SAS® Intelligent Monitoring: User's Guide

0.9.0

This document might apply to additional versions of the software. Open this document in SAS Help Center and click on the version in the banner to see all available versions.

Product Overview

Intelligent monitoring is the process of tracking and analyzing sensor data in order to detect anomalous behavior. You can use analysis results to create rules to trigger alerts for specific conditions.

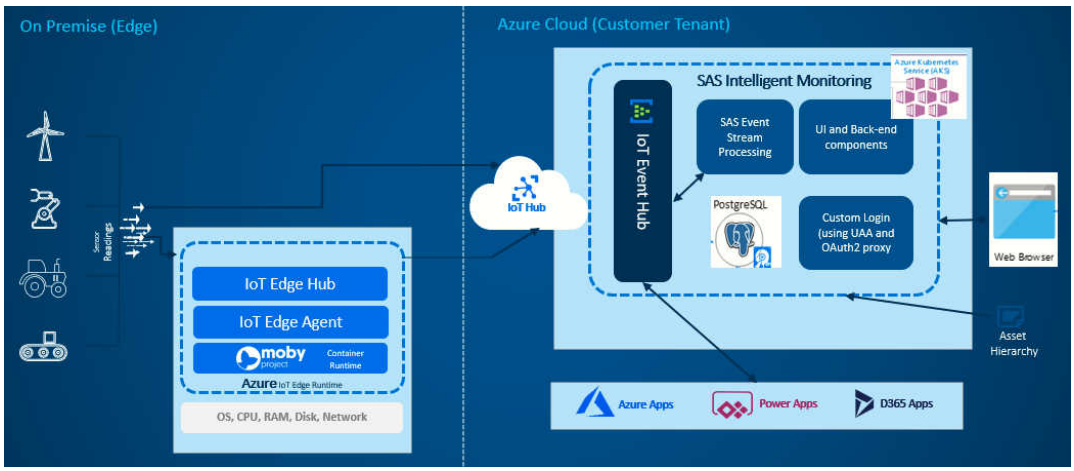
SAS Intelligent Monitoring is an application available from the Azure Marketplace. You can use SAS Intelligent Monitoring to continually monitor your organization's assets and get alerts when pre-defined conditions occur. This can help with effective scheduling of maintenance and maximum asset uptime.

Specifically, you can use SAS Intelligent Monitoring to do the following:

- Create projects targeted at assets to monitor.
- Set up analytics-based scenarios to trigger meaningful alerts in order to take the right action.
- Deploy validated projects to monitor assets in real time.
- Modify projects to keep them running at an optimal level.

SAS Intelligent Monitoring is provisioned in a customer tenant in the Microsoft Azure Cloud

Figure 1 *SAS Intelligent Monitoring Architecture*



Within a Kubernetes environment, SAS Intelligent Monitoring uses SAS Event Stream Processing to communicate with a Microsoft Azure Event Hub. The Event Hub acquires data from Microsoft IoT Hub, which is the cloud gateway for devices to publish IoT data to the Microsoft Azure cloud.

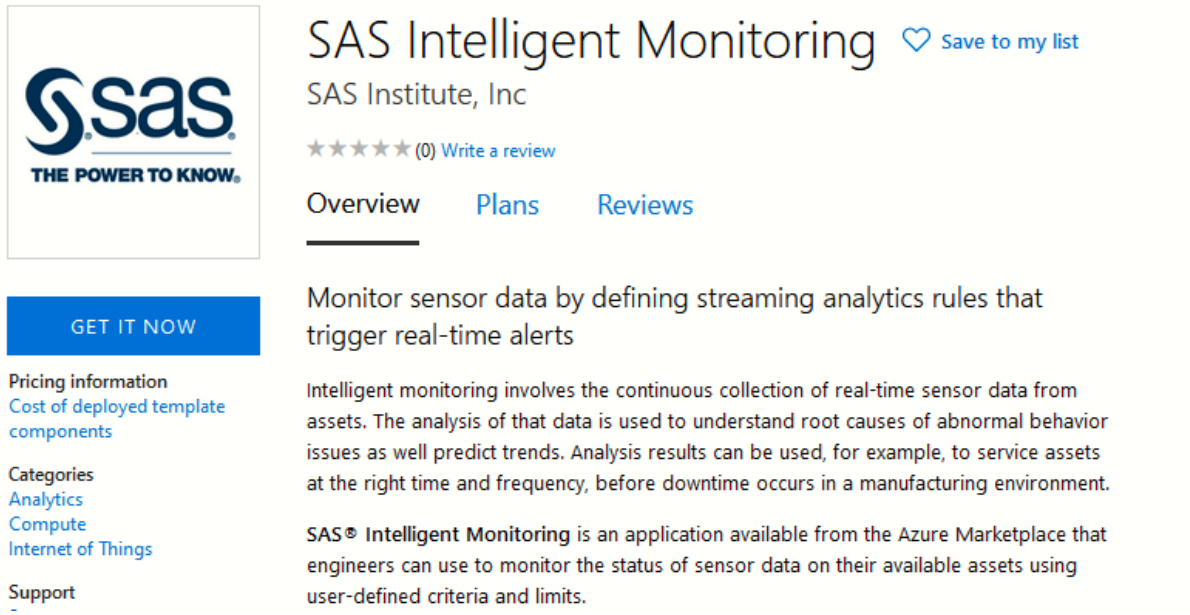
With a relatively light footprint, SAS Intelligent Monitoring uses Kubernetes services to facilitate easy management of output data. This can lead to easy consumption of alerts through downstream services such as PowerBI.

Deploying SAS Intelligent Monitoring in Microsoft Azure

The following steps enable you to deploy SAS Intelligent Monitoring in Microsoft Azure.

> **IMPORTANT** SAS Intelligent Monitoring is currently provided as a limited access preview of the live application. Access to the preview is by invitation only and for a limited time. During the preview phase, there is no cost to use the software. Users must deploy the application in their own Microsoft Azure environment and cover any environment-related costs.

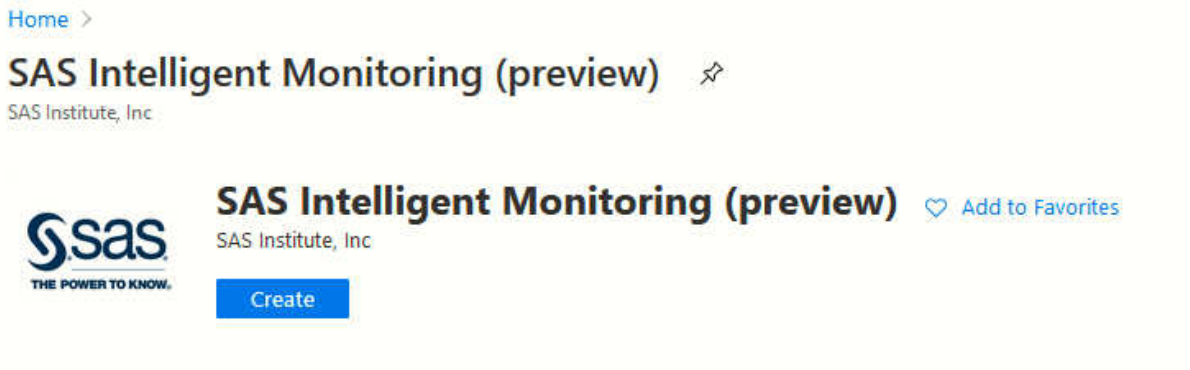
The notification that you received in order to use SAS Intelligent Monitoring provides a link to the application's page in the Azure Marketplace.



1 On the application's page, click **GET IT NOW**.

2 Click **Continue** to agree to the terms of use and privacy policy.

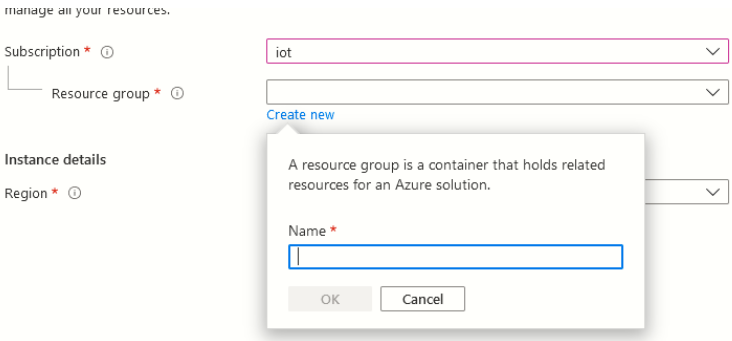
3 On the SAS Intelligent Monitoring (preview) screen. click **Create**.



4 On the **Basics** tab of the Create SAS Intelligent Monitoring screen, do the following:

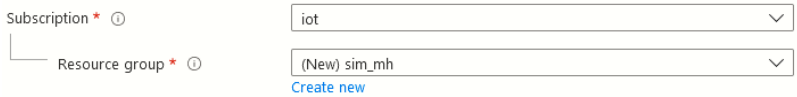
   a Choose the appropriate subscription for the application (for example, **iot**).

   b Choose the appropriate resource group or create a new one.



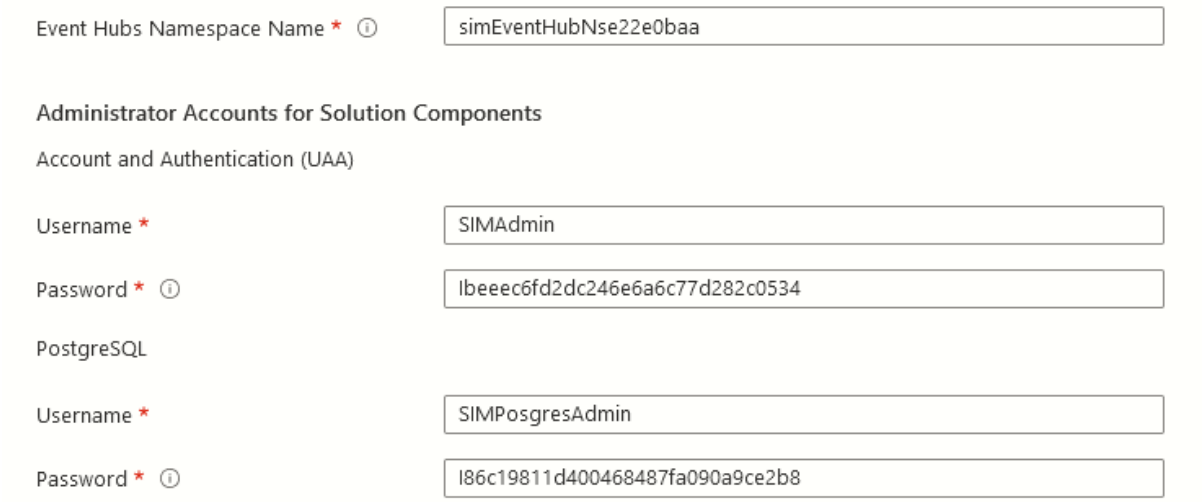
For example, enter **sim_mh** and click **OK**.

4



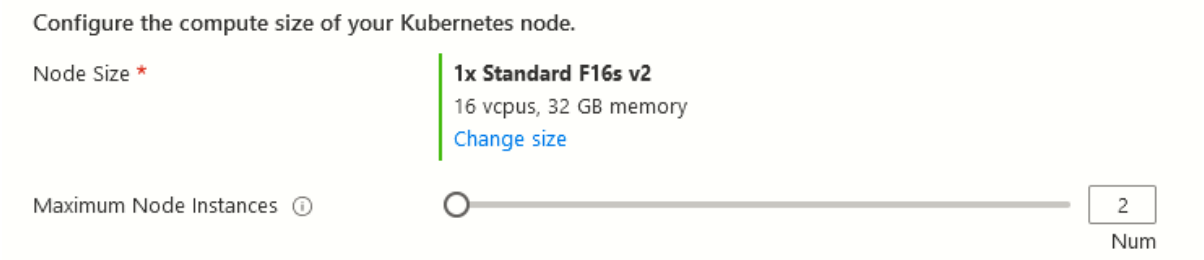
5 At the bottom of the screen, click **Next: Component Details**. On the **Component Details** tab, fields that display component names and details about administrator accounts appear.

> **IMPORTANT** Use the listed **Event Hubs Namespace Name** when you add the Event Hub endpoint to your IoT Hub. Adding this endpoint must occur before you start the data ingestion process .

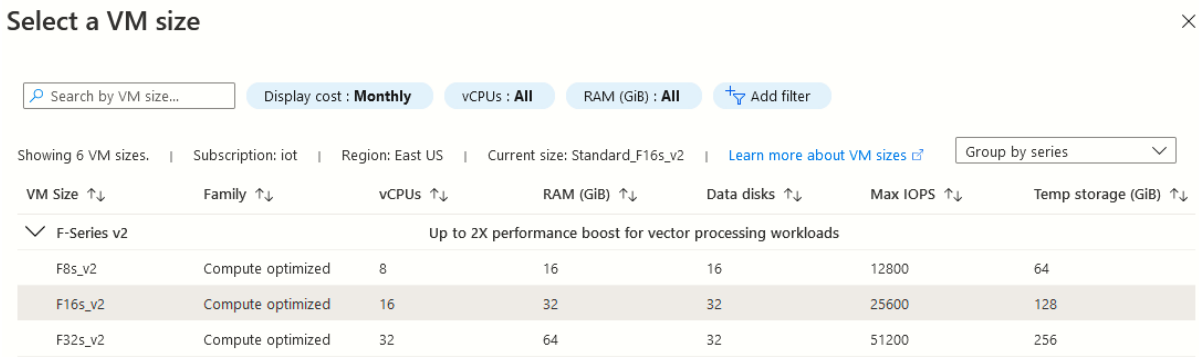


> **IMPORTANT** Record the passwords for the two administrator accounts or change them as needed.

6 At the bottom of the screen, click **Next: Configure Cluster**. On the **Configure Cluster** tab, the values for the preferred cluster node size and the maximum number of node instances appear.



7 (Optional) Click **Change size** to access and select alternative node sizes.



Note: SAS Intelligent Monitoring starts one cluster node when it runs a SAS Event Stream Processing project. When resources on the node are depleted, it starts additional nodes until it reaches the specified **Maximum Node Instances** that you have set. The number of nodes used affects the cost of your deployment in Azure.

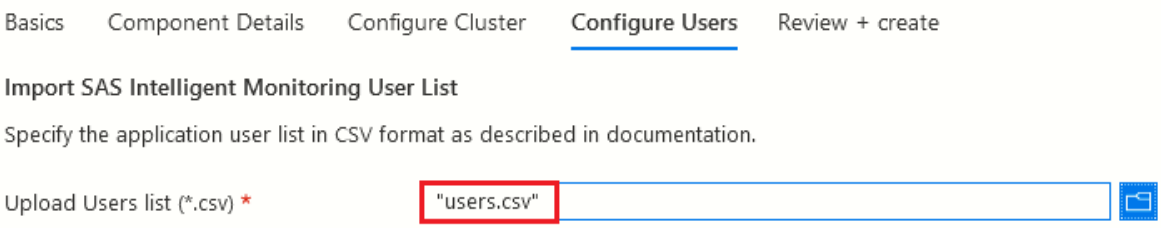
8 At the bottom of the screen click **Next: Configure Users**. On the **Configure Users** tab, click  to upload a CSV file that contains one row for each authorized application user. Each row should consist of the following three fields:

Table 1 *Format of Users List*

User name (for example, `paulmc`)	email address (for example, `paulmc@orion.com`)	password (for example, `MtH%1956`)

IMPORTANT Each row must end with a comma, and each file must end with one empty line.

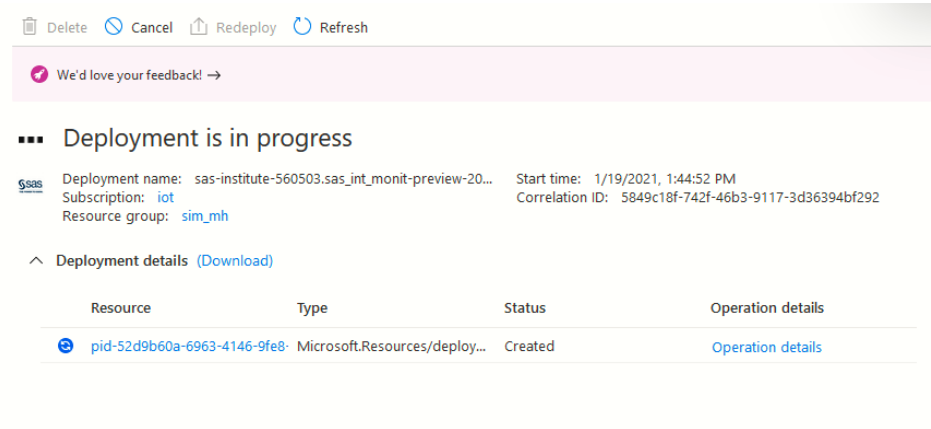
Suppose you uploaded a file named `users.csv`.



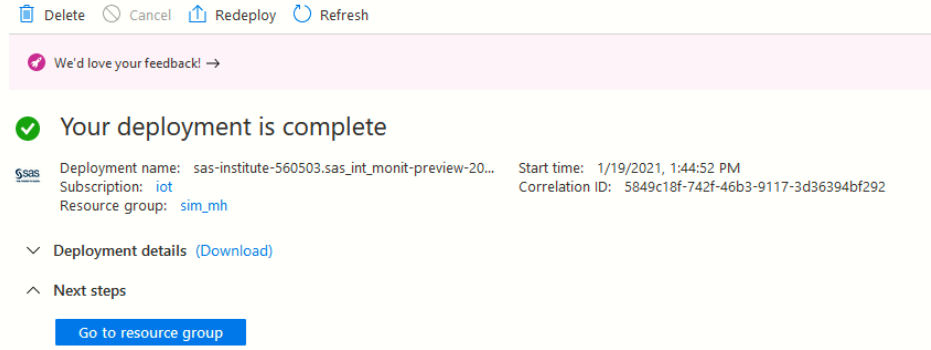
9 Click **Next: Review + create** to initiate the Microsoft Azure validation process.

10 At the bottom of the page, click **Create** to initiate the Microsoft Azure deployment process.

6



After some time, the deployment completes successfully.



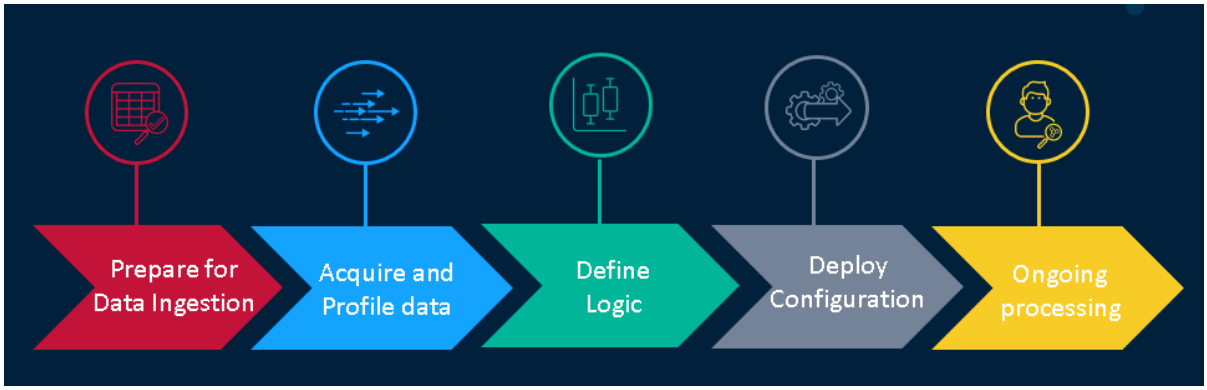
11 On the left navigation pane, click **Outputs** to display the list of Deployment output files.

12 Click the **Copy to clipboard** icon to copy the **applicationURL** value. Paste the value in a browser in order to navigate to the SAS Intelligent Monitoring application.

Workflow

After you have deployed SAS Intelligent Monitoring, the workflow encapsulates a definition of assets and rules for monitoring them.

Figure 2 *SAS Intelligent Monitoring Workflow*



1. Prepare for data ingestion. This involves the following:
   - Linking the Azure IoT Hub collecting data from your device to the Azure Event Hub running in SAS Intelligent Monitoring.
   - Creating a JSON data structure that expresses the format of the data collected from your device.
   - Creating input configuration files for the SAS Intelligent Monitoring application.
2. To better understand the data that streams from your device, you can create a data profile before you set up monitoring rules. Through data injection from the IoT hub, the data profile uses SAS Event Stream Processing to acquire data from sensors on relevant assets.
3. Define a logic block to determine the range of data to be monitored.
4. Create a project that configures an asset hierarchy, a logic block, a range of devices to monitor, and alarm settings. Implement the project through a deployment. There is a one-to-one connection between a project and a deployment.
5. Run a deployment. As the deployment runs, triggered alarms are visible on the landing page. Based on what the alarms indicate, you can update project parameters and redeploy the configuration.

Preparing for Data Ingestion

Overview

SAS Intelligent Monitoring uses an instance of Azure Event Hub to receive data from an externally deployed Azure IoT Hub. The Azure IoT Hub collects data from external devices. The Azure Event Hub must share the same subscription as the Azure IoT Hub.

The data ingestion process involves the following steps:

- Linking the Azure IoT Hub that is collecting data from your device to the Azure Event Hub that is used by SAS Intelligent Monitoring.

> **IMPORTANT** This is the responsibility of the Microsoft Azure administrator and must occur before any subsequent step of the process.

- Creating a JSON data structure that expresses the format of the input data collected from your device.
- Creating input configuration files for the SAS Intelligent Monitoring application.

Creating a JSON Data Structure for Input Data

The input data collected from external devices must conform to a specific format expressed in a JSON structure. The JSON objects within the structure must have a name and must contain the following attribute-value pairs:

- an arbitrary ID name and value for a device to be monitored
- an arbitrary ID name and value for a specific sensor on the device
- a value name and measurement obtained from the specified sensor
- a time stamp label and value

You customize the data structure name and attribute names in the Event Hub Mappings CSV input file .

Here is the JSON data structure:

```
{
      telemetryDataList:[
        {
          devId:"3",
          varId:"2527",
          value:value,
          date:"Jan 21, 2021 01:28:19 PM"

        },
        {
          devId:"4",
          varId:"3328",
          value:value,
          date:"Jan 21, 2021 01:28:19 PM"

        },
      ]
}
```

This data structure has the name `telemetryDataList`. There are two JSON objects within the structure that contain the following specific attribute-value pairs.

- for the device: `devID:"3"` and `devId:"4"`
- for the sensor: `varID:"2527"` and `varId:"3328"`
- for the measurement: `value:value`
- for the time stamp: `date:"Jan 21, 2021 01:28:19 PM"`

IMPORTANT It is critical that all values specified within the JSON objects are STRING.

Notice that each JSON object in the array (that is, each { } object that contains `devID`, `varID`, `value`, and `date`) represents one single-value sensor.

Each device (represented by `devId`) can have multiple single-value sensors. Each sensor for a device is its own JSON object in the array. For example, device `3` (`devId:"3"`) could have a total of ten sensors, each with different `varIds`. This results in a `telemetryDataList` array that contains nine additional JSON objects.

```
{
      telemetryDataList:[
        {
          devId:"3",
          varId:"2527",
          value:value,
          date:"Jan 21, 2021 01:28:19 PM"
        },
        {
          devId:"3",
          varId:"2538",
          value:value,
          date:"Jan 21, 2021 01:28:19 PM"
        },
        {
          devId:"3",
          varId:"2539",
          value:value,
          date:"Jan 21, 2021 01:28:19 PM"
        },
        .
        .
        .
        {
          devId:"4",
          varId:"3328",
          value:value,
          date:"1/13/21"
        },
      ]
}
```

Creating Input Configuration Files

Using information from the JSON objects created, you must manually create the following input configuration files in CSV format:

Table 2 *Input Configuration Files Required to Define an Asset Hierarchy*

File	Description
EventHub Mappings	Associates devices, data formats, and the structure of the data to expect from the IoT Hub. . SAS Event Stream Processing uses the EventHub mapping file to process data from the IoT Hub.
Asset Tags	Associates device categories, subsystems, sensors, and characteristics of the data collected from those sensors. All the sensors that you expect to monitor must be defined in this file.
Asset Hierarchy	Specifies a set of devices from which you obtain data. All devices that you expect to monitor must be listed in this file. The hierarchy is specific to your workplace. It is usually location-based with the hierarchy drilling down several levels (for example, country, state, city, facility, floor, and so on).

You can download templates of these CSV files from the Azure Marketplace. Put the templates on a local file system and edit them as you need.

IMPORTANT Every device listed in the Asset Hierarchy file must have at least one corresponding sensor in the Asset Tags file.

Here is the sample Event Hub Mappings file.

Figure 3 *Sample Event Hub Mappings File*

DEVICE_CAT	DEVICE_SUB_CAT	Data Format (CSV/JSON)	DEVICE_ID	SENSOR_ID	Value	Timestamp
CNC	Controller1	JSON	telemetryDataList[*].devId	telemetryDataList[*].varId	telemetryDataList[*].value	telemetryDataList[*].date
CNC	Controller2	JSON	telemetryDataList[*].devId	telemetryDataList[*].varId	telemetryDataList[*].value	telemetryDataList[*].date

Each row of this CSV file represents the format of the data received for a specific device category.

Table 3 *Fields of an Event Hub Mappings File*

Field	Description
DEVICE_CAT	Specifies a device category. Example: `CNC`
DEVICE_SUB_CAT	Specifies the device subcategory. Example: `Controller1` and `Controller2` **Note:** All devices of the same subcategory have exactly the same sensors.
Data Format (CSV/JSON)	**IMPORTANT** You must specify `JSON`.

Field	Description
DEVICE_ID	Corresponds to the name of the previously defined JSON structure, referring to the `devID` attribute. Example: `telemetryDataList[*].devID` The `devID` value must be in string format. **IMPORTANT** The `devId` attribute name must directly map to the one specified in the JSON structure.
SENSOR_ID	Corresponds to the name of the previously defined JSON structure, referring to the `varID` attribute. Example: `telemetryDataList[*].varID` The `varID` value must be in string format. **IMPORTANT** The `varId` attribute name must directly map to the one specified in the JSON structure.
Value	Corresponds to the name of the previously defined JSON structure, referring to the `value` attribute. Example: `telemetryDataList[*].value` The `value` value must be numerical, either INT or FLOAT. **IMPORTANT** The `value` attribute name must directly map to the one specified in the JSON structure.
Timestamp	Corresponds to the name of the previously defined JSON structure, referring to the `date` attribute. Example: `telemetryDataList[*].date` The timestamp value must be a STRING value that represents the time. The following format is recommended: `'Jan 21, 2021 01:28:19 PM'`. **IMPORTANT** The `date` attribute name must directly map to the one specified in the JSON structure.

Here is the corresponding Asset Tags file.

Figure 4 *Sample Asset Tags File*

DEVICE_CAT	DEVICE_SUB_CAT	DEVICE_SUBSYSTEM	SENSOR_ID	SENSOR_ALIAS (OPT)	DATA_TYPE	UNITS_OF_MEASURE	UPPER_LIMIT (OPT)	LOWER_LIMIT (OPT)
CNC	Controller1	Bearing	2527	value of the axis/spindle current - Axis X	DOUBLE	A	250	50
CNC	Controller1	Bearing	2528	value of the axis/spindle current - Axis Y	DOUBLE	A	250	50
CNC	Controller1	Bearing	2529	value of the axis/spindle current - Axis Z	DOUBLE	A	250	50
CNC	Controller1	Bearing	2551	Actual tool base position Axis X	DOUBLE	mm	250	50
CNC	Controller1	Bearing	2552	Actual tool base position Axis Y	DOUBLE	mm	250	50
CNC	Controller1	Bearing	2553	Actual tool base position Axis Z	DOUBLE	mm	250	50
CNC	Controller2	Bearing	3427	value of the axis/spindle current - Axis X	DOUBLE	A	250	50
CNC	Controller2	Bearing	3328	value of the axis/spindle current - Axis Y	DOUBLE	A	250	50
CNC	Controller2	Bearing	3529	value of the axis/spindle current - Axis Z	DOUBLE	A	250	50
CNC	Controller2	Bearing	3451	Actual tool base position Axis X	DOUBLE	mm	250	50
CNC	Controller2	Bearing	4352	Actual tool base position Axis Y	DOUBLE	mm	250	50
CNC	Controller2	Bearing	5453	Actual tool base position Axis Z	DOUBLE	mm	250	50

Table 4 Fields of an Asset Tags File

Field	Description
DEVICE_CAT	Corresponds to the values of DEVICE_CAT in the Event Hub mapping file.
DEVICE_SUB_CAT	Corresponds to the values of DEVICE_SUB_CAT in the Event Hub mapping file.
DEVICE_SUBSYSTEM	A descriptive label for a specific subsystem on the specified device subcategory.
SENSOR_ID	Corresponds to the values of SENSOR_ID in the Event Hub mapping file.
SENSOR_ALIAS(OPT)	A description of what measurement is collected from the specified sensor.
DATA_TYPE	Specifies the data type of the measurement collected from the specified sensor. This must be expressed as a numerical value.
UNITS_OF_MEASURE	Specifies the units of measurement collected from the specified sensor.
UPPER_LIMIT (OPT)	Specifies the upper limit of values to be accepted from the specified sensor. You can set alerts on these thresholds.
LOWER_LIMIT (OPT)	Specifies the lower limit of values to be accepted from the specified sensor. You can set alerts on these thresholds.

Here is the corresponding Asset Hierarchy file.

Figure 5 Sample Asset Hierarchy File

DEVICE_ID	DEVICE_ALIAS	DEVICE_CAT	DEVICE_SUB_CAT	DEVICE_DESCR	MANUFACTURER	MANUFACTURER_MODEL	INSERVICE_DATE	LOC_LVL_1	LOC_LVL_2	LOC_LVL_3	LOC_LVL_4	LOC_LVL_5
3	SiemensCNCPowerLine1	CNC	Controller1	CNC Siemens 840D PL	Siemens	SRV250	1/1/2020	United States	Ohio	Columbus	XYZ Facility	Floor1
4	SiemensCNCPowerLine2	CNC	Controller2	CNC Siemens 840D PL	Siemens	STP500	1/1/2020	United States	Ohio	Columbus	ABC Plant	Floor2

This hierarchy must contain the following three fields: `DEVICE_ID`, `DEVICE_CAT`, and `DEVICE_SUB_CAT`, which correspond to the values specified in the Event Hub mapping file.

IMPORTANT The values in `DEVICE_ID` must directly match the values provided in the **`devId`** field in the JSON structure.

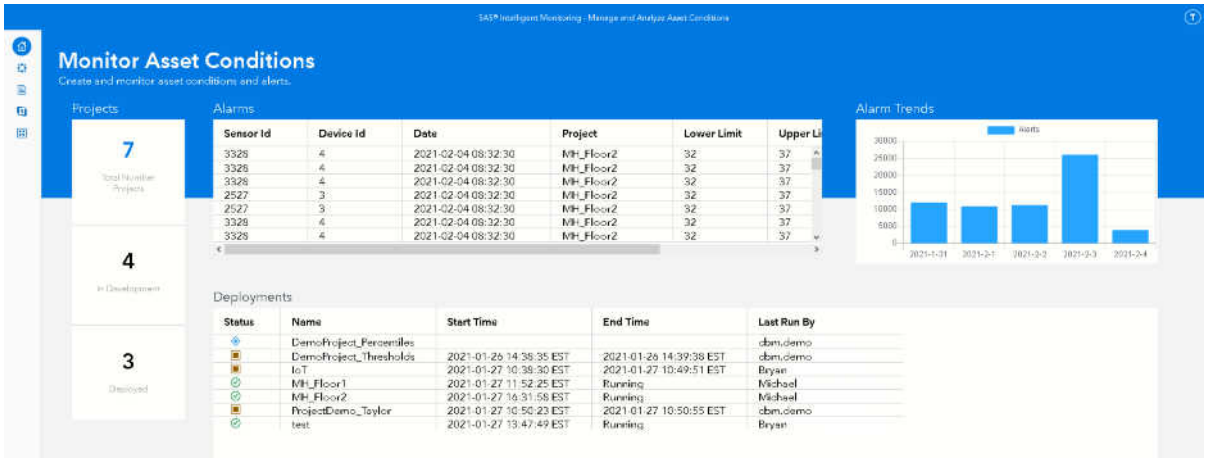
There can be only one root node at the top of the Asset Hierarchy. Thus, the value of LOC_LVL_1 must be the same for each of the listed Devices.

The remainder of the file can be structured as needed in order to reflect the distribution of devices within an industrial setting.

Navigating the User Interface

When you access SAS Intelligent Monitoring, the landing page appears.

Figure 6 *SAS Intelligent Monitoring Landing Page*



The center of the page displays the following:

Table 5 *Landing Page*

Item	Description
Projects	A count of the total number of active projects, projects in development, and deployed projects.
Alarms	A list of the latest alarms.
Alarm Trends	A graphical representation of trends that have appeared over a specified period.
Deployments	A list of deployments, by date and time of update, that shows their status and who last updated them.

The toolbar on left side of the landing page enables you to create and maintain the following items.

Table 6 *Icons on the Toolbar*

Icon	Item
	Landing page
	Import Configuration Files

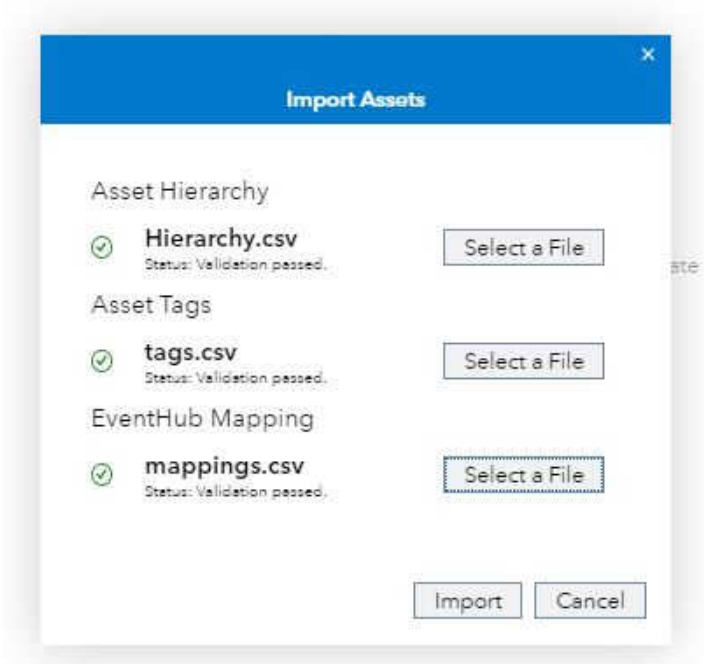
Icon	Item
	Data Profiles
	Projects
	Deployments

Importing Input Configuration Files

1 Click to navigate to the Asset Hierarchy and Sensor Attributes page. If no hierarchy exists, you are prompted to import the assets.

2 Click **Import Assets** to import the CSV files you had previously preparedinto the system.

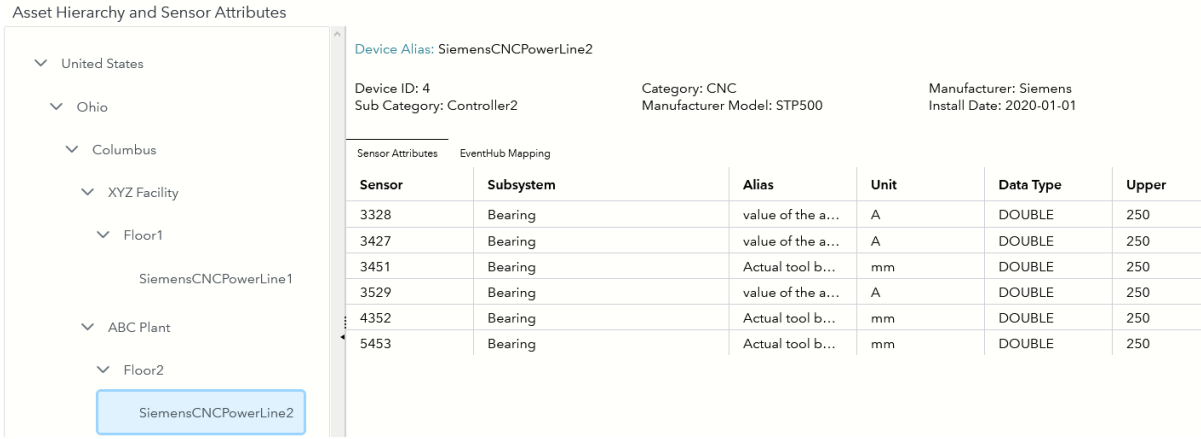
3 In the Import Assets dialog box, click **Select a File** to import CSV files that you had imported to your local file system. After you have selected each of the required files, you are prompted to import them into the system.



4 Click **Import**.

Note: Only one hierarchy can exist at a time, and every leaf node on a hierarchy is a device.

5 Select a device to see details about it. A device almost always contains multiple sensors. For example, in the figure below, the selected power line device contains six sensors from which data is collected.



A specific device has the same sensors regardless of location. For example, SiemensCNCPowerLine2 contains sensors 3328, 3427, and so on regardless of how many floors you place it.

You can click  to change the **Device Alias**.

Creating a Data Profile

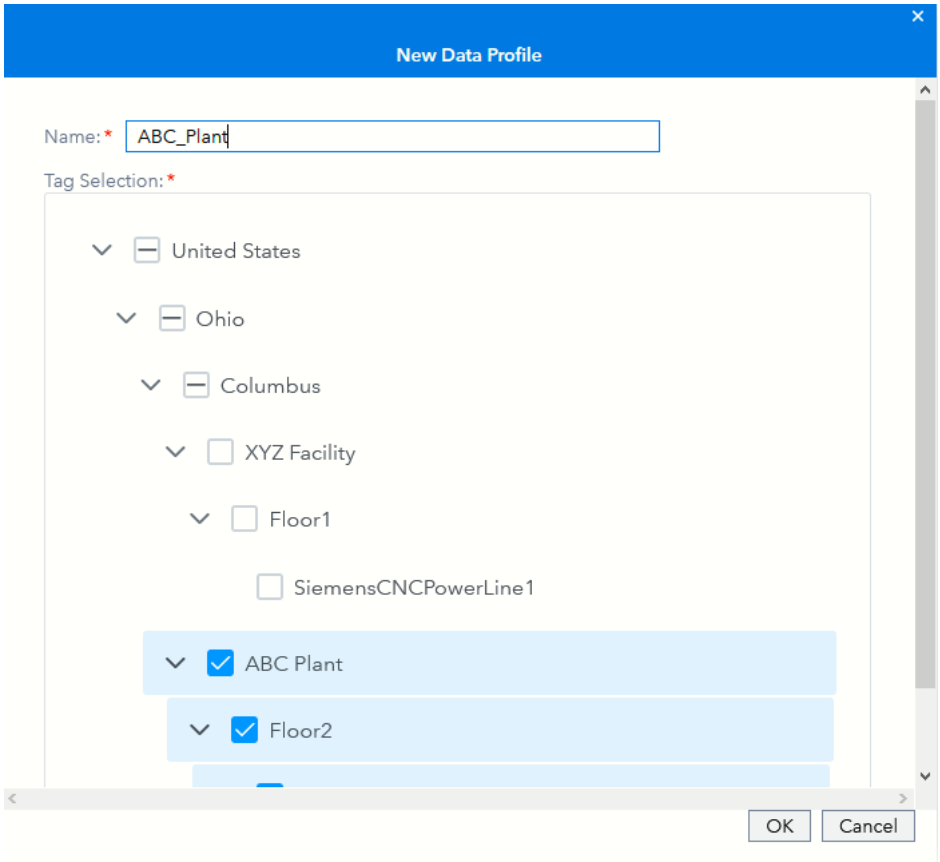
To better understand the data that streams from your device, you can create a data profile before you set up monitoring rules. The data profile collects sensor data from a selected set of devices for a time period that you specify. It generates a standard set of statistics to enable a better understanding of the data. Those statistics can help you create more targeted monitoring rules.

The first time that you click  , the new data profile page appears.

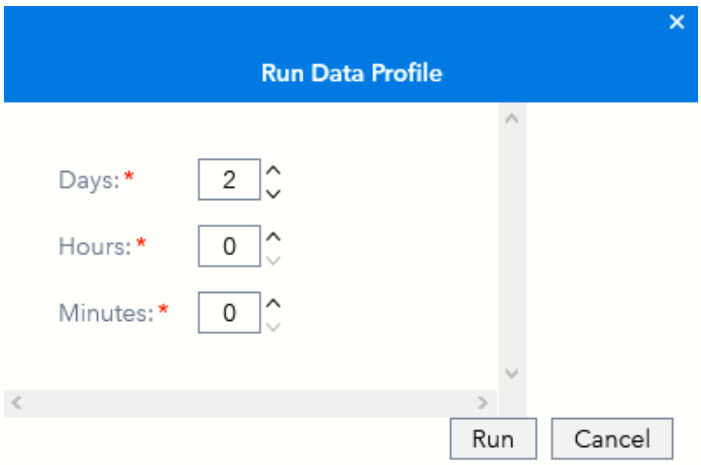
1 Click **New Data Profile**.

2 In the **New Data Profile** dialog box, name the data profile, select your desired subset, and click **OK**.

Figure 7 Create a Data Profile

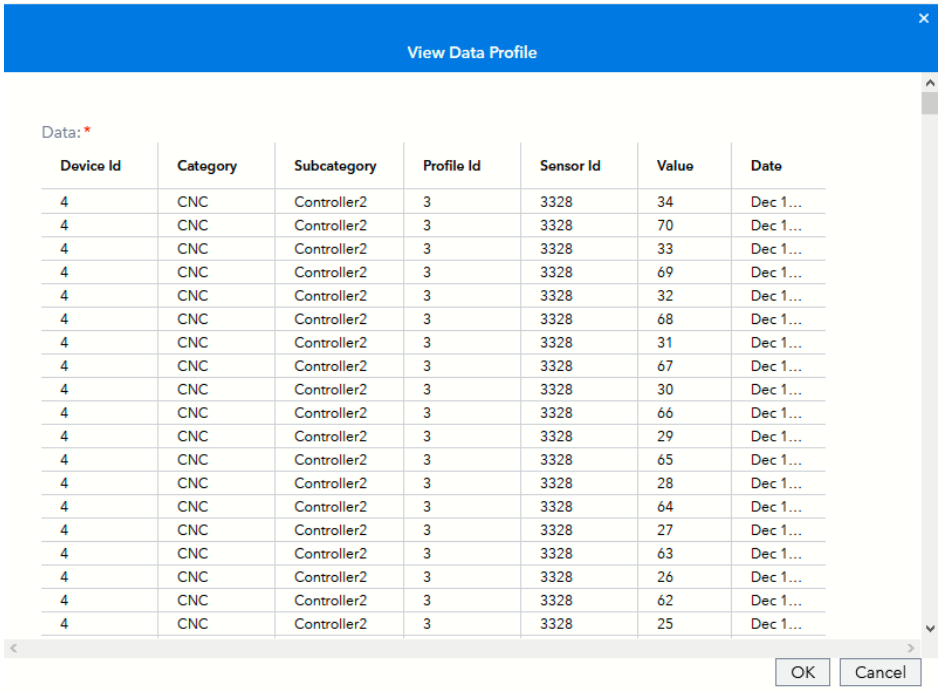


3 After the data profile is created, click ▷. This prompts you to specify an interval to run the data profile.



4 After you specify the interval, click **Run** to initiate the data profile. After a short interval, the ✓ icon appears.

5 After the data profile finishes its run, click to see the generated data. Each sensor has one numerical value.

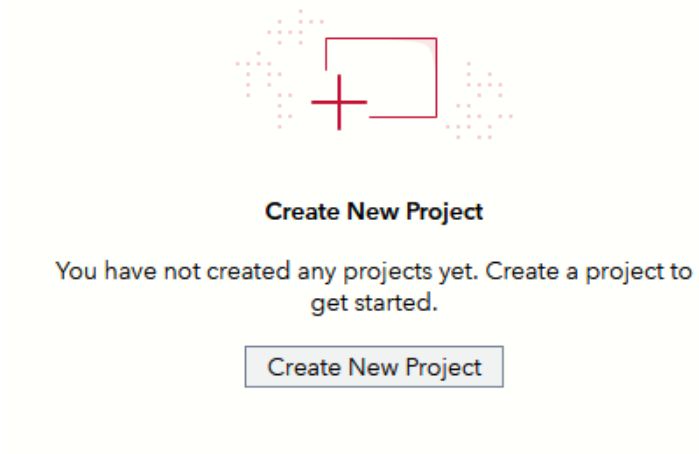


Creating and Maintaining Projects

A project is a workspace that assembles your selected devices and tags with your chosen monitoring logic. You deploy a project in order to apply it to a live data stream from your devices.

To create a project, click  to navigate to the Project Configuration page.

If no projects have been created, then you are prompted to create a new one.



If projects exist, then a list of them appears. Click  to create a new one.

Either action initiates a wizard that guides you through the process of setting the following:

- logic blocks or statistical process controls
- relevant asset tags
- thresholds to be applied to the appropriate device
- alarms

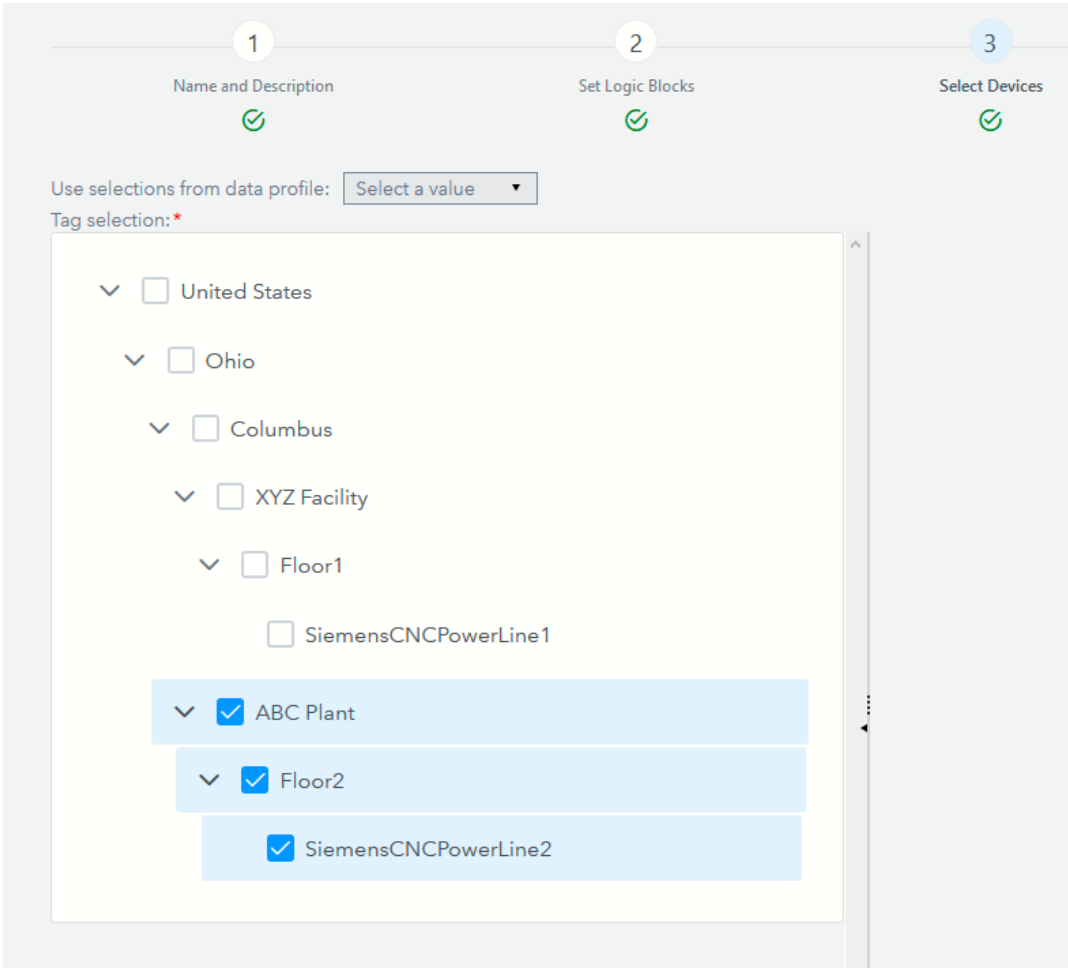
1 Enter a project name and, if wanted, a project description.

2 Set logic blocks. You can use one of two standard logic blocks or you can apply one or a combination of eight Shewhart rules.

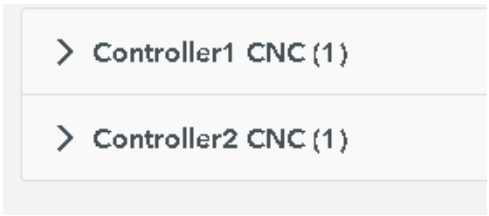
Table 7 Set Logic Block Selections

Selection	Description	Where to Adjust Values
Use standard logic blocks: **Use values from imported assets**	Uses the upper and lower limit thresholds that are imported from your asset hierarchy files.	You can adjust these values on the Select Sensors and Manage Thresholds page.
Use standard logic blocks: **Use data profile values** **Note:** You can use values only from a stopped data profile.	Select a **Data Profile**. Then set a **Lower Threshold** and an **Upper Threshold**.	You can adjust these values on theSelect Sensors and Manage Thresholds page
Use statistical process controls	Provides eight standard Shewhart rules.	You cannot adjust these values.

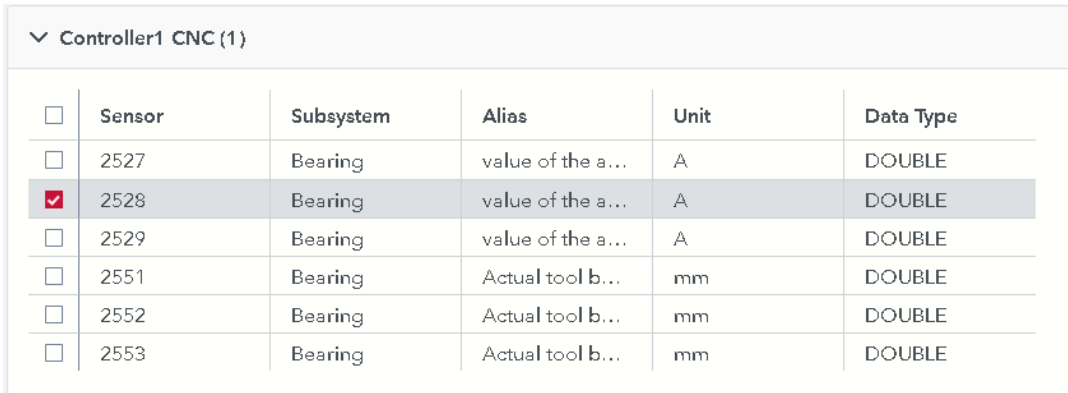
3 Unless you have chosen the logic block **Use standard logic blocks**: **Use values from imported assets**, select a specific set of devices here.

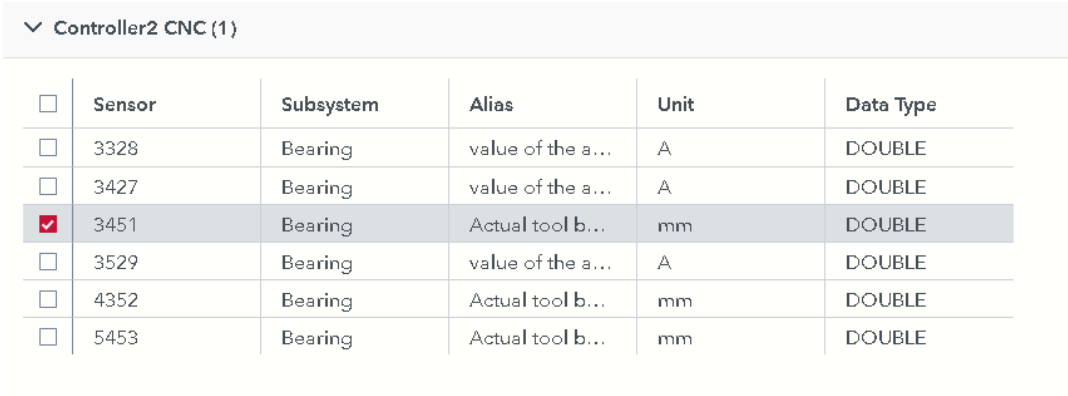


4 Select the desired set of sensors to monitor. You must select at least one sensor from each drop-down. For example, suppose you have two drop-downs: **Controller1 CNC (1)** and **Controller2 CNC (1)**.

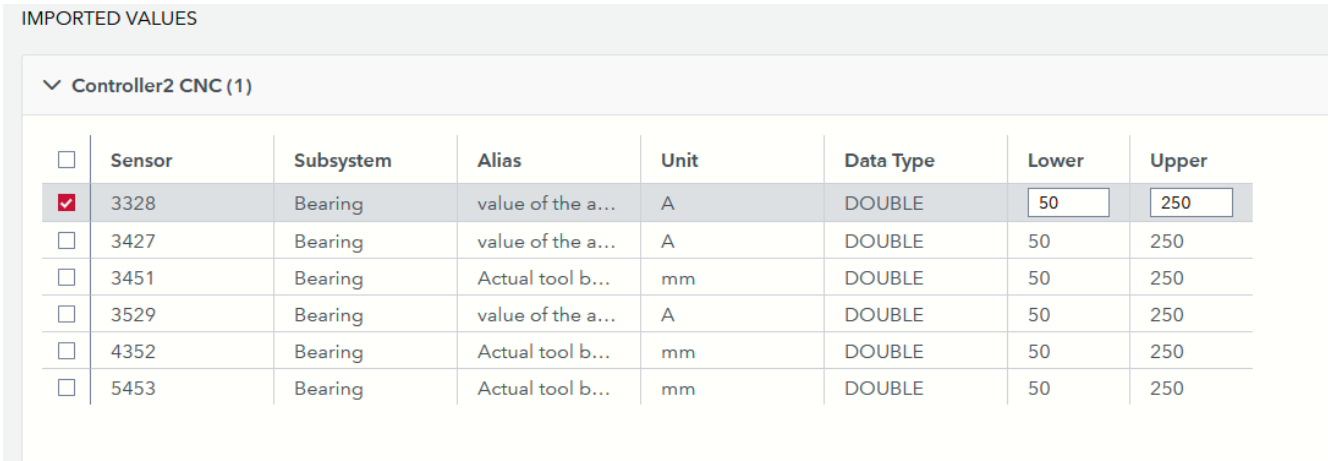


You must select at least one sensor from each.



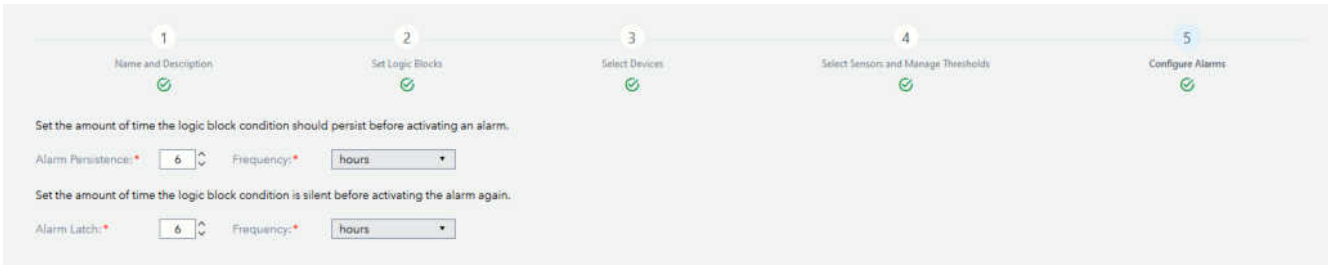


Depending on the logic that you selected, you can adjust upper and lower limits per sensor.



The lower value that you specify must be less than the upper value that you specify.

5 Set alarm parameters.



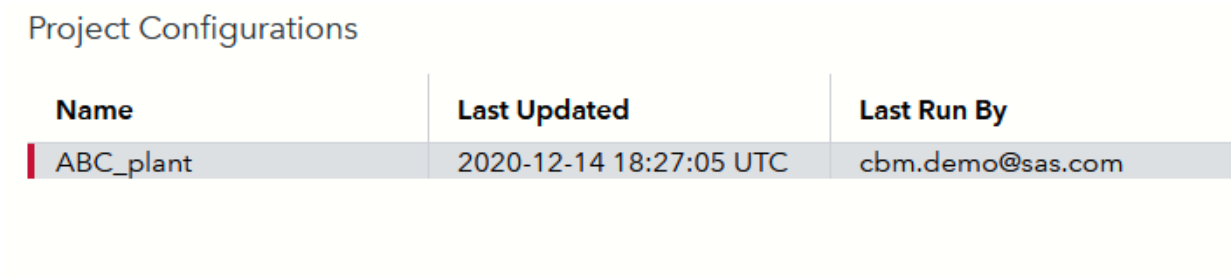
For Alarm Persistence, specify a period of time over which to apply your logic block to data streaming from the sensors. This should be an interval over which you think that data meeting the condition of the logic block constitutes an event worth investigating.

For Alarm Latch, specify a period of time to let the logic block lapse before reactivating the alarm. This should be a function of how long it takes to address a discovered issue.

> **IMPORTANT** This selection is inoperative for Release 0.9.0.

6 Click Finish.

The project definition appears on the Project Configurations page.



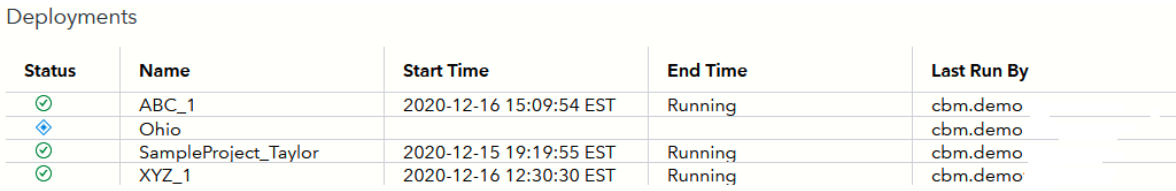
The system automatically creates a deployment that corresponds to the project.

7 Click to navigate to the Deployments page.

Creating and Running Deployments

After you add a project, a deployment for it is automatically populated.

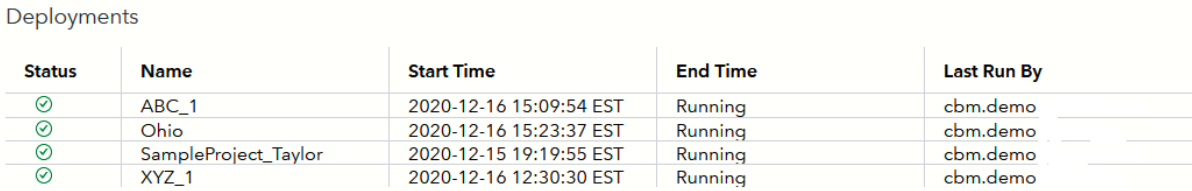
1 Click to navigate to the Deployments page. Any deployments that are ready to run ( ), already running ( ), and stopped ( ) are displayed.



2 Select a deployment ready to run.

3 Click to run the deployment.

After a short time, the deployment is displayed with the running **Status**.



SAS Event Stream Manager uses the ESP operator to spin up an ESP server that runs the associated project. The project then does the following:

- attaches to the Event Hub
- consumes data as specified
- analyzes the data as specified in the ESP project
- pushes results into a database

- pushes information onto the Event Hub
- continues to monitor the IoT Hub as more data comes in

You can also get the status of a deployment in the form of an alarm or you can check status on the landing page.

After a deployment is running, you can stop it at any time.

1 Click the running deployment.

2 Click ■. After a period of time, the deployment shows a stopped status (▣).