



DATADOG

Application Performance Monitoring (APM) Distributed Tracking with MuleSoft

January 28, 2022

Send Traces to Agent Via API

Join us at the Dash virtual conference! October 26-27

PRODUCT CUSTOMERS PRICING SOLUTIONS



ABOUT BLOG DOCS LOGIN

GET STARTED FREE

Datadog Docs

Search

APM & Continuous Profiler

- Send traces to Datadog
- Trace Search and Analytics
- Trace Retention and Ingestion
- Generate Metrics from Spans
- Deployment Tracking

Send traces to the Agent by API

Datadog APM allows you to collect performance metrics by tracing your code to determine which parts of your application are slow or inefficient.

Tracing data is sent from your instrumented code to the Datadog Agent through an HTTP API. Datadog tracing libraries simplify sending metrics to the Datadog Agent. However you might want to interact directly with the API to instrument applications that cannot use the

This feature allows you to manually create APM traces for technologies that have strong capabilities in calling APIs and making the appropriate payloads, e.g. MuleSoft or other Enterprise Service Bus technologies.

EDIT

The tracing API is an Agent API rather than a service side API. Submit your traces to the

`http://localhost:8126/v0.3/traces` local endpoint so your Agent can forward them to Datadog.

Continuous Profiler

APM Glossary

Error Tracking

Guides

Troubleshooting

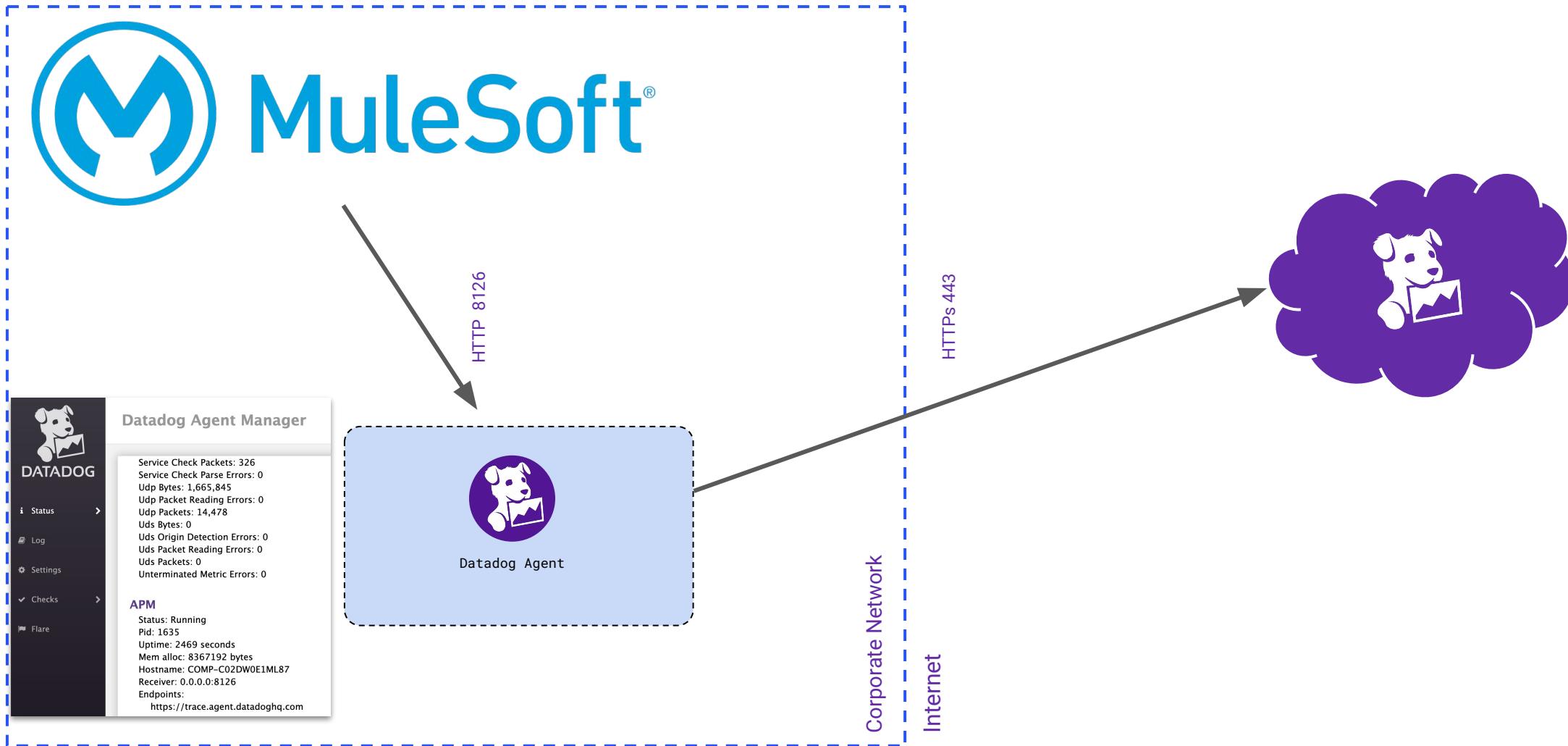
Path

`PUT http://localhost:8126/v0.3/traces`

Response

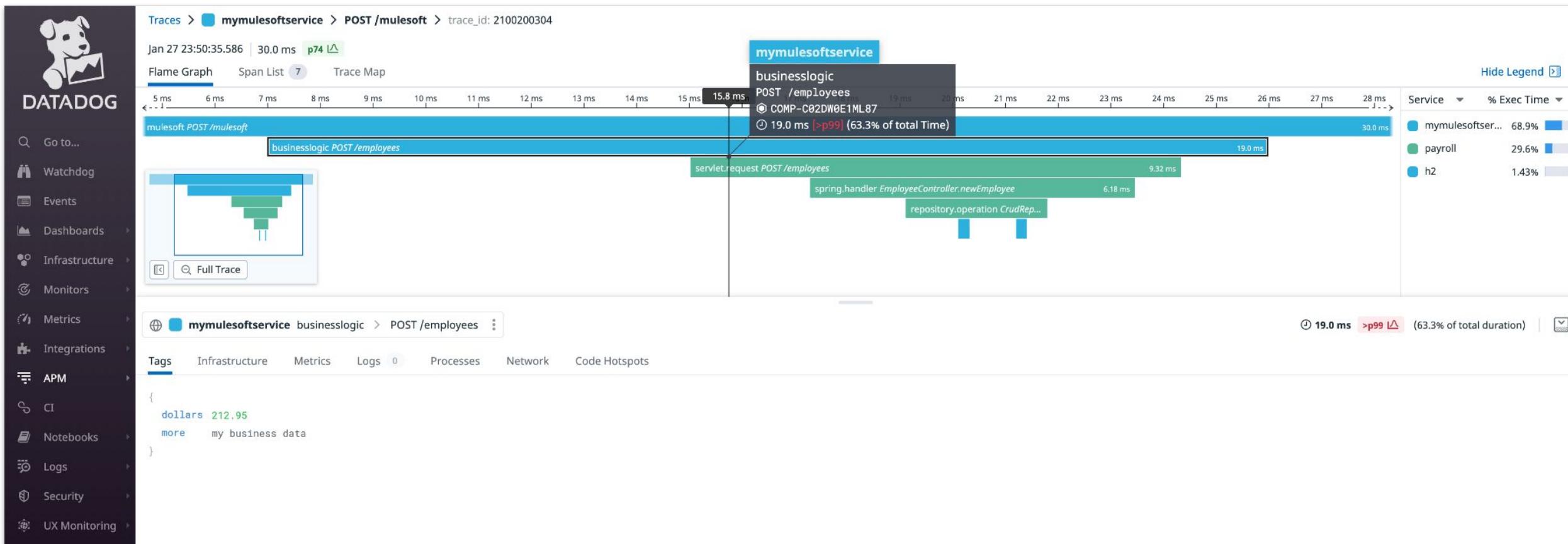
Further Reading

Send Traces to Agent Via API



DATADOG

Datadog APM



Datadog APM

The screenshot displays the Datadog APM interface, specifically the Traces view for the service `mymulesoftservice` performing a `POST /mulesoft` request.

Flame Graph: Shows the execution flow with the top span being `mulesoft POST /mulesoft` (50.0 ms) and the second span being `businesslogic POST /employees` (43.0 ms).

Metrics: A chart shows 2 total requests (< 0.1 req/s) over a 10-second period.

Facets: Facets for `Duration` show a range from 26ms to 50ms.

Tags: Tags for the request include `cost: 22.05`, `name: Lloyd Williams`, and `role: Enterprise Sales Engineer`.

Errors: One error was found: `error: true`.

Send Traces to Agent Via API

Request

Traces can be sent as an array of traces:

```
[ trace1, trace2, trace3 ]
```

And each trace is an array of spans:

```
trace1 = [ span, span2, span3 ]
```

and each span is a dictionary with a `trace_id`, `span_id`, `resource` and so on. Each span within a trace should use the same `trace_id`. However, `trace_id` and `span_id` must have different values.

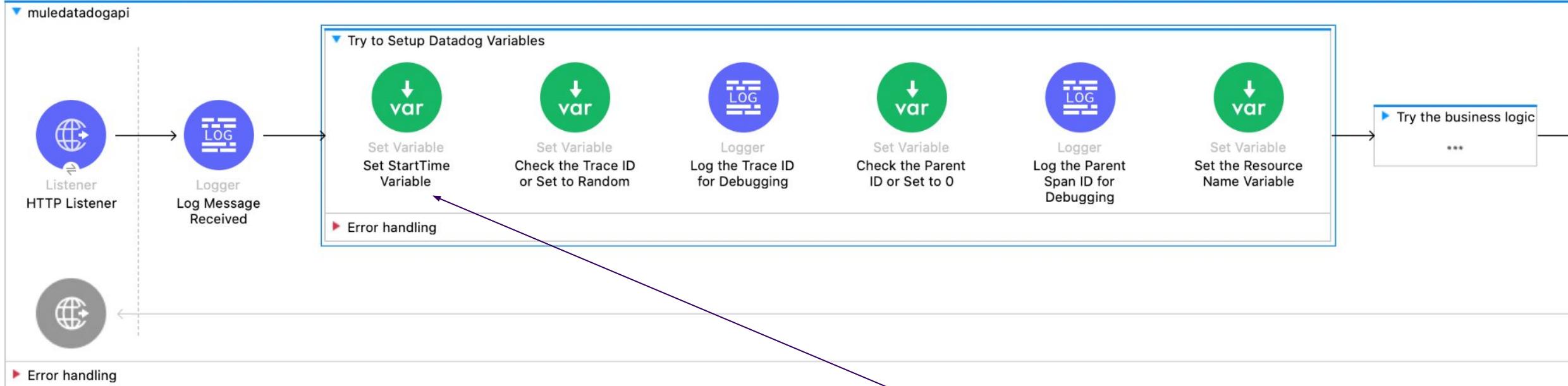
You can send multiple traces at once, but more importantly, you can send multiple spans within the same trace all at once.

Model

FIELD	TYPE	DESCRIPTION
<code>duration</code>	int64	The duration of the request in nanoseconds.
<code>error</code>	int32	Set this value to 1 to indicate if an error occurred. If an error occurs, you should pass additional information, such as the error message, type and stack information in the meta property.

The “Model” section of the documentation explains how to create the required payload (i.e. the ‘Body’ of the API call.)

Send Traces to Agent Via API



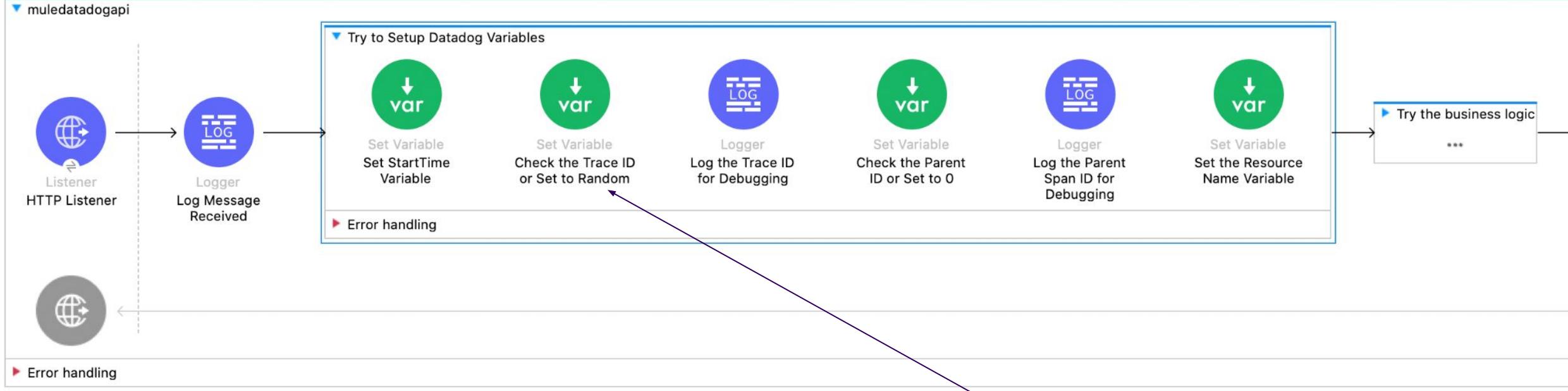
Display Name: Set StartTime Variable

Settings

Name: startTime

Value: `#[now() as Number {unit: "milliseconds"}]`

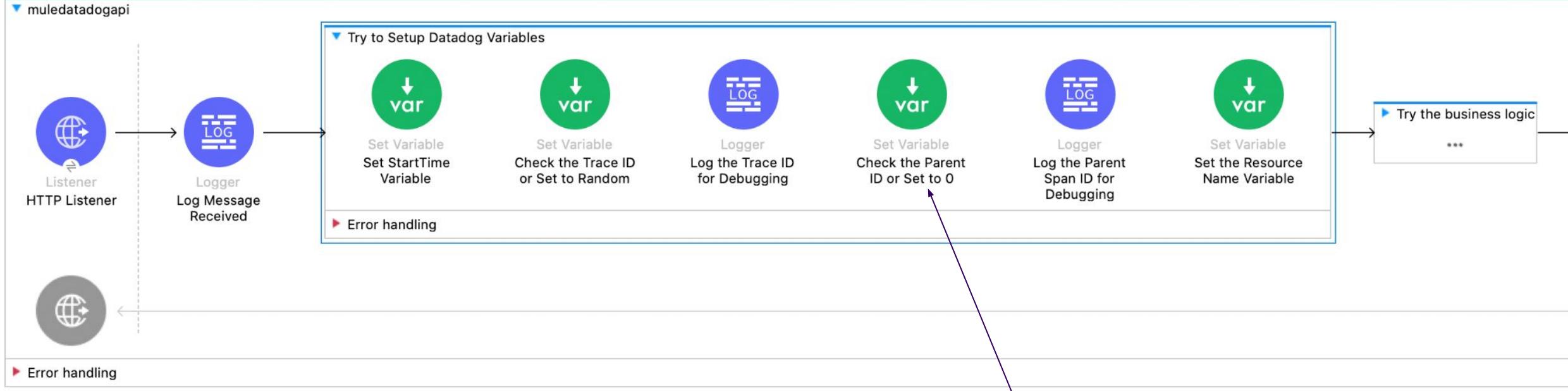
Send Traces to Agent Via API



Check if there is an existing Datadog Trace ID. In this case, 'x-datadog-trace-id' in the HTTP header or generate a random one.

```
1 %dw 2.0
2 output application/java
3 ---
4 if (attributes.headers['x-datadog-trace-id'] == null)
5 round(random() * 9223372036854775807) as String
6 else
7 attributes.headers['x-datadog-trace-id']
```

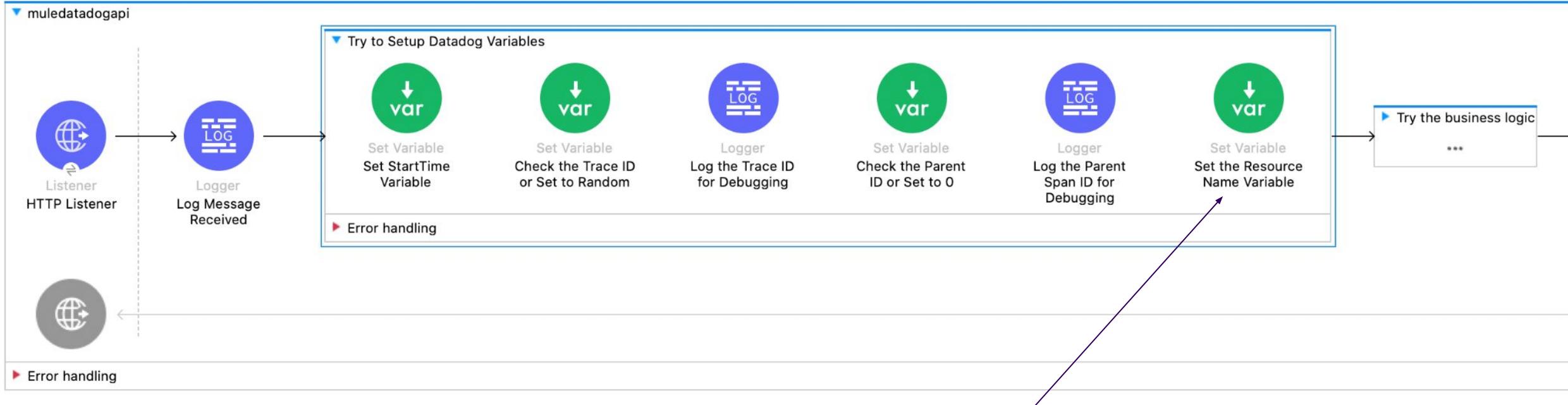
Send Traces to Agent Via API



Also, check if there is an existing Datadog Parent ID. In this case, 'x-datadog-parent-id' in the HTTP header or use 0.

```
1 ⊕ %dw 2.0
2   output application/java
3   ---
4 ⊕ if (attributes.headers['x-datadog-parent-id'] == null)
5   0 as String
6   else
7   attributes.headers['x-datadog-parent-id']
```

Send Traces to Agent Via API



Usually, the resource name is based on the URL. You can use automatically populated variables in MuleSoft to set the name.

Display Name: Set the Resource Name Variable

Settings

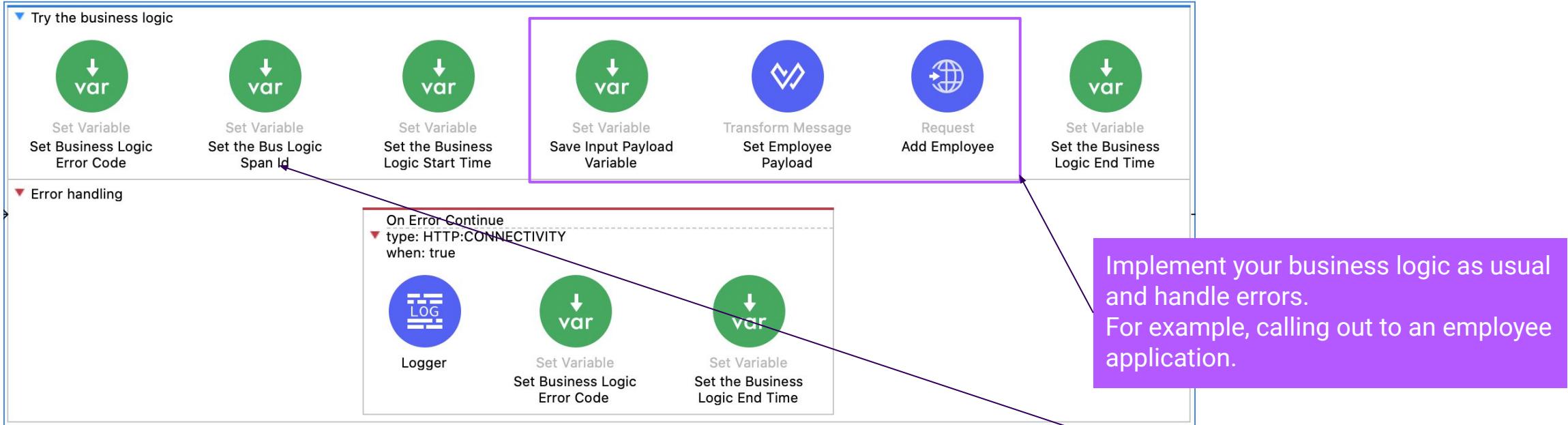
Name:

Value:



DATADOG

Send Traces to Agent Via API



With the business logic, you need to generate a random Datadog Span ID, set the start time and do error handling that you should always do anyway except you will set an error code for the span. Otherwise, do all Business Logic as usual. You do this separately for each major section of business logic that you want to trace.

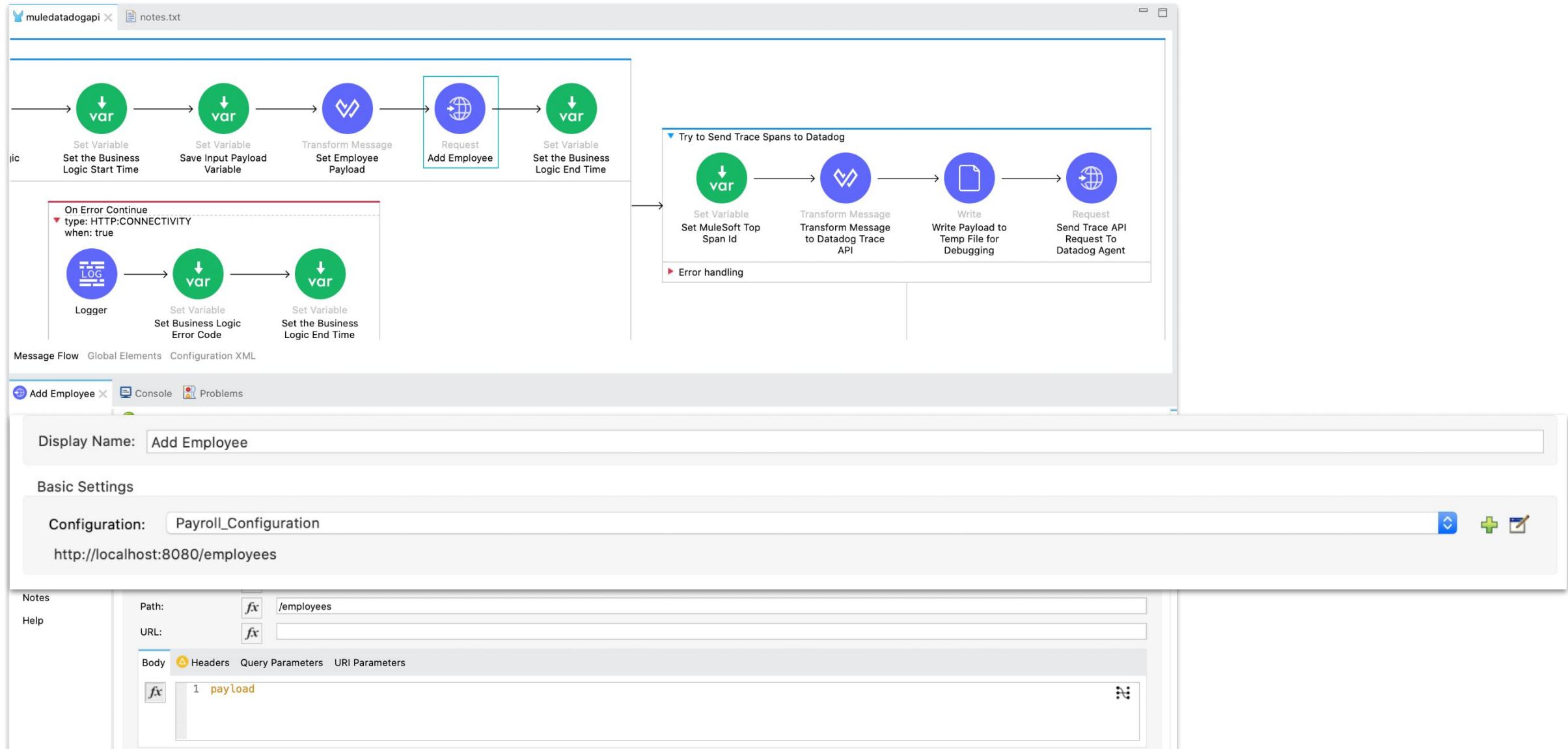
Display Name: Set the Bus Logic Span Id

Settings

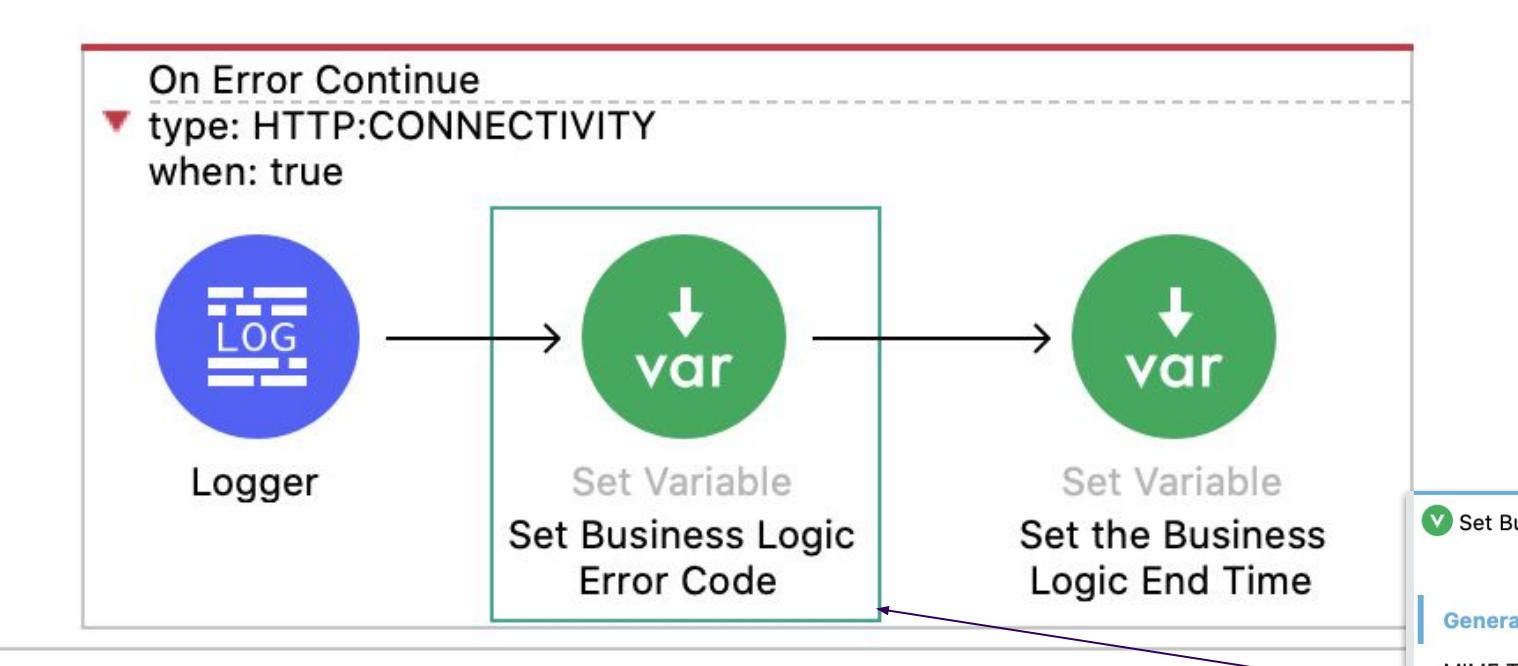
Name: busLogicSpanId

Value: #[round(random() * 9223372036854775807)

Send Traces to Agent Via API



Send Traces to Agent Via API



If there is an error, you want to set that in a variable so that it can be passed in the data for that Datadog "Span". 1 = Error

Set Business Logic Error Co... X Console Problems

General

Display Name: Set Business Logic Error Code

MIME Type

Metadata

Notes

Help

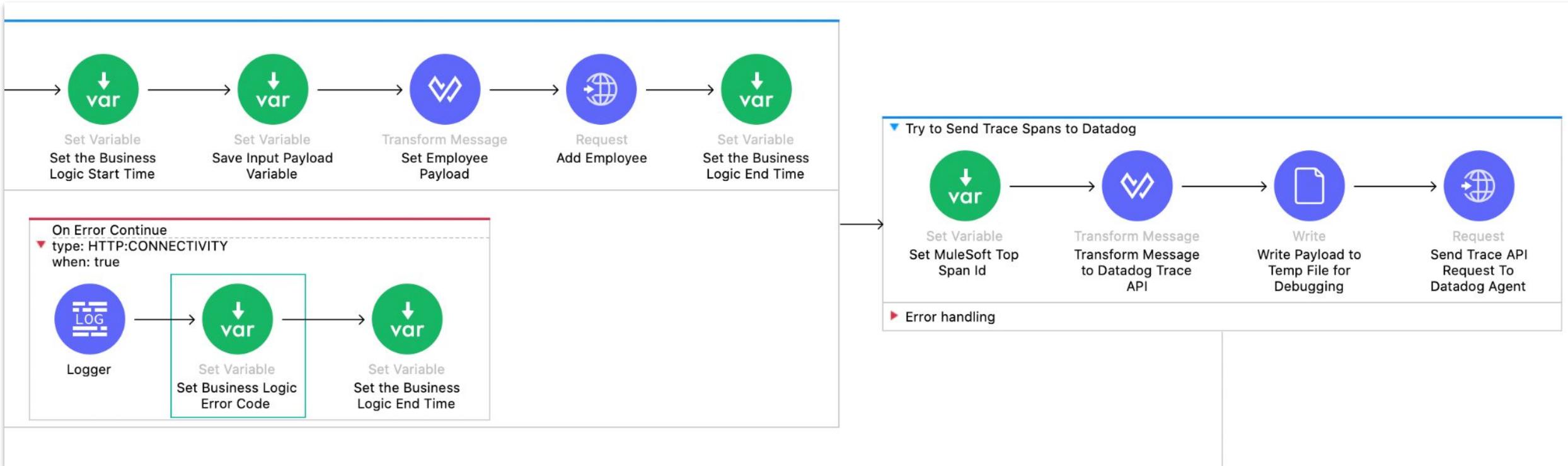
Settings

Name: busLogicErrorCode

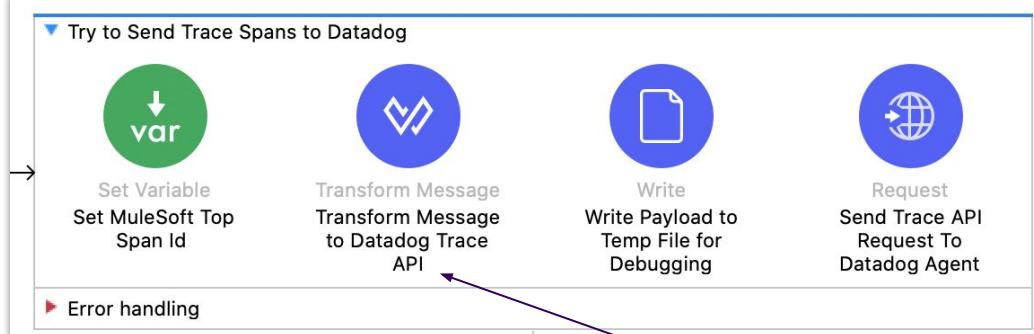
Value: #[1 as Number]

error	int32	Set this value to 1 to indicate if an error occurred. If an error occurs, you should pass additional information, such as the error message, type and stack information in the meta property.
-------	-------	---

Send Traces to Agent Via API



Send Traces to Agent Via API



Then put together the trace and spans to send to Datadog. Convert times to nanoseconds and set any other business related metrics that you want to show on the trace.

```
Output Payload ▾ + ⚪ Preview
1 %dw 2.0
2 output application/json
3 ---
4 [
5   [
6     {
7       "duration": ((now() as Number {unit: "milliseconds"}) - vars.startTime) * 1000000,
8       "error": 0,
9       "meta": vars.inputPayload,
10      "metrics": {
11        "cost": 22.05
12      },
13      "name": "mulesoft",
14      "parent_id": vars.parentSpanId as Number,
15      "resource": vars.resource,
16      "service": "mymulesoftservice",
17      "span_id": vars.topSpanId,
18      "start": vars.startTime * 1000000,
19      "trace_id": vars.traceid as Number,
20      "type": "web"
21    }
22 ],
23 [
24   [
25     {
26       "duration": (vars.busLogicEnd - vars.busLogicStart) * 1000000,
27       "error": vars.busLogicErrorCode,
28       "meta": {
29         "more": "my business data"
30       },
31       "metrics": {
32         "dollars": 212.95
33       },
34       "name": "businesslogic",
35       "parent_id": vars.topSpanId,
36       "resource": "POST /employees",
37       "service": "mymulesoftservice",
38       "span_id": vars.busLogicSpanId,
39       "start": vars.busLogicStart * 1000000,
40       "trace_id": vars.traceid as Number,
41       "type": "web"
42     }
43 ]]
```

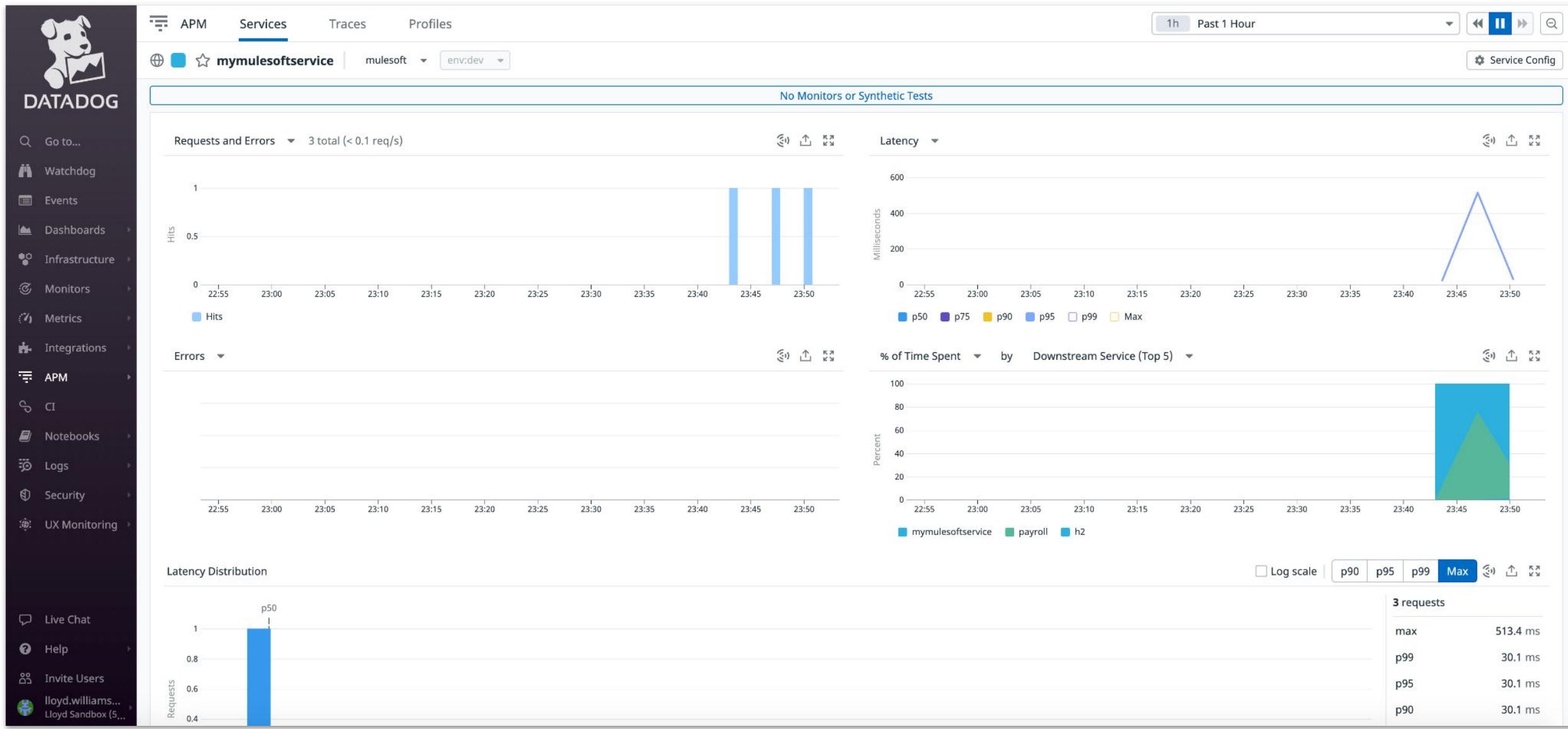
Send Traces to Agent Via API

The screenshot shows the Datadog Service List interface. The top navigation bar includes APM, Services (selected), Traces, and Profiles. A time range of "1h Past 1 Hour" is set. On the left, a sidebar lists various monitoring categories like Events, Dashboards, Infrastructure, Monitors, and Metrics. The main area displays a table of services with the following data:

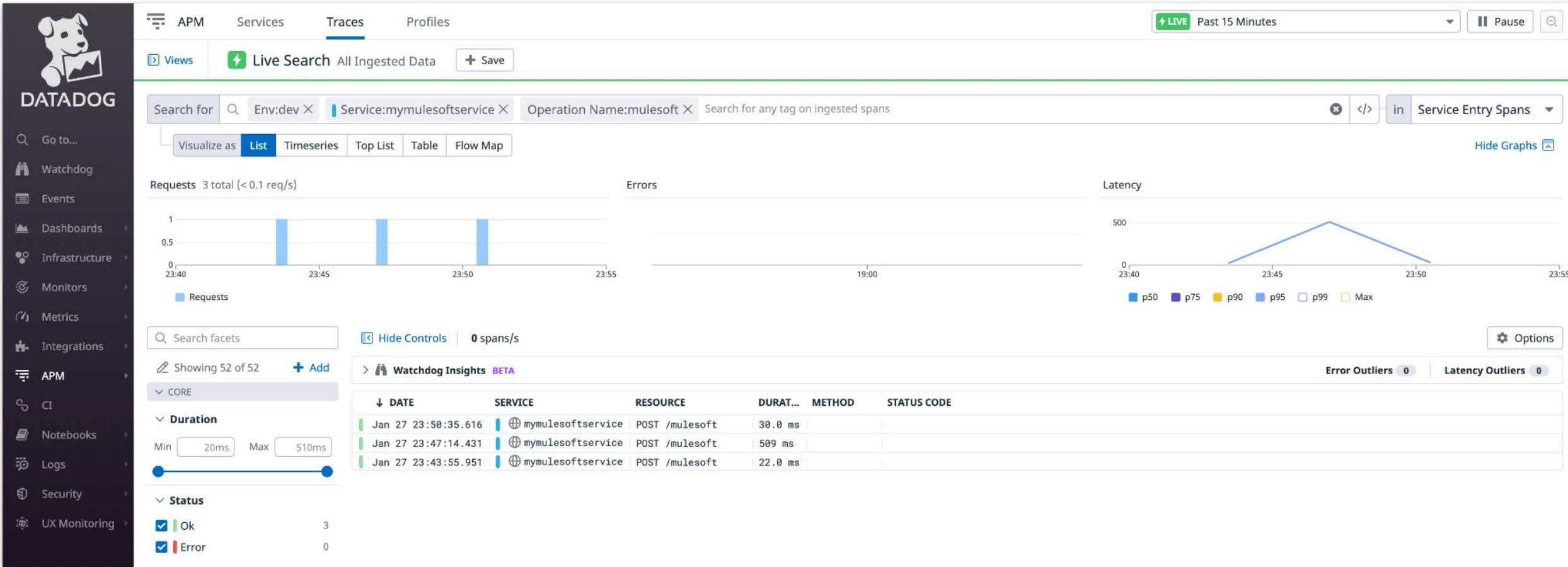
TYPE	SERVICE	LAST DEPLOY	REQUESTS	P50 LATENCY	P95 LATENCY	ERROR RATE	INFRA	MONITORS
Web	bookdetails	2 mo ago	< 0.1 req/s	4.73 ms	6.92 ms	0%		
DB	payroll	6 m ago	< 0.1 req/s	9.35 ms	386 ms	0%		
Cache	mymulesoftservice		< 0.1 req/s	29.8 ms	510 ms	0%		
Custom	h2	6 m ago	< 0.1 req/s	494 µs	4.64 ms	0%		

Facets on the left show the distribution of services by type: Web (2), DB (0), Cache (0), Function (0), and Custom (2). A search bar at the top allows filtering by tag or service name.

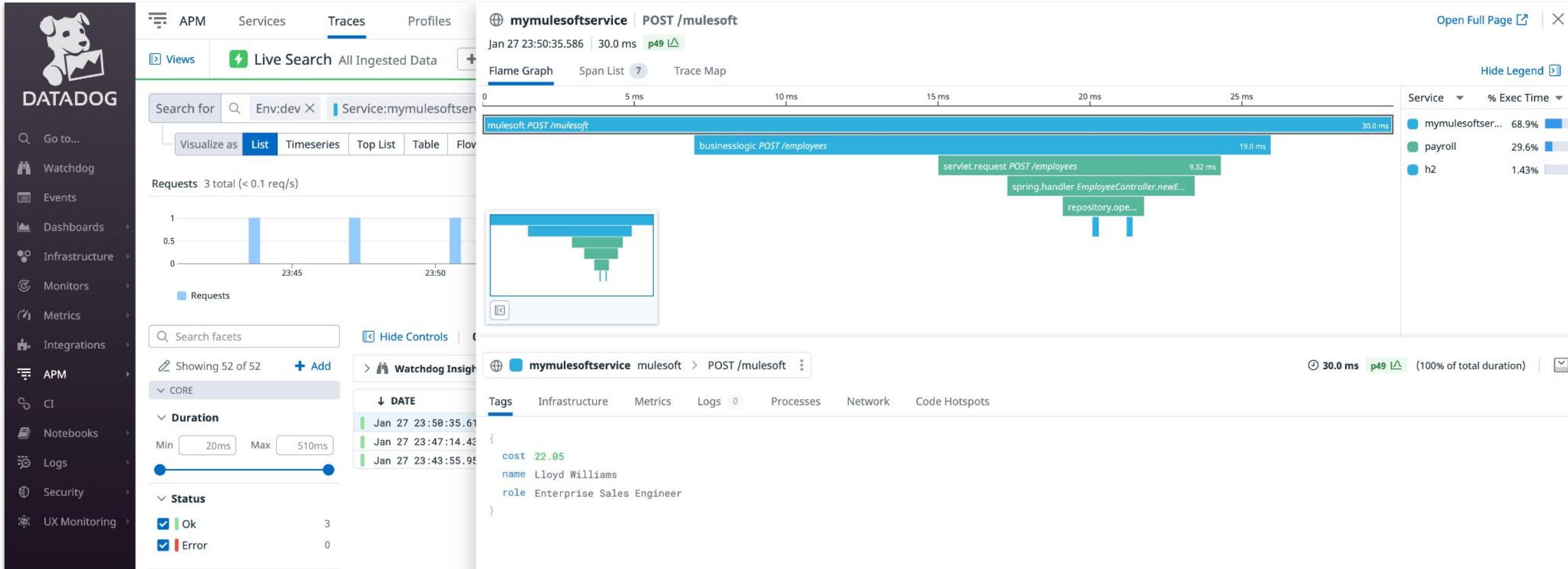
Send Traces to Agent Via API



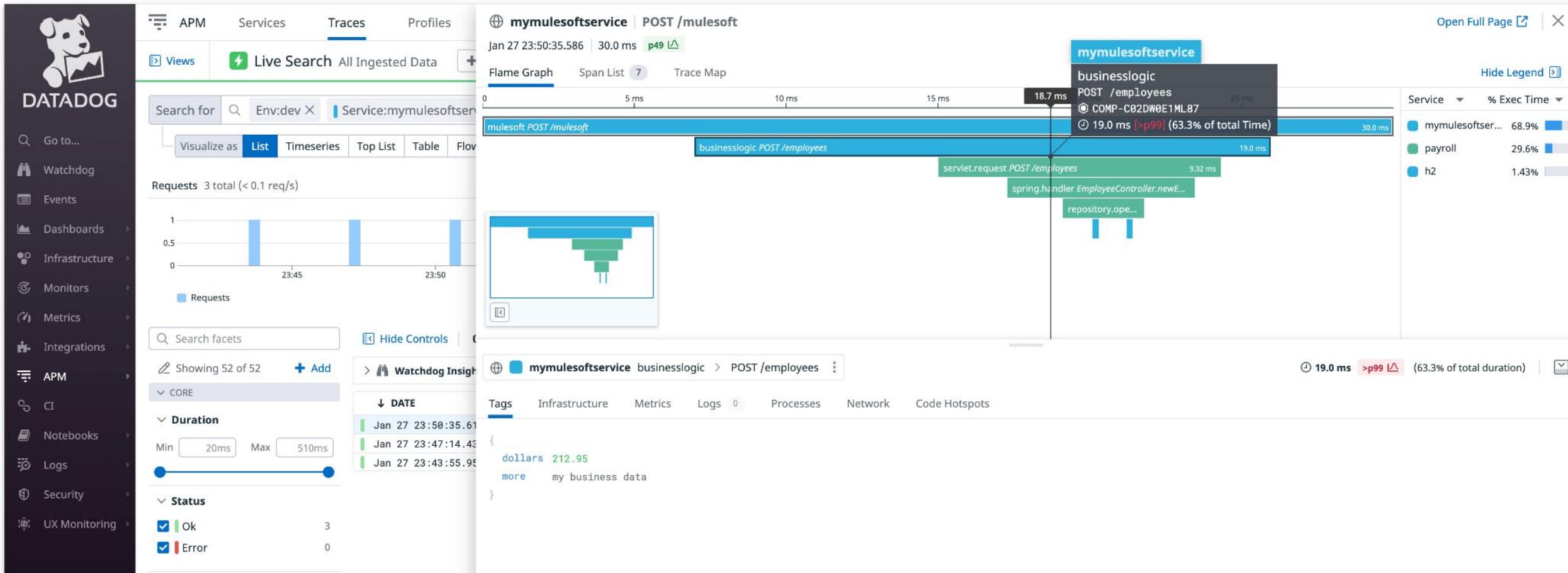
Send Traces to Agent Via API



Send Traces to Agent Via API



Send Traces to Agent Via API

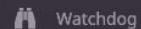




DATADOG



Go to...



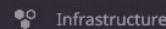
Watchdog



Events



Dashboards



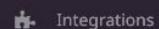
Infrastructure



Monitors



Metrics



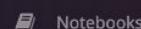
Integrations



APM



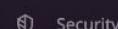
CI



Notebooks



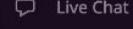
Logs



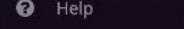
Security



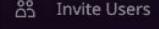
UX Monitoring



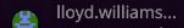
Live Chat



Help



Invite Users

lloyd.williams...
Lloyd Sandbox (5...)

APM Services Traces Profiles

mymulesoftservice | POST /mulesoft

Open Full Page X

Views

Live Search All Ingested Data

Jan 27 23:50:35.586 | 30.0 ms p49 ↗

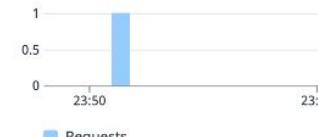
Flame Graph Span List 7 Trace Map

Hide Legend

Search for Env:dev X Service:mymulesoftser...

Visualize as List Timeseries Top List Table Flow

Requests 3 total (< 0.1 req/s)



Search facets

Hide Controls

> Watchdog Insights

payroll servlet.request > POST /employees

9.32 ms >p99 ↗ (31.1% of total duration)

Tags Infrastructure Metrics Logs 0 Processes Network Code Hotspots

DATE

- component tomcat-server
- env dev
- http
 - host localhost
 - hostname localhost
 - method POST
 - path_group /employees
 - route /employees
 - status_code 200
 - url http://localhost:8080/employees ↗
- language jvm
- peer
 - ipv4 127.0.0.1
 - port 53166
- runtime_id 0635c2c2-175b-417d-892f-1f2ef75ce856
- servlet

#!/bin/sh

java -javaagent:/Users/lloyd.williams/u01/datadog/dd-java-agent.jar -Ddd.profiling.enabled=false -Ddd.logs.injection=true -Ddd.service=payroll -Ddd.env=dev -Ddd.version=0.0.2 -jar target/payroll-0.0.2.jar



DATADOG

“Payroll” called from MuleSoft is a Java Spring Boot microservice instrumented with the dd-trace-agent that also calls a database.

Datadog APM—Tracing without Limits™

Ingest all traces, search them live & retain the most valuable

Your Application

Any Stack



Any Environment



Anywhere



Search 100% Traces Live. No Sampling

DATE	SERVICE	DURATION	HTTP MET...	STATUS CODE	RESOURCE
Aug 29 17:19:53.129	web-store	88.4 ms		200	> CartItemsController#show
Aug 29 17:19:52.923	web-store	410 ms	POST	200	> ShoppingCartController#apply_coupon
Aug 29 17:19:52.892	web-store	20.2 ms			> rails.render_template
Aug 29 17:19:52.430	web-store	29.8 ms	GET	200	> CartItemsController#show
Aug 29 17:19:51.783	web-store	4.55 ms			> rails.render_template
Aug 29 17:19:51.419	web-store	718 ms	POST	200	> ShoppingCartController#apply_coupon
Aug 29 17:19:51.003	web-store	5.30 ms			> rails.render_template
Aug 29 17:19:50.977	web-store	548 ms	GET	200	> ProductsController#index
Aug 29 17:19:50.976	web-store	60.4 ms			> ProductsController#index

Retain High-Value Traces

- Store all error & high latency traces
- Set tag-based retention rules
- Correlate with logs, synthetics & infrastructure

Generate Out-of-the-box Metrics

- Setup alerts & notifications
- Visualize and alert on application metrics
- Detect anomalies automatically

High Cardinality App Analytics

- Drill down using tags & filters
- Automate root cause analysis
- Track browser, mobile, application & database

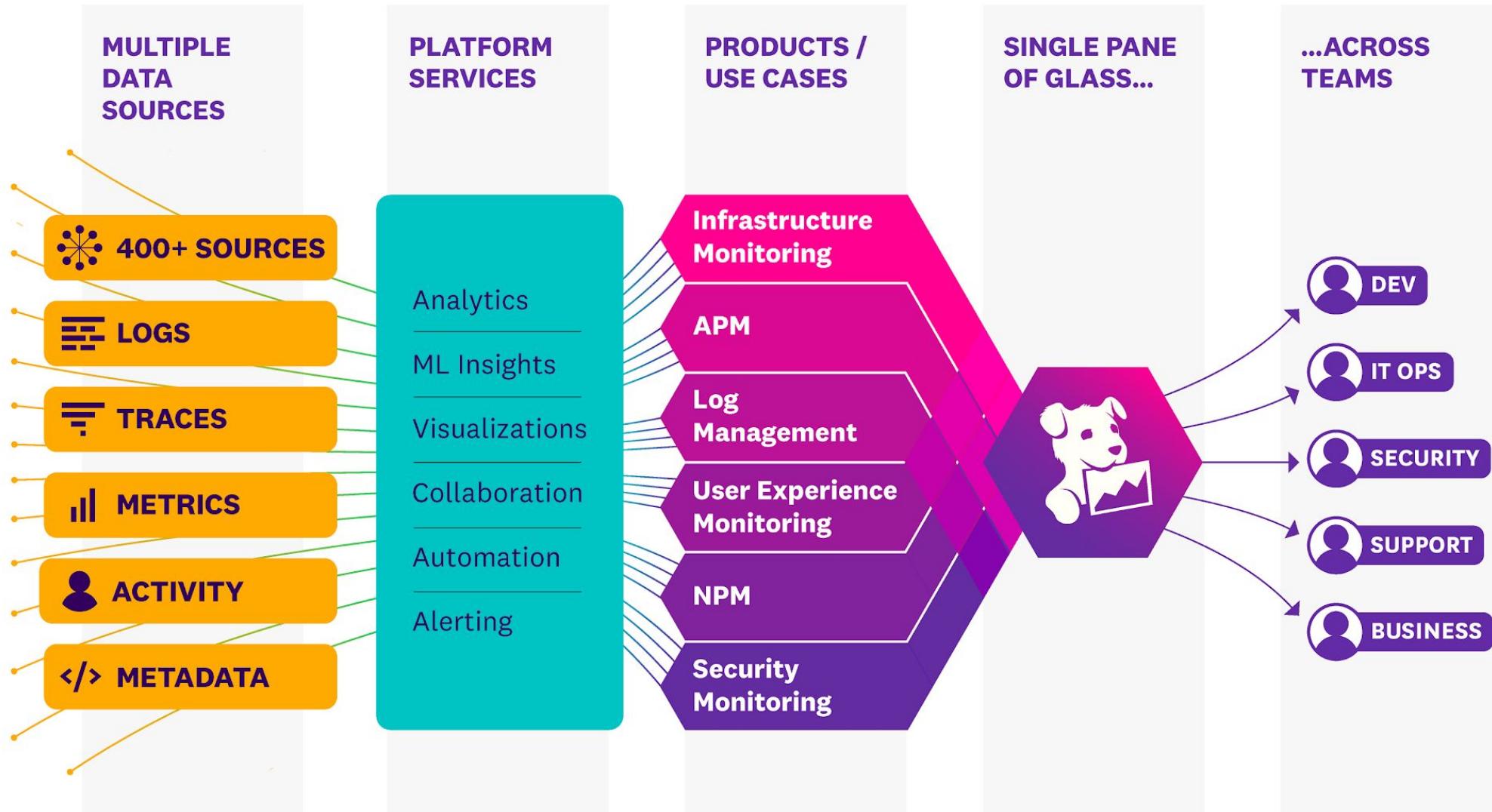
Datadog Application Performance Monitoring (APM)

- Datadog Application Performance Monitoring (APM) provides end-to-end distributed tracing from frontend devices to databases—with no sampling.
- If you want to sample, you can do it more intelligently with Tracing without Limits™.
- Tagging is built into the core of the Datadog architecture and makes it a platform that uses your language for searching, filtering and correlating different types of monitoring together.
- Datadog is a market leader (Gartner MQ) in Application Performance Monitoring (APM) for tracing requests through distributed applications.
- The Datadog APM SDK allows you to easily add custom metrics to a trace with very little code.
- Follow the User Experience (UX) Real User Monitoring (RUM) session to its related back-end traces.
- Trace through messaging layers like Kafka and view calls (SQL) into databases.



DATADOG

Datadog – Unified SaaS Observability, Monitoring and Analytics Platform



Why Datadog?

Unified observability - Integrated platform, used by everyone

Cloud-agnostic - Built for the modern stack, deployed everywhere

Ease of use - OOTB seamless correlation, no manual querying

Logging without Limits - ingest all logs cost-effectively & analyze in real-time

Superior granularity and retention - 15 secs for 15 months, no rollups

Data-driven, actionable alerting - Reduce alert fatigue with machine learning



DATADOG

Datadog solves complexity

