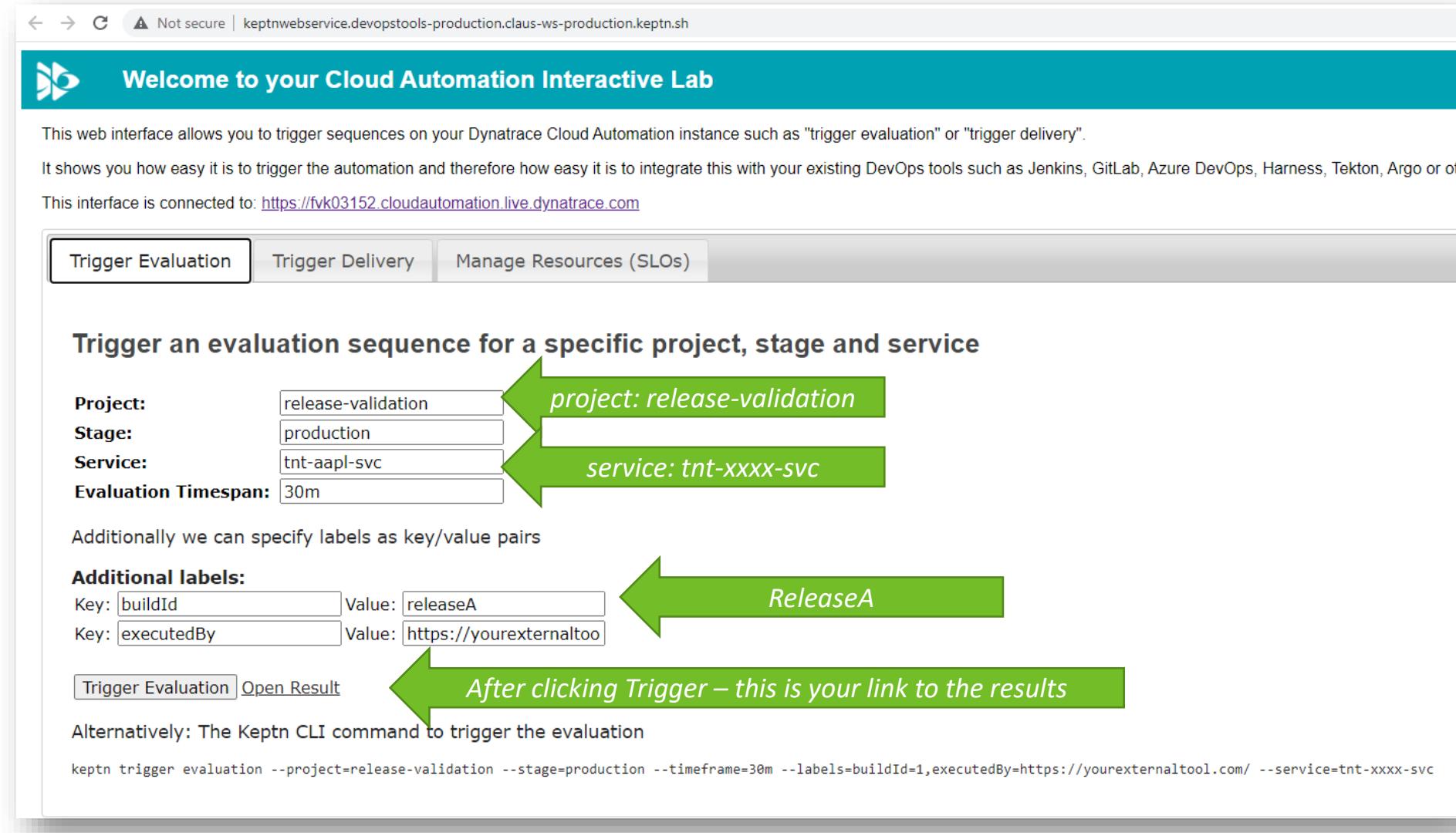
Service Performance (SLI/SLO)
Response time (P95);sli=svc_rt_p95;pass=<+10%,<600
3.95ms
Response time (95th percentile)

Service Errors & Throughput (SLI/SLO)
Failure Rate (Avg);sli=svc_fr;pass=<+10%,<2
0.25%
Failure rate (server side errors) (Average)

Process Metrics (SLI/SLO)
Process CPU;sli=process_cpu;pass=<20;warning=<50;k...
0.01%
Process CPU usage (Average)

Host-based (SLI/SLO)
Host CPU %;sli=host_cpu;pass=<20;warning=<50;k...
0.92%
CPU usage % (Average)

Step 4a: Trigger through our „Simple CI/CD Tool“



Not secure | keptnwebservice.devopstools-production.claus-ws-production.keptn.sh

Welcome to your Cloud Automation Interactive Lab

This web interface allows you to trigger sequences on your Dynatrace Cloud Automation instance such as "trigger evaluation" or "trigger delivery". It shows you how easy it is to trigger the automation and therefore how easy it is to integrate this with your existing DevOps tools such as Jenkins, GitLab, Azure DevOps, Harness, Tekton, Argo or other CI/CD tools.

This interface is connected to: <https://fvk03152.cloudautomation.live.dynatrace.com>

Trigger Evaluation **Trigger Delivery** **Manage Resources (SLOs)**

Trigger an evaluation sequence for a specific project, stage and service

Project: *project: release-validation*

Stage: *service: tnt-xxxx-svc*

Service:

Evaluation Timespan:

Additionally we can specify labels as key/value pairs

Additional labels:

Key: Value: *ReleaseA*

Key: Value:

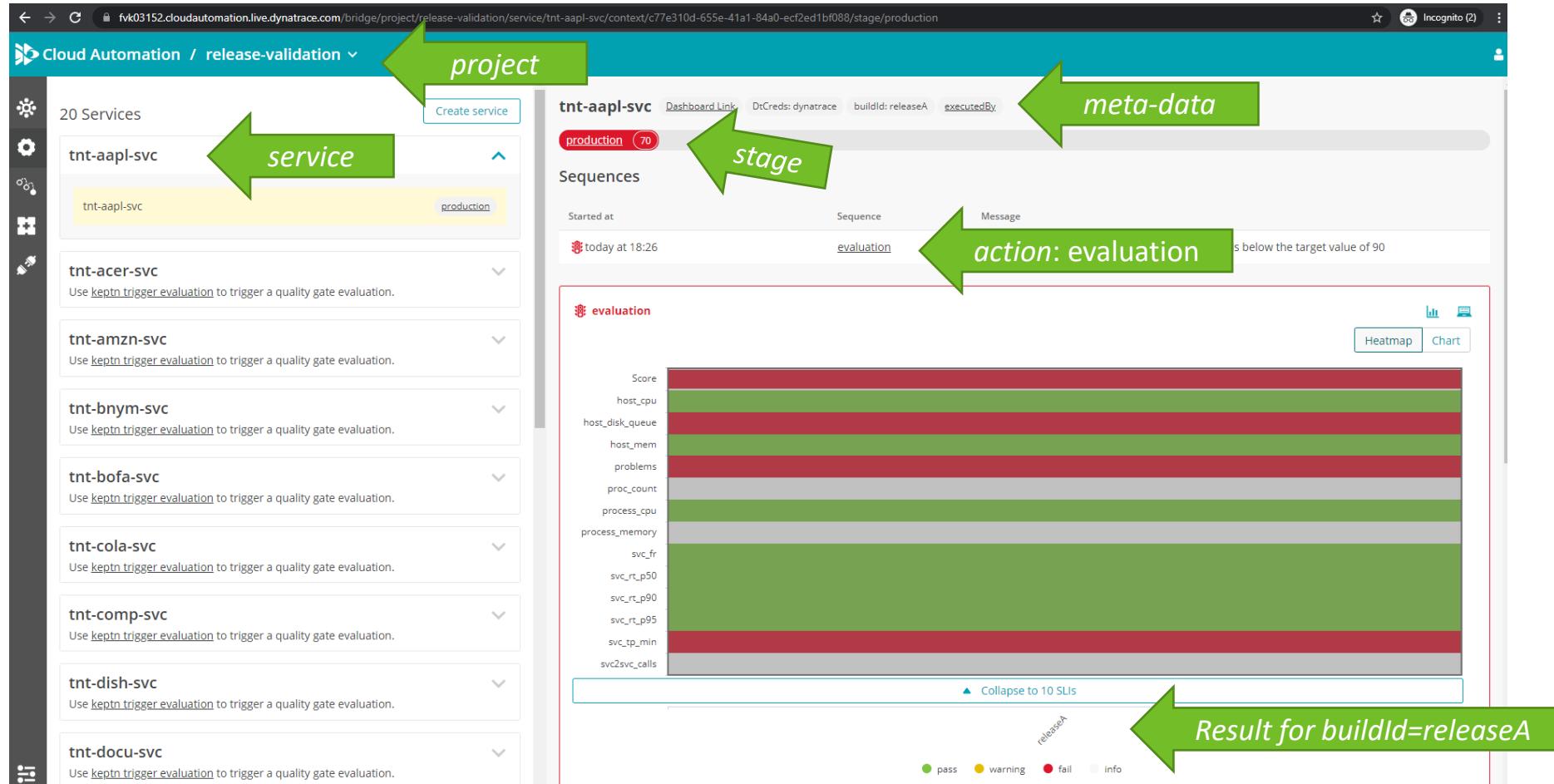
Trigger Evaluation **Open Result**

After clicking Trigger – this is your link to the results

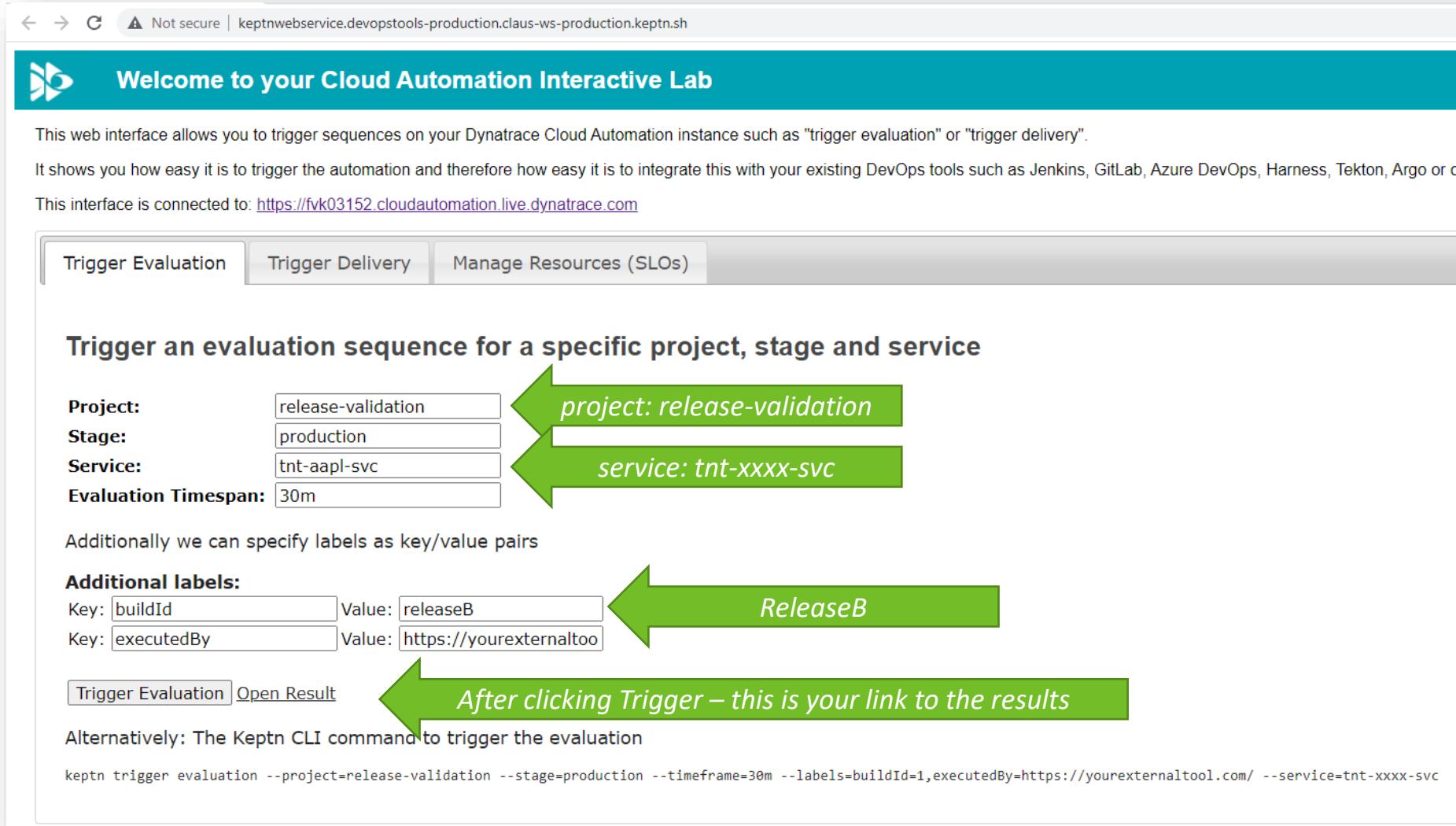
Alternatively: The Keptn CLI command to trigger the evaluation

```
keptn trigger evaluation --project=release-validation --stage=production --timeframe=30m --labels=buildId=1,executedBy=https://yourexternaltool.com/ --service=tnt-xxxx-svc
```

Step 4a: Analyze result for „Release A”



Step 4a: Trigger through our „Simple CI/CD Tool“



This web interface allows you to trigger sequences on your Dynatrace Cloud Automation instance such as "trigger evaluation" or "trigger delivery". It shows you how easy it is to trigger the automation and therefore how easy it is to integrate this with your existing DevOps tools such as Jenkins, GitLab, Azure DevOps, Harness, Tekton, Argo or other. This interface is connected to: <https://fvk03152.cloudautomation.live.dynatrace.com>

Trigger Evaluation **Trigger Delivery** **Manage Resources (SLOs)**

Trigger an evaluation sequence for a specific project, stage and service

Project: release-validation *project: release-validation*

Stage: production

Service: tnt-aapl-svc *service: tnt-xxxx-svc*

Evaluation Timespan: 30m

Additionally we can specify labels as key/value pairs

Additional labels:

Key: Value: *ReleaseB*

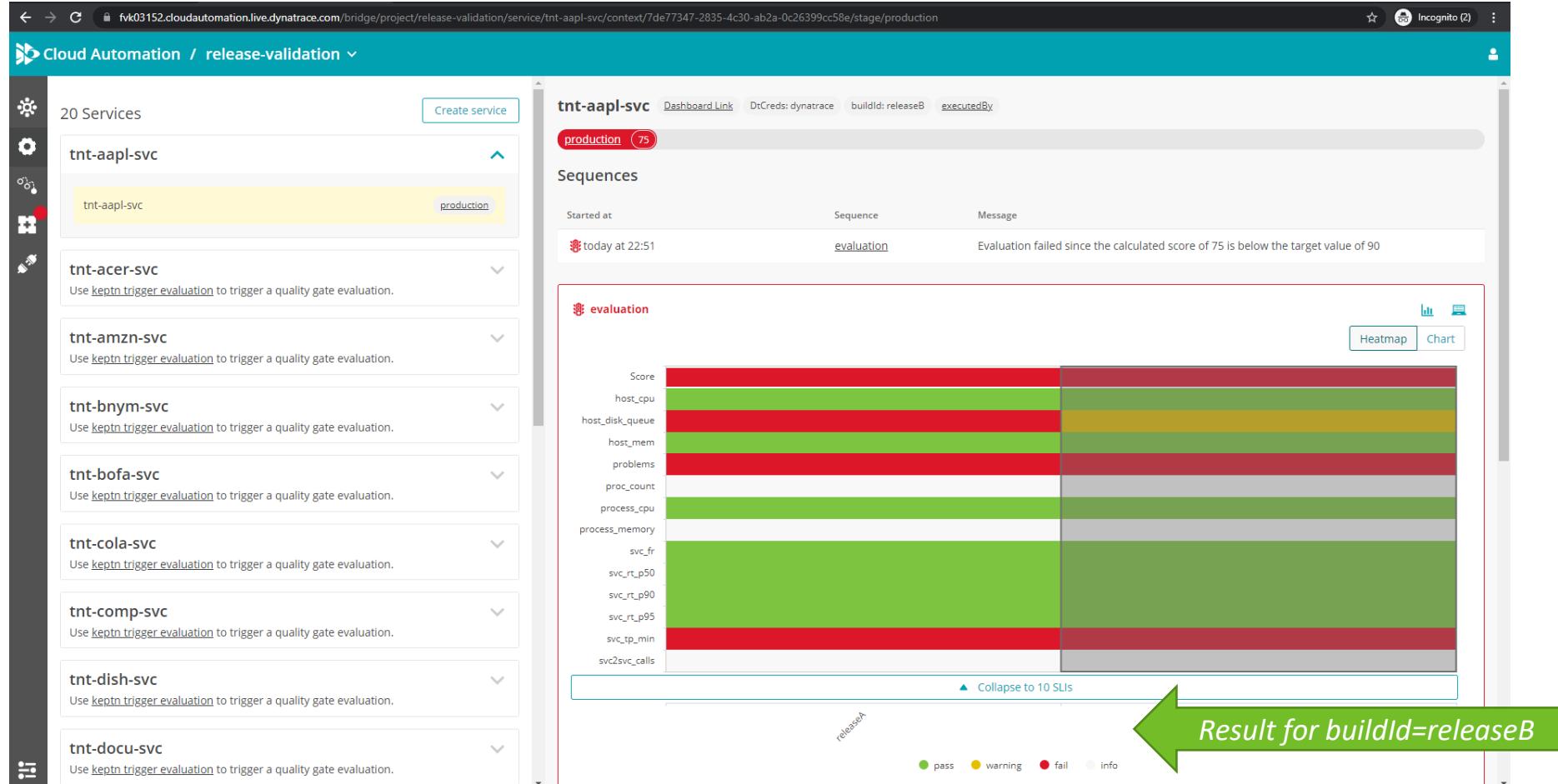
Key: Value:

Trigger Evaluation **Open Result** *After clicking Trigger – this is your link to the results*

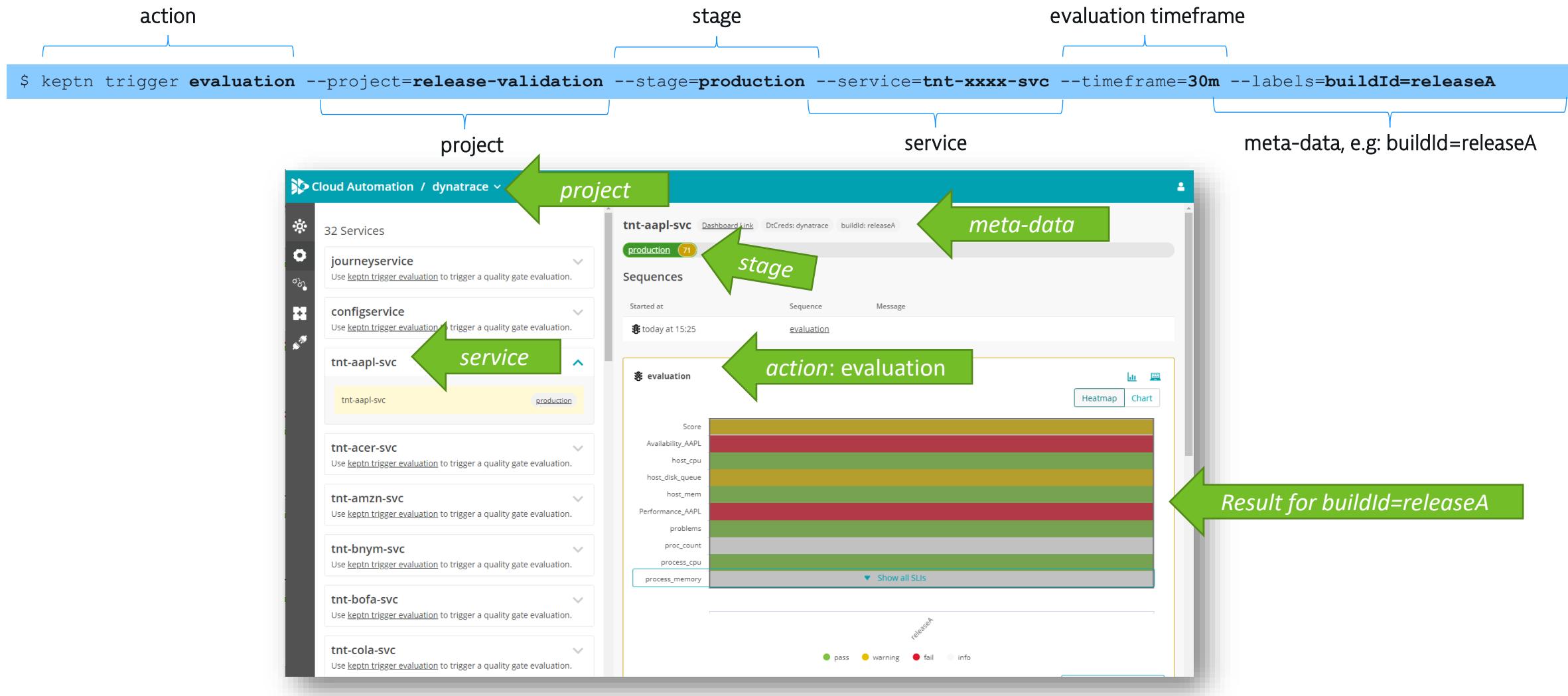
Alternatively: The Keptn CLI command to trigger the evaluation

```
keptn trigger evaluation --project=release-validation --stage=production --timeframe=30m --labels=buildId=1,executedBy=https://yourexternaltool.com/ --service=tnt-xxxx-svc
```

Step 4a: Analyze result for „Release B“



Step 4b: Using the CLI Trigger an evaluation for your service

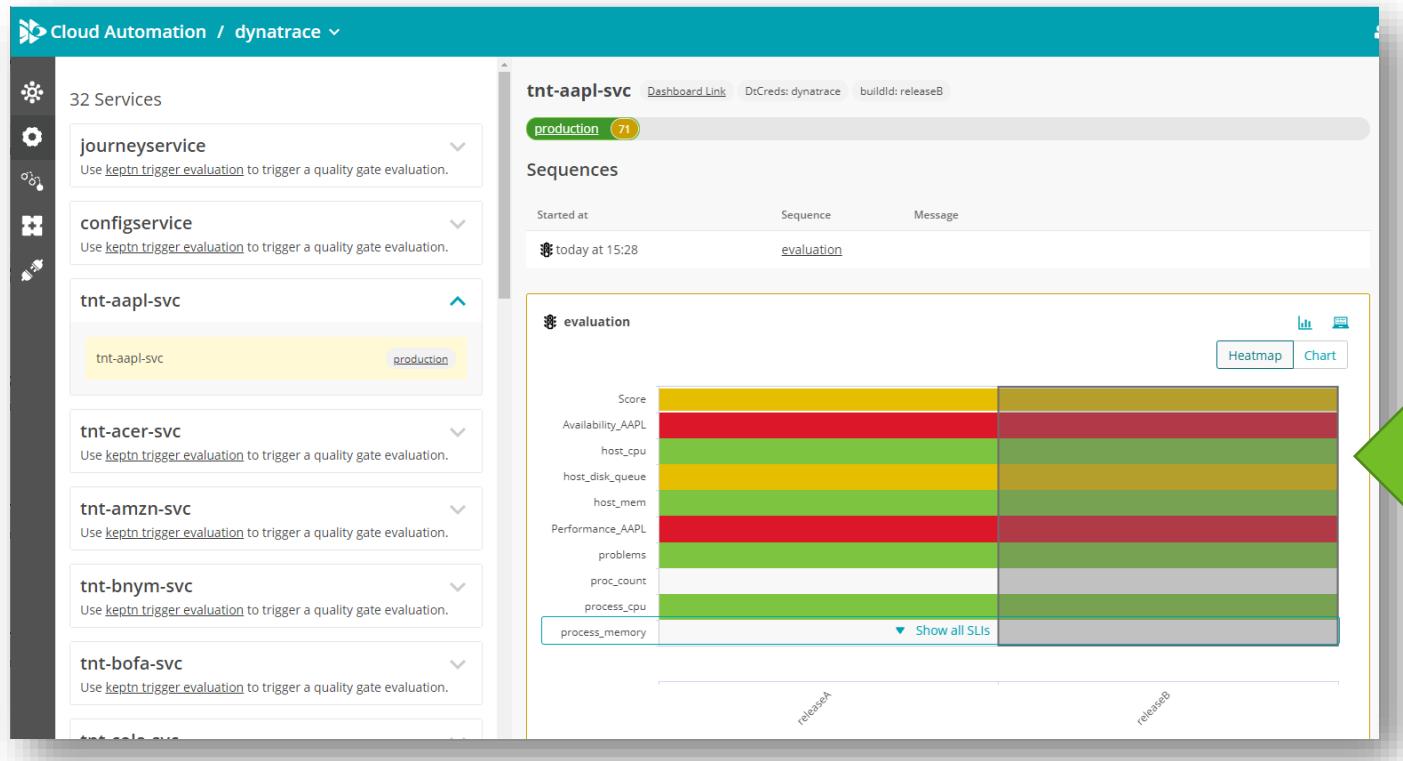


Hint: copy command from slide notes!

Step 4b: Trigger another evaluation through the CLI

```
$ keptn trigger evaluation --project=release-validation --stage=production --service=tnt-xxxx-svc --timeframe=30m --labels=buildId=releaseB
```

meta-data, e.g: buildId=releaseB



Result for buildId=releaseB

Step 4c: Trigger an evaluation through the API

- Open the Swagger UI; select the controlPlane API, Authenticate with the API Token, execute the evaluation endpoint

controlPlane API

action: evaluation

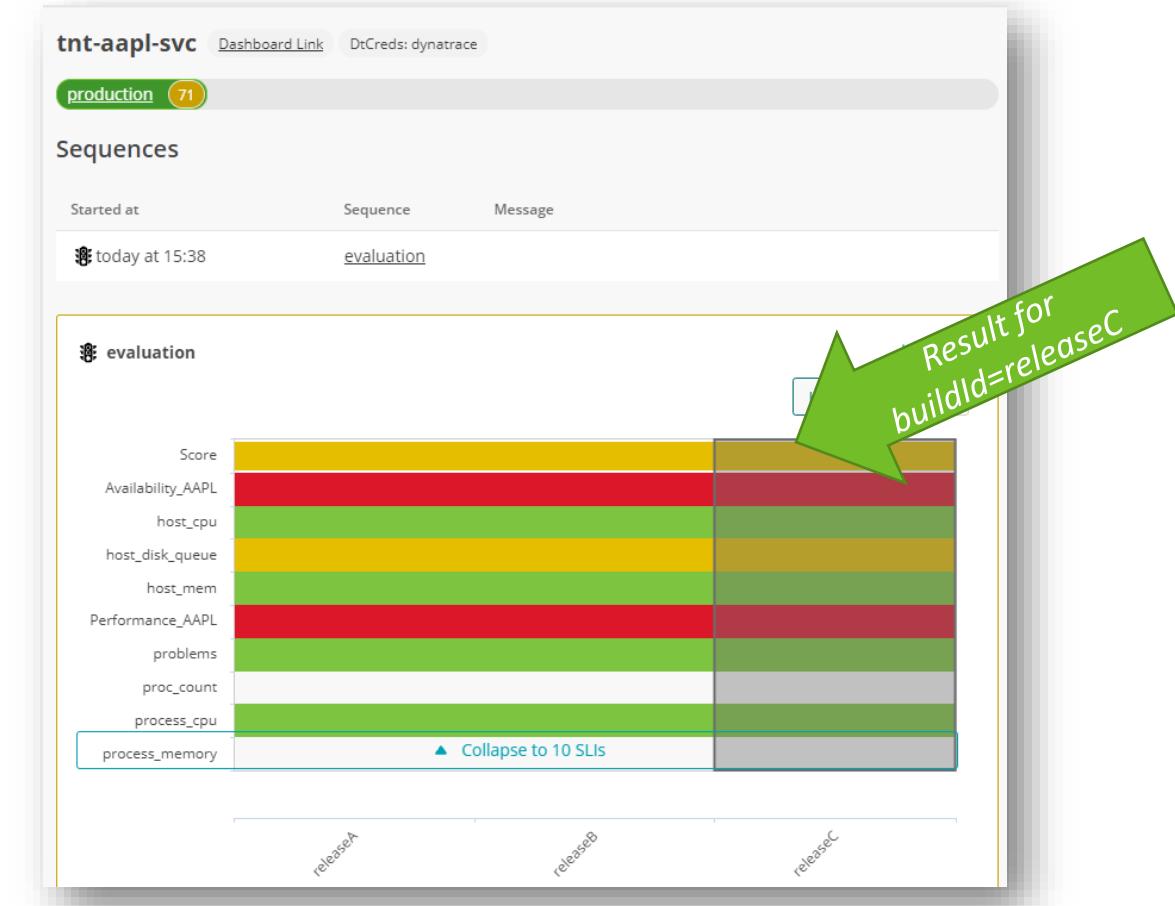
project: release-validation

stage: production

service: tnt-xxxx-SVC

meta-data

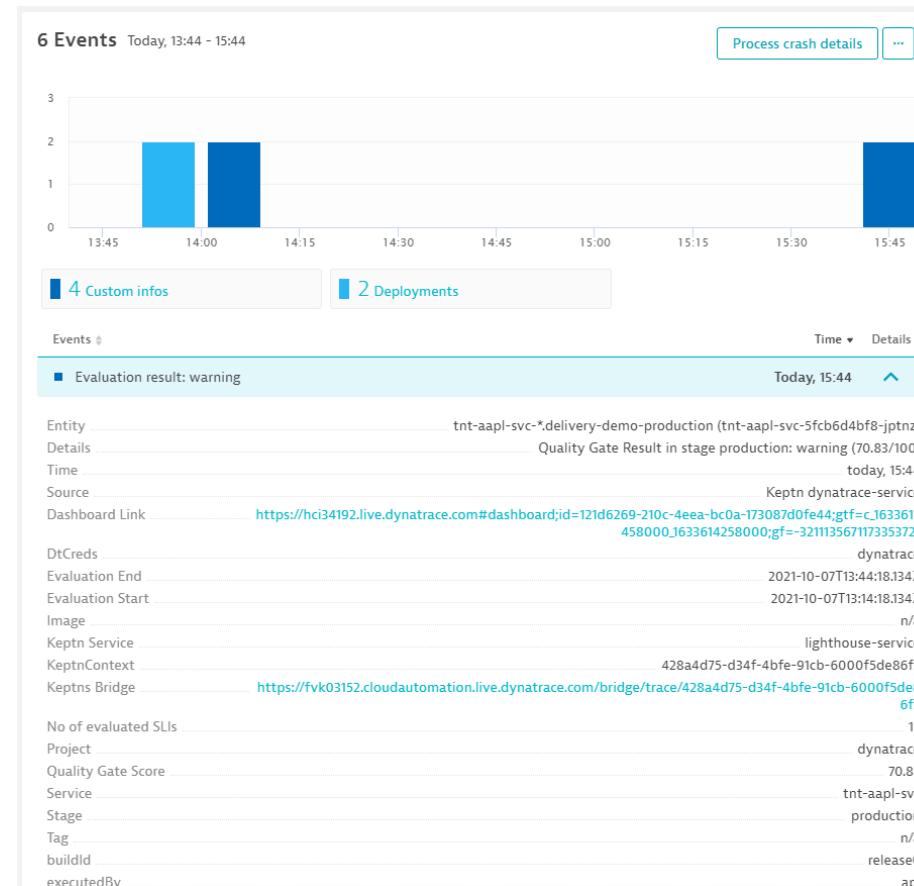
timeframe



Hint: copy HTTP body from slide notes!

Phase #2 – Step 4: Automation Events also available in Dynatrace

- Evaluation Events also sent to Dynatrace monitored entities



Quick Status Check: are we all good with accessing our environments?

- Please mark your tasks accordingly in the Excel file

Release Validation		
Create/Clone your Release Validation Dashboard	Trigger an evaluation for Release A	Trigger evaluation for Release B, C

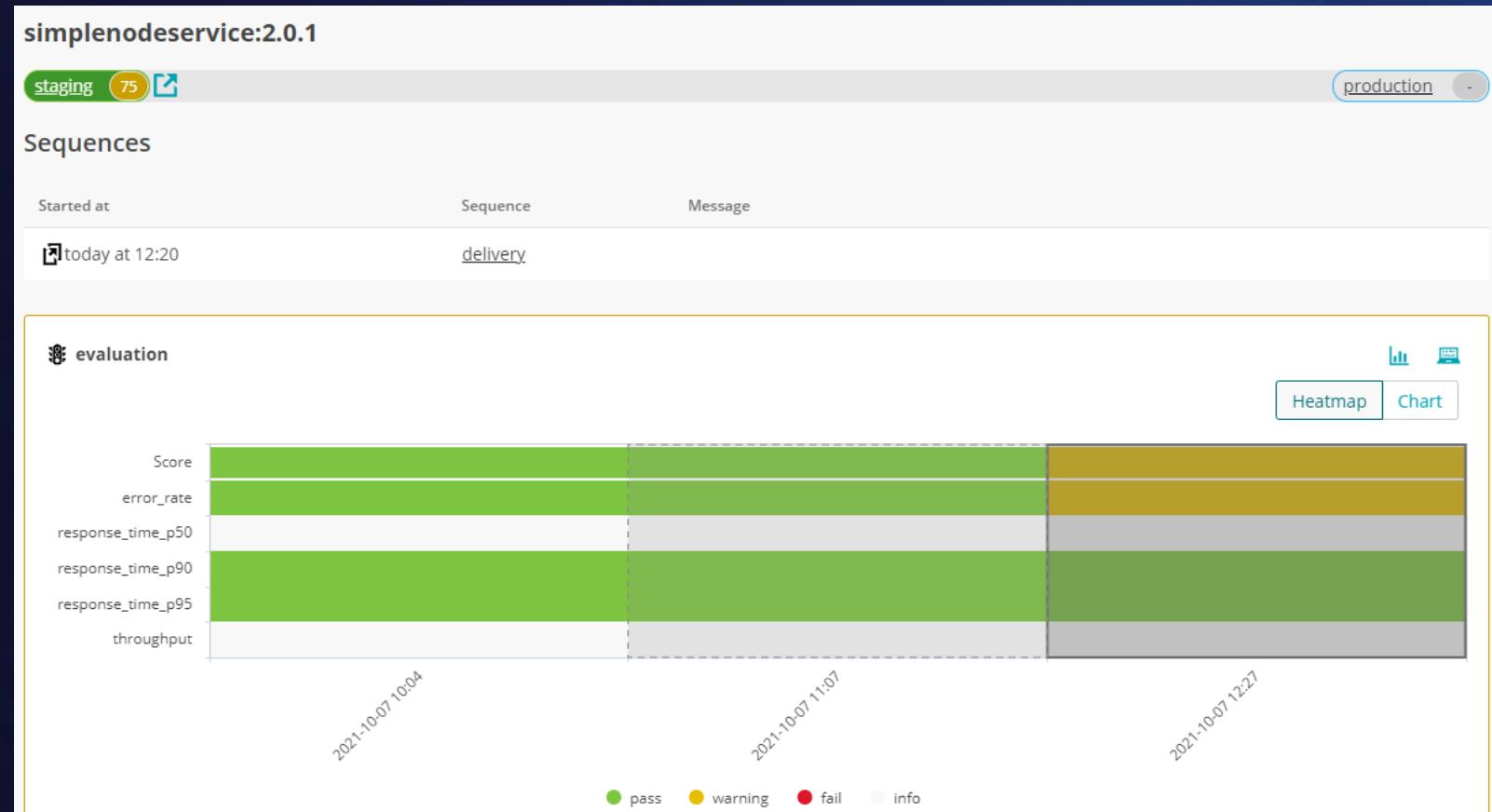
Lab 3: Delivery Pipelines

Automating Quality Gates as part of Delivery Pipelines

Demo: SLO evaluation part of DevOps delivery

```
SLO

spec_version: '0.1.0'
comparison:
  compare_with: "single_result"
  include_result_with_score: "pass"
  aggregate_function: avg
objectives:
  - sli: response_time_p95
    pass:
      - criteria:
          - "<=1000"
    warning:
      - criteria:
          - "<=1500"
  - sli: throughput
  - sli: error_rate
    weight: 2
    pass:
      - criteria:
          - "<=1"
    warning:
      - criteria:
          - "<=2"
  - sli: response_time_p50
  - sli: response_time_p90
    pass:
      - criteria:
          - "<=500"
    warning:
```



After you login to Cloud Automation you see the project „delivery-demo“

Delivery-demo project

The screenshot shows the Cloud Automation interface with a green arrow pointing to the 'delivery-demo' project. The project details are as follows:

- delivery-demo**
- 2 Stages, 30 Services
- Shipyard version: 0.2.0
- A note: No Git upstream configured. [Set Git upstream](#)
- Recent sequences:
- tnt-yumb-svc in production 100 | delivery succeeded today at 14:00
- tnt-wday-svc in production 88 | delivery succeeded today at 13:59
- tnt-vrtx-svc in production 100 | delivery succeeded today at 13:59
- tnt-tsla-svc in production 100 | delivery succeeded today at 13:58
- tnt-siri-svc in production 100 | delivery succeeded today at 13:57

Explore your delivery automation for your tenant

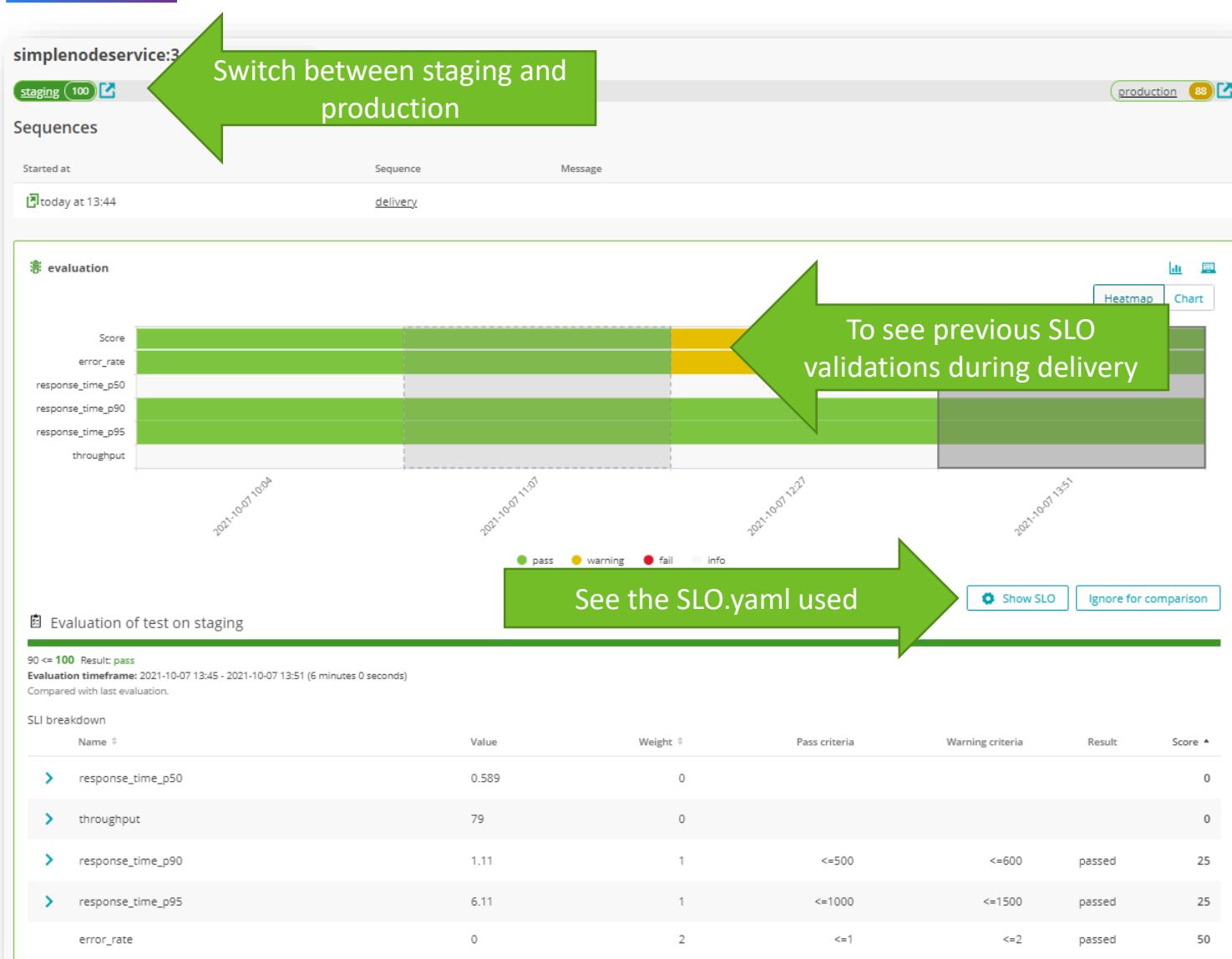
The screenshot shows the Cloud Automation interface for a tenant named "delivery-demo". It displays two stages: "staging" and "production".

- Filter Bar:** At the top, there is a "Filter by service" dropdown set to "Services: tnt-aapl-svc" with a clear button, and a "Clear all" button.
- staging:** This section shows 0 errors, 0 warnings, and 0 shields. It contains a card for "tnt-aapl-svc" which is a "simplenodeservi..." delivery with a status of 100%.
- production:** This section shows 0 errors, 0 warnings, and 0 shields. It contains a card for "tnt-aapl-svc" which is a "simplenodeservi..." delivery.

Annotations on the screenshot:

- A large green arrow points to the "tnt-aapl-svc" card in the staging section, with the text "Clock on the latest delivery automation".
- A green arrow points to the "Filter by service" dropdown, with the text "Filter for your service tenant".

Explore previous delivery automation runs



```

SLO
---
spec_version: '0.1.0'
comparison:
  compare_with: "single_result"
  include_result_with_score: "pass"
  aggregate_function: avg
objectives:
  - sli: response_time_p95
    pass:
      - criteria:
          - "<=1000"
    warning:
      - criteria:
          - "<=1500"
  - sli: throughput
  - sli: error_rate
    weight: 2
    pass:
      - criteria:
          - "<=1"
    warning:
      - criteria:
          - "<=2"
  - sli: response_time_p50
  - sli: response_time_p90
    pass:
      - criteria:
          - "<=500"

```

Copy **Close**

Lets change the SLO.yaml

Step 1: Retrieve current SLO.yaml

← → ⚡ Not secure | keptnwebservice.devopstools-production.claus-ws-production.keptn.sh

Welcome to your Cloud Automation Interactive Lab

This web interface allows you to trigger sequences on your Dynatrace Cloud Automation instance such as "trigger evaluation" or "trigger delivery". It shows you how easy it is to trigger the automation and therefore how easy it is to integrate this with your existing DevOps tools such as Jenkins. This interface is connected to: <https://fvk03152.cloudautomation.live.dynatrace.com>

Trigger Evaluation Trigger Delivery Manage Resources (SLOs)

Manage Git Resources

Project: delivery-demo **Stage:** staging **Service:** tnt-aapl-svc **Resource:** slo.yaml

Get Resource **Add Resource**

project: delivery-demo

service: tnt-xxxx-SVC

Get Reesource

Resource Content

```
---  
spec_version: '0.1.0'  
comparison:  
  compare_with: "single_result"  
  include_result_with_score: "pass"  
  aggregate_function: avg  
objectives:  
  - sli: response_time_p95  
    pass:  
      - criteria:  
        - "<=1000"  
    warning:  
      - criteria:  
        - "<=1500"  
  - sli: throughput  
    pass:  
      - criteria:  
        - "<=1000"  
    warning:  
      - criteria:  
        - "<=1500"
```

This is the current SLO.yaml

Here is the Keptn CLI command to add resources

```
keptn add-resource --project=delivery-demo --stage=staging --service=tnt-xxxx-svc --resource=slo.yaml
```

Step 2: Update SLO.yaml

← → ⚡ Not secure | keptnwebservice.devopstools-production.claus-ws-production.keptn.sh

Welcome to your Cloud Automation Interactive Lab

This web interface allows you to trigger sequences on your Dynatrace Cloud Automation instance such as "trigger evaluation" or "trigger delivery". It shows you how easy it is to trigger the automation and therefore how easy it is to integrate this with your existing DevOps tools such as Jenkins. This interface is connected to: <https://fvk03152.cloudautomation.live.dynatrace.com>

Trigger Evaluation Trigger Delivery Manage Resources (SLOs)

Manage Git Resources

Project: delivery-demo **Stage:** staging **Service:** tnt-aapl-svc **Resource:** slo.yaml

Get Resource **Add Resource**

Then click on Add Resource

Resource Content

```
---  
spec_version: '0.1.0'  
comparison:  
  compare_with: "single_result"  
  include_result_with_score: "pass"  
  aggregate_function: avg  
objectives:  
  - sli: response_time_p95  
    pass:  
      - criteria:  
        - "<=800"  
    warning:  
      - criteria:  
        - "<=1500"  
  - sli: throughput  
    pass:  
      - criteria:  
        - "<=800"  
    warning:  
      - criteria:  
        - "<=1500"
```

e.g: change criteria to 800

Here is the Keptn CLI command to add resources

```
keptn add-resource --project=delivery-demo --stage=staging --service=tnt-xxxx-svc --resource=slo.yaml
```

Lets trigger a new deployment

Not secure | keptnwebservice.devopstools-production.claus-ws-production.keptn.sh

Welcome to your Cloud Automation Interactive Lab

This web interface allows you to trigger sequences on your Dynatrace Cloud Automation instance such as "trigger evaluation" or "trigger delivery". It shows you how easy it is to trigger the automation and therefore how easy it is to integrate this with your existing DevOps tools such as Jenkins, CircleCI, Travis CI, etc.

This interface is connected to: <https://fvk03152.cloudautomation.live.dynatrace.com>

Trigger Evaluation **Trigger Delivery** **Manage Resources (SLOs)**

Trigger a delivery sequence

Project: delivery-demo *project: demo-delivery*

Stage: staging

Service: tnt-aapl-svc *service: tnt-xxxx-svc*

Image to deploy: grabnerandi/simplenodeservice:3.0.1 *version: e.g: 3.0.1*

Sequence name (default=delivery): delivery

Additionally we can specify labels as key/value pairs

Additional labels: Key: Value:

Trigger Delivery **Open Result** *After clicking Trigger – this is your link to the delivery sequence*

Here is the Keptn CLI command to trigger the evaluation

```
keptn trigger delivery --project=delivery-demo --stage=staging --image=grabnerandi/simplenodeservice:1.0.0 --labels=
```

delivery waiting

Context: fa96de7a-1a54-4a20-83da-91d01cc6f71c

simplenodeservice:3.0.1

A Sequence might be waiting for previous sequences to be finished.

staging

13:06

13:06 **delivery**

Labels:

13:06 **monaco**

Labels:

13:06 **deployment**

Labels: DtCreds:

Last time fetched: today at 13:08:26

Confidential

Just as we did for release validation automation – we need a dashboard

- The automation will look for a Dynatrace dashboard with the following naming schema:
 - KQG;project=<PROJECT>;stage=<STAGE>;service=<SERVICE>
- In our case project=delivery-demo, stage=staging and service=tnt-xxxx-svc
- We could start from scratch, or – start from a template 😊

Step 1 – Clone the existing dashboard template

Dashboards

Dashboards

Overview of all dashboards you are permitted to view or edit.

Please provide feedback and find planned enhancements at [Dynatrace answers](#).

Show all tenant dashboards (for admin users only)

Ownership

Any
 Mine
 Shared with me

Favorite

Any
 Yes
 No

2 Dashboards

Favorite	Name	Modified at	Owner
★	KQG;project=delivery-demo;stage=staging;service=tnt-xxxx-svc	Oct 07 15:52	andreas.grabner@dynatrac...
★	KQG;project=delivery-demo;stage=staging;service=tnt-xxxx-svc	Oct 07 15:55	andreas.grabner@dynatrac...

CLONE  

KQG;project=delivery-demo;stage=staging;service=tnt-xxxx-svc

Step 2 – Give it proper name, change MZs and add your SLOs

The screenshot shows the configuration interface for a dashboard titled "SLO-based Quality Gate Dashboard based on important quality metrics". The top navigation bar includes a search bar, tenant selection ("Tenant: tnt-aapl-svc"), and time range ("Last 2 hours"). The URL in the address bar is "KQG;project=delivery-demo;stage=staging;service=tnt-aapl-svc". A green arrow points from the text "Replace xxxx with your service" to the URL.

The dashboard itself has sections for "Release Validation Criteria", "Service Performance (SLI/SLO)", and "Process Metrics (SLI/SLO)". It displays various metrics with their current values and thresholds:

- Release Validation Criteria:** KQG.Total.Pass=90%;KQG.Total.Warning=70%;KQG.Compare.WithScore=pass;KQG.Compare.Result
- Service Performance (SLI/SLO):**
 - Response time (P95);sli=svc_rt_p95;pass=<+10%,<600
Value: 1.82ms
 - Failure Rate (Avg);sli=svc_fr;pass=<+10%,<2
Value: 0%
 - Throughput (per min);sli=svc_tp_min;pass=<+10%,<200
Value: 26.3/min
 - Response time (P90);sli=svc_rt_p90;pass=<+10%,<550
Value: 629µs
- Process Metrics (SLI/SLO):**
 - Process CPU;sli=process_cpu;pass=<20%
Value: 87.5MB

On the right side of the dashboard, there are configuration options:

- Default filters:** Includes "Default timeframe" (radio button) and "Default management zone" (radio button). The "Management Zone" dropdown is set to "Tenant: tnt-aapl-svc". A green arrow points from the text "Select your MZ" to this dropdown.
- Reports:** Includes a "Enable reports" checkbox.

Below the dashboard, a note says: "Below are a few quick configuration options for your dashboard." It links to "Advanced settings".

Step 3 – Save the dashboard!

Search Dynatrace - hci34192: hci34192...

Tenant: tnt-aapl-svc Last 2 hours

KQG;project=delivery-demo;stage=staging;service=tnt-aapl-svc

2 tags applied Edit ...

tnt-aapl-svc validation

SLO-based Quality Gate Dashboard based on important quality metrics

This dashboard will be analyzed as part of your DevOps pipelines when your service gets quality analyzed in staging

TODO: Clone dashboard, select your Management Zone, Add your relevant metrics (SLIs/SLOs), replace XXXX with your tenant name

Release Validation Criteria: KQG.Total.Pass=90%;KQG.Total.Warning=70%;KQG.Compare.WithScore=pass;KQG.Compare.Results=1;KQG.Compare.Function=avg

Service Performance (SLI/SLO)

Failure Rate (Avg);sli=svc_fr;pass=<+10%,<2

Process Metrics (SLI/SLO)

Process CPU;sli=process_cpu;pass=<20;warning=<50;k...

Response time (P95);sli=svc_rt_p95;pass=<+10%,<600

0%

0.01%

1.82ms

Response time (95th percentile)

Failure rate (server side errors) (Average)

Process CPU usage (Average)

14:00 14:30 15:00 15:30

Response time (P90);sli=svc_rt_p90;pass=<+10%,<550

26.3/min

Process Memory;sli=process_memory

629 μ s

Request count (Sum)

87.5MB

Process memory (Average)

14:00 14:30 15:00 15:30

14:00 14:30 15:00 15:30

14:00 14:30 15:00 15:30

The dashboard displays six key metrics:

- Response time (P95):** 1.82ms (95th percentile)
- Failure Rate (Avg):** 0% (server side errors, Average)
- Process CPU usage (Average):** 0.01%
- Response time (P90):** 629 μ s (90th percentile)
- Request count (Sum):** 26.3/min
- Process memory (Average):** 87.5MB

Each metric includes a corresponding chart showing historical data from 14:00 to 15:30.

Triggering end-2-end delivery of version 2.0.1 of our sample app

action

```
$ keptn trigger delivery --project=delivery-demo --service=tnt-xxxx-svc --image=grabnerandi/simplenodeservice:2.0.1
```

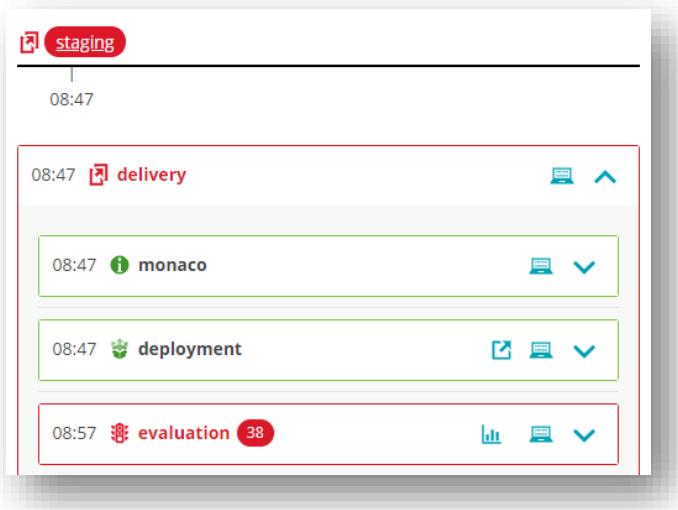
service

project

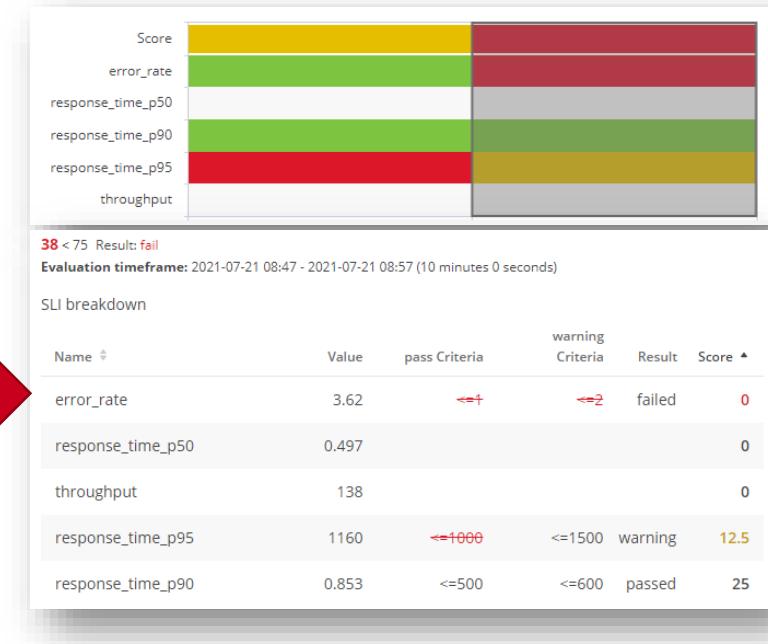
image

Other optional parameters:
 --service defaults to staging
 --labels is also possible
 --values additional artifact metadata
 --sequence if sequence other than delivery

Build #2 should fail due to high error rate and wont be promoted to production!



Failure Rate not meeting SLO of <2



Triggering end-to-end delivery of version 3.0.1 of our sample app

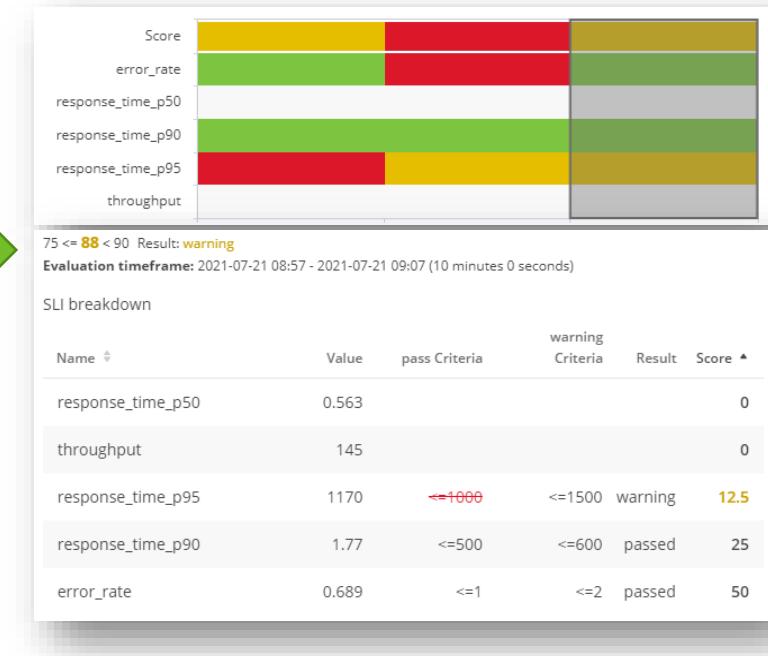
```
$ keptn trigger delivery --project=delivery-demo --service=tnt-xxxx-svc --image=grabnerandi/simplenodeservice:3.0.1
```

image

Build #3 should meet Quality Gate and gets promoted to production

The screenshot shows the Keptn delivery pipeline interface. It displays the stages from left to right: staging, delivery, deployment, evaluation (with 88 issues), get-sli, and approval. The evaluation stage is highlighted with a green border. The time axis at the top shows 08:56 and 09:07.

No major issues any more



Validate Deployment Events in Release Inventory

Releases

Release monitoring

Overview of deployed component versions and release events. For details, see [Release monitoring](#) or activate demo mode to view sample data.

Filtered by Tag: [Environment]WorkshopTenant:angr Monitor state: Active Clear all

Release inventory

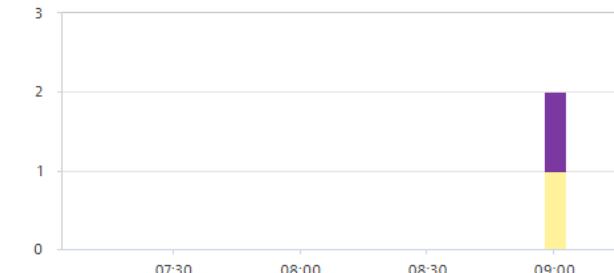
2 Releases

Name	Version	Stage	Product	Instances	Throughput
tnt-angr-svc-*delivery-d...	3.0.0	production	delivery-demo	1	-
tnt-angr-svc-*delivery-d...	3.0.0	staging	delivery-demo	1	-

< 1 >

Release events

2 events match your query and filtering



Events Time Details

Deploy tnt-angr-svc 3.0.0 with strate... today, 09:09

Evaluation result: warning today, 09:07

Filter on active for your tenant xxxx

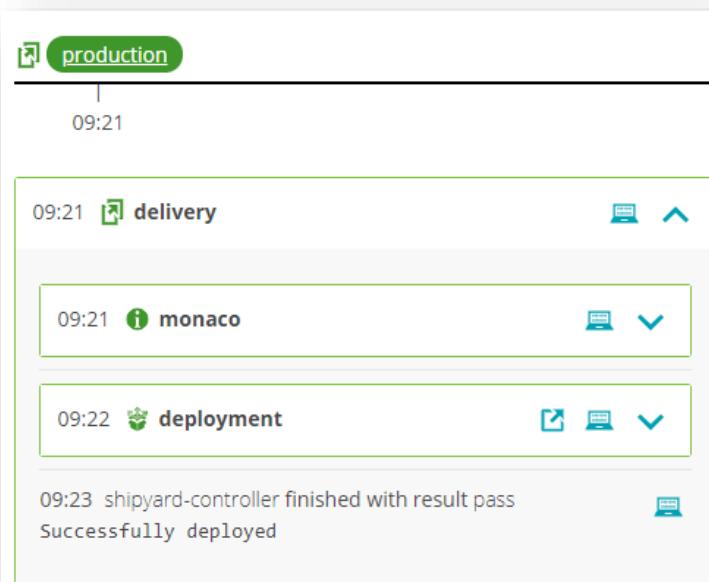
Triggering direct deployment of 4.0.1 into production

```
$ keptn trigger delivery --project=delivery-demo --stage=production --service=tnt-xxxx-svc --image=grabnerandi/simplenodeservice:4.0.1
```

stage

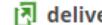
image

Build #4 deploys straight into production



production

09:21

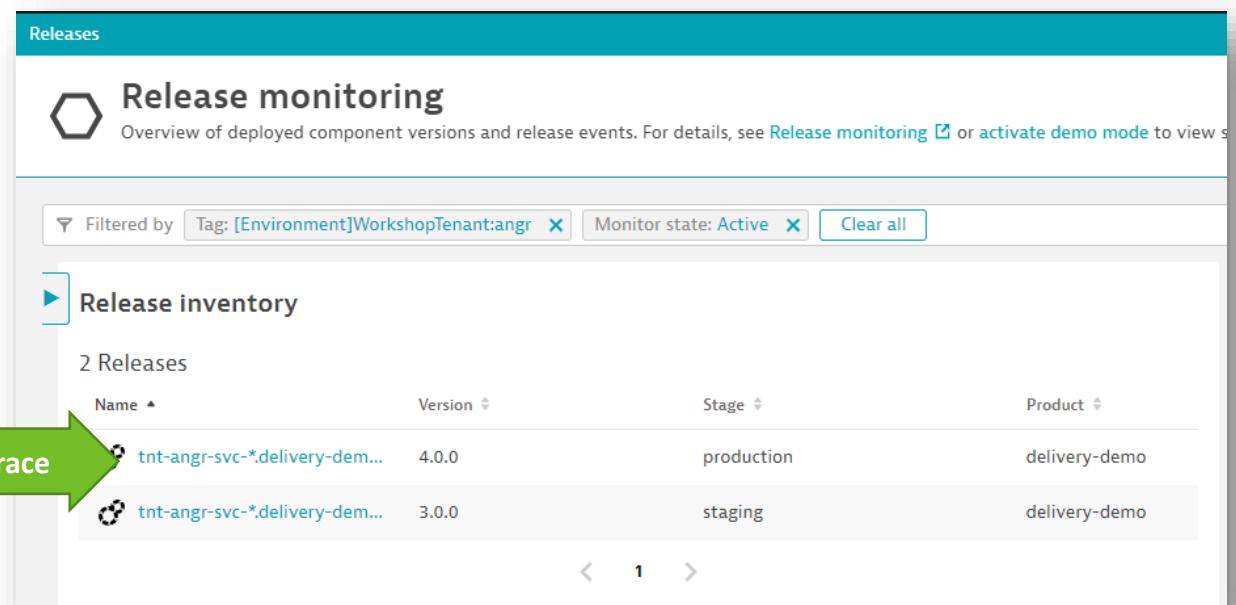
09:21  delivery

09:21  monaco

09:22  deployment

09:23 shipyard-controller finished with result pass
Successfully deployed

Also reflected in Dynatrace



Releases

Release monitoring

Overview of deployed component versions and release events. For details, see [Release monitoring](#) or [activate demo mode](#) to view s

Filtered by Tag: [Environment]WorkshopTenant:angr Monitor state: Active Clear all

Release inventory

2 Releases

Name	Version	Stage	Product
tnt-angr-svc-*.delivery-dem...	4.0.0	production	delivery-demo
tnt-angr-svc-*.delivery-dem...	3.0.0	staging	delivery-demo

Quick Status Check: are we all good with accessing our environments?

- Please mark your tasks accordingly in the Excel file

Delivery Pipelines		
Create / Clone your SLO-based Quality Gate Dashboard for staging	Trigger new deployments and validate dashboard is used	Validate results and events in Dynatrace release monitoring



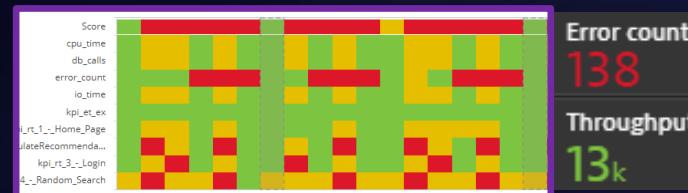
Hands-On Lab: Wrap Up!

How Dynatrace helps DevOps & SRE

Dynatrace helps DevOps & SREs to Shift-Left SLOs



Delivery Pipelines



Speed up high-quality value creation

Release Validation



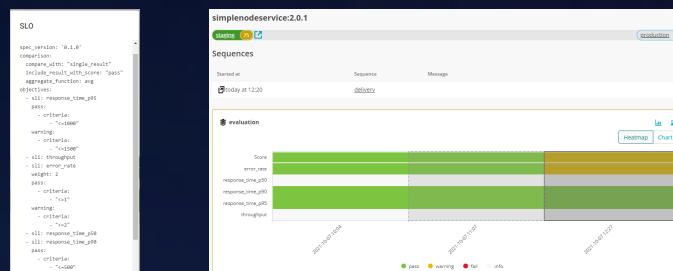
Eliminate Failed Releases

Production Reliability

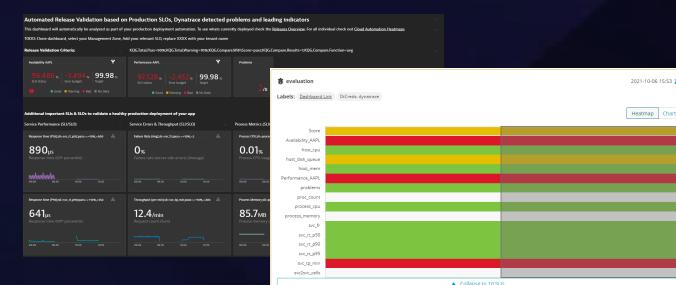


Ensure 100% Business Up-Time

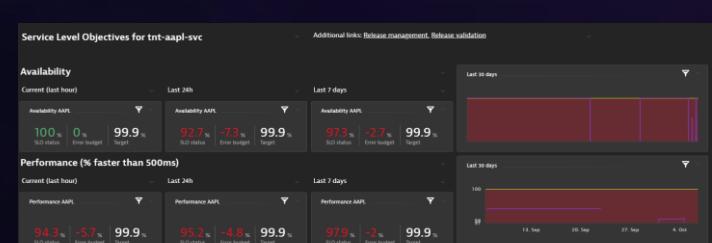
Lab 3: SLO-based Quality Gates



Lab 2: SLOs for Release Validation



Lab 1: SLOs for Reporting



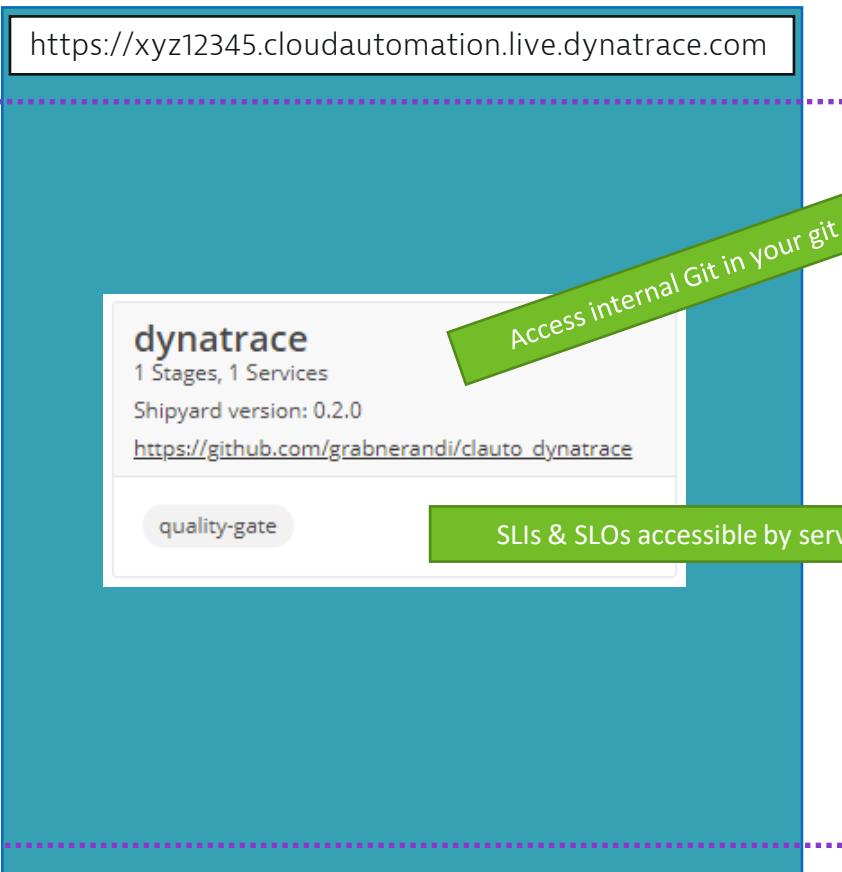
GitOps

Quality Gates as Code & Quality Gates as Dashboard

Here is what we are trying to achieve

Your Cloud Automation SaaS Tenant

https://xyz12345.cloudautomation.live.dynatrace.com



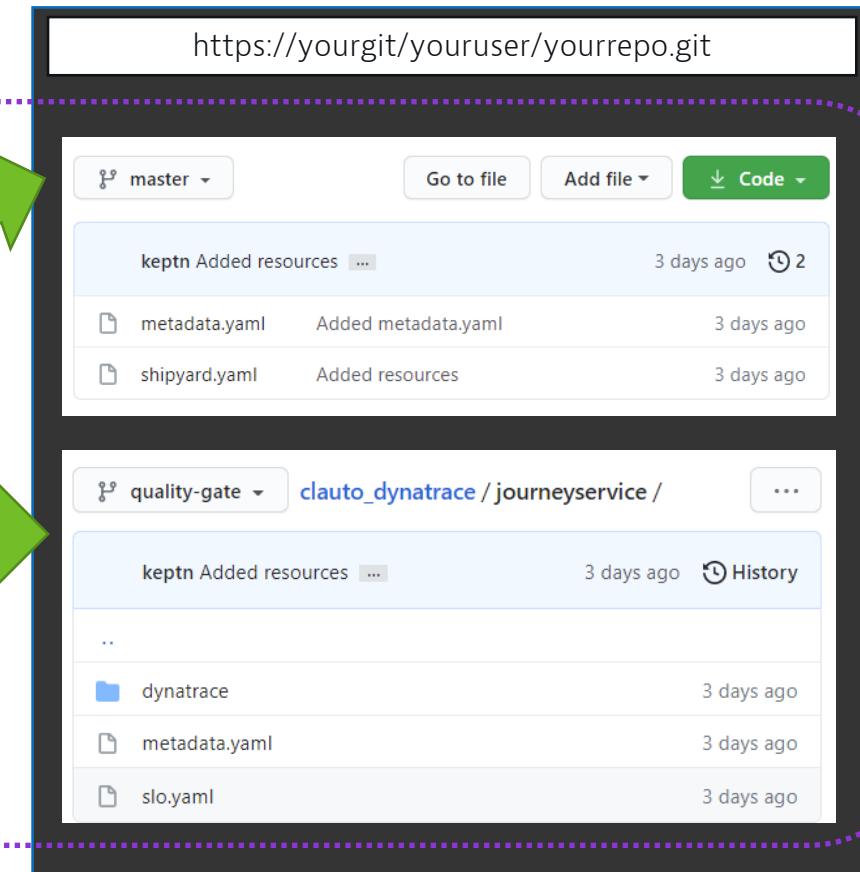
dynatrace
1 Stages, 1 Services
Shipyard version: 0.2.0
https://github.com/grabnerandi/clauto_dynatrace

quality-gate

Access internal Git in your git repo

Your „internet accessible“ Git Repository

https://yourgit/youruser/yourrepo.git



Commit	File	Message	Time
keptn Added resources		3 days ago	⌚ 2
metadata.yaml	Added metadata.yaml	3 days ago	
shipyard.yaml	Added resources	3 days ago	
dynatrace		3 days ago	

keptn Added resources ... 3 days ago ⌚ 2

metadata.yaml Added metadata.yaml 3 days ago

shipyard.yaml Added resources 3 days ago

dynatrace 3 days ago

quality-gate clauto_dynatrace / journeyservice /

keptn Added resources ... 3 days ago ⌚ History

..

dynatrace 3 days ago

metadata.yaml 3 days ago

slo.yaml 3 days ago

Phase 3
GitOps

Phase #3 - Step 1: Create an empty git repository, e.g: GitHub, Bitbucket, Azure, GitLab ...

- Example: create an empty GitHub repository
 - More documentation for [other git systems](#)
- Besides a Git repo you also need
 - Git username
 - Git token with privileges [described here](#)
- Pre-Req
 - Repository must be accessible *from the internet*
 - *Can be private*

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner * Repository name *

 grabnerandi / clauto_dynatrace ✓

Great repository names are short and memorable. Need inspiration? How about [expert-carnival!](#)

Description (optional)

Git upstream for my Cloud Automation dynatrace project

Public Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file This is where you can write a long description for your project. [Learn more.](#)

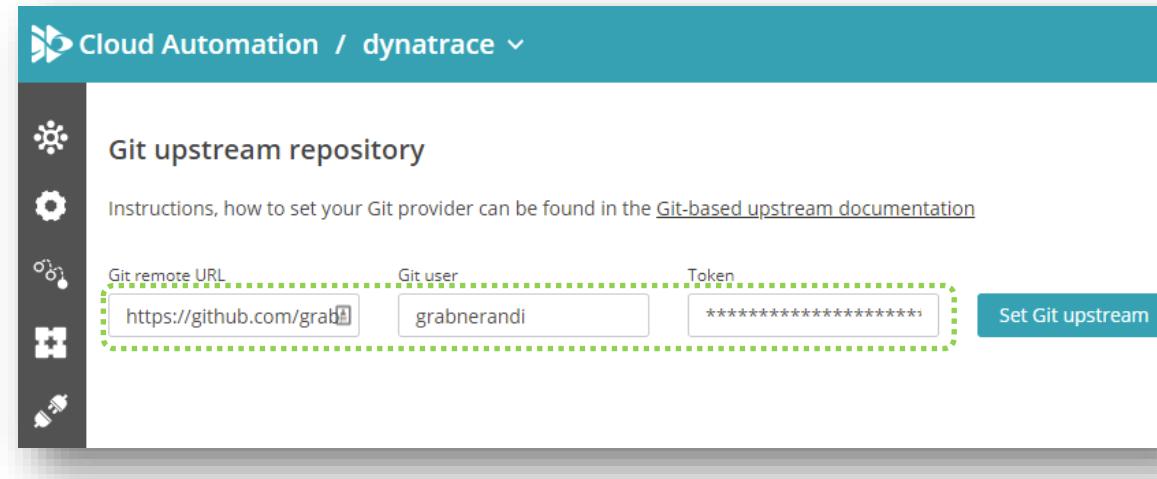
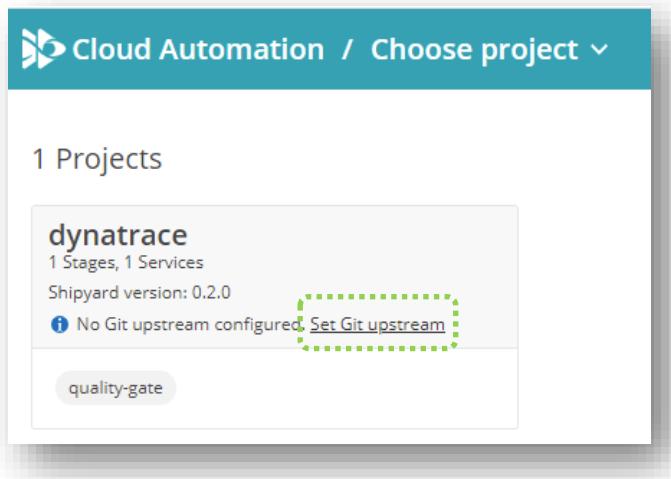
Add .gitignore Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

Phase #3 - Step 2: Set the git upstream credentials on your project

- Option A: Via the Cloud Automation UI



- Option B: Via the Keptn CLI

```
$ keptn update project dynatrace --git-user=YOUR_GIT_USER --git-token=YOUR_GIT_TOKEN --git-remote-url=YOUR_GIT_REMOTE_URL
Starting to update project
Project updated successfully
```

- Option C: Via the Keptn API

PUT **/project** Updates a project

Phase #3 - Step 3: Explore dynatrace project upstream Git: shipyard, SLI, SLO ...

1 Cloud Automation / Choose project ▾

1 Projects

dynatrace
1 Stages, 1 Services
Shipyard version: 0.2.0
https://github.com/grabnerandi/clauto_dynatrace

quality-gate

Link to upstream Git

2 master ▾

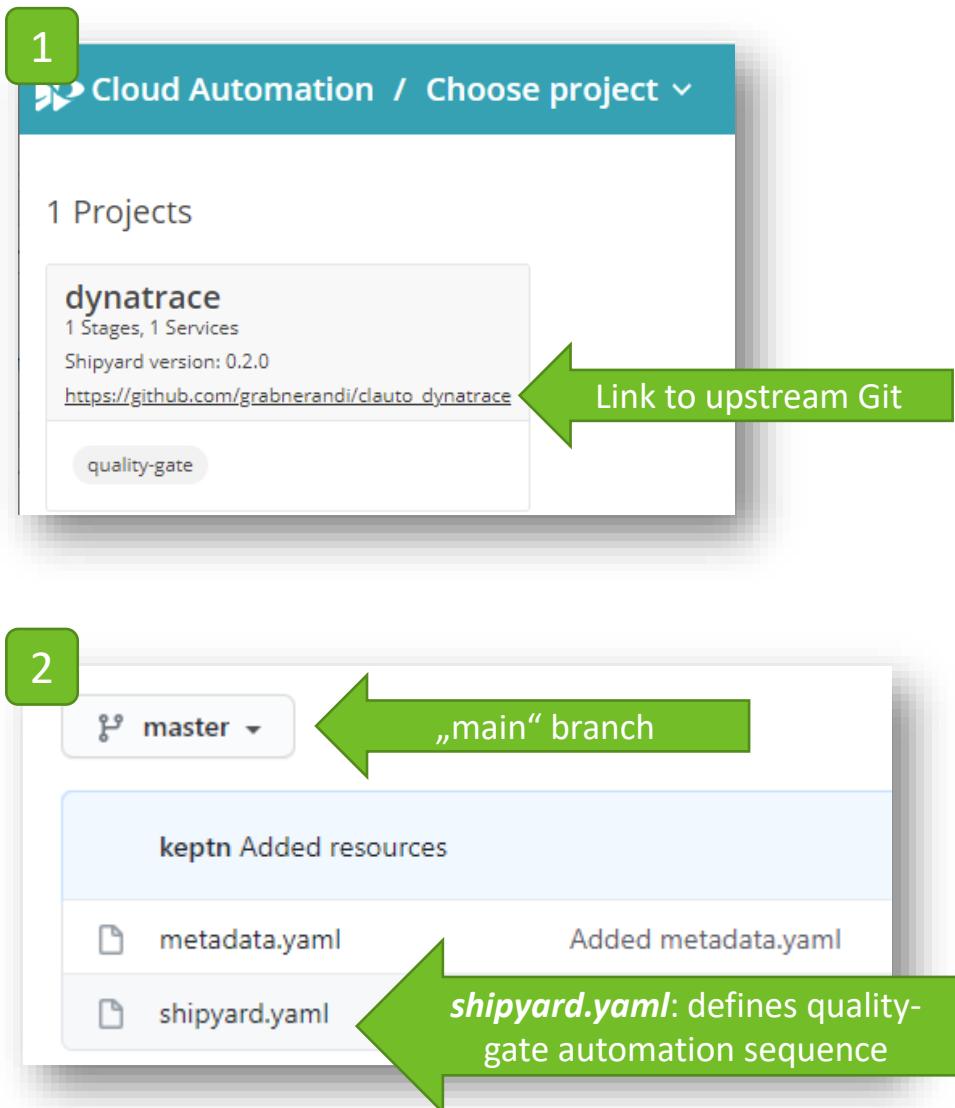
keptn Added resources

metadata.yaml Added metadata.yaml

shipyard.yaml

„main“ branch

shipyard.yaml: defines quality-gate automation sequence



3 quality-gate ▾

„quality-gate“ branch

Branch-wide dynatrace settings

Folder per automation-enabled service

dynatrace

journeyservice

metadata.yaml

4 quality-gate ▾ clauto_dynatrace / journeyservice /

Service specific folder

Added resources

metadata.yaml Added service: journeyservice

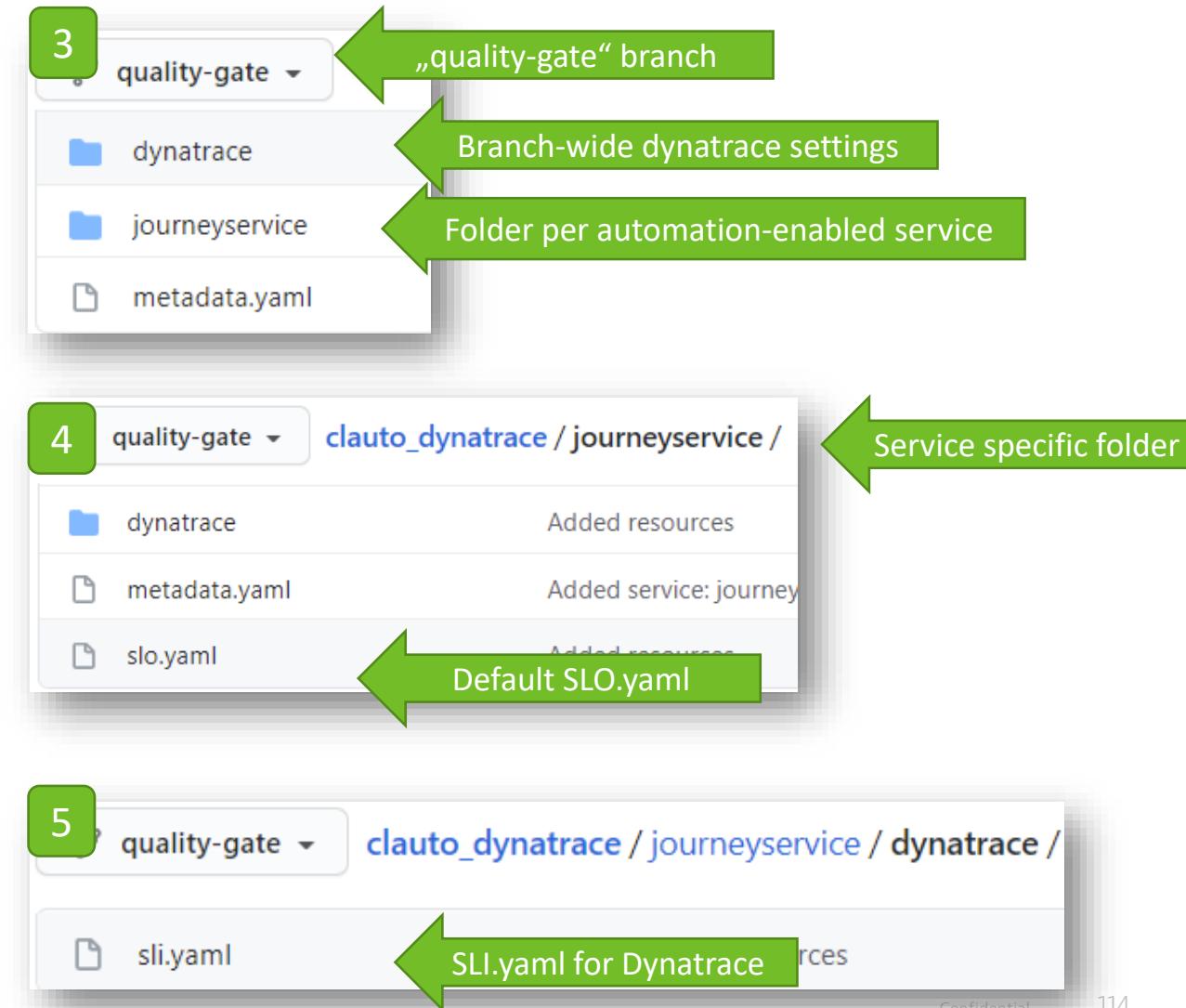
slo.yaml

Default SLO.yaml

5 quality-gate ▾ clauto_dynatrace / journeyservice / dynatrace /

SLI.yaml for Dynatrace

sli.yaml



Phase #3 - Step #4: Customize your SLOs: add, remove or edit your SLOs

Update SLO.yaml

```

Branch: quality-gate
dynatrace / tnt-angr-svc / slo.yaml

27 lines | 520 B

1 ---  

2 spec_version: "1.0"  

3 comparison:  

4   aggregate_function: "avg"  

5   compare_with: "single_result"  

6   include_result_with_score: "pass"  

7   number_of_comparison_results: 1  

8 filter:  

9 objectives:  

10  - sli: "response_time_p95"  

11    key_sli: false  

12    pass:  

13      - criteria:  

14        - "<600"  

15    warning:  

16      - criteria:  

17        - "<=800"  

18    weight: 1  

19  - sli: "error_rate"  

20    key_sli: false  

21    pass:  

22      - criteria:  

23        - "<5"  

24  - sli: throughput  

25 total_score:  

26  pass: "90%"  

27  warning: "75%"
```

Edit File

Preview Changes

```

1 ---  

2 spec_version: "1.0"  

3 comparison:  

4   aggregate_function: "avg"  

5   compare_with: "single_result"  

6   include_result_with_score: "pass"  

7   number_of_comparison_results: 1  

8 filter:  

9 objectives:  

10  - sli: "response_time_p95"  

11    key_sli: false  

12    pass:  

13      - criteria:  

14        - "<500"  

15    warning:  

16      - criteria:  

17        - "<=1000"  

18    weight: 2  

19  - sli: "error_rate"  

20    key_sli: true  

21    pass:  

22      - criteria:  

23        - "<5"  

24  - sli: throughput  

25    pass:  

26      - criteria:  

27        - ">10"  

28  - sli: "response_time_p50"  

29  - sli: "response_time_p90"  

30 total_score:  

31  pass: "90%"  

32  warning: "50%"
```

e.g: change thresholds for rt_p95

e.g: double the weight of this sli

e.g: Make „error_rate“ a „key_sli“

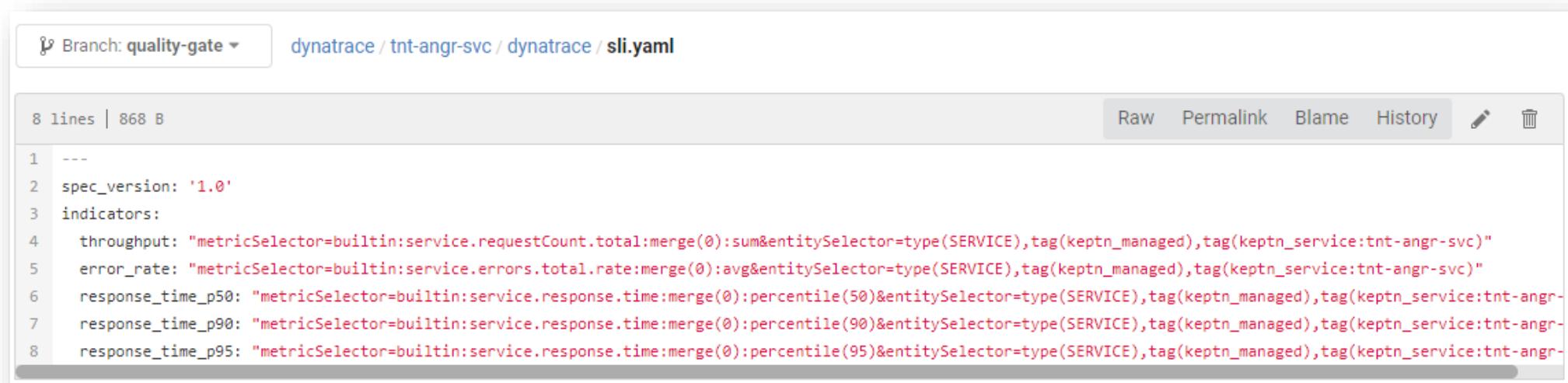
e.g: Add criteria to throughput

e.g: Add additional SLOs

e.g: Change total score criteria

Phase #3 - Step #5: Customize the SLI.yaml: update or add new SLI definitions

- Add additional SLI definitions by providing an sli name and the Dynatrace Metrics API v2 Query
 - <https://www.dynatrace.com/support/help/dynatrace-api/environment-api/metric-v2/get-data-points/>
- You can also use the following PLACEHOLDERS as part of your query
 - \$PROJECT: Cloud Automation projectname, e.g: dynatrace
 - \$STAGE: Cloud Automation stage, e.g: quality-gate
 - \$SERVICE: Cloud Automation service, e.g: journeyservice



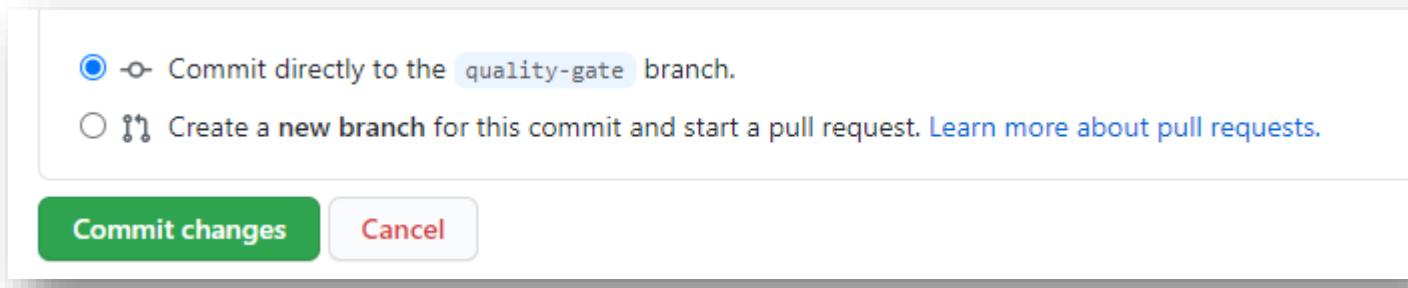
The screenshot shows a code editor interface with the following details:

- Header: Branch: quality-gate ▾, Repository: dynatrace / tnt-angr-svc, File: sli.yaml
- Toolbar: Raw, Permalink, Blame, History, Edit, Delete
- Text Area:

```
1 ---  
2 spec_version: '1.0'  
3 indicators:  
4   throughput: "metricSelector= builtin:service.requestCount.total:merge(0):sum&entitySelector= type(SERVICE),tag(keptn_managed),tag(keptn_service:tnt-angr-svc)"  
5   error_rate: "metricSelector= builtin:service.errors.total.rate:merge(0):avg&entitySelector= type(SERVICE),tag(keptn_managed),tag(keptn_service:tnt-angr-svc)"  
6   response_time_p50: "metricSelector= builtin:service.response.time:merge(0):percentile(50)&entitySelector= type(SERVICE),tag(keptn_managed),tag(keptn_service:tnt-angr-svc)"  
7   response_time_p90: "metricSelector= builtin:service.response.time:merge(0):percentile(90)&entitySelector= type(SERVICE),tag(keptn_managed),tag(keptn_service:tnt-angr-svc)"  
8   response_time_p95: "metricSelector= builtin:service.response.time:merge(0):percentile(95)&entitySelector= type(SERVICE),tag(keptn_managed),tag(keptn_service:tnt-angr-svc)"
```

Phase #3 - Step #6: Update SLI & SLO yaml

- You can either edit sli.yaml and slo.yaml in the git repository



- Or upload / overwrite a new version of sli.yaml & slo.yaml via *keptn add-resource*
 - Use slo.yaml for resourceUri
 - Use dynatrace/sli.yaml for resourceURI

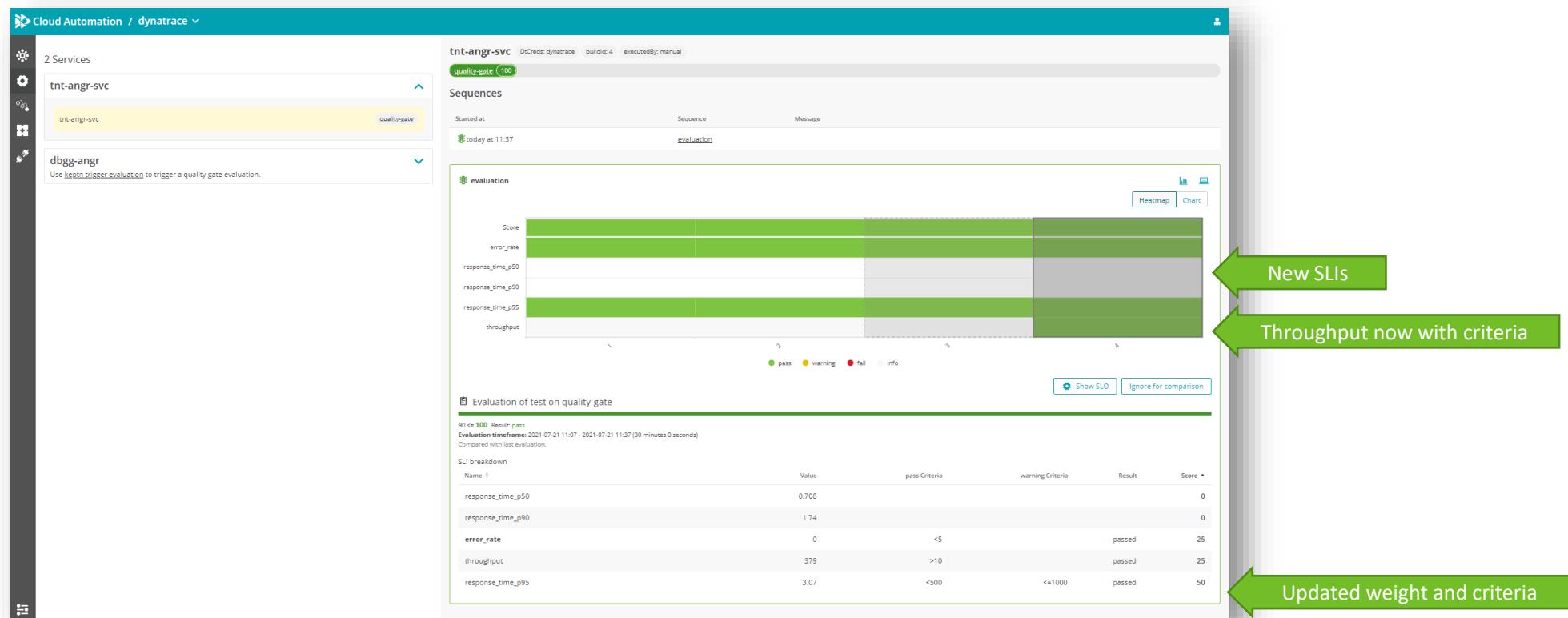
```
$ keptn add-resource --project=dynatrace --stage=quality-gate --service=journeyservice --resource=mylocalslo.yaml --resourceUri=slo.yaml
```

```
$ keptn add-resource --project=dynatrace --stage=quality-gate --service=journeyservice --resource=mylocalsli.yaml --resourceUri=dynatrace/sli.yaml
```

Phase #3 - Step #7: Run a new quality gate with the changed settings

```
$ keptn trigger evaluation --project=dynatrace --stage=quality-gate --service=tnt-xxxx-svc --timeframe=30m --labels=buildId=4,executedBy=manual
```

meta-data, e.g: buildId=4



Behind the scenes on dashboards: What gets generated?

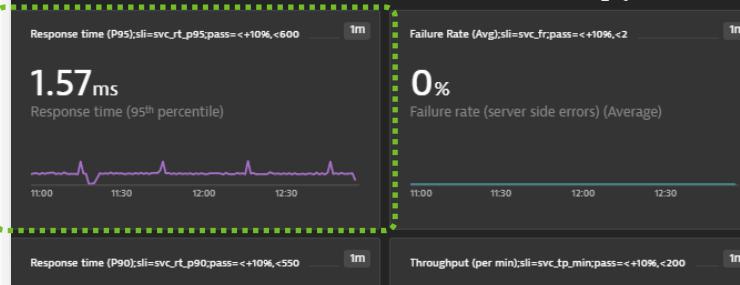
KQG;project=dynatrace;service=demo;stage=quality-gate

Welcome to your first SLI/SLO-based Quality Gate Dashboard - view results in your Keptn Bridge

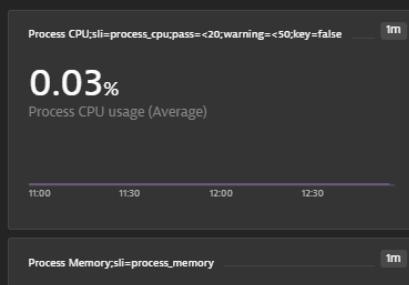
This default dashboard includes a set of base metrics (SLIs) that produce values in any Dynatrace deployment. Use this to make yourself familiar with defining your own SLIs (by adding more dimensions such as service, process, or host; however, splitting is supported by Keptn and is encouraged. For more best practices on how to create these SLI/SLO dashboards please have a look at our documentation.

KQG.Total.Pass=90%;KQG.Total.Warning=70%;KQG.Compare.WithScore=pass;KQG.Compare.Results=1;KQG.Compare.Function=avg

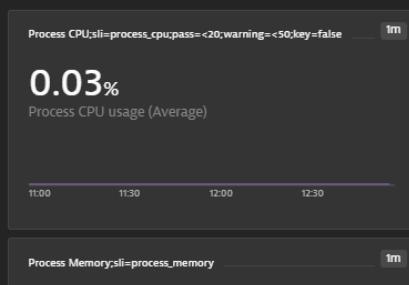
Service Performance (SLI/SLO)



Service Errors & Throughput (SLI/SLO)



Process Metrics (SLI/SLO)



Branch: quality-gate

dynatrace / demo / dynatrace / sli.yaml

15 lines | 1.5 KiB

```
1 spec_version: 0.1.4
2 indicators:
3   host_cpu: MV2;Percent;metricSelector=builtin:host.cpu.usage:merge(0):avg:names&entitySelector=type(HOST)
4   host_disk_queue: MV2;Count;metricSelector=builtin:host.disk.queueLength:merge(1):merge(0):max:names&entitySelector=type(HOST)
5   host_mem: MV2;Percent;metricSelector=builtin:host.mem.usage:merge(0):avg:names&entitySelector=type(HOST)
6   proc_count: MV2;Count;metricSelector=builtin:tech.generic.processCount:merge(0):avg:names&entitySelector=type(PROCESS_GROUP_INSTANCE)
7   process_cpu: MV2;Percent;metricSelector=builtin:tech.generic.cpu.usage:merge(0):avg:names&entitySelector=type(PROCESS_GROUP_INSTANCE)
8   process_memory: MV2;Byte;metricSelector=builtin:tech.generic.mem.workingSetSize:merge(0):avg:names&entitySelector=type(PROCESS_GROUP_INSTANCE)
9   svc_fr: MV2;Percent;metricSelector=builtin:service.errors.server.rate:merge(0):avg:names&entitySelector=type(SERVICE)
10  svc_rt_p50: MV2;MicroSecond;metricSelector=builtin:service.response.time:merge(0):percentile(50.000000):names&entitySelector=type(SERVICE)
11  svc_rt_p90: MV2;MicroSecond;metricSelector=builtin:service.response.time:merge(0):percentile(90.000000):names&entitySelector=type(SERVICE)
12  svc_rt_p95: MV2;MicroSecond;metricSelector=builtin:service.response.time:merge(0):percentile(95.000000):names&entitySelector=type(SERVICE)
13  svc_tp_min: MV2;Count;metricSelector=builtin:service.requestCount.total:merge(0):value:names&entitySelector=type(SERVICE)
14  svc_svc_calls: MV2;Count;metricSelector=builtin:service.nonDbChildCallCount:merge(0):value:names&entitySelector=type(SERVICE)
```

Dynatrace Metrics
Query based on
chart definition

SLO definition based on
chart title

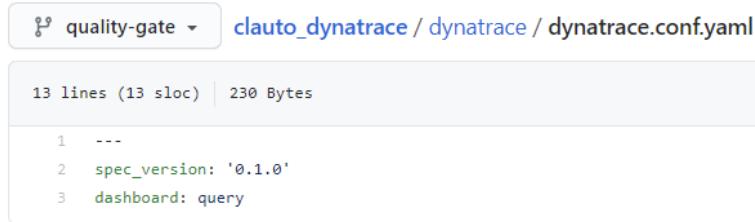
Branch: quality-gate
dynatrace / demo / slo.yaml

115 lines | 1.6 KiB

```
1 spec_version: ""
2 filter: {}
3 comparison:
4   compare_with: single_result
5   include_result_with_score: pass
6   number_of_comparison_results: 1
7   aggregate_function: avg
8   objectives:
9     - sli: proc_count
10    displayName: ""
11    pass: []
12    warning: []
13    weight: 1
14    key_sli: false
15    - sli: svc_rt_p95
16      displayName: ""
17      pass:
18        - <+10%
19        - <600
20      warning: []
21      weight: 1
22      key_sli: false
23      - sli: svc_rt_p90
24        displayName: ""
25        pass:
26          - <+10%
27          - <550
28        warning: []
29        weight: 1
30        key_sli: false
31        - sli: svc_rt_p50
32          displayName: ""
33          pass:
34            - <+10%
```

Behind the scenes: dynatrace.conf.yaml enables dashboard parsing

- Its *enabled by default* for the *quality-gate* stage in the *dynatrace* project



The screenshot shows a GitHub code editor interface. The top bar has a dropdown menu set to "quality-gate". Below it, the repository path "clauto_dynatrace / dynatrace / dynatrace.conf.yaml" is shown. The code editor displays the following YAML content:

```
1 ---  
2 spec_version: '0.1.0'  
3 dashboard: query
```

- Read all details on github
 - <https://github.com/keptn-contrib/dynatrace-sli-service#configurations-of-dashboard-slislo-queries-through-dynatraceconfyaml>
- Essentially you need to upload a dynatrace.conf.yaml on project, stage or service level
 - Either add or edit that file in your git upstream repository



The screenshot shows a GitHub code editor interface. The top bar has a dropdown menu set to "Branch: master". Below it, the repository path "dynatrace / dynatrace / dynatrace.conf.yaml" is shown. The code editor displays the following YAML content:

```
1 spec_version: '0.1.0'  
2 dashboard: query
```

- Our create that file and upload it through the keptn CLI like this

```
$ keptn add-resource --project=dynatrace --stage=quality-gate --service=yourservice --resource=dynatrace.conf.yaml --  
resourceUri=dynatrace/dynatrace.conf.yaml
```

Phase #3 - Troubleshooting if you receive

Git Upstream URL could not be set

- Validate your Git repository URL is internet accessible
 - Cloud Automation SaaS currently runs in the cloud and therefore needs access to that URL
 - You can validate by pinging the URL from a machine outside your company network
- Validate your Git username is valid
 - It should be the same username that shows up when logged in to your github, gitlab, azure ...
- Validate your Git token has the right privileges
 - See the required privileges [in the documentation](#)
- Improvement request for easier troubleshooting in UI
 - <https://github.com/keptn/keptn/issues/4626>

Quick Status Check: are we all good with accessing our environments?

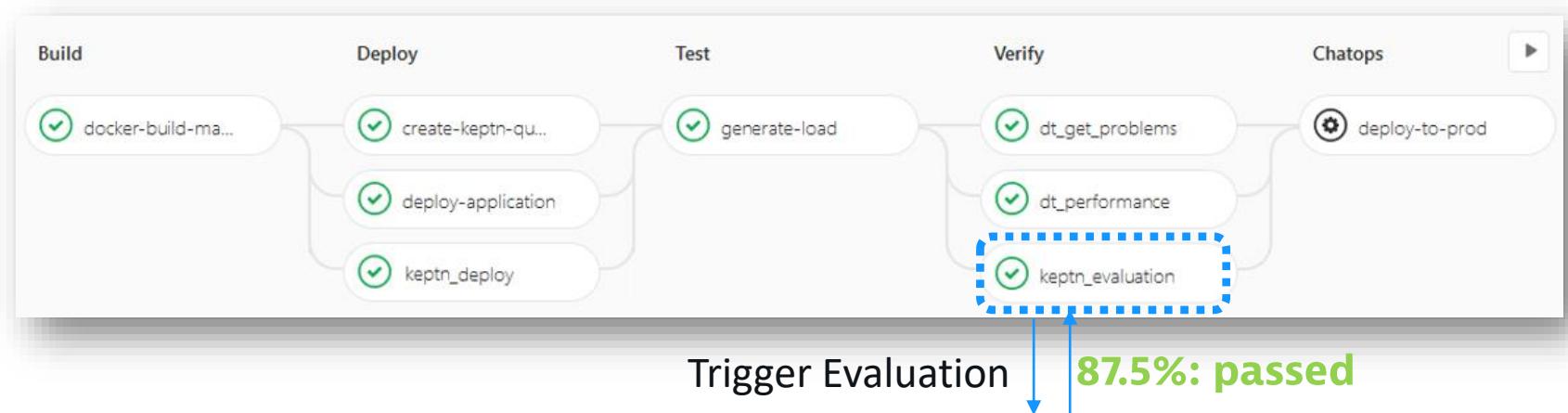
- Please mark your tasks accordingly in the Excel file

Codify Quality Gates		
Understand how to set Git Upstream for a Cloud Automation project	Customize SLI.yaml & SLO.yaml via Git Interface	Execute Quality Gate with modified SLI / SLO.yaml

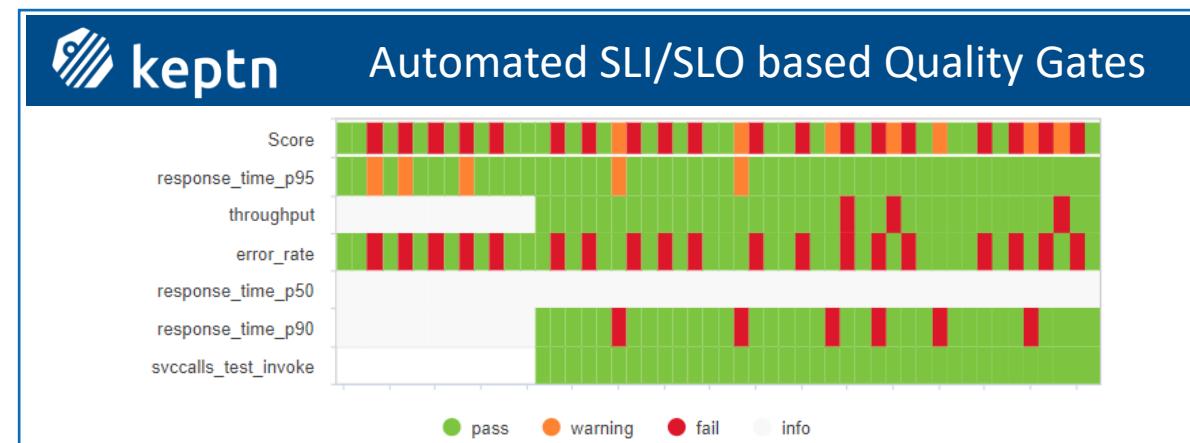
GitOps

Integrate into your existing CI/CD DevOps Processes & Tools

Example: Quality Gates accelerates GitLab Pipelines lead time by 80%



87.5%: passed



Pull SLI Metrics



Christian Heckelmann
Senior Systems Engineer



Phase #5 - Step 1: Trigger Quality Gate from your CI/CD, e.g: Jenkins, GitLab, Azure DevOps

- Either leverage the existing libraries & extensions for
 - Jenkins: <https://github.com/keptn-sandbox/keptn-jenkins-library/>
 - Azure DevOps: <https://github.com/keptn-sandbox/keptn-azure-devops-extension>
 - GitLab: <https://gitlab.com/checkelmann/keptn-templates>
- Or simply call the Keptn API yourself from your CI/CD tool like we did it in prior steps

Jenkins

Dashboard > Cloud Automation Quality Gates

Pipeline Cloud Automation Quality Gates

Last Successful Artifacts

- shipyard.yaml
- keptn.context.4.json
- keptn.evaluationresult.0ddde294-5908-408b-9357-6e0744ab1d62.json
- keptn.html

137 B view
56 B view
6.12 KB view
406 B view

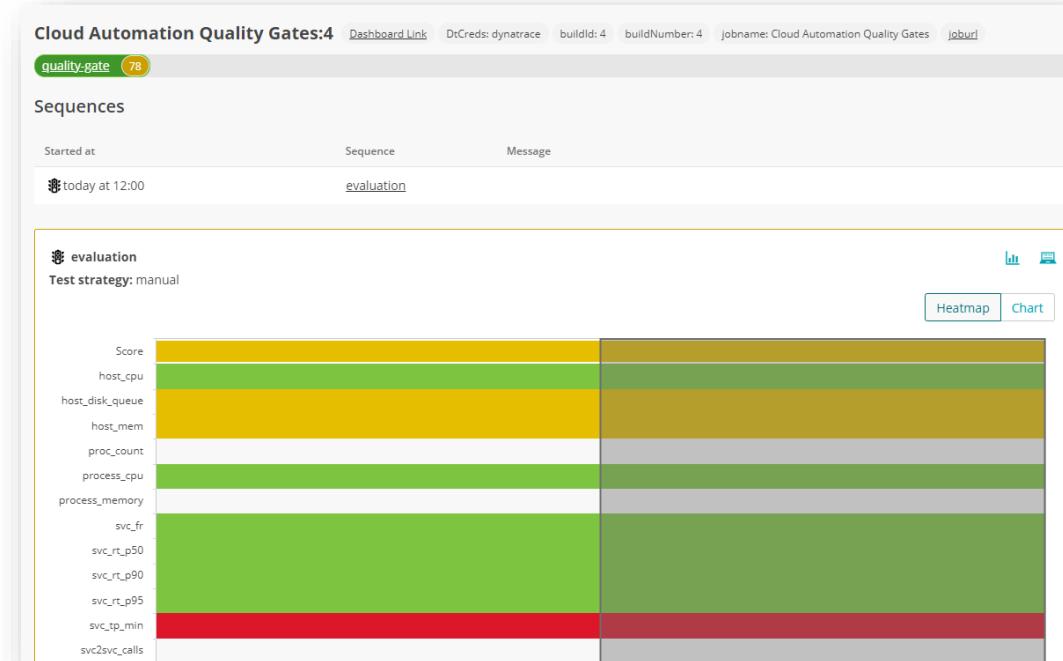
Recent Changes

Stage View

Average stage times:
(Average full run time: ~14s)

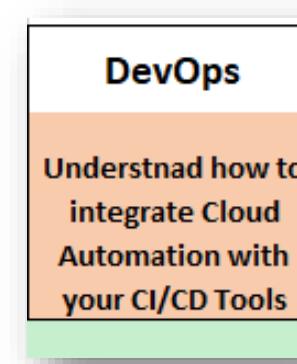
Stage	Time
Initialize Keptn	2s
Trigger Quality Gate	1s
Wait for Result	22s

#4 Jul 13, 2021 9:59 AM



Quick Status Check: are we all good with accessing our environments?

- Please mark your tasks accordingly in the Excel file



Extra

Monaco and Release Inventory

Behind the scenes of Release Inventory

- 3 Environment Variables define Application, Environment and Version
 - More on <https://www.dynatrace.com/support/help/how-to-use-dynatrace/cloud-automation/release-monitoring/>

```
- name: DT_RELEASE_VERSION
  valueFrom:
    fieldRef:
      fieldPath: "metadata.labels['app.kubernetes.io/version']"
- name: DT_RELEASE_PRODUCT
  value: "{{ .Values.keptn.project }}"
- name: DT_RELEASE_STAGE
  value: "{{ .Values.keptn.stage }}"
```

Properties and tags

[Environment]DT_APPLICATION_ENVIRONMENT: production [Environment]DT_APPLICATION_NAME: delivery-demo

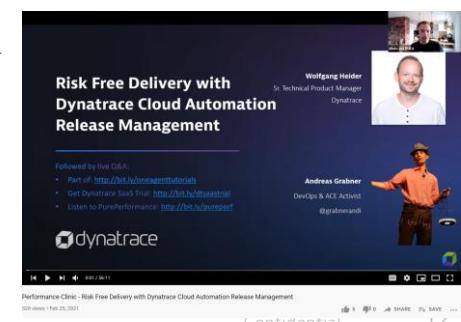
[Environment]DT_APPLICATION_RELEASE_VERSION: 3.0.0 [Environment]WorkshopTenant: angr

2 Releases

Name	Version	Stage	Product
tnt-angr-svc-*.delivery-demo...	4.0.0	production	delivery-demo
tnt-angr-svc-*.delivery-demo...	3.0.0	staging	delivery-demo

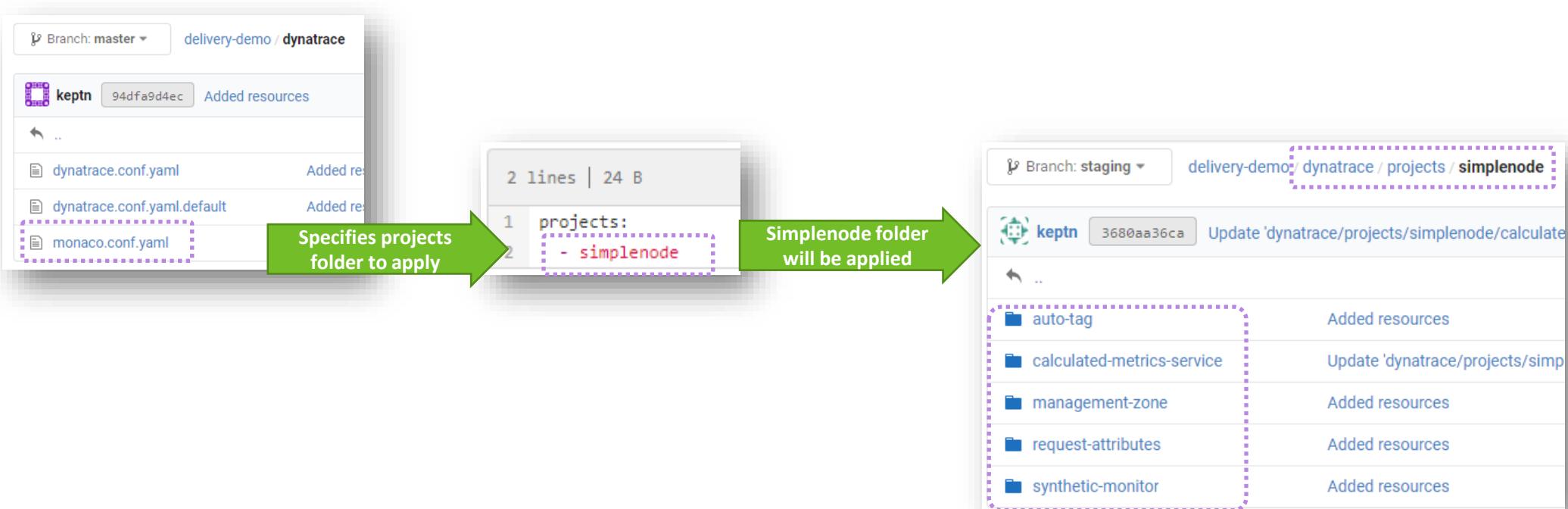
- More details about Release Management with Dynatrace in our Performance Clinic

- YouTube: <https://www.youtube.com/watch?v=TacwGZXX9pw&list=PLqt2rd0eew1YFx9m8dBFSiGYSBcDuWG38&index=17>



Lets explore Monaco (Monitoring as Code)

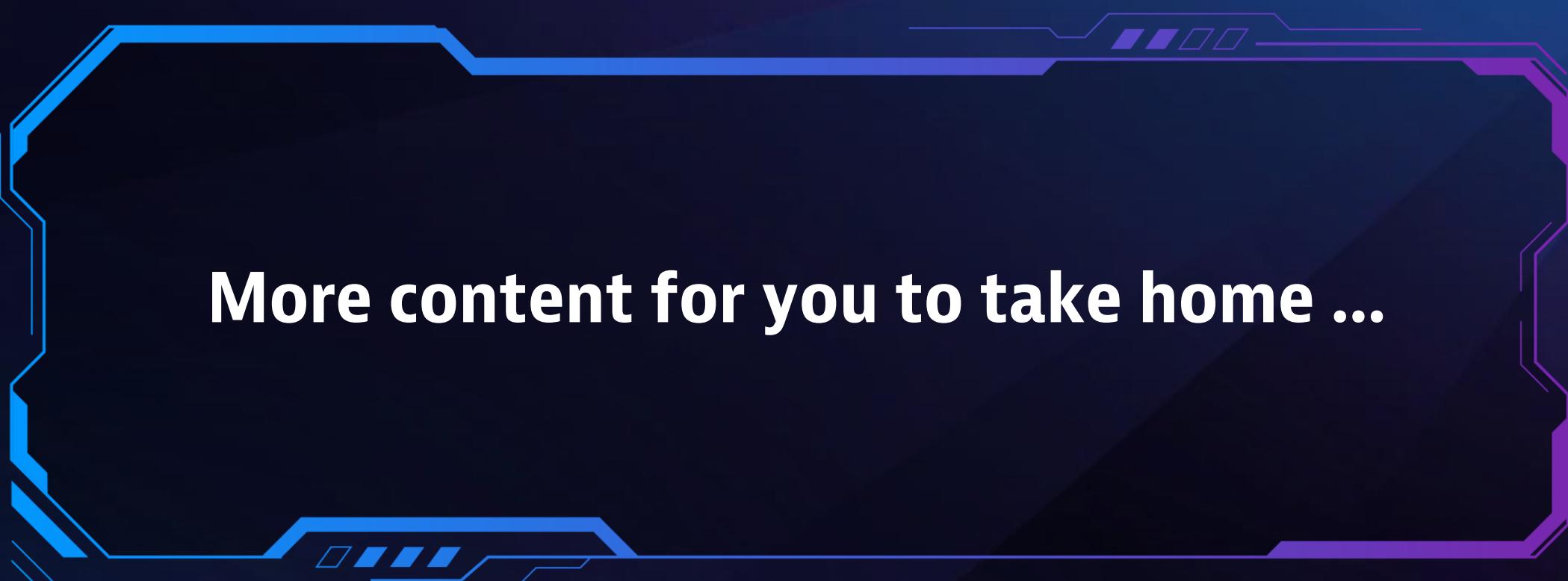
- Monaco is a Dynatrace Open Source project driven by our own internal Autonomous Cloud Team
 - <https://dynatrace-oss.github.io/dynatrace-monitoring-as-code/>
 - You can simply use it from the command line to apply any dynatrace configuration!
- Cloud Automation can use Keptn's monaco-service to apply Monaco configuration to Dynatrace



Quick Status Check: are we all good with accessing our environments?

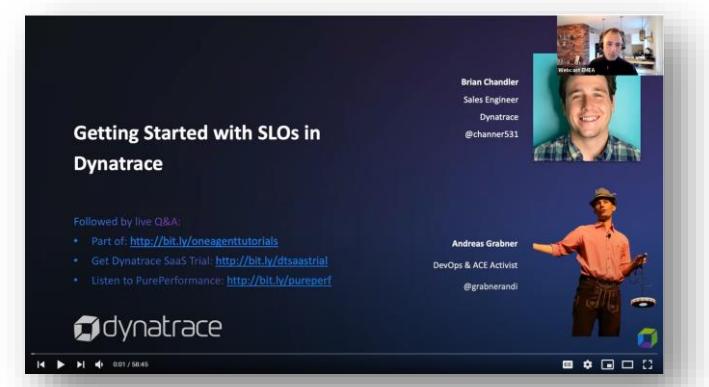
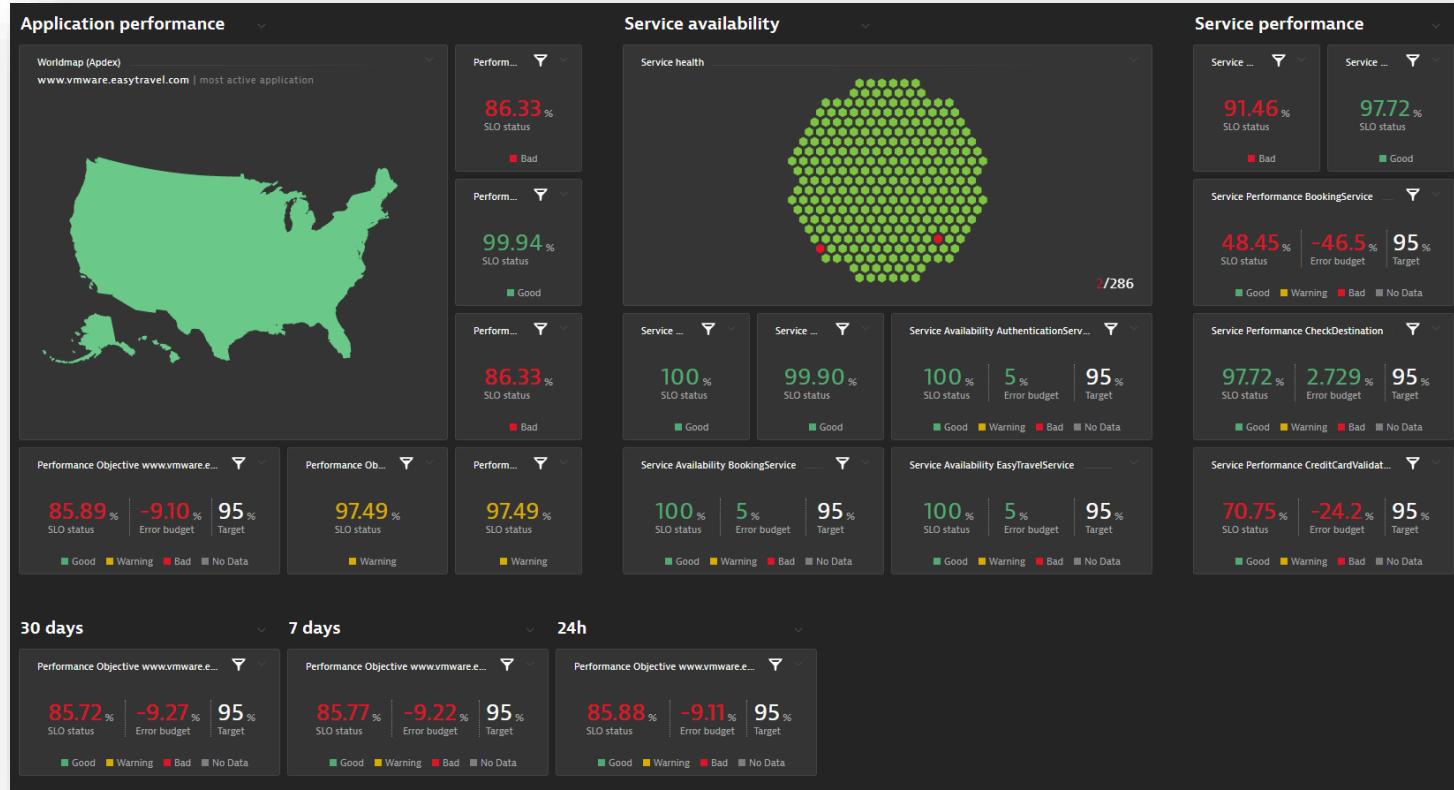
- Please mark your tasks accordingly in the Excel file

Monaco & Release Inventory	
Understand the basics behind Release Inventory and where to look for more	Understand the basics behind Monaco and where to look for more



More content for you to take home ...

Service Level Objectives for Production Monitoring



https://www.youtube.com/watch?v=4fhcxCa3m_c

Release Awareness / Version Detection

Releases

Release monitoring

Overview of deployed component versions and release events. For details, see [Release monitoring](#) or activate demo mode to view sample data.

Filter by

Monitor state

- Any (selected)
- Active
- Inactive

Problem impact

- Any (selected)
- Impacted
- Not impacted

Security vulnerability

- Any (selected)
- Detected
- Not detected

Release inventory

59 Releases

Name	Version	Stage	Product	Instances	Throughput
helm-service-*:keptn	0.8.6	keptn	-	2	-
tnt-addo-svc-*:delivery-dem...	1.0.0	production	delivery-demo	1	-
tnt-addo-svc-*:delivery-dem...	1.0.0	staging	delivery-demo	1	-
tnt-angr-svc-*:delivery-dem...	1.0.0	production	delivery-demo	1	-
tnt-angr-svc-*:delivery-dem...	1.0.0	staging	delivery-demo	1	-
tnt-brng-svc-*:delivery-dem...	1.0.0	production	delivery-demo	1	-
tnt-brng-svc-*:delivery-dem...	1.0.0	staging	delivery-demo	1	-
tnt-chsa-svc-*:delivery-dem...	1.0.0	production	delivery-demo	1	-
tnt-chsa-svc-*:delivery-dem...	1.0.0	staging	delivery-demo	1	-
tnt-cosm-svc-*:delivery-dem...	1.0.0	production	delivery-demo	1	-
tnt-cosm-svc-*:delivery-dem...	1.0.0	staging	delivery-demo	1	-
tnt-duhe-svc-*:delivery-dem...	1.0.0	production	delivery-demo	1	-
tnt-duhe-svc-*:delivery-dem...	1.0.0	staging	delivery-demo	1	-
tnt-dyst-svc-*:delivery-dem...	1.0.0	production	delivery-demo	1	-

Release events

44 events match your query and filtering

Events

Time Details

- 29 Custom in...
- 12 Deployme...
- 3 Process rest...
- Process tnt-cosm-svc-*:delivery-demo-pro... today, 17:49
- Process tnt-kaca-svc-*:delivery-demo-pro... today, 17:49
- Process tnt-kaca-svc-*:delivery-demo-stagi... today, 17:34
- Deploy tnt-duhe-svc 1.0.0 with strategy us... today, 17:06
- Evaluation result: warning today, 17:05
- Deploy tnt-luwr-svc 1.0.0 with strategy us... today, 17:05

Risk Free Delivery with Dynatrace Cloud Automation Release Management

Followed by live Q&A:

- Part of: <http://bit.ly/oneagenttutorials>
- Get Dynatrace SaaS Trial: <http://bit.ly/dtsaastrial>
- Listen to PurePerformance: <http://bit.ly/pureperf>

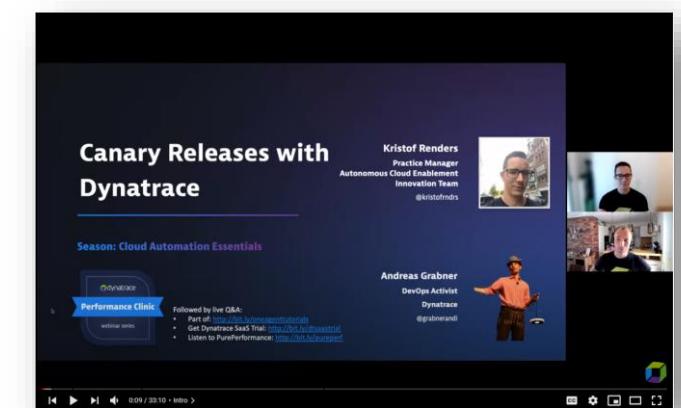
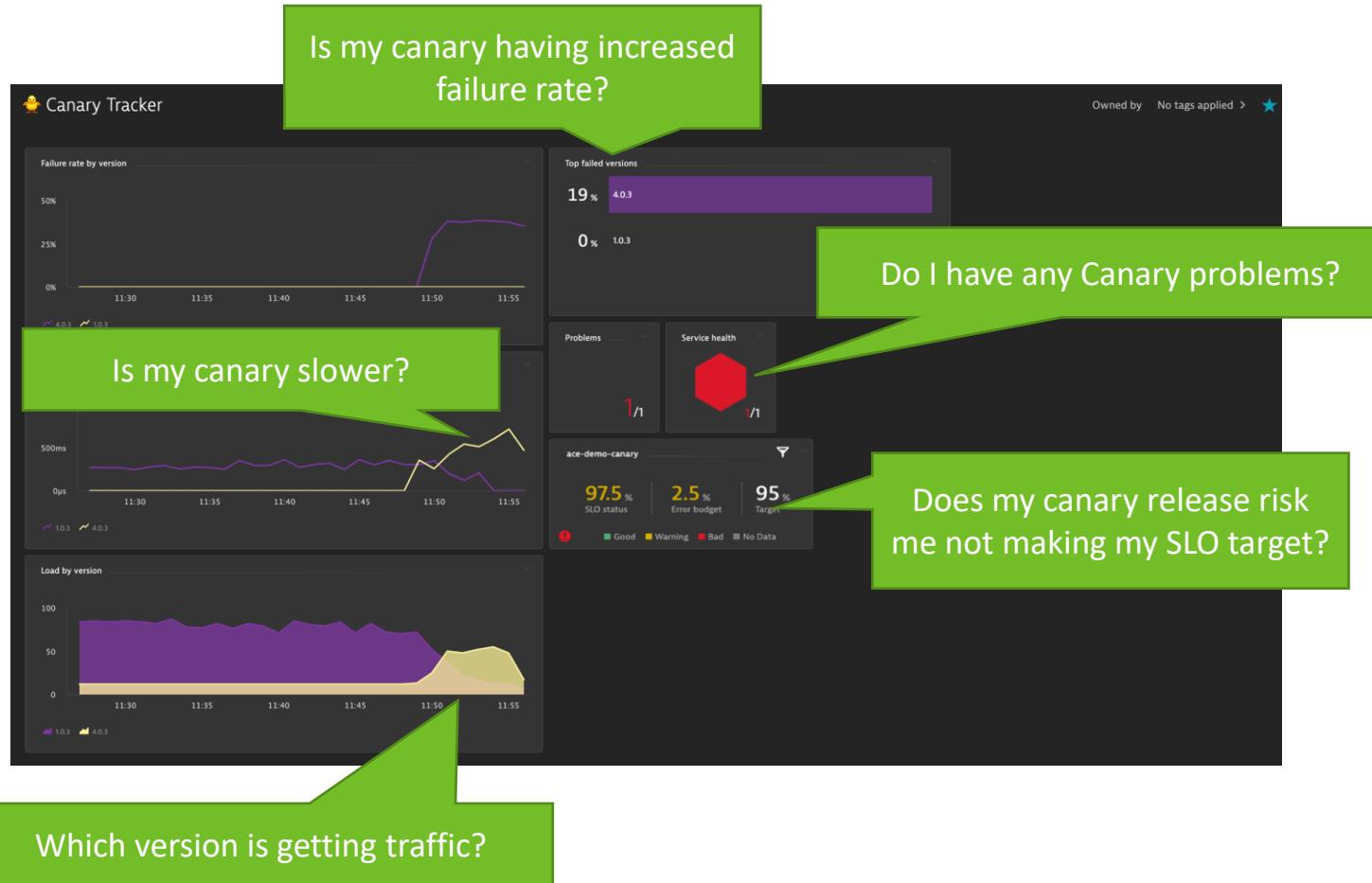
Wolfgang Heider
Sr. Technical Product Manager
Dynatrace

Andreas Grabner
DevOps & ACE Activist
@grabnerandi

dynatrace

https://www.youtube.com/watch?v=TacwGZXX9pw

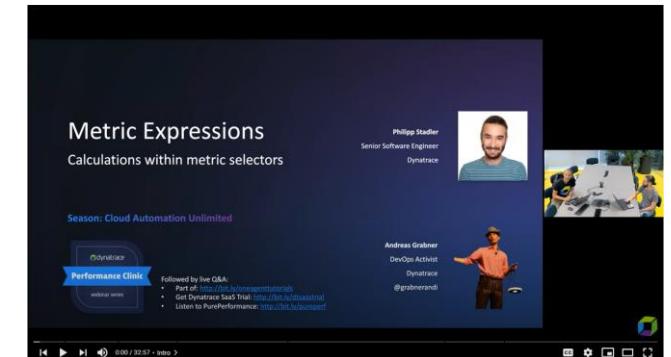
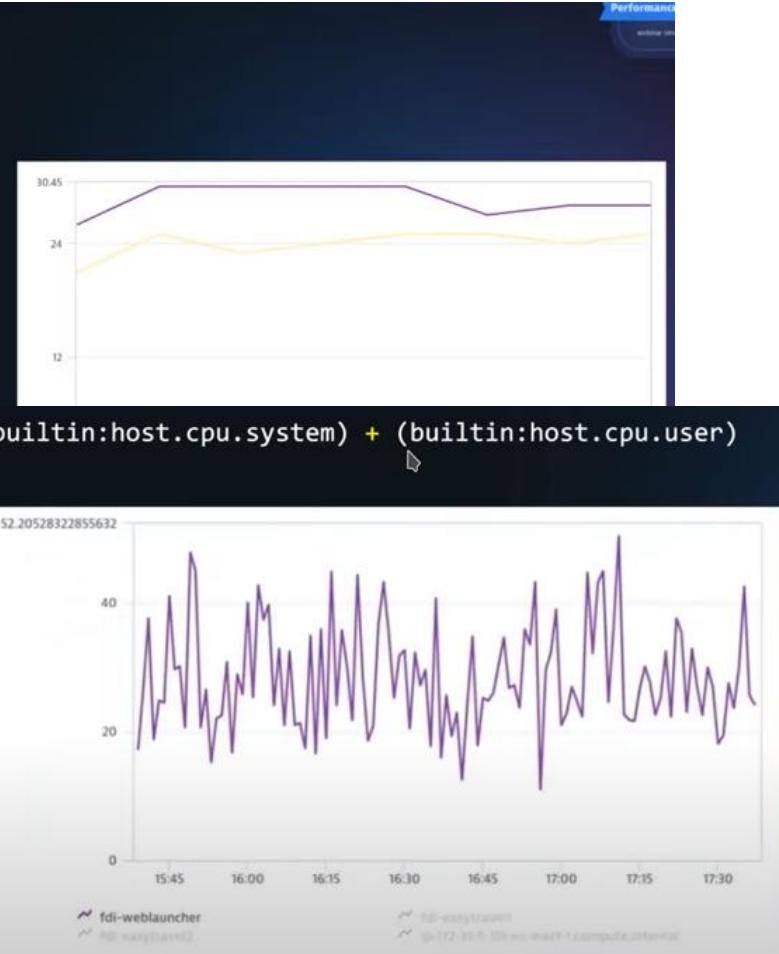
Better Zero Downtime Deployment Decisions for Canary, Blue/Green, Feature Flagging



<https://www.youtube.com/watch?v=VgBw3amiL-c>

Metrics expressions give better answers through calculation support

Background: What is a Metric Selector?



For more details on the dashboarding check out the following Performance Clinic



The screenshot shows a YouTube video player. The title of the video is "Building an SLO-based Quality Gate in 5 Minutes with Dynatrace & Keptn". Below the title, it says "Followed by live Q&A:" followed by a bulleted list: "Part of: <http://bit.ly/onlineperfclinic>", "Get Dynatrace SaaS Trial: <http://bit.ly/dtsaastrial>", and "Listen to PurePerformance: <http://bit.ly/pureperf>". At the bottom of the video player, there are logos for "dynatrace" and "keptn". The video player also includes standard controls like play, volume, and a progress bar showing 0:02 / 58:28.

Building an SLO-based Quality Gate in 5 Minutes with Dynatrace & Keptn

Followed by live Q&A:

- Part of: <http://bit.ly/onlineperfclinic>
- Get Dynatrace SaaS Trial: <http://bit.ly/dtsaastrial>
- Listen to PurePerformance: <http://bit.ly/pureperf>

Andreas Grabner
DevOps & ACE Activist
@grabnerandi

  keptn

<https://www.youtube.com/watch?v=650Gn--XEQE&list=PLqt2rd0eew1YFx9m8dBFSiGYSBcDuWG38&index=12&t=2s>