aws

Implementation Guide

IoT Device Simulator

IoT Device Simulator: Implementation Guide

Table of Contents

Create and simulate hundreds of virtual connected devices without having to configure and manage physical devices

Publication date: May 2018 (last update: April 2024)

Amazon Web Services (AWS) provides many services to help customers build serverless IoT applications that gather, process, analyze, and act on connected device data, without having to manage any infrastructure. With AWS, customers can also build a secure, agile, and scalable backend for their IoT applications. This eliminates the need for customers to develop and manage their own backend resources and can help reduce costs and increase productivity and innovation. However, it is costly and can be a challenge to test IoT applications and backend services without a large pool of physical, connected devices.

IoT Device Simulator is designed to help customers more easily test device integration and IoT backend services, without the need for physical devices. This solution provides a web-based graphical user interface (GUI) that allows customers to create and simulate hundreds of connected devices, without having to configure and manage physical devices, or develop time-consuming scripts. This solution is designed to work out-of-the-box, or you can use this solution as a reference implementation to build a custom simulation engine for your specific use case.

IoT Device Simulator provides a web interface that lets users launch fleets of virtually connected devices from a user-defined template and then simulate them to publish data at regular intervals to AWS IoT. You can also monitor devices from the simulator or observe how backend services are processing the data.

This implementation guide discusses architectural considerations and configuration steps for deploying the IoT Device Simulator in the Amazon Web Services (AWS) Cloud. It includes a link to an AWS CloudFormation template that launches, configures, and runs the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, and DevOps professionals who have practical experience with IoT devices, and the AWS Cloud.

Note

This solution is designed to simulate device data for testing. It is not recommended for use in production environments.

Cost

You are responsible for the cost of the AWS services used while running this solution. As of this revision, the estimated cost for running the IoT Device Simulator solution using the 100 automotive demo device types in a single simulation, sending a message every two seconds in the US East (N. Virginia) Region is **$3.05** per month for a simulation running six hours per day, **$6.11** per month for a simulation running 12 hours per day, and **$12.22** per month for a simulation running 24 hours per day. This includes estimated charges for Amazon API Gateway, AWS Lambda, AWS Step Functions, Amazon DynamoDB, and AWS IoT Core.

The following table provides an example cost breakdown to run 100 device simulations per month in the US East (N. Virginia) Region.

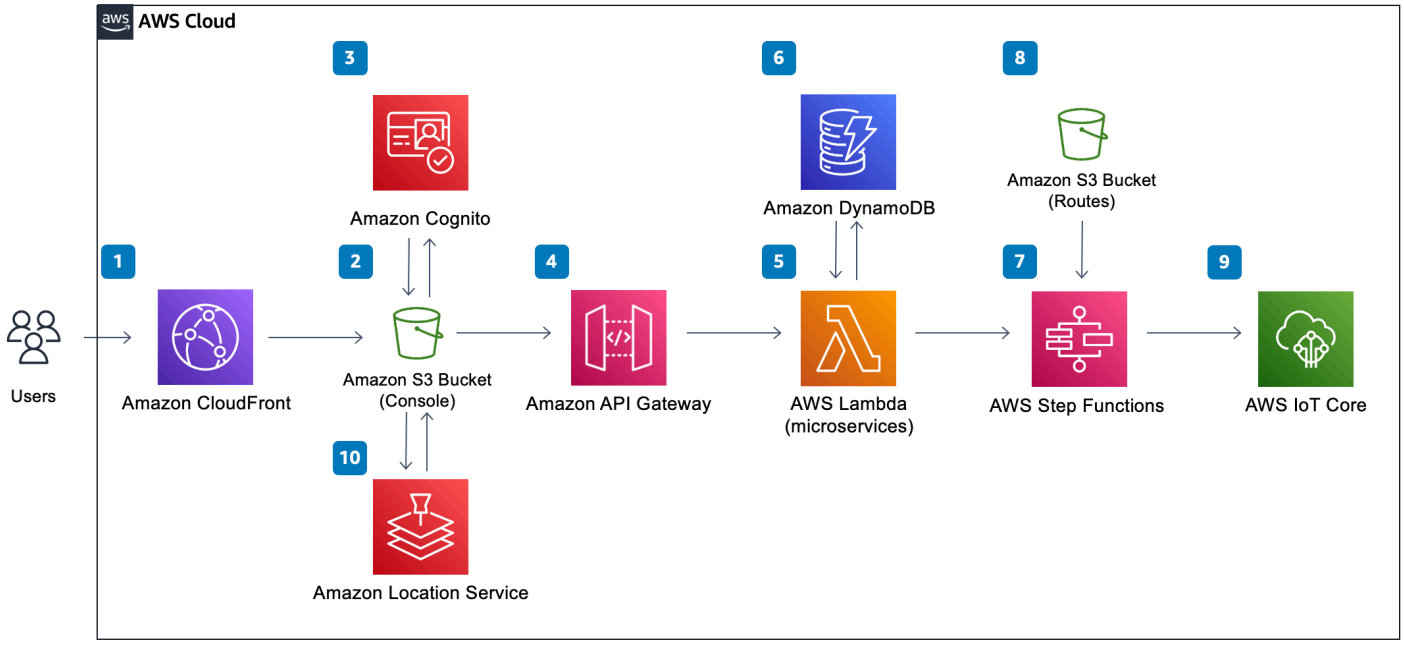
AWS services	6 hours per day	12 hours per day	24 hours per day
Amazon API Gateway	$0.000105	$0.000105	$0.000105
AWS Step Functions	$0.02	$0.04	$0.08
AWS Lambda	$2.70	$5.40	$10.80
Amazon DynamoDB	$0.01	$0.02	$0.04
AWS IoT Core messaging	$0.32	$0.65	$1.30
Total monthly cost [USD]:	**$3.05***	**$6.11***	**$12.22***

*Cost to run 100 device simulations per month.

We recommend creating a budget through AWS Cost Explorer to help manage costs. 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.



IoT Device Simulator architecture

1. Amazon CloudFront to serve the web interface content from an Amazon Simple Storage Service (Amazon S3) bucket.
2. The Amazon S3 bucket hosts the web interface.
3. Amazon Cognito user pool authenticates the API requests.
4. An Amazon API Gateway provides the solution's API layer.
5. AWS Lambda serves as the solution's microservices and routes API requests.
6. Amazon DynamoDB stores simulation and device type information.
7. AWS Step Functions include an AWS Lambda simulator function to simulate devices and send messages.
8. An Amazon S3 bucket stores pre-defined routes that are used for the automotive demo.
9. AWS IoT Core serves as the endpoint to which messages are sent.
10. Amazon Location Service provides the map display showing the location of automotive devices for the automotive demo.

Note

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

Solution components

AWS Step Functions workflow

An AWS Step Functions state machine runs the device simulator. The state machine consists of an AWS Lambda function which provides the logic to create the device messages and send them to the IoT endpoint. An architecture diagram of the workflow and a complete walkthrough can be found in the AWS Step Functions workflow section.

Device Simulator microservices

The IoT Device Simulator microservices are a series of AWS Lambda functions that provide the business logic and data access layer for all device simulation operations. This includes create, read, update, and delete (CRUD) operations for the Amazon DynamoDB simulation and device type tables, as well as starting the step functions workflow. Each Lambda function assumes an AWS Identity and Access Management (IAM) role with least privilege access (minimum permissions necessary) to perform its designated functions.

Web interface

The solution includes an intuitive web interface which is hosted in Amazon S3 and used to simulate the devices. You can use the interface to create and manage simulations and device types, and start device simulations to simulate devices and send messages to the AWS IoT endpoint.

The interface is designed to simulate devices that publish to an AWS IoT endpoint at regular intervals in order to test backend integration.

> **Note**
>
> Device creation and starting and stopping device simulations are routed through the Amazon API Gateway.

Routes bucket

The solution includes an IoT device simulator automotive demo. An Amazon S3 bucket is used to host the pre-defined routes for the automotive demo. The routes are used to provide a pathway for the simulated vehicle to follow. For more details about these routes, refer to Routes in this guide.

Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce 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 read and write to Amazon DynamoDB, publish to an IoT endpoint, read from the Amazon S3 bucket used to host routes, and start the AWS Step Functions state machine.

AWS IoT Core policies

AWS IoT Core policies allow you to control access to the AWS IoT data plane. The AWS IoT data plane consists of operations that allow you to connect to the AWS IoT message broker and send and receive MQ Telemetry Transport (MQTT) messages. The IoT Device Simulator solution creates an AWS IoT policy which allows the web interface to connect to AWS IoT Core, subscribe, and receive MQTT messages.

Amazon CloudFront

This solution deploys a web interface hosted in an Amazon Simple Storage Service (Amazon S3) bucket. To help reduce latency and improve security, this solution includes an AWS CloudFormation 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*.

Amazon API Gateway

This solution deploys an Amazon API Gateway REST API and uses the default API endpoint and Secure Sockets Layer (SSL) certificate. The default API endpoint supports only the TLSv1 protocol.

To use a later version of Transport Layer Security (TLS), use your own domain name and custom SSL certificate. For more information, refer to Choosing a minimum TLS version for a custom domain in API Gateway in the *Amazon API Gateway Developer Guide*.

Design considerations

Updating the solution

This implementation guide contains information about how to set up and configure IoT Device Simulator version 3.0.0. You cannot update version 2.x or earlier versions of this solution to version 3 using the AWS CloudFormation console due to changes with how resources are deployed. To use version 3, you must launch a new stack using version 3.0.0 of the AWS CloudFormation template. You can uninstall your previous version of this solution.

> **Note**
>
> If you have device types saved in a previous version of this solution, we recommend recreating these types using the web interface.

Simulation limits

A simulation is limited to running up to 100 devices; however, multiple simulations can be run concurrently. The number of simulations that can be run concurrently is limited by the number of AWS Lambda concurrently running. For more information on AWS Lambda service quotas, refer to Lambda quotas in the *AWS Lambda Developer Guide*.

Amazon Cognito limits

This solution uses Amazon Cognito user pools to manage users. Amazon Cognito sends an email every time you create a user, change a password, or reset a password. Amazon Cognito limits the number of emails sent daily per user pool to 50. For customers who plan to use this solution for a large number of users, we recommend using Amazon Simple Email Service (Amazon SES) for these emails. For more information, refer to Authorizing Amazon Cognito to Send Amazon SES Email on Your Behalf in the *Amazon Cognito Developer Guide*.

Regional deployments

This solution uses Amazon Location Service, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Amazon Location Service is available. For the most current availability by Region, refer to the AWS Regional Services List.

AWS CloudFormation template

To automate deployment, this solution uses the following AWS CloudFormation template, which you can download before deployment.

View template

iot-device-simulator.template: Use this template to launch the solution and all associated components. The default configuration deploys Amazon S3 buckets, an AWS CloudFormation distribution, an Amazon Cognito user pool, an Amazon API Gateway REST API, AWS Lambda functions, an AWS Step Functions state machine, Amazon DynamoDB tables, and an Amazon Location Service map and place index. You can customize the template to meet your specific needs.

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

Automated deployment

Time to deploy: Approximately 10 minutes

Deployment overview

> **Important**
>
> You cannot update version 2.x or earlier versions of this solution to version 3 using the AWS CloudFormation console due to changes with how resources are deployed. To use version 3, you must launch a new stack using version 3.0.0 of the AWS CloudFormation template. You can uninstall your previous version of this solution.

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

Step 1. Launch the stack

- Launch the AWS CloudFormation template into your AWS account.
- Review the template parameters and enter the values as needed.

Step 2. Sign in to the web interface

Step 3. Create the device types

- Create device types which your devices will represent.

Step 4. Create the simulations

- Create simulations to define the simulation you want to run.

Step 5. Run and manage the simulations

- Start one or more simulations.
- View simulation details and the messages of a simulation if it is currently running.

Step 1. Launch the stack

Important

This solution includes an option to send anonymized 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 though this survey. Data collection is subject to the AWS Privacy Notice.

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 template and deploy the solution. For more information, refer to the Anonymized data collection section in this guide.

This automated AWS CloudFormation template deploys IoT Device Simulator in the AWS Cloud.

Note

You are responsible for the cost of the AWS services used while running this solution. For more details, refer to the Cost 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 `iot-device-simulator.template` 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 Amazon Location Service, which is not currently available in all AWS Regions. You must launch this solution in an AWS Region where Amazon Location

Service is available. For the most current availability by Region, refer to 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, refer to 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
User email	`<Requires input>`	The email used to sign in to the IoT Device Simulator web interface.

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 10 minutes.

Note

In addition to the primary AWS Lambda functions (`microservices` and `simulator`), this solution also includes the `custom resources helper` Lambda function, which runs only during initial configuration or when resources are updated or deleted. When you run this solution, you will notice both Lambda functions in the AWS console. Only the `microservices` and `simulator` functions are regularly active. However, you must not delete the `custom resource helper` function, as it is necessary to manage associated resources.

Step 2. Sign in to the web interface

After the AWS CloudFormation stack is created, the resources for the web interface are deployed. You should also receive an email containing the URL for the web interface, your admin credentials, and a temporary password. Use the following procedure to sign in to the web interface for the first time.

1. Open the email, note your username and temporary password, and select the URL link.
2. On the IoT Device Simulator sign in page, enter the username and temporary password.
3. On the **Change password** page, enter a new password.

> **Note**
>
> Password requirements: minimum of 12 characters, requiring at least one upper case character, one number, and one symbol.

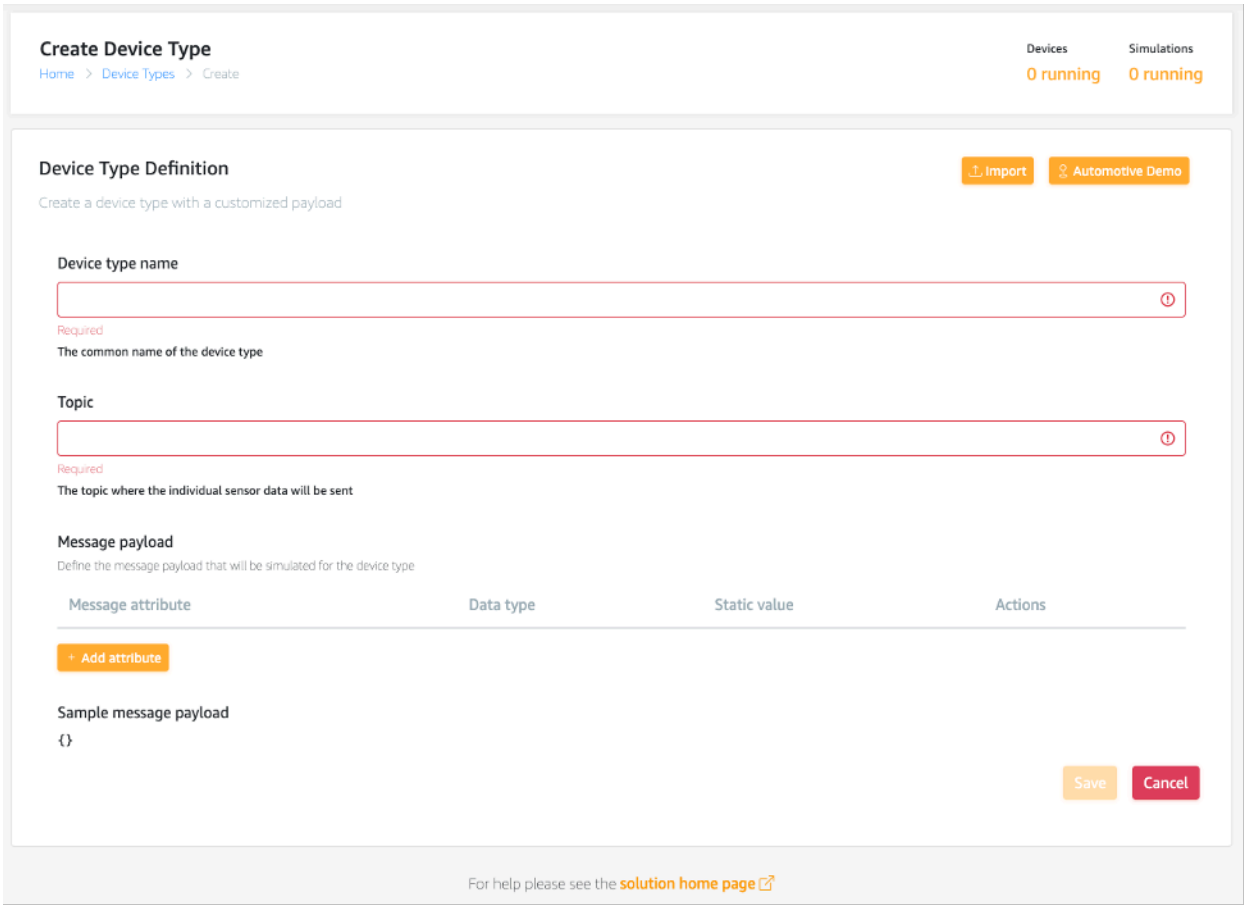
After you sign in to the web interface, follow the remaining steps to create the device types, simulations, and other activities.

Step 3. Create the device types

Use the following procedure to define your device types.

1. Navigate to the **Device Types** page.
2. Choose **Add Device Type**.

On this page, you can either manually add device types or import your device types by uploading a JSON file containing the necessary attributes. To view the structure for the JSON, refer to the Device Type JSON structure section.



Web interface - create a device type

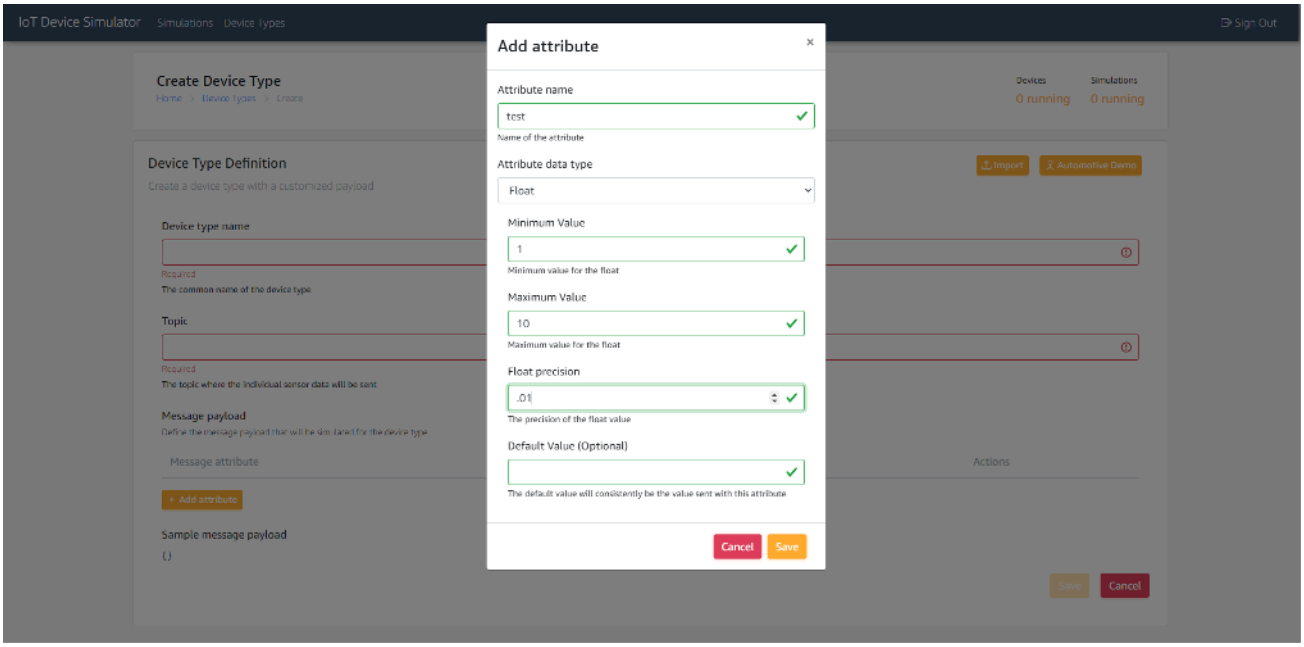
Use the following procedure to manually create a device type.

1. On the **Create Device Type** page, take the following actions:

    1. In the **Device type name** field, enter a name.
    2. In the **Topic** field, enter the topic where the device type will publish to.
    3. Under **Message payload**, choose **Add Attribute**.

        You will define the payload of the device type which will serve as a template for the messages that devices of the device type will send.

    4. In the **Add attribute** dialog box, enter an attribute name and complete the additional data fields as needed. You can specify multiple attributes with different data types to structure your payload.



Web Interface – add an attribute

> **Note**
>
> The fields are dynamic based on the Attribute data type that you select.

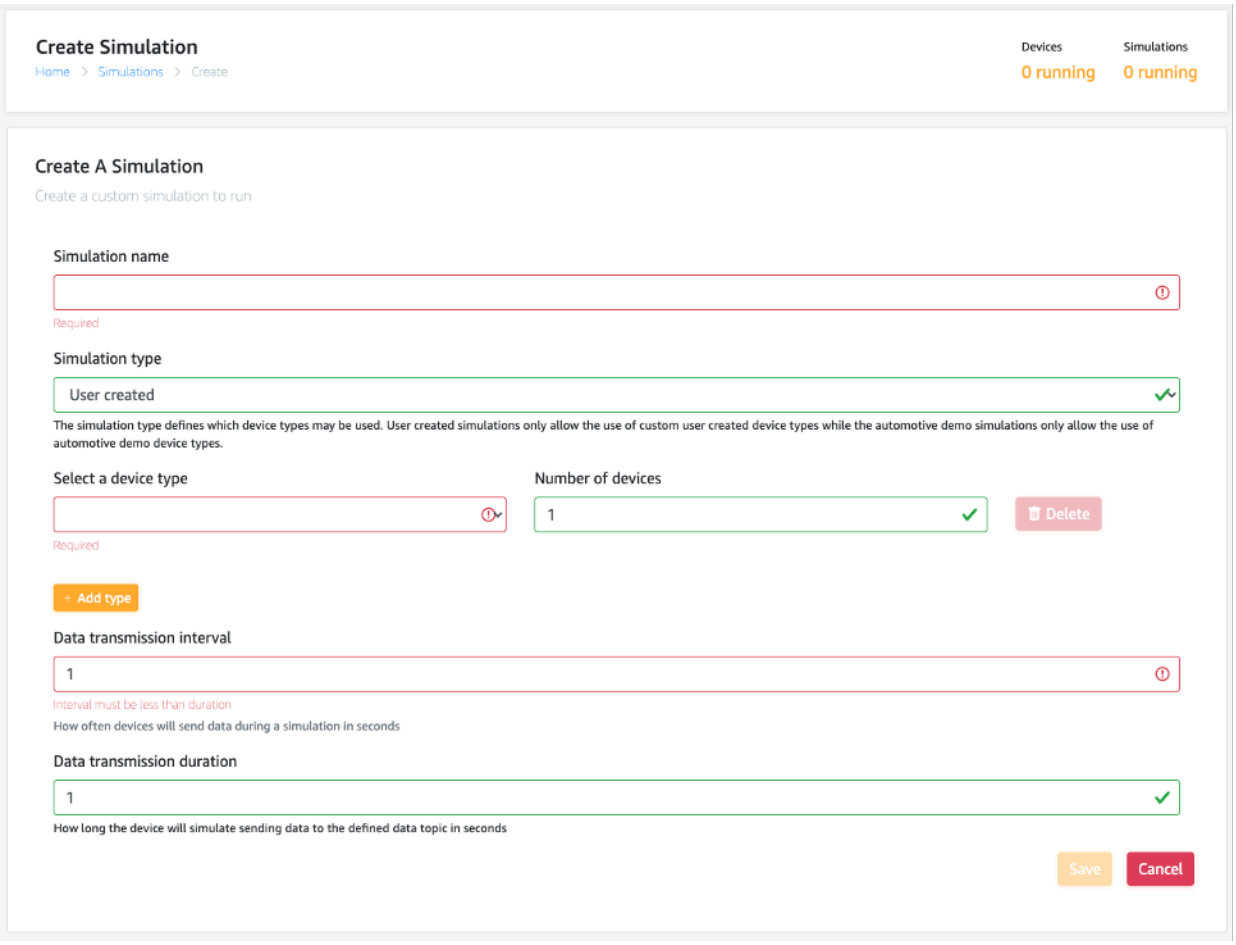
5. Choose **Save**.

2. Enter additional attributes, as needed.
3. Choose **Save**.

Step 4. Create the simulations

Use the following procedure to define the simulations to run.

1. Navigate to the **Simulations** page.
2. Choose **Add Simulation**
3. On the **Create Simulation** page, take the following actions:



Web interface - create a simulation

1. In the **Simulation name** field, enter a name.
2. In the **Select a device type** drop-down menu, select the device type you want to simulate. To select more than one device type, choose **Add type**.
3. In the **Number of devices** drop-down menu, select the number of devices you want to simulate.
4. In the **Data transmission interval** field, enter the interval time that the devices will send data.

> **Note**
>
> Data transmission interval is a key cost driver of AWS IoT Core messaging expense.

5. In the **Data transmission duration** field, enter the length of time that the simulation will run.

4. Choose **Save**.

Step 5. Run and manage the simulations

Use the following procedure to run one or more simulations.

1. Navigate to the **Simulations** page.
2. On the **Simulations** page, select the checkbox for the simulations you want to run.
3. Choose **Start Simulations** to run one or more simulations.

View the simulations

Use the following procedure to view simulations from the **Simulations** page.

1. Select the **View** button that corresponds to the simulation you want to view. This will take you to the **Simulation Details** page.
2. If the simulation is running, you will be able to view the incoming messages in the **Messages** section.

   > **Note**
   >
   > If you are running the automotive simulation demo, you will see a map with the locations of each automotive device.

3. You can filter the messages by device using the filter button for by topic by clicking the corresponding topic.
4. Alternatively, you can also start or stop a simulation from the **Simulation Details** page using the **Start** or **Stop** buttons.

   > **Note**
   >
   > An attribute labeled `_id_` is automatically added to each device in order to identify the device.

Manage the device types

Device types are used to define the type of data your simulated IoT devices will send. The IoT Device Simulator provides a web interface to help you manage your device types, letting you view and edit your device types when needed. You can manage the device types from the **Device Types** page. On the **Device Types** page, you can:

- View your device types. A list of all device types associated with your account is displayed on this page.
- Review and update the details of a specific device type. To make updates, find the applicable device type and select **Edit**. The **Device Type Edit** page shows the device type definition details including the name, the data topic, and the message payload. To make updates, change the applicable values, and choose **Save**. You can remove existing attributes or add new attributes to the message payload.

Manage the simulations

Simulations define which devices will run, how long they will run, and at what intervals they will send messages to the IoT topic. You can create and delete simulations, view the simulation details, and view the messages a simulation is sending.

To manage your simulations, navigate to the **Simulation** page. A list of all simulations associated with your account are displayed.

- To start one or more simulations, select the checkbox next to the simulation you want to start, then choose **Start simulations**. Use the select all checkbox in the table header to select all simulations on the page.
- To stop simulations, choose **Stop simulations**.
- To delete a simulation, select the simulation you want to delete and choose **Delete**.

Post-deployment configuration

This section provides recommendations for configuring the solution after deployment.

Amazon CloudWatch dashboard and alarms

We recommend creating CloudWatch alarms and adding notifications based on the use case. In addition, you can create a CloudWatch dashboard to monitor your resources in a single view.

Additional security considerations

We recommend the following additional security best practices:

- **Customer managed AWS KMS encryption keys** - The solution uses AWS managed AWS KMS keys by default, as these are available at no additional cost. Review your use case to determine if you need to update the solution to use customer managed AWS KMS keys.
- **Activate AWS CloudTrail** - As a recommended security practice, consider enabling AWS CloudTrail in the AWS account where the solution is deployed to log API calls in the AWS account. For more details, see the AWS CloudTrail User Guide.
- **Activate AWS WAF** - As an additional security measure, activate AWS WAF to protect against common web exploits and bots that can affect availability, compromise security, or consume excessive resources.

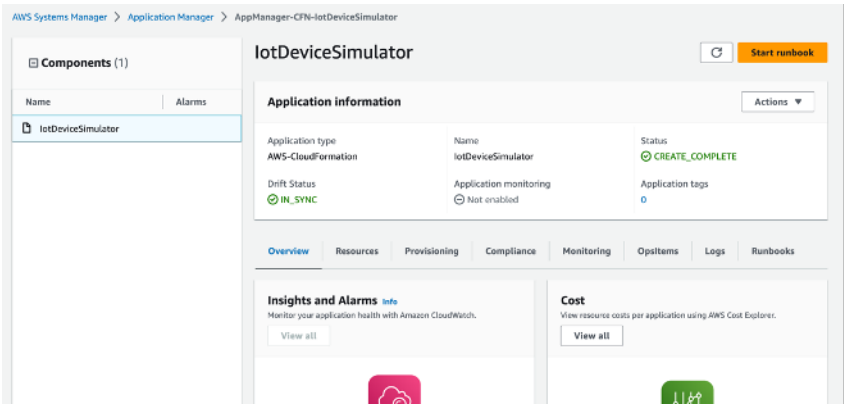
Monitoring the solution with AWS Service Catalog AppRegistry

The IoT Device Simulator solution includes a Service Catalog AppRegistry resource to register the CloudFormation template and underlying resources as an application in both AWS Service Catalog AppRegistry and AWS Systems Manager Application Manager.

AWS Systems Manager Application Manager gives you an application-level view into this solution and its resources so that you can:

- Monitor its resources, costs for the deployed resources across stacks and AWS accounts, and logs associated with this solution from a central location.
- View operations data for the resources of this solution in the context of an application, such as deployment status, CloudWatch alarms, resource configurations, and operational issues.

The following figure depicts an example of the application view for the IoT Device Simulator stack in Application Manager.



IoT Device Simulator stack in Application Manager

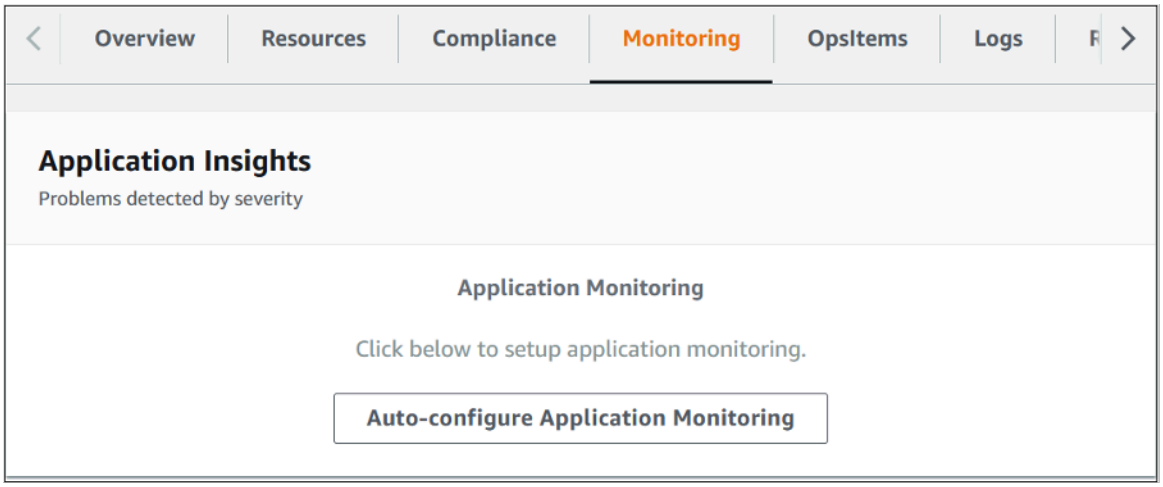
> **Note**
>
> You must activate CloudWatch Application Insights, AWS Cost Explorer, and cost allocation tags associated with this solution. They are not activated by default.

Activate CloudWatch Application Insights

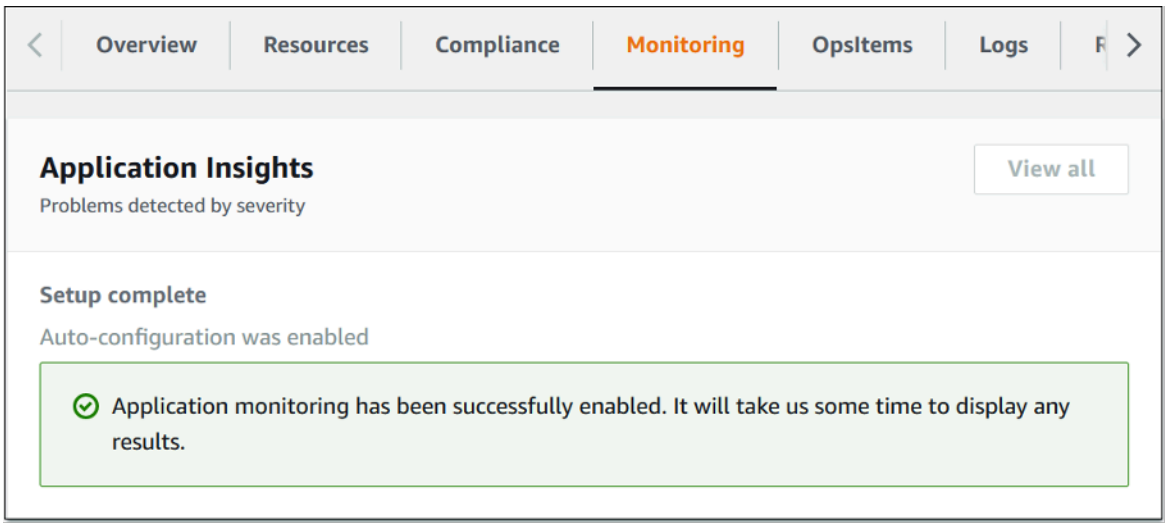
1. Sign in to the Systems Manager console.
2. In the navigation pane, choose **Application Manager**.
3. In **Applications**, choose **AppRegistry applications**.
4. In **AppRegistry applications**, search for the application name for this solution and select it.

The next time you open Application Manager, you can find the new application for your solution in the **AppRegistry application** category.

5. In the **Components** tree, choose the application stack you want to activate.
6. In the **Monitoring** tab, in **Application Insights**, select **Auto-configure Application Monitoring**.



Monitoring for your applications is now activated and the following status box appears:



Activate AWS Cost Explorer

You can see the overview of the costs associated with the application and application components within the Application Manager console through integration with AWS Cost Explorer which must be first activated. Cost Explorer helps you manage costs by providing a view of your AWS resource costs and usage over time. To activate Cost Explorer for the solution:

1. Sign in to the AWS Cost Management console.
2. In the navigation pane, select **Cost Explorer**.
3. On the **Welcome to Cost Explorer** page, choose **Launch Cost Explorer**.

The activation process can take up to 24 hours to complete. Once activated, you can open the Cost Explorer user interface to further analyze cost data for the solution.

Activate cost allocation tags associated with the solution

After you activate Cost Explorer, you must activate a cost allocation tag to see the costs for this solution. The cost allocation tags can only be activated from the management account for the organization. To activate cost allocation tags:

1. Sign in to the AWS Billing and Cost Management console.
2. In the navigation pane, select **Cost Allocation Tags**.
3. On the Cost allocation tags page, filter for the `AppManagerCFNStackKey` tag, then select the tag from the results shown.

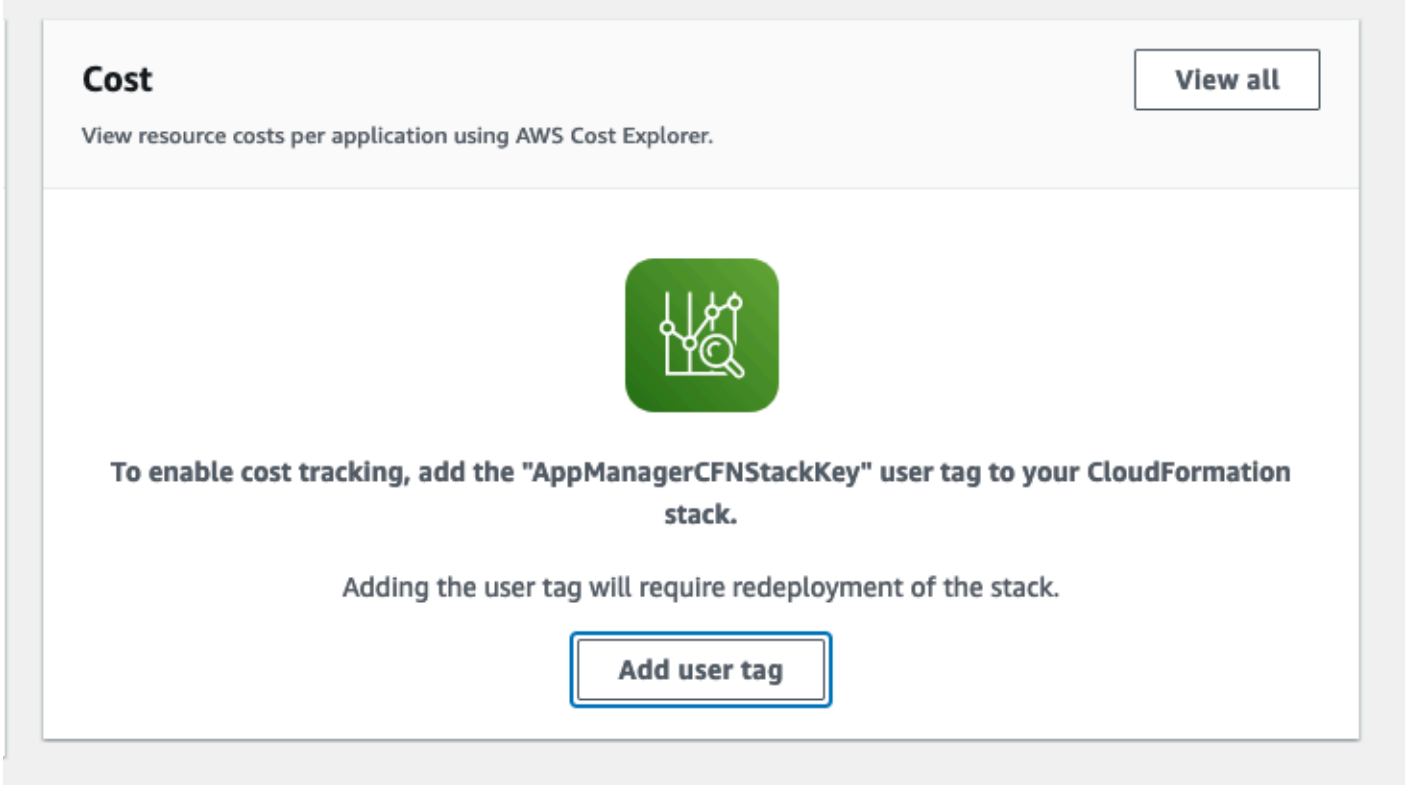
4. Choose **Activate**.

The activation process can take up to 24 hours to complete and the tag data to appear.

Confirm cost tags associated with the solution

After you activate cost allocation tags associated with the solution, you must confirm the cost allocation tags to see the costs for this solution. To confirm cost allocation tags:

1. Sign in to the Systems Manager console.
2. In the navigation pane, choose **Application Manager**.
3. In **Applications**, choose the application name for this solution and select it.
4. In the **Overview** tab, in **Cost**, select **Add user tag**.



5. On the **Add user tag** page, enter `confirm`, then select **Add user tag**.

The activation process can take up to 24 hours to complete and the tag data to appear.

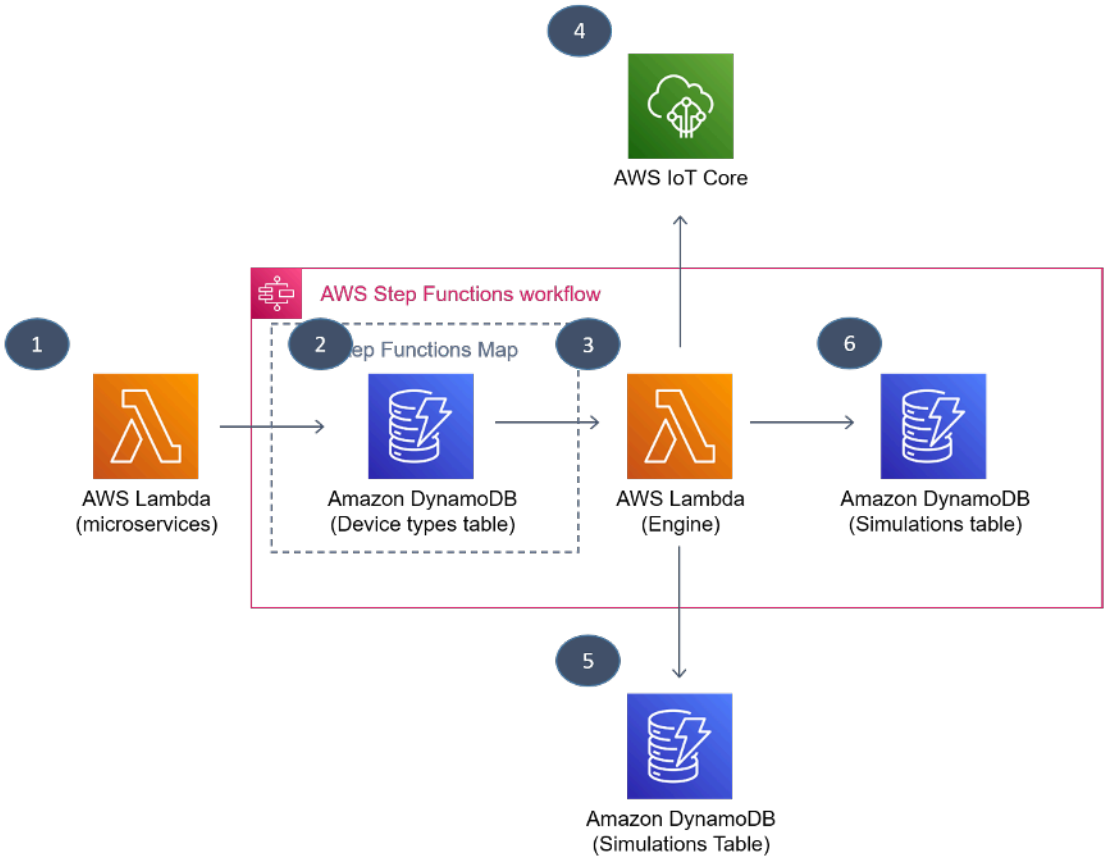
Additional resources

AWS services

- AWS Lambda
- AWS Fargate
- Amazon Elastic Container Service
- Amazon DynamoDB
- Amazon Cognito
- Amazon CloudWatch
- Amazon Virtual Private Cloud
- AWS CloudFormation
- Amazon Simple Queue Service
- AWS IoT
- Amazon Simple Storage Service
- Amazon API Gateway
- AWS Identity and Access Management
- Amazon CloudFront

AWS Step Functions workflow

The following detailed breakdown shows the steps involved in the AWS Step Functions state machine when running a simulation.



Simulation workflow

1. The microservices AWS Lambda function receives the run simulation request and passes the simulation information to AWS Step Functions.
2. The device type information is retrieved from the corresponding DynamoDB table for each device type specified in the simulation.
3. The simulation and device type information are passed to the simulator AWS Lambda function.
4. The simulator AWS Lambda function creates messages and sends them to the AWS IoT Core endpoint. The AWS Lambda function restarts every 15 minutes until it has run for the specified duration.

5. Every 30 seconds, the simulator AWS Lambda function polls the simulator Amazon DynamoDB table to check if the simulation has been stopped externally.
6. When the simulation has finished, the corresponding simulation in the simulations Amazon DynamoDB table is updated.

Automotive demo

Routes

The solution includes an Amazon S3 bucket to store pre-defined routes for the automotive demo. The routes are used to define the path the vehicle will take, and some routes include random triggers that may arise throughout the route, such as high oil temperature. The route locations are defined in the stages. Each stage contains a start and end position. The automotive demo device will move along the stages based on various calculations, such as the speed of the device. The latitude and longitude of the device is reflective of which stage the device is currently navigating. The device location updates each time it moves between a stage. It may take multiple messages before a device moves from one stage to the next.

Running the automotive demo

The solution contains an automotive demo which simulates a connected vehicle. The automotive demo uses one of 18 defined routes that exist in an Amazon Simple Storage Service (Amazon S3) bucket that is created when the solution is launched. It then uses various calculations in the AWS Lambda simulator function to simulate data such as fuel consumption, vehicle speed, acceleration and more. With the automotive demo simulation, you will be able to view a map with the location of the running devices. To use the automotive demo, perform the following steps:

1. Create an Automotive Demo device type.
    1. Navigate to the **Device Type** page.
    2. Choose **Add Device Type**.
    3. Choose **Automotive Demo**.
    4. The payload is auto populated. Enter the rest of the fields such as **Name** and **Topic**.
    5. Choose **Save**.
2. Create a simulation.
    1. Navigate to the **Simulations** page.
    2. Choose **Create Simulation**.
    3. Change the **Simulation Type** field to **Automotive**.
    4. In the dropdown field for devices, you can view your automotive demo device types.

5. Enter the required fields.
6. Choose **Save**.

3. Run the simulation.

    1. Run the simulation from the **Simulations** page by checking the desired simulations, then choose **Start Simulations**.
    2. Alternatively, choose **View** next to the simulation you want to run, then choose **Start** to run the simulation.

4. View the simulation.

    1. Choose **View** next to the simulation you want to view.
    2. If the simulation is running, you can view a map with the locations of the devices, and up to 100 of the most recent messages sent to the IoT topic.

Device type JSON structure

The structure of the JSON to import a device type should contain the following:

Parameter	Required	Type	Description
name	Yes	String	The name of the device type.
topic	Yes	String	The topic to which the device type sends its messages.
payload	Yes	Array of attribute objects	The payload of the device type. See the attribute structure table below to see valid payload contents.

General attribute parameters that are required in all attribute objects:

Parameter	Required	Type	Description
name	Yes	String	The name of the attribute.
type	Yes	String	The type of attribute , must be one of the values listed in the attribute table below.

Parameters that are specific to each attribute:

Type	**Parameters**			
id	**Name**	**Required**	**Type**	**Description**
	charSet	No	String	An alphabet to limit the characters that will be used in the ID.
	length	No	Number	The length of the ID.
	static	No	Boolean	If true, the value will only be generated once per simulation.
bool	**Name**	**Required**	**Type**	**Description**
	default	No	Number	A default value to be used instead of generating a value.
decay	**Name**	**Required**	**Type**	**Description**
	min	Yes	Number	The floor for the decay.
	max	Yes	Number	The starting value for the decay.
	default	No	Number	The static value to be used.
float	**Name**	**Required**	**Type**	**Description**

Type	Parameters			
	min	Yes	Number	The minimum value to be generated.
	max	Yes	Number	The maximum value to be generated.
	precision	Yes	Number	The decimal precision of the float (for example, .01).
	default	No	Number	A default value to be used instead of generating a value.
int	**Name**	**Required**	**Type**	**Description**
	min	Yes	Number	The minimum of the range from which a number will be generated.
	max	Yes	Number	The maximum of the range form which a number will be generated.

Type	Parameters			
	default	No	Number	A default value to be used rather than have one generated.
location	**Name**	**Required**	**Type**	**Description**
	lat	Yes	Number	The center position latitude.
	long	Yes	Number	The center position longitude.
	radius	Yes	Number	The radius (in meters) from the center position for the random coordinates to be generated.
object	**Name**	**Required**	**Type**	**Description**
	payload	Yes	Array	An array of attribute objects.
string	**Name**	**Required**	**Type**	**Description**
	min	Yes	Number	The minimum length of the string.
	max	Yes	Number	The maximum length of the string.

Type	Parameters			
	static	No	Boolean	If true, the value will be generated once per simulation.
	default	No	String	A default value to be used instead of generating a value.
sinusoidal	**Name**	**Required**	**Type**	**Description**
	min	Yes	Number	The minimum value.
	max	Yes	Number	The maximum value.
	default	No	Number	A default value to be used instead of generating a value.
timestamp	**Name**	**Required**	**Type**	**Description**
	tsformat	Yes	String	The timestamp format to be used. Must be one of the following: `default` or `unix`.

Type	Parameters			
	default	No	Number	A default value to be used instead of generating a value.
pickOne	**Name**	**Required**	**Type**	**Description**
	arr	Yes	Array	An array of strings from which a value will be chosen.
	static	No	Boolean	If true, the value will be generated only once per simulation.

Uninstall the solution

You can uninstall the IoT Device Simulator solution from the AWS Management Console or by using the AWS Command Line Interface. You must manually delete the Amazon S3 buckets, and Amazon DynamoDB tables created by this solution. AWS Solutions Implementations do not automatically delete these resources in case you have stored data to retain.

Using the AWS Management Console

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

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 Amazon S3 bucket

This solution is configured to retain the solution-created Amazon S3 buckets if you decide to delete the AWS CloudFormation stack to prevent accidental data loss. After uninstalling the solution, you can manually delete the S3 buckets 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 *<stack-name>* S3 buckets.
4. 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 DynamoDB tables

This solution is configured to retain the solutions Amazon DynamoDB tables if you decide to delete the AWS CloudFormation stack to prevent accidental data loss. After uninstalling the solution, you can manually delete the Amazon DynamoDB tables if you do not need to retain the data. Follow these steps to delete the Amazon DynamoDB tables.

1. Sign in to the Amazon DynamoDB console.
2. Choose **Tables** from the left navigation pane.
3. Select the *<stack-name>* Amazon DynamoDB table you want to delete and choose **Delete table**.

Anonymized data collection

This solution includes an option to send anonymized 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 deployment
- **Timestamp:** Data-collection timestamp
- **Device Type Data:** The type of attributes used when defining a device type payload

  Example data:

```
{
  eventType: "create device type"
  uniquePayloadAttrs: [string, float, sinusoidal]
}
```

- **Simulation Creation Data:** The amount of devices, and the duration of the simulation

  Example data:

```
{
  eventType: "create simulation",
  duration: 120,
  numDevices: 70
}
```

- **Simulation Run Data:** The amount of devices, and the duration of the simulation

  Example data:

```
{
  eventType: "start simulation",
  duration: 120,
  numSimulations: 2
  type: 'autoDemo'
}
```

Note

AWS will own the data gathered via this survey. Data collection will be subject to the AWS Privacy Notice. 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 Automated deployment section of this guide.

Source code

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

Contributors

The following individuals contributed to this document:

- George Lenz
- Ajay Swamy
- Manish Jangid
- Abhishek Patil

Revisions

Date	Change
May 2018	Initial release
December 2018	Added information about the Amazon CloudFront distribution for the static website hosted in the Amazon S3 bucket
March 2019	Added information about Amazon DynamoDB on-demand, the Amazon ECS service-linked role, additional device type attributes and functionality, and managing device types, widgets, and users
December 2019	Added information on support for Node.js update
July 2020	Added cost considerations for Amazon ECS; updated information on support for Node.js update and AWS Lambda
November 2021	Release v3.0.0: Used AWS Cloud Developme nt Kit to create the AWS CloudFormation template; migrated UI to React and simplifie d the UI; added the ability to import/export device types; changed simulator from running on Amazon ECS to AWS Lambda with AWS Step Functions; removed widgets and changed to a device type/simulation workflow to run devices; added Amazon Location Service as map provider; changed the UI and custom resource Lambda to Typescript; and aggregate d automotive demo messages. For more information, refer to the CHANGELOG.md file in the GitHub repository.

Date	Change
April 2023	Release v3.0.1: Upgraded to Node.js 18, added AppRegistry integration, upgraded to AWS CDK v2, upgraded UI build system to React Scripts 5, upgraded to Axios1, and upgraded to use ES2022 for all TypeScript modules. Additionally, mitigated impact caused by new default settings for S3 Object Ownership (ACLs disabled) for all new S3 buckets. For more information, refer to the CHANGELOG.md file in the GitHub repository.
June 2023	Release v3.0.2: Fixed fast-xml-parser vulnerability, and added deployment details in README.md. For more information, refer to the CHANGELOG.md file in the GitHub repository.
August 2023	Release v3.0.3: Upgraded AWS Amplify from v4.x to v5.x, migrated AWS SDK for JavaScript from v2.x to v3.x, library version updates, security patches, UI bug fixes, and removed CDK bootstrap requirement for provisioning the CloudFormation stack. For more information, refer to the CHANGELOG.md file in the GitHub repository. Added information to documentation about CloudWatch alarms creation and additional security recommendations.
October 2023	Release v3.0.4: Updated package versions to resolve security vulnerabilities. For more information, refer to the CHANGELOG.md file in the GitHub repository.

Date	Change
November 2023	Documentation update: Added Confirm cost tags associated with the solution to the Monitoring the solution with AWS Service Catalog AppRegistry section.
February 2024	Release v3.0.5: CDK updates. For more information, refer to the CHANGELOG.md file in the GitHub repository.
April 2024	Release v3.0.6: Updated package versions to resolve security vulnerabilities. For more information, refer to the CHANGELOG.md file in the GitHub repository.

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.

IoT Device Simulator is licensed under the terms of the of the Apache License Version 2.0 available at The Apache Software Foundation.

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.