Performance Dashboard on AWS

Implementation Guide

Performance Dashboard on AWS: Implementation Guide

Table of Contents

Open-source solution for creating customizable performance dashboards

Publication date: April 2021 (last update (p. 27): October 2021)

This implementation guide describes architectural considerations and configuration steps for deploying the Performance Dashboard on AWS in the Amazon Web Services (AWS) Cloud. It includes links to an AWS CloudFormation template that launches and configures the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT architects, developers, DevOps, data analysts, and marketing technology professionals who have practical experience architecting in the AWS Cloud.

Overview

This solutions implementation is an open-source solution for creating customizable dashboards to communicate the data-driven performance of public sector services. Citizens expect these services to be operational and useful. Measuring and openly sharing key performance indicators helps you to build trust in your stakeholder relationships, demonstrate success through data, and promote accountability. Customizing dashboards with a variety of components, including charts and narrative text, helps you to achieve your communication goals and offer an accessible experience. Access to centralized performance data provides up-to-date information for your stakeholders and supports effective decision making.

Choose either option to upload the data:

- Automatically via a standard API
- Manually uploading data files through the solution's web interface

This guide provides infrastructure and configuration information for planning and deploying the solution in the AWS Cloud.

Cost

You are responsible for the cost of the AWS services used while running this solution. As of April 2021, the estimated cost for running this solution for 5,000 users viewing dashboards, and four editors working eight hours per day creating dashboards, with default settings in the US East (N. Virginia) Region, is approximately **$38.00 per month**. The cost estimate assumes the users spend daily sessions of 10 minutes viewing the dashboards, clicking twice a minute, and each dashboard having 10 charts and using 500 KB datasets. This user load results in approximately three million requests a month against the system. If you have a larger number of users viewing the dashboards, or if your dashboards have more charts or use larger datasets, your cost will be higher.

Table 1: Cost per month

This solution uses the following resources that are billed on a monthly basis.

AWS service	Quantity	Cost
Amazon S3	For datasets: 500 GB storage, 3 million retrieval, 12 thousand store	$12.75
Amazon API Gateway	3 million requests from public and admin users	$10.50
AWS DynamoDB	5 GB storage of dashboard metadata	$ 6.00
Amazon CloudFront	15 GB data transfer out for web page requests	$ 4.75

AWS service	Quantity	Cost
Amazon CloudWatch	7 GB data ingested for logging requests	$ 3.55
AWS Lambda	3 million requests from public and admin users	$ 0.50
	Total monthly cost:	$38.05

Prices are subject to change. For full details, refer to the pricing webpage for each AWS service you will be using in this solution.

Architecture overview

Deploying this solution with the default parameters builds the following environment in the AWS Cloud.

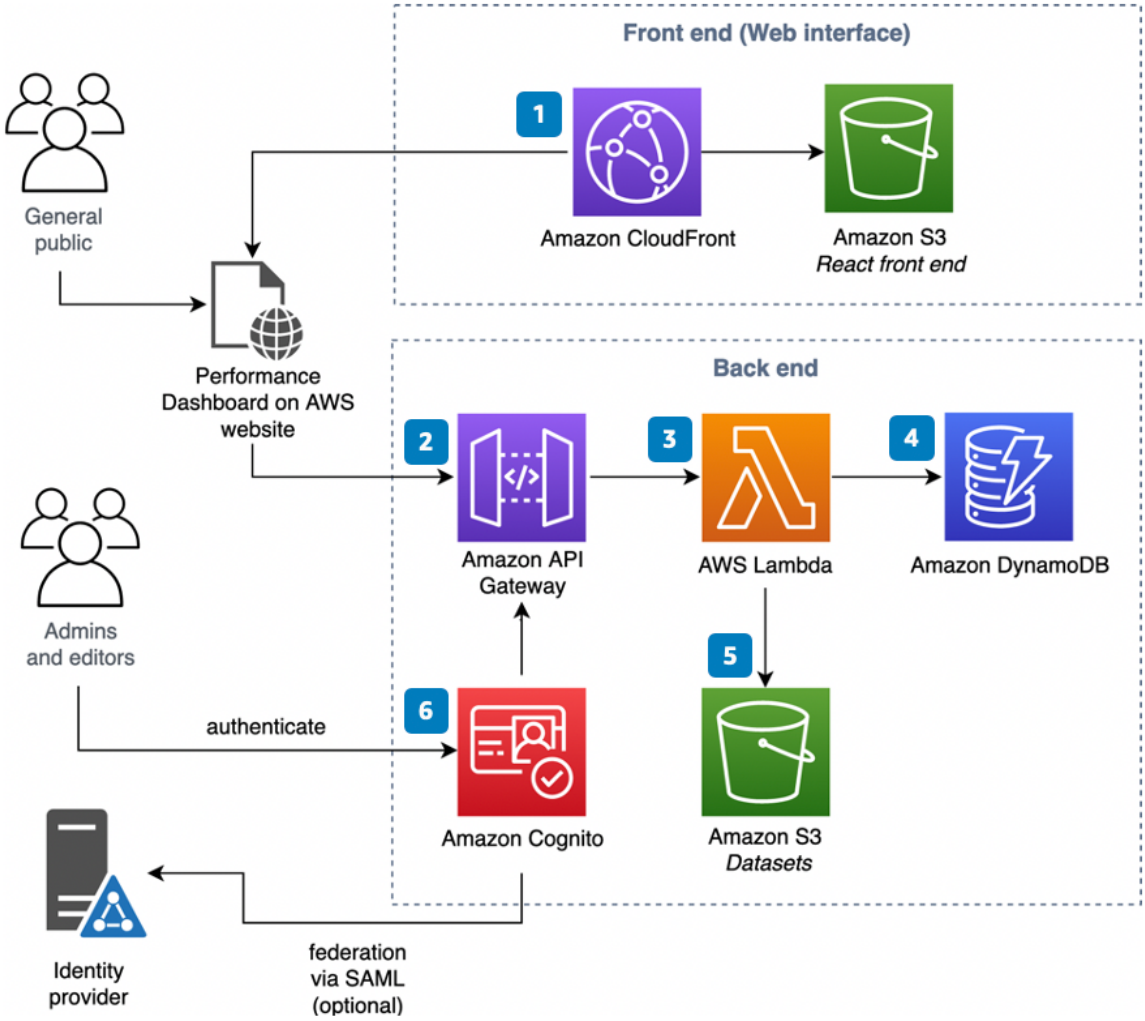

Figure 1: Performance Dashboard on AWS architecture

The AWS CloudFormation template provisions the following resources:

1. An Amazon CloudFront distribution and an Amazon Simple Storage Service (Amazon S3) bucket to host and serve the web front end, which includes HTML pages, CSS stylesheets, and Javascript code.
2. An Amazon API Gateway resource to host the APIs called by the web front end to access the AWS Lambda functions that perform the application functions.
3. AWS Lambda functions that utilize Node.js to perform functions and access data related to creating and serving dashboards.
4. An Amazon DynamoDB table to store metadata about the dashboards and datasets.
5. An Amazon S3 bucket to store the datasets used with the dashboards.
6. An Amazon Cognito user pool to store the identities of the users creating the dashboards.

Solution components

Web front end

This solution includes a web front end for creating dashboards and availing them to end users. The web front end serves as both the public-facing website and the administrator (admin) portal. Users accessing the public-facing website do not need to sign in, while users accessing the admin portal requires Amazon Cognito user authentication.

Public users load the website on their browser by using the CloudFront distribution URL. As a content delivery network (CDN) service, CloudFront then serves the content from the edge location with the lowest latency to the user. If the content is not in that edge location already, CloudFront retrieves it from the S3 bucket.

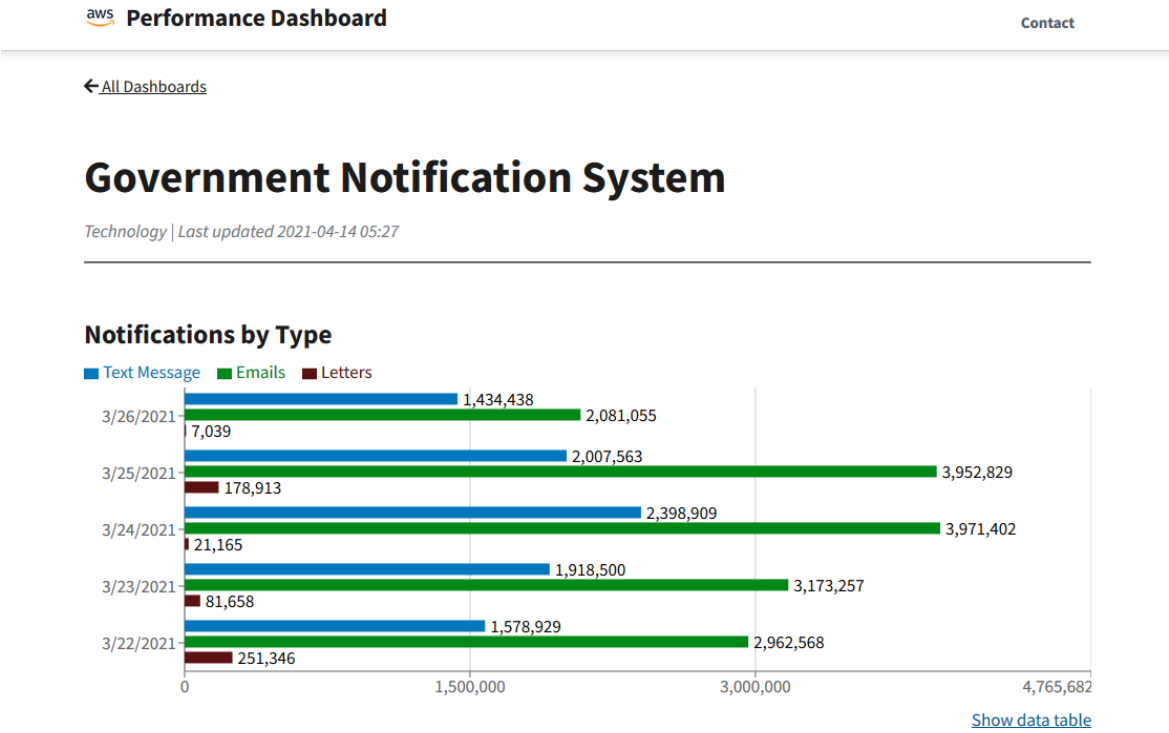

Figure 2: Sample dashboard created by Performance Dashboard on AWS

For details about using the web interface, refer to Working with dashboards (p. 17).

Roles

The admin portal supports two roles, *Admin* and *Editor*. The Editor role is responsible for creating dashboards and publishing them for viewing. The Admin role also manages users, and makes sitewide

setting updates, such as the logo and colors for styling. After signing in to the web interface, the Admin and Editor users are presented with different features based on their role.

Application backend

The application backend of the solution handles requests from the web front end to create and serve dashboards. The front end calls APIs hosted in the Amazon API Gateway to make requests to the backend. The APIs then route the calls to the AWS Lambda functions to process the requests and access data about the dashboards stored in Amazon DynamoDB. For example, when the front end displays a dashboard, it calls an API in the backend, which invokes a Lambda function to retrieve the content items and datasets of the dashboard to return to the front end to display.

The solution provisions two DynamoDB tables during initial deployment: `Main`, and `AuditTrail`. The processing performed by the Lambda functions, such as 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.

Data Ingestion API

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 via 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

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 `PerfDashIngestUsngePlan`.

User management

When this solution is installed for the first time, it creates an Admin user in Amazon Cognito. This Admin user can create other Admin and/or Editor users in Amazon Cognito. 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. An Admin user can assign the Admin or Editor role to other users and also delete users.

This solution supports authentication for users in a Security Assertion Markup Language (SAML) identity provider (IdP). Simply configure Amazon Cognito to support federation with that IdP. A user signing 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 the Performance Dashboard on AWS Admin to grant them the proper role.

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 constructs to access Regional resources, such as:

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

Amazon Cognito

This solution uses Amazon Cognito user pools and identity pools. User pools are user directories that provide sign-in functionality for the web users. Identity pools provide AWS credentials to grant the web users access to other AWS services, such as the ability to access data stored in Amazon S3 to render in dashboards. After a successful user pool sign-in, the solution's web front end receives user pool tokens from Amazon Cognito. These tokens are used to control access to server-side resources. For example, the API Gateway instance is configured with a Cognito authorizer that validates web requests for the presence of a proper token (for example, signed by the user pool and hasn't expired).

User pool attributes are also used to manage permissions, and to represent different types of users in the solution, such as Editor and Administrator. Public users don't sign in to view dashboards published by the solution. Those users are assigned temporary credentials by the identity pool in order to access data in Amazon S3 rendered in dashboards. The 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.

Amazon CloudFront

This solution deploys a web console hosted in an 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, refer to Restricting Access to Amazon S3 Content by Using an Origin Access Identity in the *Amazon CloudFront Developer Guide*.

A Lambda@Edge function is deployed to run at the CloudFront edge locations to inject HTTP security headers (for example, Content-Security-Policy, X-XSS-Protection) into the HTTP responses returned by the solution to improve the security of the web users. For additional details, refer to Adding HTTP Security Headers Using Lambda@Edge and Amazon CloudFront.

This solution also deploys an API Gateway to serve APIs. The edge-optimized API endpoints use a CloudFront distribution to facilitate client access from across Regions to reduce latency and improve security.

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 don't originate in an allow-list 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 repository to configure AWS WAF to limit access to an allow-list of CIDR ranges.

Amazon API Gateway

This solution has private APIs that are called by the web front end. Amazon Cognito manages access control for these APIs. For details, refer to Amazon Cognito (p. 8). The solution's Data Ingestion API is a public API, which is used by consumers to feed datasets for the dashboards created. Use one of the following techniques to control access to this API:

- Use AWS WAF to activate an allow-list of IP range that API calls can originate from.
- Use the API Gateway resource policy to create an allow-list of IP range that API calls can originate from.
- Use mutual TLS authentication for API Gateway to allow calls from trusted parties only.
- Configure the API to be private using Amazon VPC endpoint to limit access to callers within a particular VPC or on-premises connecting via Direct Connect or VPN.
- Use IAM to restrict access to the API.

By default, the Data Ingestion API is configured with a resource policy that stops the API from being called. To start using the API, stop that policy and use one of the previous methods to control access to the API.

Design considerations

Regional deployments

This solution uses the Amazon Cognito service, 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, refer to the AWS Regional Services List.

Quotas

The solution is designed to scale based on the usage volume. However, certain AWS service quotas must be adjusted to increase the scaling limits. The default quota for a CloudFront distribution is currently 250,000 requests per second. To request an increase, refer to Quotas in the *Amazon CloudFront Developer Guide*. The API Gateway can process 10,000 requests per second. For more information, refer to Amazon API Gateway quotas and important notes in the *Amazon API Gateway Developer Guide*. The Lambda functions can process 1,000 concurrent runs. For additional details, refer to Lambda quotas and AWS Lambda function scaling in the *AWS Lambda Developer Guide*.

AWS CloudFormation templates

This solution uses AWS CloudFormation to automate the deployment of Performance Dashboard on AWS in the AWS Cloud. It includes the following CloudFormation templates, which you can download before deployment:

View Template **LambdaEdge.template**: If deploying into an AWS Region other than US East (N. Virginia), deploy this template first to launch the Lambda@Edge component in the AWS US East (N. Virginia) Region.

View Template **performance-dashboard.template**: Use this template to launch the solution and all associated components. The default configuration deploys Amazon CloudFront, Amazon API Gateway, AWS Lambda, Amazon Cognito, Amazon DynamoDB, Amazon S3, AWS IAM, and AWS X-Ray resources. If deploying into the AWS US East (N. Virginia) region, Lambda@Edge resources are also deployed.

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

Automated deployment

Before you launch the solution, review the architecture, solution components, security, and design 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

Update the stack

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

If you had deployed the solution into a Region other than US East (N. Virginia), then update the Lambda@Edge stack in the US East (N. Virginia) Region first. If you had deployed the solution into the US East (N. Virginia) Region, then you can skip this procedure. To get the latest version, follow these steps to update CloudFormation stack for the Lambda@Edge deployment.

1. Sign in to the AWS Cloudformation console, select your existing Performance Dashboard on AWS CloudFormation stack, and select **Update**.
2. Select **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. Refer to Step 1. Launch the Stack (p. 14) for details about the parameters.
5. Choose **Next**.
6. On the **Configure stack options** page, choose **Next**.
7. On the **Review** page, review and confirm the settings. Check the boxes acknowledging that the template creates 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 see a status of UPDATE_COMPLETE in approximately about 30 minutes.

Next, follow this procedure to update the CloudFormation 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. Select **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. Refer to Step 1. Launch the Stack (p. 14) for details about the parameters.
5. Choose **Next**.
6. On the **Configure stack options** page, choose **Next**.
7. On the **Review** page, review and confirm the settings. Check the boxes acknowledging that the template creates 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 see a status of UPDATE_COMPLETE in approximately about 30 minutes.

Launch the Lambda@Edge stack

If you plan to deploy the solution into a Region other than US East (N. Virginia), then you must launch the Lambda@Edge stack in US East (N. Virginia) Region first. If you plan to deploy the solution into the US East (N. Virginia) Region, then you can skip this procedure. This automated AWS CloudFormation template deploys the AWS Lambda@Edge function in the AWS Cloud. You must have access to an AWS account with permission to deploy resources before launching the stack.

1. Sign in to the AWS Management Console and select the button to launch the `LambdaEdge` AWS CloudFormation template.

   

   Alternatively, you can download the template as a starting point for your own implementation.
2. Launch this template in the US East (N. Virginia) Region.
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. 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 creates AWS Identity and Access Management (IAM) resources.
8. 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 five minutes.

Deployment overview

Use the following steps to deploy this solution on AWS. For detailed instructions, follow the links for each step.

Step 1. Launch the stack

This automated AWS CloudFormation template deploys the solution implementation in the AWS Cloud. You must have access to an AWS account with permission to deploy resources before launching the stack.

> **Note**
> You are responsible for the cost of the AWS services used while running this solution. For more details, visit to the Cost (p. 2) section in this guide, and refer to the pricing webpage for each AWS service used in this solution.

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

   

   Alternatively, you can download the template as a starting point for your own implementation.
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 service, 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, refer to the AWS Service Region Table.
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 value.

Parameter	Default	Description
AdminEmail	*<Requires input>*	The email address of the user who will initially administer the solution. A login will be created for this user. An email inviting the user to sign in will be sent to the email address provided.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the boxes acknowledging that the template creates 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 email invitation template

As the initial user to set up the solution on AWS, add the Administrators and Editors who will be creating and publishing dashboards. 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 **Manage User Pools**, then select the user pool created for Performance Dashboard on AWS.
3. In the navigation page, choose **Message customizations**.
4. 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 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. {username} and {####} - the username and temporary password created by Cognito. Don't replace these, leave them as is.
   e. Update the logo used by the email. In the email template, find the text `src="https://`*<your domain>*`/logo.png"`, and replace 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 home page, and copy the link address. Paste that address into the email template to replace the placeholder logo.

Step 3. Sign in to Performance Dashboard on AWS

After launching the solution, obtain the URL of the Performance Dashboard on AWS instance and sign in.

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.

   At this stage, your public homepage will not have any dashboards to display.
5. To create dashboards, add `/admin` to the web address.
6. Use the username and temporary password from your invitation email to sign in to Performance Dashboard on AWS.
7. After signing in, follow the prompts to change your password.

Step 4. Add users

After signing in to the web interface, add users to the Administrator and Editor roles. Use the following procedure to add users.

1. Go to the Performance Dashboard on AWS admin page. For details, refer to Step 3. Sign in to Performance Dashboard on AWS (p. 15).
2. In the top navigation bar, choose **Manage users**.
3. Select **Add user(s)**:
   a. Enter email addresses for all new users, separated by a comma.
   b. Choose the Editor and Admin role for the users to be added.
   c. Choose **Add user(s) and send invite**.

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

Step 5. Create topic areas

Before dashboards can be created, you must create topic areas, which are categories used to organize and group your dashboards.

1. Go to the Performance Dashboard on AWS admin page. For details, refer to Step 3. Sign in to Performance Dashboard on AWS (p. 15).
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, and choose **Create**. Choose a name that reflects how you want to group your dashboards for your organization, such as by department, for example `Accounting`.
5. Create additional topic areas, as needed.

Working with dashboards

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.

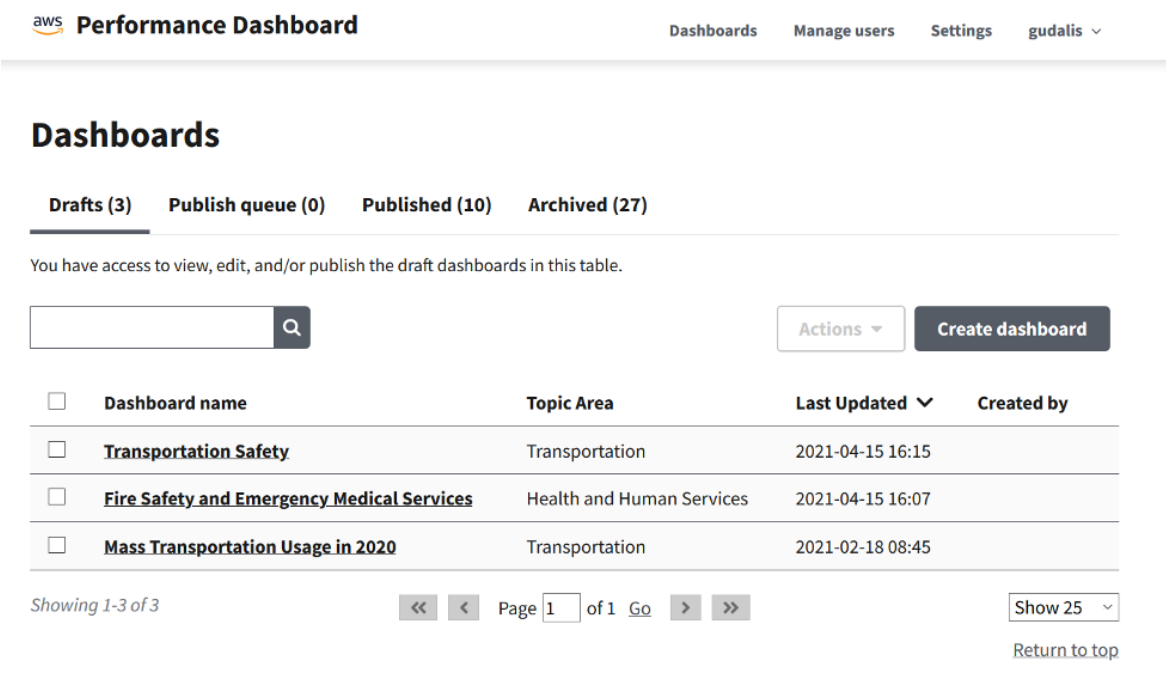

Figure 3: Performance Dashboard on AWS architecture Dashboards page

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 order when 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 files** 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. Review the file content, and choose **Continue**.
7. Enter a chart title.
8. Select **Line** as chart type.
9. 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.

Additional resources

AWS services	
• Amazon API Gateway	• AWS CloudFormation
• Amazon CloudFront	• AWS Lambda
• Amazon CloudWatch	• AWS Lambda@Edge
• Amazon Cognito	• AWS X-Ray
• Amazon DynamoDB	• AWS Web Application Firewall (AWS WAF)
• Amazon Simple Storage Service (Amazon S3)	

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 back-end 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)

Activate 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 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 follows this naming convention: `PerformanceDash-{stage}-Frontend-EnvConfig`.

   Be careful not to select the Lambda function with a similar name that includes the word *Provider*.
3. Identify 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 `performancedash-`.
3. Choose the **Invalidations** tab, and choose **Create Invalidation**.
4. In the **object-paths** input field, enter **/***, then choose **Invalidate**.
5. After waiting a few minutes for CloudFront to finish invalidating the cache, open a new browser session to sign in to your Performance Dashboard on AWS 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.

Uninstall the solution

To uninstall the Performance Dashboard solution, if you installed both the Lambda@Edge stack and the Performance Dashboard stack, then delete both these stacks, starting with the Performance Dashboard stack. If you installed only the Performance Dashboard stack into `us-east-1`, then delete it.

Using the AWS Management Console

1. Sign in to the AWS CloudFormation console.
2. Select this solution's installation stack.
3. Choose **Delete**.

   The nested stacks (`opsStack`, `frontendStack`, `frontend-support`, `backendStack`, and `authStack`) delete first, then 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, refer to 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

After the stack deletion has completed, delete the Amazon S3 bucket.

1. Sign in to the Amazon S3 console.
2. Select the 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 bucket named `performancedash-<random string>-datasets`. This bucket contains datasets that you created to be used with your dashboard.
3. Back up the data in that bucket that you want to keep, such as by copying the data to another S3 bucket where you keep all your backups.
4. Choose **Empty**.
5. Choose **Delete**.

Deleting the DynamoDB tables

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 the 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 the name.
6. Choose **Delete table**.
7. Type *delete*, and choose **Delete**.

Collection of operational metrics

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 <solution-name> deployment
- **Timestamp:** Data-collection timestamp

AWS owns the data gathered though this survey. Data collection is subject to the AWS Privacy Policy. To opt out of this feature, modify the AWS CloudFormation template mapping section from:

```
"Send" : {

"AnonymousUsage" : { "Data" : "Yes" }

},
```

to:

```
"Send" : {

"AnonymousUsage" : { "Data" : "No" }

},
```

Source code

Visit our Github repository to download the source files for this solution and to share your customizations with others. The Performance Dashboard templates are generated using the AWS Cloud Development Kit (CDK). Refer to the README.md file for additional information.

Revisions

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

Contributors

- Jason Gudalis
- Triet Lu
- Fernando Dingler
- Miguel Pavon Diaz
- Addy Upreti

Notices