

Implementation Guide

Performance Dashboard on AWS



Performance Dashboard on AWS: Implementation Guide

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Overview	1
Features and benefits	2
Concepts and definitions	3
Architecture overview	4
Architecture diagram	4
AWS Well-Architected	6
Architecture details	7
Web UI	7
Dashboards	7
Admin roles	8
Amazon S3 (frontend)	8
Amazon CloudFront	9
Amazon API Gateway	9
AWS Lambda	9
Amazon DynamoDB	9
Amazon S3 (backend)	10
Amazon Cognito	10
Security Assertion Markup Language (SAML)	10
AWS services in this solution	10
Plan your deployment	12
Cost	12
Sample cost table	12
Security	13
IAM roles	13
AWS WAF	13
Amazon CloudFront	14
Supported AWS Regions	14
Quotas	15
Quotas for AWS services in this solution	15
AWS CloudFormation quotas	15
Deploy the solution	16
AWS CloudFormation template	16
Deployment process overview	17
Step 1: Launch the stack	18

Step 2: Configure user invitation email template (post-deployment)	19
Step 3: Log in to Performance Dashboard on AWS as an initial user	20
Step 4: Add users	21
Step 5: Create topic areas	21
Update the solution	22
Uninstall the solution	23
Using the AWS Management Console	23
Using AWS Command Line Interface	23
Deleting the Amazon S3 buckets	23
Deleting the DynamoDB tables	24
Use the solution	25
Create a new draft dashboard	25
Add content items	25
Add content items with datasets	26
Preview your dashboard	26
Developer guide	27
Source code	27
Integration guide	27
Export dashboards from a deployed Performance Dashboard on AWS instance	27
Import dashboard template into deployed Performance Dashboard on AWS instance	28
Limiting deployed solution access to on-premise networks (IP addresses)	29
Activate Security Assertion Markup Language (SAML) federation	31
Customization guide	32
Change domain names (post-deployment)	32
Change UI text language (during deployment)	33
Customize new user invitation email template (pre-deployment)	33
Change frontend details (post-deployment)	34
Change homepage title and description (post-deployment)	35
Require user credentials for all users	36
Customize application header logo	36
Data Ingestion API	37
Enable the API	40
Obtain the API Key	41
Scaling	42
Monitoring	43
Reference	44

Anonymous data collection	44
Contributors	45
Notices	45
Revisions	46

Solution Overview

Publication date: April 2021 ([last update](#): December 2023)

Performance Dashboard on AWS is an open-source dashboard management solution (*application*) created for public-sector organizations that want to visualize data to track and communicate organizational performance. In contrast with complex data visualization/analysis offerings, Performance Dashboard is a simple and cost-effective data presentation tool that users with minimal design and analytic skills can use to quickly create and publish illustrative dashboards to the web.

Dashboards can be created using a variety of included solution components, such as charts, UI text, and metadata. Both static and dynamic datasets (using the [Data Ingestion API](#)) can be used as data sources to populate dashboards.

You can upload datasets to use with dashboards individually and in bulk through two methods:

- the solution's [web UI](#)
- the solution's [Data Ingestion REST API](#)

This implementation guide provides an overview of the Performance Dashboard on AWS solution, its reference architecture and components, considerations for planning the deployment, and configuration steps for deploying the solution to the Amazon Web Services (AWS) Cloud.

Use this navigation table to quickly find answers to these questions:

If you want to . . .	Read . . .
Know the cost for running this solution.	Cost
Understand the security considerations for this solution.	Security
Know how to plan for quotas for this solution.	Quotas
Know which AWS Regions are supported for this solution.	Supported AWS Regions

If you want to . . .	Read . . .
Deploy the solution.	Deploy the solution

This guide is intended for solution architects, DevOps engineers, business decision makers, engineers, data scientists, and cloud professionals who want to implement Performance Dashboard on AWS in their environment.

Features and benefits

Performance Dashboard on AWS provides the following features:

Creating and publishing dashboards within a pre-designed web UI and workflow

A supportive workflow guides users through dashboard creation step-by-step, from creating an initial draft to publishing a finished dashboard.

Minimal costs using serverless AWS technology

Performance Dashboard on AWS utilizes serverless AWS products to keep costs at a minimum. You only incur costs when your deployed solution is actively being used.

Adding and changing domain names

Configure a custom domain name for your deployed instance of Performance Dashboard on AWS.

Add custom logo and UI text

Insert your organization's custom logo into the Performance Dashboard on AWS UI.

Customizable email templates for user invites

When you create a new user in Performance Dashboard on AWS, an email is sent to invite that new user to login to the application. The solution provides a default invite email template, which you can customize to suit your organization.

Multiple UI language options

Choose between English, Spanish, and Portuguese for the solution's starter UI content when deploying Performance Dashboard on AWS.

Federated identity management

Performance Dashboard on AWS leverages Security Assertion Markup Language (SAML) 2.0-compliant identity providers.

Concepts and definitions

This section describes key concepts and defines terminology specific to this solution:

Admin

An admin role that manages users and makes sitewide setting updates, such as logo or branding (colors) updates.

Editor

An admin role that creates dashboards and publishes them for website users to view.

Dynamic dataset

Datasets containing data that continues to update. These datasets update the solution data visualization automatically as new data is added through the [Data Ingestion API](#). Dynamic datasets are useful for visualizing streaming data that is updated frequently and doesn't require manual intervention.

Static dataset

Datasets containing unchanging data. These datasets have to be updated manually to change a dashboard's rendered data.

Topic areas

Categories used in this solution to organize and group dashboards.

For a general reference of AWS terms, see the [AWS glossary](#) in the AWS General Reference.

Architecture overview

This section provides a reference implementation architecture diagram for the components deployed with this solution.

Architecture diagram

Deploying this solution with the default parameters deploys the following components in your AWS account.

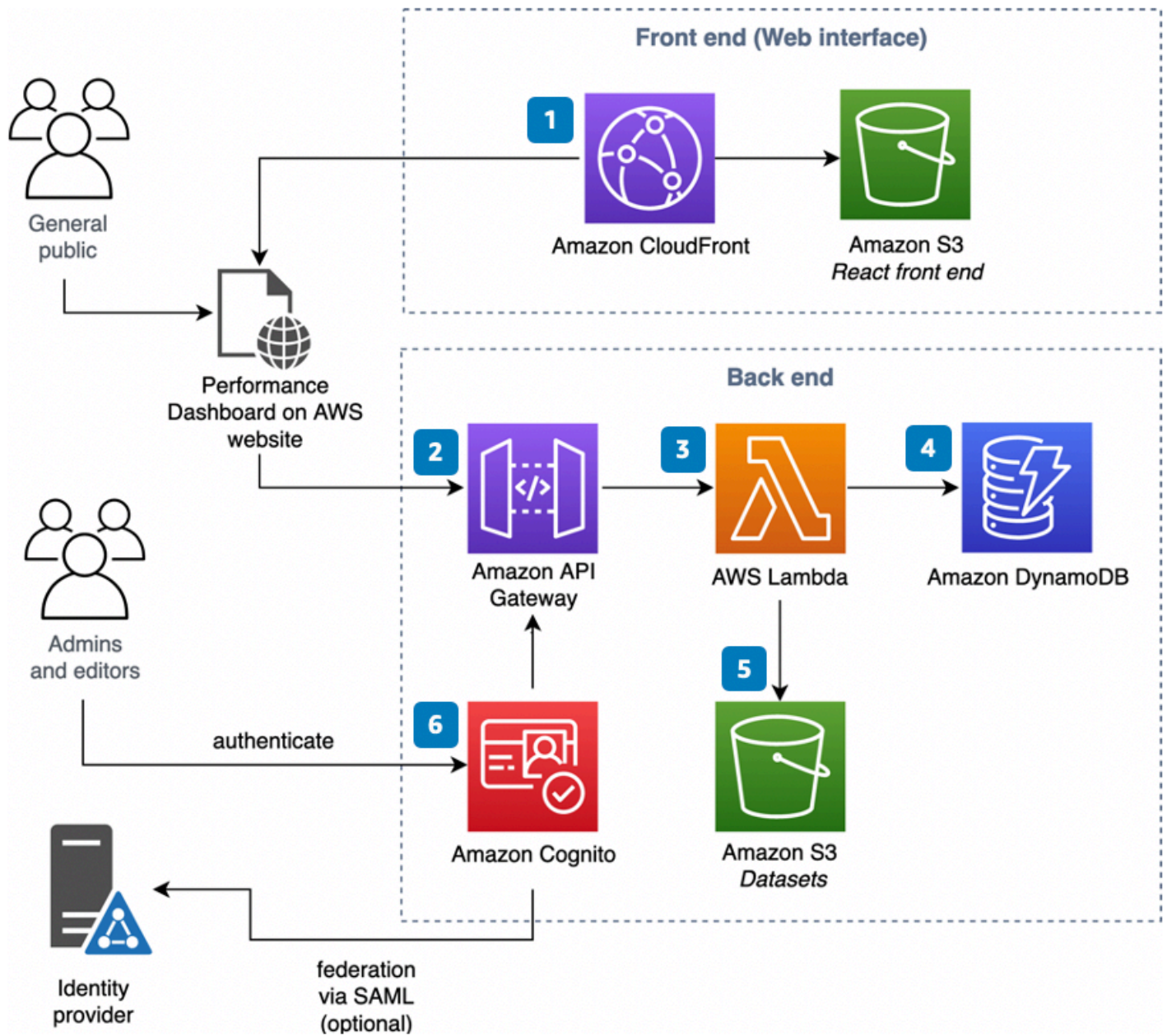


Figure 1: Performance Dashboard on AWS architecture

The high-level process flow for the solution components deployed with the AWS CloudFormation template is as follows:

1. An [Amazon CloudFront](#) distribution and [Amazon Simple Storage Service \(Amazon S3\)](#) bucket host and serve the web frontend, which includes HTML pages, CSS stylesheets, JavaScript code, and solution-associated digital assets.

2. An [Amazon API Gateway](#) resource hosts the APIs called by the web frontend to access the AWS Lambda functions that perform application functions.
3. [AWS Lambda](#) functions in this solution, powered by Node.js, handle and access data related to creating and serving dashboards.
4. An [Amazon DynamoDB](#) database stores metadata about the dashboard and related datasets.
5. An [Amazon S3](#) bucket located in the solution's backend stores the datasets used with the dashboards.
6. An [Amazon Cognito](#) user pool stores information on the solution users' identities, such as role type and permissions for private dashboard view access.

AWS Well-Architected design considerations

This solution was designed with best practices from the [AWS Well-Architected Framework](#) which helps customers design and operate reliable, secure, efficient, and cost-effective workloads in the cloud. Among the AWS Well-Architected lenses, the [Serverless Applications Lens](#) was used to create Performance Dashboard on AWS.

Architecture details

This section describes the components and AWS services that make up this solution and the architecture details on how these components work together.

Web UI

Performance Dashboard on AWS includes a pre-built [React.js](#) web UI for creating and sharing dashboards. The UI functions as both as regular and admin. website, depending on a user's role type.

Dashboards

You can create both public and private dashboards within Performance Dashboard on AWS. Public dashboards are visible to all solution users, while private dashboards are permissions-based. Solution users can designate which users can view their private dashboards.

Dashboards created in the UI can undergo the following states:

- **Draft**

The **Drafts** subsection includes dashboards that were previously created but not yet published.

- **Publish Pending**

The **Publish** queue subsection includes dashboards that have begun the publishing process but are not yet published.

- **Published**

The **Published** subsection includes published dashboards that are currently visible on the UI homepage.

- **Archived**

The **Archived** subsection includes previously published dashboards that are no longer visible on the UI homepage. You can re-publish archived dashboards to make them viewable.

[← All Dashboards](#)

Government Notification System

Technology | Last updated 2021-04-14 05:27

Notifications by Type

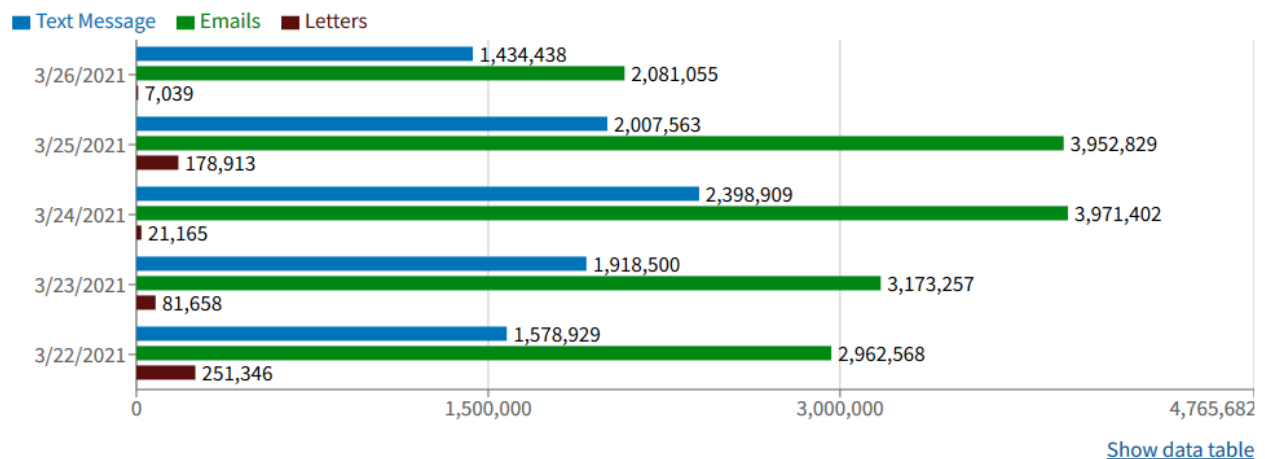


Figure 2: Sample dashboard created in Performance Dashboard on AWS

Admin roles

By default, the Performance Dashboard on AWS website is public, meaning users do not need to log in to view dashboards. However, users wanting to log in as an admin must undergo Amazon Cognito user authentication.

The admin portal supports two admin role types: **admin** and **editor**. The admin role manages users and makes sitewide setting updates, such as logo or branding (colors). The editor role creates dashboards and publishes them for website users to view. Based on their assigned role, admins and editors are presented with different features upon logging in.

Amazon S3 (frontend)

Performance Dashboard on AWS utilizes S3 to host and serve the web UI for all users. The frontend S3 bucket also stores the solution's default frontend components like photos and logos.

Amazon CloudFront

Performance Dashboard on AWS utilizes CloudFront to speed up distribution of static and dynamic web content, such as .html, .css, .js, and image files, to users. CloudFront delivers your content through a worldwide network of data centers called edge locations.

When a user requests Performance Dashboard on AWS content that you are serving with CloudFront, the request is routed to the edge location that provides the lowest latency (time delay), so that content is delivered with the best possible performance.

- If the content is already in the edge location with the lowest latency, CloudFront delivers it immediately.
- If the content is not in that edge location, CloudFront retrieves it from an origin that you've defined—such as an Amazon S3 bucket, that you have identified as the source for the definitive version of your content.

Amazon API Gateway

Performance Dashboard on AWS uses API Gateway to transfer frontend user data to solution's backend for AWS Lambda and Amazon DynamoDB to handle serverless function and data storage purposes.

AWS Lambda

Performance Dashboard on AWS uses Lambda to execute serverless [Node.js](#) functions on data received and stored from API Gateway and DynamoDB.

Amazon DynamoDB

Performance Dashboard on AWS uses a DynamoDB table to store user data and metadata related to dashboards and datasets.

The solution creates two DynamoDB tables upon initial deployment: `Main` and `AuditTrail`. The Lambda functions responsible for creating, editing, and serving dashboards, access the `Main` table to get and store dashboard metadata. Metadata includes dashboard name, version, content items, and data sets used. To keep an audit trail of actions taken by users on dashboard creation, editing, and publishing, events are logged in the `AuditTrail` table.

Amazon S3 (backend)

Performance Dashboard on AWS utilizes S3 in its backend (located in `backend/src/lib/services/s3.ts`) to store and retrieve datasets used to create dashboards.

Amazon Cognito

Performance Dashboard on AWS uses Amazon Cognito to authenticate application users and admins. When this solution is first deployed, it creates an initial admin user in the Cognito user pool. This admin can create subsequent admin/editor users in Cognito to further define the solution's user pool. An admin can assign the admin or editor role to other users, in addition to being able to delete users.

When a user is created, Amazon Cognito sends an invite email with the login credentials. You can resend the invitation email if the original invite was lost.

Security Assertion Markup Language (SAML)

This solution supports authentication for users in a SAML identity provider (IdP). The Cognito user pool can be configured for SAML 2.0 federation. In that scenario, users authenticate to the SAML identity provider when signing in. The Cognito user pool handles the SAML assertion and returns tokens based on the SAML user identity.

To enact this, configure Amazon Cognito to support federation with that IdP. A user logging in to Performance Dashboard on AWS will then authenticate against the IdP. That user initially will not have the admin or editor roles, and they will see a warning advising them to request access. They can then request that the Performance Dashboard on AWS admin grant them the proper role.

AWS services in this solution

AWS service	Description
Amazon Simple Storage Solution (Amazon S3)	Core. This solution uses Amazon S3 for frontend and backend storage purposes.
Amazon CloudFront	Core. This solution uses CloudFront to cache and serve the web UI to solution users.

AWS service	Description
Amazon DynamoDB	Core. This solution uses DynamoDB to create a table that stores dashboards and the datasets from which they source information.
Amazon API Gateway	Core. This solution uses API Gateway to host APIs called by the web frontend to access the AWS Lambda functions that perform backend application operations.
AWS Lambda	Core. This solution uses serverless Lambda functions, using a Node.js runtime, to perform common operations like accessing data for creating and displaying dashboards.
Amazon Cognito	Core. This solution uses Cognito user pools to store user and admin identities.
AWS X-Ray	Supporting. This solution utilizes AWS X-Ray in its backend to record and analyze information on data requests that occur when the solution is deployed.
Amazon CloudWatch	Supporting. This solution can be extended with CloudWatch to collect and visualize real-time logs, metrics, and event data in automated dashboards. Additionally, you can monitor the deployed solution's resource usage and performance issues.
AWS WAF	Optional. This solution can be extended with AWS WAF to prevent off-premise (IP) network access through the creation of an IP address allowlist that prevents unapproved IP addresses from accessing.

Plan your deployment

Performance Dashboard requires you to have an AWS account before deploying.

Cost

You are responsible for the cost of the AWS services used while running this solution. As of December 2023, the cost for running this solution under [common usage dimensions](#), with default settings in the US East (N. Virginia) Region is approximately **\$38.00 a month**.

See the pricing webpage for each AWS service used in this solution.

We recommend creating a [budget](#) through [AWS Cost Explorer](#) to help manage costs. Prices are subject to change. For full details, see the pricing webpage for each AWS service used in this solution.

Sample cost table

The following table provides a sample cost breakdown for deploying this solution with the default parameters in the US East (N. Virginia) Region for one month.

AWS service	Dimensions	Cost [USD]
Amazon API Gateway	3,000,000 requests from normal and admin. users	\$ 10.50
Amazon CloudFront	15 GB data transfer out for web page requests	\$ 4.75
Amazon S3	For datasets: 500 GB storage, 3 million retrieval, 12,000 store	\$ 12.75
AWS Lambda	3,000,000 requests from normal and admin. users	\$ 0.50
Amazon DynamoDB	5 GB storage of dashboard metadata	\$ 6.00

AWS service	Dimensions	Cost [USD]
Amazon CloudWatch	7 GB ingested for logging requests	\$ 3.55

Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This [shared responsibility model](#) reduces your operational burden because AWS operates, manages, and controls the components including the host operating system, the virtualization layer, and the physical security of the facilities in which the services operate. For more information about AWS security, visit [AWS Cloud Security](#).

IAM roles

AWS Identity and Access Management (IAM) roles allow customers to assign granular access policies and permissions to services and users on the AWS Cloud. This solution creates IAM roles that grant the solution's AWS Lambda functions access to create Regional resources, such as:

- An IAM role used by the Lambda function that enables the APIs to read/write data in S3 buckets and DynamoDB tables
- An IAM role used by code that runs in the browser to access the data object in S3 buckets used to render charts

AWS WAF

Optionally, this solution deploys [AWS WAF](#), a web application firewall that helps protect the solution against common web exploits that might affect availability, compromise security, or consume excessive resources. AWS WAF provides control over how traffic reaches the solution, such as using security rules that block requests that do not originate in an allowlist of CIDR IP range.

For example, we recommend that you use AWS WAF to limit access to the **/admin** portion of your Performance Dashboard on AWS instance. Use the CloudFormation template in our [GitHub repo](#) to configure AWS WAF to limit access to an allowlist of CIDR ranges.

Amazon CloudFront

This solution deploys a web console [hosted](#) in an Amazon Simple Storage Service (Amazon S3) bucket. To help reduce latency and improve security, this solution includes an Amazon CloudFront distribution with an origin access identity, which is a CloudFront user that provides public access to the solution's website bucket contents. For more information, see [Restricting Access to Amazon S3 Content by Using an Origin Access Identity](#) in the *Amazon CloudFront Developer Guide*.

Supported AWS Regions

This solution uses the Amazon Cognito, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Amazon Cognito is available. For the most current availability of AWS services by Region, see the [AWS Regional Services List](#).

Performance Dashboard on AWS is supported in the following AWS Regions:

Region name	
US East (Ohio)	Asia Pacific (Tokyo)
US East (N. Virginia)	Europe (Paris)
US West (N. California)	Europe (Stockholm)
US West (Oregon)	Europe (Frankfurt)
Africa (Cape Town)	Europe (Milan)
Asia Pacific (Mumbai)	South America (São Paulo)
Asia Pacific (Seoul)	Canada (Central)
Asia Pacific (Singapore)	Middle East (Bahrain)
Asia Pacific (Sydney)	Europe (Ireland)

Quotas

Service quotas, also referred to as limits, are the maximum number of service resources or operations for your AWS account.

Quotas for AWS services in this solution

Make sure you have sufficient quota for each of the [services implemented in this solution](#). For more information, see [AWS service quotas](#).

Use the following links to go to the page for that service. To view the service quotas for all AWS services in the documentation without switching pages, view the information in the [Service endpoints and quotas](#) page in the PDF instead.

AWS CloudFormation quotas

Your AWS account has AWS CloudFormation quotas that you should be aware of when [launching the stack](#) in this solution. By understanding these quotas, you can avoid limitation errors that would prevent you from deploying this solution successfully. For more information, see [AWS CloudFormation quotas](#) in the *AWS CloudFormation User's Guide*.

Deploy the solution

This solution uses [AWS CloudFormation templates and stacks](#) to automate its deployment. The CloudFormation template(s) describes the AWS resources included in this solution and their properties. The CloudFormation stack provisions the resources that are described in the template(s).

AWS CloudFormation template

You can download the AWS CloudFormation template for this solution before deploying it.

[View template](#)

performance-dashboard.template - Use this template to launch the solution and all associated components. The default configuration deploys authorization, backend, Lambda@Edge, frontend, operations, and dashboard example resources. However, you can customize the template to meet your specific needs.

Note

AWS CloudFormation resources are created from AWS Cloud Development Kit (AWS CDK) constructs.

Before you launch the solution, review the [cost](#), [architecture](#), [security](#), and other considerations discussed earlier in this guide.

Important

This solution includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. AWS owns the data gathered through this survey. Data collection is subject to the [AWS Privacy Policy](#).

To opt out of this feature, download the template, modify the AWS CloudFormation mapping section, and then use the AWS CloudFormation console to upload your updated

template and deploy the solution. For more information, see the [Anonymous data collection](#) section of this guide.

Deployment process overview

Before you launch the solution, review the cost, architecture, security, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the solution into your account.

Time to deploy: Approximately 30 minutes

Note

If you have previously deployed this solution, see [Update the solution](#) for update instructions.

[Step 1. Launch the stack](#)

- Launch the AWS CloudFormation template into your AWS account.
- Enter values for required parameters: **AdminEmail** and **ExampleLanguage**.
- Review the other template parameters, and adjust if necessary.

[Step 2. Configure email invitation template](#)

- Update the email template used to invite users to login to Performance Dashboard on AWS.

[Step 3. Log in to Performance Dashboard on AWS](#)

- Log in to Performance Dashboard on AWS using the login information provided in the invitation email.

[Step 4. Add users](#)

- Add users who will create and publish dashboards.

[Step 5. Create topic areas](#)

- Create topic areas to organize and group your dashboards.

Step 1: Launch the stack

Follow the step-by-step instructions in this section to configure and deploy the solution into your account. You must have [access](#) to an AWS account with permission to deploy resources before launching the stack.

Time to deploy: Approximately 30 minutes

1. A blue rectangular button with rounded corners containing the text "Launch solution" in white.

Sign

in to the AWS Management Console and select the button to launch the performance-dashboard AWS CloudFormation template.

2. The template launches in the US East (N. Virginia) Region by default. To launch the solution in a different AWS Region, use the Region selector in the console navigation bar.

Note

This solution uses the Amazon Cognito, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Amazon Cognito is available. For the most current availability by Region, see the [AWS Regional Services List](#).

3. On the **Create stack** page, verify that the correct template URL is in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, see [IAM and STS Limits](#) in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for this solution template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
AdminEmail	<Requires input>	The email address of the user who will initially administer the solution, and a login will be created for this user. An email inviting the user to sign in will be sent to the provided email address.
ExampleLanguage	<Requires input>	The language options for the Performance Dashboard on AWS UI: English, Spanish, or Portuguese.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a CREATE_COMPLETE status in approximately 30 minutes.

Step 2: Configure user invitation email template (post-deployment)

When you create a new user, an email invitation is sent to the user, inviting them to sign in to the dashboard. Customize the default invite email template to suit your organization. Use the following procedure to customize the email template.

1. Sign in to the [Amazon Cognito console](#).
2. Choose **User pools**.
3. Select the user pool created for Performance Dashboard on AWS.

4. In the navigation tab, choose **Messaging**.
5. In the **Message templates** section, select **Invitation message** and choose **Edit**.
6. Select the **Email message** text box, then make edits to the placeholder values. Replace the placeholder values enclosed within "[]" with your own:
 - a. [Organization] - replace with your organization's name.
 - b. [Administrator] - replace with the name of person sending the invitation.
 - c. [your domain] - replace with the domain name you're using for Performance Dashboard on AWS. For example, change the placeholder value `https://<your-domain>/admin` to `https://example.com/admin`. There are multiple instances of [your domain].
 - d. **Optional.** Update the logo used in the email template by finding the text `src="https://<your domain>/logo.png"`, and replacing it with the link to your logo. If you want to use the default logo, retrieve the link of that logo by opening a browser to Performance Dashboard on AWS. Select the logo on the public website home page, and copy the link address. Paste that address into the email template to replace the placeholder logo.

Step 3: Log in to Performance Dashboard on AWS as an initial user

After launching the solution, you need to retrieve the Performance Dashboard on AWS deployment URL and log in as a user.

1. Sign in to the [AWS CloudFormation console](#).
2. Select this solution's installation stack.
3. Choose the **Outputs** tab and record the value for **CloudFrontURL**.

It follows the `<name.cloudfront.net>` format, which is the URL of Performance Dashboard on AWS.

4. Enter the URL into your browser and navigate to it.

At this stage, your public homepage will not have any dashboards to display.

5. To create a dashboard, add `/admin` in the web address.
6. Use the username and temporary password from your invitation email to log in to Performance Dashboard on AWS.
7. After logging in, follow the UI prompts to change your password.

Step 4: Add users

Since you are the first user of this deployed solution, you must add admin. and editor roles for users who will use the solution. Follow these steps to add users:

1. Navigate to the deployed solution's admin page.
2. In the top navigation bar, choose **Manage users**.
3. Select **Add user(s)**:
 - a. Enter email addresses for all news users, separated by a comma.
 - b. Choose the **Editor** and **Admin** role you want to assign to the users.
 - c. Select **Add user(s) and send invite**.

The users added will receive an email invitation to log in to Performance Dashboard on AWS.

Step 5: Create topic areas

Before dashboards can be created, you must create [topic areas](#). Topic areas are categories used to organize and group your dashboards within the UI. Follow these steps to create a topic area:

1. Navigate to the Performance Dashboard on AWS admin page.
2. In the top navigation bar, choose **Settings**.
3. In the left navigation menu, choose **Topic areas**.
4. Enter a name for the topic area.
5. Choose **Create**.

Choose a name that reflects how you want to group your dashboards for your organization (for example: "Accounting").

6. **Optional.** Create additional topic areas as needed.

Update the solution

If you have previously deployed the solution, follow this procedure to update the Performance Dashboard on AWS CloudFormation stack to get the latest version of the solution's framework.

1. Sign in to the [AWS CloudFormation console](#), select your existing Performance Dashboard on AWS CloudFormation stack, and select **Update**.
2. Choose **Replace current template**.
3. Under Specify template:
 - a. Select **Amazon S3 URL**.
 - b. Copy the link of the [latest template](#).
 - c. Paste the link in the **Amazon S3 URL** box.
 - d. Verify that the correct template URL shows in the **Amazon S3 URL** text box, and choose **Next**. Choose **Next** again.
4. Under **Parameters**, review the parameters for the template and modify them as necessary. For details about the parameters, see [Step 1. Launch the Stack](#).
5. Choose **Next**.
6. On the **Configure stack options** page, choose **Next**.
7. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
8. Choose **View change set** and verify the changes.
9. Choose **Update stack** to deploy the stack.

You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a UPDATE_COMPLETE status in approximately 30 minutes.

Uninstall the solution

You can uninstall Performance Dashboard on AWS from the AWS Management Console or by using the AWS Command Line Interface. You must manually delete the S3 buckets and DynamoDB tables created by this solution. AWS Solutions Implementations do not automatically delete S3 buckets and DynamoDB tables in case you have stored data to retain.

Using the AWS Management Console

1. Sign in to the AWS CloudFormation console.
2. On the **Stacks** page, select this solution's installation stack.
3. Choose **Delete**.
4. First, delete the nested stacks (opStack, frontendStack, lambdaEdgeStack, backendStack, examplesStack, and authStack).
5. Delete the main stack.

Using AWS Command Line Interface

Determine whether the AWS Command Line Interface (AWS CLI) is available in your environment. For installation instructions, see *What Is the AWS Command Line Interface* in the *AWS CLI User Guide*. After confirming that the AWS CLI is available, run the following command.

```
$ aws cloudformation delete-stack --stack-name <installation-stack-name>
```

Deleting the Amazon S3 buckets

This solution is configured to retain the solution-created Amazon S3 bucket (for deploying in an opt-in Region) if you decide to delete the AWS CloudFormation stack to prevent accidental data loss. After uninstalling the solution, you can manually delete this S3 bucket if you do not need to retain the data. Follow these steps to delete the Amazon S3 bucket.

1. Sign in to the [Amazon S3 console](#).
2. Choose **Buckets** from the left navigation pane.
3. Locate the S3 bucket that starts with the same name as your stack and ends with "datasets."

For example, if you named your stack *performancedash*, look for a S3 bucket named *performancedash-**random-string**-datasets*.

4. **Optional.** Back up the data stored in the bucket before deleting it.
5. Select the S3 bucket and choose **Delete**.

To delete the S3 bucket using AWS CLI, run the following command:

```
$ aws s3 rb s3://<bucket-name> --force
```

Deleting the DynamoDB tables

You will need to manually delete the two tables that contain metadata about your dashboard and datasets.

1. Sign in to the [DynamoDB console](#).
2. Select the table that starts with the same name as your stack, and has the string `AuditTrail` in its name.
3. Choose **Delete table**.
4. Type *delete*, and choose **Delete**.
5. Select the table that starts with the same name as your stack, and has the string `MainTable` in its name.
6. Choose **Delete table**.
7. Type *delete*, and choose **Delete**.

Use the solution

A dashboard consists of customizable content items to illustrate how services are performing. All dashboards, regardless of state, are accessible through the **Dashboards** tab in the navigation bar.

Dashboards

Drafts (3) **Publish queue (0)** **Published (10)** **Archived (27)**

You have access to view, edit, and/or publish the draft dashboards in this table.

Search:

<input type="checkbox"/>	Dashboard name	Topic Area	Last Updated	Created by
<input type="checkbox"/>	Transportation Safety	Transportation	2021-04-15 16:15	
<input type="checkbox"/>	Fire Safety and Emergency Medical Services	Health and Human Services	2021-04-15 16:07	
<input type="checkbox"/>	Mass Transportation Usage in 2020	Transportation	2021-02-18 08:45	

Showing 1-3 of 3

Page 1 of 1 Go

Show 25

[Return to top](#)

Figure 3: Performance Dashboard on AWS Dashboards page

Create a new draft dashboard

1. Choose **Create dashboard**.
2. Enter a name.
3. Select a topic area.
4. **Optional:** Enter a description.
5. Choose **Create**.

Add content items

In the dashboard editor, you can add and manage new content items, reorder content items, and preview your dashboard's presentation. When you create the draft dashboard, it will not initially have content items. Build out your dashboard by adding content items.

1. Choose **+Add content item**.
2. Select **Text**, and choose **Continue**.
3. Enter a text title.
4. Enter text.
5. Choose **Add text**.

Content items display in the order that they were added to the dashboard.

Add content items with datasets

To learn how to build content items with datasets, start with a content item with an example dataset.

1. Choose **+Add content item**.
2. Select **Chart**, and choose **Continue**.
3. Select **Static dataset**, and choose the **How do I format my CSV file?** link.
4. In the new **Formatting CSV file** tab, choose **Download line chart example CSV**.
5. Return to the **Add chart** tab, select a folder to upload the example CSV to, and choose **Continue**.
6. Enter a chart title.
7. Select **Line** as chart type.
8. Choose **Add chart**.

Preview your dashboard

You can preview what your dashboard will look like if published. Upon review, you can then return to the dashboard editor to make any changes.

1. Choose **Preview**.
2. Review the dashboard's chart and text content items.
3. Choose **Close Preview**.

Continue to add content items to the draft dashboard from the dashboard editor until your dashboard is complete.

Developer guide

Source code

Visit our [GitHub repository](#) to download the source files for this solution.

The Performance Dashboard on AWS templates are generated using the [AWS Cloud Development Kit \(AWS CDK\)](#). See the [README.md file](#) for additional information.

Integration guide

Export dashboards from a deployed Performance Dashboard on AWS instance

The exporting step is designed to let you download a template of dashboards from a live instance of the solution. For example, if you configured a dashboard that has all the properties you want, then you can use the **export** tool to create a template for it. The template will be added to the solution's [resources folder](#). After that, you can import that template into another deployed instance.

Exporting dashboards using the CLI

Example:

```
export AWS_PROFILE=<value>
export AWS_REGION=<value>
export MAIN_TABLE=<value>
export DATASETS_BUCKET=<value>
export USER_EMAIL=<value>
npm install
npm run export
```

```
Welcome to Performance Dashboard on AWS Export Tool
What is the template name? ... template
Which dashboard you want to export? ... dashboardId
-----
exported dashboard: {}
```


Import dashboard template into deployed Performance Dashboard on AWS instance

The importing step will deploy an existing dashboard template configuration into a live (deployed) instance of Performance Dashboard on AWS. You can take this action if you have dashboard set-ups that you would like to move into a different solution instance. In this case, you will run the **import** tool to create new dashboards. Using either the CLI or by altering a Lambda function, you can import a Amazon CloudFront template and choose to reuse any topic areas, dashboards, and datasets.

Importing dashboards using the CLI

Example:

```
export AWS_PROFILE=<value>
export AWS_REGION=<value>
export MAIN_TABLE=<value>
export DATASETS_BUCKET=<value>
export EXAMPLES_BUCKET=<value>
export USER_EMAIL=<value>
npm install
npm run import
```

```
Welcome to Performance Dashboard on AWS Import Tool
  Which template you want to install? english
  Do you want to use the default config? ... yes
Using default config: {
  example: "english",
  author: "some@example.com",
  reuseTopicArea: true,
  reuseDashboard: false,
  reuseDataset: false
}
-----
dashboard created
```

Import dashboards using a Lambda function

CFT imports can also be done by changing Lambda function configurations.

1. Sign in to the [AWS Lambda console](#).

2. Identify the Lambda function with Example in its name (example: PerformanceDash-dev-Dashb-SetupExampleDashboardLam-1pJMkog1wsTa).
3. Navigate to the config. tab.
4. Edit the example you want to deploy.
5. Trigger the Lambda function with empty parameters.

Limiting deployed solution access to on-premise networks (IP addresses)

By default, Performance Dashboard on AWS is configured to grant all users access to the web UI, regardless of their physical location or device. To mitigate the threat of malicious attackers accessing the solution as admins., you can configure the solution to only allow access to the **/admin** path (admin UI) for users strictly located within a designated on-premises network (example: an office).

Additionally, you can also restrict access to the entire application so that all users must be connected to your on-premises network to use the solution. These implementations add security by helping to prevent malicious admin activity from unknown sources.

Prevent malicious solution access with AWS WAF

You can use [AWS WAF](#) to restrict access so that only requests originating in your on-premises network are authorized. To do so, create a [web access control list \(web ACL\)](#) to allow or block requests based on the IP addresses that the requests originate from. Then, you can use that web ACL to protect the CloudFront distribution used with Performance Dashboard on AWS. For more information, see the diagram below:

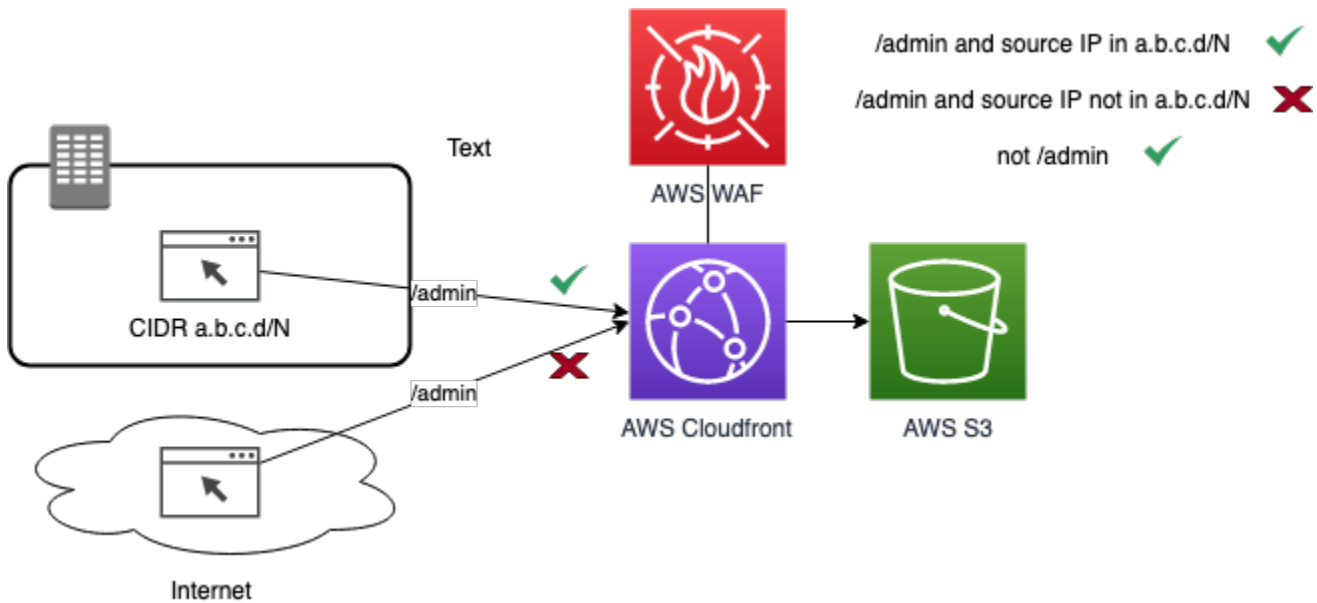


Figure 4: Using AWS WAF to limit access to Performance Dashboard on AWS.

Configure a web ACL to limit solution access

To configure a web ACL that restricts access to certain IP addresses, you need to manually create and associate it with the Performance Dashboard CloudFront distribution. Refer to the following steps below to configure a web ACL:

To create the web ACL described above, you can run the Cloud Formation template (CFT) [WAF-enterprise-only](#) in the us-east-1 region.

1. Download the ["WAF-enterprise-only.json" file](#) locally.
2. Sign in to the [CloudFormation console](#).
3. Choose **Create Stack**.
4. Choose **Upload a template file**.
5. Select your downloaded `WAF-enterprise-only.json` file to upload.
6. Enter a stack name (example: "PerfDash-WAF").

The template also requires a list of CIDRs to limit access to (allowlist).

7. Enter the CIDRs for your internal network.

For example, if the CIDRs for your internal network is 1.1.0.0/16 and 2.2.0.0/16, then enter those values as a comma-delimited list "1.1.0.0/16, 2.2.0.0/16".

8. Choose **Next** to skip optional steps.

9. Choose **Create Stack** to finish creating the web ACL.
10. After running the CFT to create the web ACL, navigate to the [CloudFront console](#).
11. Locate the CloudFront distribution that has an **Origin** starting with `performancedash-` and select it.
12. Choose **Edit**.
13. On the **Edit Distribution** screen, configure the [AWS WAF web ACL](#) field and select the web ACL you just created.
14. Choose **Yes, Edit** to save.

Activate Security Assertion Markup Language (SAML) federation

You can activate SAML 2.0 identity federation with your organization's Identity Provider (IdP). Refer to [Adding SAML identity providers to a user pool](#). As part of the setup, you must map the IdP attributes to Amazon Cognito user pool attributes according to the details in the *Amazon Cognito Developer Guide*.

After activating the Amazon Cognito user pool for SAML federation, you must configure Performance Dashboard on AWS to recognize the IdP that you set up. Run a Lambda function to set the environment context used by Performance Dashboard on AWS.

Run the AWS Lambda function

1. Sign in to the [AWS Lambda console](#).

Ensure that you are in the same AWS Region where Performance Dashboard on AWS is deployed. To verify, identify several Lambda functions that start with the prefix `PerformanceDash-`.

2. Select the Lambda function that follows this naming convention: `PerformanceDash-{stage}-Frontend-EnvConfig`.

Note

Do not select the Lambda function with a similar name. (It includes the word *Provider*.)

3. Locate the section where the environment variables are defined, and choose **Edit**.
4. Modify the **FRONTEND_DOMAIN** variable with the URL of your instance of Performance Dashboard on AWS, for example, <https://example.com/admin>.

5. Modify the **COGNITO_DOMAIN** variable with the value that you configured for the **Amazon Cognito** domain value of your user pool.
6. Modify the **SAML_PROVIDER** variable with the name of the IdP that you configured with the user pool.
7. Choose **Test** to invoke the Lambda function. When prompted to provide a **Test Event**, enter any name that you like, then copy and paste the following JSON input into the content area:

```
{
  "RequestType": "Update"
}
```

8. Choose **Test** to run the Lambda function. The success response confirms a new `env.js` file was generated with the new values and uploaded to the Amazon S3 bucket.

Remove old file from cache

Now, you must invalidate the CloudFront distribution so that the old `env.js` file is removed from the CloudFront cache.

1. Sign in to the [CloudFront console](#).
2. Select the CloudFront distribution that starts with `env.js`.
3. Select the **Invalidations** tab.
4. Select **Create Invalidation**.
5. In the **object-paths** input field, enter `/*`, then choose **Invalidate**.
6. After waiting a few minutes for CloudFront to finish invalidating the cache, open a new browser session to sign in to your solution instance. You should see a new **Enterprise Sign-In** button on the sign-in page. Choose that button to initiate the sign-in process with your IdP.

Customization guide

Change domain names (post-deployment)

You can use your organization's personal domain name with Performance Dashboard on AWS. Assuming your organization already has a personal domain name, you can add that domain to Performance Dashboard by assigning it in [Amazon Route 53](#). then you can assign your organization's domain name to the current web address through [Amazon Route 53](#). Otherwise,

you can register and host a new domain to use with Performance Dashboard through [Amazon Route 53](#).

Change UI text language (during deployment)

By default, Performance Dashboard on AWS is deployed with English UI text. However, you can use the AWS CDK to deploy Performance Dashboard with pre-written UI text in other languages like Spanish and Portuguese. This is done by using the CDK to change the solution's Lambda generator function.

1. Run the deployment script for your desired environment, replacing `{env}` with your desired environment (example: `dev` or `staging`), and adding the desired language's name to the end of the deployment script:

```
./deploy.sh {env} spanish
```

Customize new user invitation email template (pre-deployment)

When you create a new user in Performance Dashboard on AWS, an email is sent to invite that user to login to the application. Performance Dashboard provides a default invite email template, but you can customize it before deploying with the AWS CDK to better suit your organization.

1. Open `cdk/lib/data/email-template.html` in your editor.
2. Replace the placeholder values enclosed in brackets with your own values:
 - `[Organization]` - Replace this with your organization name.
 - `[Administrator]` - Replace this with the name of the person sending the invite.
 - `[your domain]` - Replace this with the domain name you're using for Performance Dashboard on AWS. For example, change the placeholder value `https://<your domain>/admin` to `https://example.com/admin`. There are multiple instances of `<your domain>`.

Note

Do not change the username (`{username}`) and temporary password (`{#####}`) values in the email template.

1. After making the edits, install your changes using the CDK. See the section above on installing Performance Dashboard using the CDK.

Note

You can also change the Performance Dashboard email template after deploying the application. For more information, refer to [Step 2: Configure user invitation email template \(post-deployment\)](#).

Change frontend details (post-deployment)

The appropriate way to modify the values (topic area label, site name, and contact email address) of the `env.js` file is to change the environment variables of the Lambda Function (Custom Resource) that generates it.

1. Sign in to the [AWS Lambda console](#).

Make sure you are on the same AWS region where the Performance Dashboard is deployed. You should see several Lambda functions that start with the prefix `PerformanceDash-`.

2. Select the Lambda function for your environment, named like `PerformanceDash-{stage}-Frontend-EnvConfig`.

Do not select the similarly named function with "Provider" in its name.

3. Locate the defined environment variables (topic area label, site name, and contact email address) for the selected Lambda function.
4. Select **Edit** and modify their values.
5. Choose **Save**.
6. Invoke the Lambda function by selecting **Test** within the console.

You will be asked to provide a **Test Event**.

7. Enter any name you want for the test event, and copy/paste the following JSON input in the content area:

```
{  
  "RequestType": "Update"  
}
```

```
}
```

8. Choose **Test** again to execute the Lambda function.

You should receive a success response. This means Lambda generated a new `env.js` file, containing new values, that was uploaded to the solution's S3 bucket. You need to invalidate the CloudFront distribution so that the old `env.js` file is removed from the CloudFront cache.

1. Sign in to the [CloudFront console](#).

Locate and select the CloudFront distribution with an **Origin** that starts `performancedash-`.

2. Navigate to the **Invalidations** tab.

3. Choose **Create Invalidation**.

4. In the `object-paths` input field, enter the following text: `/env.js`.

5. Choose **Invalidate**.

Wait a few minutes while CloudFront finishes invalidating the cache. After that, your new custom values will be reflected in the solution's frontend.

Change homepage title and description (post-deployment)

The homepage is the first page that public users of the solution see when they visit the website, and you can customize them.

1. Sign in to the [DynamoDB console](#).

2. Navigate to the **Tables** item in the left menu and select the Main table.

3. Choose **Create Item**.

4. From within the modal, switch to **Text** view in the dropdown tab.

5. Copy/paste the following JSON content into the text area to replace the existing text:

```
{
  "description": "This is my description", <-- Insert a custom string as a
description.
  "title": "Welcome to my Performance Dashboard", <-- Insert a custom string as a
title.
  "pk": "Homepage",
  "sk": "Homepage",
```



```
"type": "Homepage"  
}
```

6. Choose **Save**. (Your changes should be reflected automatically in the homepage.)

Require user credentials for all users

By default, Performance Dashboard on AWS grants any non-admin user access to the web UI without requiring them to provide credentials. However, you can use the AWS CDK to deploy Performance Dashboard so that all users must provide credentials to access the site.

1. Run the deployment script for your desired environment, replacing `{env}` with your desired environment (example: `dev` or `staging`) and adding `true` to the end of the deployment script:

```
./deploy.sh {env} true
```

Users will now have to sign in to access the public website, and a new IAM role ("Public") is also available to assign to users.

Customize application header logo

You can replace the logo displayed on the Performance Dashboard on AWS web page header.

1. Sign in to the [Amazon S3 console](#).
2. Choose **Buckets** from the left navigation pane.
3. Look for a bucket that has `frontendstack` and `"reactapp"` as part of the name, such as `"performancedash-beta-frontend-reactapp05daba0d-e5r481e3f18t"`.
4. Select that bucket, then select the folder `"static"`, and then select the folder `"media"`.
5. Look for the object with a name starting with `"logo"`.
6. Upload your logo to replace that object.

Note

Be sure to use the same name as the logo that exists. For example, if the existing logo is named `"logo.27a7fd8b.svg"`, when uploading your logo, make sure to use that same name.

Data Ingestion API

Performance Dashboard on AWS has private APIs called by the web frontend component to access the AWS Lambda functions that perform the application functions, and also a public Data Ingestion API that can be used by external parties to upload datasets to be used with dashboards. All APIs are documented in this [OpenAPI specification](#), and you can use the specification to configure your API client to call the Data Ingestion API.

The Data Ingestion API can be used to incorporate [dynamic datasets](#) stored in Amazon S3 into created dashboards. Invoking the API updates dashboards. To populate the dashboards that you create, you must feed data into the solution. You can do this by uploading files through the web interface or pushing the data through the Data Ingestion API. The Data Ingestion API is a REST API with the following endpoints:

- POST `/ingestapi/dataset` - create a dataset using the request body below. On a successful call, the identifier of the created dataset is returned.

```
{
  "metadata": {
    "name": "<your-dataset-name>",
    "type": "json"
  },
  "data": {
    <JSON-data>
  }
}
```

- PUT `/ingestapi/dataset/<id>` - update a dataset using a request body with the same schema as used in the POST operation. The `<id>` parameter is the identifier returned when the dataset was created with the POST operation.
- DELETE `/ingestapi/dataset/<id>` - delete a dataset. The `<id>` parameter is the identifier returned when the dataset was created with the POST operation.

Create new dataset (example)

The API call (shown below) will create a new dataset. This process programmatically mirrors the user experience of uploading datasets through the web UI. Upon successfully making this POST


call, the dataset will be available in the UI as a [dynamic dataset](#) when creating a chart or table. Selecting a dynamic dataset will link the API-produced dataset to a chart and table in a dashboard.

POST: `{YourBaseURL}/ingestapi/dataset` - create a dataset with the following payload:

```
{
  "metadata": {
    "name": "New dataset",
    "type": "/json "
  },
  "data": [{
    "Date": "1/25/2021",
    "Emails": 2290973,
    "Letters": 122180,
    "Text Message": 2027903
  },
  {
    "Date": "1/26/2021",
    "Emails": 2962568,
    "Letters": 251346,
    "Text Message": 1578929
  },
  {
    "Date": "1/27/2021",
    "Emails": 3173257,
    "Letters": 81658,
    "Text Message": 1918500
  },
  {
    "Date": "1/28/2021",
    "Emails": 3971402,
    "Letters": 21165,
    "Text Message": 2398909
  },
  {
    "Date": "1/29/2021",
    "Emails": 3952829,
    "Letters": 178913,
    "Text Message": 2007563
  }
  ]
}
```

Update existing dataset (example)

The API call (shown below) will update the existing dataset created above. This acts as a replacement, not an append. Upon a successful PUT call, the dataset will now be composed of the data included in your PUT call's payload. Any existing chart or table linked to this dataset will automatically reflect the new data in their dashboard.

 **Note**

You must include the dataset ID that you want to update in the PUT URL.

```
PUT: {YourBaseURL}/ingestapi/dataset/<datasetId>:
```

```
{
  "metadata": {
    "name": "Notification Type Running Totals",
    "type": "json"
  },
  "data": [{
    "Date": "1/25/2021",
    "Emails": 2290973,
    "Letters": 122180,
    "Text Message": 2027903
  },
  {
    "Date": "1/26/2021",
    "Emails": 2962568,
    "Letters": 251346,
    "Text Message": 1578929
  },
  {
    "Date": "1/27/2021",
    "Emails": 3173257,
    "Letters": 81658,
    "Text Message": 1918500
  },
  {
    "Date": "1/28/2021",
    "Emails": 3971402,
    "Letters": 21165,
    "Text Message": 2398909
  },
  {
    "Date": "1/29/2021",
```

```
"Emails": 3952829,
"Letters": 178913,
"Text Message": 2007563
},
{
  "Date": "1/30/2021",
  "Emails": 2081055,
  "Letters": 7039,
  "Text Message": 1434438
}
]
```

Enable the API

By default, the Data Ingestion API is restricted from being called from any IP address after solution deployment. To enable the Data Ingestion API to be called from an allowed CIDR range:

1. Sign in to the [Amazon API Gateway console](#).
2. Select the solution's API (named "APIGateway").
3. While on **Resource Policy**, edit the following policy:

```
{
  "Effect": "Deny",
  "Principal": "",
  "Action": "execute-api:Invoke",
  "Resource": "arn:aws:execute-api:::/prod//ingestapi/*",
  "Condition": {
    "IpAddress": {
      "aws:SourceIp": "0.0.0.0/0"
    }
  }
}
```

1. Change the Deny clause to Allow.
2. Replace the "0.0.0.0/0" value with an allowed CIDR range.

In place of the API Gateway resource policy (or in conjunction with), you can control access to the Data Ingestion API using techniques like these:

- Use the AWS WAF to enable an IP-range allowlist that API calls can originate from. The WAF can also be setup with rules to prevent common web attacks.
- Use mutual [Transport Layer Security](#) with the API Gateway to allow calls from trusted parties only.
- Configure the API to be private using VPC endpoint to limit access to callers within a particular VPC or on-premises connecting through Direct Connect.
- Control access with IAM permissions.

Obtain the API Key

This API is configured to require an API key passed in the AWS-API-KEY HTTP header field on every call, or else the call is rejected. By default, Performance Dashboard on AWS installs an API key with API Gateway for this API with a usage plan of 25 requests per second and a burst of 50 requests. The usage plan is named **PerfDashIngestUsagePlan**.

To retrieve the API key configured for the API:

1. Sign in to the [API Gateway console](#).
2. Select the "ApiGateway" API configured for this solution.
3. Choose the **API Keys** option in the left menu bar.
4. For the API key that begins with "Perfo-ApiGa-", select and copy its API key value.

Scaling

We load tested this solution to determine how it scales. Our benchmark considers that Performance Dashboard on AWS is the only application deployed in an AWS account. Having other applications running on the same account and in the same Region might impact your experience because AWS quotas are shared across all applications in the account and Region.

Note

The following numbers are for guidance and not a guarantee. They are the result of the load testing we did, but we recommend you do your own load testing on your Performance Dashboard on AWS installation so that you can get more accurate numbers for your specific situation.

We experienced a P99 latency for back-end requests to be 255ms. In other words, 99% of the requests to the backend finished in less than 255ms. This means that a single instance of a Lambda function can handle approximately 3.9 requests per second (RPS). So, in an AWS Account with the default quota of 1,000 concurrent Lambda runs, we observed that a load of 3,000 RPS to the Performance Dashboard on AWS was handled without any throttles or errors. In our benchmark, throttles started occurring right after we passed that threshold of 3,000 RPS.

If you plan to serve more than 3,000 requests per second, consider requesting an increase of the [AWS Lambda Concurrent Executions quota](#) for your account. And if you plan to serve more than 10,000 requests per second, consider requesting an increase of the [API Gateway RPS quota](#) as well.

Monitoring

This solution comes with an operations dashboard in Amazon CloudWatch that displays the current health of the Performance Dashboard on AWS infrastructure. This dashboard is installed by default when the solution is deployed. View the dashboard by going to the Amazon CloudWatch console, then opening the dashboard that starts with OpsDashboard in the name.

In the dashboard, you can track the following operational metrics:

- **API Latency (P99)**
- **Count of API Requests**
- **Count of API Error**

You can also track graphs of the following metrics across a period of duration:

- **API Latency**
- **API Requests**
- **Lambda invocations**
- **DynamoDB latency by request**
- **DynamoDB errors**

Performance Dashboard on AWS also has alarms and thresholds that you can configure to be notified and take action upon:

- **DynamoDB streams throttle alarm** (throttles ≥ 10 for 2 datapoints within 10 minutes)
- **DynamoDB streams error alarm** (error Rate (%) ≥ 5 for 2 datapoints within 10 minutes)
- **API error rate alarm** (error Rate (%) ≥ 5 for 2 datapoints within 10 minutes)
- **API throttle rate alarm** (throttles ≥ 10 for 2 datapoints within 10 minutes)

Reference

This section includes information about an optional feature for collecting unique metrics for this solution, pointers to related resources, and a list of builders who contributed to this solution.

Anonymous data collection

This solution includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. When invoked, the following information is collected and sent to AWS:

- **Solution ID** - The AWS solution identifier
- **Unique ID (UUID)** - Randomly generated, unique identifier for each Performance Dashboard on AWS deployment
- **Timestamp** - Data-collection timestamp

AWS owns the data gathered through this survey. Data collection is subject to the [AWS Privacy Policy](#). To opt out of this feature, complete the following steps before launching the AWS CloudFormation template.

1. Download the [AWS CloudFormation template](#) to your local hard drive.
2. Open the AWS CloudFormation template with a text editor.
3. Modify the AWS CloudFormation template mapping section from:

```
AnonymousData:  
  SendAnonymousData:  
    Data: Yes
```

to:

```
AnonymousData:  
  SendAnonymousData:  
    Data: No
```

4. Sign in to the [AWS CloudFormation console](#).
5. Select Create stack.

6. On the Create stack page, Specify template section, select Upload a template file.
7. Under **Upload a template file**, choose **Choose file** and select the edited template from your local drive.
8. Choose **Next** and follow the steps in [Launch the stack](#) in the Deploy the solution section of this guide.

Contributors

- Jason Gudalis
- Triet Lu
- Fernando Dingler
- Miguel Pavon Diaz
- Addy Upreti
- Brandi Hopkins
- Luke Cooper

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. AWS responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Performance Dashboard on AWS is licensed under the terms of the of the Apache License Version 2.0 available at [The Apache Software Foundation](#).

Revisions

For more information, see the [CHANGELOG.md](#) file in the GitHub repository.

Date	Change
April 2021	Initial release
September 2021	Documentation update: Updated the instructions for uninstalling the solution.
October 2021	Release version 1.1.1: Code changes. For more information, refer to the CHANGELOG.md file in the GitHub repository.
September 2022	Release version 1.7.0: Code changes. For more information, refer to the CHANGELOG.md file in the GitHub repository.
December 2022	Release version 1.8.0: Code changes. For more information, refer to the CHANGELOG.md file in the GitHub repository.
July 2023	Release version 1.9.0: Code changes. For more information, refer to the CHANGELOG.md file in the GitHub repository.
December 2023	Release version 1.9.1: Code changes. For more information, refer to the CHANGELOG.md file in the GitHub repository.
February 2024	Release version 1.9.2: Code changes. For more information, refer to the CHANGELOG.md file in the GitHub repository.