Full Stack Self-Service Diagnostics with Dynatrace

dynatrace
Perform

Stefan Hofbauer
Team Lead Sales Engineer

Sergio Hinojosa
Sales Engineering Manager

Omar Mazzeo
Sales Engineer

Prerequisites

- Personal AWS Account:
  - https://aws.amazon.com/free/
- Postman API Client
  - https://www.getpostman.com/downloads/
- Download/Clone the Repository:
  - https://github.com/stehofbauer/Dynatrace-Hotday-2019-SelfServiceDiagnostics

  or
  - https://tinyurl.com/yydxjm4m

Agenda

- Formalities
  - Introductions
  - What we're covering
  - Level of Expertise required
  - Individual EC2 Instances
  - EasyTravel – Book Exciting travel to far away destinations
- Setup and Launching our EC2 instance with EasyTravel
- Configuration API
- Use Case 1 – CPU Increase
- Use Case 2 – Booking Failure
- Use Case 3 – Slow Login
- Use Case 4 – Slow Loading Recommendations
- Tips and Tricks

What we're covering/not covering

- This is an exploration of common
- This is not a product tutorial
- This is not a test
- There is no need to be an AWS expert and everything we are covering can be applied to ANY environment (e.g. Azure, GCP, Kubernetes, On-Prem, etc...)
- The goal of this sessions is:
  - Help you become more comfortable using Dynatrace to diagnose application problems
  - Make the diagnosis easy for others in your organization
  - Not everybody needs to be a "Dynatrace Expert"

How we're covering

- We will walk through several Use Case Problem patterns
- Each Student will have their own private EC2 instance running a version of EasyTravel (created via CloudFormation)
- Each Student will have their own private Dynatrace SaaS tenant
  - Tenant ID and Credentials on your attached card

- During the course of this session we will be doing the following:
  - Creating our environment via CloudFormation
  - Performing Dynatrace configurations via the UI and the API
  - Triggering EasyTravel Problem Patterns via the EasyTravel Config UI

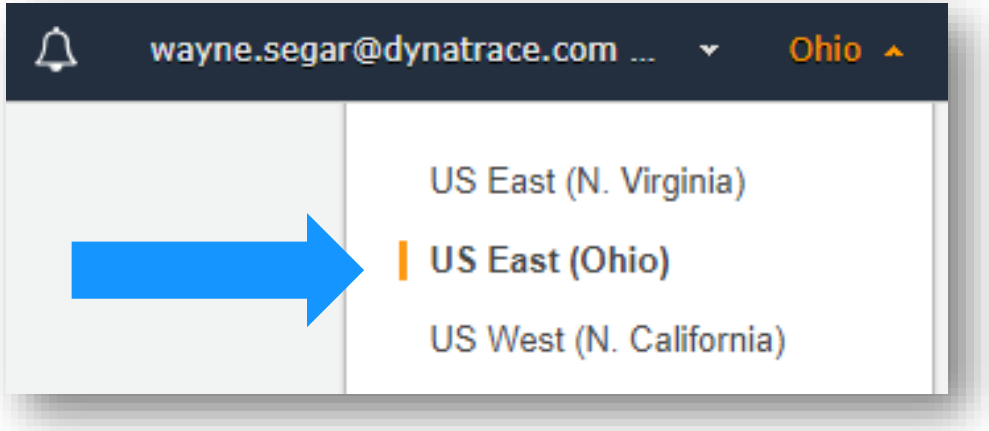
Level of Expertise

- Basic Dynatrace familiarity
- Expert certification not required
- REST API basics

Setup Work

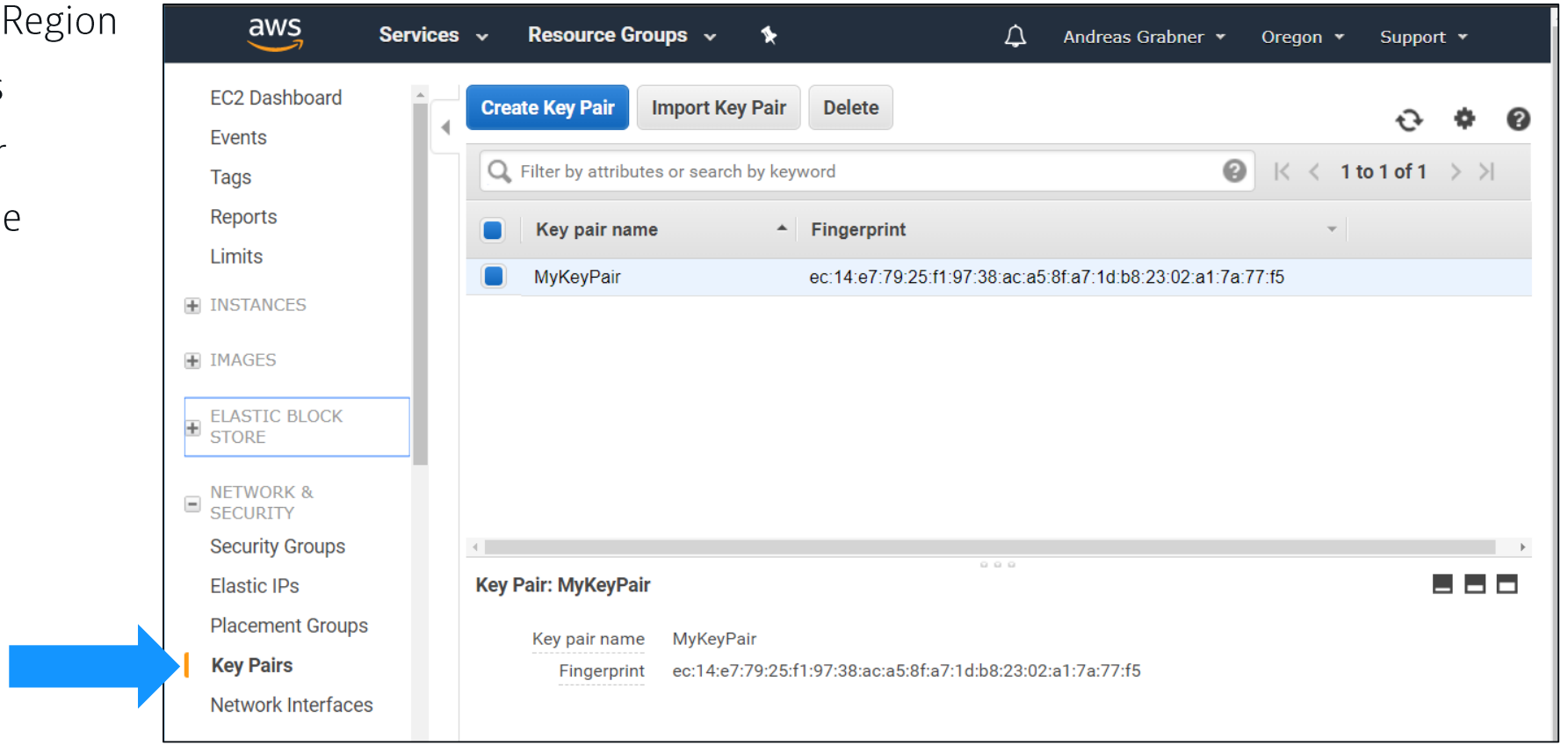
Setup AWS: Default Region

- Login to your AWS Account: https://aws.amazon.com
  - Validate and make note of your default region



Setup AWS: EC2 Key Pair

1. Create EC2 Key Pair
    1. Go to EC2 – Pick Region
    2. Click on Key Pairs
    3. Create a new Pair
    4. Store the .pem file

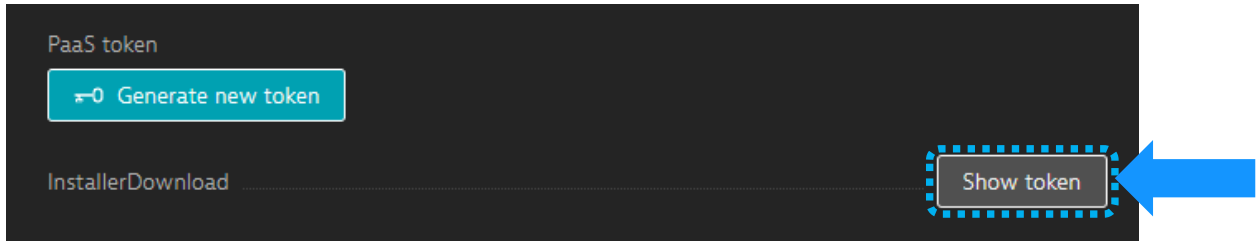


Setup Dynatrace: Retrieve API Token

- Login to your Dynatrace SaaS Tenant provided to you:

https://<TENANT_ID>.sprint.dynatracelabs.com

- Navigate to:
  - Click "Deploy Dynatrace" from the left side menu
  - Click "Start Installation"
  - Click "Set up PaaS monitoring"
  - Click "Show token"



  - Copy your token to your clipboard

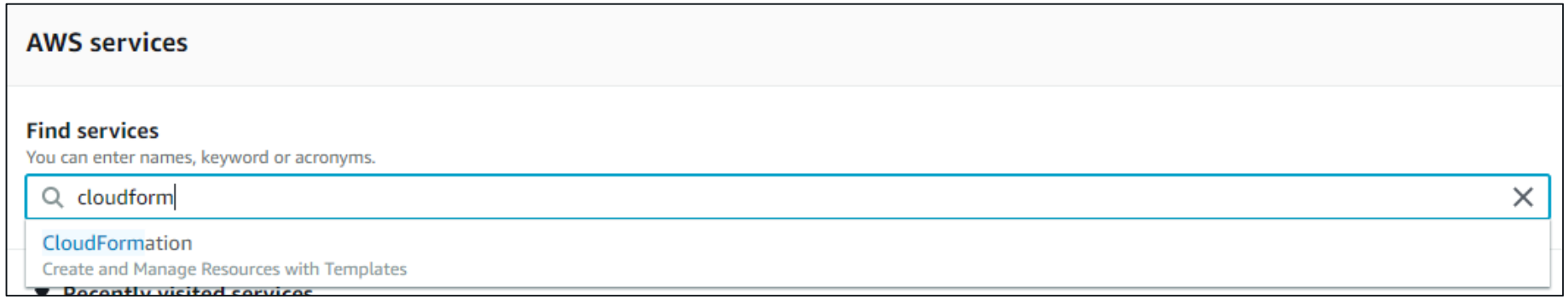
Launching our EC2 Instance

Our CloudFormation Template will create a Production Linux server with EasyTravel installed/running and deploy a OneAgent pointing to your Dynatrace SaaS Tenant

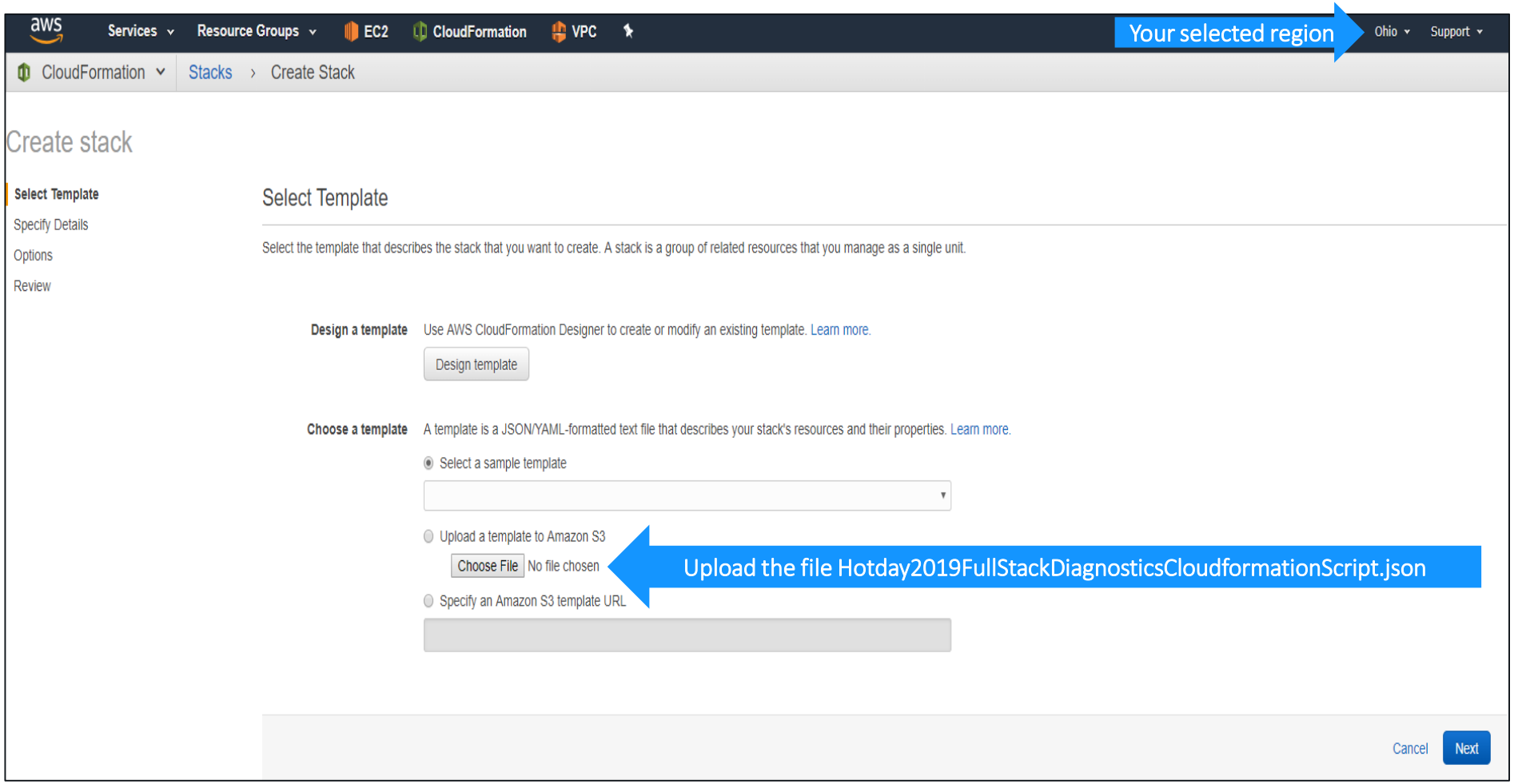
dynatrace Perform

Launching the CloudFormation Stack

- What is CloudFormation?
  - A declarative way to define AWS Resources, e.g: EC2, CodeDeploy, CodePipeline, Lambda, Roles, ELBs, ...
  - Learn more about CloudFormation: https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide
- Our Steps
  - In your AWS Console browse to the CloudFormation Service



Upload Template



CloudFormation can be found in the GitHub repository

Launching the Stack

Upload Template

Filling in the Details

Create and Confirm

Follow Events

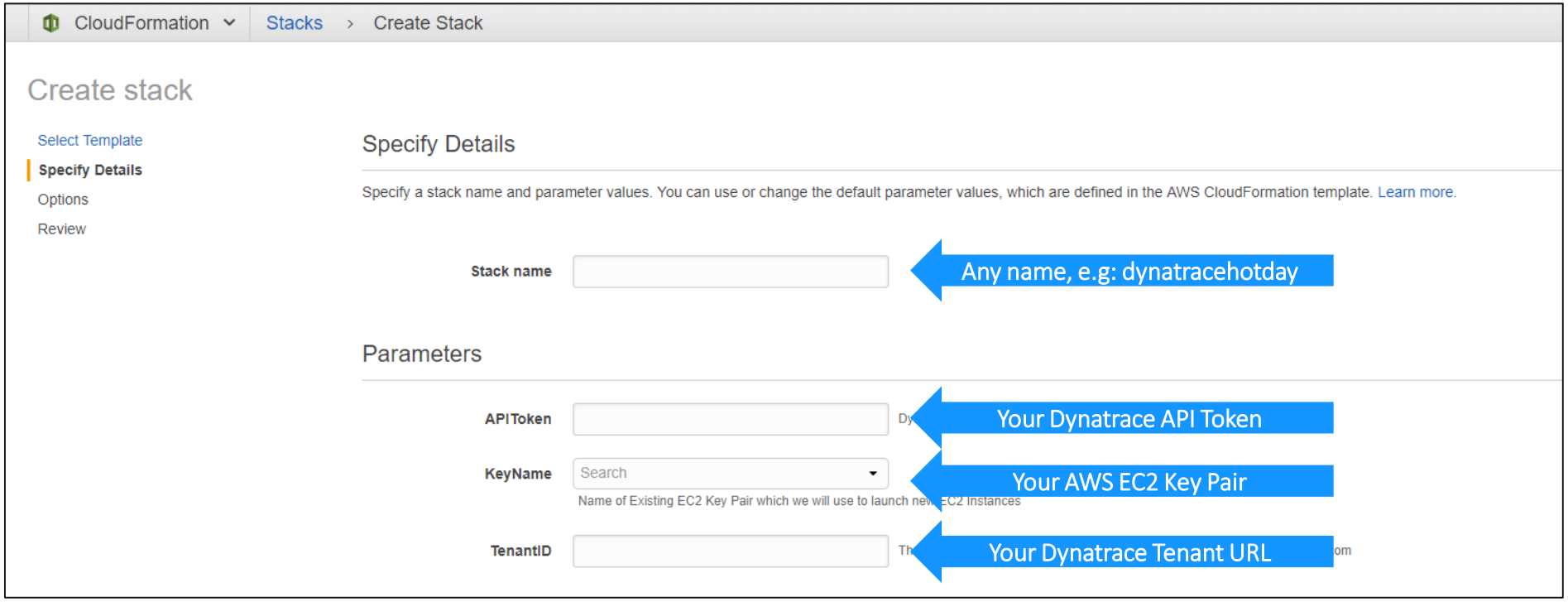
View Outputs

Validate Environment

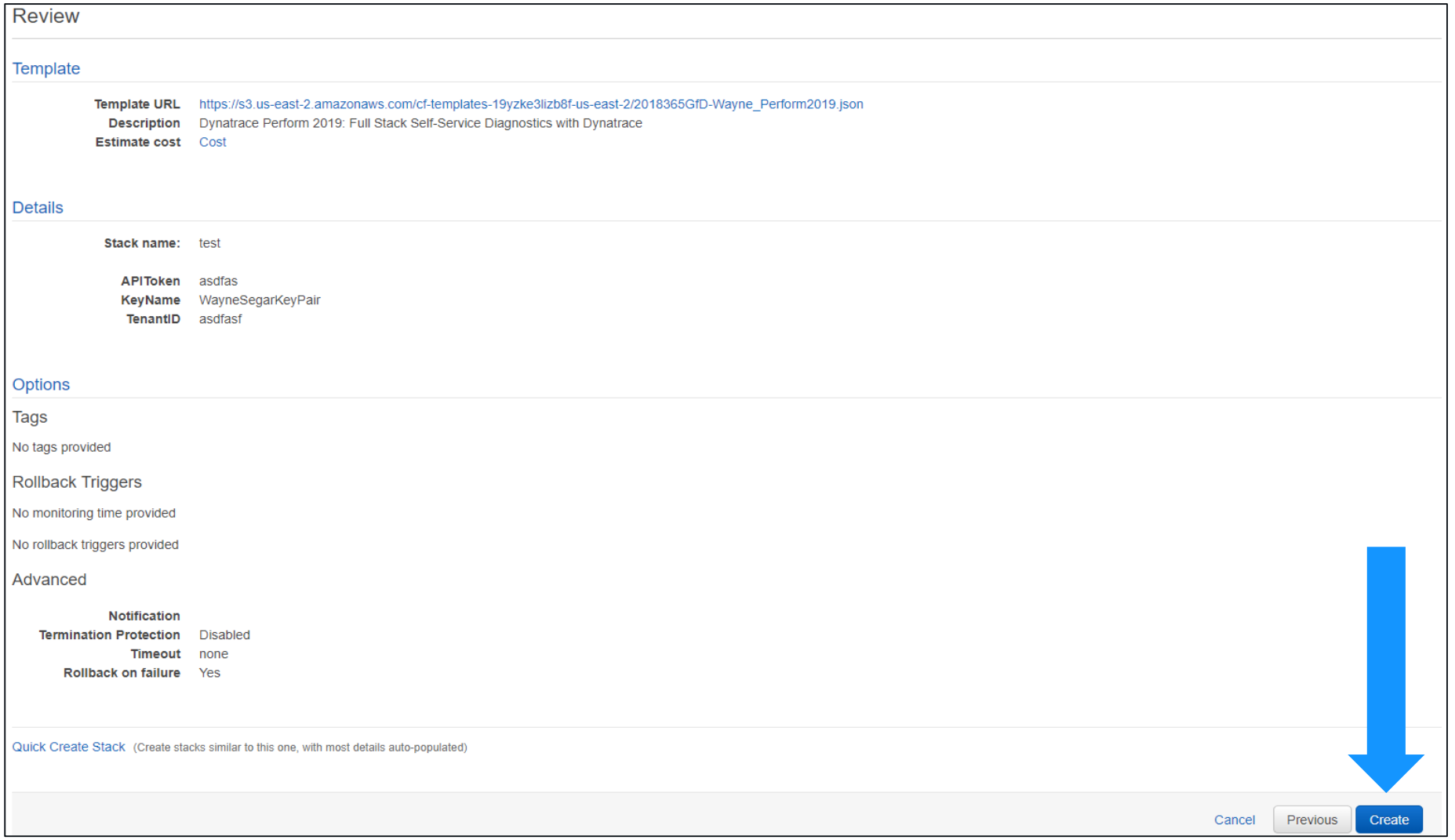
Cleanup

dynatrace Perform

Filling in the Details



Create and Confirm



Launching the Stack

Upload Template

Filling in the Details

Create and Confirm

Follow Events

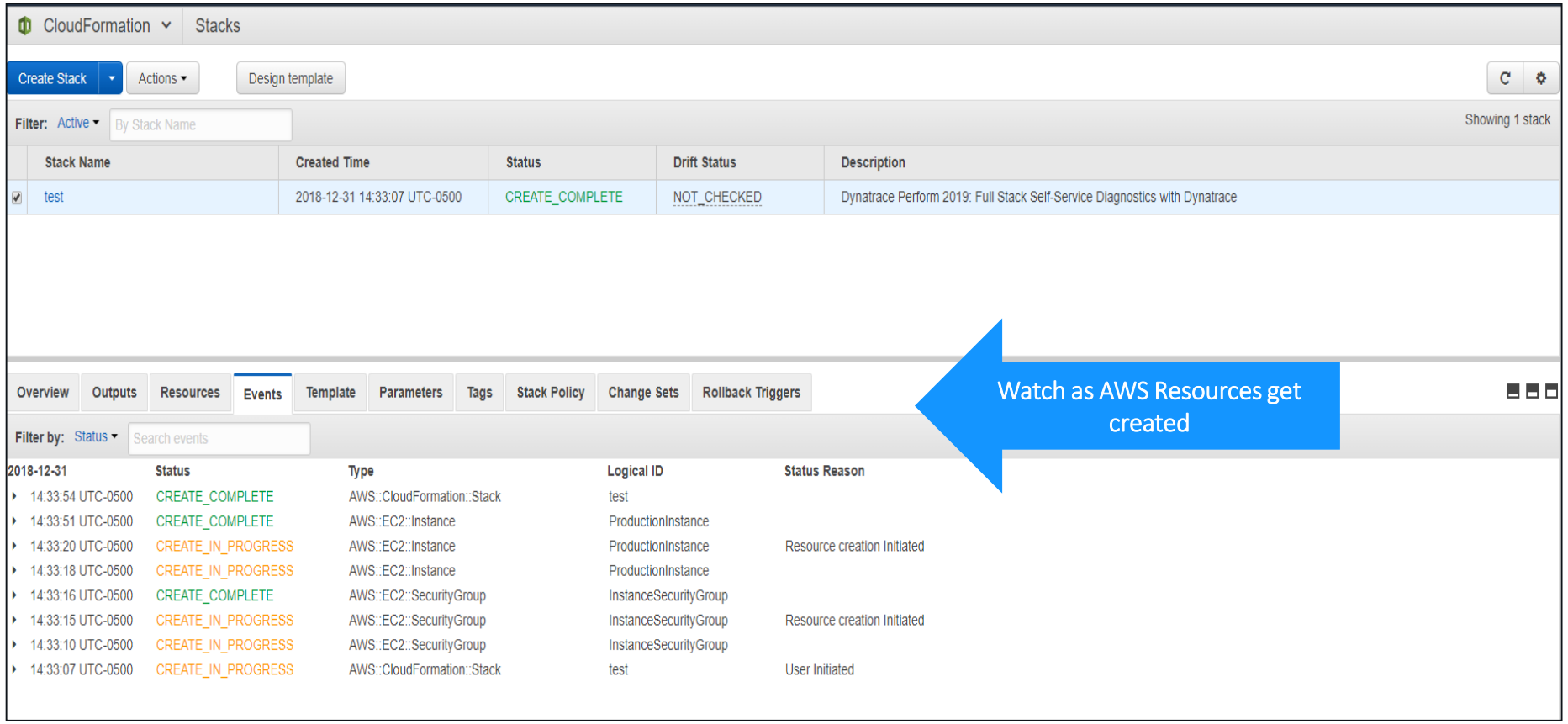
View Outputs

Validate Environment

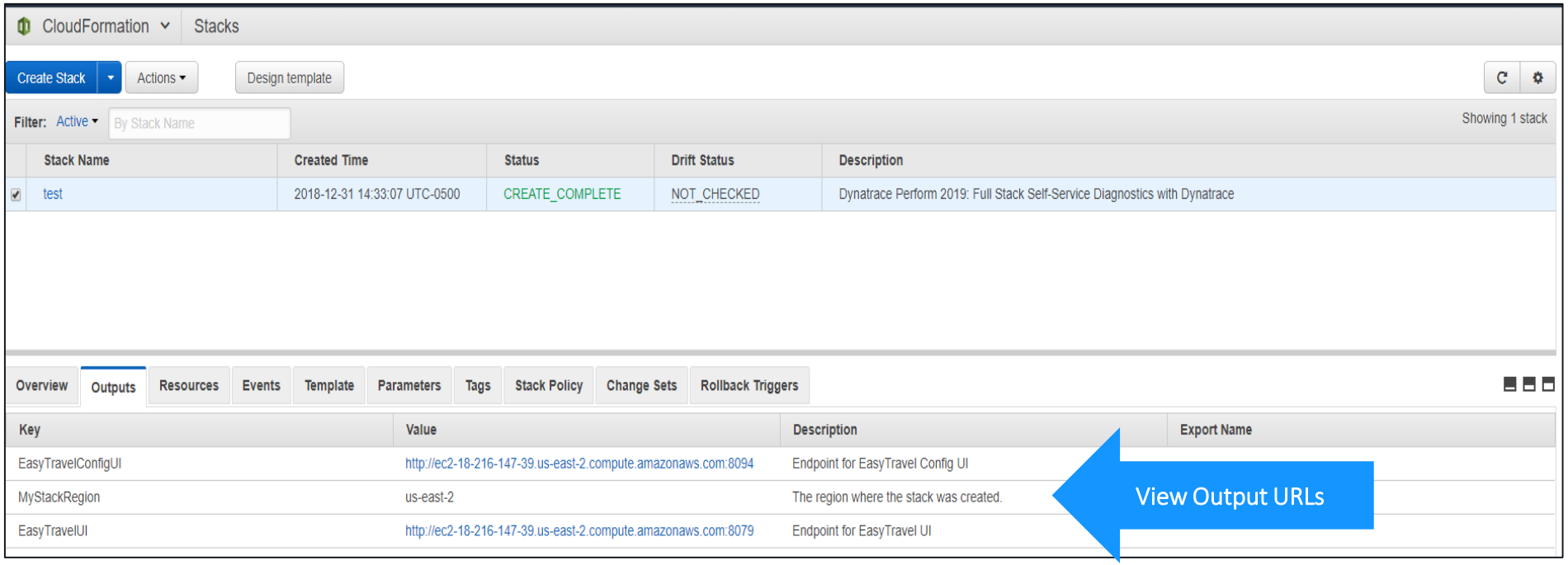
Cleanup

dynatrace Perform

Follow Events

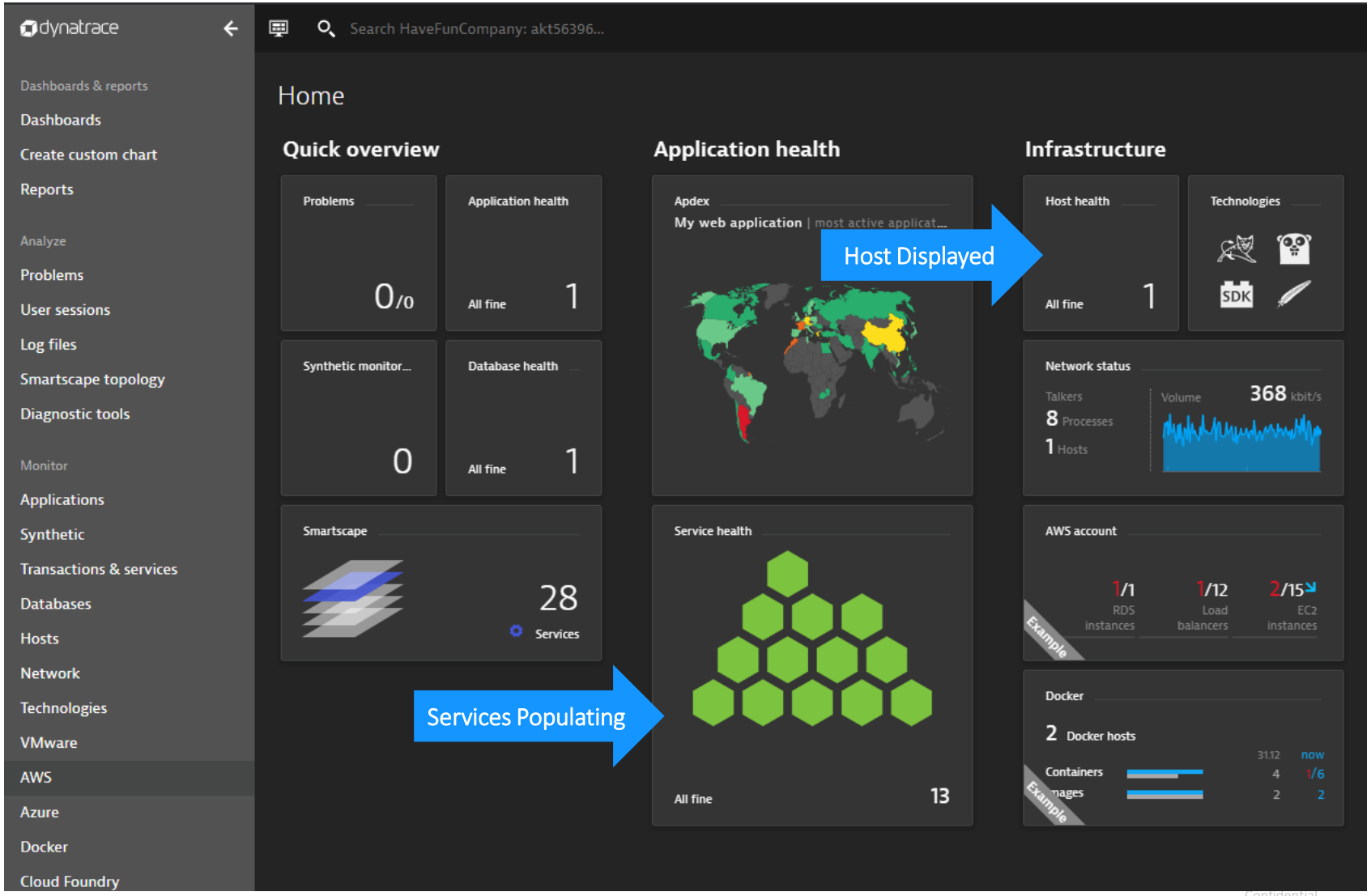


View Outputs



Note: It will take roughly 10 minutes for EasyTravel to fully come up

Validate Environment



Launching the Stack

Upload Template

Filling in the Details

Create and Confirm

Follow Events

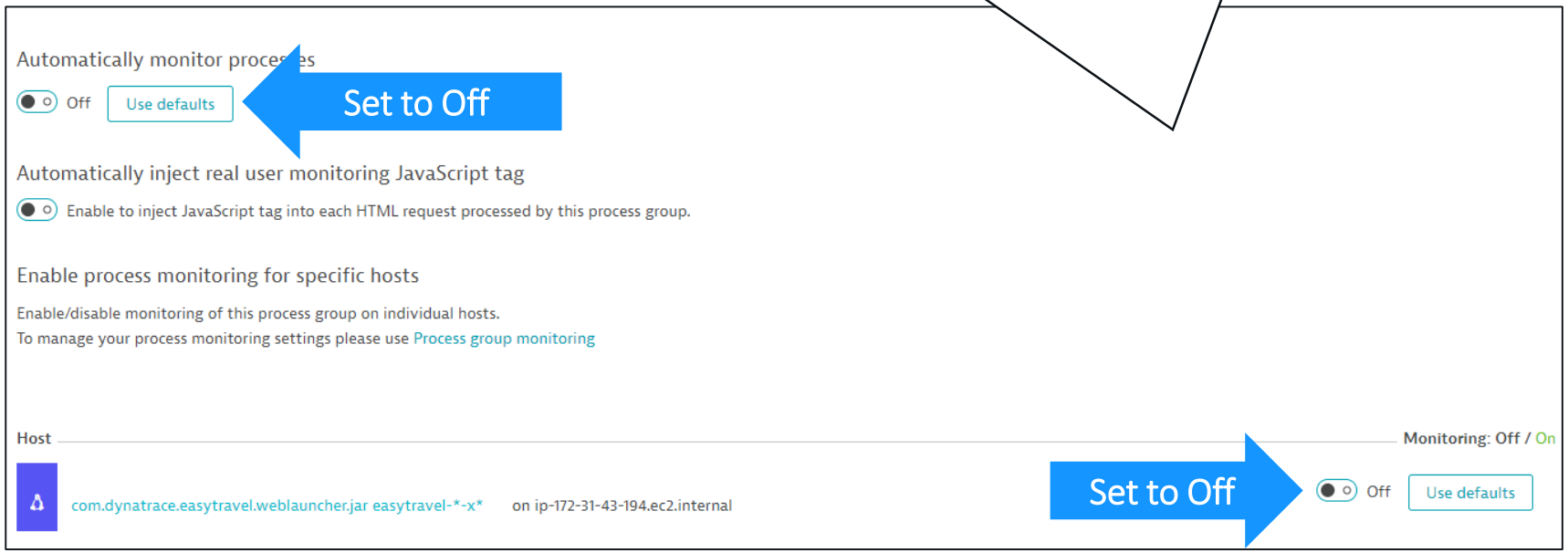
View Outputs

Validate Environment

Cleanup

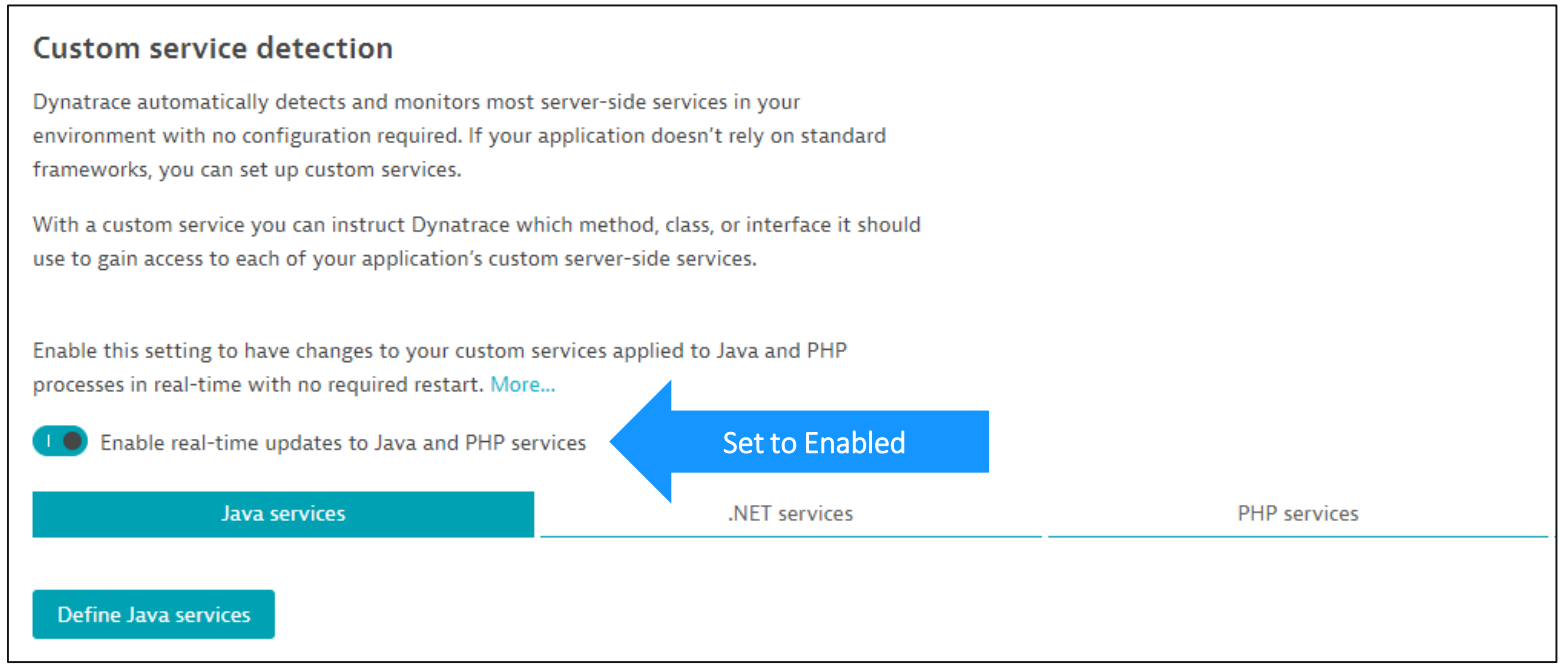
Would it be nice if this was automated via the ConfigAPI?
Vote for RFE: https://bit.ly/2RGytHF

details > Edit > Disable for the host.



Cleanup

- Enable Java Hotsensor Placement
  - We will be creating Custom Services in the next section so we will need this in order to avoid a required restart of the Java Processes
  - Settings > Server-side > service monitoring > Custom service detection > Enable real-time updates to Java and PHP services



Configuration API

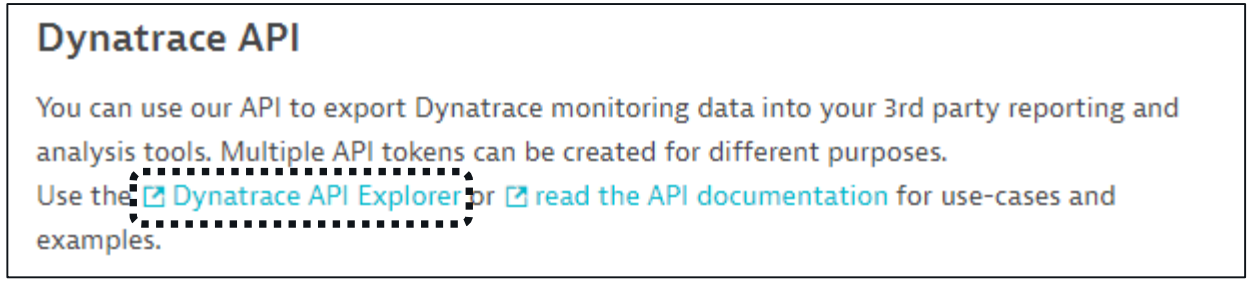
Dynatrace Configuration API

- The Dynatrace Configuration API allows API consumers to globally read and write the configuration settings of any Dynatrace environment

- The primary use cases for the Configuration API are:
  - Read and copy an existing environment configuration over to a new environment.
  - Create or change individual configuration settings via the API when setting up a new environment.
  - Read and store configurations, along with their histories, in a version management system (for example, Git).
  - Provide a mechanism to automate Monitoring/Performance-as-a-Service
    - Provide easy access to meaningful data to Application Owners with creation of Management Zones
    - Simplify complex environment with auto tagging rules
    - Extract specific business context information with request attributes

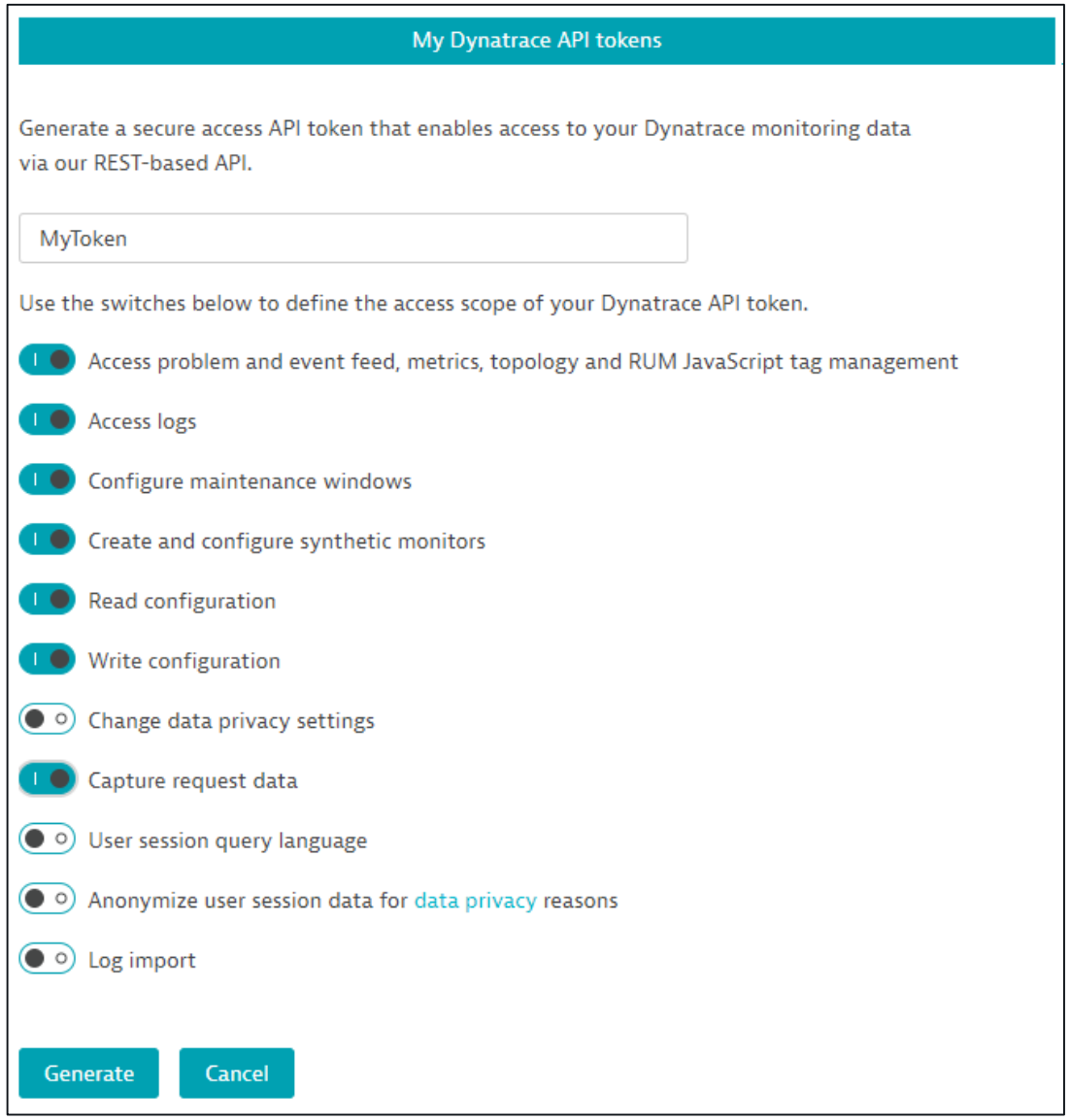
View the Configuration API Explorer

- In your Dynatrace tenant navigate to:
  - Settings > Integration > Dynatrace API
  - Click "Dynatrace API Explorer"



Create API Token to use the Config API

- In the Dynatrace API screen, click “Generate token”
- Enter a name for your token (ex. MyToken)
- Make sure to enable the following switches
- Click the “Generate” button

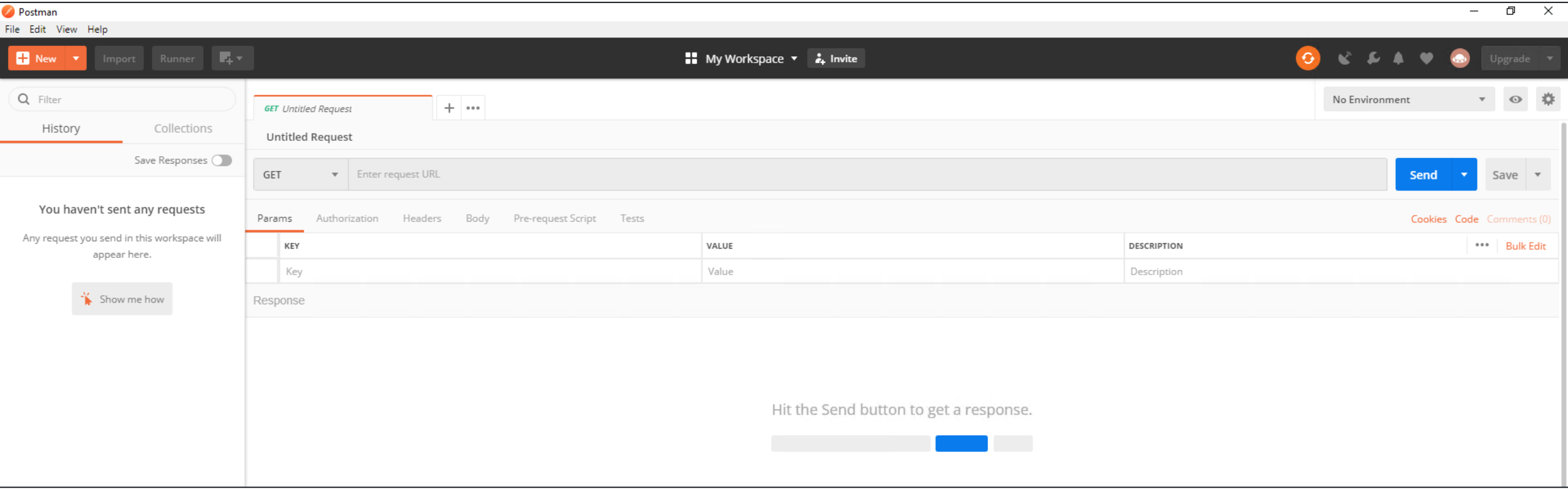


Using the ConfigAPI via Postman

- We will be using Postman to run a series of API requests to the ConfigAPI in order to do the following:
  - Create Auto Tagging rules
  - Create Management Zones
  - Application setup (URL Grouping and Framework support)
  - Create Custom Services
  - Create Request Attributes
  - Create Synthetics Monitors
  - Create a Dashboard
- You can use any API client to interact with any of the Dynatrace APIs, however, for this course we are using Postman because it is simple to share a collection of API requests.

Setup Postman

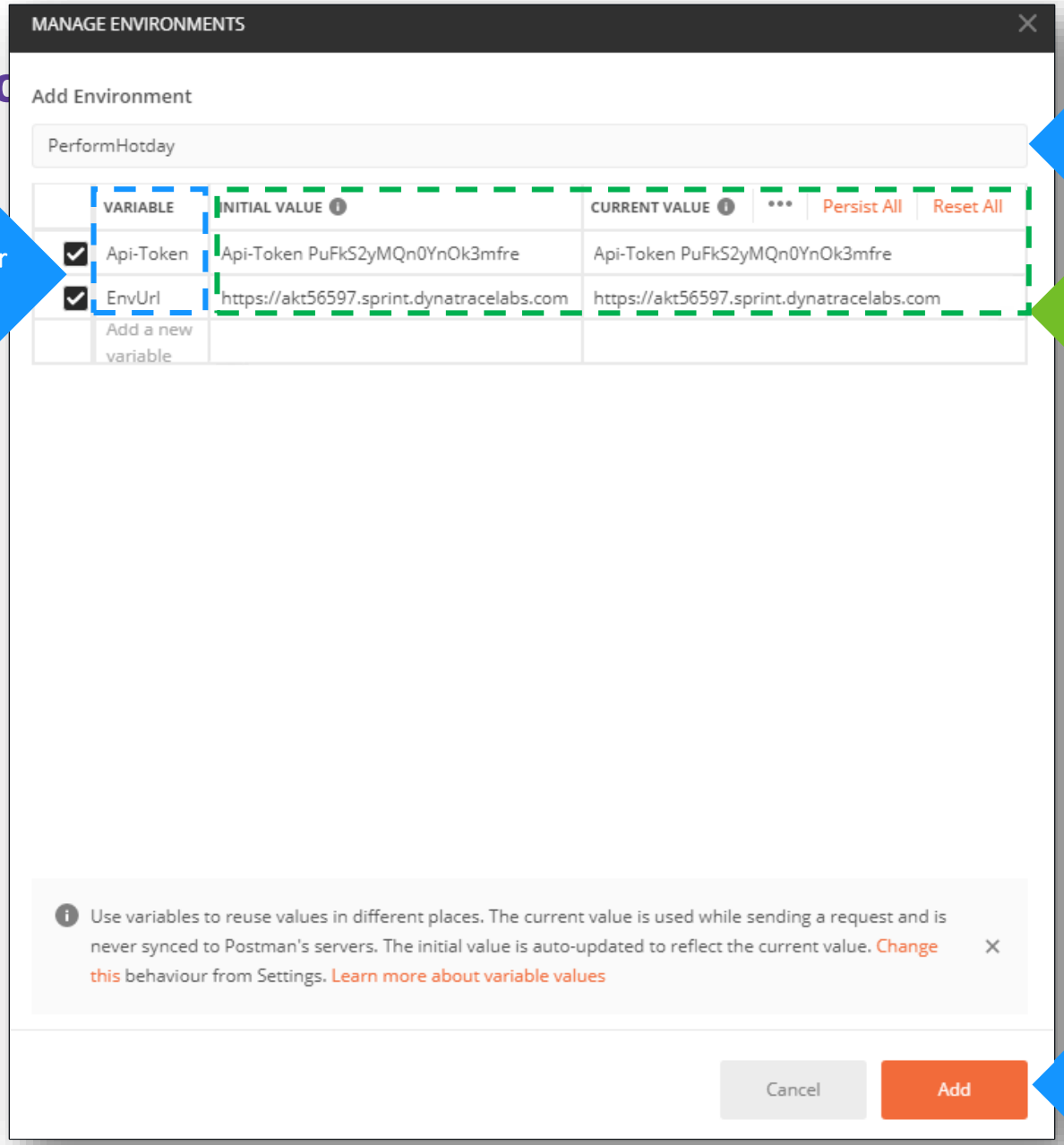
- If this is the first time you are using Postman, after you download and install, you will need to create a free account.
  - You can press "setup later"/"skip" on the Preferences and Team screens



Setup Postman (cc

- Create an Environm
- Click the icon

Create 2 Variables EXACTLY as they appear here

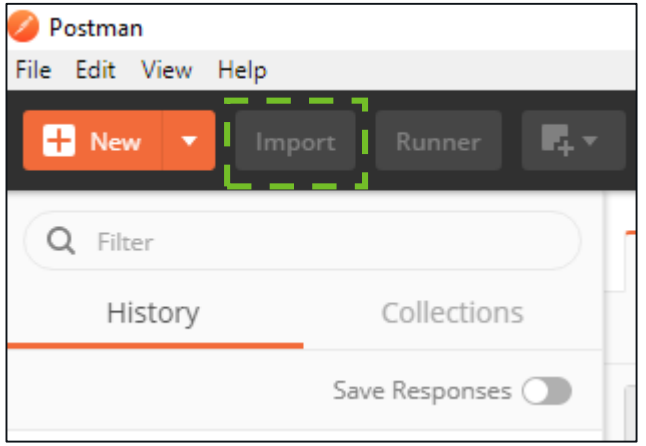


Enter a name for your Environment

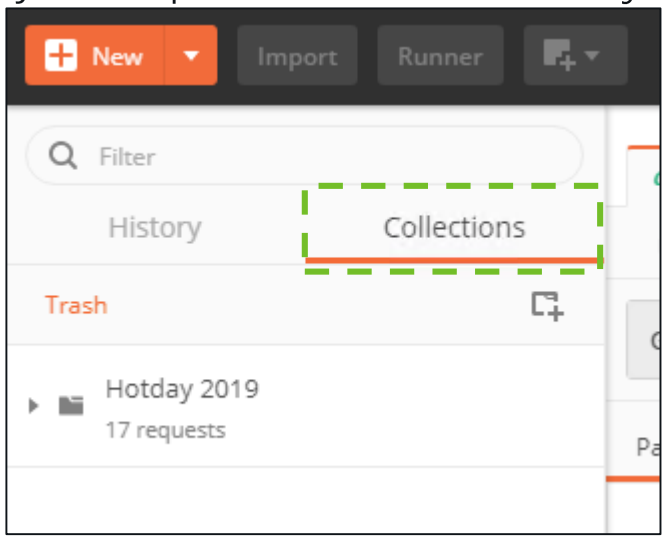
Populate the values with your Full Tenant URL and API-Token

Click “Add” to create the Variables

Importing the Config API Collection

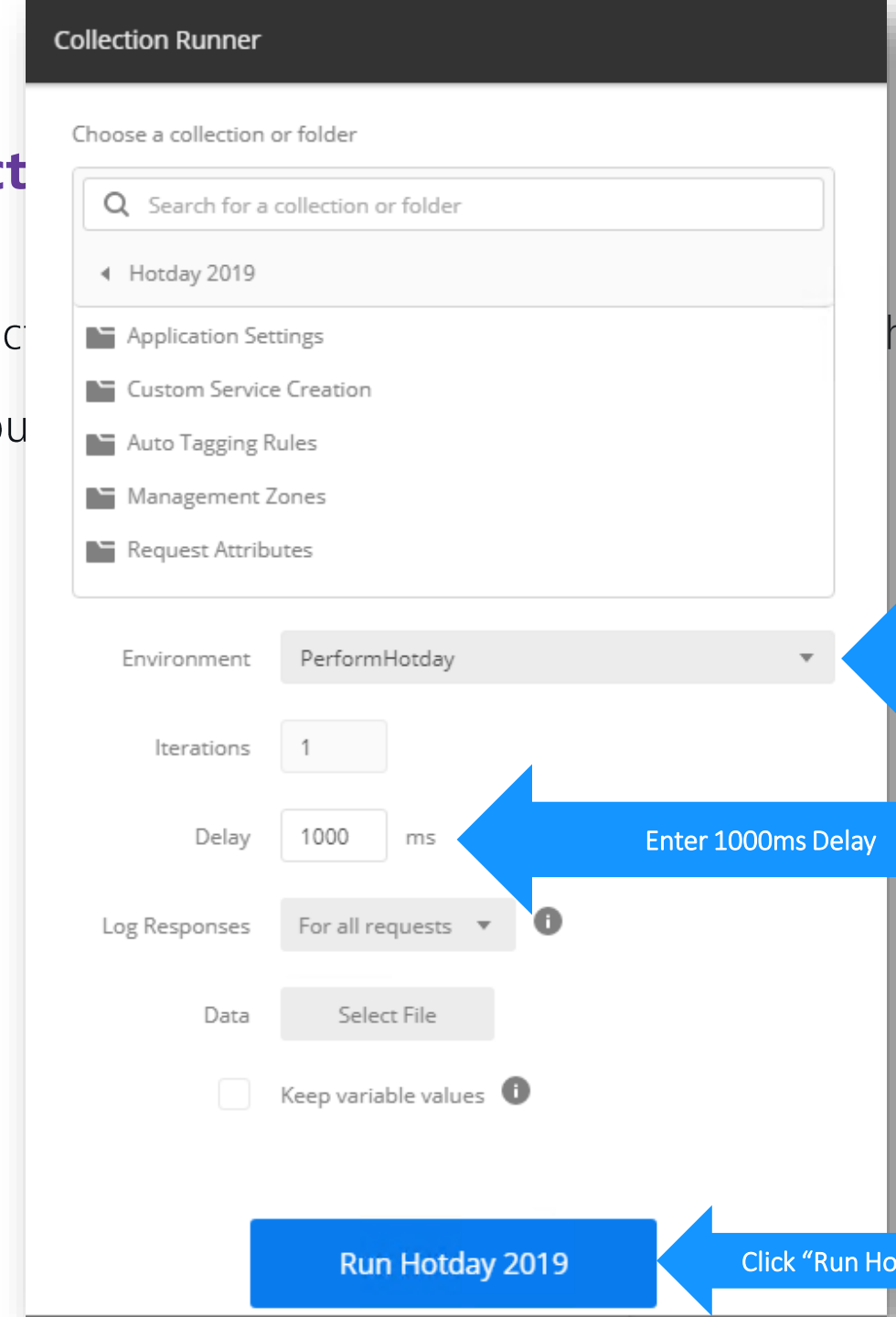


- Click the "Import" button in the upper left corner
- In the Import dialog box, click "Choose Files"
- Browse to the location where you downloaded the Github repository, and select the file: "Hotday 2019.postman_collection.json"
- After the file is uploaded, select "Collections" in the upper left corner and you should now see the collect that was just imported titled "Hotday 2019"



Running the API Collect

- In the "Hotday 2019" collec he popout menu
- The Collection Runner shou



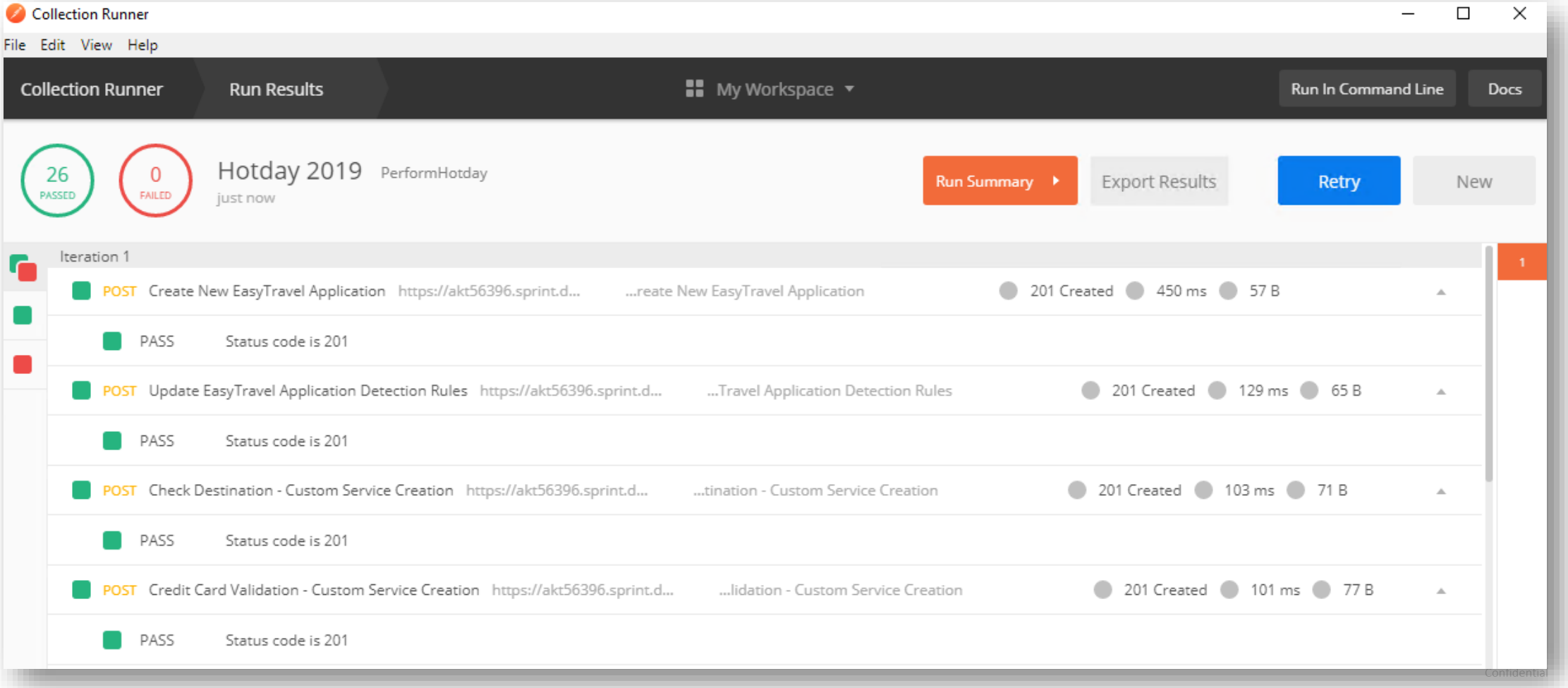
Select the Environment you created from the dropdown

Enter 1000ms Delay

Click "Run Hotday 2019"

Run Validation

- After running the collection, a Run Results window should popup with all requests successful



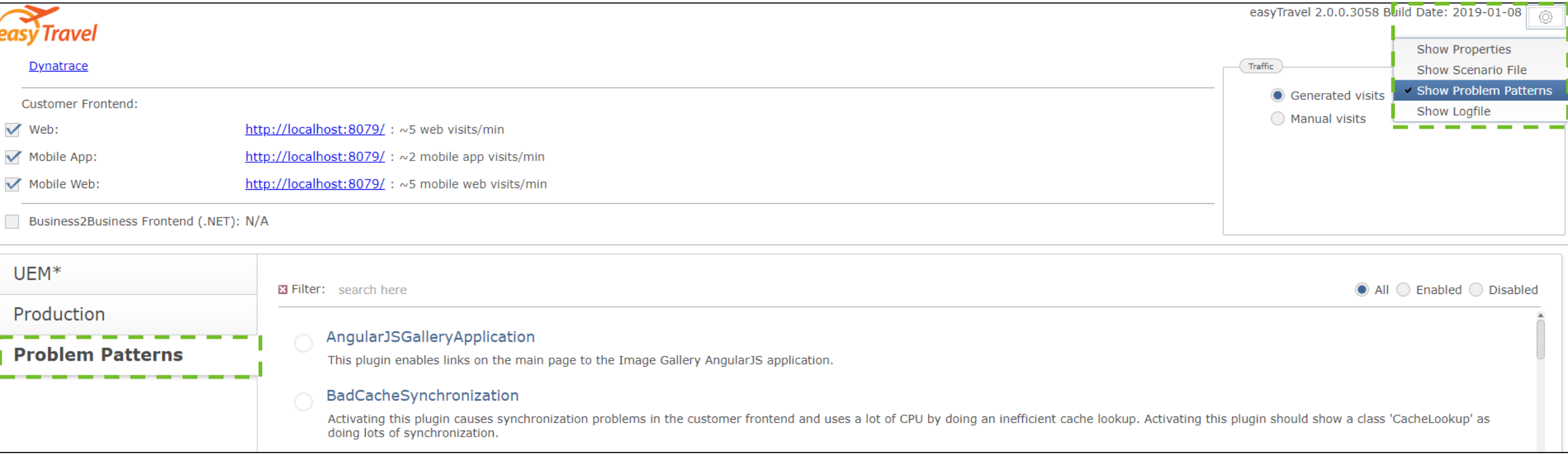
Verify the Results

- Back in your Dynatrace tenant, you should now see the results of the Config API (ex. Management Zones, Tags, etc…)

Use Case #1 – CPU Increase

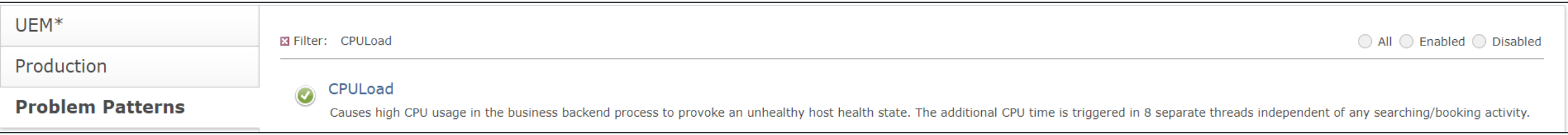
Problem Pattern #1 – CPU Increase

- How to trigger Problem:
  1. Navigate to the EasyTravelConfigUI – This URL was Output as part of the CloudFormation script
  2. Once in the EasyTravelConfigUI, enable "Show Problem Patterns"



Problem Pattern #1 – CPU Increase

- How to trigger Problem:
  1. In the Filter bar search for the Problem Pattern "CPULoad"
  2. Click the "CPULoad" Problem Pattern to trigger the problem



Use Case #1 – CPU Increase

Background

EasyTravel is the main revenue generating application for booking travel. After a recent deployment to one of the backend components, your team notices that CPU usage is much higher than usual. This of course could impact users attempting to book travel on our site.

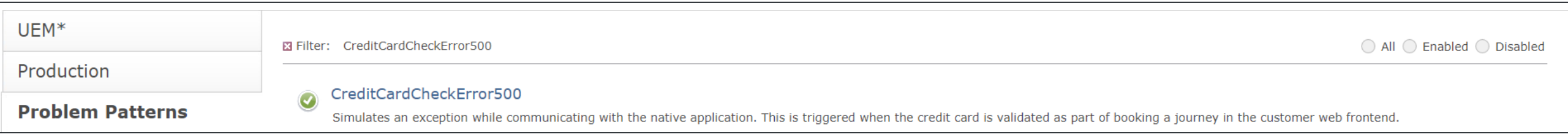
Quick Survey: https://www.surveymonkey.com/r/RJRTW7N

1. Investigate and figure out why this is happening
2. How would you instruct the Dev team to go about finding the problem and triaging it?

Use Case #2 – Booking Failure

Problem Pattern #2 – Booking Failure

- How to trigger Problem:
  1. In the Filter bar search for the Problem Pattern "CreditCardCheckError500"
  2. Click the "CreditCardCheckError500" Problem Pattern to trigger the problem



Use Case #2 – Booking Failure

Background

The Marketing Department just called in a panic that bookings dropped to near zero over the past few minutes! Obviously, this is a pretty big issue because the core business is being impacted.

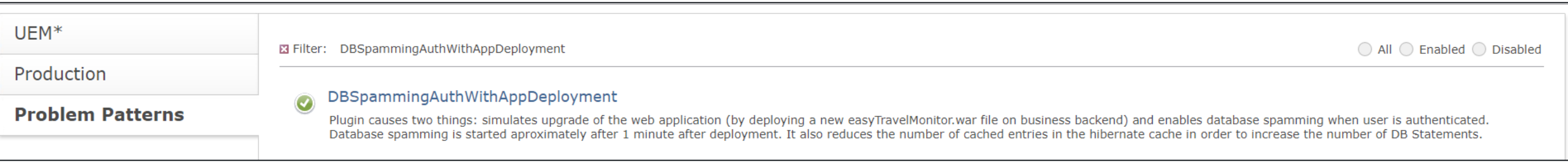
Quick Survey: https://www.surveymonkey.com/r/XYBTDWR

1. Investigate and figure out why this is happening
2. How would you instruct the Dev team to go about finding the problem and triaging it?

Use Case #3 – Slow User Login

Problem Pattern #3 – Slow User Login

- How to trigger Problem:
  1. In the Filter bar search for the Problem Pattern "DBSpammingAuthWithAppDeployment"
  2. Click the "DBSpammingAuthWithAppDeployment" Problem Pattern to trigger the problem



Use Case #3 – Slow User Login

Background

After a recent deployment to a component of EasyTravel customers are beginning to call the helpdesk and complain that logging in to view/book travel is “slow”.

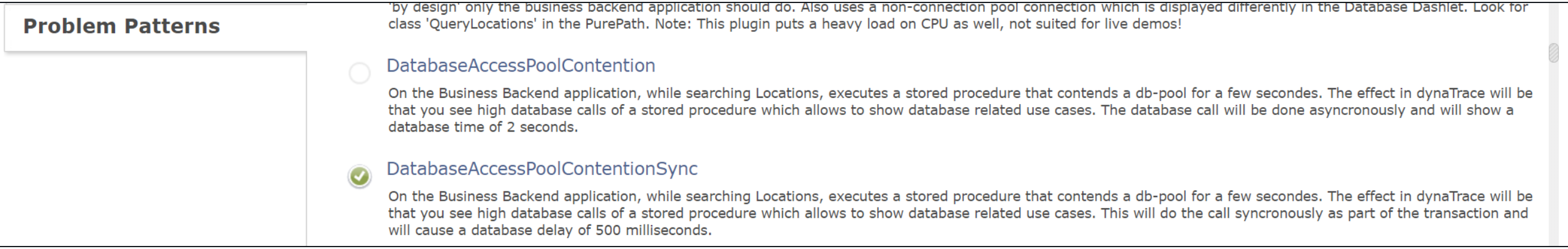
Quick Survey: https://www.surveymonkey.com/r/XDSSXSJ

1. Investigate and figure out why this is happening
2. How would you instruct the Dev team to go about finding the problem and triaging it?

Use Case #4 – Slow Loading Recommendations

Problem Pattern #4 – Slow Loading Recommendations

- How to trigger Problem:
  1. In the Filter bar search for the Problem Pattern "DatabaseAccessPoolContentionSync"
  2. Click the "DatabaseAccessPoolContentionSync" Problem Pattern to trigger the problem



Use Case #4 – Slow Loading Recommendations

Background

Your shop manager is calling you up and complaining that booking recommendations on the page are taking an unusually long time to show up.

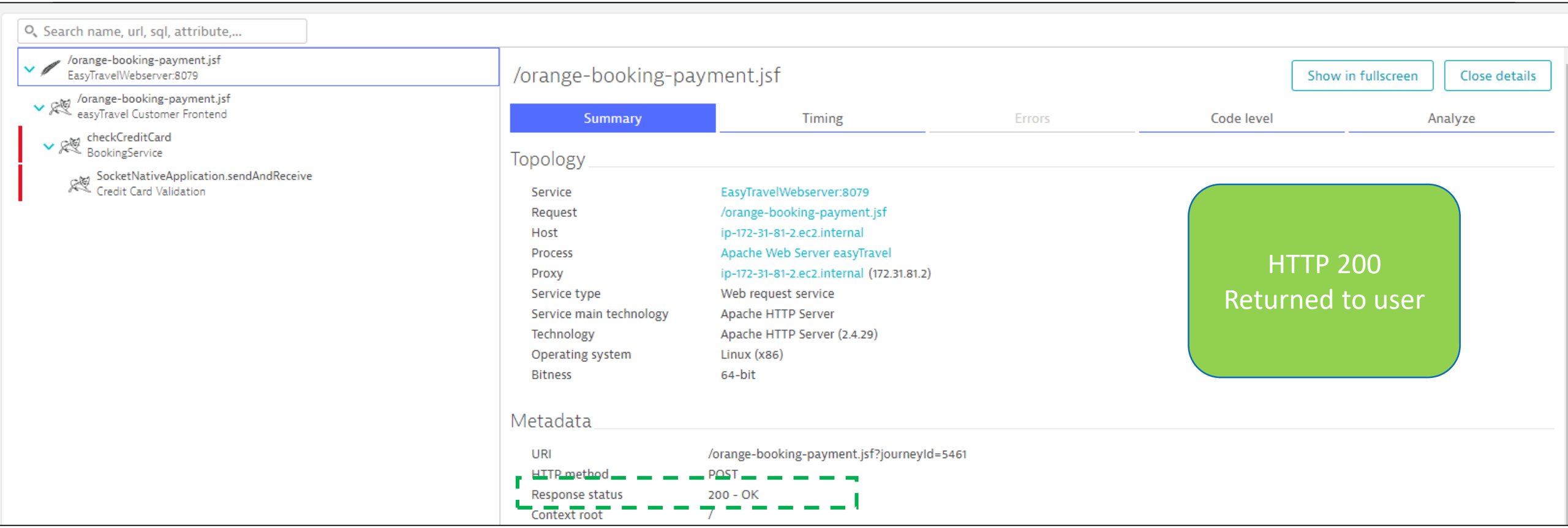
Quick Survey: https://www.surveymonkey.com/r/X9QGGK2

1. Investigate and figure out why this is happening
2. How would you instruct the Dev team to go about finding the problem and triaging it?

Tips & Tricks

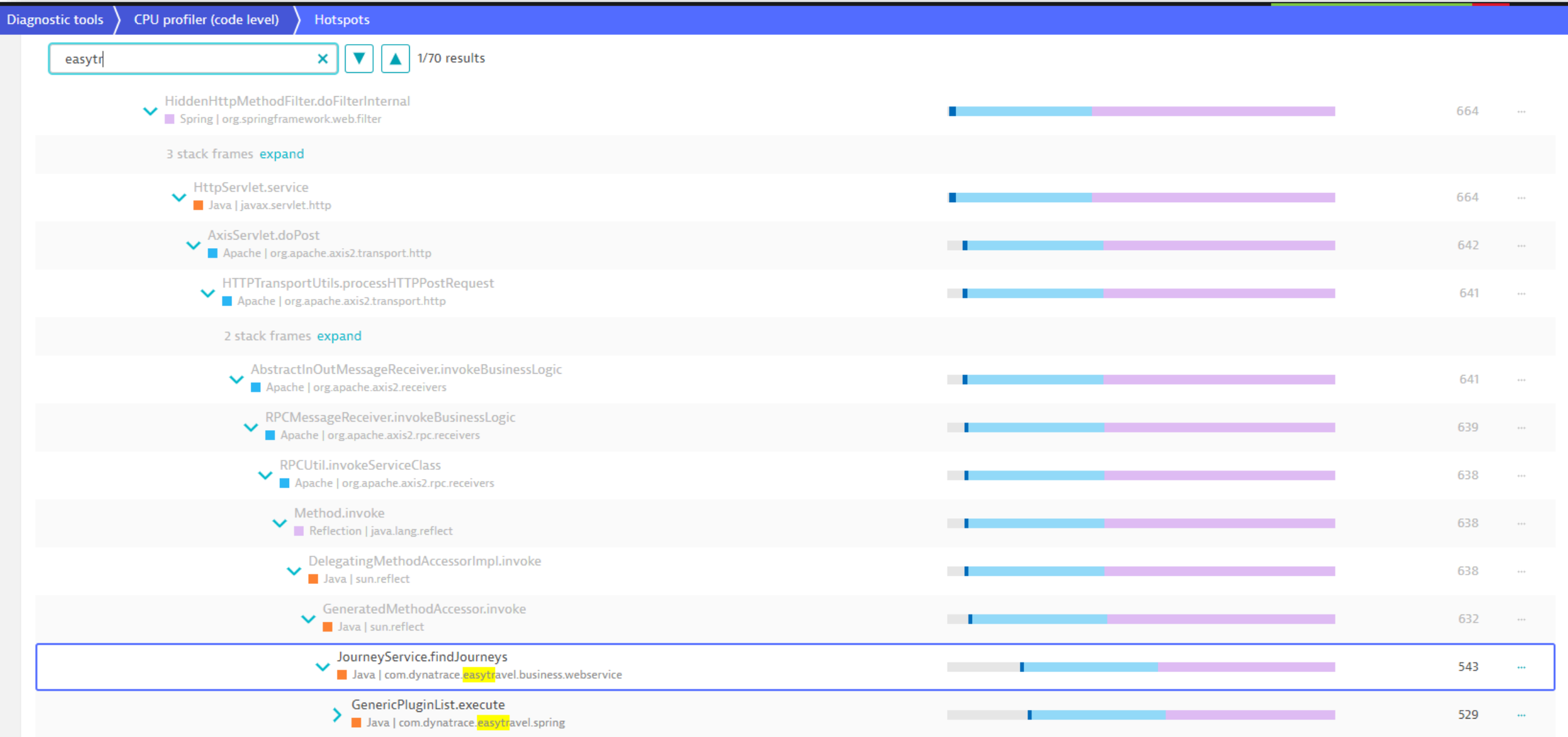
Returning HTTP 200 on Failure

- Try to avoid returning a HTTP 200 to the user, when the transaction actually fails:



dynatrace
Perform

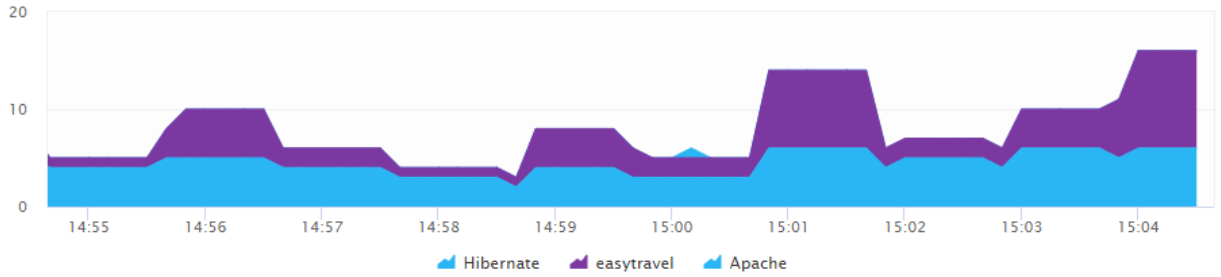
Finding your code in the CPU Profiler



Execution time breakdown

Apache	56 %
easytravel	44 %
Hibernate	< 1 %

Top APIs



Filtered by API: easytravel

Call tree

Hotspots

Top contributors

Method	Contribution	Stacktrace samples
JourneyUpdate.doExecute easytravel | com.dynatrace.easytravel.database		200
SocketInputStream.socketRead0 Java | java.net		77
CorrelationNative.registerCachedString Java | <agent>		1

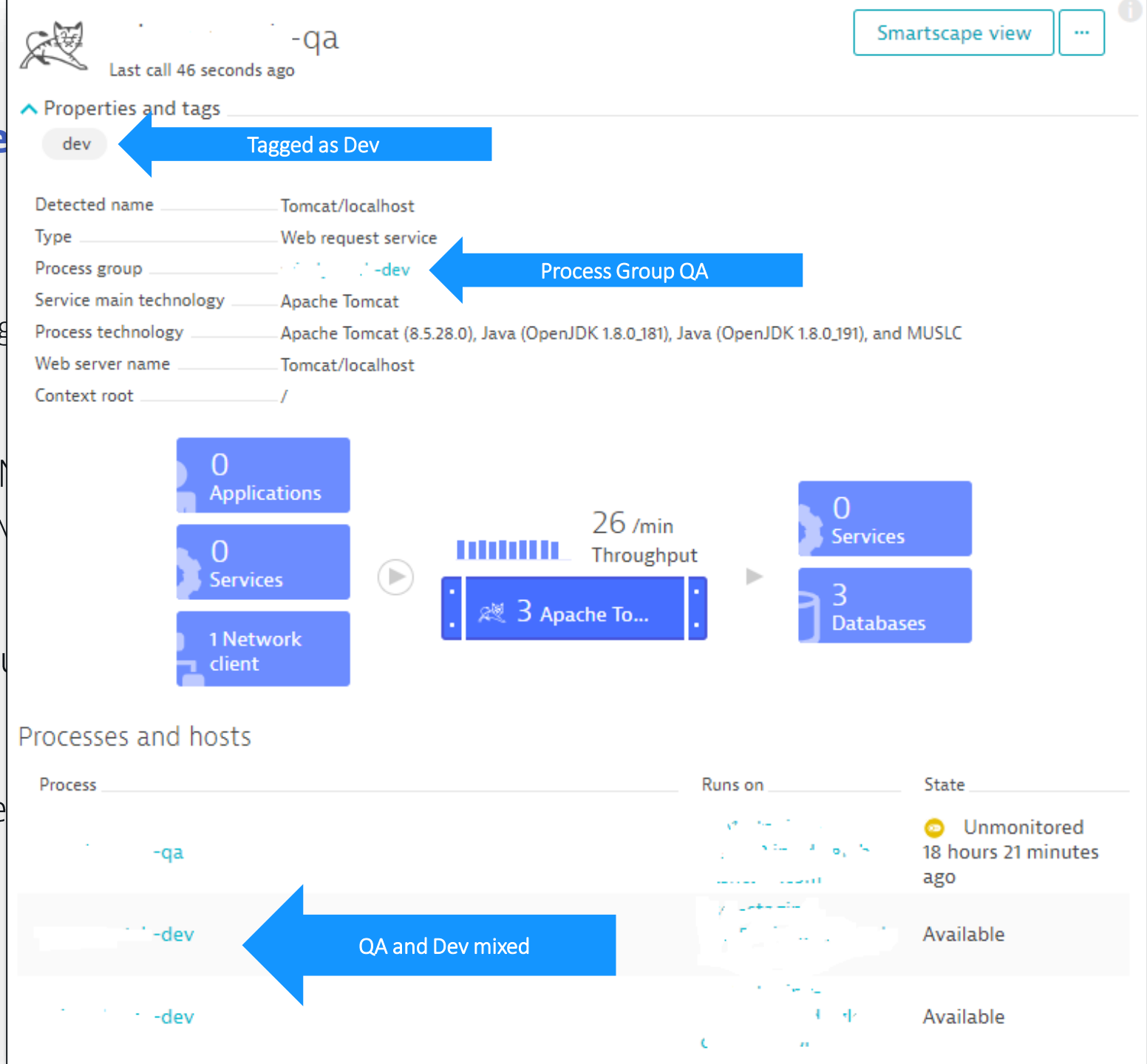
Calling methods of hotspot 'JourneyUpdate.doExecute'

Search

Method	Contribution	Stacktrace samples	Actions
JourneyUpdate.doExecute easytravel | com.dynatrace.easytravel.database		507	...
2 stack frames expand			
GenericPluginList.execute easytravel | com.dynatrace.easytravel.spring		507	...
JourneyService.findJourneys easytravel | com.dynatrace.easytravel.business.webservice		491	...

Properly Se

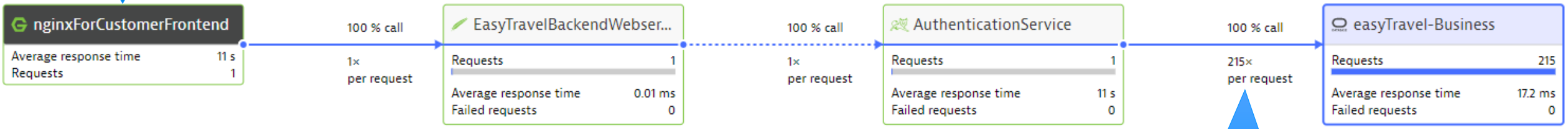
- Get Process
  - Including
- Services CA
  - Service M
- Process Grou
- Tagging rule



dynatrace Perform

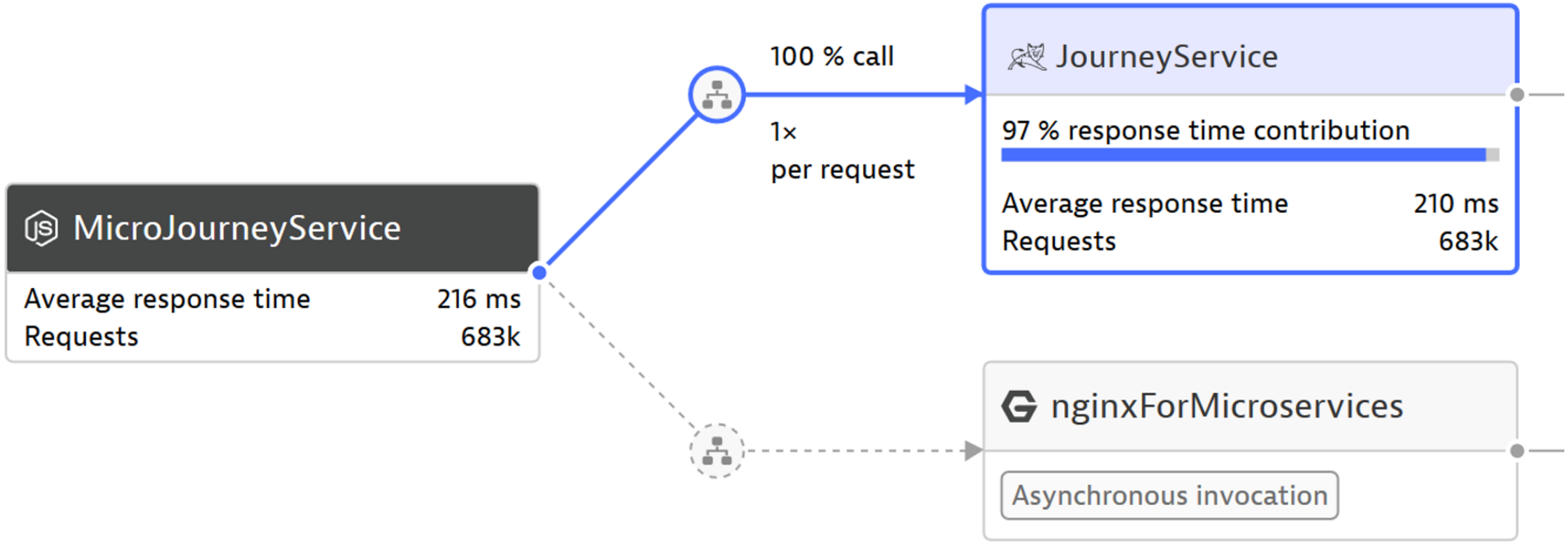
DB Spamming

Total Transaction took 11s, for a single request



Single request spawns 215 DB queries!

Microservices – Considerations for tightly coupled services



Don’t forget about the Logs

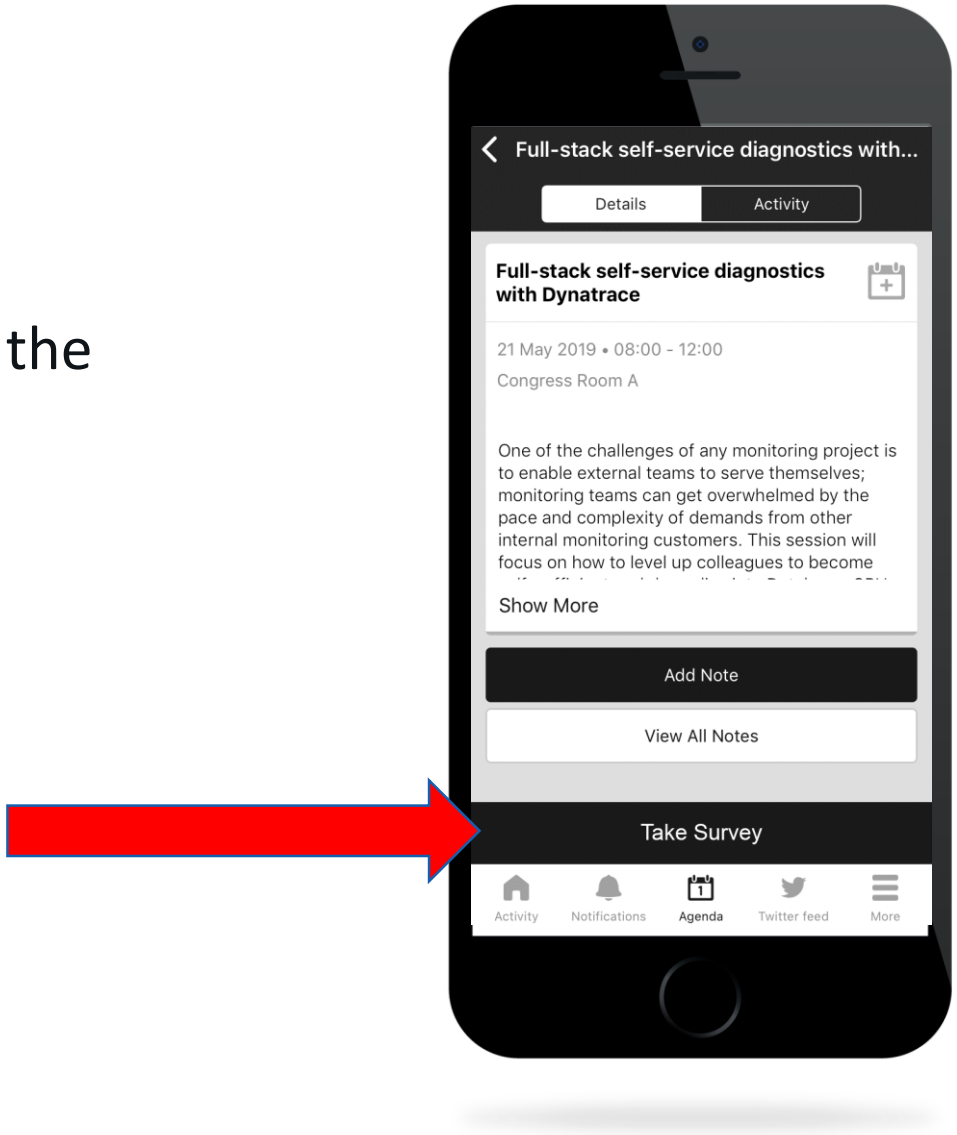
- Using Log Event Detection can help find application errors
  - Common Use Cases:
    - Detect Application Errors in Native Processes
    - Detect Errors with the Application Infrastructure (syslog, Windows Event Logs, etc…)
    - Detect Errors in CloudTrail logs

Remember to Delete your AWS Resources!!!

Q & A

Let us know how we did!

- 2 minute survey
- Find it in the Perform app under the HOT Day session you attended




Thank you
dynatrace
Perform