

Building and Monitoring Generative AI Azure Logic Apps



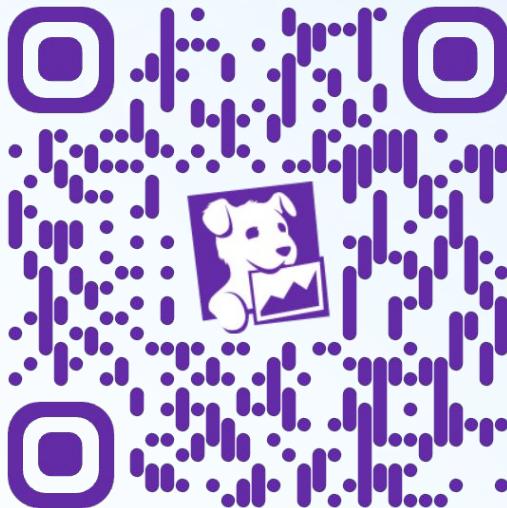
DATADOG



Jason Hand

Senior Developer Advocate
Datadog

Agenda



- 1** Building low-code applications using Azure Logic Apps
- 2** What can be collected from generative AI integrations
- 3** How to send logs and metrics to Datadog from Azure Logic Apps
- 4** How to add real user monitoring (RUM) to a web application
- 5** How to visualize Azure Logic App, OpenAI, and web app data in Datadog
- 6** Takeaways & Resources



Nvidia GTC Conf

notion.so/jasonhand/Nvidia-GTC-Conf-ca4a9c5952d84fc4bb... | New Chrome available | All Bookmarks

Conference Notes / Nvidia GTC Conf

Share | All Bookmarks

Nvidia GTC Conf

Sessions attended

17 sessions including the following:

- Nvidia GTC 2024 -
- How to Apply Generative AI
- Multi-Modal Integration
- LLMOps: The New Frontier
- A Culture of Open and Inclusive Collaboration
- The Unsolved Challenges of Large Language Models
- Scaling Creative Horizons
- Visualizing New Narratives
- Energy-Efficient Generative AI
- Driving Accessibility and Inclusion in AI
- Information Integrity and Trust
- How to Monitor Your Generative AI Tech Stack [EXPT63170]
- Learn How Educators Are Using Generative AI
- "The Creative Source": A Deep Dive into AI Art
- Modernizing Games with Generative AI
- State of PyTorch [Session 2]

notion.so/jasonhand/How-to-Monitor-Your-Generative-AI-Tech-Stack-[EXPT63170] | New Chrome available | All Bookmarks

Conference Notes / Nvidia GTC Conf / How to Monitor Your Generative AI Tech Stack [EXPT63170]

Share | All Bookmarks

How to Monitor Your Generative AI Tech Stack [EXPT63170]

Datadog's generative AI integrations are designed to provide visibility into your applications and services that make use of large language models, and with these integrations you can effectively monitor the performance of your AI stack. Learn how these integrations work and how the various pieces fit together as a whole when deploying models or developing your own. By taking a layered approach, we'll delve into specific topics related to each layer, ensuring that the entire process is covered.

Ryan MacLean, Senior Tech Evangelist, Datadog

Industry: All Industries

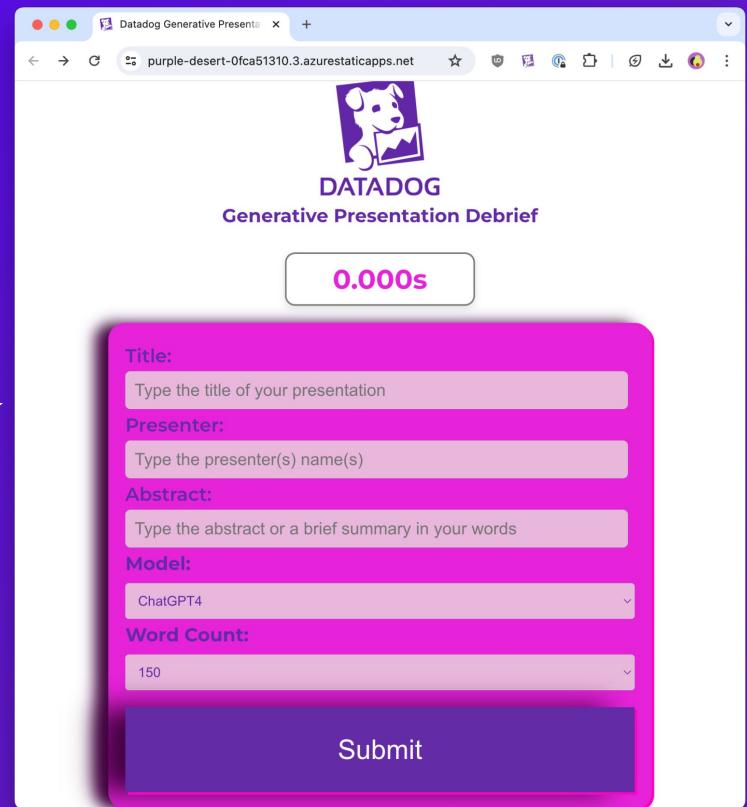
- Wednesday, Mar 20 4:15 PM - 4:30 PM PDT Expo Theater

+ :: The presentation focused on using Datadog's monitoring and analytics platform to enhance observability and security for AI and machine learning (ML) infrastructures, particularly Large Language Models (LLMs). Here's a summary of the key topics, takeaways, and notable quotes:

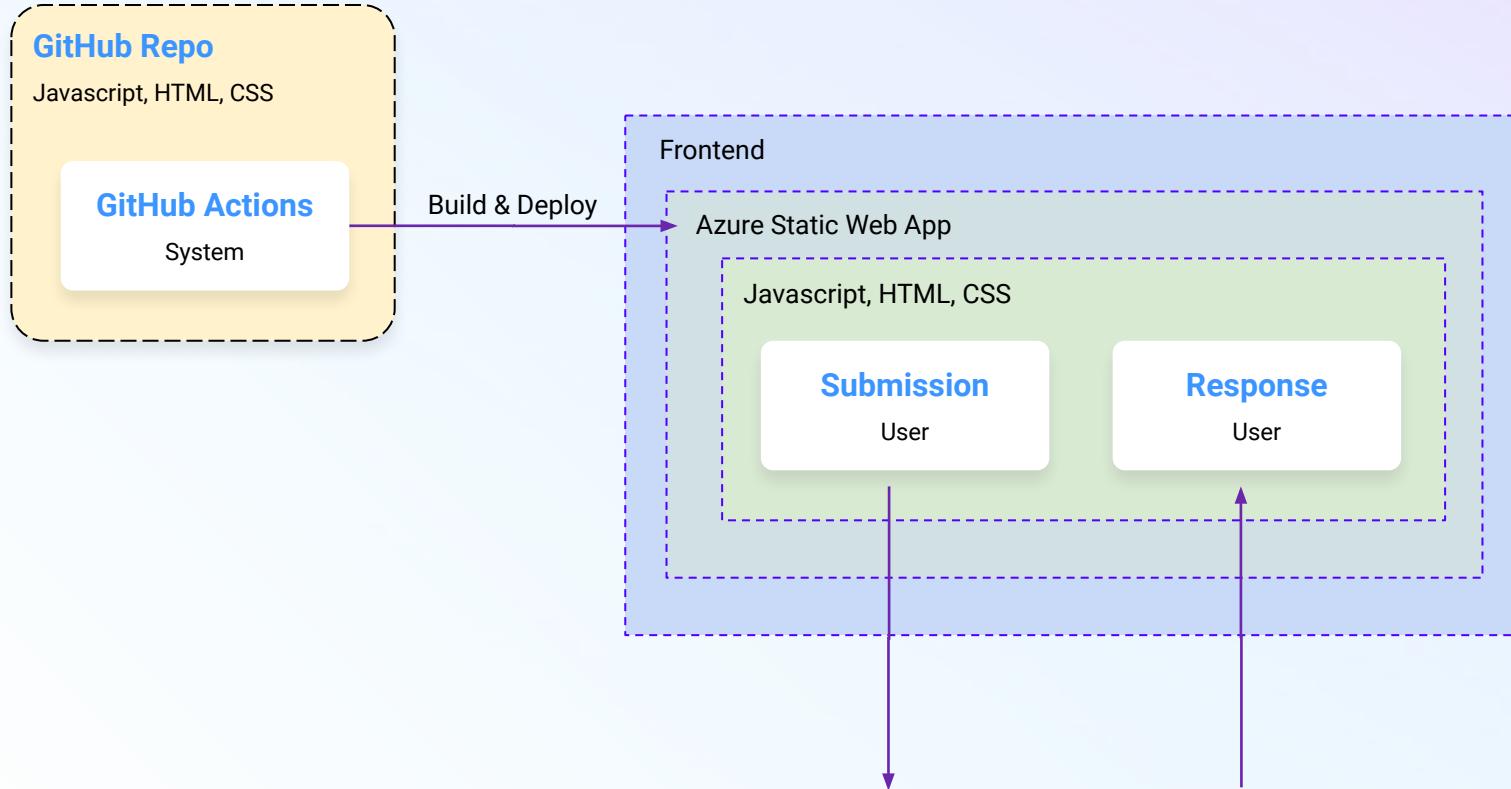
A screenshot of a developer's workspace. At the top, a code editor shows `App.js` with logic for a DataDog browser run. Below it is a GitHub repository page for `generative-debrief`. A large white arrow points from the GitHub page down towards the Microsoft Azure portal. The Azure portal shows the deployment of a static web app named `generative-debrief`, which is currently running on port 80.

```
import { datadogRun } from '@datadog/browser-run';
import { datadogsLogs } from '@datadog/browser-logs';
import React, { useState, useEffect } from 'react';
import './App.css';
import ResponseComponent from './ResponseComponent';

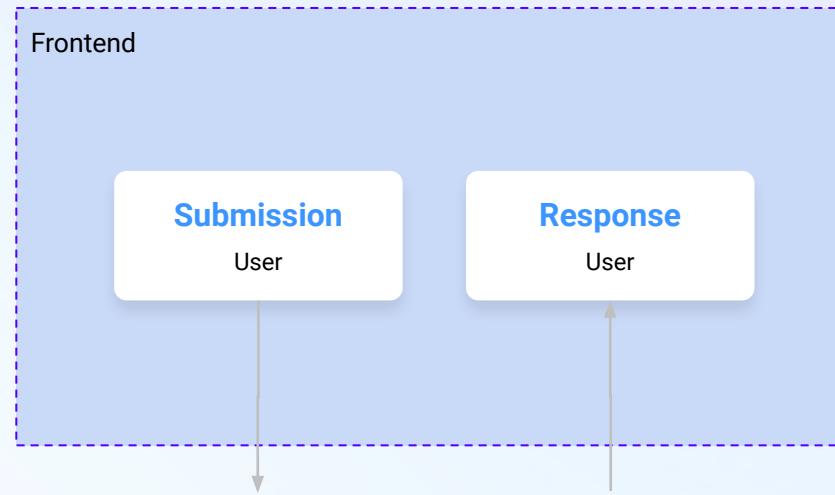
datadogsLogs.init({
  clientToken: process.env.REACT_APP_DD_CLIENT_TOKEN,
  site: process.env.REACT_APP_DD_SITE,
  forwardErrorsToLogs: true,
  sessionSampleRate: 100,
})
```



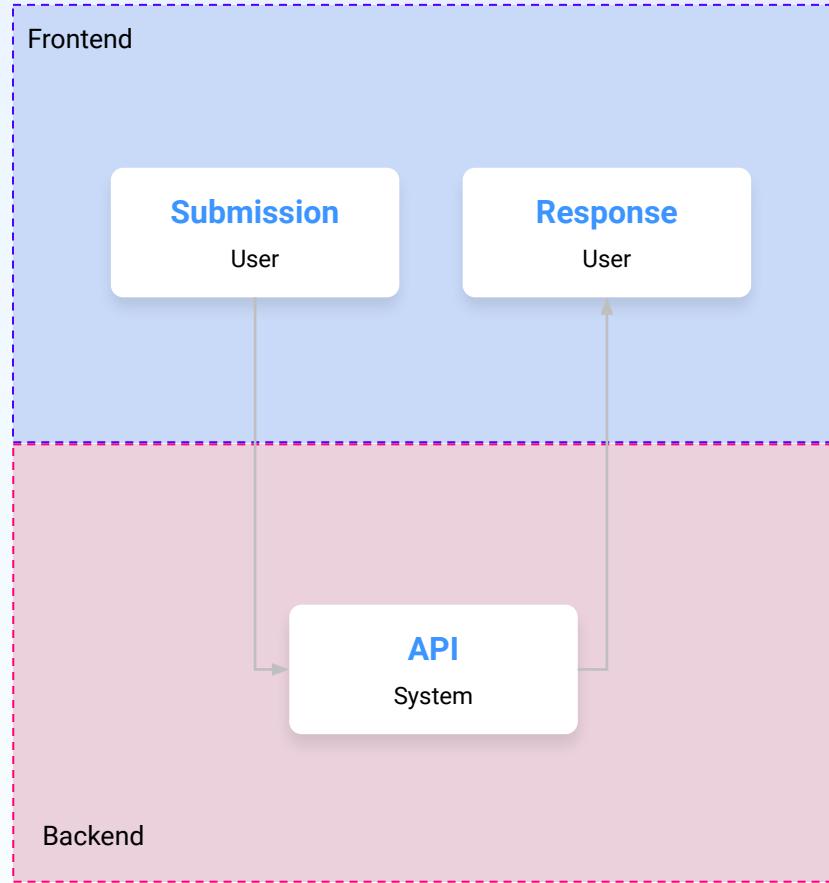
Generative Debrief

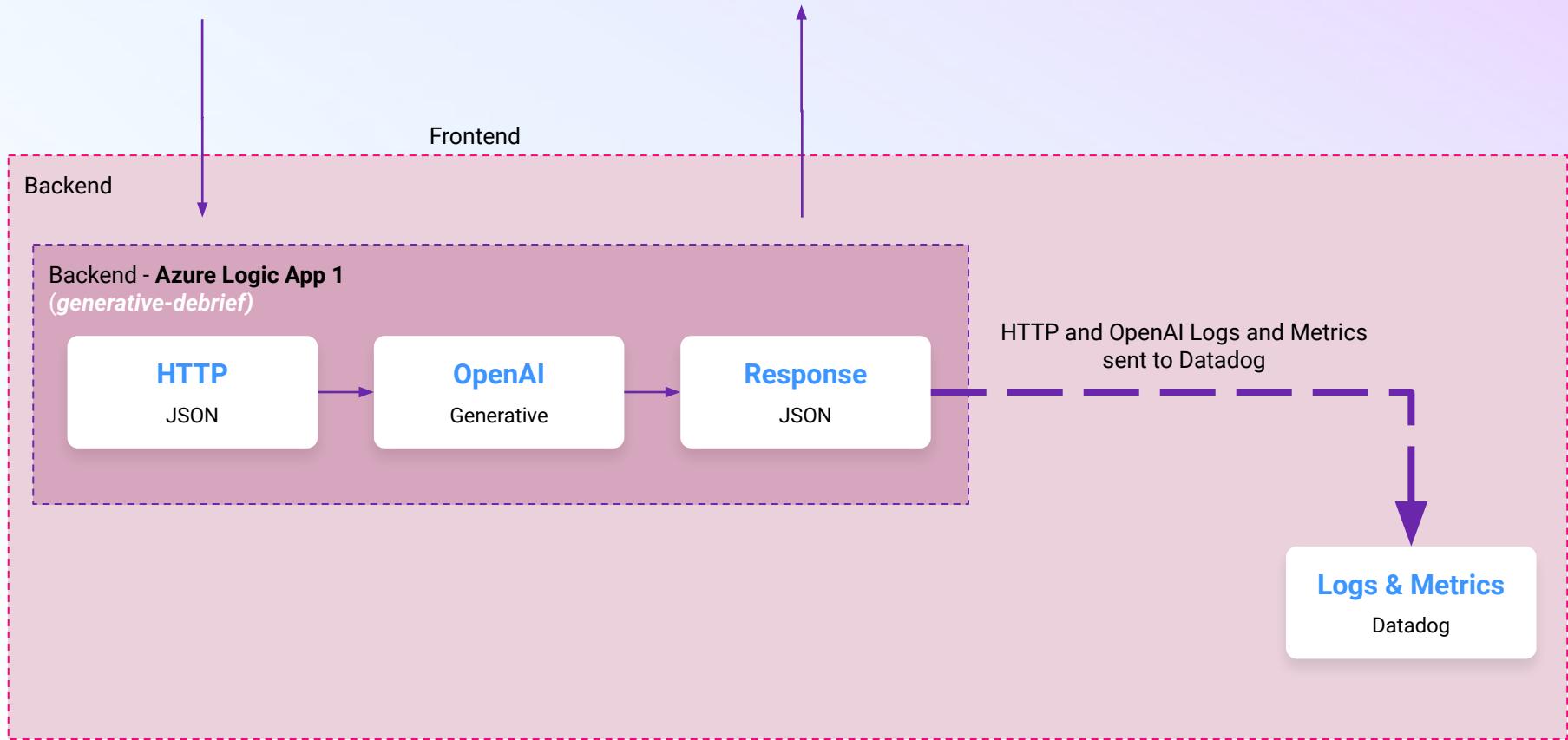


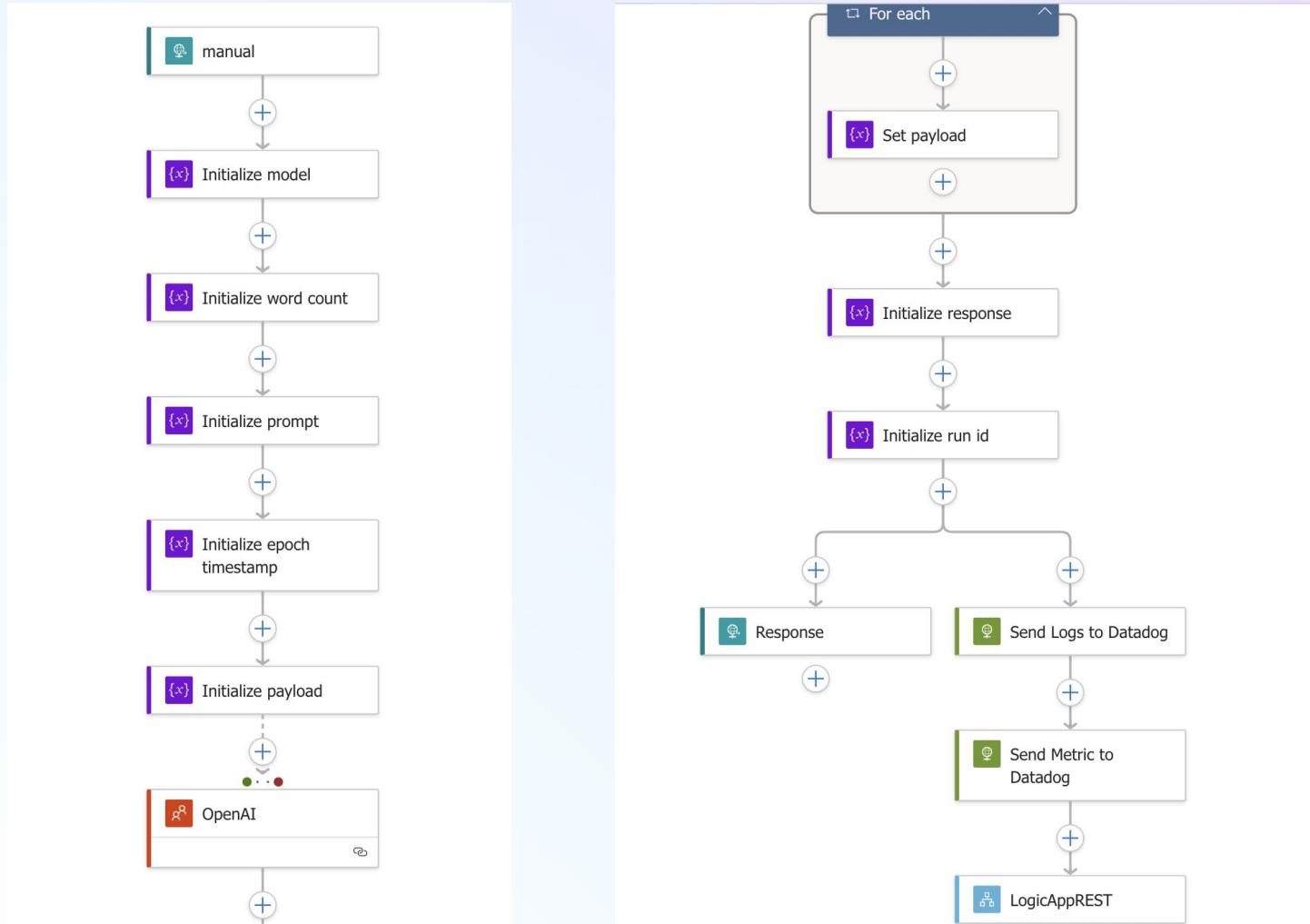
Simplified

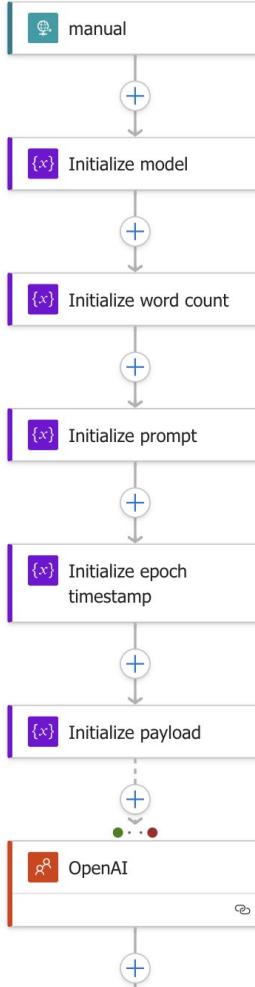


Generative Debrief







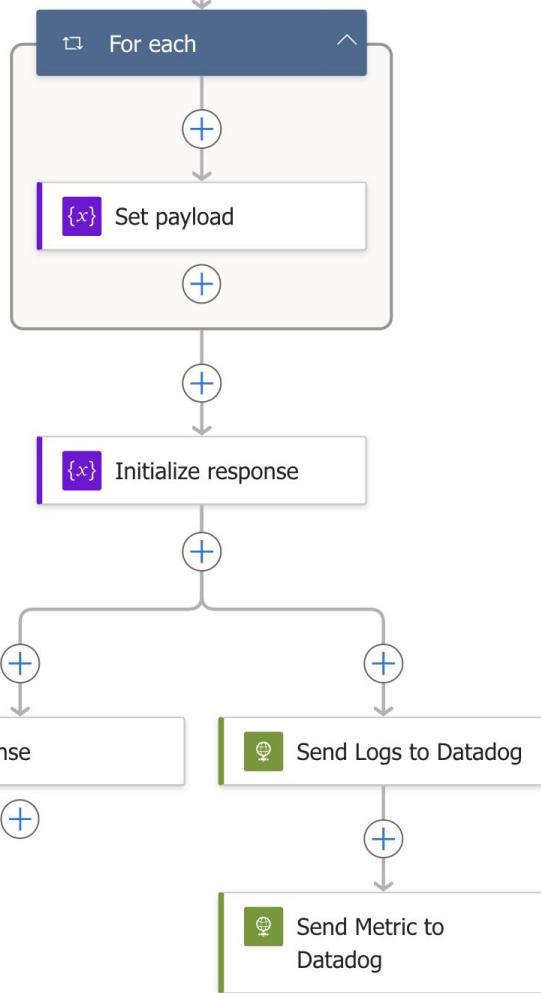


HTTP Request

Create variables

- Model
- Word count
- Prompt
- Timestamp
- Payload

Call OpenAI



Put openai response in variable

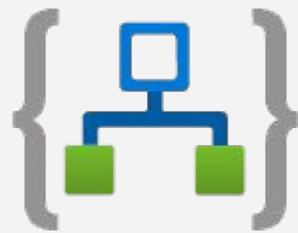
Create response variable

Respond to User

Send Logs & Metrics to Datadog

Live Demo

(Low / No Code + Generative AI + Datadog)



Azure Logic Apps

Let's look at the Logic App inside Azure

What can be collected from generative AI integrations

The screenshot shows a browser window with two tabs open, both displaying the Datadog OpenAI Overview Dashboard.

OpenAI Overview Dashboard: This panel shows a search bar with filters for env, service, version, model, organization, and openai.user.api_key. The time range is set to "1w Past 1 Week".

Azure OpenAI Overview: This panel shows a search bar with filters for openai_resource_name, resource_group, and subscription_name. The time range is set to "1h Past 1 Hour".

Usage Overview: This section displays four key metrics:

- Azure OpenAI Instances:** Value: 8
- Total Token Usage:** Value: 0
- Total Requests:** Value: 0
- Average Request Latency:** Value: 0s

Azure OpenAI Logo: The Azure OpenAI logo is prominently displayed in the center of the dashboard.

Further reading:

- [Datadog OpenAI Integration Documentation](#)
- [Monitor OpenAI with Datadog Blog](#)

Left Sidebar: The sidebar contains several sections:

- Recent, Dashboards, Monitors, Watchdog, Service Mgmt
- Infrastructure, APM, Digital Experience, Software Delivery, Security
- Metrics, Logs
- Integrations, Bits AI, Screen Share, jason.hand@datadog.com (11 items)

Azure Monitoring & Integrations

Integrations | Datadog

app.datadoghq.com/integrations?category=Azure

57 results for Azure in Integrations

 Azure by Datadog	 Azure Active Directory by Datadog	 Azure Analysis Services by Datadog	 Azure API Management by Datadog	 Azure App Service Envir... by Datadog
 Azure App Service Plan by Datadog	 Azure App Services by Datadog	 Azure Application Gatew... by Datadog	 Azure Arc by Datadog	 Azure Automation by Datadog
 Azure Backup by Datadog	 Azure Blob Storage by Datadog	 Azure Cognitive Services by Datadog	 Azure Container Apps by Datadog	 Azure Container Instanc... by Datadog
 Azure Data Lake Storage by Datadog	 Azure Data Factory by Datadog	 Azure Data Lake Analytics by Datadog	 Azure DevOps by Datadog	 Azure Event Grid by Datadog
Azure Functions by Datadog	Azure Front Door by Datadog	Azure Government by Datadog	Azure Logic Apps by Datadog	Azure Monitor by Datadog
Azure Monitor Metrics by Datadog	Azure Monitor Metrics Insights by Datadog	Azure Monitor Metrics Insights Insights by Datadog	Azure Monitor Metrics Insights Insights Insights by Datadog	Azure Monitor Metrics Insights Insights Insights Insights by Datadog

Recent Dashboards Monitors Watchdog Service Mgmt Infrastructure APM Digital Experience Software Delivery Security Metrics Logs Integrations Bits AI Screen Share Jason.hand@datadoghq.com Datadog Demo (11...) Help ?

New Chrome available :

How to send logs to Datadog from Azure Logic Apps

The screenshot shows a web browser displaying the Datadog documentation for Log Collection and Integrations. The page has a purple header with the Datadog logo and navigation links for PRODUCT, CUSTOMERS, PRICING, SOLUTIONS, ABOUT, BLOG, LOGIN, and GET STARTED FREE. Below the header, there's a banner for the 2023 State of Cloud Security Study. The main content area has a breadcrumb navigation: DOCS > LOG MANAGEMENT > LOG COLLECTION AND INTEGRATIONS. On the left, there's a sidebar for Log Management with sections for Browser, Android, iOS, Flutter, React Native, Roku, C#, Go, Java, NodeJS, PHP, Python, and Ruby. A search bar is also present. The main content starts with an "Overview" section, followed by a "Setup" section with a note about reserved attributes. On the right, there are language and site selection dropdowns (set to English and US1), an "EDIT" button, and a "ON THIS PAGE" sidebar with links to Overview, Setup, Additional configuration options, and Next steps.

Log Collection and Integrations

OVERVIEW

Choose a configuration option below to begin ingesting your logs. If you are already using a logshipper daemon, refer to the dedicated documentation for [Rsyslog](#), [Syslog-ng](#), [NXlog](#), [FluentD](#), or [Logstash](#).

Consult the [list of available Datadog log collection endpoints](#) if you want to send your logs directly to Datadog.

Note: When sending logs in a JSON format to Datadog, there is a set of reserved attributes that have a specific meaning within Datadog. See the [Reserved Attributes](#) section to learn more.

SETUP

Host Application Container Serverless Cloud/Integration

Action: Send Logs to Datadog

The screenshot shows a Logic App flow with the following steps:

- A "Response" step with a plus icon.
- A "Send Logs to Datadog" step (highlighted in blue).
- A "Send Metric to Datadog" step.
- A "LogicAppREST" step with a plus icon.

Below the flow, there is a detailed configuration for the "Send Logs to Datadog" step:

URI *: https://http-intake.logs.datadoghq.com/api/v2/logs

Method *: POST

Headers:

Content-Type	application/json
DD-API-KEY	aa6589 b3

Queries:

Enter key	Enter value
-----------	-------------

Body:

```
{  
  "host": "Azure",  
  "service": "Logic App",  
  "model": "({}) model x",  
  "word_count": "({}) word_count x",  
  "prompt": "({}) prompt x",  
  "payload": "({}) payload x",  
  "Host": "({}) Host x",  
  "User-Agent": "({}) User-Agent x",  
  "Origin": "({}) Origin x",  
  "sec-ch-ua": "({}) sec-ch-ua x",  
  "sec-ch-ua-platform": "({}) sec-ch-ua-platform x",  
}
```

NOTE:
You will need
a Datadog
API Key.

Action:
Send Logs to
Datadog

Logs API Endpoint URL:

<https://http-intake.logs.datadoghq.com/api/v2/logs>

Action: Send Logs to Datadog

JSON sent to Datadog Use the Body below

```
{  
    "service": "Logic App",  
    "model": "@{variables('model')}",  
    "word_count": "@{variables('word_count')}",  
    "prompt": "@{variables('prompt')}",  
    "payload": "@{variables('payload')}",  
    "Host": "@{triggerOutputs()['headers']['Host']}",  
    "User-Agent": "@{triggerOutputs()['headers']['User-Agent']}",  
    "Origin": "@{triggerOutputs()['headers']['Origin']}",  
    "sec-ch-ua": "@{triggerOutputs()['headers']['sec-ch-ua']}",  
    "sec-ch-ua-platform": "@{triggerOutputs()['headers']['sec-ch-ua-platform']}",  
    "CLIENT-IP": "@{triggerOutputs()['headers']['CLIENT-IP']}",  
    "DISGUISED-HOST": "@{triggerOutputs()['headers']['DISGUISED-HOST']}",  
    "Content-Length": "@{triggerOutputs()['headers']['Content-Length']}"  
}
```

URI *

Method *

Headers

Content-Type	application/json
DD-API-KEY	a b3

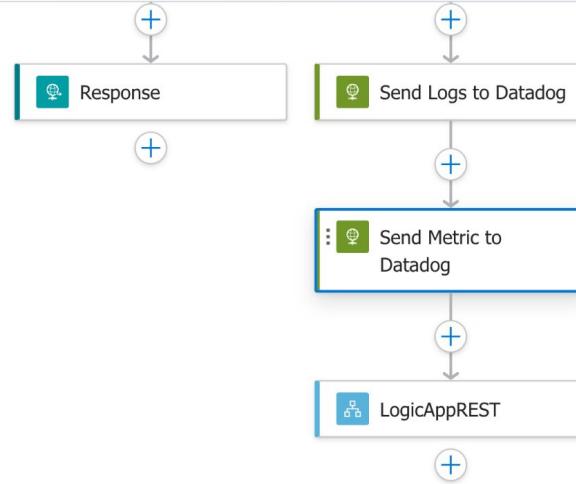
Queries

Enter key	Enter value
-----------	-------------

Body

```
{  
    "host": "Azure",  
    "service": "Logic App",  
    "model": "@{variables('model')}",  
    "word_count": "@{variables('word_count')}",  
    "prompt": "@{variables('prompt')}",  
    "payload": "@{variables('payload')}",  
    "Host": "@{triggerOutputs()['headers']['Host']}",  
    "User-Agent": "@{triggerOutputs()['headers']['User-Agent']}",  
    "Origin": "@{triggerOutputs()['headers']['Origin']}",  
    "sec-ch-ua": "@{triggerOutputs()['headers']['sec-ch-ua']}",  
    "sec-ch-ua-platform": "@{triggerOutputs()['headers']['sec-ch-ua-platform']}"  
}
```

Action: Send Metrics to Datadog



https://api.datadoghq.com/api/v2/series

Method *
POST

Headers

Content-Type	application/json
DD-API-KEY	2e6! 01

Queries

Enter key	Enter value
-----------	-------------

Body

```
{
  "series": [
    {
      "metric": "logicapp.openai.ratelimit.limit.requests",
      "type": 3,
      "points": [
        {
          "timestamp": "{{epoch_timestamp}}",
          "value": {{int(...)}}
        }
      ],
      "resources": [
        {
          "id": "12345"
        }
      ]
    }
  ]
}
```

How to send metrics to Datadog from Azure Logic Apps

The screenshot shows a web browser displaying the Datadog Metrics API documentation. The URL in the address bar is docs.datadoghq.com/api/latest/metrics/. The page has a purple header with the text "Read the 2023 State of Cloud Security Study!". Below the header, there's a navigation bar with links for PRODUCT, CUSTOMERS, PRICING, and SOLUTIONS. The main content area features the Datadog logo (a dog icon) and navigation links for ABOUT, BLOG, and LOGIN, along with a "GET STARTED FREE" button. On the left side, there's a sidebar with a search bar and a list of API endpoints under the "Metrics" heading. The main content area starts with a section about the metrics endpoint, followed by a note about graphing, and ends with a note about managing tags.

PRODUCT CUSTOMERS PRICING SOLUTIONS

ABOUT BLOG LOGIN GET STARTED FREE

Datadog Docs API

DOCS > API REFERENCE > METRICS

LANGUAGE English DATADOG SITE US1

Metrics

Create a tag configuration
Get active metrics list
Query timeseries data across multiple products
Submit distribution points
Submit metrics
Get metric metadata
List tag configuration by name
Query scalar data across multiple products
Edit metric metadata
Update a tag configuration
Delete a tag configuration

The metrics endpoint allows you to:

- Post metrics data so it can be graphed on Datadog's dashboards
- Query metrics from any time period
- Modify tag configurations for metrics
- View tags and volumes for metrics

Note: A graph can only contain a set number of points and as the timeframe over which a metric is viewed increases, aggregation between points occurs to stay below that set number.

The Post, Patch, and Delete `manage_tags` API methods can only be performed by a user who has the `Manage Tags for Metrics` permission.

See the [Metrics page](#) for more information.

Create a tag configuration

Action:
Send Metrics
to Datadog

Metrics API Endpoint URL:

<https://api.datadoghq.com/api/v2/series>

```
{
  "series": [
    {
      "metric": "logicapp.openai.ratelimit.limit.requests",
      "type": 3,
      "points": [
        {
          "timestamp": "@{variables('epoch_timestamp')}",
          "value": "@{int(outputs('OpenAI')['headers']['x-ratelimit-limit-requests'])}"
        }
      ],
      "resources": [
        {
          "name": "generativebrief",
          "type": "logicapp"
        }
      ],
      "tags": [
        "@{outputs('OpenAI')['headers']['openai-model']}",
        "openai-call"
      ]
    },
    {
      "metric": "logicapp.openai.ratelimit.limit.tokens",
      "type": 3,
      "points": [
        {
          "timestamp": "@{variables('epoch_timestamp')}",
          "value": "@{int(outputs('OpenAI')['headers']['x-ratelimit-limit-tokens'])}"
        }
      ],
      "resources": [
        {
          "name": "generativebrief",
          "type": "logicapp"
        }
      ],
      "tags": [
        "@{outputs('OpenAI')['headers']['openai-model']}",
        "openai-call"
      ]
    }
  ]
}
```

Action: Send Metrics to Datadog

Part 1

https://api.datadoghq.com/api/v2/series

Method *

Headers

Content-Type	application/json
DD-API-KEY	2e65c941d9c6e600656e3b0a7a2a12 01

Queries

Enter key	Enter value
-----------	-------------

Body

```
{
  "series": [
    {
      "metric": "logicapp.openai.ratelimit.limit.requests",
      "type": 3,
      "points": [
        {
          "timestamp": "{epoch_timestamp}",
          "value": "int(..)"
        }
      ],
      "resources": [
        {
          "name": "generativebrief",
          "type": "logicapp"
        }
      ],
      "tags": [
        "@{outputs('OpenAI')['headers']['openai-model']}",
        "openai-call"
      ]
    }
  ]
}
```

```
{
  "metric": "logicapp.openai.ratelimit.remaining.requests",
  "type": 3,
  "points": [
    {
      "timestamp": "@{variables('epoch_timestamp')}",
      "value": "@{int(outputs('OpenAI')['headers']['x-ratelimit-remaining-requests'])}"
    }
  ],
  "resources": [
    {
      "name": "generativebrief",
      "type": "logicapp"
    }
  ],
  "tags": [
    "@{outputs('OpenAI')['headers']['openai-model']}",
    "openai-call"
  ]
},
{
  "metric": "logicapp.openai.ratelimit.remaining.tokens",
  "type": 3,
  "points": [
    {
      "timestamp": "@{variables('epoch_timestamp')}",
      "value": "@{int(outputs('OpenAI')['headers']['x-ratelimit-remaining-tokens'])}"
    }
  ],
  "resources": [
    {
      "name": "generativebrief",
      "type": "logicapp"
    }
  ],
  "tags": [
    "@{outputs('OpenAI')['headers']['openai-model']}",
    "openai-call"
  ]
}
]
```

Action: Send Metrics to Datadog

Part 2

The screenshot shows a browser-based API client interface. At the top, there is a URL input field containing <https://api.datadoghq.com/api/v2/series>. Below it, a dropdown menu shows the method as **POST**. Under the **Headers** section, there are two entries: **Content-Type** set to **application/json**, and **DD-API-KEY** set to **01**. In the **Queries** section, there are two fields: **Enter key** and **Enter value**. The **Body** section contains the following JSON payload:

```
{
  "series": [
    {
      "metric": "logicapp.openai.ratelimit.limit.requests",
      "type": 3,
      "points": [
        {
          "timestamp": "@{epoch_timestamp} epoch_timestamp * ",
          "value": "f@ int(...) * "
        }
      ],
      "resources": [
        {
          "name": "generativebrief",
          "type": "logicapp"
        }
      ]
    },
    {
      "metric": "logicapp.openai.ratelimit.limit.tokens",
      "type": 3,
      "points": [
        {
          "timestamp": "@{epoch_timestamp} epoch_timestamp * ",
          "value": "f@ int(...) * "
        }
      ],
      "resources": [
        {
          "name": "generativebrief",
          "type": "logicapp"
        }
      ]
    }
  ]
}
```

How to add real user monitoring (RUM) to a web application

The screenshot shows a browser window with the URL docs.datadoghq.com/real_user_monitoring/. The page title is "Read the 2023 State of Cloud Security Study!". On the left, there's a sidebar with navigation links for "PRODUCT", "CUSTOMERS", "PRICING", "Datadog Docs", "Search documentation...", "Real User Monitoring" (which is expanded to show "Browser Monitoring", "Mobile and TV Monitoring", "Product Analytics", "Platform", "Session Replay", "Exploring RUM Data", "Feature Flag Tracking", "Error Tracking", "Guides", "Data Security"), and "Synthetic Monitoring" and "Continuous Testing". The main content area shows a code editor with the file "JS App.js". The code is as follows:

```
src > JS App.js > App
1 import { datadogRum } from '@datadog/browser-rum';
2 import { datadogLogs } from '@datadog/browser-logs';
3 import React, { useState, useEffect } from "react";
4 import "./App.css";
5 import ResponseComponent from "./ResponseComponent";
6
7 datadogLogs.init({
8   clientToken: process.env.REACT_APP_DD_CLIENT_TOKEN,
9   site: process.env.REACT_APP_DD_SITE,
10  forwardErrorsToLogs: true,
11  sessionSampleRate: 100,
12 })
13
14 datadogRum.init({
15   applicationId: process.env.REACT_APP_DD_APPLICATION_ID,
16   clientToken: process.env.REACT_APP_DD_CLIENT_TOKEN,
17   site: process.env.REACT_APP_DD_SITE,
18   service: 'generative-debrief',
19   env: 'azure-production',
20   // Specify a version number to identify the deployed version of your application in Datadog
21   // version: '1.0.0'
22 })
```

How to visualize Azure Logic App, OpenAI, and web app data in Datadog

The screenshot displays two Datadog dashboards side-by-side, illustrating how to visualize data from Azure Logic Apps and OpenAI.

Left Dashboard: Azure Logic App

- Logs:** Shows logs matching "host"github.com service"github-actions"Deployment C...".
- Metric:** Avg run latency: 0.2 seconds.
- Metrics:** Recent, Dashboards, Monitors, Watchdog, Service Mgmt, Infrastructure, APM, Digital Experience, Software Delivery, Security, Metrics, Logs, Integrations, Bits AI, Screen Share, Jason Hand@dat..., Datadog Demo (1...).

Right Dashboard: Jhand - Logic App / Static Web App / ...

- Logs:** Shows logs for the "generative-debrief" service.
- Metric:** Successful runs: 6.
- Metrics:** Recent, Dashboards, Monitors, Watchdog, Service Mgmt, Infrastructure, APM, Digital Experience, Software Delivery, Security, Metrics, Logs, Integrations, Bits AI, Screen Share, Jason Hand@dat..., Datadog Demo (1...).
- Logic App logs:** Shows log entries for the Logic App service.
- Timeline:** Logic App logs over 29.75 hours with 22 logs.

Datadog Dashboards

Dashboard Groups

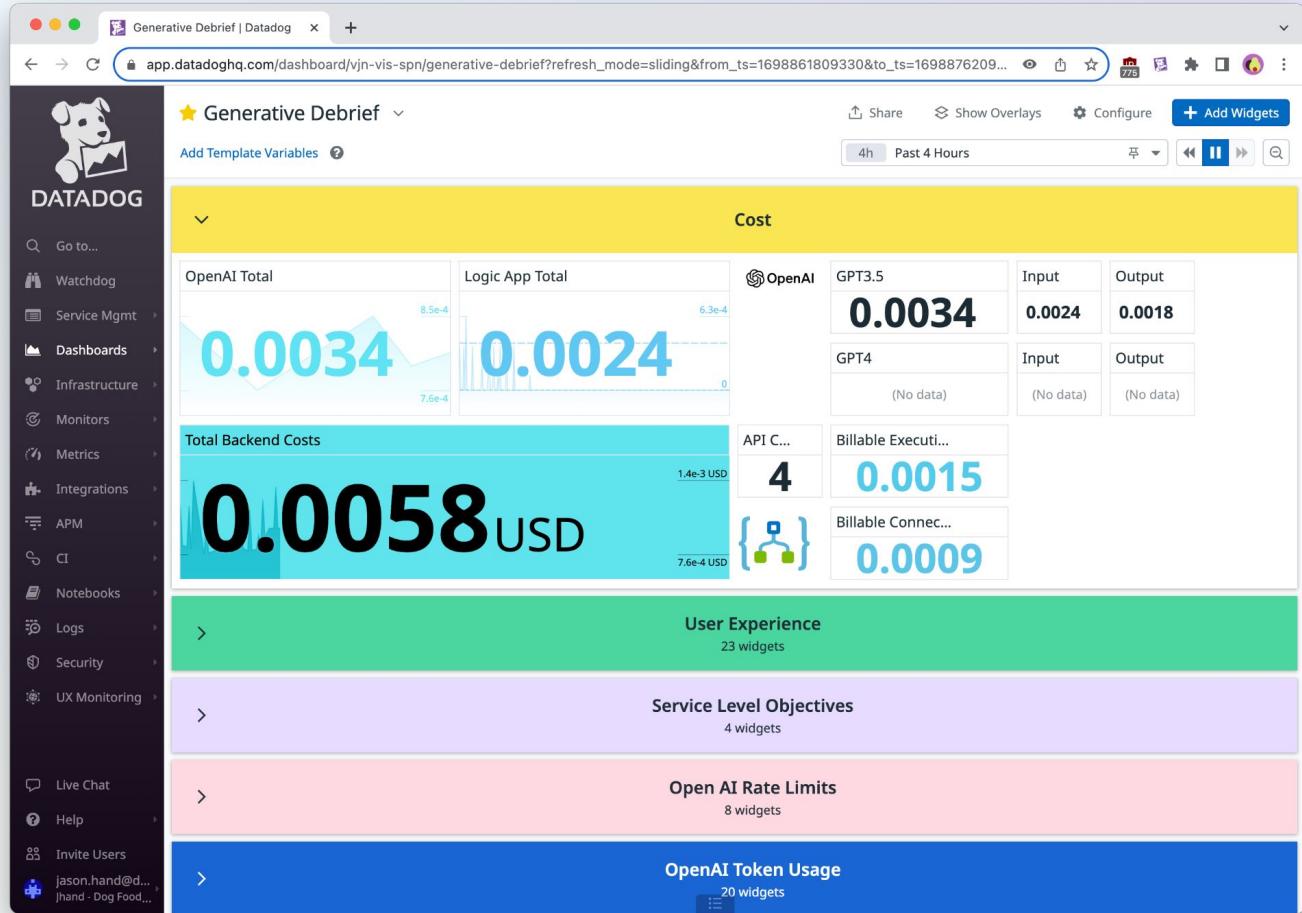
The screenshot shows a Datadog dashboard titled "Generative Debrief". The dashboard interface includes a sidebar with navigation links like Watchdog, Service Mgmt, Dashboards, Infrastructure, Monitors, Metrics, Integrations, APM, CI, Notebooks, Logs, Security, UX Monitoring, Live Chat, Help, Invite Users, and a user profile for jason.hand@... . The main area displays eight colored cards representing different service groups:

- Cost (Yellow card, 14 widgets)
- User Experience (Green card, 23 widgets)
- Service Level Objectives (Purple card, 4 widgets)
- Open AI Rate Limits (Pink card, 8 widgets)
- OpenAI Token Usage (Blue card, 20 widgets)
- Geo Data (Red card, 3 widgets)
- CI/CD - GitHub Actions (Orange card, 2 widgets)
- Logic App (Light Blue card, 12 widgets)

At the top of the dashboard, there are buttons for Share, Show Overlays, Configure, and Add Widgets, along with a time range selector set to "4h Past 4 Hours".

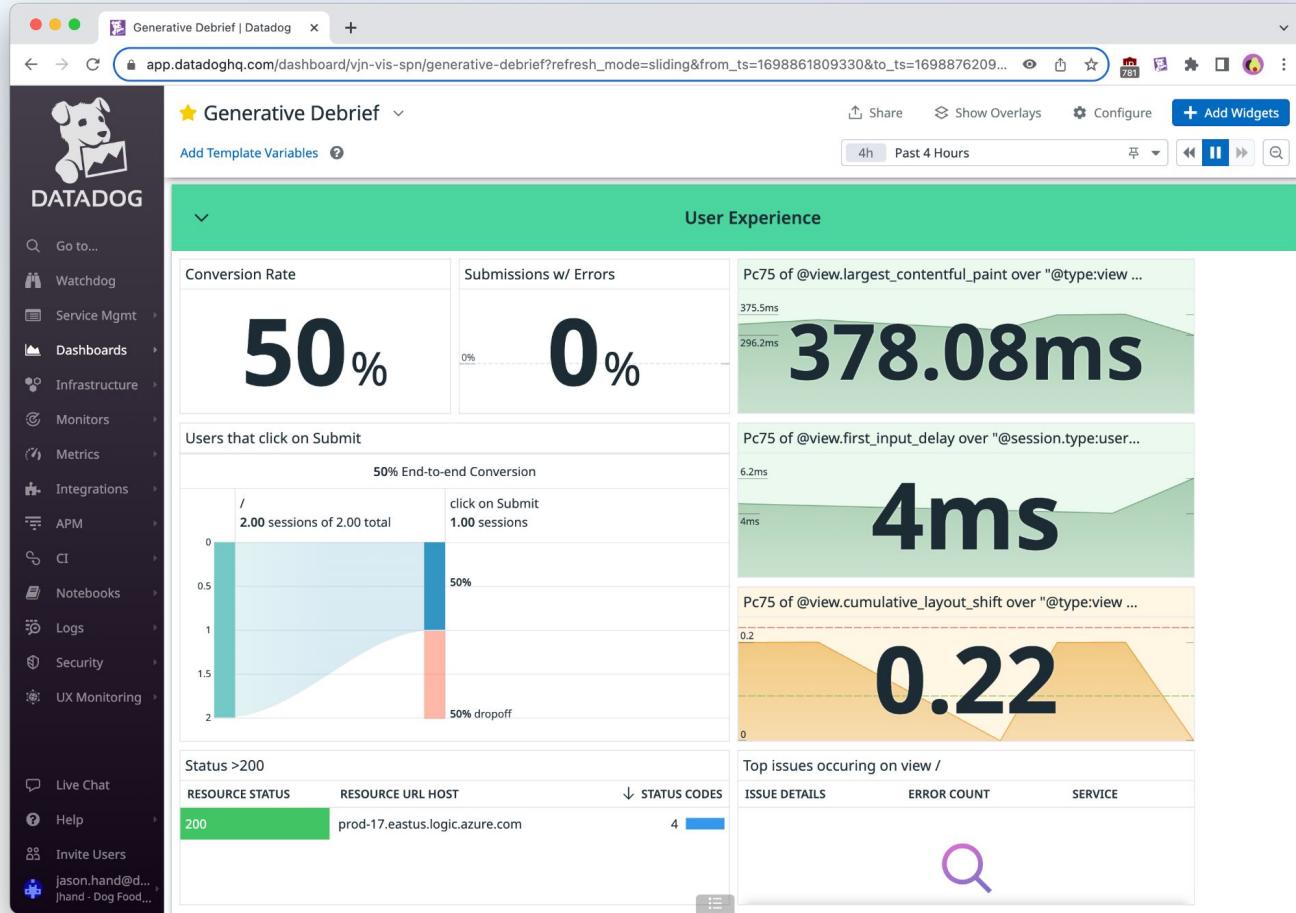
- Cost
- User Experience
- Service Level Objectives
- OpenAI Rate Limits
- OpenAI Token Usage
- Geo Data
- CI/CD - GitHub Actions
- Logic App

Operational Costs



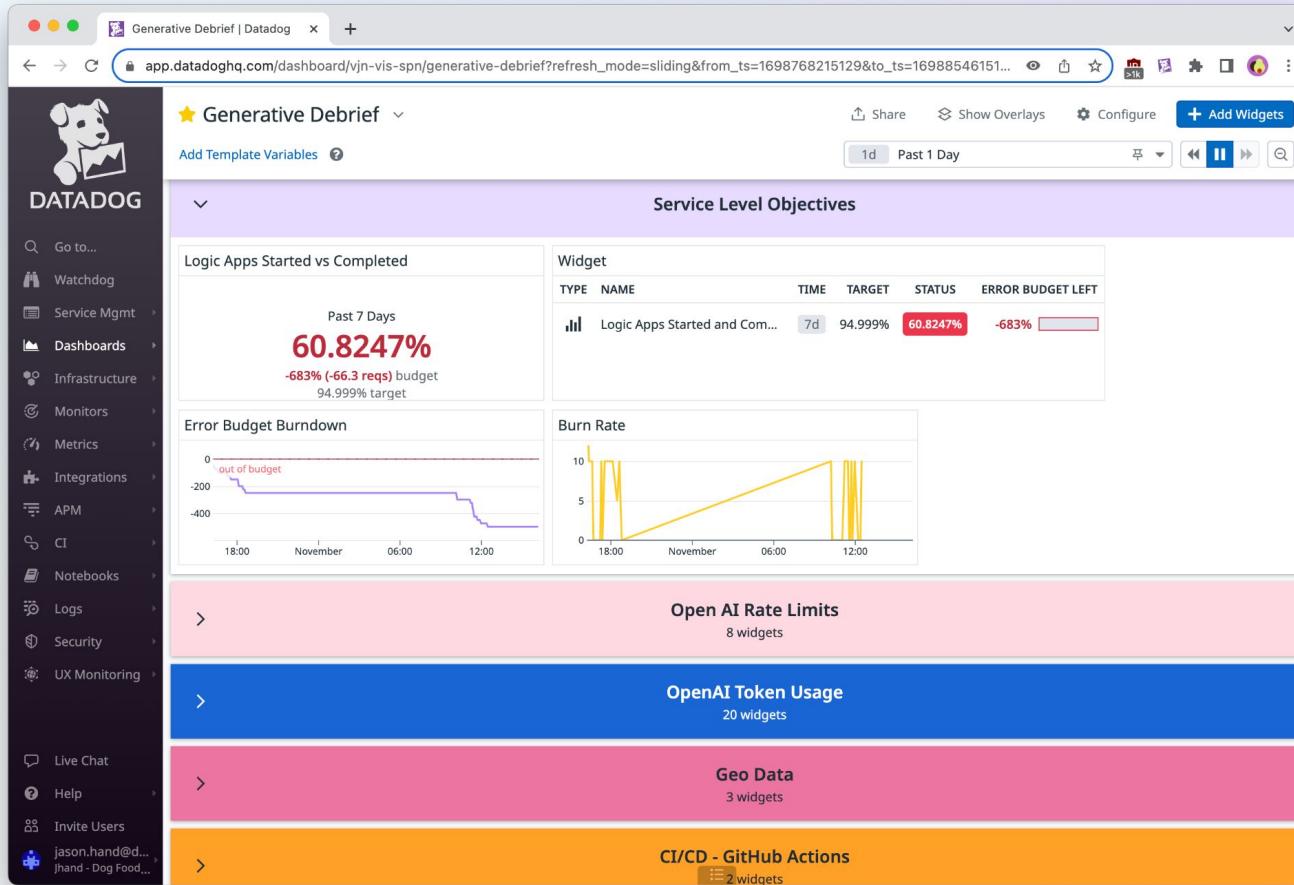
- Total Backend Costs
- OpenAI Total Cost
- Logic App Total Cost
- Breakdown of model costs
- Breakdown of billable Logic App executions

User Experience



- Conversion Rate
- Submission Errors
- “Submit” count
- Pc75 - content pain
- Pc75 - input delay
- Pc75 - cumulative layout shift
- API call responses
- Errors

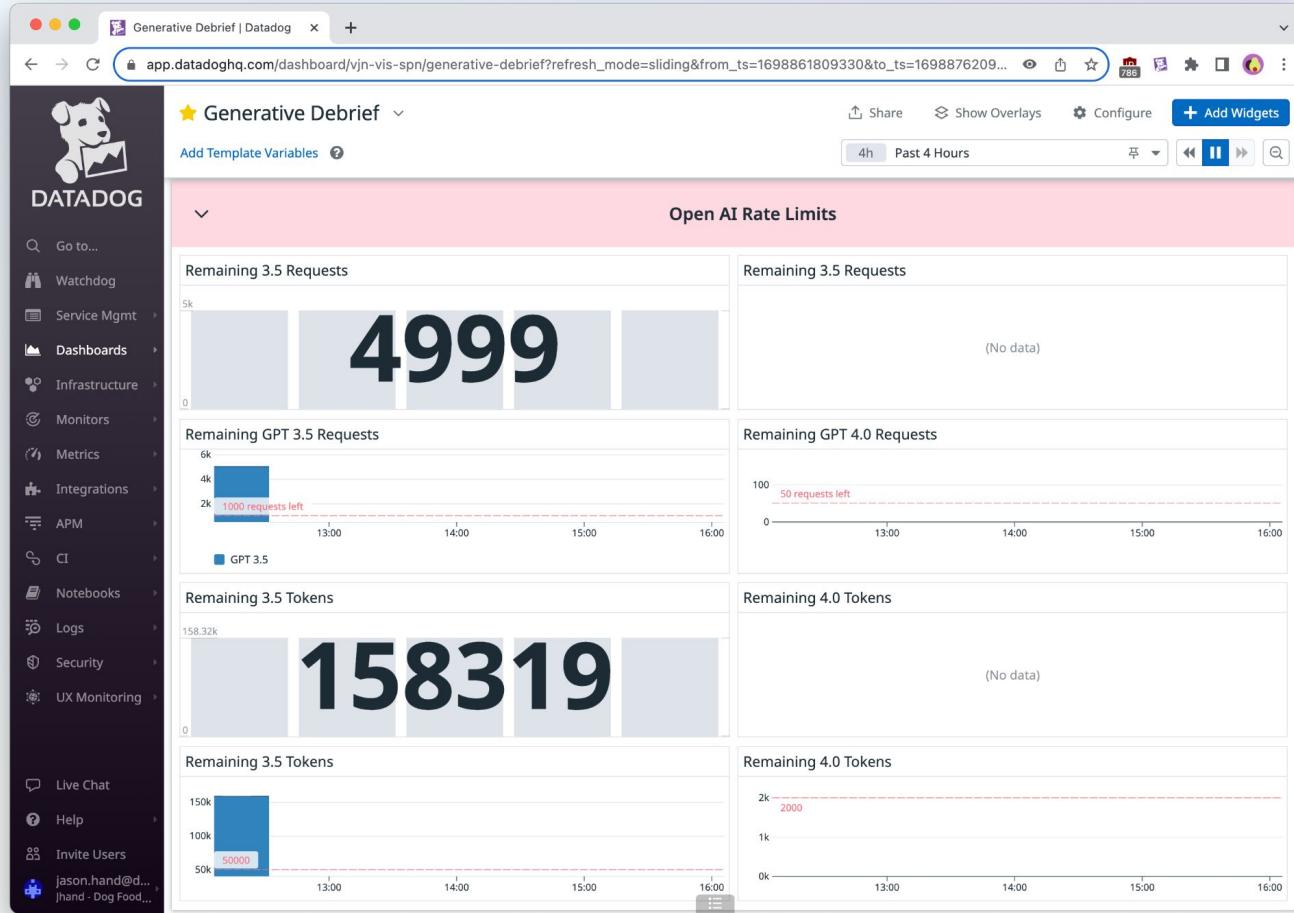
Service Level Objectives



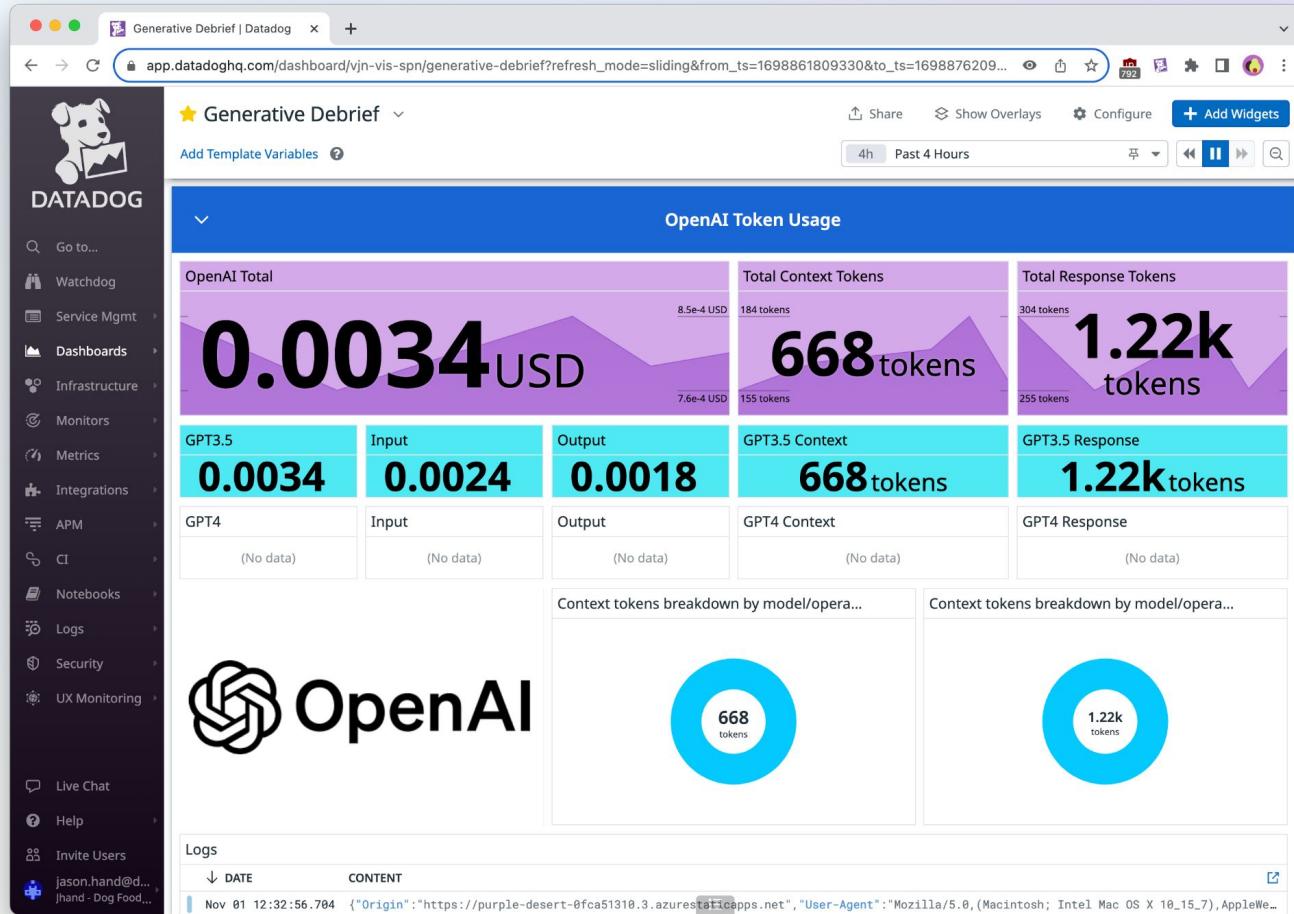
- Error Budget
- Burndown
- Burn Rate

OpenAI Rate Limits

- Remaining Tokens
- Remaining Requests



OpenAI token Usage



- Total cost
- Total context tokens
- Total response tokens
- Breakdown of model usage

Geo data

- RUM - page views
- RUM - location

The screenshot shows a Datadog dashboard titled "Generative Debrief". The dashboard includes a sidebar with navigation links like "Go to...", "Watchdog", "Service Mgmt", "Dashboards", "Infrastructure", "Monitors", "Metrics", "Integrations", "APM", "CI", "Notebooks", "Logs", "Security", "UX Monitoring", "Live Chat", "Help", "Invite Users", and a user profile for "jason.hand@...". The main content area has a header "Geo Data" with a "Views by Country" map showing the United States highlighted. Below this are sections for "RUM - Generative Debrief - G..." (a donut chart showing 100% for one category), "Countries with most page vie...", and "Views by Country" (a world map). Further down are sections for "CI/CD - GitHub Actions" (2 widgets) and "Logic App" (12 widgets). At the bottom, there's a purple box with a plus sign and the text "Add Widgets or Powerpacks". The URL in the browser bar is app.datadoghq.com/dashboard/vjn-vis-spn/generative-debrief?refresh_mode=sliding&from_ts=1698861809330&to_ts=1698876209....

CI/CD - GitHub Actions

- Logs - “Deployment Complete”

The screenshot shows a Datadog dashboard titled "Generative Debrief". The left sidebar contains navigation links for Go to..., Watchdog, Service Mgmt, Dashboards, Infrastructure, Monitors, Metrics, Integrations, APM, CI, Notebooks, Logs, Security, UX Monitoring, Live Chat, Help, Invite Users, and a user profile for jason.hand@... .

The main dashboard area has a title "CI/CD - GitHub Actions". It displays a table of logs and a chart of log transactions.

Logs Table:

DATE	HOST	SERVICE	CONTENT
Oct 31 10:19:45.582	github.com	github-actions	Deployment Complete :)

Log Transactions List:

SERVICE	TIMELINE	DURATION	MAX LOG SEVERITY	COUNT
github-actions	Timeline bar from Oct 31 10:19:45.582 to Oct 31 10:19:45.582	2.77m	INFO	977 logs

Widgets:

- OpenAI Token Usage (20 widgets)
- Geo Data (3 widgets)
- Logic App (12 widgets)
- Add Widgets or Powerpacks

At the bottom, there is a footer with copyright information: Copyright Datadog, Inc. 2023 - 35.22635834 - Master Subscription Agreement - Privacy Policy - Cookie Policy - Datadog Status → All Systems Operational.

Logic App

The screenshot shows the Datadog Generative Debrief dashboard for a Logic App. The top section displays key performance indicators (KPIs) in large blue numbers:

- Total Cost: 0.0164
- Execution Cost: 0.0158
- Connector Cost: 0.0005

Below these are three cards with time-series charts:

- Avg run latency: 0.02s
- Successful runs: 66 events
- Failed runs: 29 events

The bottom section contains two log tables:

- Logs**: A table with columns DATE and CONTENT, showing log entries from Nov 01 at 12:32 to 12:16. The content includes Origin and User-Agent details.
- Logs**: A table with columns SERVICE, TIMELINE, DURATION, MAX LOG SEVERITY, and COUNT, showing data for the Logic App with a duration of 43.67h and 51 logs.

- Total cost
- Latency
- Successful runs
- Failed runs
- Logs

Takeaways

- Integrations for Low Code tools (like Azure Logic Apps)
- Integrations for Generative tools (such as OpenAI)
- Augment with Logs & Metrics
- Measure customer experience (RUM)
- Visualize, monitor, and alert on what is important
- Continue to learn and experiment



Thank
You

Resources

- [OpenAI Integration](#)
- [OpenAI Integration Blog](#)
- [OpenAI Monitoring](#)
- [Azure Integration](#)
- [Logic App Integration Docs](#)
- [Logic App Integration Blog](#)
- [RUM Documentation](#)
- [Azure OpenAI Integration Docs](#)
- [CI Pipelines](#)
- [Monitor your GitHub Actions workflows with Datadog CI Visibility](#)
- [GitHub Integration](#)