Contents

[**Deploy DiversityLeadershipIdentifier Azure Logic App templates** 1](#_Toc103618740)

[**Deploy through Azure portal** 1](#_Toc103618741)

[**Deploy with Visual Studio** 1](#_Toc103618742)

[**Deploy with Azure PowerShell** 2](#_Toc103618743)

[**Deploy with Azure CLI** 2](#_Toc103618744)

[**Deploy with Azure DevOps** 2](#_Toc103618745)

[Review runs history 4](#_Toc103618746)

[Review trigger history 11](#_Toc103618747)

[Set up monitoring alerts 14](#_Toc103618748)

**Deploy DiversityLeadershipIdentifier Azure Logic App templates**

Azure logic app can be deployed from templates in these ways:

* [Azure portal](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-deploy-azure-resource-manager-templates#portal)
* [Visual Studio](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-deploy-azure-resource-manager-templates#visual-studio)
* [Azure PowerShell](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-deploy-azure-resource-manager-templates#powershell)
* [Azure CLI](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-deploy-azure-resource-manager-templates#cli)
* [Azure Resource Manager REST API](https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/deploy-rest)
* [Azure DevOps](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-deploy-azure-resource-manager-templates#azure-pipelines)

**Deploy through Azure portal**

To automatically deploy a logic app template to Azure, you can choose the following **Deploy to Azure** button, which signs you in to the Azure portal and prompts you for information about your logic app. You can then make any necessary changes to the logic app template or parameters.

[Deploy to Azure](https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Fraw.githubusercontent.com%2FAzure%2Fazure-quickstart-templates%2Fmaster%2Fquickstarts%2Fmicrosoft.logic%2Flogic-app-create%2Fazuredeploy.json)

For example, you're prompted for the following information after you sign in to the Azure portal:

* Azure subscription name
* Resource group that you want to use
* Logic app location
* The name for your logic app
* A test URI
* Acceptance of the specified terms and conditions

For more information, see these topics:

* [Overview: Automate deployment for logic apps with Azure Resource Manager templates](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-azure-resource-manager-templates-overview)
* [Deploy resources with Azure Resource Manager templates and the Azure portal](https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/deploy-portal)

**Deploy with Visual Studio**

To deploy a logic app template from an Azure Resource Group project that you created by using Visual Studio, follow these [steps to manually deploy your logic app](https://docs.microsoft.com/en-us/azure/logic-apps/quickstart-create-logic-apps-with-visual-studio#deploy-logic-app-to-azure) to Azure.

**Deploy with Azure PowerShell**

To deploy to a specific *Azure resource group*, use the following command:

PowerShellCopy

New-AzResourceGroupDeployment -ResourceGroupName <Azure-resource-group-name> -TemplateUri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/quickstarts/microsoft.logic/logic-app-create/azuredeploy.json

For more information, see these topics:

* [Deploy resources with Resource Manager templates and Azure PowerShell](https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/deploy-powershell)
* [New-AzResourceGroupDeployment](https://docs.microsoft.com/en-us/powershell/module/azurerm.resources/new-azurermresourcegroupdeployment)

**Deploy with Azure CLI**

To deploy to a specific *Azure resource group*, use the following command:

Azure CLICopy

az deployment group create -g <Azure-resource-group-name> --template-uri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/quickstarts/microsoft.logic/logic-app-create/azuredeploy.json

For more information, see these topics:

* [Deploy resources with Resource Manager templates and Azure CLI](https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/deploy-cli)
* [az deployment group create](https://docs.microsoft.com/en-us/cli/azure/deployment/group#az-deployment-group-create)

**Deploy with Azure DevOps**

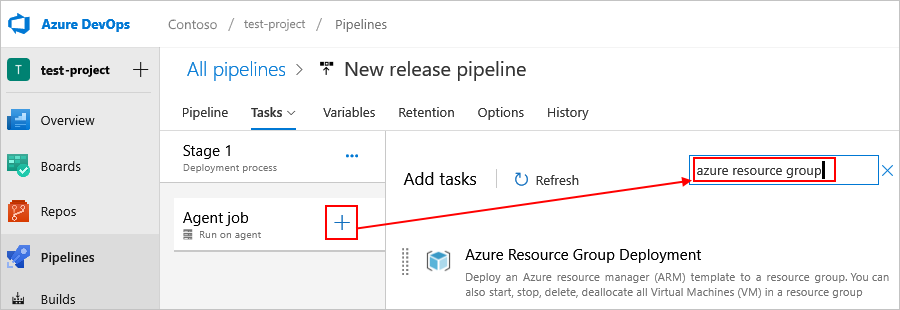
To deploy logic app templates and manage environments, teams commonly use a tool such as [Azure Pipelines](https://docs.microsoft.com/en-us/azure/devops/pipelines/get-started/what-is-azure-pipelines) in [Azure DevOps](https://docs.microsoft.com/en-us/azure/devops/user-guide/what-is-azure-devops-services). Azure Pipelines provides an [Azure Resource Group Deployment task](https://github.com/Microsoft/azure-pipelines-tasks/tree/master/Tasks/AzureResourceGroupDeploymentV2) that you can add to any build or release pipeline. For authorization to deploy and generate the release pipeline, you also need an Azure Active Directory (AD) [service principal](https://docs.microsoft.com/en-us/azure/active-directory/develop/app-objects-and-service-principals). Learn more about [using service principals with Azure Pipelines](https://docs.microsoft.com/en-us/azure/devops/pipelines/library/connect-to-azure).

For more information about continuous integration and continuous deployment (CI/CD) for Azure Resource Manager templates with Azure Pipelines, see these topics and samples:

* [Integrate Resource Manager templates with Azure Pipelines](https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/add-template-to-azure-pipelines)
* [Tutorial: Continuous integration of Azure Resource Manager templates with Azure Pipelines](https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/deployment-tutorial-pipeline)
* [Sample: Orchestrate Azure Pipelines by using Azure Logic Apps](https://github.com/Azure-Samples/azure-logic-apps-pipeline-orchestration)
* [Sample: Connect to Azure Storage accounts from Azure Logic Apps and deploy with Azure Pipelines in Azure DevOps](https://github.com/Azure-Samples/azure-logic-apps-deployment-samples/tree/master/storage-account-connections)
* [Sample: Connect to Azure Service Bus queues from Azure Logic Apps and deploy with Azure Pipelines in Azure DevOps](https://github.com/Azure-Samples/azure-logic-apps-deployment-samples/tree/master/service-bus-connections)
* [Sample: Set up an Azure Functions action for Azure Logic Apps and deploy with Azure Pipelines in Azure DevOps](https://github.com/Azure-Samples/azure-logic-apps-deployment-samples/tree/master/function-app-actions)
* [Sample: Connect to an integration account from Azure Logic Apps and deploy with Azure Pipelines in Azure DevOps](https://github.com/Azure-Samples/azure-logic-apps-deployment-samples/tree/master/integration-account-connections)

Here are the general high-level steps for using Azure Pipelines:

1. In Azure Pipelines, create an empty pipeline.
2. Choose the resources you need for the pipeline, such as your logic app template and template parameters files, which you generate manually or as part of the build process.
3. For your agent job, find and add the **Azure Resource Group Deployment** task.



1. Configure with a [service principal](https://docs.microsoft.com/en-us/azure/devops/pipelines/library/connect-to-azure).
2. Add references to your logic app template and template parameters files.
3. Continue to build out steps in the release process for any other environment, automated test, or approvers as needed.

**Monitor DiversityLeadershipIdentifier Azure Logic Apps**

You can check that workflow's run status, [runs history](https://docs.microsoft.com/en-us/azure/logic-apps/monitor-logic-apps#review-runs-history), [trigger history](https://docs.microsoft.com/en-us/azure/logic-apps/monitor-logic-apps#review-trigger-history), and performance. To get notifications about failures or other possible problems, set up [alerts](https://docs.microsoft.com/en-us/azure/logic-apps/monitor-logic-apps#add-azure-alerts). For example, you can create an alert that detects "when more than five runs fail in an hour."

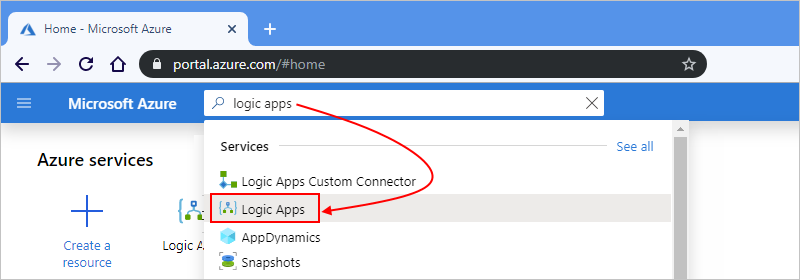
For real-time event monitoring and richer debugging, set up diagnostics logging for your logic app by using [Azure Monitor logs](https://docs.microsoft.com/en-us/azure/azure-monitor/overview). This Azure service helps you monitor your cloud and on-premises environments so that you can more easily maintain their availability and performance. You can then find and view events, such as trigger events, run events, and action events. By storing this information in [Azure Monitor logs](https://docs.microsoft.com/en-us/azure/azure-monitor/logs/data-platform-logs), you can create [log queries](https://docs.microsoft.com/en-us/azure/azure-monitor/logs/log-query-overview) that help you find and analyze this information. You can also use this diagnostic data with other Azure services, such as Azure Storage and Azure Event Hubs. For more information, see [Monitor logic apps by using Azure Monitor](https://docs.microsoft.com/en-us/azure/logic-apps/monitor-logic-apps-log-analytics).

## Review runs history

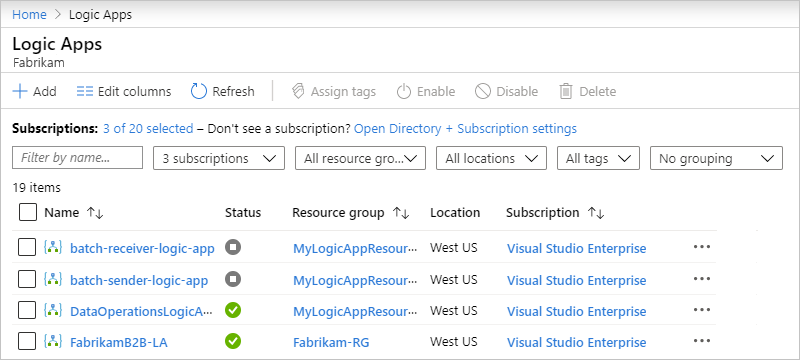
Each time that the trigger fires for an item or event, the Logic Apps engine creates and runs a separate workflow instance for each item or event. By default, each workflow instance runs in parallel so that no workflow has to wait before starting a run. You can review what happened during that run, including the status for each step in the workflow plus the inputs and outputs for each step.

1. In the [Azure portal](https://portal.azure.com/), find and open your logic app workflow in the designer.

To find your logic app, in the main Azure search box, enter logic apps, and then select **Logic apps**.



The Azure portal shows all the logic apps that are associated with your Azure subscriptions. You can filter this list based on name, subscription, resource group, location, and so on.

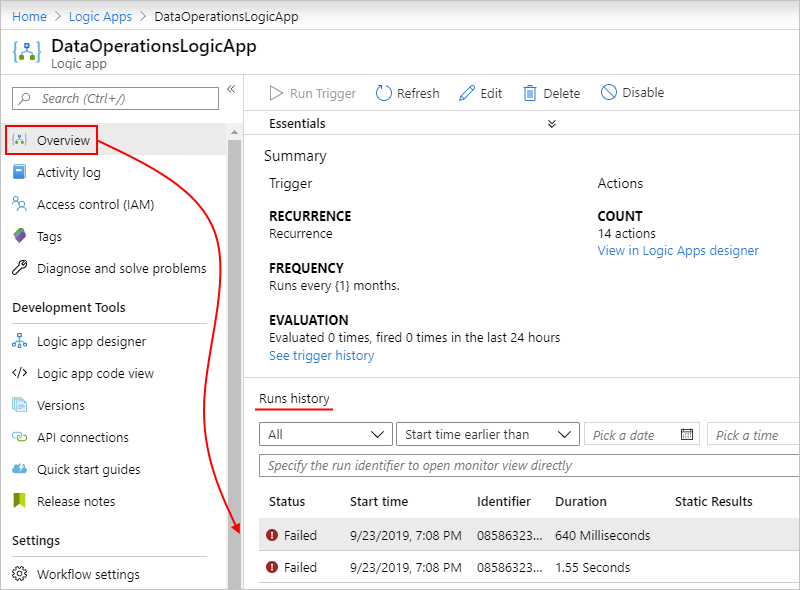


1. Select your logic app, and then select **Overview**.

On the overview pane, under **Runs history**, all the past, current, and any waiting runs for your logic app appear. If the list shows many runs, and you can't find the entry that you want, try filtering the list.

**Tip**

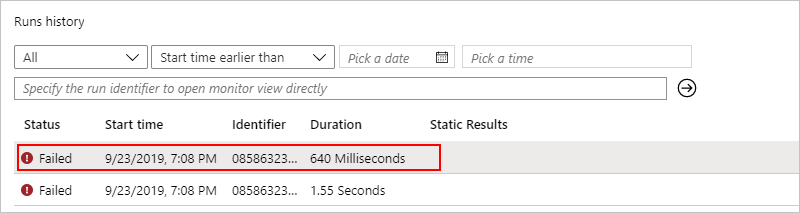
If the run status doesn't appear, try refreshing the overview page by selecting **Refresh**. No run happens for a trigger that's skipped due to unmet criteria or finding no data.



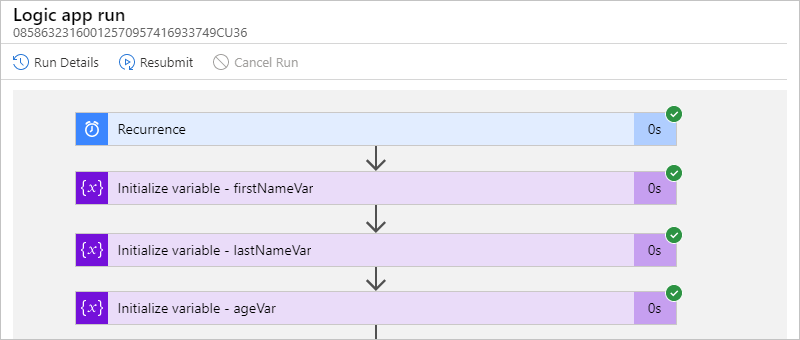
Here are the possible run statuses:

| **Run status** | **Description** |
| --- | --- |
| **Aborted** | The run stopped or didn't finish due to external problems, for example, a system outage or lapsed Azure subscription. |
| **Cancelled** | The run was triggered and started but received a cancellation request. |
| **Failed** | At least one action in the run failed. No subsequent actions in the workflow were set up to handle the failure. |
| **Running** | The run was triggered and is in progress, but this status can also appear for a run that is throttled due to [action limits](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-limits-and-config) or the [current pricing plan](https://azure.microsoft.com/pricing/details/logic-apps/).  **Tip**: If you set up [diagnostics logging](https://docs.microsoft.com/en-us/azure/logic-apps/monitor-logic-apps-log-analytics), you can get information about any throttle events that happen. |
| **Succeeded** | The run succeeded. If any action failed, a subsequent action in the workflow handled that failure. |
| **Timed out** | The run timed out because the current duration exceeded the run duration limit, which is controlled by the [**Run history retention in days** setting](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-limits-and-config#run-duration-retention-limits). A run's duration is calculated by using the run's start time and run duration limit at that start time.  **Note**: If the run's duration also exceeds the current run history retention limit, which is also controlled by the [**Run history retention in days** setting](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-limits-and-config#run-duration-retention-limits), the run is cleared from the runs history by a daily cleanup job. Whether the run times out or completes, the retention period is always calculated by using the run's start time and current retention limit. So, if you reduce the duration limit for an in-flight run, the run times out. However, the run either stays or is cleared from the runs history based on whether the run's duration exceeded the retention limit. |
| **Waiting** | The run hasn't started or is paused, for example, due to an earlier workflow instance that's still running. |
|  |  |

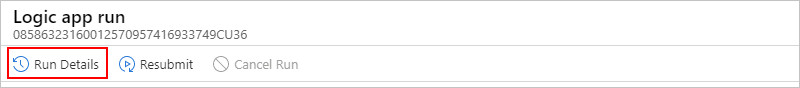
1. To review the steps and other information for a specific run, under **Runs history**, select that run.



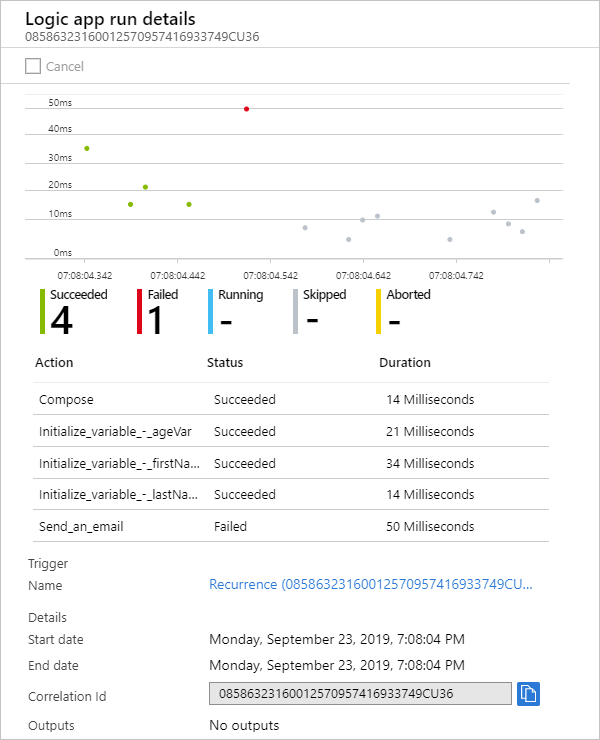
The **Logic app run** pane shows each step in the selected run, each step's run status, and the time taken for each step to run, for example:



To view this information in list form, on the **Logic app run** toolbar, select **Run Details**.

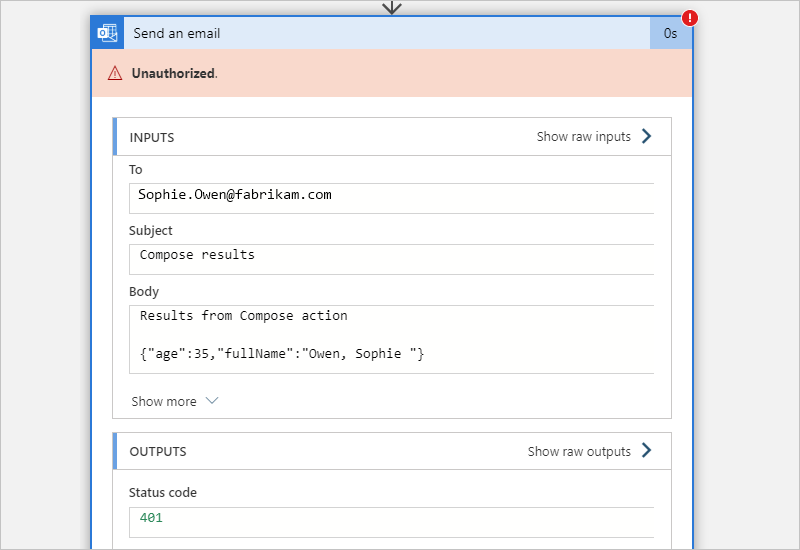


The Run Details view shows each step, their status, and other information.

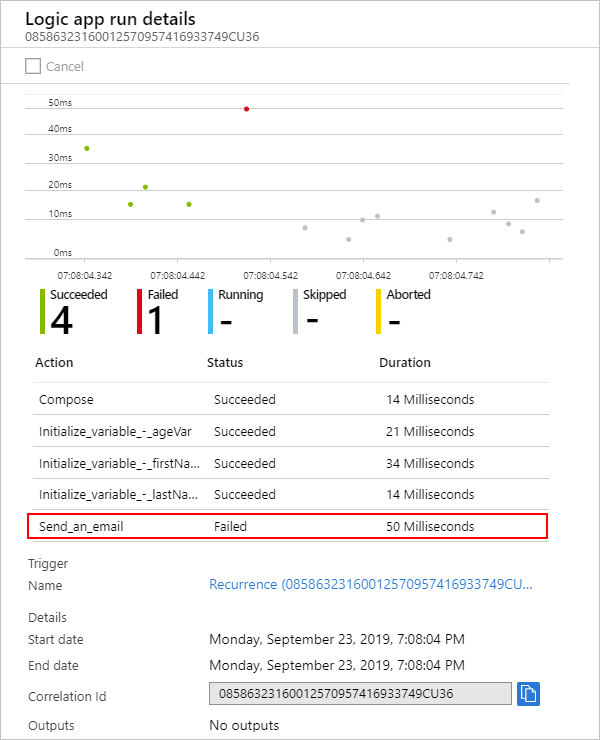


For example, you can get the run's **Correlation ID** property, which you might need when you use the [REST API for Logic Apps](https://docs.microsoft.com/en-us/rest/api/logic).

1. To get more information about a specific step, select either option:
   * In the **Logic app run** pane select the step so that the shape expands. You can now view information such as inputs, outputs, and any errors that happened in that step, for example:



* + In the **Logic app run details** pane, select the step that you want.



You can now view information such as inputs and outputs for that step, for example:

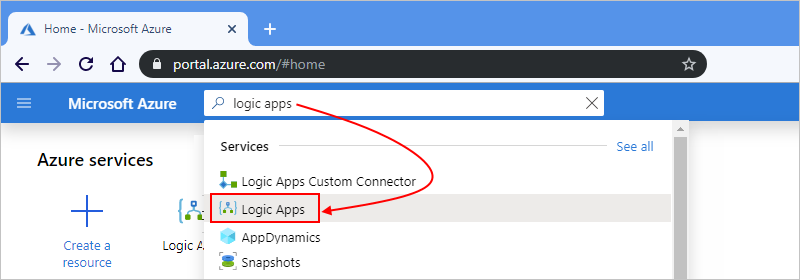
1. **Note**
2. All runtime details and events are encrypted within the Logic Apps service. They are decrypted only when a user requests to view that data. You can [**hide inputs and outputs in run history**](https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-securing-a-logic-app#obfuscate) or control user access to this information by using [**Azure role-based access control (Azure RBAC)**](https://docs.microsoft.com/en-us/azure/role-based-access-control/overview).

## Review trigger history

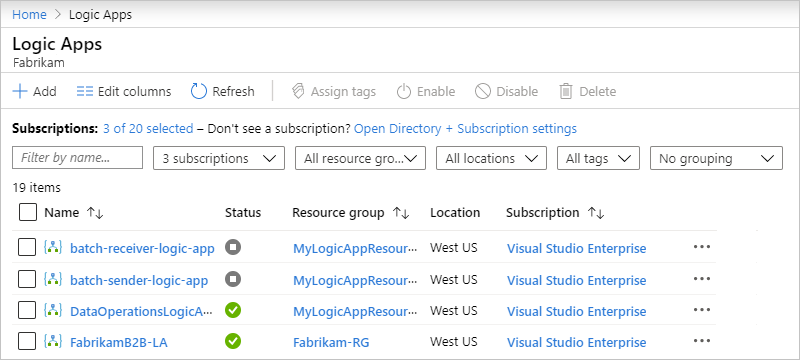
Each logic app run starts with a trigger. The trigger history lists all the trigger attempts that your logic app made and information about the inputs and outputs for each trigger attempt.

1. In the [Azure portal](https://portal.azure.com/), find and open your logic app workflow in the designer.

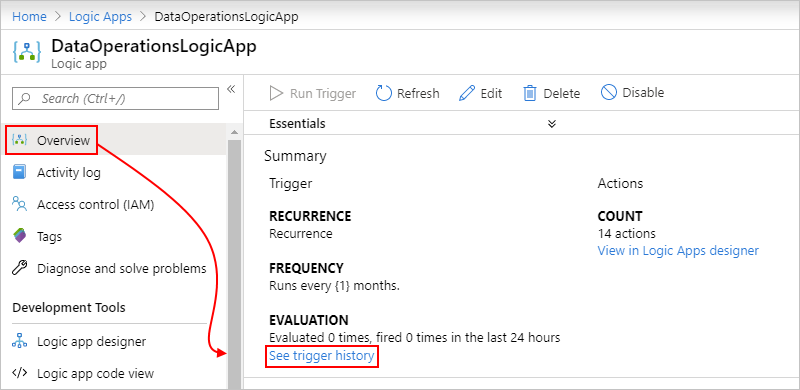
To find your logic app, in the main Azure search box, enter logic apps, and then select **Logic Apps**.



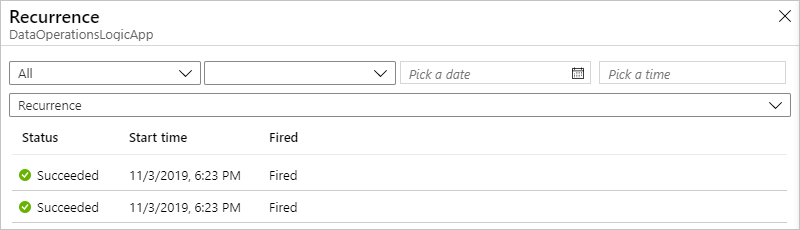
The Azure portal shows all the logic apps that are associated with your Azure subscriptions. You can filter this list based on name, subscription, resource group, location, and so on.



1. Select your logic app, and then select **Overview**.
2. On your logic app's menu, select **Overview**. In the **Summary** section, under **Evaluation**, select **See trigger history**.



The trigger history pane shows all the trigger attempts that your logic app has made. Each time that the trigger fires for an item or event, the Logic Apps engine creates a separate logic app instance that runs the workflow. By default, each instance runs in parallel so that no workflow has to wait before starting a run. So if your logic app triggers on multiple items at the same time, a trigger entry with the same date and time appears for each item.



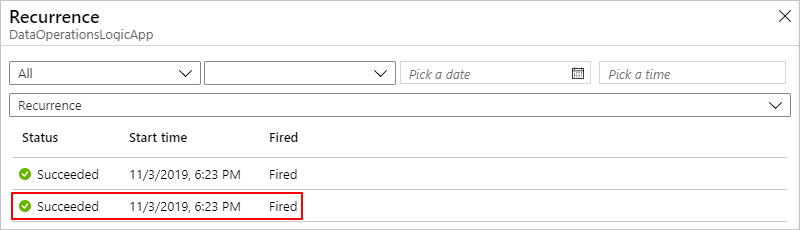
Here are the possible trigger attempt statuses:

| **Trigger status** | **Description** |
| --- | --- |
| **Failed** | An error occurred. To review any generated error messages for a failed trigger, select that trigger attempt and choose **Outputs**. For example, you might find inputs that aren't valid. |
| **Skipped** | The trigger checked the endpoint but found no data that met the specified criteria. |
| **Succeeded** | The trigger checked the endpoint and found available data. Usually, a **Fired** status also appears alongside this status. If not, the trigger definition might have a condition or SplitOn command that wasn't met.  This status can apply to a manual trigger, recurrence trigger, or polling trigger. A trigger can run successfully, but the run itself might still fail when the actions generate unhandled errors. |
|  |  |

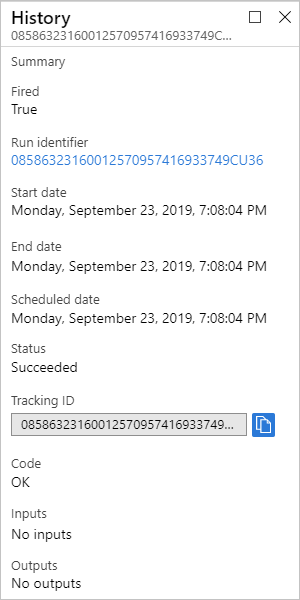
**Tip**

You can recheck the trigger without waiting for the next recurrence. On the overview toolbar, select **Run Trigger**, and select the trigger, which forces a check. Or, select **Run Trigger** on designer toolbar.

1. To view information about a specific trigger attempt, on the trigger pane, select that trigger event. If the list shows many trigger attempts, and you can't find the entry that you want, try filtering the list. If you don't find the data that you expect, try selecting **Refresh** on the toolbar.



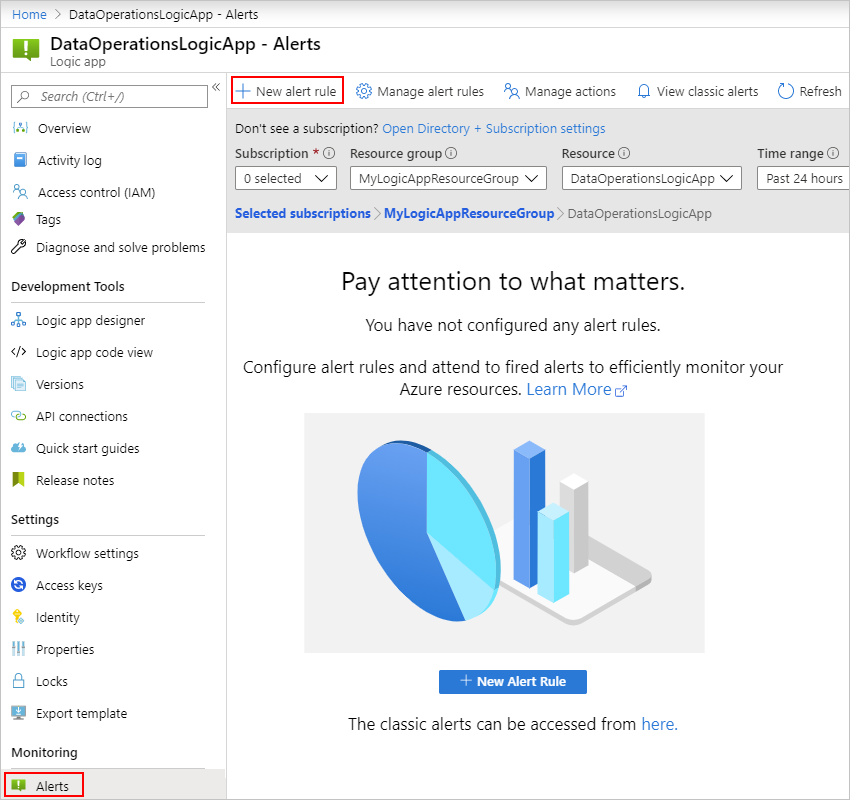
You can now review information about the selected trigger event, for example:



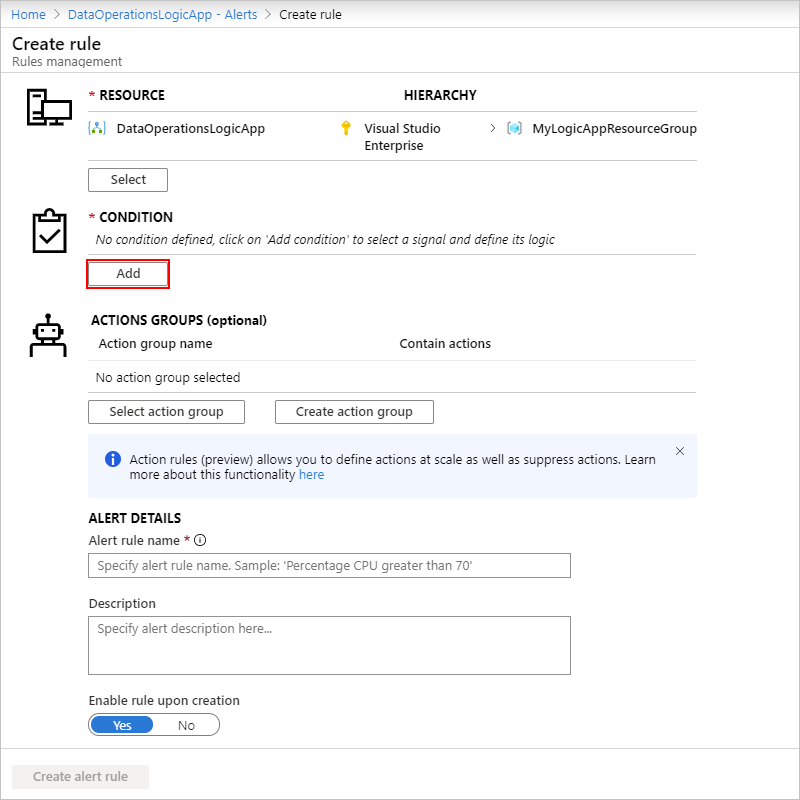
## Set up monitoring alerts

To get alerts based on specific metrics or exceeded thresholds for your logic app, set up [alerts in Azure Monitor](https://docs.microsoft.com/en-us/azure/azure-monitor/alerts/alerts-overview). Learn about [metrics in Azure](https://docs.microsoft.com/en-us/azure/azure-monitor/data-platform). To set up alerts without using [Azure Monitor](https://docs.microsoft.com/en-us/azure/azure-monitor/logs/log-query-overview), follow these steps.

1. On your logic app menu, under **Monitoring**, select **Alerts** > **New alert rule**.



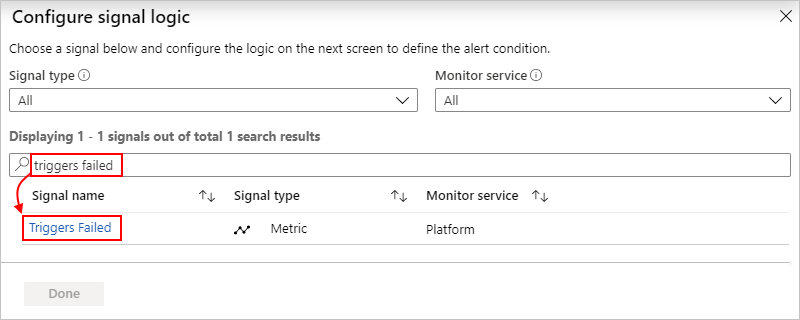
1. On the **Create rule** pane, under **Resource**, select your logic app, if not already selected. Under **Condition**, select **Add** so that you can define the condition that triggers the alert.



1. On the **Configure signal logic** pane, find and select the signal for which you want to get an alert. You can use the search box, or to sort the signals alphabetically, select the **Signal name** column header.

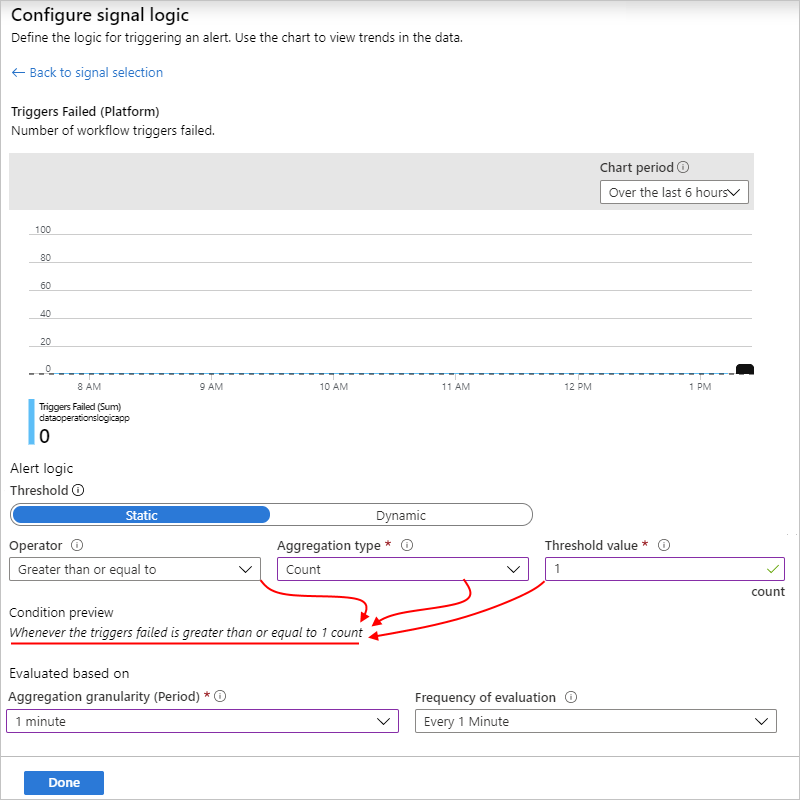
For example, if you want to send an alert when a trigger fails, follow these steps:

* 1. In the **Signal name** column, find and select the **Triggers Failed** signal.

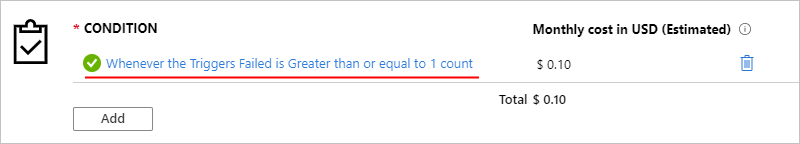


* 1. On the information pane that opens for the selected signal, under **Alert logic**, set up your condition, for example:
  2. For **Operator**, select **Greater than or equal to**.
  3. For **Aggregation type**, select **Count**.
  4. For **Threshold value**, enter 1.
  5. Under **Condition preview**, confirm that your condition appears correct.
  6. Under **Evaluated based on**, set up the interval and frequency for running the alert rule. For **Aggregation granularity (Period)**, select the period for grouping the data. For **Frequency of evaluation**, select how often you want to check the condition.
  7. When you're ready, select **Done**.

Here's the finished condition:



The **Create rule** page now shows the condition that you created and the cost for running that alert.



1. Specify a name, optional description, and severity level for your alert. Either leave the **Enable rule upon creation** setting turned on, or turn off until you're ready to enable the rule.
2. When you're done, select **Create alert rule**.