Microsoft Virtual Labs
Power BI

Developing a Web App with Azure Power BI Embedded

Overview

**The estimated time to complete the lab is 60 minutes**

In this lab, you will integrate a Power BI report into an ASP.NET MVC web app by using the Microsoft Power BI Embedded Azure service. The report will use DirectQuery to connect to an Azure SQL Database.

The exercises in this lab involve provisioning an Azure resource group consisting of Power BI Embedded and Azure SQL Database, and then using a sample Visual C# console application to provision a Power BI workspace, import a pre-developed Power BI Desktop file, and configure its database connection and credentials. A pre-developed Visual C# web app is then configured to integrate a report. Finally, an optional exercise involves publishing the web app to Azure.

In order to complete this lab, you will require an Azure subscription. There is no requirement to sign in to Power BI.

For an introduction to Power BI Embedded, review the Azure Documentation article What is Microsoft Power BI Embedded.

The final solution, with report embedded, will look like the following.
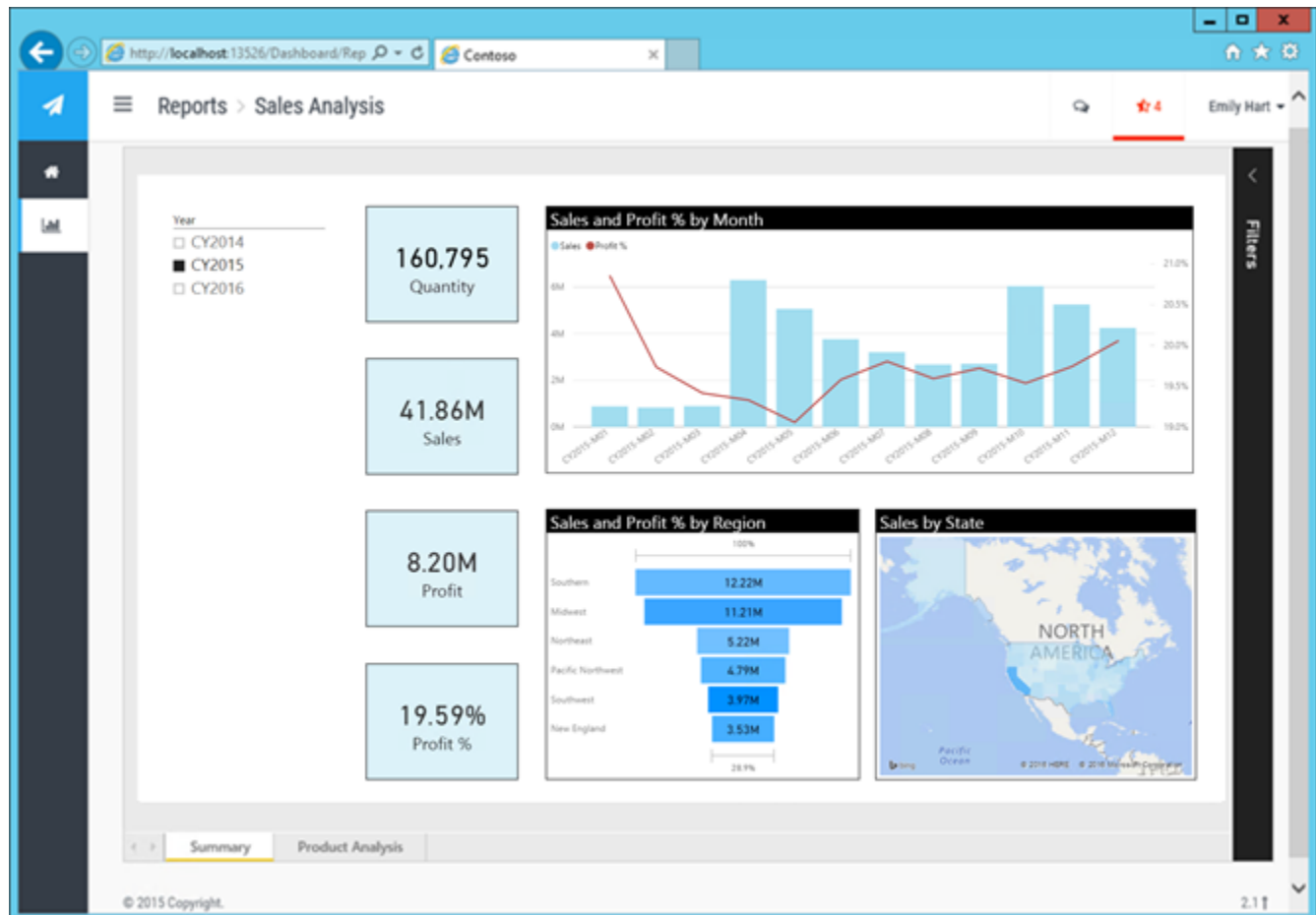
You will learn how to:

- Use the Azure Portal to create a Power BI workspace collection

- Use a sample application (based on the Power BI REST API) to create a Power BI workspace, import a Power BI Desktop file, and configure a dataset connection

- Configure an existing web app to integrate a Power BI report

- Use the Azure Portal to monitor embed renders

- Publish the app to Azure Web Sites

Connecting to the Virtual Machine

In this exercise, you will use the lab hosting portal to connect to the virtual machine, and optimize the virtual machine environment for your language and location.

Signing In

In this task, you will sign in to the virtual machine.

1. To sign in to the virtual machine, using the portal menu, click **Commands**, and then select **Ctrl + Alt + Delete**.
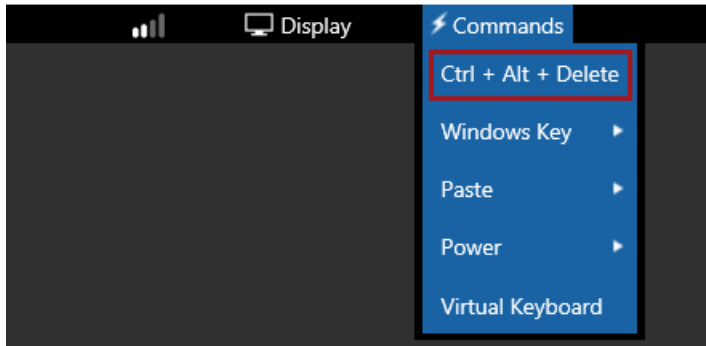
2. In the password box, enter **Pass@word1** (do not enter the period), and then click **Submit**.

   If you are not using a US English keyboard, the password you enter may not be correctly received by the virtual machine. You must complete the following task to sign in, and then update the virtual machine language.

   Note: For lab users with English keyboards, if the @ symbol is above the 2, then your keyboard is a US English keyboard, and you should not complete the following task.

Updating the Virtual Machine Language

In this task, you will sign in to the virtual machine by using the on-screen keyboard, and then update the virtual machine language. This is important to ensure that your keyboard characters are correctly received by the virtual machine.

1. Located at the bottom-left of the virtual machine screen, click **Ease of Access**, and then select **On-Screen Keyboard**.

2. Use the on-screen keyboard to enter the password **Pass@word1**. (Do not enter the period.)

   Tip: To reveal the input password before submitting, click the following.
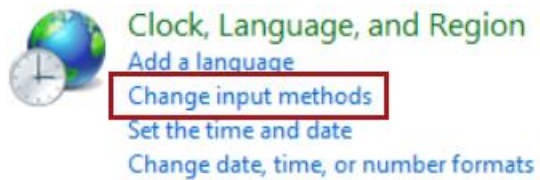

3. Submit the password.

4. Once signed in, to add a new language, on the taskbar, click the **Windows** button.

5. In the **Start** screen, click the **Control Panel** tile.
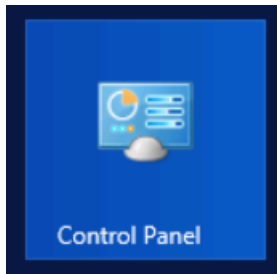
6. In the **Control Panel** window, from inside the **Clock, Language, and Region** group, click **Change Input Methods**.

7. In the **Language** window, click **Add a Language**.

8. In the **Add Languages** window, locate and select your language, and then click **Add** (or **Open**).

   *If the selected language has regional variants, you will be directed to the **Regional Variants** window, in which case, select a variant, and then click **Add**.*
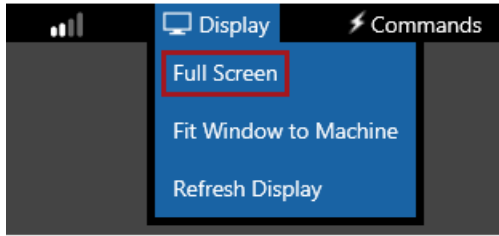
9. Close the **Language** window.

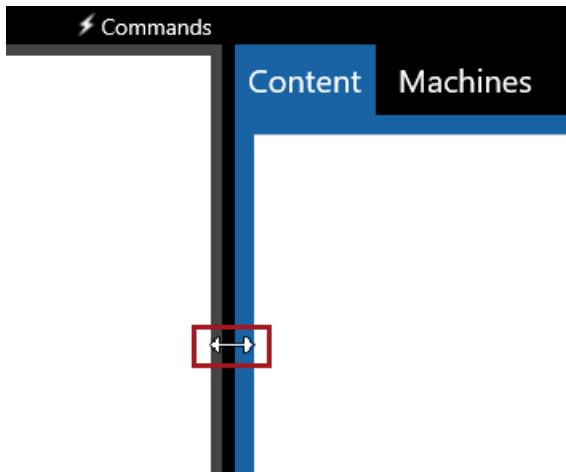10. In the taskbar, click **ENG**, and then select your language.

Changing the Screen Resolution (Optional)

It is recommended that you change the virtual machine screen resolution to take full advantage of your screen size.

1. Using the portal menu, click **Display**, and then select **Full Screen**.

2. First, notice that the portal left pane can be resized.

3. Resize the width of the pane, with the aim of arriving at a minimum width that allows the font size of the content (lab manual) to remain easily readable.

   Tip: Less width occupied by the pane allows for more room for the virtual machine screen.

4. In the virtual machine screen, right-click the desktop, and then select **Screen Resolution**.

5. In the **Screen Resolution** window, in the **Resolution** dropdown list, select a higher resolution.

   1024 x 768 is the recommended minimum, but use a higher resolution if this fits your screen size.

6. Click **Apply**.

7. If the entire virtual machine screen is fully visible within the portal, click **Keep Changes**, otherwise, click **Revert**, and try a different resolution.
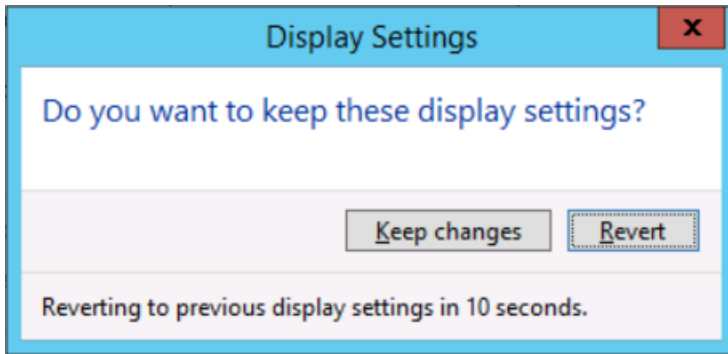
Changing the Clock (Optional)

Changing the clock is for your convenience only, and therefore this task is optional.

1. In the taskbar, click the clock, and then select **Change Date and Time Settings**.

2. In the **Date and Time** window, click **Change Time Zone**.

3. In the **Time Zone Settings** window, in the **Time Zone** dropdown list, select your time zone.

4. Click **OK**.

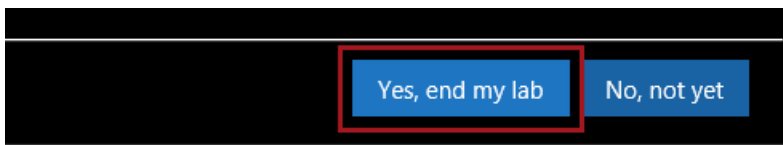5. In the **Date and Time** window, click **OK**.

Ending the Lab Session

In this task, you will explore how to end the lab session—when you are ready to do so.

1. At the top right-corner of the portal, click **Exit**, and then select **End Lab**.

2. When prompted to end the lab, click **Yes, End My Lab**.

3. Once redirected to the evaluation form, please take a few moments to complete and submit your evaluation of this lab—your feedback assists us to deliver a great lab experience.

You are now ready to commence the lab.

Getting Started

In this exercise, you will execute a T-SQL script to prepare the **TailspinToys-US** database for this lab, and then explore a completed Power BI Desktop solution.

Preparing the Database

In this task, you will execute a T-SQL script to prepare the **TailspinToys-US** database for this lab.

1. To open File Explorer, on the taskbar, click the **File Explorer** shortcut.

2. In the **File Explorer** window, navigate to the **D:\PowerBI\Lab10\Assets** folder.

3. Right-click the **Setup.cmd** file, and then select **Open**.

4. In the Command window, when prompted to press any key to continue, press any key.

Exploring the Power BI Desktop Solution

In this task, you will explore the **Sales Analysis** Power BI Desktop solution.

1. In the **File Explorer** window, right-click the **Sales Analysis.pbix** file, and then select **Open**.

   This Power BI Desktop file connects to an on-premises SQL Server database by using DirectQuery. This means that the solution does not import any data, and every report interaction results in a live query to the database. The final solution produced in this lab will involve connecting to an Azure SQL Database.

   Presently, the only connected sourced supported for Power BI Embedded are cloud sources: Azure SQL Database, Azure SQL Data Warehouse and Azure HDInsight Spark.

2. In the **Access a SQL Server Database** dialog window, click **Connect**.

3. In the **Encryption Support** dialog window, click **OK**.

4. At the bottom left, notice that the report consists of two pages.
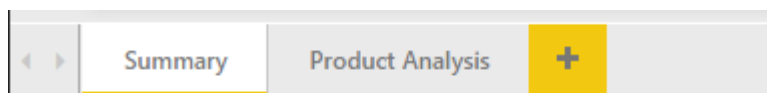
5. On the **Summary** page, review the report layout consisting of a slicer for years, four cards to display numeric values, a column chart, a funnel and a filled map.

6. Hover over the columns, funnel bars, and state areas to reveal tooltips of information.

7. Explore the report by clicking columns, funnel bars, and state areas to cross-filter the other visuals on the page.

8. Select the **CY2015** slicer value to filter the page by that year.
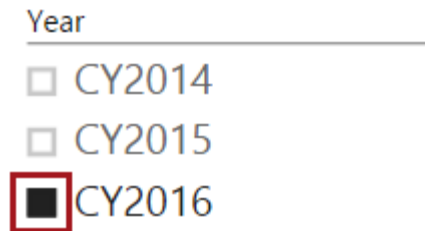
9. Select the **Product Analysis** page.

10. Select the **CY2016** slicer value to filter the page, and the custom chord visual, by that year.

11. Switch back to the **Summary** page.

12. To save the file, click the **File** ribbon tab, and then select **Save**.

13. To close the file, click the **File** ribbon tab, and then select **Exit**.

The Power BI Desktop file must be closed, as it will be uploaded later in this lab by using the Power BI REST API.

Provisioning Azure Services

In this exercise, you will provision an Azure resource group, and then add in a Power BI workspace collection, and SQL database server. You will also deploy the on-premises **TailspinToys-US** database to the Azure SQL database server, and update the Power BI Desktop file to connect to the cloud database.

Signing In to the Azure Portal

In this task, you will sign in to the Azure Portal.

1. To open Internet Explorer, on the taskbar, click the **Internet Explorer** program shortcut.

2. Navigate to https://portal.azure.com, and then sign in to the Azure Portal by using your own account.

   *Tip: You can also use the **Sign In to New Azure Portal** Internet Explorer favorite.*

Creating a Resource Group

In this task, you will create a resource group to manage and monitor related resources as a single entity.

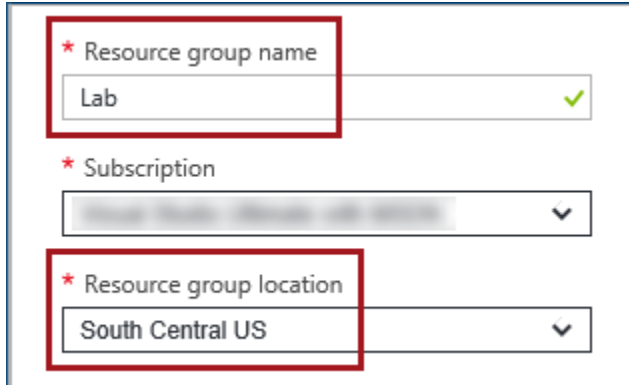1. In the menu (located at the left), select **Resource Groups**.
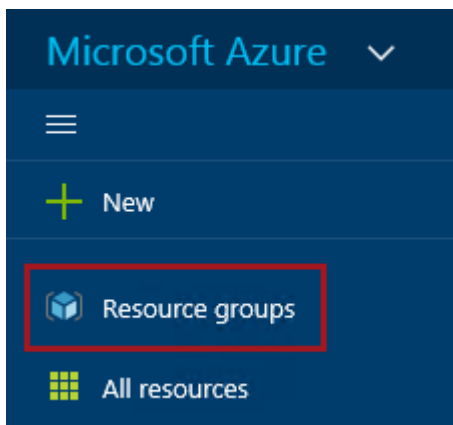
2. In the **Resource Groups** blade, click **Add**.

---
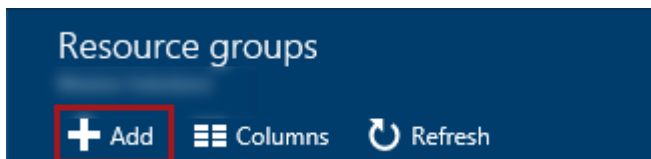
3. In the Resource Group blade, in the **Resource Group Name** box, enter **Lab**.

    Usually Azure resource names must be globally unique. The resource group name is an exception, and must only be unique within your Azure subscription.

4. In the **Resource Group Location** box, ensure that **South Central US** is selected.

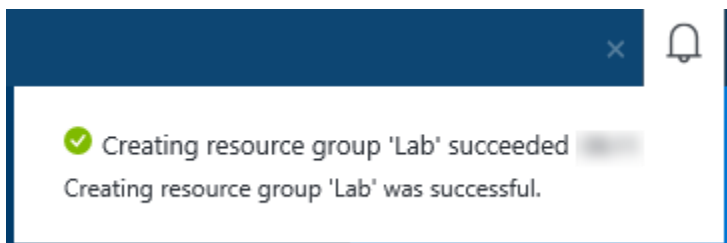5. Verify that the configuration looks like the following.
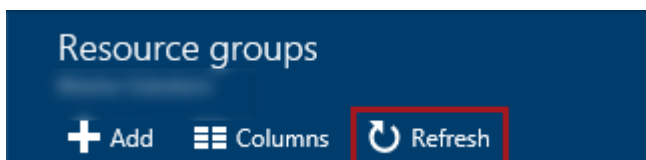

6. Click **Create**.

7. At the top-right corner of the portal, notice that there is a single notification.

8. Click the **Notifications** icon, and verify that the resource group was successfully deployed.

9. In the **Resource Groups** blade, to reload the list, click **Refresh**.

10. Verify that the **Lab** resource group is listed.

Managing Configurations

In this task, you will open the **Configurations.txt** file. This file has been designed to support you to complete this lab, specifically to store, and later copy-and-paste configuration values.

1. In the **File Explorer** window (for the **D:\PowerBI\Lab10\Assets** folder), notice the **Configurations.txt** file.

   This file will be used to store values specific to your Azure resources, and this will enable you to easily retrieve values when configuring the app later in this lab.

2. To open the file in Notepad, right-click the **Configurations.txt** file, and then select **Open**.
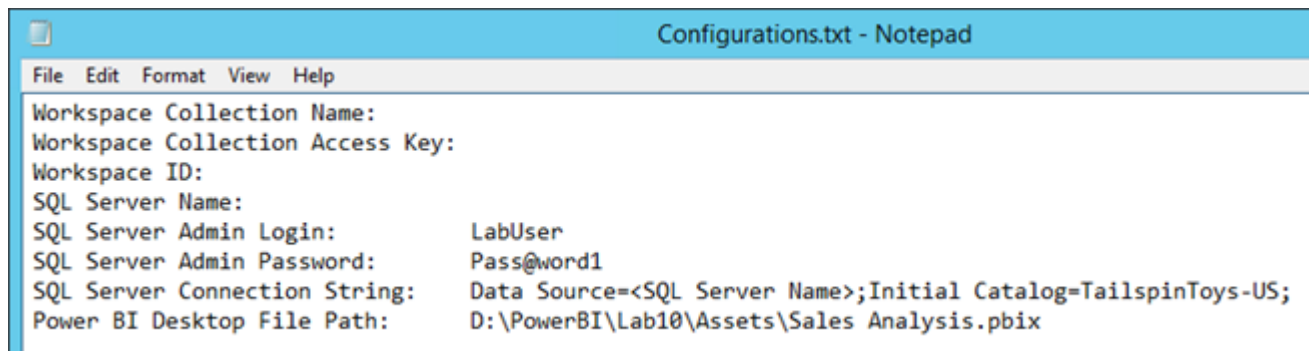
3. Leave the file open.

Creating a Power BI Workspace Collection

In this task, you will create a Power BI workspace collection.

Workspace collections and Workspaces are containers of content that are used and organized in whichever way best fits the design of the application you are building. In this lab, you will create only one workspace which will store a single dataset and report.

Presently it is not possible to create a workspace in the Azure Portal. You will create the workspace in the next exercise by using a console application based on the Power BI REST API.

1. In the **Resource Groups** blade, select the **Lab** resource group.

2. In the **Lab** resource group blade, click **Add**.

3. In the **Everything** blade, in the search box, enter **Power BI**, and then press **Enter**.

4. In the **Results** list, select **Power BI Embedded**.
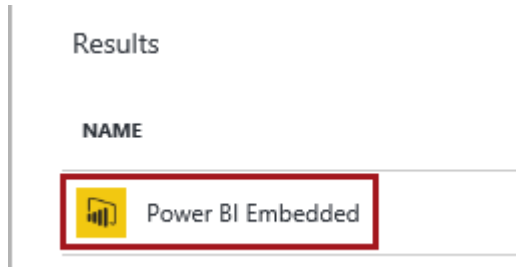
5. In the **Power BI Embedded** blade, review the service details, and then click **Create**.

6. In the **Power BI Workspace Collection** blade, in the **Workspace Collection Name** box, enter a unique name (perhaps based on your name).

   When inputting the name, the Azure Portal does verify that the name you enter meets naming standards, but it does not check that the name is globally unique. If the name is not unique, the deployment will fail.

7. In the **Resource Group** section, ensure that the existing **Lab** resources group is selected.

8. In the **Location** box, ensure that **South Central US** is selected.

9. Verify that the configuration looks like the following.

10. Click **Create**.

11. Review the portal notifications, and wait until the deployment has succeeded.

12. To configure the Power BI workspace collection, on the bread crumb trail, click the **Lab** resource group.
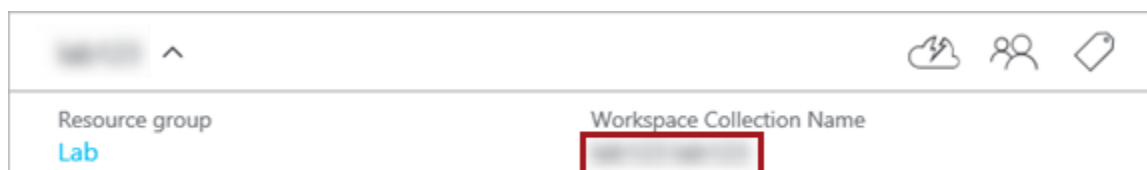

13. To refresh the **Lab** resource group blade, click **Refresh**.
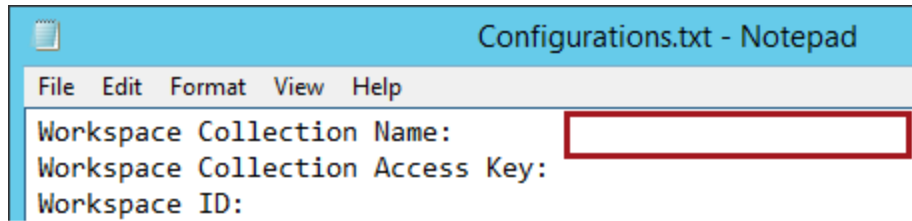

14. Select your Power BI workspace collection.

15. In the Power BI workspace collection blade, copy the workspace collection name to the clipboard.
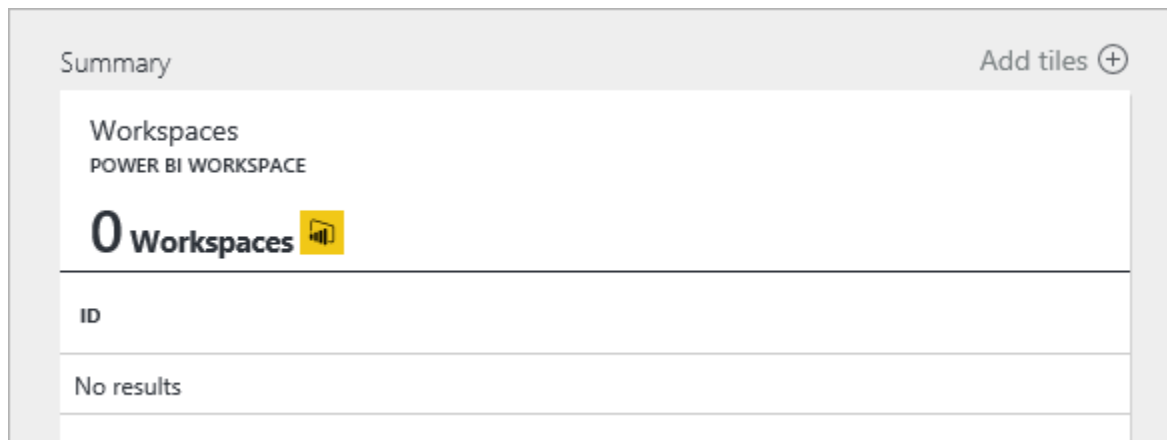
16. In the **Configurations.txt** window, paste the clipboard content to the right of the
    Workspace Collection Name label, taking care not to paste any leading or training spaces.

```
Configurations.txt - Notepad
File  Edit  Format  View  Help
Workspace Collection Name:
Workspace Collection Access Key:
Workspace ID:
```

*The **Workspace Collection Name** value will be retrieved later when creating a workspace.*

17. In the Power BI workspace collection blade, in the **Summary** tile, notice that the workspace
   collection—by default—contains zero workspaces.

Summary   Add tiles ⊕

Workspaces
POWER BI WORKSPACE

0 Workspaces

ID

No results

18. Notice also that this blade contains a tile to monitor of embed renders (you may need to scroll
    down).

Details

Getting Started

EMBED RENDERS ⓘ
0 times

*Embed renders represent the number of requests made to display data in a Power BI report, and
they are used to compute billing amounts. You will monitor the embed renders later in this lab.*
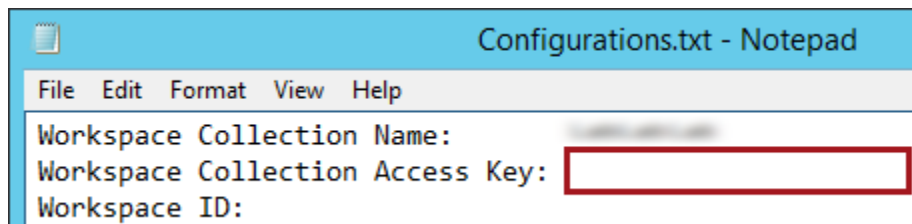
19. To retrieve an access key, in the **Settings** blade, click **Access Keys**.
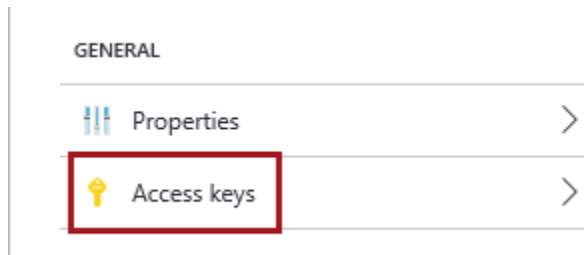
20. In the **Access Keys** blade, to copy the first key to the clipboard, for **Key 1**, click the **Copy** command.
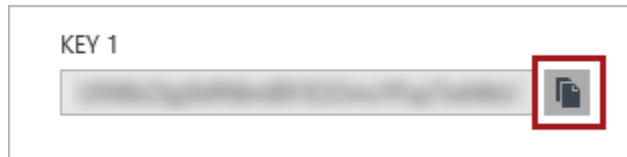
21. In the **Configurations.txt** window, paste the clipboard content to the right of the **Workspace Collection Access Key** label.

The **Workspace Collection Access Key** value will be retrieved later when creating a workspace.

Creating a SQL Database Server

In this task, you will create an Azure SQL Database server.

*This server will be used to host a single database which you will deploy the **TailspinToys-US** database in the next exercise.*
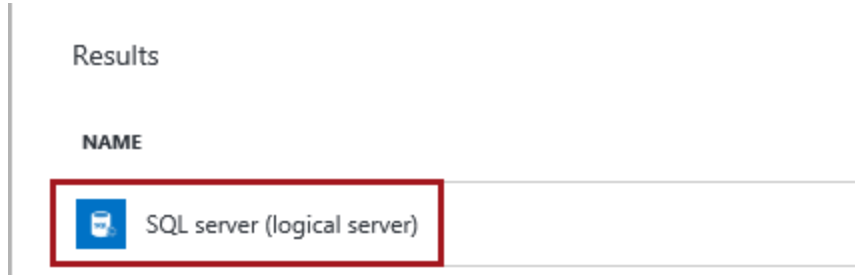
1. Return to the **Lab** resource group blade.

2. Click **Add**.

---

3. In the **Everything** blade, in the search box, enter **SQL Server**, and then press **Enter**.

4. In the **Results** list, select **SQL Server (Logical Server)**.

5. In the **SQL Server (Logical Server)** blade, review service details, and then click **Create**.

6. In the **SQL Server (Logical Server Only)** blade, in the **Server Name** box, enter a value to form a unique address.

7. In the **Server Admin Login** box, enter **LabUser**.

    *Tip: The login and password are available to copy from the **Configurations.txt** file.*

8. In the **Password** box, enter **Pass@word1**. (Do not enter the period.)

9. In the **Confirm Password** box, re-enter **Pass@word1**. (Do not enter the period.)

10. In the **Resource Group** section, ensure that the existing **Lab** resource group is selected.

11. In the **Location** dropdown list, ensure that **South Central US** is selected.

12. Verify that the configuration looks like the following.

13. Click **Create**.

14. Review the portal notifications, and wait until the deployment has succeeded.

15. To configure the SQL Server, on the bread crumb trail, click the **Lab** resource group.
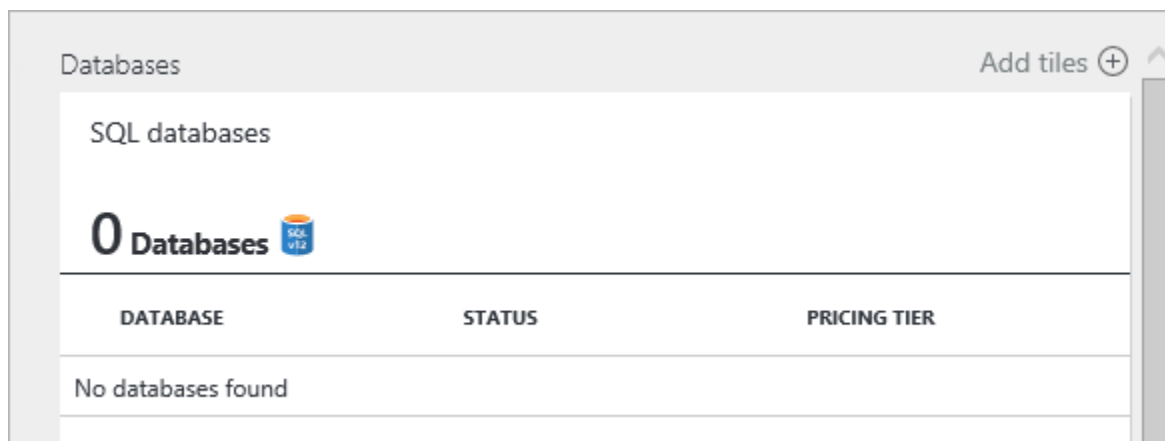

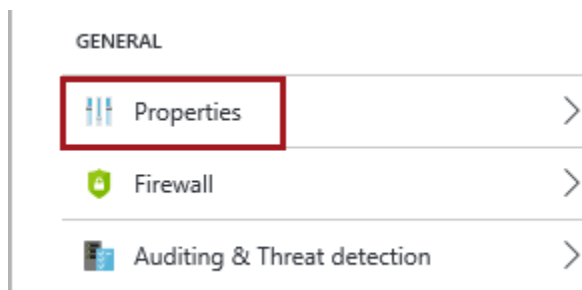16. To refresh the **Lab** resource group blade, click **Refresh**.

17. In the **Lab** resource group blade, select the SQL Server resource.
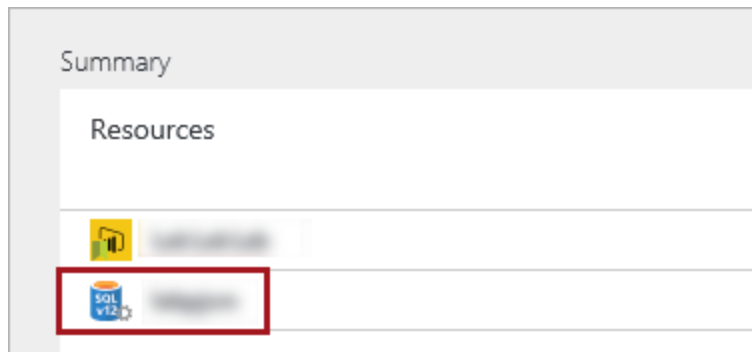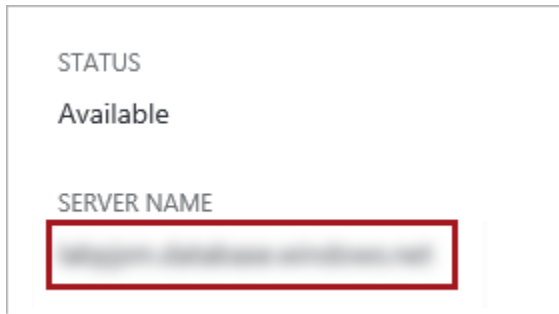

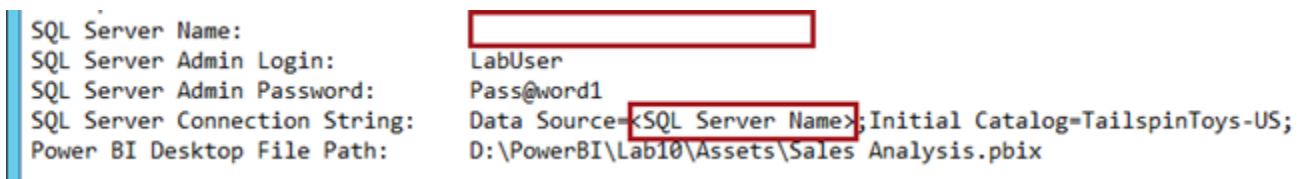18. In the SQL Server blade, notice that it contains zero databases.

19. To retrieve the server address, in the **Settings** blade, click **Properties**.

20. In the **Properties** blade, copy the full **Server Name** value to the clipboard.

STATUS

Available

SERVER NAME

21. In the **Configurations.txt** window, paste the clipboard content to the right of the **SQL Server Name** label.

22. Also in the **Configurations.txt** window, modify the **SQL Server Connection String** value, by carefully replacing the entire **<SQL Server Name>** token with the **SQL Server Name** value.

```
SQL Server Name:
SQL Server Admin Login:          LabUser
SQL Server Admin Password:       Pass@word1
SQL Server Connection String:    Data Source=<SQL Server Name>;Initial Catalog=TailspinToys-US;
Power BI Desktop File Path:      D:\PowerBI\Lab10\Assets\Sales Analysis.pbix
```

*The **SQL Server Name** will be retrieved later when deploying the local database to Azure.*

*The **SQL Server Connection String** will be retrieved later when updating the Power BI Desktop solution to connect to the Azure SQL Database.*

23. In the Azure Portal, to configure the firewall, in the **Settings** blade, click **Firewall**.

GENERAL

Properties   >

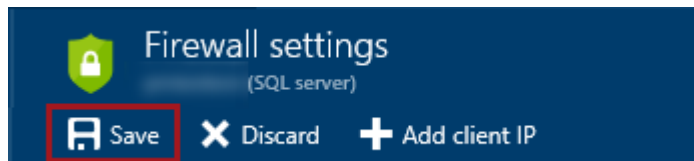Firewall   >

Auditing & Threat detection   >
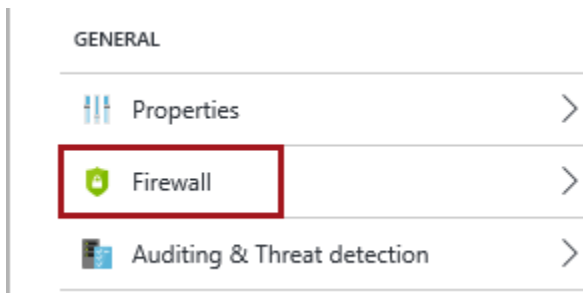
24. In the **Firewall Settings** blade, click **Add Client IP**.

*This will enable you to connect to the server from the lab virtual machine instance in order to deploy the **TailspinToys-US** database.*

Firewall settings
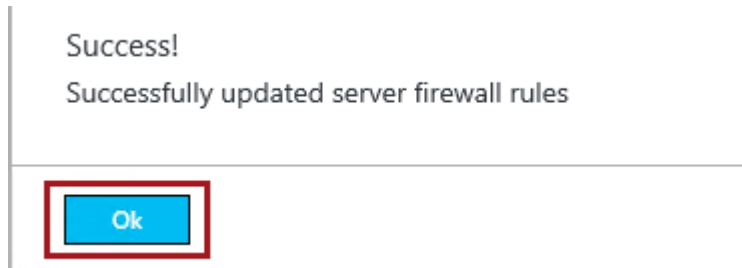(SQL server)

Save   Discard   + Add client IP
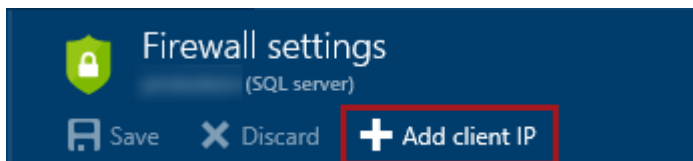
25. Notice that a rule was added, with both the start and end ranges based on your IP address.

26. Click **Save**.

27. When the firewall rules have been updated, click **OK**.

Deploying the Database to Azure SQL Server

In this task, you will use SQL Server Management Studio to deploy the **TailspinToys-US** database to the Azure SQL Database server.

1. To open SQL Server Management Studio, on the taskbar, click the
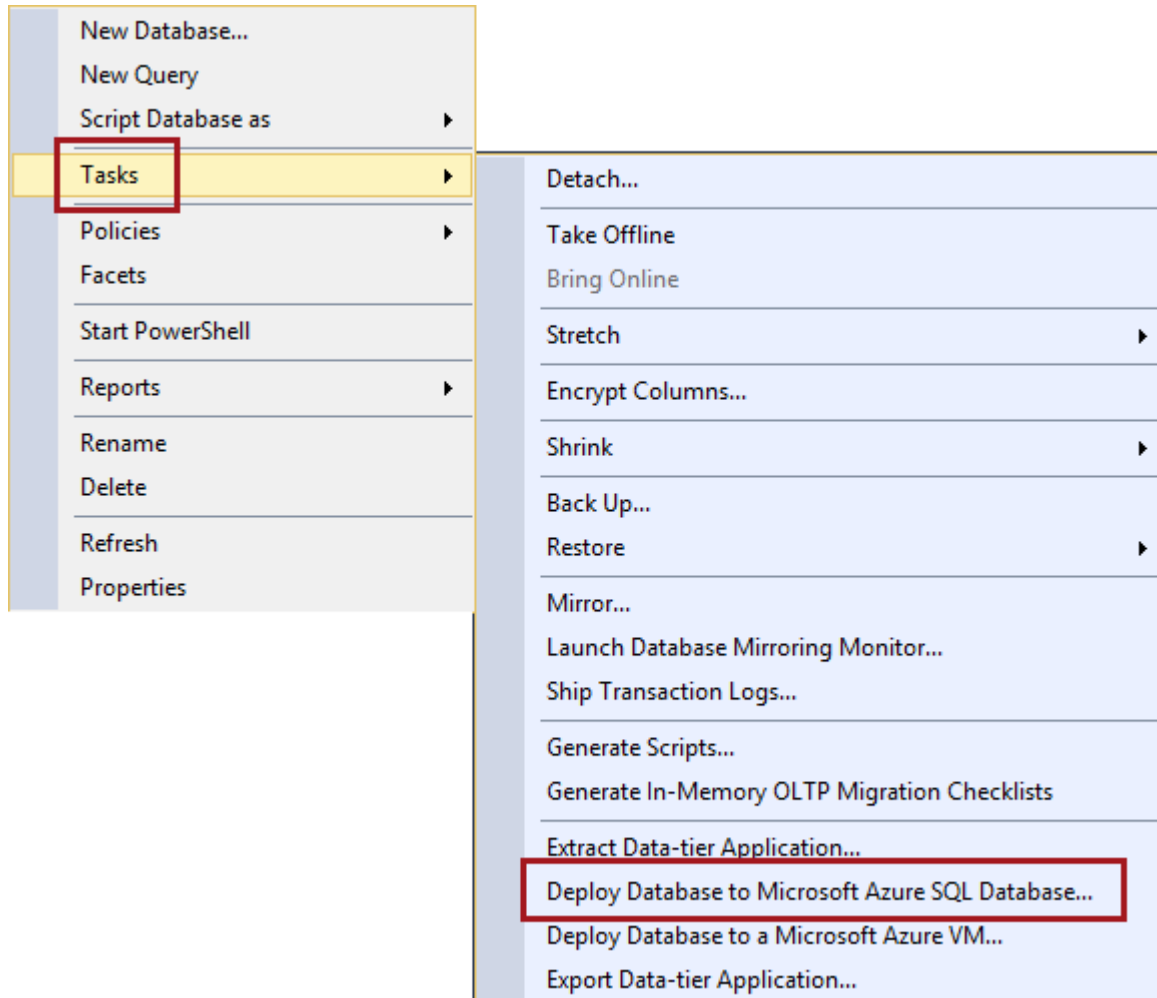   **SQL Server Management Studio** shortcut.

2. In the **Connect to Server** window, configure the following connection properties.
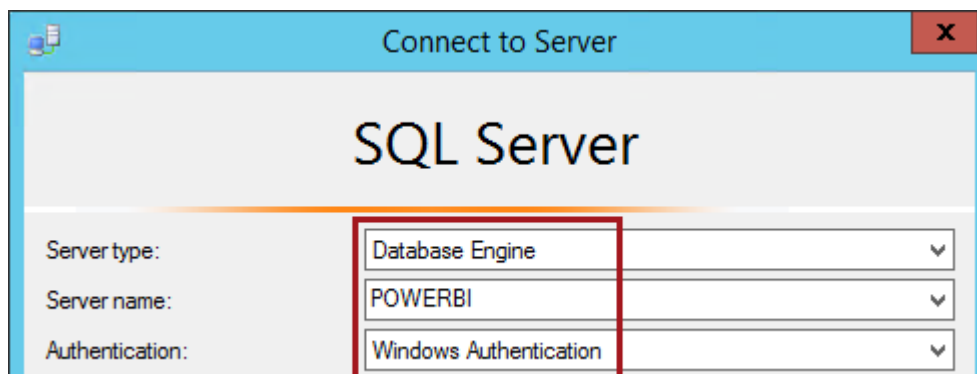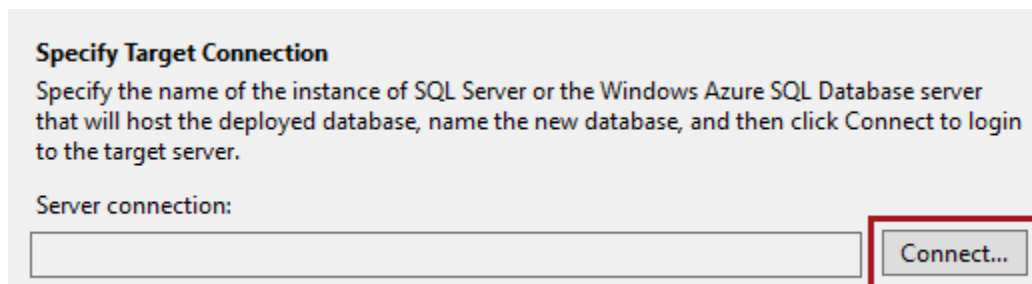

3. Click **Connect**.

4. In **Object Explorer** (located at the left), expand the **Databases** folder.

5. Right-click the **TailspinToys-US** database, and then select
Tasks | Deploy Database to Microsoft Azure SQL Database.

6. In the **Deploy Database** window, click **Next**.

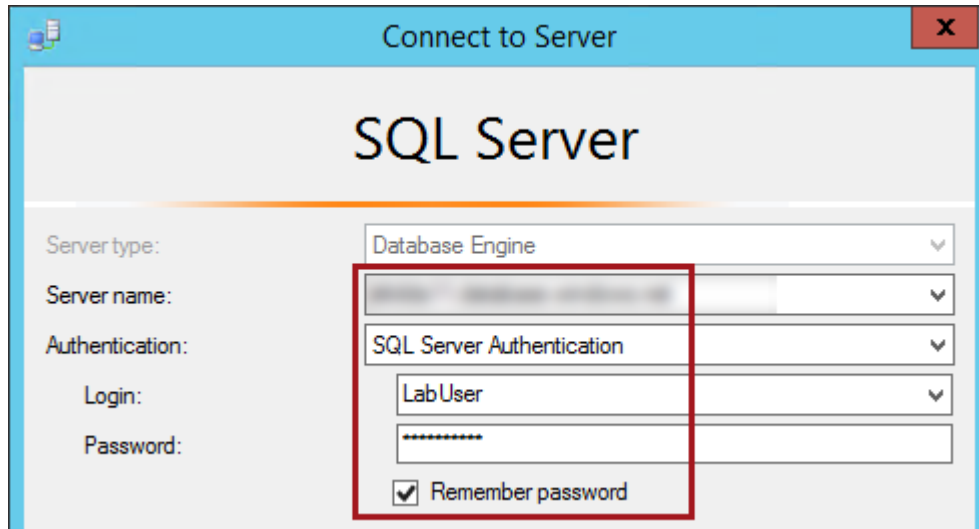7. In the **Deployment Settings** page, click **Connect**.

8. In the **Connect to Server** window, in the **Server Name** box, replace the text with the full name of your Azure SQL Database server.

    *Tip: You can retrieve the **SQL Server Name** value from the **Configurations.txt** file.*

9. In the **Authentication** dropdown list, select **SQL Server Authentication**.

10. In the **Login** and **Password** boxes, enter the SQL Server Admin credentials.

    *Tip: You can retrieve the login credentials from the **Configurations.txt** file.*

11. Check the **Remember Password** checkbox.

12. Click **Connect**.

13. In the **Deploy Database** window, click **Next**.

14. In the **Summary** page, click **Finish**.

15. Wait for the database deployment to complete.

    The deployment may take 2-4 minutes to complete.

16. When the deployment has completed, click **Close**.

17. To close SQL Server Management Studio, on the **File** menu, select **Exit**.

Provisioning a Power BI Workspace

In this exercise, you will open a sample solution designed to work with integrated a Power BI report into a web application.

*The solution consists of two projects. The **ProvisionSample** project to manage the workspace collection, content and data connections; and, the **EmbedSample** project which is a web application designed to integrate Power BI reporting.*

Opening the Sample Solution

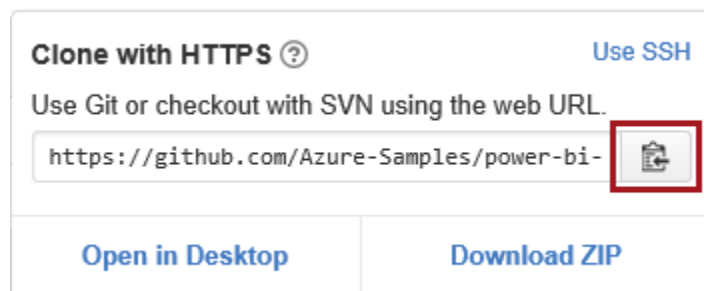In this task, you will open the sample solution.

1. In the Internet Explorer window, in a new browser tab, navigate to **https://github.com/Azure-Samples/power-bi-embedded-integrate-report-into-web-app**.

    *For you convenience, the URL can be copied from the **D:\PowerBI\Lab10\Assets\Snippets.txt** file.*

2. To copy the Git repository URL, click the **Clone or Download**.
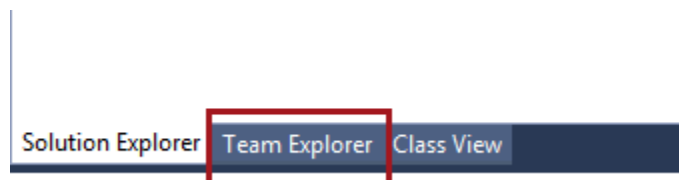

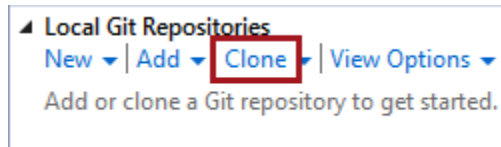3. Click the **Copy to Clipboard** command.

4. To open Visual Studio, on the taskbar, click the **Visual Studio** shortcut.

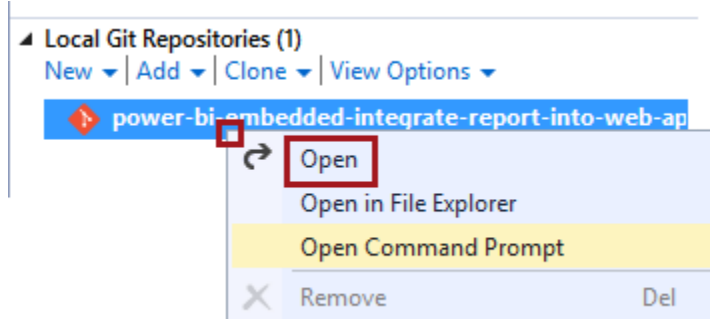5. In the top-right pane, select the **Team Explorer** view.

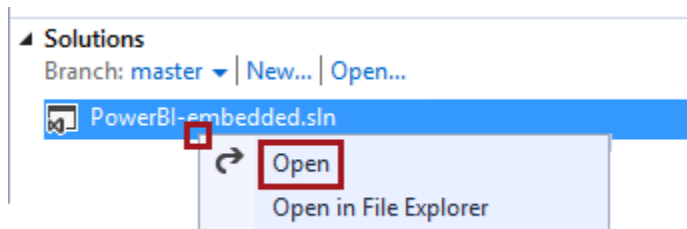6. In the **Local Git Repositories** section, click the **Clone** link.

7. In the first box, paste the Git repository URL.
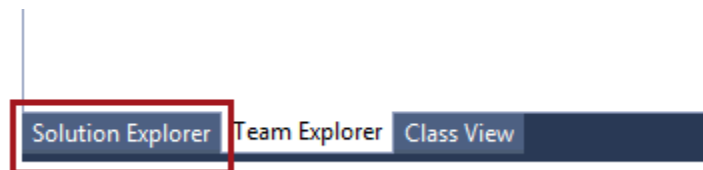
8. Click **Clone**.

9. Once cloned, to open the repository, right-click the repository, and then select **Open**.

10. To open the solution, right-click the solution, and then select **Open**.
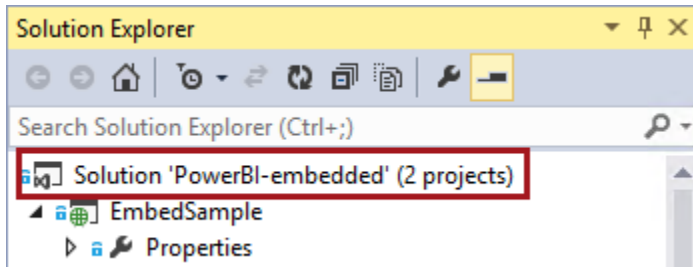

11. Select the **Solution Explorer** view.

12. In **Solution Explorer**, notice the two projects.

The projects contain references to external assemblies, and the solution must be rebuilt to download and restore several NuGet packages.

13. To rebuild the solution, right-click the **PowerBI-embedded** solution, and then select **Rebuild Solution**.

14. Wait until all NuGet packages have been restored.

15. Notice that the second project—**ProvisionSample**—is formatted bold.
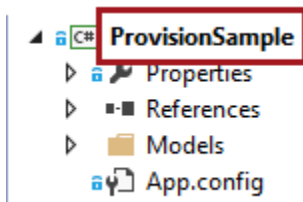
The bold font format indicates that it is the startup project.

*The **ProvisionSample** project is a simple menu-driven Visual C# console application, and leverages the Power BI REST API.*

*If you are interested to understand the code implementation for the functionality of this application, you can review the code in the **Program.cs** file.*

16. To run the **ProvisionSample** project, on the toolbar, click **Start**.

Creating a Workspace

In this task, you will use the console application to create a Power BI workspace.

You may be familiar with workspaces in the Power BI service (available for individuals and groups). While workspaces play a similar role, these are not workspaces hosted by the Power BI service.

Rather, they are workspaces managed within your Azure Power BI Embedded resource. A notable difference between the two models is that there is no explicit requirement to have your users registered in an Azure Active Directory (Azure AD) tenant.

1. In the console window, to provision a new workspace, enter **5**—take care not to press **Enter**.

2. When prompted for the **Workspace Collection Name**, from the **Configurations.txt** file, copy the **Workspace Collection Name** value, right-click the console title, and then select **Edit | Paste**.

Be sure not to paste in any trailing spaces.
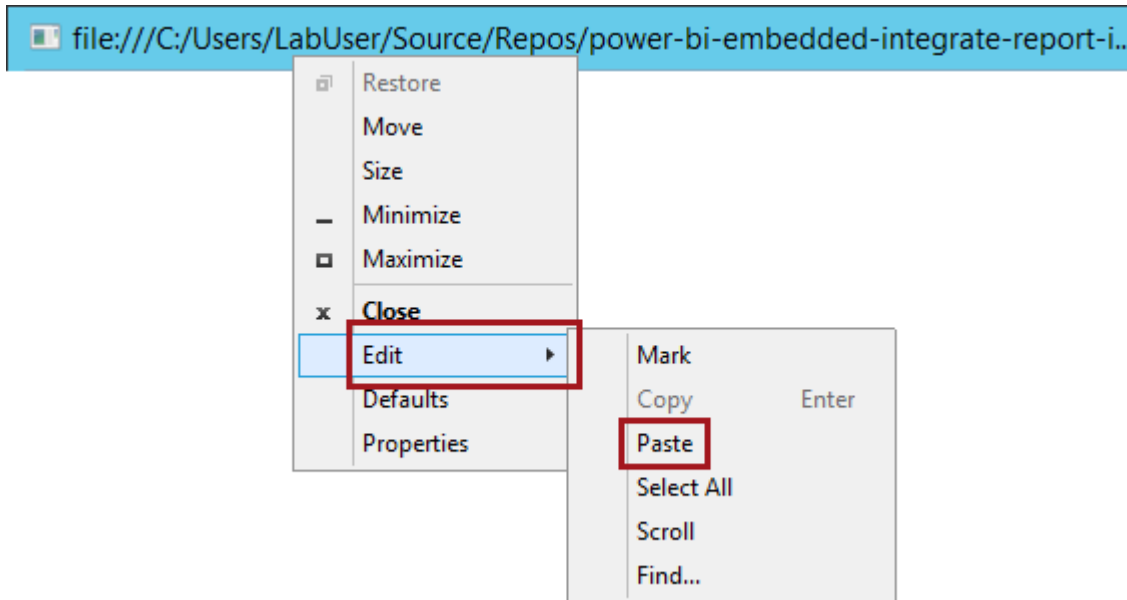
3. Press **Enter**.

4. When prompted for the **Access Key**, from the **Configurations.txt** file, copy in the **Workspace Collection Access Key** value, and then press **Enter**.

5. Verify that a Workspace ID is returned, and that the list of menu of options is output again.

If you receive an error, you should close the console window, and start the project again.

6. Leave the console application running.

7. Switch to the Azure Portal.

8. Navigate to your Power BI workspace collection blade, and verify that it contains a single workspace.
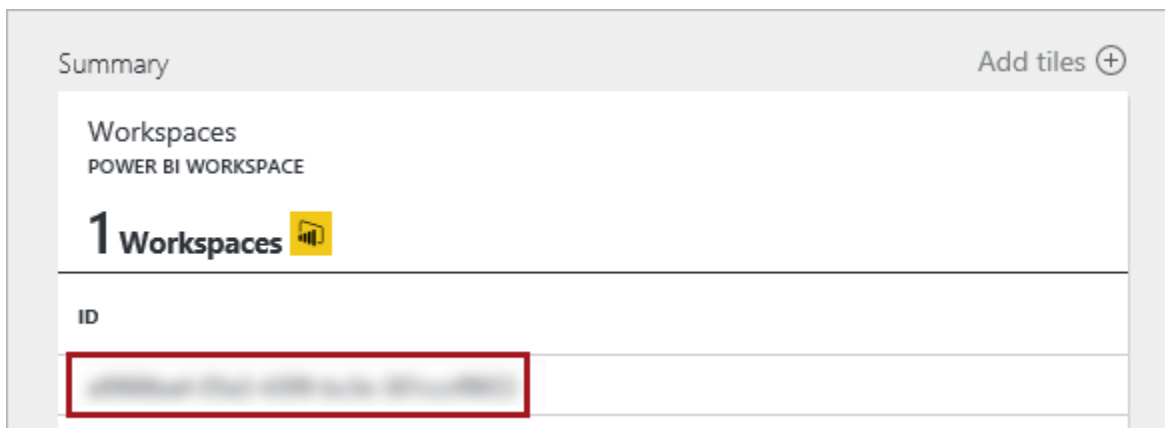
9. From the **Summary** tile, copy the **Workspace ID** value (GUID) to the clipboard.

10. In the **Configurations.txt** window, paste the clipboard content to the right of the **Workspace ID** label.
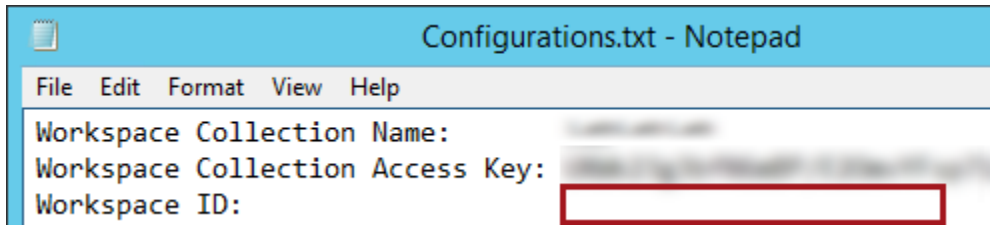
*The **Workspace ID** will be retrieved later when configuring the integration of a Power BI report into the web app.*
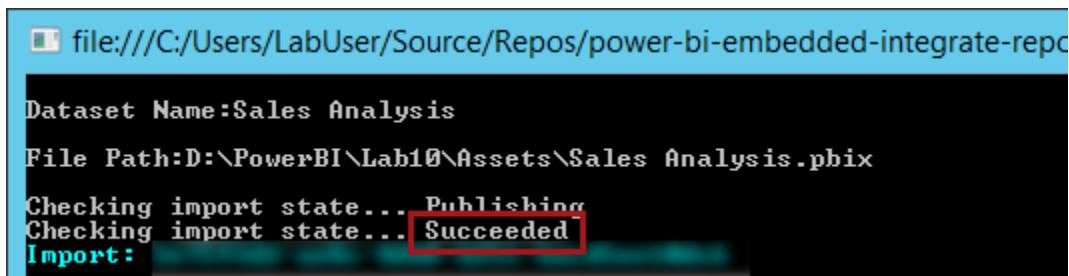
Uploading the Power BI Desktop File

In this task, you will upload the Power BI Desktop file to the workspace.

1. Switch to the console application.

2. To upload the Power BI Desktop file, enter **6**—take care not to press **Enter**.

3. When prompted for the **Dataset Name**, enter **Sales Analysis**, and then press **Enter**.

   *The web app will display the **Dataset Name** as the report name, providing a list of reports that the app user can select.*

4. When prompted for the **File Path**, copy and paste the **Power BI Desktop File** value— **D:\PowerBI\Lab10\Assets\Sales Analysis.pbix**—from the **Configurations.txt** file, and then press **Enter**.

5. Verify that the file was successfully imported.

6. Leave the console application running.

Updating the Power BI Desktop File Database Connection

In this task, you will update the Power BI desktop file database connection.

When the Power BI Desktop file was saved it was connected to a local SQL Server database instance. This task programmatically, and securely, updates the connection string and applies the database user credentials.

1. To update the Power BI Desktop file, enter **7**—take care not to press **Enter**.

   *Tip: You can copy and paste all values from the **Configurations.txt** file.*

2. When prompted for the **Username**, enter **LabUser**, and then press **Enter**.

3. When prompted for the **Password**, enter **Pass@word1**, and then press **Enter**.

4. When prompted for the **Connection String**, paste in the **SQL Server Connection String** value from the **Configurations.txt** file— and then press **Enter**.

5. Verify that the connection string was successfully updated.

   The Power BI Desktop file has now been imported into the workspace, and its connection string and database credentials allow it to connect to the Azure SQL Database deployed earlier in the lab.

6. To close the console application, click the **X** located at the top-right corner.
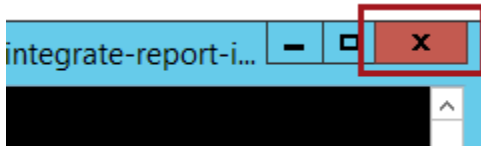
7. In **Solution Explorer**, collapse the **ProvisionSample** project.
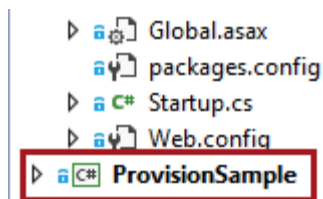
Configuring the Web App

In this exercise, you will complete the development of the sample ASP.NET MVC web application by configuring **web.config** values.

*The **EmbedSample** project is a Visual C# web app designed to enable users to view and interact with integrated Power BI reports, and simulates an online operational application. While it appears to represent a session with an authenticated customer, this is a simulation only to provide a realistic application experience. For the objectives of this lab, you will only be concerned with selecting reports from the left pane.*

The topic of authorization with app tokens is not covered in this lab.

*If you are interested to understand the code implementation for the functionality of this application, you can review the code in the **DashboardController.cs** file.*

Completing the Web App Development

In this task, you will set three **Web.config** app setting values so that the app can integrate with reports available in your workspace.

1. In **Solution Explorer**, right-click the **EmbedSample** project, and then select **Set as Startup Project**.

2. In **Solution Explorer**, right-click the **Web.config** file, and then select **Open**.

3. Scroll towards the bottom of the file, to locate the **appSettings** element.

```xml
<appSettings>
    <add key="powerbi:AccessKey" value="" />
    <add key="powerbi:ApiUrl" value="https://api.powerbi.com" />
    <add key="powerbi:WorkspaceCollection" value="" />
    <add key="powerbi:WorkspaceId" value="" />
</appSettings>
```

4. Insert values for the following three keys by retrieving values stored in the **Configurations.txt** file.

Key	Value
powerbi:AccessKey	(Paste in the **Workspace Collection Access Key** value)
powerbi:WorkspaceCollection	(Paste in the **Workspace Collection Name** value)
powerbi:WorkspaceId	(Paste in the **Workspace ID** value)

5. To save the file, on the **File** menu, select **Save Web.config**.

6. The close the fil, on the **File** menu, select **Close**.

7. To run the project, on the toolbar, click **Internet Explorer**.

---

8. In the new Internet Explorer window, when the app loads, hover the cursor in the left panel to reveal the panel contents.

9. Click **Reports** to expand the list of published reports.
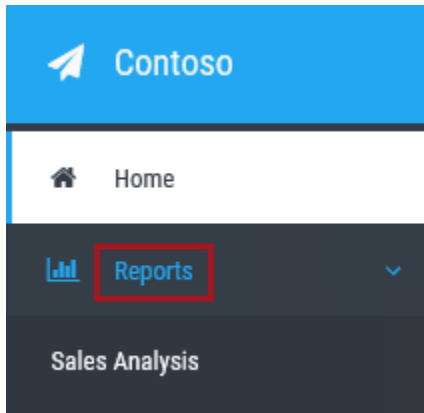
*The web app enumerates all Power BI Desktop files in the workspace, and lists them by dataset name. **Sales Analysis** is the dataset name you entered when you uploaded the Power BI Desktop file.*

Tip: You can upload additional Power BI Desktop files to the workspace, and view and interact with the reports in this web app.

10. Select the **Sales Analysis** report.

11. Review the integrated report, noticing that it is identical in look and functionality to the report developed in Power BI Desktop.

12. Interact with the report by modifying slicer filters, and clicking several columns, funnel bars, and state areas.

Recognize that each refresh of a visualization is retrieving data from the Azure SQL Database, and is also recorded as an embed render.

13. Close the Internet Explorer window.

Reviewing the Embed Renders

In this task, you will monitor the number of embed renders for your workspace collection.

1. Switch to the Azure Portal.

2. To reload the portal, press **F5** (the Power BI workspace collection blade does not yet have a refresh command).

3. In your Power BI workspace collection blade, review the value in the **Embed Renders** tile.

   There is a small degree of latency, and so continue to reload the portal until a non-zero value appears.
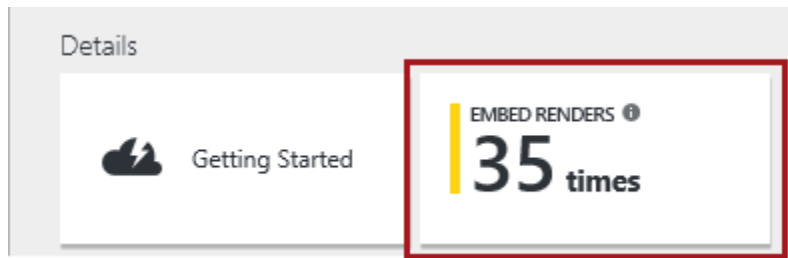
   *Until 31 August, 2016, preview pricing bills per 1,000 renders. Consider the **Summary** page which contains eight visuals. By filtering the page by a different year, it is possible that this would result in eight renders. However, it is important to bear in mind that it is also possible that a previous request (in the same session) for that year means the result could be in cache, and so would not be a new render.*

   From 1 September, 2016, general availability pricing will make the first 100 sessions per month free, and bill per session. Sessions will be billed at a flat rate, independent of the number of visual elements in a report or how frequently the report content is refreshed. A session ends when either the user closes the report, or one hour after the session was initiated, whichever comes first.

   For further information about Azure Power BI Embedded pricing, visit the Power BI Embedded Pricing page.

   To summarize the benefits derived from using Power BI Embedded in this app:

   - *Embed stunning, fully interactive visualizations in customer-facing apps by using REST APIs and SDK*

   - *Get the same data visualization experience on any device—desktop, mobile, or tablet*

   - *Build reports by using a broad range of modern Power BI data visualizations out-of-the-box*

   - *Build reports by using custom visualizations for even greater flexibility*

   - *Completely control user authentication and authorization choices with app tokens*

   - *Speed up time-to-value without redesigning an existing application*

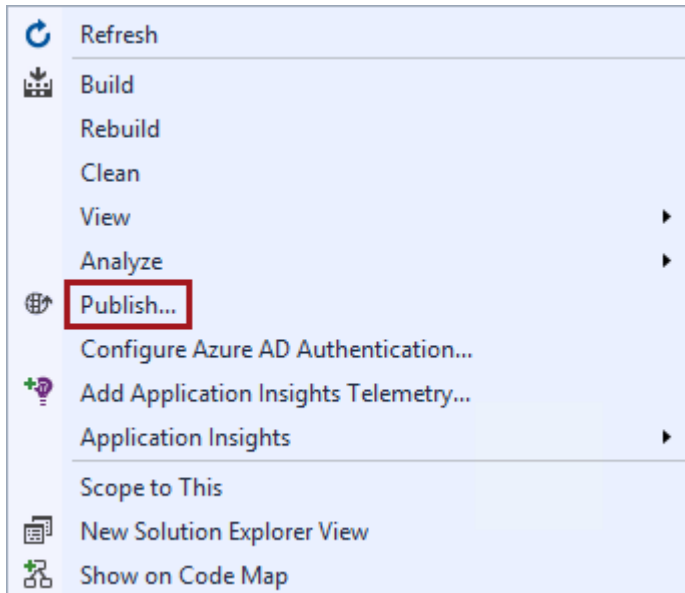   - *Pay only for what users request, with no upfront costs*

(Optional) Publishing the Web App

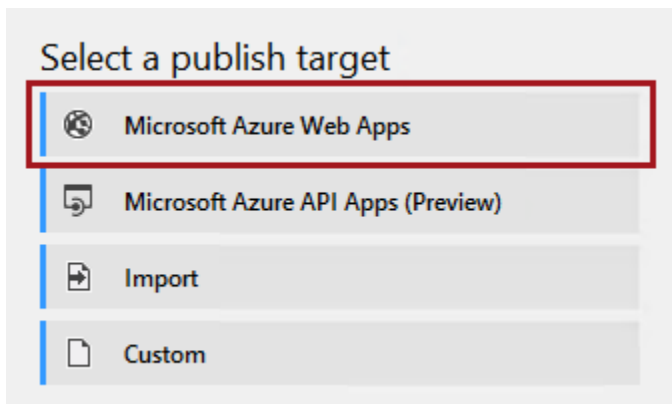In this exercise, you will publish the **EmbedSample** project to Azure Web Apps.

Publishing the Web App

In this task, you will publish the web app.

1. Return to Visual Studio.

2. In **Solution Explorer**, right-click the **EmbedSample** project, and then select **Publish**.
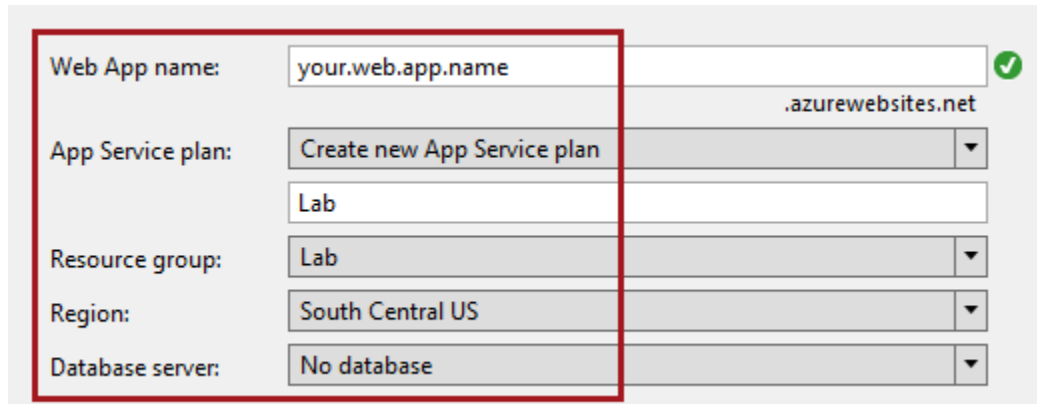

3. In the **Publish Web** window, in the **Select a Publish Target** list, select **Microsoft Azure Web Apps**.

4. In the **Microsoft Azure Web Apps** dialog window, in the **Add an Account** dropdown list, select **Add an Account**.

5. In the **Sign In to Your Account** window, complete the authentication process by using the same account with which you signed in to the Azure Portal.

---
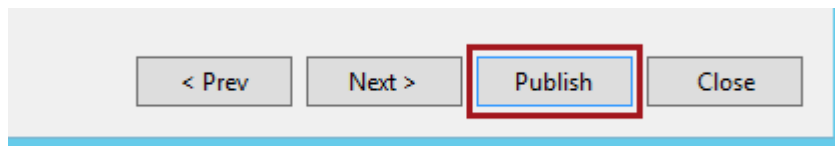
6. When signed in, in the **Microsoft Azure Web Apps** dialog window, to create a new web app, click **New**.

7. In the **Create Web App on Microsoft Azure** window, in the **Web App Name** box, enter a unique web app name.

8. In the **App Service Plan** dropdown list, select **Create New App Service Plan**.

9. In the following box, enter **Lab**.

10. In the **Resource Group** dropdown list, select **Lab**.

11. In the **Region** dropdown list, select **South Central US**.

12. Verify that the configuration looks like the following.

13. Click **Create**.

    It may take 1-2 minutes to create the web app.

14. When the web app has been created, in the **Publish Web** window, click **Publish**.

    It may take 3-5 minutes to publish the web app. When the web app has been published, an Internet Explorer window will open (it may open in the background).

15. Explore, and interact, with the published web app.

16. When finished, close the Internet Explorer window.

    You can also use any mobile device to browse to the published web app at http://<your.web.app.name>.azurewebsites.net.

Finishing Up

In this exercise, you will finish up by undoing the configurations made in this lab, and by closing opened applications.

If you no longer intend to work with the solution developed in this lab, it is important that you delete the Azure resource group to release data center resources.

Finishing Up

In this task, you will finish up by undoing the configurations made in this lab, and by closing opened applications.

1. In the Azure Portal, navigate to the **Lab** resource group.

2. Click **Delete**.

3. In the confirmation blade, in the **Type the Resource Group Name** box, enter **Lab**.

4. Click **Delete**.

5. Close the Internet Explorer window (all tabs).

6. In the Visual Studio window, on the **File** menu, select **Exit**.

7. To restore the **TailspinToys-US** database, open the **D:\PowerBI\Lab10\Assets\Cleanup.cmd** file.

Summary

In this lab, you integrated a Power BI report into an ASP.NET MVC web app by using the Microsoft Power BI Embedded Azure service. The report used DirectQuery to connect to an Azure SQL Database.

The exercises in this lab involved provisioning an Azure resource group consisting of Power BI Embedded and Azure SQL Database, and then using a sample Visual C# console application to provision a Power BI workspace, import a pre-developed Power BI Desktop file, and configure its database connection and credentials. A pre-developed Visual C# web app was then configured to integrate a report. Finally, an optional exercise involved publishing the web app to Azure.

Terms of Use

Document Version

#	Date	Author	Comments
1	02-MAY-2016	Peter Myers	Initial release Power BI Desktop v2.33.4337.281 Azure Power BI Embedded Preview Git repository commit 33a16fe4e9eeb98fa2c79776268336c91bbe088f
2	14-MAY-2016	Peter Myers	Error corrections Power BI Desktop v2.34.4372.501 Azure Power BI Embedded Preview Git repository commit 33a16fe4e9eeb98fa2c79776268336c91bbe088f
3	30-MAY-2016	Peter Myers	Updated report style Power BI Desktop v2.35.4399.381 Azure Power BI Embedded Preview Git repository commit 421e8fc228b0910dc907588f5c8b5de5153f4902
4	20-JUL-2016	Peter Myers	Updated Power BI logo Updated to new powerbi v1.00 VM set Updated to SQL Server 2016 GA TailspinToys-US database v1.02 Visual Studio 2015 Update 2 Power BI Desktop v2.36.4434.381 Git repository commit 016a842517433a3c6f147a9845a9611ae984177f Connection string is now set by an API call