

# Full Stack Self-Service Diagnostics with Dynatrace

---

 **dynatrace**  
**Perform**

HOT Day  
sponsored by



**Wayne Segar**  
Senior Architect



**Matt Miller**  
Solution Architect



**Chris Schwarzbauer**  
Lead Product Architect

## Prerequisites

---

- Personal AWS Account:
  - <https://aws.amazon.com/free/>
- Postman API Client
  - <https://www.getpostman.com/downloads/>
- Download/Clone the Repository:
  - <https://github.com/wmsegar/Dynatrace-Hotday-2019-SelfServiceDiagnostics>
  - or
  - <https://bit.ly/2MkMkgI>

# 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 – Homepage Slowdown
- 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

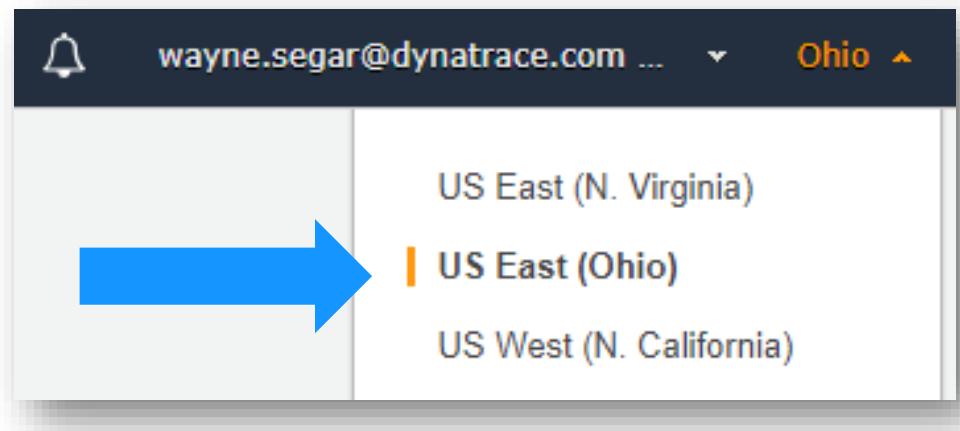
# Setup Work

---

## Setup AWS: Default Region

---

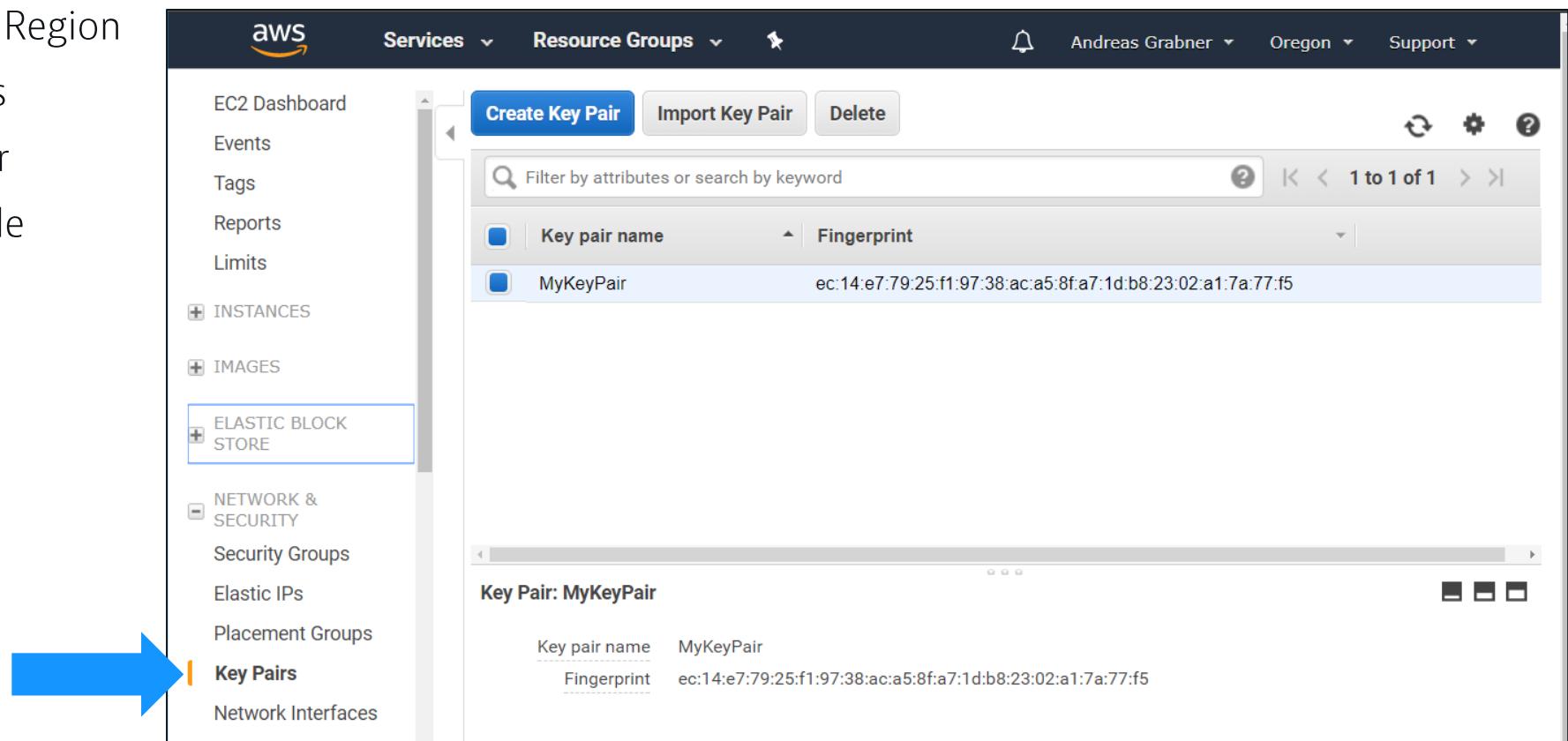
- Login to your AWS Account: <https://aws.amazon.com>
  - Validate and make note of your default region



## 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



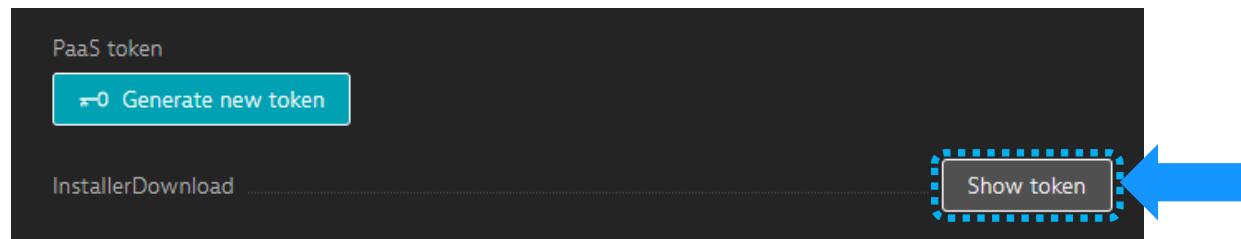
## Setup Dynatrace: Retrieve API Token

---

- Login to your Dynatrace SaaS Tenant provided to you:

[https://<TENANT\\_ID>.sprint.dynatracelabs.com](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

## Launching the Stack ◀

Upload Template

Filling in the Details

Create and Confirm

Follow Events

View Outputs

Validate Environment

Cleanup

# 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

**AWS services**

**Find services**  
You can enter names, keyword or acronyms.



The screenshot shows a search bar with the text "cloudform". Below it, a list of services is displayed, with "CloudFormation" being the first item. The "CloudFormation" entry includes the description "Create and Manage Resources with Templates" and a link to "Recently visited services".

Launching the Stack

Upload Template ◀

Filling in the Details

Create and Confirm

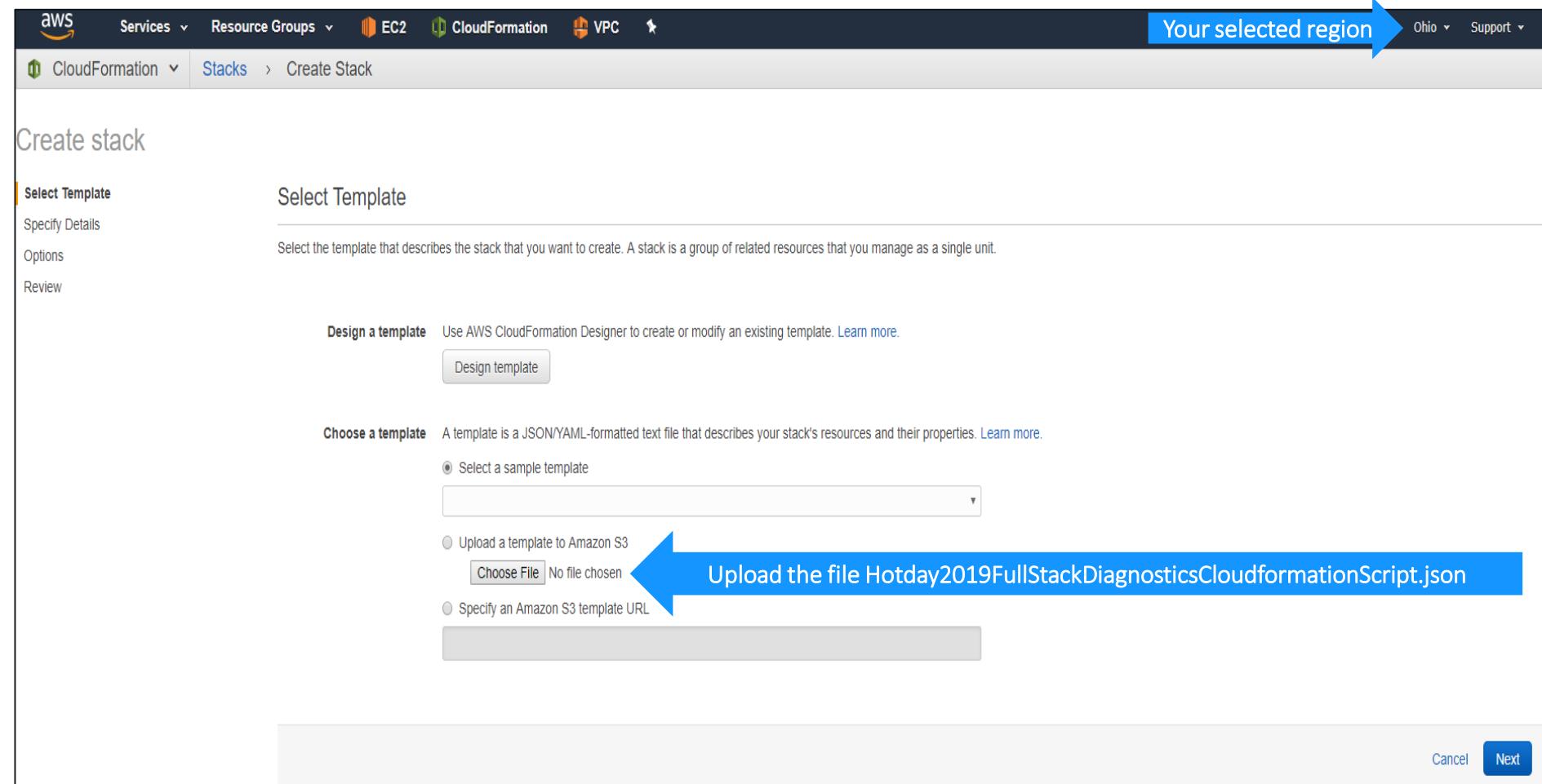
Follow Events

View Outputs

Validate Environment

Cleanup

## Upload Template



CloudFormation can be found in the GitHub repository

Launching the Stack

Upload Template

Filling in the Details ◀

Create and Confirm

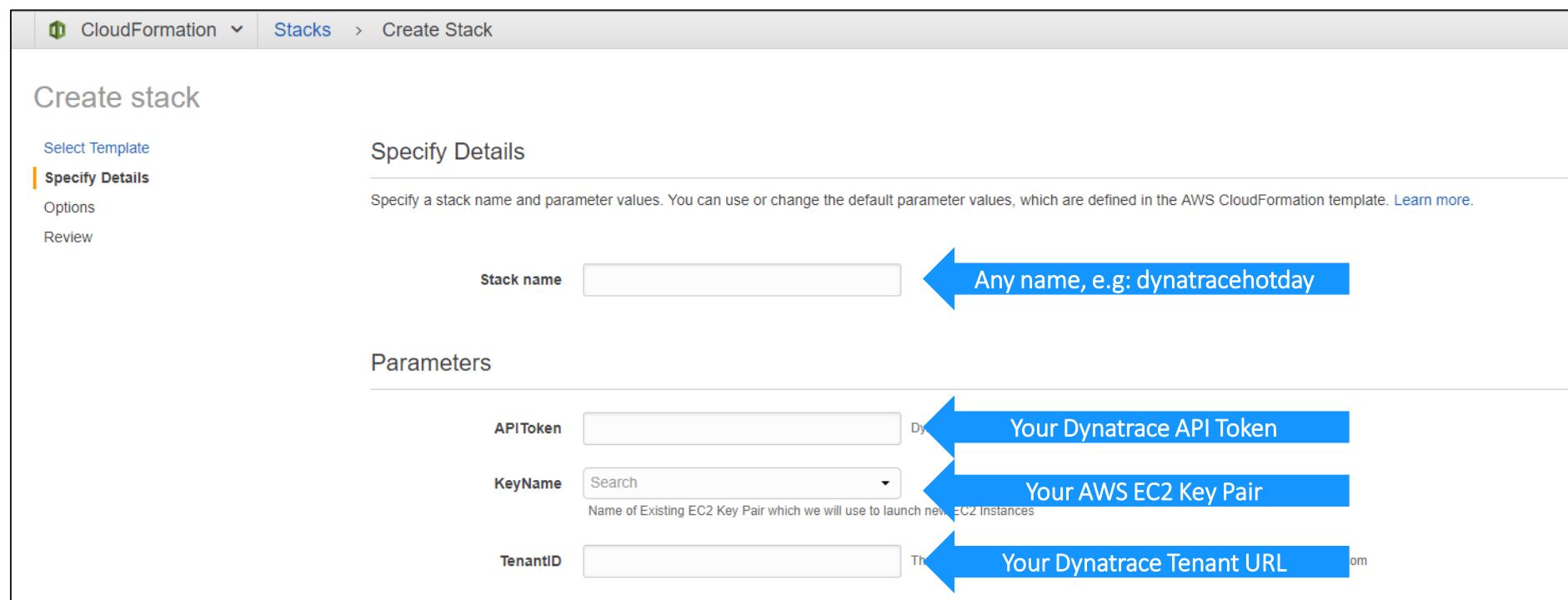
Follow Events

View Outputs

Validate Environment

Cleanup

## Filling in the Details



The screenshot shows the AWS CloudFormation 'Create stack' interface. In the top left, there's a sidebar with tabs: 'Select Template' (disabled), 'Specify Details' (selected), 'Options', and 'Review'. The main area is titled 'Specify Details' with the sub-instruction 'Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. [Learn more.](#)' Below this, there's a 'Stack name' input field with a blue arrow pointing to it labeled 'Any name, e.g: dynatracehotday'. Under 'Parameters', there are three fields: 'APIToken' (with a blue arrow labeled 'Your Dynatrace API Token'), 'KeyName' (with a blue arrow labeled 'Your AWS EC2 Key Pair'), and 'TenantID' (with a blue arrow labeled 'Your Dynatrace Tenant URL').

Launching the Stack

Upload Template

Filling in the Details

Create and Confirm ◀

Follow Events

View Outputs

Validate Environment

Cleanup

## Create and Confirm

### Review

#### Template

Template URL	<a href="https://s3.us-east-2.amazonaws.com/cf-templates-19yzke3lzb8f-us-east-2/2018365GfD-Wayne_Perform2019.json">https://s3.us-east-2.amazonaws.com/cf-templates-19yzke3lzb8f-us-east-2/2018365GfD-Wayne_Perform2019.json</a>
Description	Dynatrace Perform 2019: Full Stack Self-Service Diagnostics with Dynatrace
Estimate cost	Cost

#### Details

Stack name: test

APIToken asdfas  
KeyName WayneSegarKeyPair  
TenantID asdfasf

#### Options

##### Tags

No tags provided

##### Rollback Triggers

No monitoring time provided

No rollback triggers provided

##### Advanced

Notification  
Termination Protection Disabled  
Timeout none  
Rollback on failure Yes

[Quick Create Stack](#) (Create stacks similar to this one, with most details auto-populated)

[Cancel](#) [Previous](#) [Create](#)

Launching the Stack

Upload Template

Filling in the Details

Create and Confirm

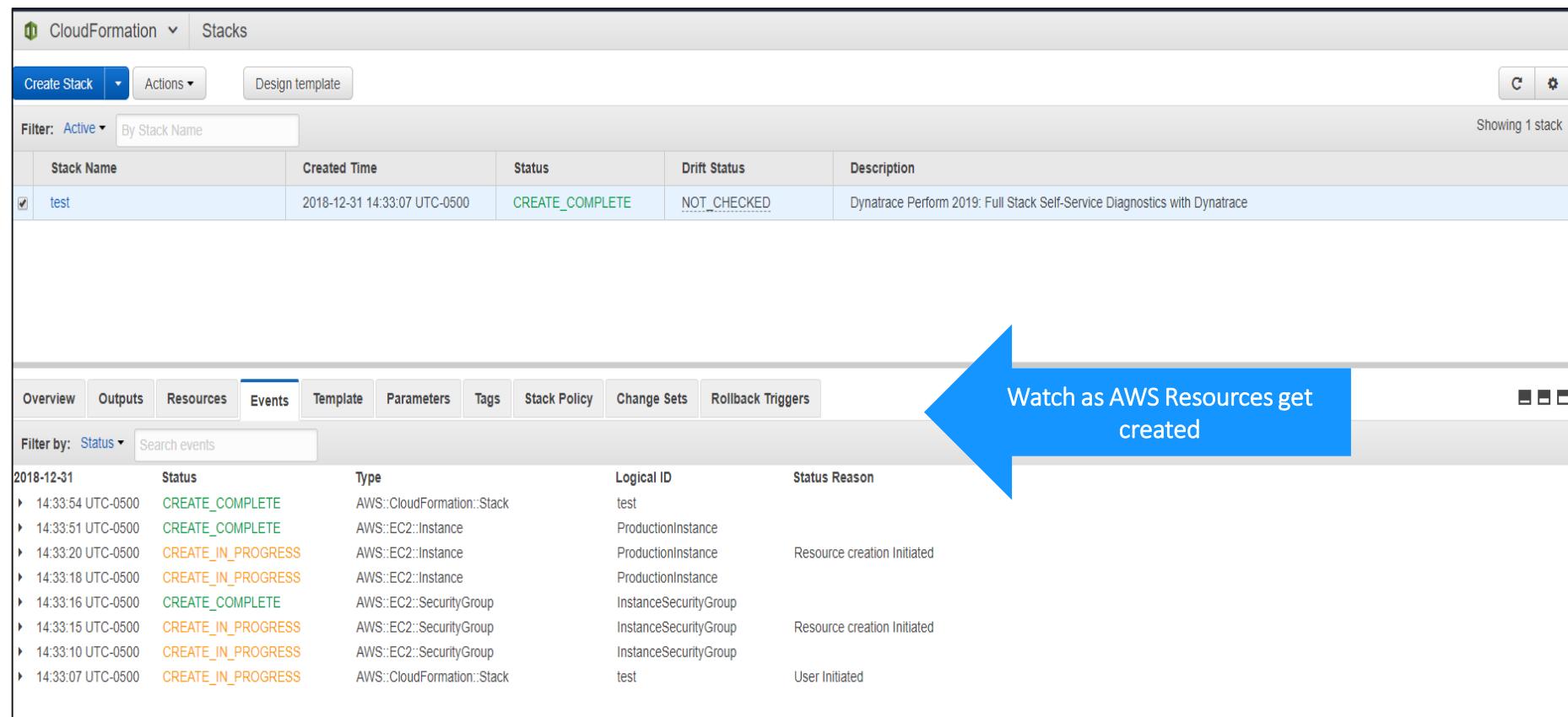
Follow Events ◀

View Outputs

Validate Environment

Cleanup

## Follow Events



The screenshot shows the AWS CloudFormation console with the 'Stacks' view open. A single stack named 'test' is listed, showing a 'Status' of 'CREATE\_COMPLETE'. Below this, the 'Events' tab is selected, displaying a detailed log of the resources being provisioned. The log includes:

Date	Status	Type	Logical ID	Status Reason
2018-12-31 14:33:54 UTC-0500	CREATE_COMPLETE	AWS::CloudFormation::Stack	test	
2018-12-31 14:33:51 UTC-0500	CREATE_COMPLETE	AWS::EC2::Instance	ProductionInstance	
2018-12-31 14:33:20 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::Instance	ProductionInstance	Resource creation initiated
2018-12-31 14:33:18 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::Instance	ProductionInstance	
2018-12-31 14:33:16 UTC-0500	CREATE_COMPLETE	AWS::EC2::SecurityGroup	InstanceSecurityGroup	
2018-12-31 14:33:15 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SecurityGroup	InstanceSecurityGroup	Resource creation initiated
2018-12-31 14:33:10 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::SecurityGroup	InstanceSecurityGroup	
2018-12-31 14:33:07 UTC-0500	CREATE_IN_PROGRESS	AWS::CloudFormation::Stack	test	User Initiated

Watch as AWS Resources get created

Launching the Stack

Upload Template

Filling in the Details

Create and Confirm

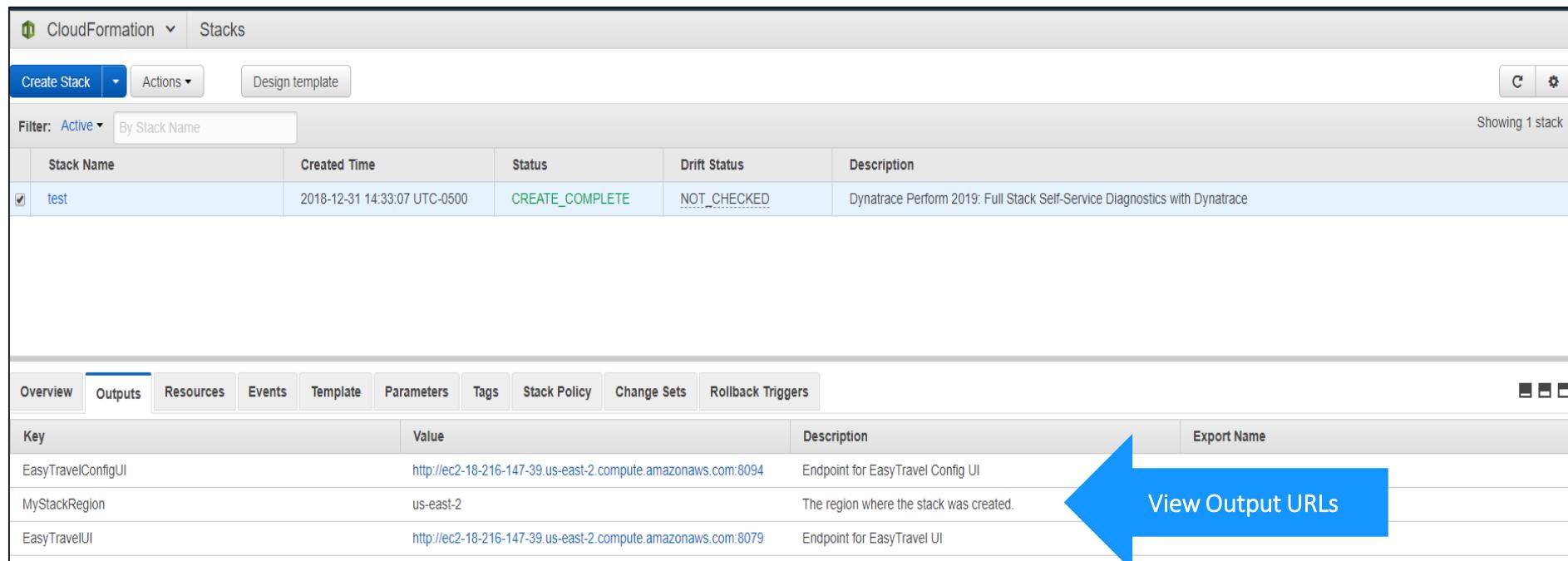
Follow Events

**View Outputs** ◀

Validate Environment

Cleanup

## View Outputs



The screenshot shows the AWS CloudFormation console with the 'Outputs' tab selected. It displays three outputs:

Key	Value	Description	Export Name
EasyTravelConfigUI	<a href="http://ec2-18-216-147-39.us-east-2.compute.amazonaws.com:8094">http://ec2-18-216-147-39.us-east-2.compute.amazonaws.com:8094</a>	Endpoint for EasyTravel Config UI	
MyStackRegion	us-east-2	The region where the stack was created.	
EasyTravelUI	<a href="http://ec2-18-216-147-39.us-east-2.compute.amazonaws.com:8079">http://ec2-18-216-147-39.us-east-2.compute.amazonaws.com:8079</a>	Endpoint for EasyTravel UI	

A blue arrow points from the text 'View Output URLs' to the 'Value' column of the EasyTravelConfigUI output row.

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

Launching the Stack

Upload Template

Filling in the Details

Create and Confirm

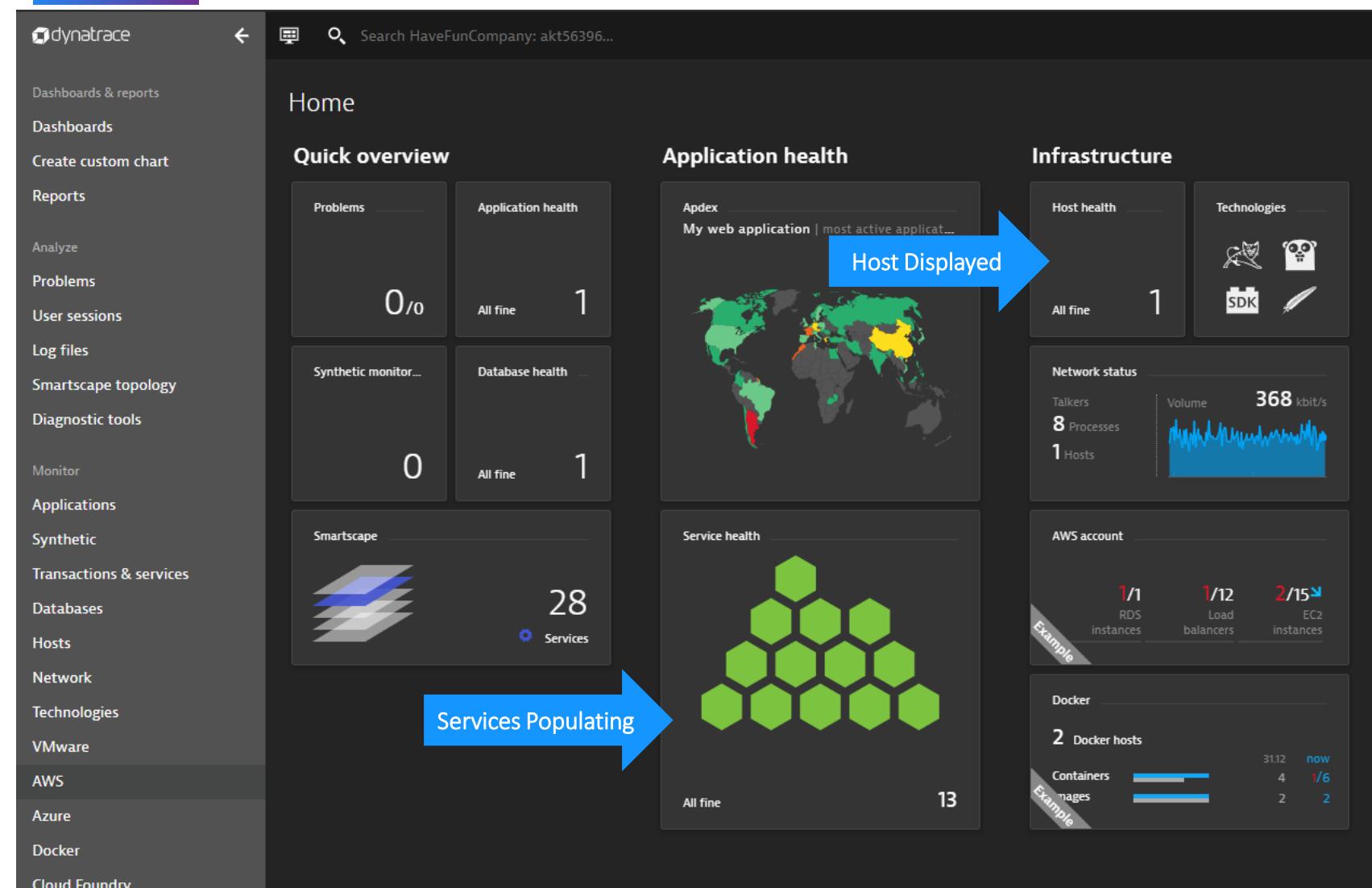
Follow Events

View Outputs

Validate Environment ◀

Cleanup

## 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>

technologies > select weblauncher.jar  
details > Edit > Disable for the host.

Automatically monitor processes

[Use defaults](#)

Set to Off

Automatically inject real user monitoring JavaScript tag



Enable to inject JavaScript tag into each HTML request processed by this process group.

Enable process monitoring for specific hosts

Enable/disable monitoring of this process group on individual hosts.

To manage your process monitoring settings please use [Process group monitoring](#)

Host



com.dynatrace.easytravel.weblauncher.jar easytravel-\*-\* on ip-172-31-43-194.ec2.internal

Monitoring: Off / On

Set to Off

[Use defaults](#)

## Launching the Stack

### Upload Template

### Filling in the Details

### Create and Confirm

### Follow Events

### View Outputs

### Validate Environment

### Cleanup ◀

## 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

#### Custom service detection

Dynatrace automatically detects and monitors most server-side services in your environment with no configuration required. If your application doesn't rely on standard frameworks, you can set up custom services.

With a custom service you can instruct Dynatrace which method, class, or interface it should use to gain access to each of your application's custom server-side services.

Enable this setting to have changes to your custom services applied to Java and PHP processes in real-time with no required restart. [More...](#)

Enable real-time updates to Java and PHP services

Set to Enabled

Java services

.NET services

PHP services

Define Java 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"

### Dynatrace API

You can use our API to export Dynatrace monitoring data into your 3rd party reporting and analysis tools. Multiple API tokens can be created for different purposes.  
Use the [Dynatrace API Explorer](#) or [read the API documentation](#) for use-cases and examples.

## 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

### My Dynatrace API tokens

Generate a secure access API token that enables access to your Dynatrace monitoring data via our REST-based API.

MyToken

Use the switches below to define the access scope of your Dynatrace API token.

Access problem and event feed, metrics, topology and RUM JavaScript tag management

Access logs

Configure maintenance windows

Create and configure synthetic monitors

Read configuration

Write configuration

Change data privacy settings

Capture request data

User session query language

Anonymize user session data for data privacy reasons

Log import

**Generate** **Cancel**

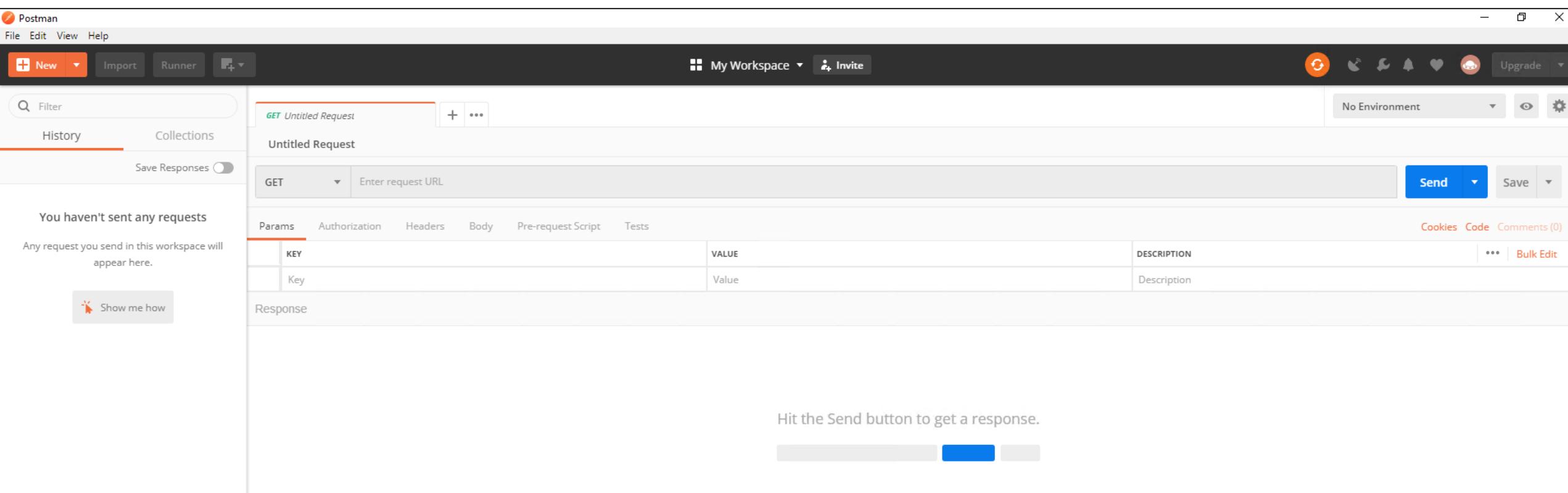
## Using the ConfigAPI via Postman

---

- 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
- 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



The screenshot shows the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', 'Help', 'New' (highlighted in orange), 'Import', 'Runner', and workspace-related buttons ('My Workspace', 'Invite'). The main workspace is titled 'Untitled Request' and shows a 'GET Untitled Request' entry. The 'Params' tab is selected, displaying a table with one row:

KEY	VALUE	DESCRIPTION
Key	Value	Description

Below the table, there's a 'Response' section with the placeholder text 'Hit the Send button to get a response.' and a large blue 'Send' button.

## Setup Postman (cont)

Create new Environment  
Create 2 Variables EXACTLY as they appear here  
• CLICK the  icon

MANAGE ENVIRONMENTS

Add Environment

PerformHotday

VARIABLE	INITIAL VALUE 	CURRENT VALUE 	...	Persist All	Reset All
<input checked="" type="checkbox"/> Api-Token	Api-Token PuFkS2yMQn0YnOk3mfre	Api-Token PuFkS2yMQn0YnOk3mfre			
<input checked="" type="checkbox"/> EnvUrl	https://akt56597.sprint.dynatracelabs.com	https://akt56597.sprint.dynatracelabs.com			
Add a new variable					

 Use variables to reuse values in different places. The current value is used while sending a request and is never synced to Postman's servers. The initial value is auto-updated to reflect the current value. [Change this behaviour from Settings](#). [Learn more about variable values](#)

Cancel  Add

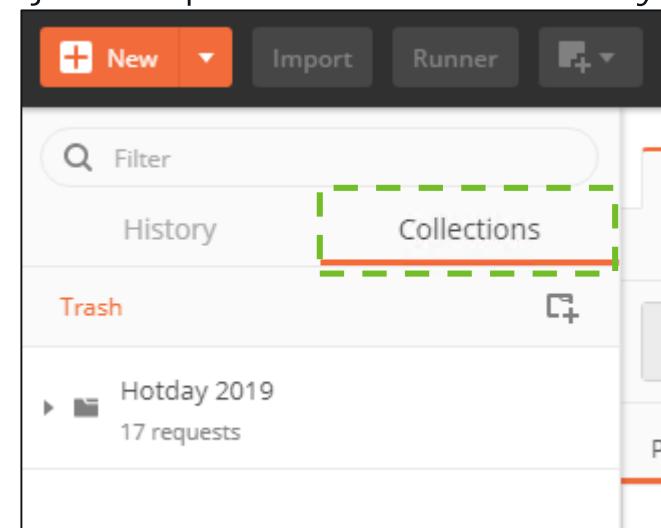
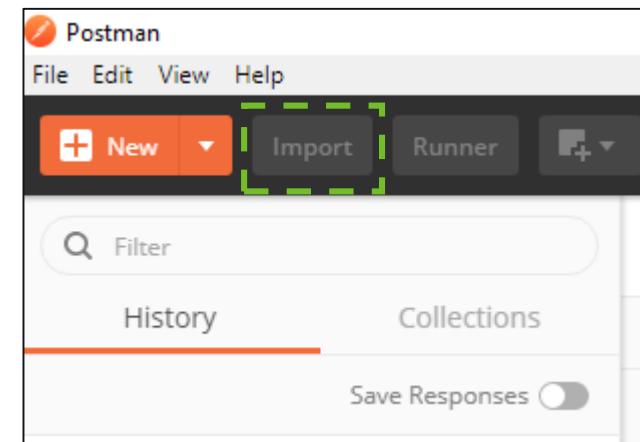
Enter a name for your Environment

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

Click "Add" to create the Variables

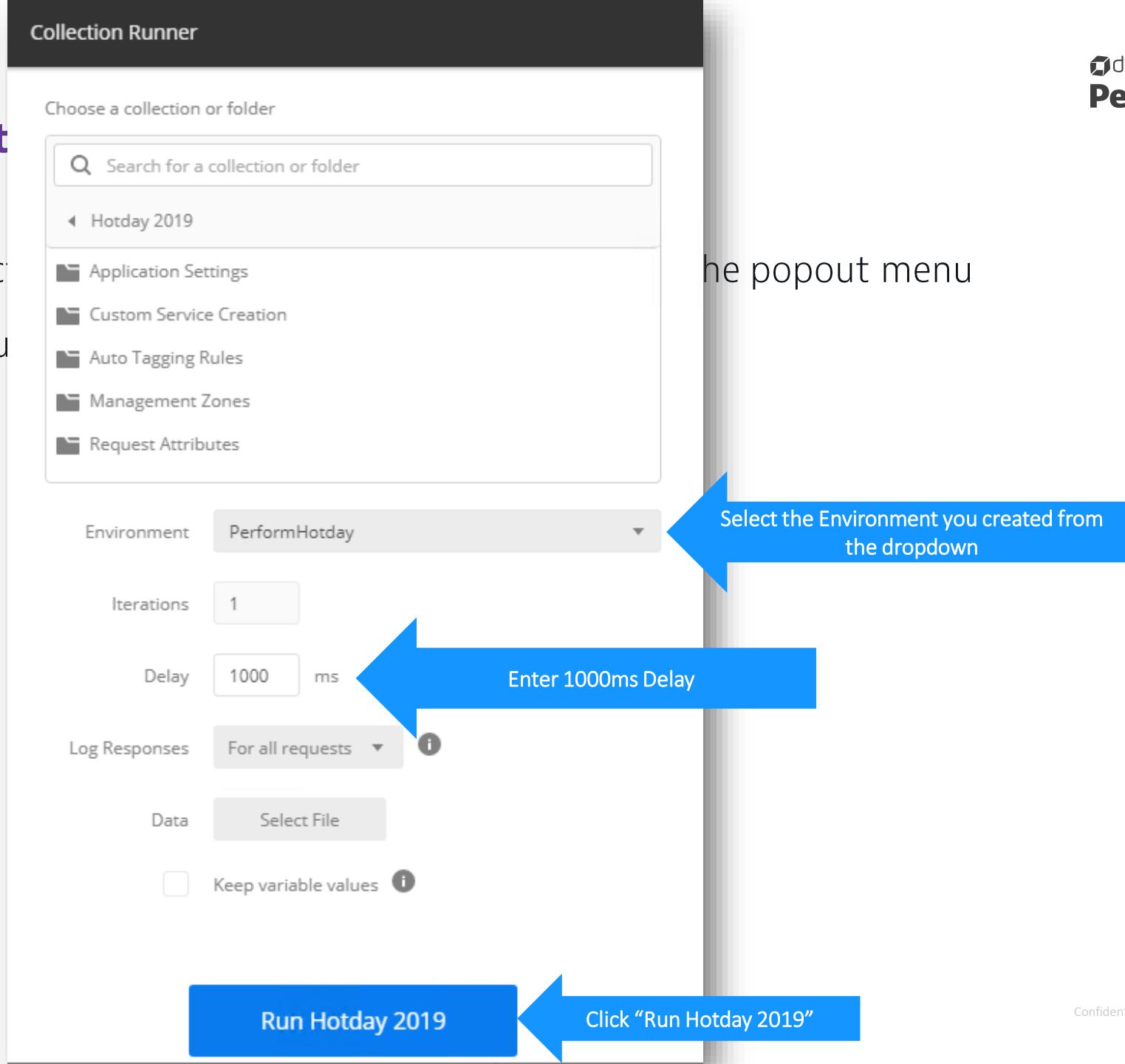
## Importing the Config API Collection

- 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”



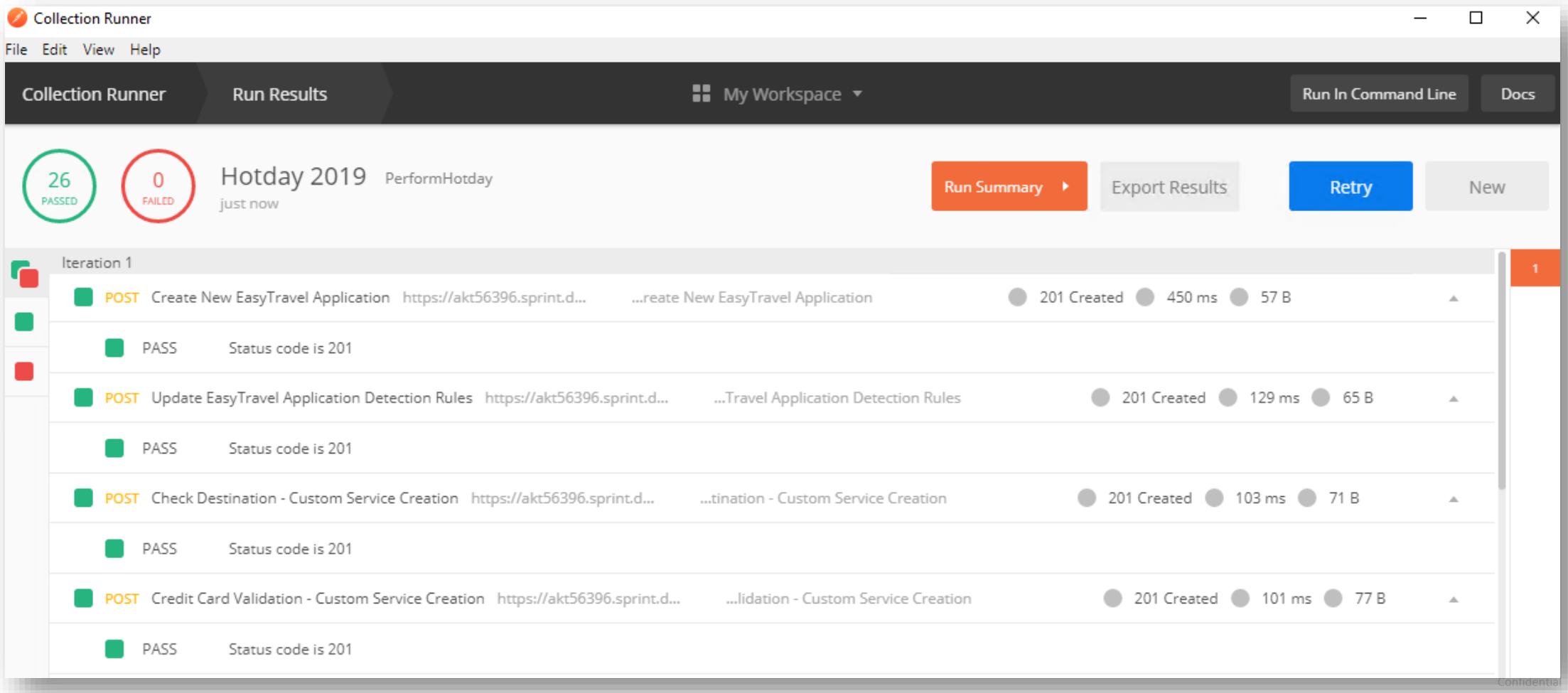
## Running the API Collector

- In the “Hotday 2019” collection
- The Collection Runner should



## Run Validation

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



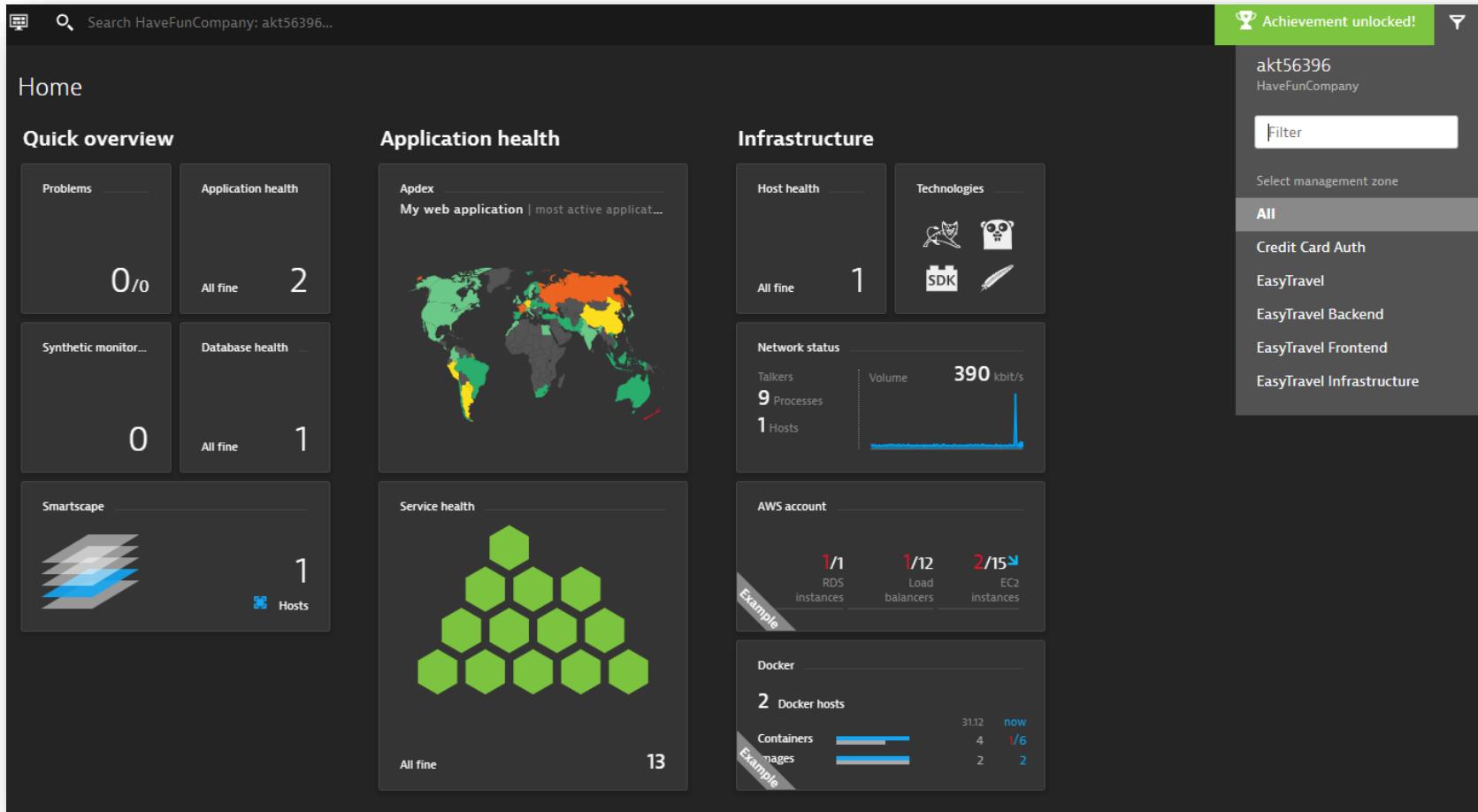
The screenshot shows the 'Collection Runner' application window. The title bar includes icons for Collection Runner, File, Edit, View, Help, and tabs for Collection Runner and Run Results. The main header has 'My Workspace' and navigation buttons for Run In Command Line and Docs. Below the header, a summary shows 26 PASSED and 0 FAILED requests. The run was named 'Hotday 2019' and occurred 'just now'. A large orange 'Run Summary' button is prominent. To the right are 'Export Results', 'Retry', and 'New' buttons. The main content area displays a table of test iterations:

Iteration	Request Type	Request URL	Description	Status	Response Code	Latency	Size
Iteration 1	POST	Create New EasyTravel Application https://akt56396.sprint.d...	...reate New EasyTravel Application	PASS	201 Created	450 ms	57 B
	POST	Update EasyTravel Application Detection Rules https://akt56396.sprint.d...	...Travel Application Detection Rules	PASS	201 Created	129 ms	65 B
	POST	Check Destination - Custom Service Creation https://akt56396.sprint.d...	...tination - Custom Service Creation	PASS	201 Created	103 ms	71 B
	POST	Credit Card Validation - Custom Service Creation https://akt56396.sprint.d...	...lidation - Custom Service Creation	PASS	201 Created	101 ms	77 B

## Verify the Results

---

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



The screenshot shows the Dynatrace Home page with the following details:

- Search Bar:** Search HaveFunCompany: akt56396...
- Achievement Unlocked:** akt56396 HaveFunCompany
- Quick overview:**
  - Problems: 0/0
  - Application health: All fine (2)
  - Synthetic monitors: 0
  - Database health: All fine (1)
  - Smartscape: 1 Hosts
- Application health:** My web application | most active applicat... (Shows a world map with green and yellow regions)
- Infrastructure:**
  - Host health: All fine (1)
  - Technologies: SDK, AWS Lambda
  - Network status: Talkers: 9 Processes, Volume: 390 kbit/s, 1 Hosts
  - AWS account: RDS instances: 1/1, Load balancers: 1/12, EC2 instances: 2/15
  - Docker: 2 Docker hosts, Containers: 31.12, Images: 4 / 6
- Filter:** Select management zone: All, Credit Card Auth, EasyTravel, EasyTravel Backend, EasyTravel Frontend, EasyTravel Infrastructure

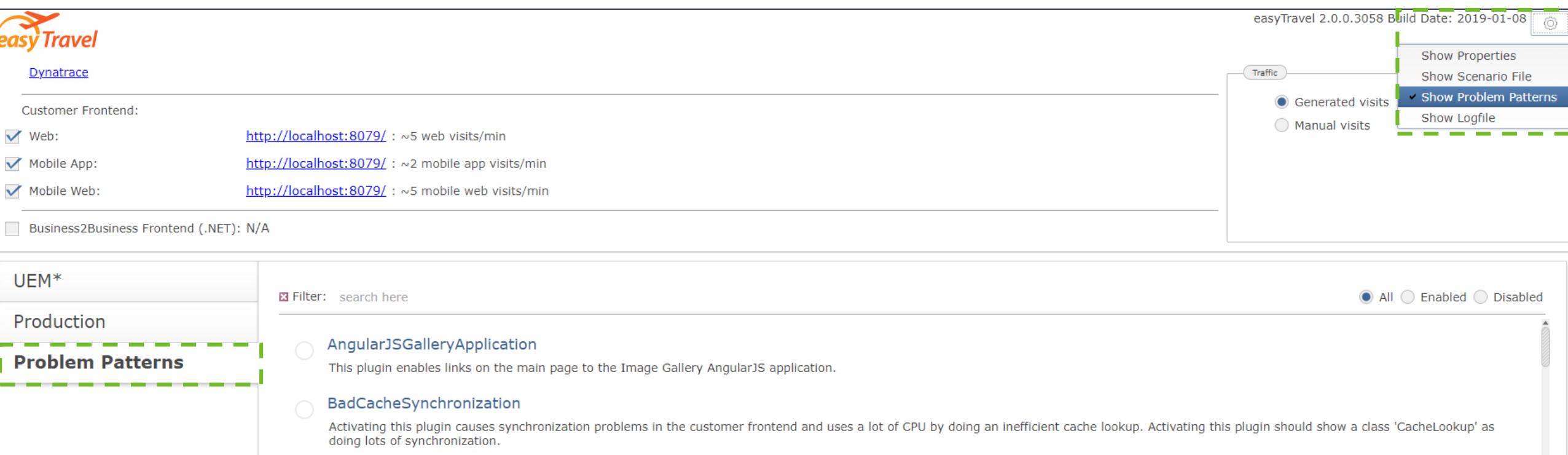
# Use Case #1 – CPU Increase

---

## 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”



The screenshot shows the Dynatrace interface for the 'easyTravel' application. On the left, there's a navigation bar with 'Customer Frontend' and 'Business2Business Frontend (.NET)' sections. The 'Customer Frontend' section lists 'Web' (localhost:8079), 'Mobile App' (localhost:8079), and 'Mobile Web' (localhost:8079) with visit counts of ~5, ~2, and ~5 respectively. The 'Business2Business Frontend (.NET)' section is marked as 'N/A'. On the right, there's a detailed view of the application version 'easyTravel 2.0.0.3058 Build Date: 2019-01-08'. A dropdown menu under 'Traffic' has 'Show Problem Patterns' selected. Below this, there are tabs for 'Generated visits' and 'Manual visits'. The main area displays two problem patterns:

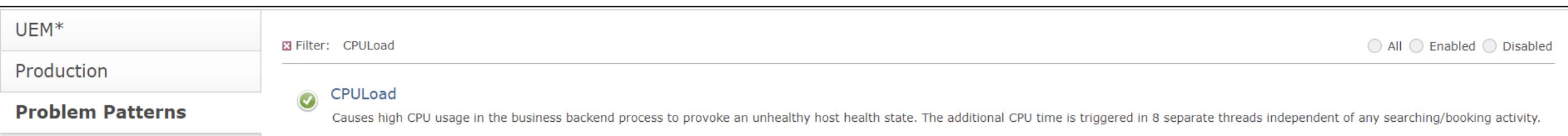
- AngularJSGalleryApplication**: Described as a plugin enabling links to the Image Gallery AngularJS application.
- BadCacheSynchronization**: Described as a plugin causing synchronization problems and high CPU usage due to inefficient cache lookups.

A filter bar at the top of the main content area allows searching for specific terms.

## Problem Pattern #1 – CPU Increase (cont.)

---

- 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



The screenshot shows the Dynatrace User Experience Management (UEM) interface. On the left, there's a navigation sidebar with tabs for 'UEM\*' (selected), 'Production' (disabled), and 'Problem Patterns' (selected). The main content area has a header with a filter bar showing 'Filter: CPULoad' and three status buttons: 'All' (disabled), 'Enabled' (disabled), and 'Disabled' (disabled). Below the filter bar, a list of 'Problem Patterns' is displayed, with 'CPULoad' being the first item. It has a green checkmark icon and a brief description: 'Causes high CPU usage in the business backend process to provoke an unhealthy host health state. The additional CPU time is triggered in 8 separate threads independent of any searching/booking activity.'

## 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/XJD6WLJ>

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 #1 – CPU Increase (cont.)

---

How to investigate the problem:

- Navigate to Diagnostic Tools and select the CPU Profiler (filter by Management Zone or Tag)
- View the Service with the Highest CPU consumption
- Navigate to the code level breakdown to determine what piece of code is causing increase in CPU consumption

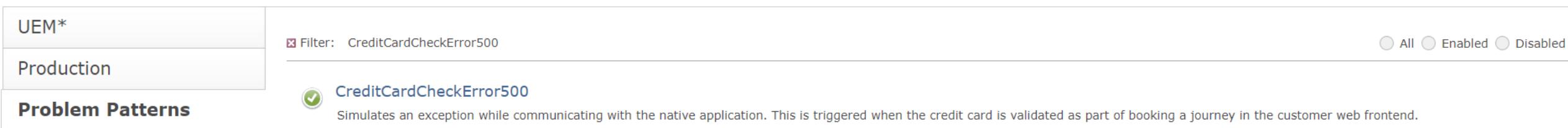
# Use Case #2 – Booking Failure

---

## 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



The screenshot shows the Dynatrace Perform interface. On the left, there's a sidebar with 'UEM\*' and 'Production' sections. The main area is titled 'Problem Patterns'. A filter bar at the top has 'Filter: CreditCardCheckError500' and buttons for 'All', 'Enabled', and 'Disabled'. Below the filter, 'CreditCardCheckError500' is listed with a green checkmark icon. A tooltip below it reads: 'Simulates an exception while communicating with the native application. This is triggered when the credit card is validated as part of booking a journey in the customer web frontend.'

## 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/XQ9L6RB>

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 (cont.)

---

How to investigate the problem:

- Navigate to Diagnostic Tools and select the Exception Analysis (filter by Management Zone or Tag)
- Determine the exception message that is “most likely” the culprit
- Determine which Service is causing this Exception and what should be done

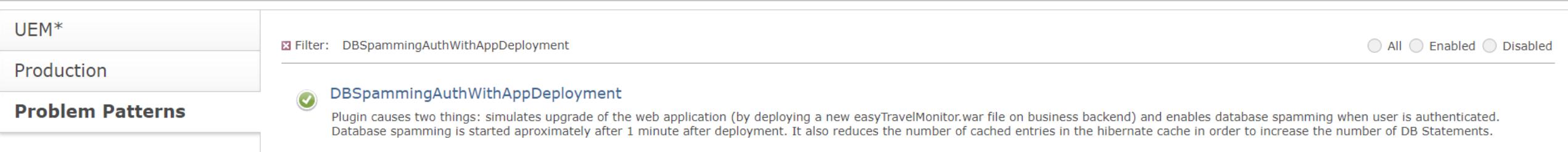
# Use Case #3 – Slow User Login

---

## 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



The screenshot shows the Dynatrace User Experience Management (UEM) interface. On the left, there's a sidebar with tabs for 'UEM\*' (selected), 'Production', and 'Problem Patterns'. The 'Problem Patterns' tab is currently active. In the main content area, there's a filter bar at the top with a checked checkbox labeled 'Filter: DBSpammingAuthWithAppDeployment'. To the right of the filter are three radio buttons labeled 'All', 'Enabled', and 'Disabled'. Below the filter, a list item is shown with a green checkmark icon and the text 'DBSpammingAuthWithAppDeployment'. A detailed description follows: 'Plugin causes two things: simulates upgrade of the web application (by deploying a new easyTravelMonitor.war file on business backend) and enables database spamming when user is authenticated. Database spamming is started approximately after 1 minute after deployment. It also reduces the number of cached entries in the hibernate cache in order to increase the number of DB Statements.'

## 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/XW2T5R9>

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 (cont.)

---

How to investigate the problem:

- Navigate to Diagnostic Tools and Select Top Database Statements
- Determine if there are any slow running queries are impacts from the recent change
- Determine what Service is responsible for this
- Is this a Database Slowdown issue?

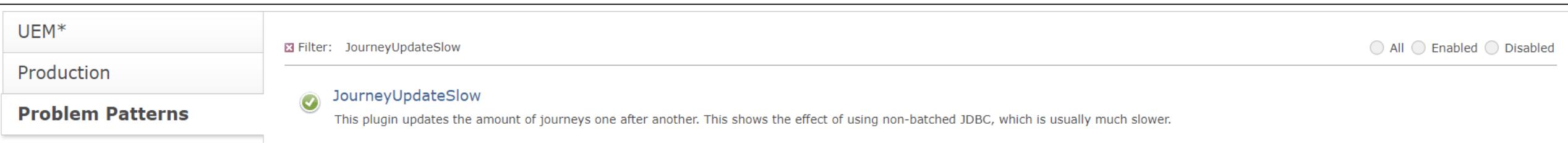
# Use Case #4 – Homepage Slowdown

---

## Problem Pattern #4 – Homepage Slowdown

---

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



The screenshot shows the Dynatrace interface for managing problem patterns. On the left, there's a sidebar with tabs for 'UEM\*', 'Production', and 'Problem Patterns'. The 'Problem Patterns' tab is currently active. At the top, there's a search bar labeled 'Filter:' with the text 'JourneyUpdateSlow' and a checked checkbox icon. To the right of the search bar are three radio buttons labeled 'All', 'Enabled', and 'Disabled'. Below the search bar, the list displays one item: 'JourneyUpdateSlow' with a checkmark icon. A tooltip below the list explains: 'This plugin updates the amount of journeys one after another. This shows the effect of using non-batched JDBC, which is usually much slower.'

## Use Case #4 – Homepage Slowdown

---

### Background

The Operations team is claiming that the 'Special-Offers.jsp' and 'CalculateRecommendations' pages are experiencing a significant increased response time. VP of Operations is not happy. Your job is to find out what's going on with these pages. Some other pages may also be affected.

Quick Survey: <https://www.surveymonkey.com/r/XSJVJ3P>

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 – Homepage Slowdown (cont.)

---

How to investigate the problem:

- In Diagnostic Tools, view the Top Web Requests and search for Special-Offers.jsp and CalculateRecommendations Web Requests
- View the Service Flow and look for the highest % of contribution
- Determine if this is a Database issue or an Application issue or both

# Tips & Tricks

---

## Returning HTTP 200 on Failure

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

The screenshot shows the Dynatrace Performance Monitor interface. On the left, there's a sidebar with a search bar and a tree view of services and requests. The main content area is titled '/orange-booking-payment.jsf' and contains several tabs: Summary (which is selected), Timing, Errors, Code level, and Analyze. Below the tabs, there's a 'Topology' section with detailed service metadata. A large green rounded rectangle highlights the 'HTTP method' and 'Response status' fields in the 'Metadata' section, both of which show 'POST' and '200 - OK' respectively. The 'Context root' field is also visible.

Search name, url, sql, attribute,...

/orange-booking-payment.jsf  
EasyTravelWebserver:8079

/orange-booking-payment.jsf  
easyTravel Customer Frontend

checkCreditCard  
BookingService

SocketNativeApplication.sendAndReceive  
Credit Card Validation

/orange-booking-payment.jsf

Show in fullscreen Close details

Summary Timing Errors Code level Analyze

Topology

Service	EasyTravelWebserver:8079
Request	/orange-booking-payment.jsf
Host	ip-172-31-81-2.ec2.internal
Process	Apache Web Server easyTravel
Proxy	ip-172-31-81-2.ec2.internal (172.31.81.2)
Service type	Web request service
Service main technology	Apache HTTP Server
Technology	Apache HTTP Server (2.4.29)
Operating system	Linux (x86)
Bitness	64-bit

HTTP 200  
Returned to user

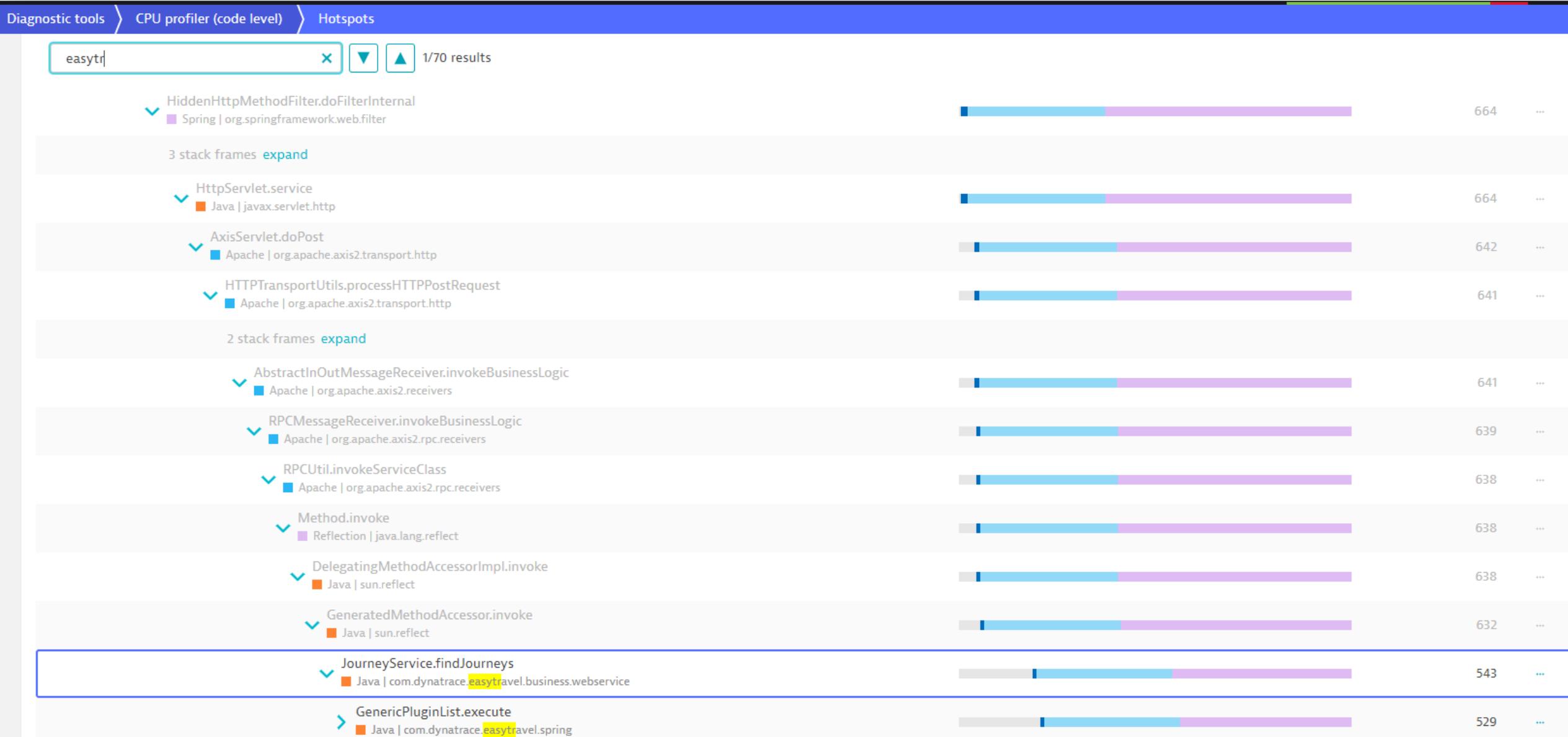
URI: /orange-booking-payment.jsf?journeyId=5461

HTTP method: POST

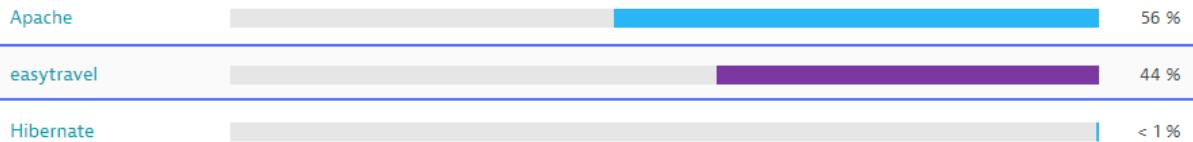
Response status: 200 - OK

Context root: /

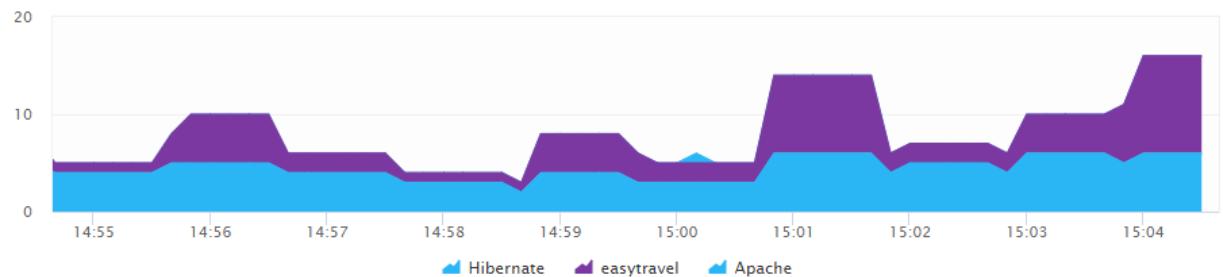
# Finding your code in the CPU Profiler



## Execution time breakdown



## Top APIs

Filtered by API: easytravel X

## Call tree

## Hotspots

## Top contributors

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

## Calling methods of hotspot 'JourneyUpdate.doExecute'

 🔍 ▼ ▲

Method	Contribution	Stacktrace samples	Actions
JourneyUpdate.doExecute easytravel   com.dynatrace.easytravel.database	507	507	...
<b>2 stack frames <span style="color: green;">expand</span></b>			
GenericPluginList.execute easytravel   com.dynatrace.easytravel.spring	507	507	...
JourneyService.findJourneys easytravel   com.dynatrace.easytravel.business.webservice	491	491	...

## Properly Set up Tags

- Get Process Groups
  - Including tags
- Services CAN be tagged
  - Service Name
- Process Groups
- Tagging rules

-qa

Last call 46 seconds ago

[Smartscape view](#) [...](#)

[Properties and tags](#)

[dev](#) **Tagged as Dev**

Detected name Tomcat/localhost  
 Type Web request service  
 Process group  -qa -dev **Process Group QA**  
 Service main technology Apache Tomcat  
 Process technology Apache Tomcat (8.5.28.0), Java (OpenJDK 1.8.0\_181), Java (OpenJDK 1.8.0\_191), and MUSLC  
 Web server name Tomcat/localhost  
 Context root /

 0 Applications  
 0 Services  
 1 Network client

26 /min Throughput

 3 Apache To...  3 Databases

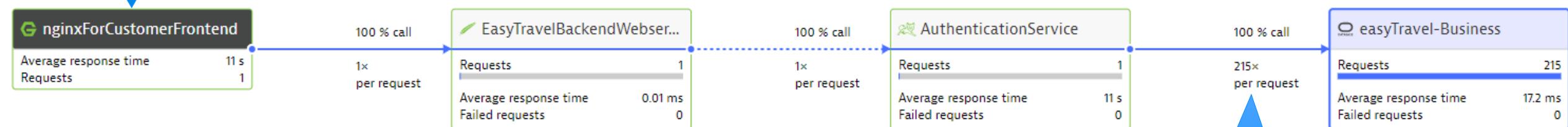
Processes and hosts

Process	Runs on	State
-qa	 Unmonitored 18 hours 21 minutes ago	 Available
-qa -dev	 Available	 Available
-qa -dev	 Available	 Available

**QA and Dev mixed**

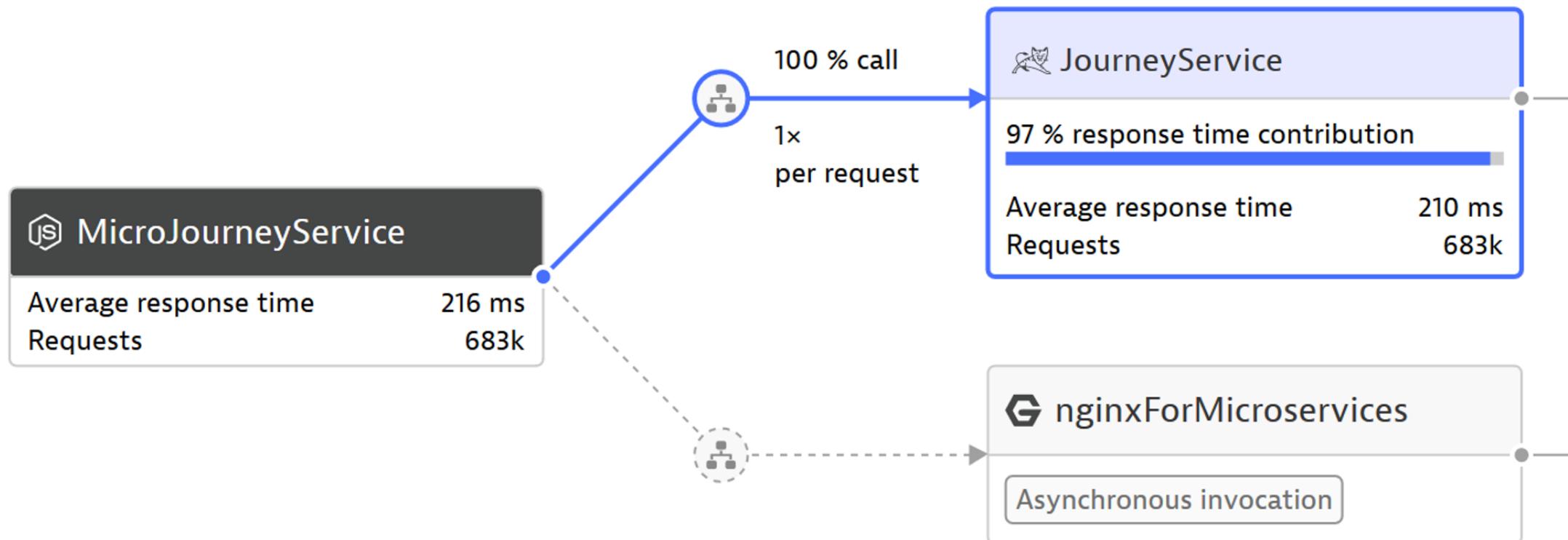
## DB Spamming

Total Transaction  
took 11s, for a  
single request



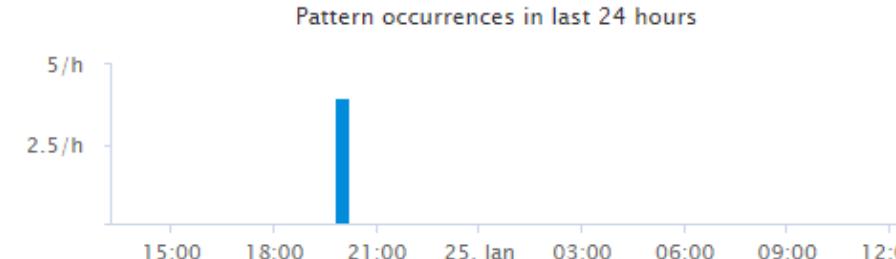
Single request  
spawns 215 DB  
queries!

## 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

Event name	Details
OS Process Error	<p>Text pattern</p> <p>OS Process Error</p> <p>Detection condition</p> <p>Generates Log Error Problem if number of occurrences is higher than 0/min</p> <p>Detection scope</p> <p>CouchDB_ET 1 log on 1 host selected</p>  <p>Pattern occurrences in last 24 hours</p> <p>5/h</p> <p>2.5/h</p> <p>15:00 18:00 21:00 25. Jan 03:00 06:00 09:00 12:00</p> <p>Delete rule Edit rule</p>

# Remember to Delete your AWS Resources!!!

---

# Q & A

---



Thank you



HOT Day  
sponsored by

