Online Lab - Authoring Serverless Applications in Azure

Topic: Deploying Serverless Workloads to Azure

Before we start

1. Ensure that you are logged in to your Windows 10 lab virtual machine using the following credentials:
   - Username: **Admin**
   - Password: **Pa55w.rd**
2. Review Taskbar located at the bottom of your Windows 10 desktop. The Taskbar contains the icons for the common applications you will use in the labs:
   - Microsoft Edge
   - File Explorer
   - Visual Studio Code
   - Microsoft Azure Storage Explorer
   - Bash on Ubuntu on Windows
   - Windows PowerShell

   Note: You can also find shortcuts to these applications in the **Start Menu**.

Exercise 1: Create Web App

Task 1: Open the Azure Portal

1. On the Taskbar, click the **Microsoft Edge** icon.
2. In the open browser window, navigate to the **Azure Portal** (https://portal.azure.com).
3. If prompted, authenticate with the user account account that has the owner role in the Azure subscription you will be using in this lab.

Task 2: Open Cloud Shell

1. At the top of the portal, click the **Cloud Shell** icon to open a new shell instance.

   Note: The **Cloud Shell** icon is a symbol that is constructed of the combination of the *greater than* and *underscore* characters.

2. If this is your first time opening the **Cloud Shell** using your subscription, you will see a wizard to configure **Cloud Shell** for first-time usage. When prompted, in the **Welcome to Azure Cloud Shell** pane, click **Bash (Linux)**.

   Note: If you do not see the configuration options for **Cloud Shell**, this is most likely because you are using an existing subscription with this course's labs. If so, proceed directly to the next task.

3. In the **You have no storage mounted** pane, click **Show advanced settings**, perform the following tasks:
   - Leave the **Subscription** drop-down list entry set to its default value.
   - In the **Cloud Shell region** drop-down list, select the Azure region matching or near the location where you intend to deploy resources in this lab
   - In the **Resource group** section, ensure that the **Create new** option is selected and then, in the text box below, type **AADesignLab0901-RG**.
   - In the **Storage account** section, ensure that the **Create new** option is selected and then, in the text box below, type a unique name consisting of a combination of between 3 and 24 characters and digits.
   - In the **File share** section, ensure that the **Create new** option is selected and then, in the text box below, type **cloudshell**.
   - Click the **Create storage** button.
4. Wait for the **Cloud Shell** to finish its first-time setup procedures before you proceed to the next task.

Task 3: Create an App Service plan

1. At the **Cloud Shell** command prompt at the bottom of the portal, type in the following command and press **Enter** to create a variable which value designates the name of the resource group you will use in this exercise:

   ```
   RESOURCE_GROUP_APP='AADesignLab0502-RG'
   ```

2. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the Azure region you will use for the deployment (replace the placeholder `<Azure region>` with the name of the Azure region to which you intend to deploy resources in this lab):

   ```
   LOCATION='<Azure region>'
   ```

3. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create the resource group:

```
az group create --name $RESOURCE_GROUP_APP --location $LOCATION
```

4. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a new App Service plan:

   ```
   az appservice plan create --is-linux --name "AADesignLab0502-$LOCATION"
   --resource-group $RESOURCE_GROUP_APP --location $LOCATION --sku B2
   ```

Task 4: Create a Web App instance

1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to view a list of possible runtimes for a Linux-based App Service web app instance:

   ```
   az webapp list-runtimes --linux
   ```

2. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a new variable which value is a randomly generated string that you will use as the name of a new web app:

   ```
   WEBAPPNAME1=webapp05021$RANDOM$RANDOM
   ```

3. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a new web app using a unique name:

   ```
   az webapp create --name $WEBAPPNAME1 --plan AADesignLab0502-$LOCATION -
   -resource-group $RESOURCE_GROUP_APP --runtime "DOTNETCORE|2.1"
   ```

   Note: In case the command fails due to duplicate web app name, re-run the last two steps until the command completes successfully

4. Wait for the deployment to complete before you proceed to the next task.

Task 5: View deployment results

1. In the hub menu in the Azure portal, click **Resource groups**.
2. On the **Resource groups** blade, click **AADesignLab0502-RG**.
3. On the **AADesignLab0502-RG** blade, click the entry representing the Azure web app you created earlier in this exercise.
4. On the web app blade, click the **Browse** button at the top of the blade.
5. Review the default page generated by Azure App Service.
6. Close the new browser tab and return to the browser tab displaying the Azure portal.

Review: In this exercise, you created a Linux-based App Service Plan that contained a blank web app.

Exercise 2: Deploy Web App code

Task 1: Deploy code with a Web App Extension using an Azure Resource Manager template and GitHub

1. At the top of the portal, click the **Cloud Shell** icon to open a new shell instance.
2. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the name of the resource group you will use in this exercise:

   ```
   RESOURCE_GROUP_APP='AADesignLab0502-RG'
   ```

3. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the Azure region you will use for the deployment:

   ```
   LOCATION=$(az group list --query "[?name == 'AADesignLab0502-
   RG'].location" --output tsv)
   ```

4. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a new variable which value is a randomly generated string that you will use as the name of a new web app:

   ```
   WEBAPPNAME2=webapp05022$RANDOM$RANDOM
   ```

5. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a new web app using a unique name:

   ```
   az webapp create --name $WEBAPPNAME2 --plan AADesignLab0502-$LOCATION -
   -resource-group $RESOURCE_GROUP_APP --runtime "NODE|9.4"
   ```

   Note: In case the command fails due to duplicate web app name, re-run the last two steps until the command completes successfully

6. In the **Cloud Shell** pane, click the **Upload/Download files** icon and, in the drop-down menu, click **Upload**.
7. In the **Open** dialog box, navigate to the **F:\Labfiles\Mod05\Starter\** folder, select the **github.json** file, and click **Open**. The file contains the following Azure Resource Manager template:

```
{ "$schema": "http://schemas.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#", "contentVersion": "1.0.0.0",
"parameters": { "webAppName": { "type": "string" }, "repositoryUrl": {
"type": "string" }, "branch": { "type": "string", "defaultValue":
"master" } }, "resources": [ { "apiVersion": "2015-08-01", "type":
"Microsoft.Web/sites", "name": "[parameters('webAppName')]",
"location": "[resourceGroup().location]", "properties": {},
"resources": [ { "apiVersion": "2015-08-01", "name": "web", "type":
"sourcecontrols", "dependsOn": [ "[resourceId('Microsoft.Web/Sites',
parameters('webAppName'))]" ], "properties": { "RepoUrl":
"[parameters('repositoryUrl')]", "branch": "[parameters('branch')]",
"IsManualIntegration": true } } ] } ] }
```

8. In the **Cloud Shell** pane, click the **Upload/Download files** icon and, in the drop-down menu, click **Upload**.
9. In the **Open** dialog box, navigate to the **F:\Labfiles\Mod05\Starter\** folder, select the **parameters.json** file, and click **Open**. The file contains the following parameters for the Azure Resource Manager template you uploaded previously:

```
{ "$schema": "http://schema.management.azure.com/schemas/2015-01-
01/deploymentParameters.json#", "contentVersion": "1.0.0.0",
"parameters": { "webAppName": { "value": "$WEBAPPNAME2" },
"repositoryUrl": { "value": "$REPOSITORY_URL" }, "branch": { "value":
"master" } } }
```

10. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the name of the GitHub repository hosting the web app code:

```
REPOSITORY_URL='https://github.com/Azure-Samples/nodejs-docs-hello-
world'
```

11. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the name of the GitHub repository hosting the web app code and which takes into account any special character the URL might include:

```
REPOSITORY_URL_REGEX="$(echo $REPOSITORY_URL | sed -e 's/\\/\\\\/g;
s/\//\\\//g; s/&/\\\&/g')"
```

Note: This is necessary because you will use the **sed** utility to insert this string into the Azure Resource Manager template parameters file. Alternatively, you could simply open the file and enter the URL string directly into the file.

12. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to replace the placeholder for the value of the **webAppName** parameter with the value of the **$WEBAPPNAME2** variable in the parameters file:

```
sed -i.bak1 's/"$WEBAPPNAME2"/"'"$WEBAPPNAME2"'"/' ~/parameters.json
```

13. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to replace the placeholder for the value of the **repositoryUrl** parameter with the value of the **$REPOSITORY_URL** variable in the parameters file:

    ```
    sed -i.bak2 's/"$REPOSITORY_URL"/"'"$REPOSITORY_URL_REGEX"'"/'
    ~/parameters.json
    ```

14. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to verify that the placeholders were successfully replaced in the parameters file:

    ```
    cat ~/parameters.json
    ```

15. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to deploy the GitHub-resident web app code by using a local Azure Resource Manager template and a local parameters file:

    ```
    az group deployment create --resource-group $RESOURCE_GROUP_APP --
    template-file github.json --parameters @parameters.json
    ```

16. Wait for the deployment to complete before you proceed to the next task.

    Note: The deployment should take about a minute.

Task 2: View deployment results

1. In the hub menu in the Azure portal, click **Resource groups**.
2. On the **Resource groups** blade, click **AADesignLab0502-RG**.
3. On the **AADesignLab0502-RG** blade, click the entry representing the Azure web app you created in the previous task.
4. On the web app blade, click the **Browse** button at the top of the blade.
5. Review the sample Node.js web application deployed from GitHub.
6. Close the new browser tab and return to the browser tab displaying the Azure portal.

Task 3: Deploy Code with a Docker Hub container image

1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the name of the resource group you will use in this task:

    ```
    RESOURCE_GROUP_CONTAINER='AADesignLab0502-RG'
    ```

2. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the Azure region you will use for the deployment:

   ```
   LOCATION=$(az group list --query "[?name == 'AADesignLab0502-
   RG'].location" --output tsv)
   ```

3. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a new variable which value is a randomly generated string that you will use as the name of a new web app:

   ```
   WEBAPPNAME3=webapp05023$RANDOM$RANDOM
   ```

4. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a new web app using a unique name:

   ```
   az webapp create --name $WEBAPPNAME3 --plan AADesignLab0502-$LOCATION -
   -resource-group $RESOURCE_GROUP_CONTAINER --deployment-container-image
   ghost
   ```

   Note: In case the command fails due to duplicate web app name, re-run the last two steps until the command completes successfully

5. Wait for the deployment to complete before you proceed to the next task.

   Note: The deployment should take less than a minute.

6. Close the **Cloud Shell** pane.

Task 4: View deployment results

1. In the hub menu in the Azure portal, click **Resource groups**.
2. On the **Resource groups** blade, click **AADesignLab0502-RG**.
3. On the **AADesignLab0502-RG** blade, click the entry representing the Azure web app you created in the previous task.
4. On the web app blade, click the **Browse** button at the top of the blade.

   Note: If the application does not appear, switch to the web app blade, click **Restart** button at the top of the blade and then click **Browse** again.

5. Review the blog application deployed from Docker Hub.
6. Close the new browser tab and return to the browser tab displaying the Azure portal.

Review: In this exercise, you deployed code using an Azure Resource Manager template and a Docker Hub image to App Service web apps.

Exercise 3: Deploy a Function App

Task 1: Deploy a Function App with code using an Azure Resource Manager template

1. In the upper left corner of the Azure portal, click **Create a resource**.
2. At the top of the **New** blade, in the **Search the Marketplace** text box, type **Template Deployment** and press **Enter**.
3. On the **Everything** blade, in the search results, click **Template deployment**.
4. On the **Template deployment** blade, click the **Create** button.
5. On the **Custom deployment** blade, click the **Build your own template in the editor** link.
6. On the **Edit template** blade, click the **Quickstart template** link.
7. In the **Load a quickstart template** pane, in the **Select a template** drop-down list, select the **201-function-app-dedicated-github-deploy** template.
8. Click the **OK** button.
9. Back on the **Edit template** blade, click the **Save** button to persist the template.
10. Back on the **Custom deployment** blade, perform the following tasks:
    - Leave the **Subscription** drop-down list entry set to its default value.
    - In the **Resource group** section, select the **Use existing** option.
    - In the **Resource group** section, ensure that **Create new** option is selected and, in the text box below, type **AADesignLab0503-RG**.
    - In the **App Name** text box, type a unique name for the new Function App.
    - In the **Sku** drop-down list, select the **Basic** option.
    - Leave the **Worker Size** drop-down list set to its default value.
    - Leave the **Storage Account Type** drop-down list set to its default value.
    - Leave the **Repo URL** field set to its default value.
    - Leave the **Branch** text box set to its default value.
    - Leave the **Location** text box set to its default value.
    - In the **Terms and Conditions** section, select the **I agree to the terms and conditions stated above** checkbox.
    - Click the **Purchase** button.
11. Wait for the deployment to complete before you proceed to the next task.

    Note: The deployment should take about a minute.

Task 2: View deployment results

1. In the hub menu in the Azure portal, click **Resource groups**.
2. On the **Resource groups** blade, click **AADesignLab0503-RG**.
3. On the **AADesignLab0503-RG** blade, click the entry representing the Function App you created in the previous task.
4. On the Function App blade, locate the **Url** entry and click the hyperlink below to see the Function App landing page in a new browser tab.
5. Close the new browser tab and return to the browser tab displaying the Azure portal.

Review: In this exercise, you deployed a Function App and code using an Azure Resource Manager template.

Exercise 4: Remove lab resources

Task 1: Open Cloud Shell

1. At the top of the portal, click the **Cloud Shell** icon to open the Cloud Shell pane.
2. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to list all resource groups you created in this lab:

   ```
   az group list --query "[?starts_with(name,'AADesignLab05')]".name --
   output tsv
   ```

3. Verify that the output contains only the resource groups you created in this lab. These groups will be deleted in the next task.

Task 2: Delete resource groups

1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to delete the resource groups you created in this lab

   ```
   az group list --query "[?starts_with(name,'AADesignLab05')]".name --
   output tsv | xargs -L1 bash -c 'az group delete --name $0 --no-wait --
   yes'
   ```

2. Close the **Cloud Shell** prompt at the bottom of the portal.

Review: In this exercise, you removed the resources used in this lab.