600 XP

Exercise - Configure, deploy, and run in Azure

12 minutes

Sandbox activated! Time remaining: **1 hr 40 min**

You have used 2 of 10 sandboxes for today. More sandboxes will be available tomorrow.

Choose your development language

C# | JavaScript

Now it's time to run our app in Azure. We need to create an Azure App Service app, set it up with a managed identity and our vault configuration, and deploy our code.

Create the App Service plan and app

Creating an App Service app is a two-step process: First create the *plan*, then the *app*.

The *plan* name only needs to be unique within your subscription, so you can use the same name we've used: **keyvault-exercise-plan**. The app name needs to be globally unique, though, so you'll need to pick your own.

In Azure Cloud Shell, run the following to create an App Service plan:

Azure CLI Copy

```
az appservice plan create \
    --name keyvault-exercise-plan \
    --sku FREE \
    --location centralus \
    --resource-group learn-41a3bcf2-9bc5-45a5-a5c5-916736ad2d89
```

Next, run the following command to create the Web App that uses the App Service plan you just created:

Azure CLI Copy

```
az webapp create \
    --plan keyvault-exercise-plan \
    --runtime "node|10.6" \
    --resource-group learn-41a3bcf2-9bc5-45a5-a5c5-916736ad2d89 \
    --name <your-unique-app-name>
```

Add configuration to the app

For deploying to Azure, we'll follow the App Service best practice of putting the VaultName configuration in an application setting instead of a configuration file. We'll also set the `SCM_DO_BUILD_DURING_DEPLOYMENT` setting to `true` so that App Service restores our application's packages on the server and creates the necessary configuration to run the app. Run this command to create the application settings:

Azure CLI Copy

```
az webapp config appsettings set \
    --resource-group learn-41a3bcf2-9bc5-45a5-a5c5-916736ad2d89 \
    --name <your-unique-app-name> \
    --settings 'VaultName=<your-unique-vault-name>'
'SCM_DO_BUILD_DURING_DEPLOYMENT=true'
```

Enable managed identity

Enabling managed identity on an app is a one-liner — run this to enable it on your app:

Azure CLI Copy

```
az webapp identity assign \
    --resource-group learn-41a3bcf2-9bc5-45a5-a5c5-916736ad2d89 \
    --name <your-unique-app-name>
```

From the JSON output that results, copy the **principalId** value. PrincipalId is the unique ID of the app's new identity in Azure Active Directory, and we're going to use it in the next step.

Grant access to the vault

The last step before deploying is to assign Key Vault permissions to your app's managed identity. Use the **principalId** value you copied from the previous step as the value for **object-id** in the command below. Running this command will grant **Get** and **List** access:

Azure CLI Copy

```
az keyvault set-policy \
    --secret-permissions get list \
    --name <your-unique-vault-name> \
    --object-id <your-managed-identity-principleid>
```

Deploy the app and try it out

All your configuration is set and you're ready to deploy! The below commands will zip up your app into `site.zip` and deploy it to App Service. We exclude `node_modules` from the zip because App Service will restore them automatically when we deploy.

> **Note**
>
> You'll need to `cd` back to the KeyVaultDemoApp directory if you're not still there.

Azure CLI | Copy

```
zip site.zip * -x node_modules/

az webapp deployment source config-zip \
    --src site.zip \
    --resource-group learn-41a3bcf2-9bc5-45a5-a5c5-916736ad2d89 \
    --name <your-unique-app-name>
```

The deployment may take a minute or two to complete. Once you get a result that indicates the site has deployed, open `https://<your-unique-app-name>.azurewebsites.net/api/SecretTest` in a browser. The app will take a moment to start up for the first time on the server, but once it does, you should see the secret value, **reindeer_flotilla**.

Your app is finished and deployed!

Next unit: Summary

Continue >

English (United States)

Azure Cloud Shell