Lab requirements

- This lab requires **Microsoft Edge** or an Azure DevOps supported browser.
- **Set up an Azure DevOps organization:** If you don't already have an Azure DevOps organization that you can use for this lab, create one by following the instructions available at Create an organization or project collection.
- Identify an existing Azure subscription or create a new one.
- Verify that you have a Microsoft account or an Azure AD account with the Owner role in the Azure subscription and the Global Administrator role in the Azure AD tenant associated with the Azure subscription. For details, refer to List Azure role assignments using the Azure portal and View and assign administrator roles in Azure Active Directory.

Lab overview

This lab covers the configuration of the deployment gates and details how to use them to control the execution of Azure Pipelines. To illustrate their implementation, you'll configure a release definition with two environments for an Azure Web App. You'll deploy to the Canary environment only when there are no blocking bugs for the app and mark the Canary environment complete only when there are no active alerts in Application Insights of Azure Monitor.

A release pipeline specifies the end-to-end release process for an application to be deployed across various environments. Deployments to each environment are fully automated by using jobs and tasks. Ideally, you don't want new updates to the applications to be simultaneously exposed to all the users. It's a best practice to expose updates in a phased manner, that is, expose them to a subset of users, monitor their usage, and expose them to other users based on the experience of the initial set of users.

Approvals and gates enable you to take control over the start and completion of the deployments in a release. You can wait for users to approve or reject deployments with approvals manually. Using release gates, you can specify application health criteria to be met before the release is promoted to the following environment. Before or after any environment deployment, all the specified gates are automatically evaluated until they pass or reach your defined timeout period and fail.

Gates can be added to an environment in the release definition from the pre-deployment conditions or the post-deployment conditions panel. Multiple gates can be added to the environment conditions to ensure all the inputs are successful for the release.

As an example:

- Pre-deployment gates ensure no active issues in the work item or problem management system before deploying a build to an environment.

- Post-deployment gates ensure no incidents from the app's monitoring or incident management system after being deployed before promoting the release to the following environment.

There are 4 types of gates included by default in every account.

- Invoke Azure Function: Trigger the execution of an Azure Function and ensures a successful completion.
- Query Azure Monitor alerts: Observe the configured Azure Monitor alert rules for active alerts.
- Invoke REST API: Make a call to a REST API and continues if it returns a successful response.
- Query work items: Ensure the number of matching work items returned from a query is within a threshold.

Objectives

After you complete this lab, you will be able to:

- Configure release pipelines.
- Configure release gates.
- Test release gates.

Estimated timing: 75 minutes

Instructions

Exercise 0: Configure the lab prerequisites

In this exercise, you will set up the prerequisites for the lab, which include the preconfigured Parts Unlimited team project based on an Azure DevOps Demo Generator template and two Azure web apps representing the **Canary** and **Production** environments, into which you'll deploy the application via Azure Pipelines.

Task 1: Configure the team project

In this task, you will use Azure DevOps Demo Generator to generate a new project based on the **ReleaseGates** template.

1. On your lab computer, start a web browser and navigate to Azure DevOps Demo Generator. This utility site will automate the process of creating a new Azure DevOps project within your account that is prepopulated with content (work items, repos, etc.) required for the lab.

2. **Note**: For more information on the site, see https://docs.microsoft.com/en-us/azure/devops/demo-gen.

3. Click **Sign in** and sign in using the Microsoft account associated with your Azure DevOps subscription.

4. If required, on the **Azure DevOps Demo Generator** page, click **Accept** to accept the permission requests for accessing your Azure DevOps subscription.
5. On the **Create New Project** page, in the **New Project Name** textbox, type **Controlling Deployments using Release Gates**, in the **Select organization** dropdown list, select your Azure DevOps organization, and then click **Choose template**.
6. In the list of templates, in the toolbar, click **DevOps Labs**, select the **ReleaseGates** template and click **Select Template**.
7. Back on the **Create New Project** page, click **Create Project**
8. **Note**: Wait for the process to complete. This should take about 2 minutes. In case the process fails, navigate to your DevOps organization, delete the project, and try again.
9. On the **Create New Project** page, click **Navigate to project**.

Task 2: Create two Azure web apps

In this task, you will create two Azure web apps representing the **Canary** and **Production** environments, into which you'll deploy the application via Azure Pipelines.

10. From the lab computer, start a web browser, navigate to the **Azure Portal**, and sign in with the user account that has the Owner role in the Azure subscription you will be using in this lab and has the role of the Global Administrator in the Azure AD tenant associated with this subscription.
11. In the Azure portal, click the **Cloud Shell** icon, located directly to the right of the search textbox at the top of the page.
12. If prompted to select either **Bash** or **PowerShell**, select **Bash**.
13. **Note**: If this is the first time you are starting **Cloud Shell** and you are presented with the **You have no storage mounted** message, select the subscription you are using in this lab, and select **Create storage**.
14. From the **Bash** prompt, in the **Cloud Shell** pane, run the following command to create a resource group (replace the `<region>` placeholder with the name of the Azure region that will host the two Azure web apps, for example 'westeurope' or 'eastus'):
15. **Note**: possible locations can be found by running the following command, use the **Name** on `<region>` : `az account list-locations -o table`
16. CodeCopy

```
RESOURCEGROUPNAME='az400m10l01-RG'
az group create -n $RESOURCEGROUPNAME -l '<region>'
```

18. To create an App service plan
19. CodeCopy

```
20.      SERVICEPLANNAME='az400m01l01-sp1'
    az appservice plan create -g $RESOURCEGROUPNAME -n
   $SERVICEPLANNAME --sku S1
```

21. Create two web apps with unique app names.

22. CodeCopy

```
23.      SUFFIX=$RANDOM$RANDOM
    az webapp create -g $RESOURCEGROUPNAME -p $SERVICEPLANNAME -n
   PU$SUFFIX-Canary
    az webapp create -g $RESOURCEGROUPNAME -p $SERVICEPLANNAME -n
   PU$SUFFIX-Prod
```

24. **Note**: Record the name of the Canary web app. You will need it later in this lab.

25. Wait for the provisioning process to complete and close the **Cloud Shell** pane.

Task 3: Configure an Application Insights resource

26. In the Azure portal, use the **Search resources, services, and docs** text box at the top of the page to search for **Application Insights** and, in the list of results, select **Application Insights**.
27. On the **Application Insights** blade, select **+ Create**.
28. On the **Application Insights** blade, on the **Basics** tab, specify the following settings (leave others with their default values):

Setting	Value
Resource group	**az400m10l01-RG**
Name	the name of the Canary web app you recorded in the previous task
Region	the same Azure region to which you deployed the web apps earlier in the previous task
Resource Mode	**Classic**

29. **Note**: Disregard the deprecation message. This is required in order to prevent failures of the Enable Continuous Integration DevOps task you will be using later in this lab.

30. Click **Review + create** and then click **Create**.
31. Wait for the provisioning process to complete.
32. In the Azure portal, navigate to the resource group **az400m10l01-RG** you created in the previous task.
33. In the list of resources, click the **Canary** web app.
34. On the **Canary** web app page, in the vertical menu on the left, in the **Settings** section, click **Application Insights**.
35. On the **Application Insights** blade, click **Turn on Application Insights**.

36. In the **Change your resource** section, click the **Select existing resource** option, in the list of existing resources, select the newly created Application Insight resource, click **Apply** and, when prompted for confirmation, click **Yes**.
37. Wait until the change takes effect and, on the Application Insights blade, click the **View Application Insights data** link.

38. **Note**: You will create monitor alerts here, which you will use in later part of this lab.

39. On the Application Insights resource blade, in the **Monitoring** section, click **Alerts** and then click **Create > Alert rule**.
40. On the **Create alert rule** blade, in the **Condition** section, click the **Add condition** link.
41. On the **Select a signal** blade, in the **Search by signal name** textbox, type **Failed Requests** and select it.
42. On the **Configure signal logic** blade, in the **Alert logic** section, leave the **Threshold** set to **Static**, in the **Threshold value** textbox, type **0**, and click on **Done**.

43. **Note**: The rule will generate an alert whenever the number of failed requests is greater than 0 within the last 5 minutes.

44. Back on the **Create alert rule** pane, go to **Details** specify the following settings and fill in information (leave others with their default values)and clikc **Review + Create>Create**:

Setting	Value
Alert rule name	**PartsUnlimited_FailedRequests**
Severity	**2- Warning**
Automatically resolve alerts	cleared

45. **Note**: Metric alert rules might take up to 10 minutes to activate.

46. **Note**: You can create multiple alert rules on different metrics such as availability < 99 percent, server response time > 5 Seconds, or server exceptions > 0

Exercise 1: Configure the release pipeline

In this exercise, you will configure a release pipeline.

Task 1: Update release tasks

In this task, you will update release tasks.

47. On the lab computer, switch to the browser window displaying the **Controlling Deployments using Release Gates** project in the Azure DevOps portal, in the vertical navigational pane, select **Pipelines** and then, within the **Pipelines** section, click **Releases**.
48. Within the **Releases** view, on the **PartsUnlimited-CD** pane, click **Edit**.

49. **Note**: The pipeline contains two stages named **Canary Environment** and **Production**.

50. On the **Pipeline** tab, in the **Artifacts** rectangle, click the **Continuous deployment trigger** button in the top right corner of the **PartsUnlimited-CI** build artifact.

51. If the continuous deployment trigger for the **PartsUnlimited-CI** build is disabled, toggle the switch to enable it. Leave all other settings at default and close the **Continuous deployment trigger** pane, by clicking the **x** mark in its upper right corner.
52. Within the **Canary Environments** stage, click the **1 job, 2 tasks** label and review the tasks within this stage.

53. **Note**: The canary environment has 2 tasks which, respectively, publish the package to Azure Web App and enable continuous monitoring of the application after deployment.

54. On the **All pipelines > PartsUnlimited-CD** pane, ensure that the **Canary Environment** stage is selected. In the **Azure subscription** dropdown list, select your Azure subscription and click **Authorize**. If prompted, authenticate by using the user account with the Owner role in the Azure subscription.
55. In the **App Service name** dropdown list, select the name of the **Canary** web app.

56. **Note**: You might need to click the **Refresh** button.

57. In the **Resource Group name for Application Insights** dropdown list, select the **az400m10l01-RG** entry.
58. In the **Application Insights resource name** dropdown list, select the name of the **Canary** Application Insights resource, which should match the name of the **Canary** web app.
59. In the **Agent Phase** for **Canary Environment**, right click on **Enable Continuous Monitoring** and **Disable selected task(s)**
60. On the **All pipelines > PartsUnlimited-CD** pane, click the **Tasks** tab and, in the dropdown list, select **Production**.
61. With the **Production** stage selected, in the **Azure subscription** dropdown list, select the Azure subscription you used for the **Canary Environment** stage, shown under **Available Azure Service connections**, as we already created the service connection before when authorizing the subscription use.
62. In the **App Service name** dropdown list, select the name of the **Prod** web app.
63. On the **All pipelines > PartsUnlimited-CD** pane, click **Save** and, in the **Save** dialog box, click **OK**.
64. In the browser window displaying the **Controlling Deployments using Release Gates** project, in the vertical navigational pane, in the **Pipelines** section, click **Pipelines**.
65. On the **Pipelines** pane, click the entry representing **PartsUnlimited-CI** build pipeline and then, on the **PartsUnlimited-CI** pane, click on **Run Pipeline**.
66. On the **Run pipeline** pane, accept the default settings and click **Run** to trigger the pipeline. **Wait for the build pipeline to finish**.

67. **Note**: After the build succeeds, the release will be triggered automatically and the application will be deployed to both the environments.

68. In the vertical navigational pane, in the **Pipelines** section, click **Releases** and, on the **PartsUnlimited-CD** pane, click the entry representing the most recent release.

69. On the **PartsUnlimited-CD > Release-1** blade, track the progress of the release and verify that the deployment to both web apps completed successfully.
70. Switch to the Azure portal interface, navigate to the resource group **az400m10l01-RG**, in the list of resources, click the **Canary** web app, on the web app blade, click **Browse**, and verify that the web page loads successfully in a new web browser tab.
71. Close the web browser tab displaying the **Parts Unlimited** web site.
72. Switch to the Azure portal interface, navigate back to the resource group **az400m10l01-RG**, in the list of resources, click the **Production** web app, on the web app blade, click **Browse**, and verify that the web page loads successfully in a new web browser tab.
73. Close the web browser tab displaying the **Parts Unlimited** web site.

74. **Note**: Now you have the application with CI/CD configured. In the next exercise we will set up Gates in the release pipeline.

Exercise 2: Configure release gates

In this exercise, you will set up Gates in the release pipeline.

Task 1: Configure pre-deployment gates

In this task, you will configure pre-deployment gates.

75. Switch to the web browser window displaying the Azure DevOps portal, in the vertical navigational pane, in the **Pipelines** section, click **Releases** and, on the **PartsUnlimited-CD** pane, click **Edit**.
76. On the **All pipelines > PartsUnlimited-CD** pane, on the left edge of the rectangle representing the **Canary Environment** stage, click the oval shape representing the **Pre-deployment conditions**.
77. On **Pre-deployment conditions** pane, set the **Pre-deployment approvals** slider to **Enabled** and, in the **Approvers** text box, type and select your Azure DevOps account name.
78. On **Pre-deployment conditions** pane, set the **Gates** slider to **Enabled**, click **+ Add**, and, in the pop-up menu, click **Query Work Items**.
79. On **Pre-deployment conditions** pane, in the **Query Work Items** section, in the **Query** dropdown list, select **Bugs** under **Shared Queries**, leave the value of **Upper threshold** set to **0**.

80. **Note**: Based on the value of the **Upper threshold** setting, if this query returns any active bug work Item, the release gate will fail.

81. On the **Pre-deployment conditions** pane, leave the value of the **Delay before evaluation** setting at **0 Minutes**.

82. **Note**: **Delay before evaluation** represents the time before the added gates are evaluated for the first time. If no gates are added, then the deployments wait for the duration before proceeding. To allow gate functions to initialize and stabilize (it may

take some time for it to begin returning accurate results), we configure a delay before the results are evaluated and used to determine if the deployment should be approved or rejected.

83. On the **Pre-deployment conditions** pane, expand the **Evaluation options** section and configure the following options:
    - Set the value of **Time between re-evaluation of gates** to **5 Minutes**.

- **Note**: **Time between re-evaluation of gates** represents the time interval between each evaluation of all the gates. At each sampling interval, new requests are sent concurrently to each gate for fresh results. The sampling interval must be greater than the longest typical response time of any configured gate to allow time for all responses to be received.

    - Set the value of **Timeout after which gates fail** to **8 Minutes**.

- **Note**: **Timeout after which gates fail** represents the maximum evaluation period for all gates. The deployment will be rejected if the timeout is reached before all gates succeed during the same sampling interval. The minimum value we can specify for timeout is 6 minutes and 5 minutes for the sampling interval.

- **Note**: In this case, when a release is triggered, the gate will validate the samples at *0th* and *5th* minutes. If the result is **Pass**, notification will be sent for approval. If the result is **Fail**, the release will time-out after *8th* minute.

- **Note**: In reality these values can span multiple hours.

84. On the **Pre-deployment conditions** pane, select **On successful gates, ask for approvals** radio button.
85. Close the **Pre-deployment conditions** pane, by clicking the **x** mark in its upper right corner. **Save** the changes in the release pipeline.
86. For the **Query Work Items** gate to work, the **Project Build Service** requires Read permission for the Azure Board queries.
87. In the Azure DevOps portal, in the vertical navigational pane, hover the mouse pointer over **Boards** hold down the **Ctrl** key and click **Queries** to open a separate browser tab with the **Queries** pane.
88. On the **Queries** pane of the **Boards** view, click **All** to get a list of all queries.
89. Right-click the folder **Shared Queries** and select **Security…** to open the pane **Permissions for Shared Queries**.
90. On the **Permissions for Shared Queries** pane, into the field **Search for users or groups**, type or paste **Controlling Deployments using Release Gates Build Service** ([Project Name] Build Service) and click the one found identity.

91. **Note**: The user **Controlling Deployments using Release Gates Build Service** has to be searched for like described above, since it does not appear as a member of the **Users** list. Don't confuse the user **Project Collection Build Service** with **Project Build Service**.
92. Select the user **Controlling Deployments using Release Gates Build Service** in the **Users** list and on the right hand site set the **Read** permission to **Allow**.
93. Close the **Permissions for Shared Queries** pane, by clicking the **x** mark in its upper right corner.
94. Navigate back to the browser tab where the release pipeline is still open.

Task 2: Configure post-deployment gates
In this task, you will enable the post-deployment gate for the Canary Environment.

95. Back on the **All pipelines > PartsUnlimited-CD** pane, on the right edge of the rectangle representing the **Canary Environment** stage, click the oval shape representing the **Post-deployment conditions**.
96. On **Post-deployment conditions** pane, set the **Gates** slider to **Enabled**, click **+ Add**, and, in the pop-up menu, click **Query Azure Monitor Alerts**.
97. On **Post-deployment conditions** pane, in the **Query Azure Monitor Alerts** section, in the **Azure subscription** dropdown list, select the **service connection** entry representing the connection to your Azure subscription, and, in the **Resource group** dropdown list, select the **az400m10l01-RG** entry.
98. On **Post-deployment conditions** pane, expand the **Evaluation options** and configure the following options:

- Set the value of **Time between re-evaluation of gates** to **5 Minutes**.
- Set the value of **Timeout after which gates fail** to **8 Minutes**.
- Select the **On successful gates, ask for approvals** option.

- **Note**: The sampling interval and timeout work together so that the gates will call their functions at suitable intervals and reject the deployment if they don't succeed during the same sampling interval within the timeout period.

99. Close the **Post-deployment conditions** pane, by clicking the **x** mark in its upper right corner.
100. Back on the **PartsUnlimited-CD** pane, click **Save**, and in the **Save** dialog box, click **OK**.

Exercise 3: Test release gates
In this exercise, you will test the release gates by updating the application, which will trigger a deployment.

Task 1: Update and deploy application after adding release gates
In this task, you will track the release process with the release gates enabled.

101. In the browser window displaying the Azure DevOps portal, in the vertical navigational pane, select **Releases**.
102. Click on **Create release** and then, on the **Create a new release** pane, click **Create**.
103. In the Azure DevOps portal, in the vertical navigational pane, in the **Pipelines** section, click **Releases**.
104. On the **Releases** tab, click the **PartsUnlimited-CD/Release-2** entry and review the progress of the deployment to the **Canary Environment**.
105. Click the oval shape representing the **Pre-deployment conditions** on the left edge of the rectangle representing the **Canary Environment** stage, which, at this point, might be labeled either **Evaluating gates** or **Pre-deployment gates failed**.
106. On the **Canary Environment** pane, note that the **Query Work Items** gate failed.

107. **Note**: This indicates that there are active work items. These work items need to be closed in order to proceed further. Next sampling time will be after 5 minutes.

108. Open a new browser tab, navigate to the Azure DevOps portal, in the vertical navigational pane, select **Boards** and, in the **Boards** section, select **Queries**.
109. On the **Queries** pane of the **Boards** view, click the **All** tab.
110. On the **All** tab of the **Queries** pane, in the **Shared Queries** section, click the **Bugs** entry, on the **Queries > Shared Queries > Bugs** pane, and click **Run query**.
111. Verify that the query returns a work item titled **Disk out of space in Canary Environment** in the **New** state.

112. **Note**: Let's assume that the infrastructure team has fixed the disk space issue.

113. Click the **Disk out of space in Canary Environment** entry.
114. On the **Disk out of space in Canary Environment** pane, in the upper left corner, next to the **State** label, click **New**, in the dropdown list, click **Closed** and then click **Save**.
115. Switch back to the **Canary Environment** pane and wait for the second evaluation to pass.

116. **Note**: In case the second evaluation already failed, hover with the mouse pointer over the rectangle representing the **Canary Environment** stage to reveal the **Redeploy** option, click **Redeploy**, and, on the **Canary Environment** blade, click **Deploy** and monitor the status of processing the pre-deployment gates.

117. **Note**: Once the evaluation is successful, you will see the request for pre-deployment approval.

118. Click **Approvers** and then click **Approve** to queue a deployment into the Canary environment

119. **Note**: Once the deployment to Canary environment is successful, we will see the post-deployment gates in action which uses Application Insights to detect presence of failed requests targeting the newly deployed application.

120. To trigger a failed request, switch to the web browser window displaying the Azure portal, navigate to the **Canary** Azure web app blade, and click **Browse**. This will open a new browser tab displaying the PartsUnlimited web site.
121. On the PartsUnlimited web site, click **More**.

122. **Note**: This part of web site is intentionally misconfigured, so it will trigger a failed request.

123. Return to the home page of the PartsUnlimited web site, click **More** again, and repeat this step a few more times.
124. Validate that failed requests were detected by Application Insights by navigating to the Application Insights blade of the **Canary** web app page, and, on the Application Insights blade, click **Alerts**, and verify that the page lists one or more **Sev 2** alerts.

125. **Note**: Since there is an alert triggered by the exception, **Query Azure Monitor** gate will fail. This, in turn, will prevent deployment to the **Production** environment.

Exercise 4: Remove the Azure lab resources

In this exercise, you will remove the Azure resources provisioned in this lab to eliminate unexpected charges.

Note: Remember to remove any newly created Azure resources that you no longer use. Removing unused resources ensures you will not see unexpected charges.

Task 1: Remove the Azure lab resources

In this task, you will use Azure Cloud Shell to remove the Azure resources provisione in this lab to eliminate unnecessary charges.

126. In the Azure portal, open the **Bash** shell session within the **Cloud Shell** pane.
127. List all resource groups created throughout the labs of this module by running the following command:

128. ShellCopy

```
129. az group list --query "[?starts_with(name,'az400m10l01-RG')].name" --output tsv
```

130. Delete all resource groups you created throughout the labs of this module by running the following command:

131. ShellCopy

```
132. az group list --query "[?starts_with(name,'az400m10l01-RG')].[name]" --output tsv | xargs -L1 bash -c 'az group delete --name $0 --no-wait --yes'
```

133. **Note**: The command executes asynchronously (as determined by the –nowait parameter), so while you will be able to run another Azure CLI command immediately afterwards within the same Bash session, it will take a few minutes before the resource groups are actually removed.

Review

In this lab, you configured release pipelines and then configured and tested release gates.