100 XP

# Exercise - Create and deploy an Azure Resource Manager template

10 minutes

Sandbox activated! Time remaining: **3 hr 50 min**

You have used 1 of 10 sandboxes for today. More sandboxes will be available tomorrow.

**Note**

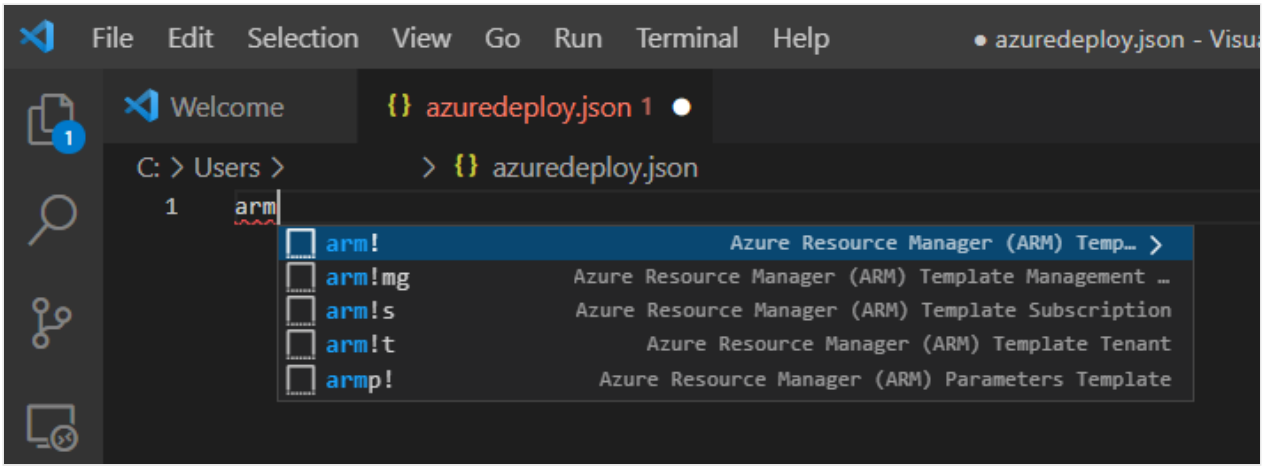
The first time you activate a sandbox and accept the terms, your Microsoft account is associated with a new Azure directory named Microsoft Learn Sandbox. You're also added to a special subscription named Concierge Subscription.

In this exercise, you create an Azure Resource Manager template (ARM template), deploy it to Azure, and then update that ARM template to add parameters and outputs.

This exercise uses Azure Resource Manager Tools for Visual Studio Code. Be sure to install this extension in Visual Studio Code before starting the exercise.

## Create an ARM template

1. Open Visual Studio Code, and create a new file called *azuredeploy.json*.
2. The Visual Studio Code ARM template extension comes configured with snippets to help you develop templates. Let's start by adding a blank template. On the first line of the file, enter *arm*.
3. The VS Code automatically displays several potential choices that start with **arm!**. Select the **Azure Resource Manager (ARM) template**. VS Code automatically processes the schemas and languages for your template.




Your file now looks like this:

JSON

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {},
  "functions": [],
  "variables": {},
  "resources": [],
  "outputs": {}
}
```

Notice that this file has all of the sections of an ARM template that we described in the previous unit.

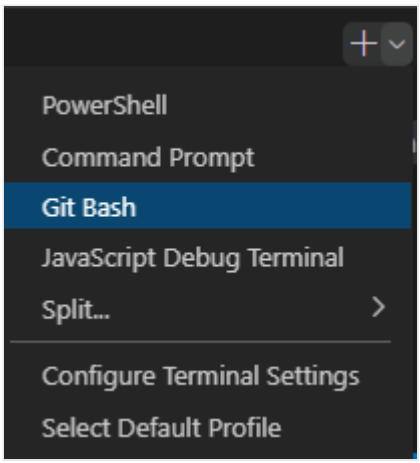
4. Save the changes to the file by pressing `Ctrl+S`.

# Deploy the ARM template to Azure

To deploy this template to Azure, you need to sign in to your Azure account from the Visual Studio Code terminal. Be sure you have the Azure CLI tools installed, and sign in with the same account you used to activate the sandbox.

1. Select **Terminal** > **New Terminal** to open a terminal window.
2. If the command bar of the terminal window says **bash**, you have the right shell to work from and you can skip to the next section.
3. If not, select the drop-down, and choose **Select Default Profile**.


4. Select **Git Bash**.




5. Select **Terminal** > **New Terminal** to open a bash shell terminal window.

## Sign in to Azure

1. In the terminal window, run this command to sign in to Azure.

```
Azure CLI
az login
```

2. A browser opens so that you can sign in to your account. After you've signed in, a list of the subscriptions associated with this account display in the terminal. If you activated the sandbox, you should see one called *Concierge Subscription*. Use this one for the rest of the exercise.

3. In the bash shell, run the following command to set the default subscription for all of the Azure CLI commands you run in this session.

```
Azure CLI
az account set --subscription "Concierge Subscription"
```

If you've used more than one sandbox recently, more than one *Concierge Subscription* might be listed. If so, use the next two steps to identify and set the default subscription.
a. Run the following command to obtain the *Concierge Subscription* IDs.

```
Azure CLI
```

```
az account list \
  --refresh \
  --query "[?contains(name, 'Concierge Subscription')].id" \
  --output table
```

a. Set the default subscription by running the following command, replacing *{your subscription ID}* with the latest Concierge Subscription ID.

Azure CLI

```
az account set --subscription {your subscription ID}
```

## Set the default resource group

By setting the default resource group to the one created when you activated the sandbox, you can omit that parameter from the Azure CLI commands in this exercise. To set the resource group, run the following command.

Azure CLI

```
az configure --defaults group=learn-6fca1147-f65f-4078-844e-d92a3152f9d8
```

## Deploy the template to Azure

Run the following commands to deploy the ARM template to Azure. The ARM template doesn't have any resources yet, so you won't see any resources created. You will see a successful deployment.

Azure CLI

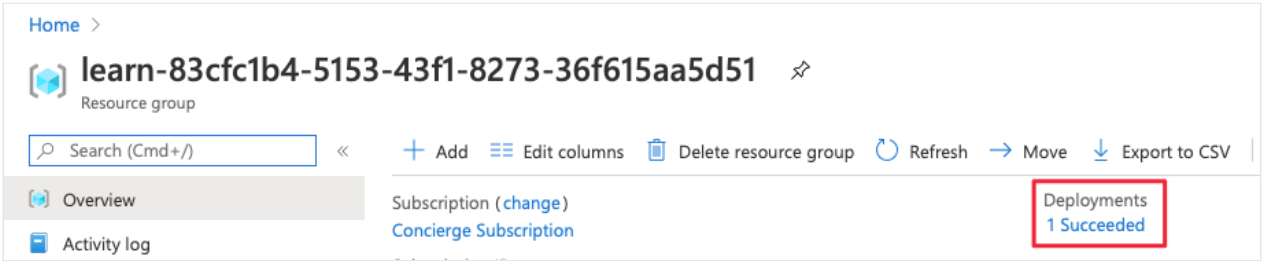
```
templateFile="azuredeploy.json"
today=$(date +"%d-%b-%Y")
DeploymentName="blanktemplate-"$today

az deployment group create \
 --name $DeploymentName \
 --template-file $templateFile
```

The top section of the preceding code sets the Azure CLI variables, which include the path to the template file to deploy and the name of the deployment. The bottom section, `az deployment group create`, deploys the template to Azure. Notice that the deployment name is `blanktemplate` with the date as a suffix.

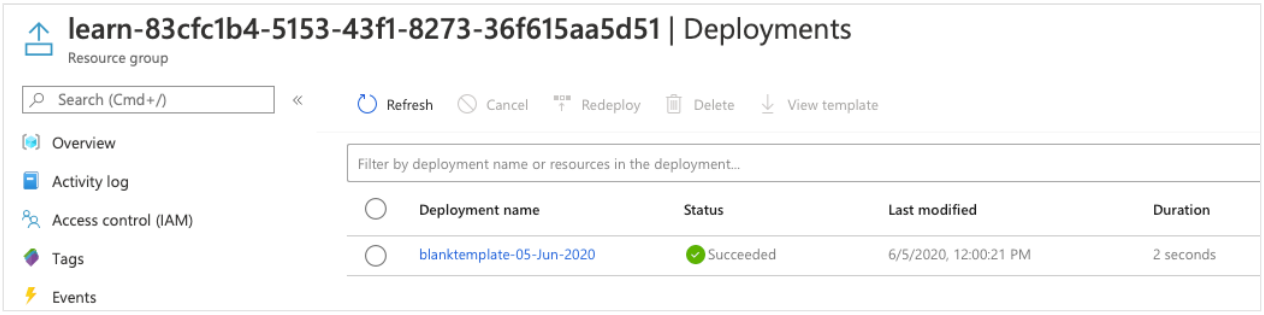
You see `Running...` in the terminal.

When you've deployed your ARM template to Azure, go to the Azure portal and make sure you're in the sandbox subscription. To do that, select your avatar in the upper-right corner of the page. Select **Switch directory**. In the list, choose the **Microsoft Learn Sandbox** directory.

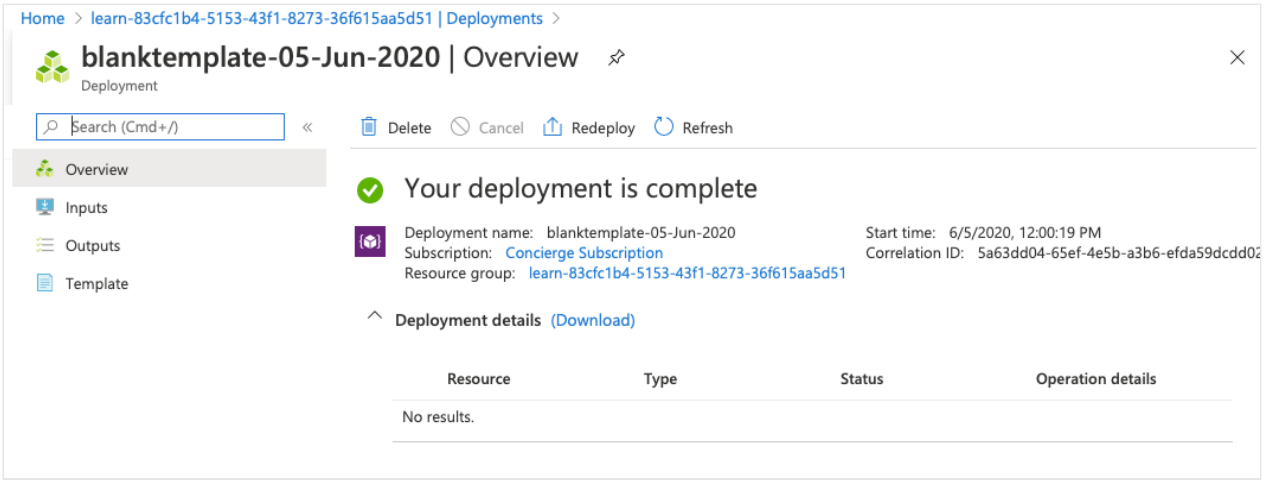
1. In the resource menu, select **Resource groups**.
2. Select the *learn-6fca1147-f65f-4078-844e-d92a3152f9d8* resource group.
3. On the **Overview** pane, you see that one deployment succeeded.




4. Select **1 Succeeded** to see the details of the deployment.




5. Select `blanktemplate` to see what resources were deployed. In this case, it will be empty because you didn't specify any resources in the template yet.




6. Leave the page open in your browser. You'll check on deployments again.

# Add a resource to the ARM template

In the previous task, you learned how to create a blank template and deploy it. Now, you're ready to deploy an actual resource. In this task, you add an Azure storage account resource to the ARM template by using a snippet from the Azure Resource Manager Tools extension for Visual Studio Code.

1. In the *azuredeploy.json* file in Visual Studio Code, place your cursor inside the brackets in the resources block `"resources":[],`.

2. Enter *storage* inside the brackets. A list of related snippets appears. Select **arm-storage**.


Your file will look like this:

JSON

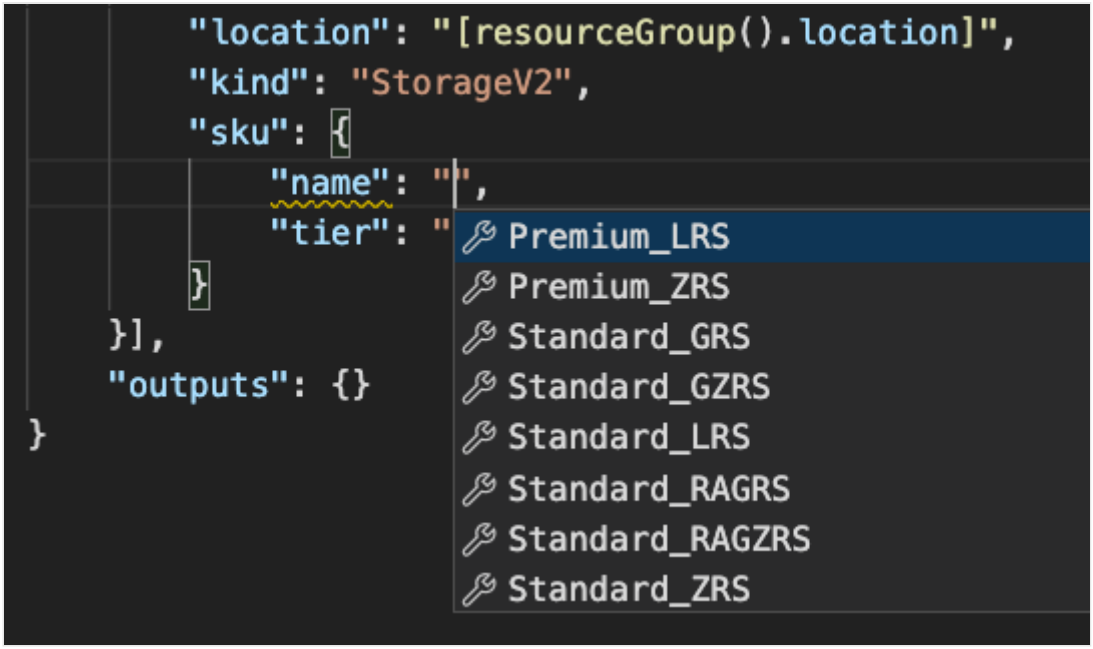
```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {},
  "functions": [],
  "variables": {},
  "resources": [
    {
      "name": "storageaccount1",
      "type": "Microsoft.Storage/storageAccounts",
      "apiVersion": "2019-06-01",
      "tags": {
        "displayName": "storageaccount1"
      },
      "location": "[resourceGroup().location]",
      "kind": "StorageV2",
      "sku": {
        "name": "Premium_LRS",
        "tier": "Premium"
      }
    }
  ],
  "outputs": {}
}
```

Values that you should edit are highlighted in the new section of your file and can be navigated by pressing the `Tab` key.

Notice the `tags` and `location` attributes are filled in. The `location` attribute uses a function to set the location of the resource to the location of the resource group. You'll learn about tags and functions in the next module.

3. Change the values of the resource *name* and *displayName* to something unique, (for example, **learnexercise12321**). This name must be unique across all of Azure, so choose something unique to you.

4. Change the value of the sku *name* from **Premium_LRS** to **Standard_LRS**. Change the value of *tier* to **Standard**. Notice that Visual Studio Code gives you the proper choices for your attribute values in IntelliSense. Delete the default value including the quotation marks, and enter quotation marks to see this work.




5. The location of the resource is set to the location of the resource group where it will be deployed. Leave the default here.

6. Save the file.

## Deploy the updated ARM template

Here, you change the name of the deployment to better reflect what this deployment does.

Run the following Azure CLI commands in the terminal. This snippet is the same code you used previously, but the name of the deployment is changed.

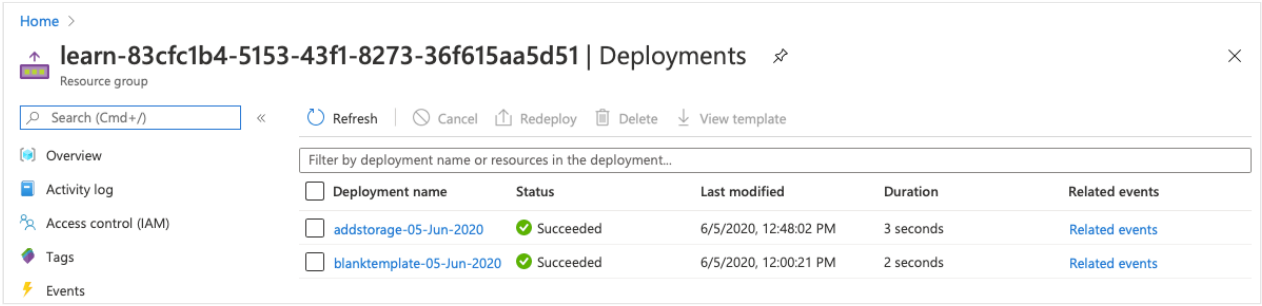
Azure CLI

```
templateFile="azuredeploy.json"
today=$(date +"%d-%b-%Y")
DeploymentName="addstorage-"$today

az deployment group create \
  --name $DeploymentName \
  --template-file $templateFile
```

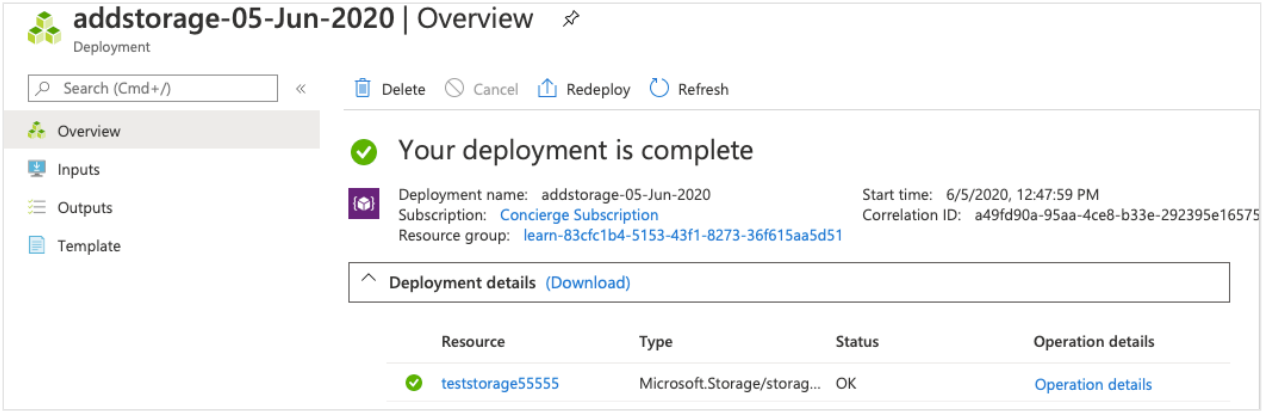
## Check your deployment

1. In your browser, go back to the Azure portal. Go to your resource group, and you'll see that there are now **2 Succeeded** deployments. Select this link.

   Notice that both deployments are in the list.




2. Select `**addstorage**`.




Notice that the storage account has been deployed.

---

## Next unit: Add flexibility to your Azure Resource Manager template by using parameters and outputs

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆