

Azure Data Services

Hands-On Lab

Configuration Guide  
Version 2.8

8/30/2016

Prepared for: Guy Haycock

Azure SQL Database

Prepared by: Prowess Consulting

Table of Contents

[Architectural Overview 3](#_Toc425000559)

[Lab-Configuration Prerequisites 3](#_Toc425000560)

[Connecting Your Azure Account 10](#_Toc425000561)

[Deploying a New WingtipTickets Tenant Environment 12](#_Toc425000562)

[Appendix: Explanation of New-WTTEnvironment Parameters 18](#_Toc425000563)

[Appendix: Explanation of Web.config Properties 20](#_Toc425000564)

Table of Contents

[Architectural Overview 3](#_Toc455660201)

[Lab-Configuration Prerequisites 3](#_Toc455660202)

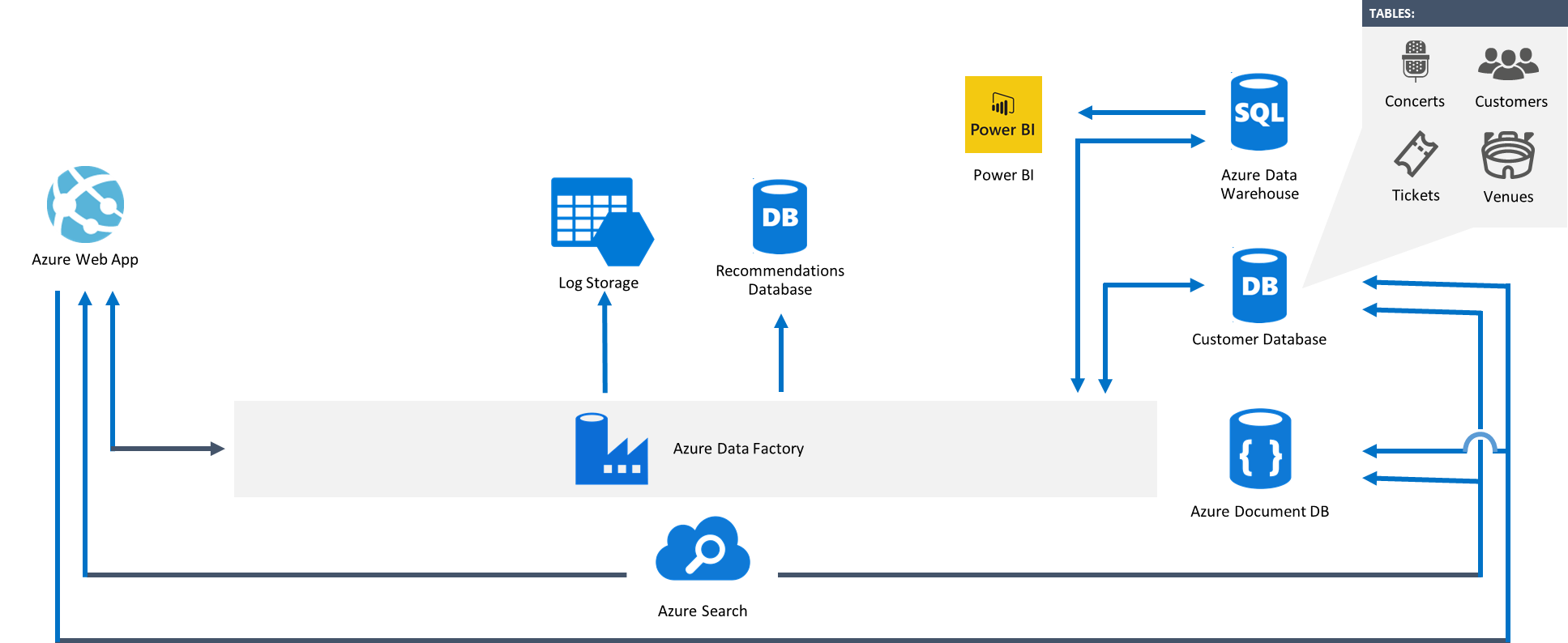
[Clone WingTip Tickets using GitHub Desktop 4](#_Toc455660203)

[Connecting Your Azure Account 5](#_Toc455660204)

[Deploying a New WingTipTickets Tenant Environment 7](#_Toc455660205)

[WingTip Tickets services deployed per PowerShell script 17](#_Toc455660206)

# Architectural Overview



**Figure 1** Overall architecture of the various lab components

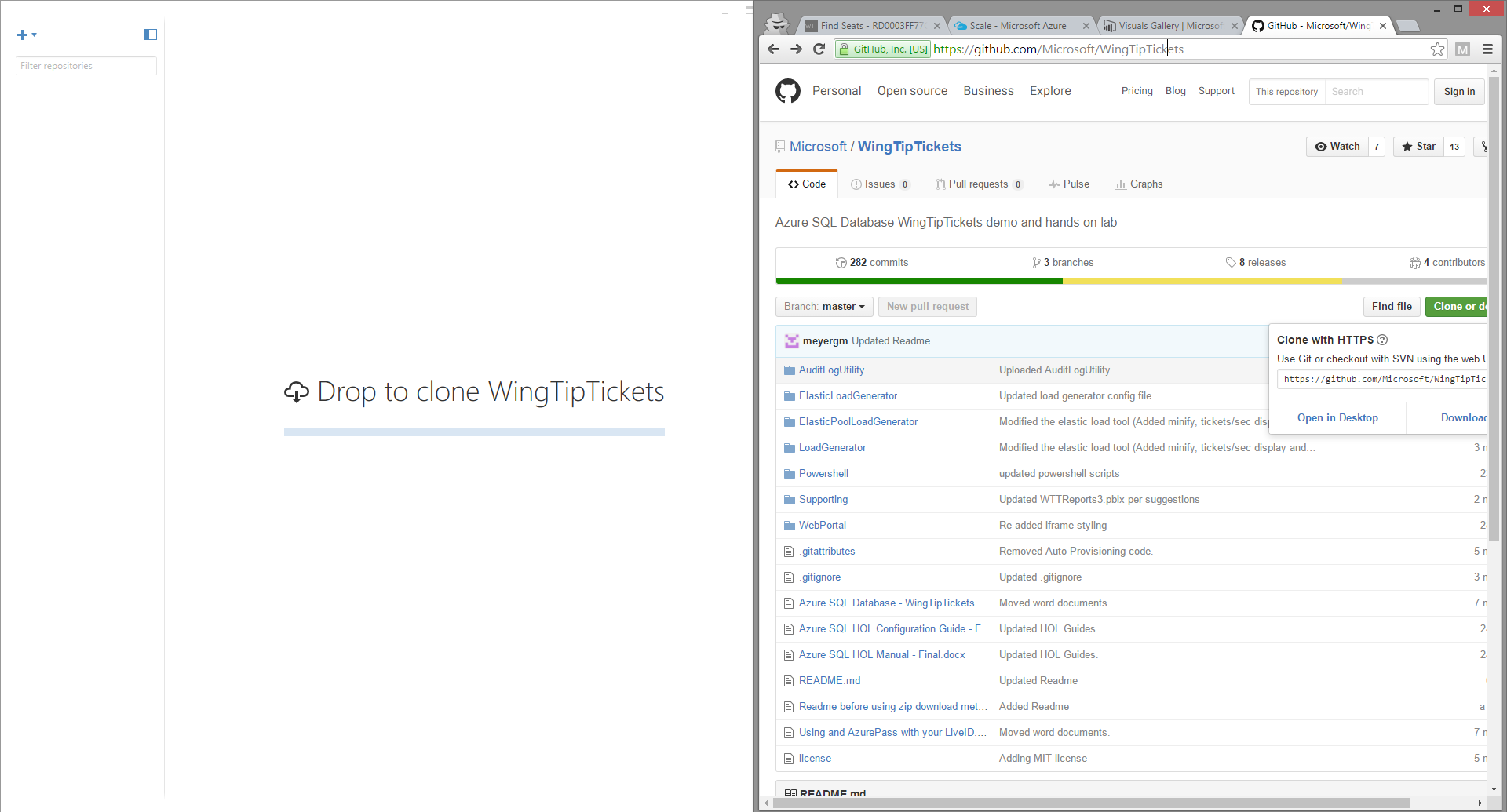
# Lab-Configuration Prerequisites

**Table 1** Lab-configuration prerequisites and setup instructions

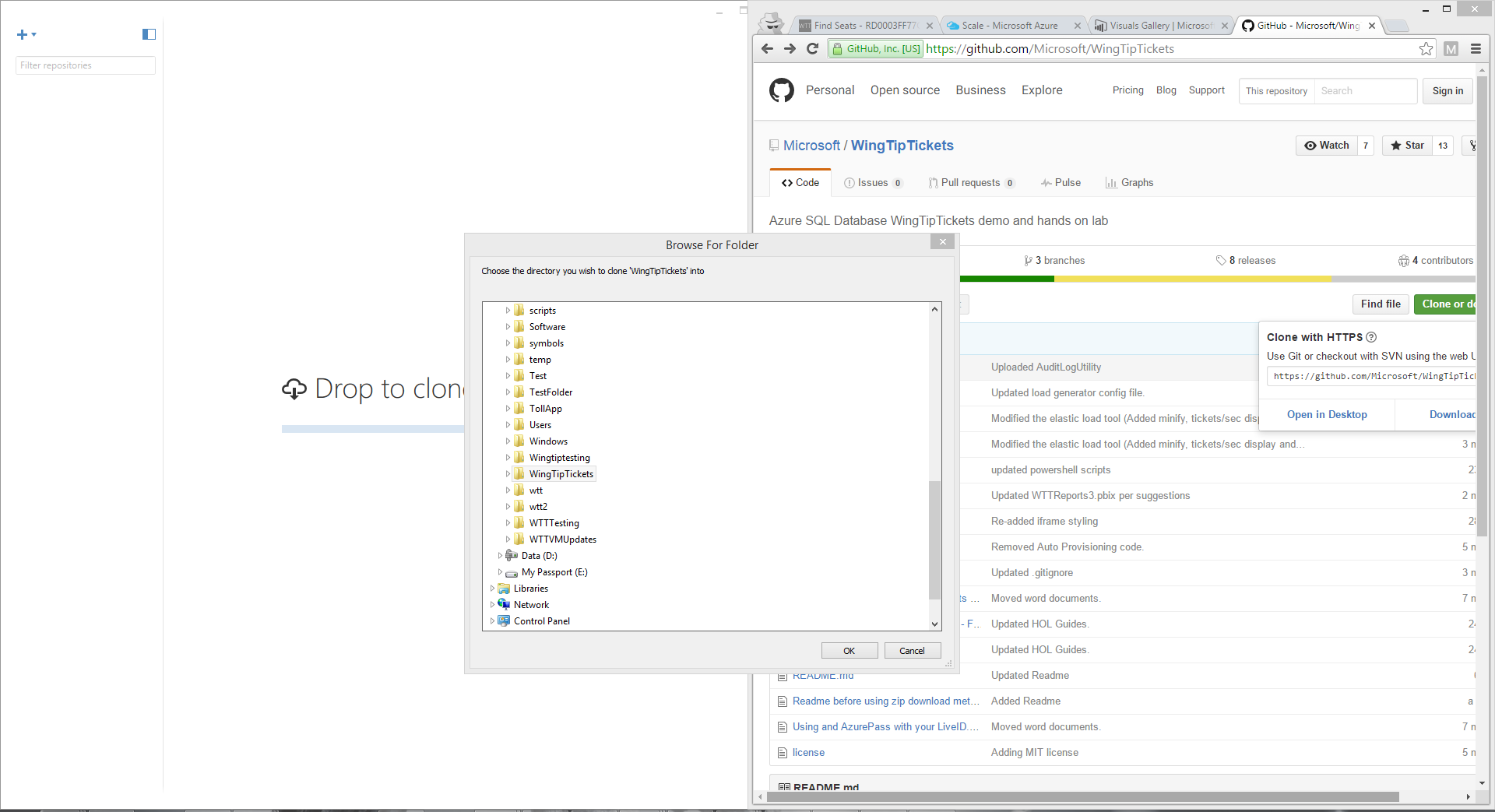
| **Prerequisite** | **Setup instructions** |
| --- | --- |
| **Microsoft account** | 1. If you don’t already have a Microsoft account, browse to [http://account.live.com](http://account.live.com/). 2. Click **Sign up now**. |
| **Microsoft Azure account** | 1. If you don’t already have an Azure account, browse to <http://azure.microsoft.com/en-us/pricing/free-trial/>. 2. Click **Try it now**. |
| **Microsoft Azure PowerShell** | 1. Browse to <http://azure.microsoft.com/en-us/downloads/>. 2. Under **Command-line tools**, **Windows PowerShell**, click **Install**. 3. Verify that the version being installed is at least 3.0.0 (released September 29, 2016). |
| **Microsoft SQL Server Management Studio** | 1. If SQL Server Management Studio is not installed, and you’d like to run queries against the databases, download and install SQL Server Management Studio from <https://msdn.microsoft.com/en-us/library/mt238290.aspx>. |
| **Microsoft SQL Server PowerShell Tools** | 1. Browse to <https://www.microsoft.com/en-us/download/details.aspx?id=52676>. 2. Click **Download**. 3. Locate **ENU\x64\PowerShellTools.msi**. 4. Click **Next**. 5. Click **Run**. |
| **GitHub client** | 1. Browse to <https://desktop.github.com/>. 2. Select **Download GitHub Desktop**. 3. Install **GitHub Desktop**. |
| **Microsoft Power Query for Excel 2013 (optional)** | 1. If Power Query for Excel is not installed, and you’d like to complete the Auditing section in the hands-on lab (HOL) manual, download and install Power Query from <https://www.microsoft.com/en-us/download/details.aspx?id=39379>. |
| **Microsoft Visual Studio (optional)** | 1. If Visual Studio is not installed, and you’d like to explore any of the source code, download and install Visual Studio from <http://go.microsoft.com/?linkid=9832446&clcid=0x409>. |
| **Microsoft Azure .NET software-development kit (SDK) (if installing Visual Studio)** | 1. Browse to <http://azure.microsoft.com/en-us/downloads/>. 2. Under **SDKs**, **.NET**, select the installer for your version of Visual Studio (for example, **VS 2013 Install**). |

# Clone WingTip Tickets by Using GitHub Desktop

1. Browse to <https://github.com/Microsoft/WingTipTickets>.
2. Launch **GitHub Desktop**.
3. Drag and drop the URL from the browser to the desktop client.

  
**Figure 2** Add WingTip Tickets to the GitHub Desktop client

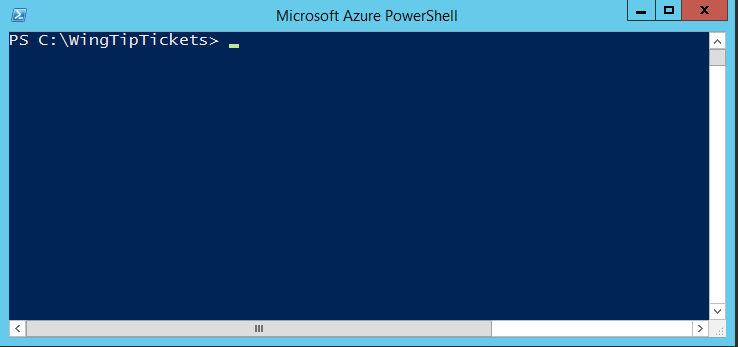
1. Select a location to store the WingTip Tickets files; for example, C:\WingTipTickets.

  
***Figure 3*** *Save the WingTip Tickets solution in a local folder*

1. Click **OK**.

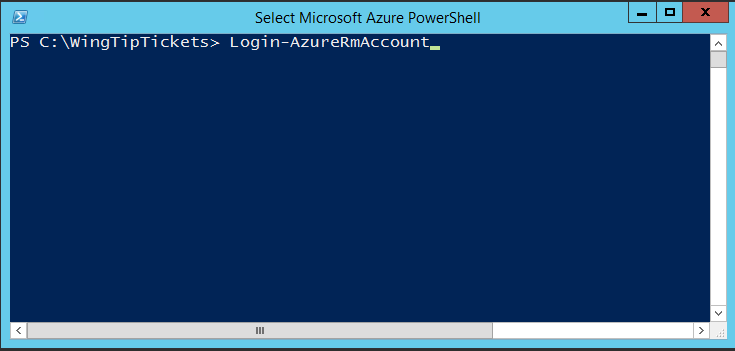
# Connecting Your Azure Account

1. Launch a Microsoft Azure PowerShell session as an administrator (click **Run as administrator**), and then browse to the folder where you’ve saved the scripts.



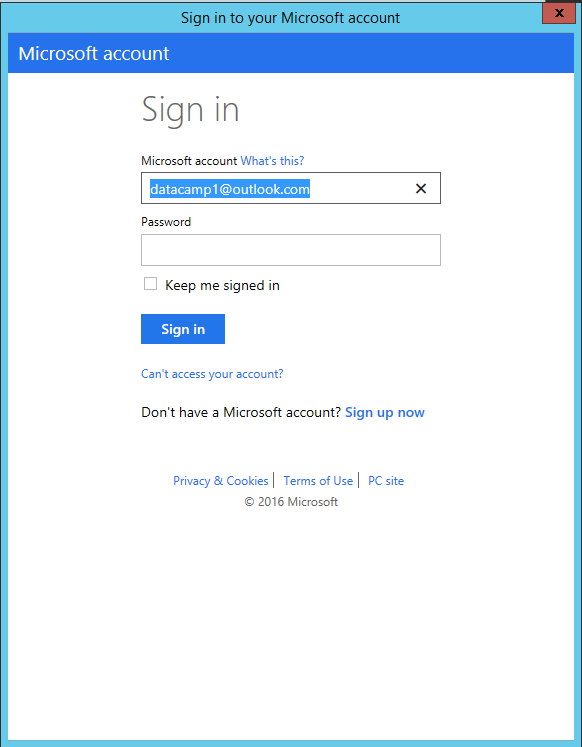
**Figure 4** A Microsoft Azure PowerShell command-line interface session

1. Connect to your Azure account by typing **Login-AzureRMAccount**.



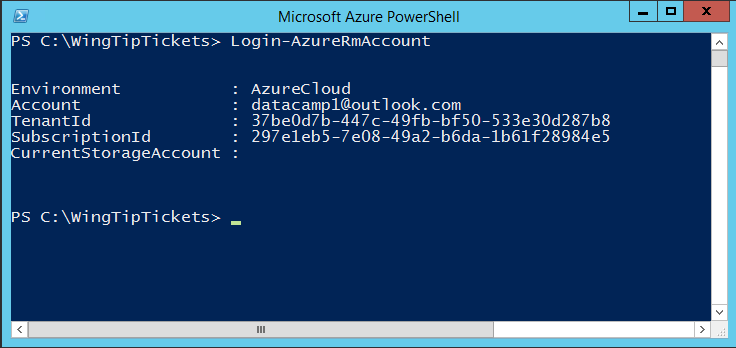
**Figure 5** Connect to your Azure account through Azure PowerShell

1. Type your Azure account credentials.



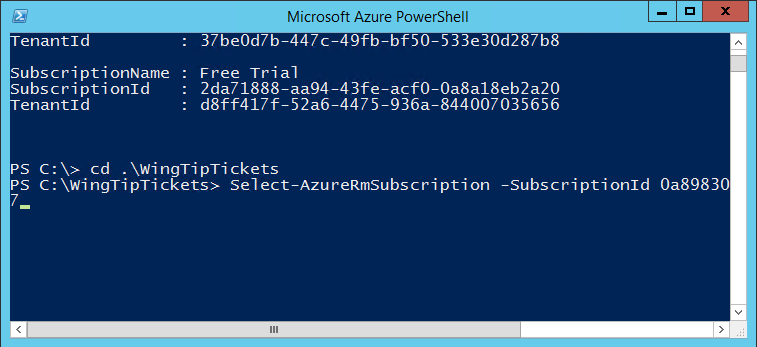
**Figure 6** Type your Azure account credentials

Your account should be linked to your primary subscription, as shown in Figure 7.



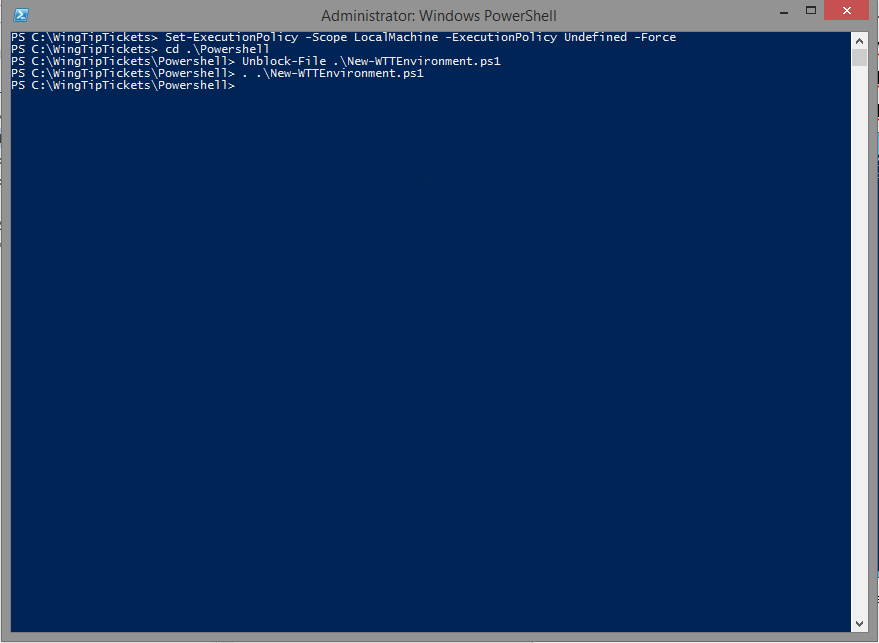
**Figure 7** Subscription confirmation in Azure PowerShell

1. If you have more than one Azure Subscription, type Get-AzureRMSubscription.
2. At the command prompt, type Select-AzureRMSubscription –SubscriptionId *YOUR SUBSCRIPTON ID* (as shown in Figure 8).

  
**Figure 8** You can also change or supply your Microsoft Azure credentials through the Select-AzureRMSubscription command in Microsoft Azure PowerShell

# Deploying a New WingTip Tickets Tenant Environment

1. Run the following commands to load the Azure PowerShell script:
   1. **PS C:\WingTipTickets\PowerShell>Set-ExecutionPolicy -Scope LocalMachine -ExecutionPolicy Unrestricted -Force**
   2. **PS C:\WingTipTickets\PowerShell>Unblock-file .\New-WTTEnvironment.ps1**
   3. **PS C:\WingTipTickets\PowerShell>. .\New-WTTEnvironment.ps1**



**Figure 9** Deploy a new tenant environment using the New-WTTEnvironment Azure PowerShell script

**Note**:To load the Azure PowerShell script, you must type a period and then a space before the path to the script: .\New-WTTEnvironment.ps1

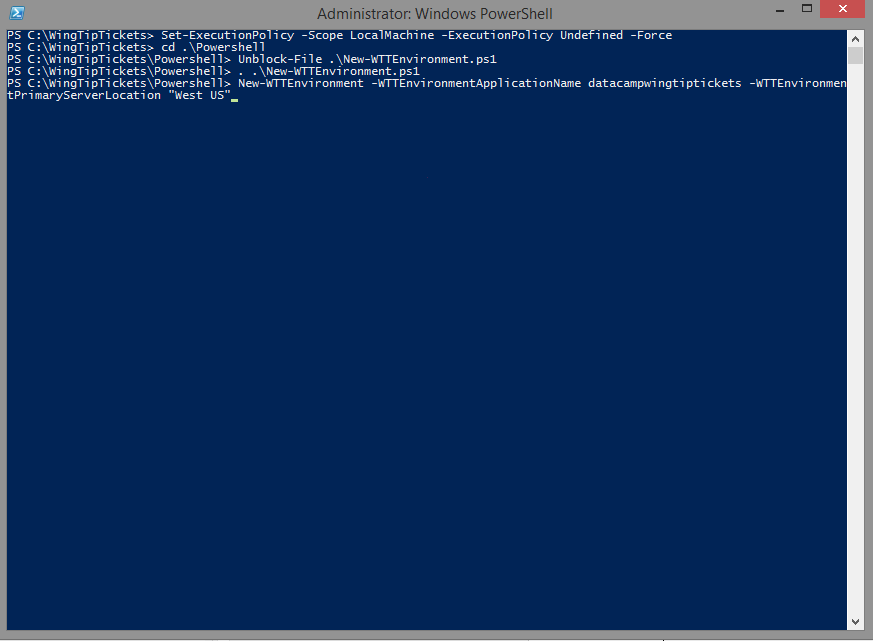
1. To verify the script is loaded, type **New-WTTEnvironment -W**, and then press Tab, which should autocomplete to WTTEnvironmentApplicationName.

**Note**: -WTTEnvironmentApplicationName is the most important value because it’s used to prefix Azure resources—for example, storage accounts, web apps, and database servers.

Because this HOL uses the Azure public cloud, it is crucial that you choose a name that is unique in order for the lab to work. For example: datacamp followed by your initials: datacamp*xx*

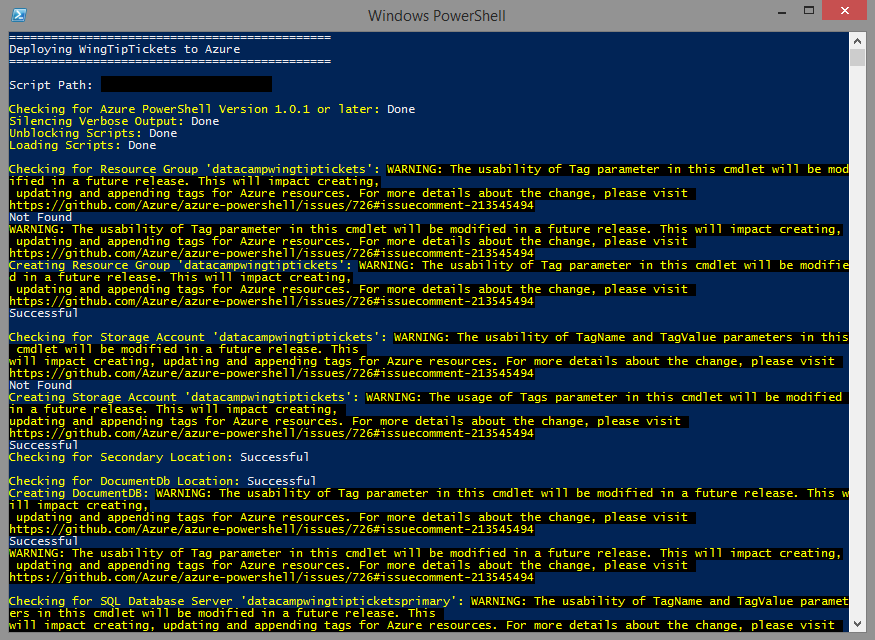
1. Select a location from the list of valid Azure datacenter locations to deploy the WingTip Tickets services. Valid locations include:
   1. East US
   2. West US
   3. South Central US
   4. North Central US
   5. Central US
   6. East Asia
   7. West Europe
   8. East US 2
   9. Japan East
   10. Japan West
   11. Brazil South
   12. North Europe
   13. Southeast Asia
   14. Australia East
   15. Australia Southeast
   16. Canada Central
   17. Canada East
2. Run the cmdlet using syntax similar to the following; remember to substitute the *xx* in datacamp*xx*for your initials, and the server location requires double quotes and the location to be spelled out according to the list above: **New-WTTEnvironment -WTTEnvironmentApplicationName *xx*julieandtheplantes –WTTEnvironmentPrimaryServerLocation “West US”**

Depending on your network connection, setup should take no more than 90 minutes.



**Figure 10** Example of creating a new deployment in Azure PowerShell

**Note**:During setup, Azure PowerShell will output a number of success messages in green and informational messages in yellow.



**Figure 11** Azure PowerShell success and informational messages that you might encounter during setup (this example shows creating a new deployment)

**Note**:If you have issues setting up the environment, you’ll need to review the errors that are returned to determine which Azure PowerShell function is causing an issue and, subsequently, what is causing the error.

The most common issues are due to name conflicts, webapp-package-upload failures, and subscription quota limits.

For name conflicts, you can remove a deployment by running: **PS C:\Scripts> Remove-WTTEnvironment -WTTEnvironmentApplicationName datacamp*xx***. This will delete any resources that were created by the New-WTTEnvironment cmdlet. Then try to re-run the New-WTTEnvironment cmdlet again as in step 4 using a different   
-WTTEnvironmentApplicationName (for example, datacamp*xx*).

For package-upload failures, re-running the same command that you previously ran in step 4 will try the upload again.

For pay-as-you-go subscription-quota-limit-related issues, check the following:

* Typically there is a limit of six Azure SQL Database servers. Make sure there are no more than four Azure SQL Database servers in your subscription before running the New-WTTEnvironment cmdlet.
* Typically there is a limit of one free Azure Search service. Make sure there are no Azure Search services in your subscription before running the New-WTTEnvironment cmdlet.
* For other issues, please refer to [FAQ section](#FAQ) of this document.

1. Once the deployment is completed, open a browser and browse to [http://datacamp*xx*.trafficmanager.net](http://datacampxx.trafficmanager.net) (remember to replace *xx* with the datacamp number or deployment name).



**Figure 12** Website running correctly on the primary web app

## Appendix A: Explanation of New-WTTEnvironment Parameters

**Table 2** New-WTTEnvironment parameters

| **Parameter** | **Purpose** | **Default Value** |
| --- | --- | --- |
| -WTTEnvironmentApplicationName | Name that will differentiate your WingTip Tickets tenant environment from others running in the Azure public cloud (in this example, datacamp*xx*) |  |
| -AzureSqlDatabaseServerAdministratorUserName | Database server-administrator user name | developer |
| -AzureSqlDatabaseServerAdministratorPassword | Database server-administrator password | P@ssword1 |
| -AzureSqlDatabaseServerVersion | Azure SQL Database server version | 12.0 |
| -AzureSqlDatabaseName | Name of the tenant database | Customer1 |
| -AzureWebSiteWebDeployPackagePath | Path to the Azure web-app web-deploy packages | *<unzip location>*\Scripts\Packages |
| -AzureWebSitePrimaryWebDeployPackageName | Primary web-app web-deploy package name | primarypackage.zip |
| -AzureWebSiteSecondaryWebDeployPackageName | Secondary web-app web-deploy package name | secondarypackage.zip |
| -WTTEnvironmentPrimaryServerLocation | Azure datacenter region | *<auto-configured based on capacity>* |

## Appendix B: Explanation of Web.config Properties

**Table 3** Web.config property names and descriptions

|  |  |  |
| --- | --- | --- |
| **Property name** | **Description** | **Value** |
| TenantEventTypeGenre | Tenant event type used to dynamically theme the site; valid values include: pop, rock, and classical | Pop |
| TenantEventName | Tenant (application) name, used to dynamically theme the site | Set from  –WTTEnvironmentApplicationName <value> |
| PrimaryDatabaseServer | Name for the primary Azure SQL Database server where concerts, customers, venues, and tickets tables exist | Set from  –WTTEnvironmentApplicationName <value> + primary |
| SecondaryDatabaseServer | Name of the Azure SQL Database server that is configured as the target server (also known as the secondary) for geo-replication | Set from  –WTTEnvironmentApplicationName <value> + secondary |
| DatabaseUserName | User name to be used for all application-related connections to the Azure SQL Database servers | Set from  -AzureSqlDatabaseServerAdministratorUserName |
| DatabaseUserPassword | Password to be used for all application-related connections to the Azure SQL Database servers | Set from  -AzureSqlDatabaseServerAdministratorPassword |
| TenantDbName | Name for the Azure SQL Database tenant database where concerts, customers, venues, and tickets tables exist | Set from  -AzureSqlDatabaseName |
| SearchServiceName | Name of the Azure Search service that indexes a view of the tables in the TenantDbName database | Set from  –WTTEnvironmentApplicationName <value> |
| SearchServiceKey | Azure Search service key | Dynamically retrieved during setup |
| DocumentDbUri | URI of the Azure DocumentDB service that stores ad-hoc venue information | Dynamically retrieved during setup |
| DocumentDbKey | Primary access key used to access the DocumentDB service | Dynamically retrieved during setup |
| RecommendationDatabaseServer | Name for the primary Azure SQL Database server where concerts tables exist | Dynamically retrieved during setup |
| RecommendationDatabase | Name for the Azure Data Factory Azure SQL Database where concert information exists | Dynamically retrieved during setup |
| powerbiSigningKey | Primary access key of the Azure Power BI Workspace Collection | Dynamically retrieved during setup |
| powerbiWorkspaceCollection | Name of the Azure Power BI Workspace Collection | Dynamically retrieved during setup |
| powerbiWorkspaceId | Name of the Azure Power BI Workspace | Dynamically retrieved during setup |
| SeatMapReportId | Report ID of the SeatMapReport in the Azure Power BI Workspace | Dynamically retrieved during setup |

## Frequently Asked Questions (FAQ)

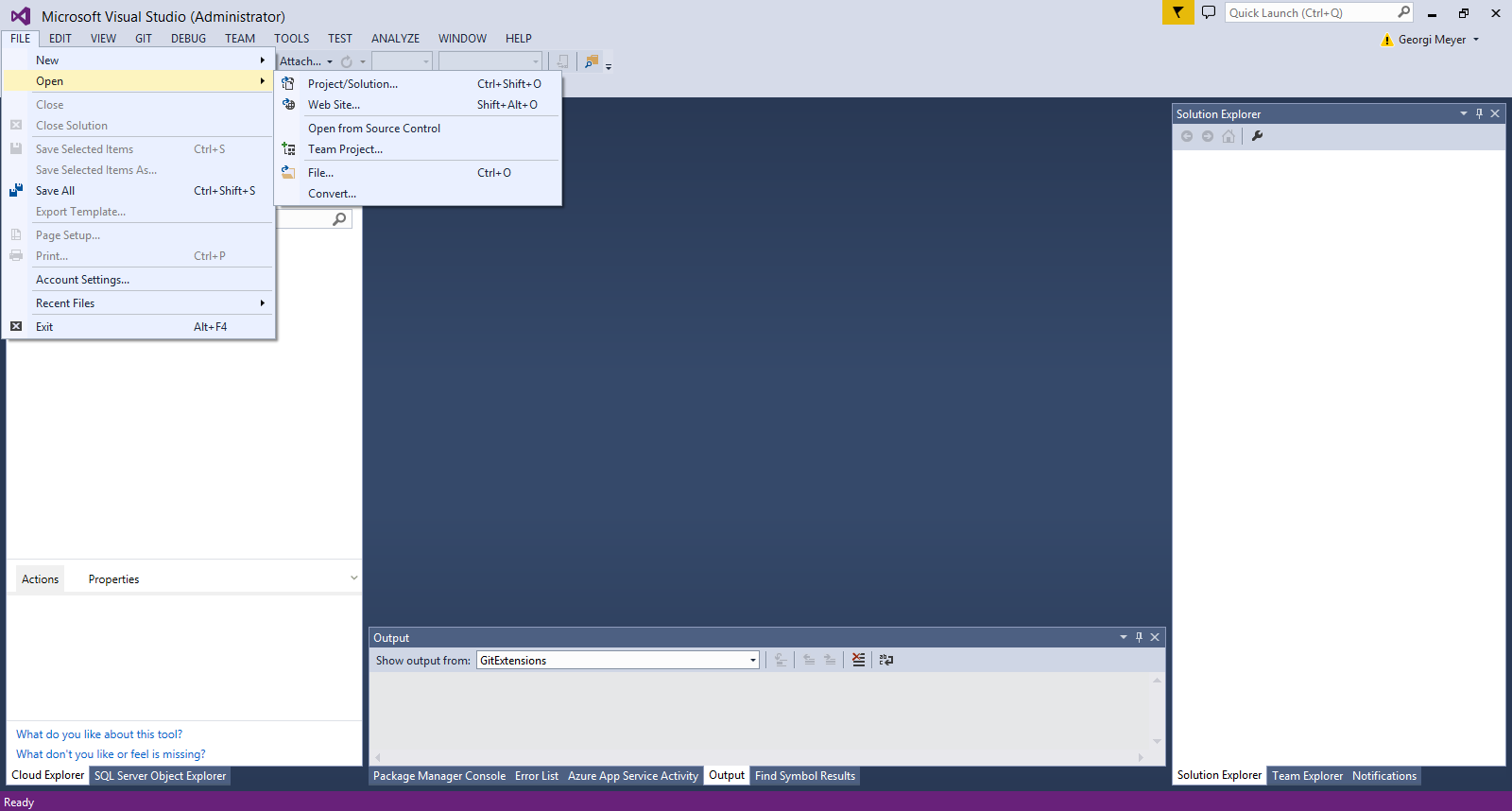
1. **Q**: If I have an active Microsoft Developer Network (MSDN) subscription and enough resources, why is my provisioning failing?

**A**: Please check that you have Azure Search in your subscription by going to <http://portal.azure.com>. Log on with your account, and then click **Browse All**. If Azure Search is not shown as available, you need to switch to a subscription that does include Azure Search.

1. **Q**: Why does my PowerShell deployment fail when uploading the web-application files?  
   **A**: There is a known issue where, if the WingTip Tickets project has been downloaded through the download-zip option, some files might be corrupted. The potentially corrupted files include:
   1. \PowerShell\Packages\PrimaryPackage.zip
   2. \PowerShell\Packages\SecondaryPackage.zip
   3. \PowerShell\Packages\ProductRecDataGenerator.zip

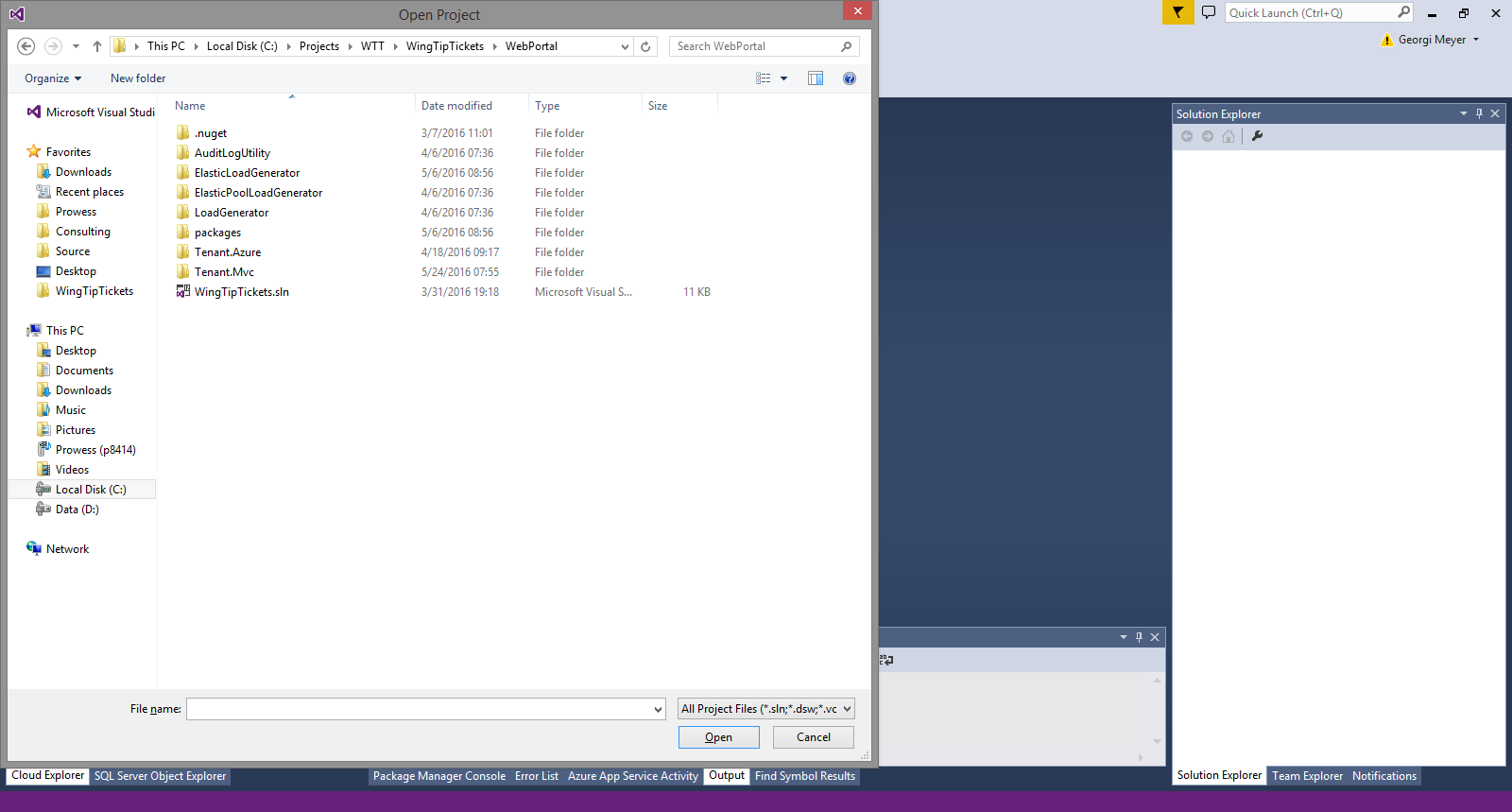
If you encounter this issue, follow these steps to repackage the potentially corrupted files for use for deployment:

1. Launch Visual Studio 2013/2015.
2. Click **File** > **Open Project/Solution**.



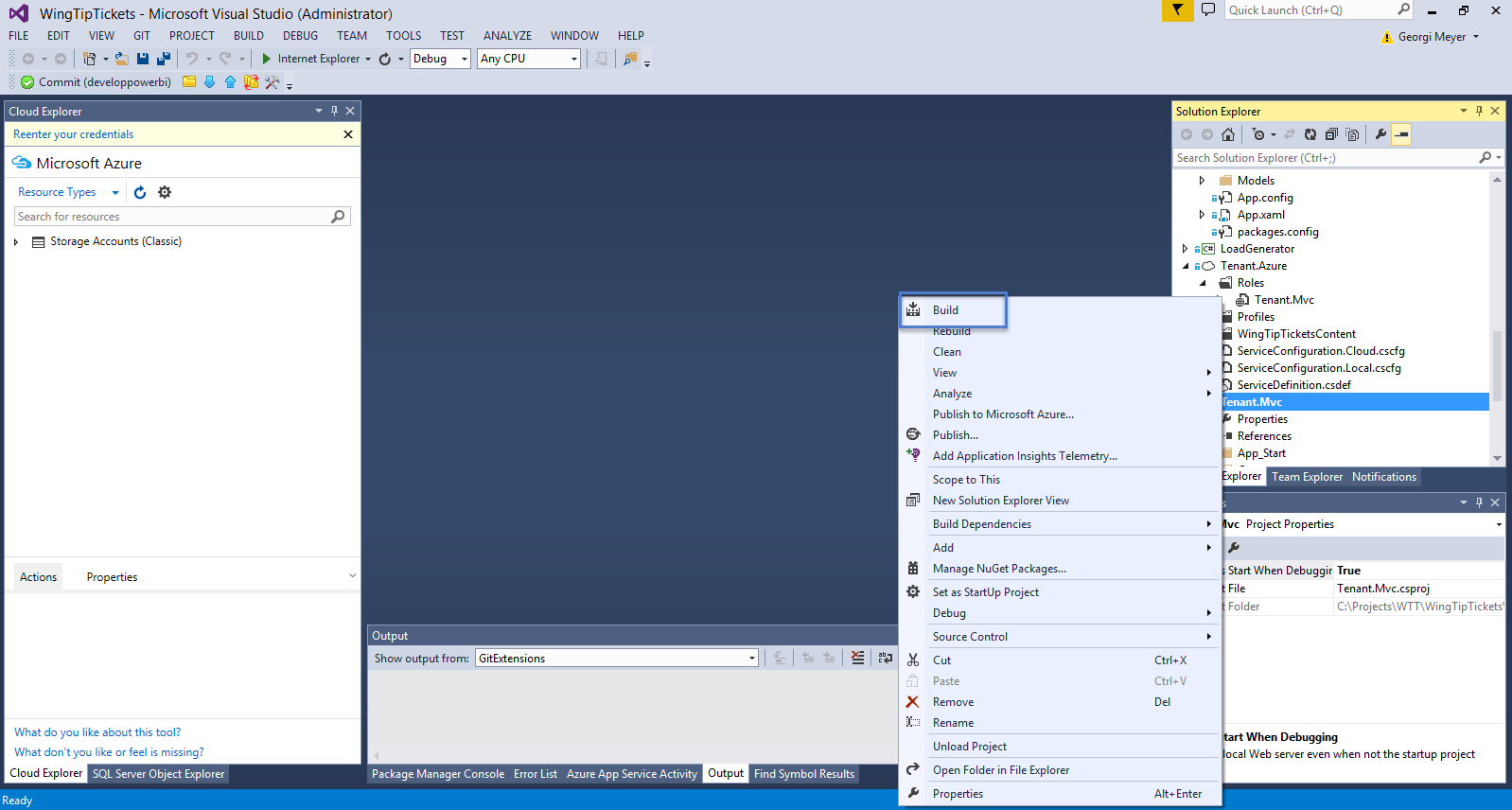
***Figure 13*** *Open the Visual Studio project*

1. Browse to the downloaded WingTip Tickets files.
2. Open **\WebPortal**, and then select **WingTipTickets.sln**.



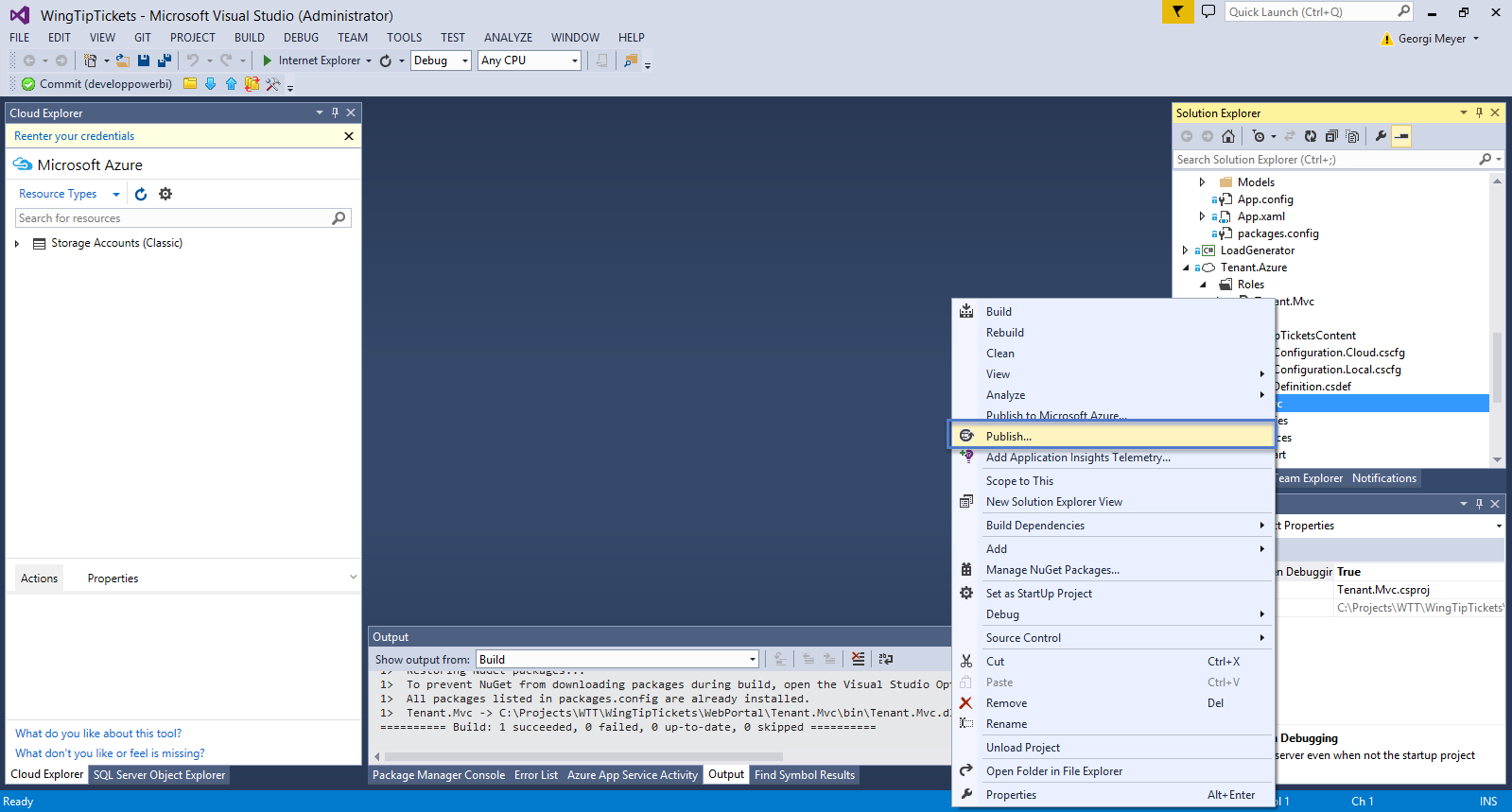
***Figure 14*** *Open the WingTipTickets.sln project solution*

1. Right-click **Tenant.mvc**, and then select **Build**.



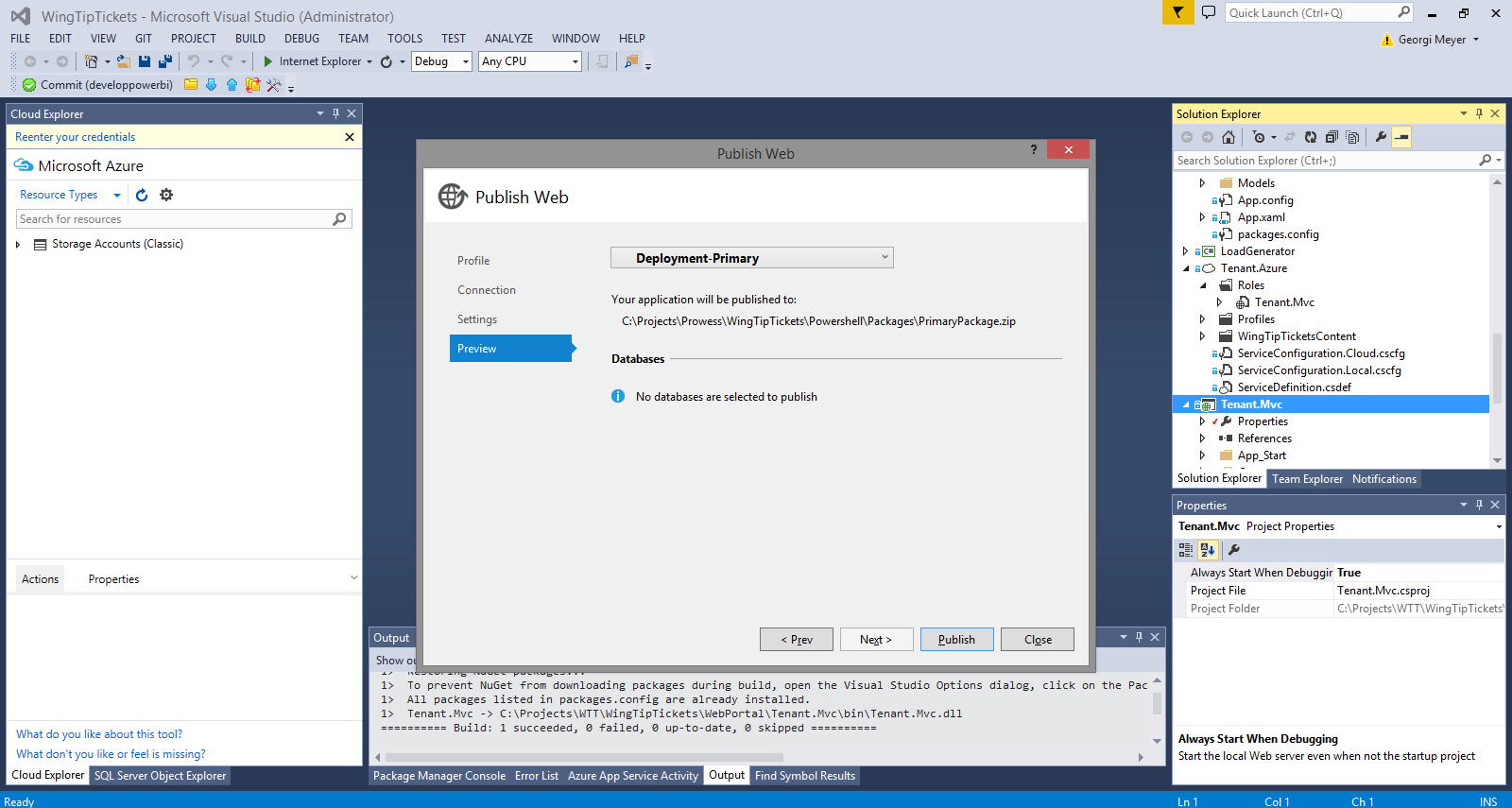
***Figure 15*** *Build the WingTip Tickets Tenant.mvc web application*

1. Right-click **Tenant.mvc**, and then select **Publish**.



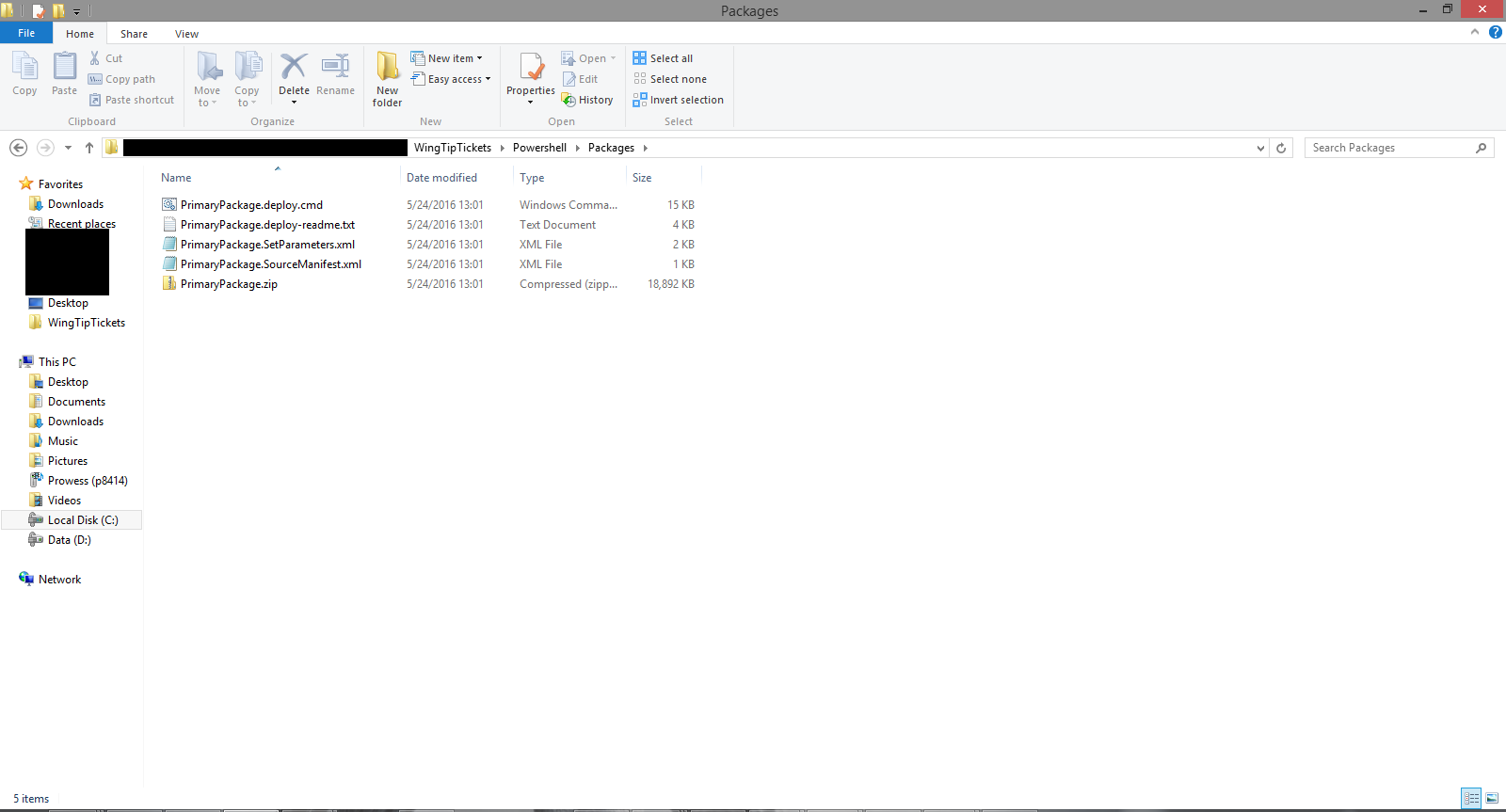
***Figure 16*** *Publish the WingTip Tickets Tenant.mvc web application*

1. Click **Publish**.



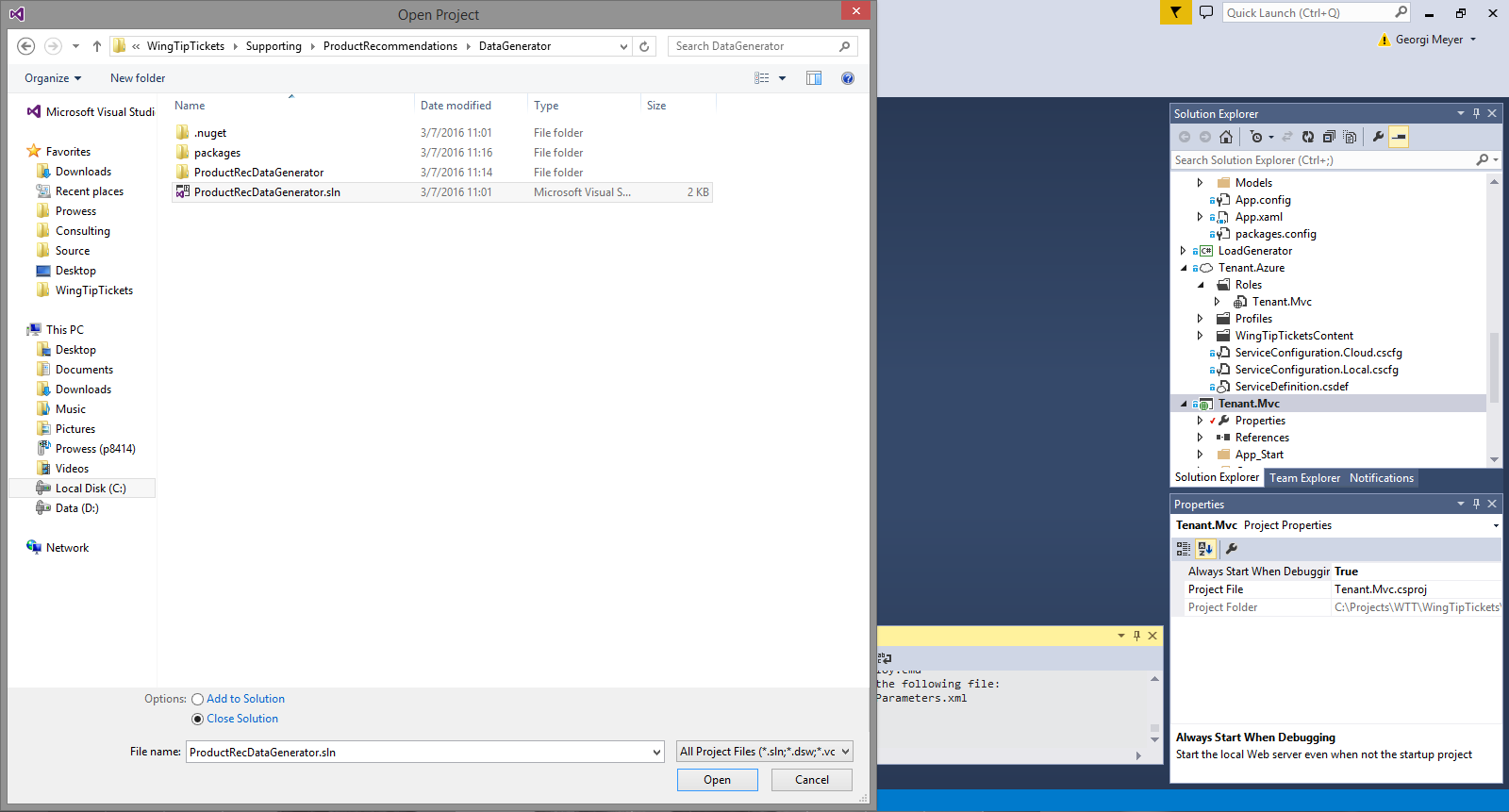
***Figure 17*** *Publish the WingTip Tickets Tenant.mvc web application*

1. Locate the published **PrimaryPackage.zip** file.



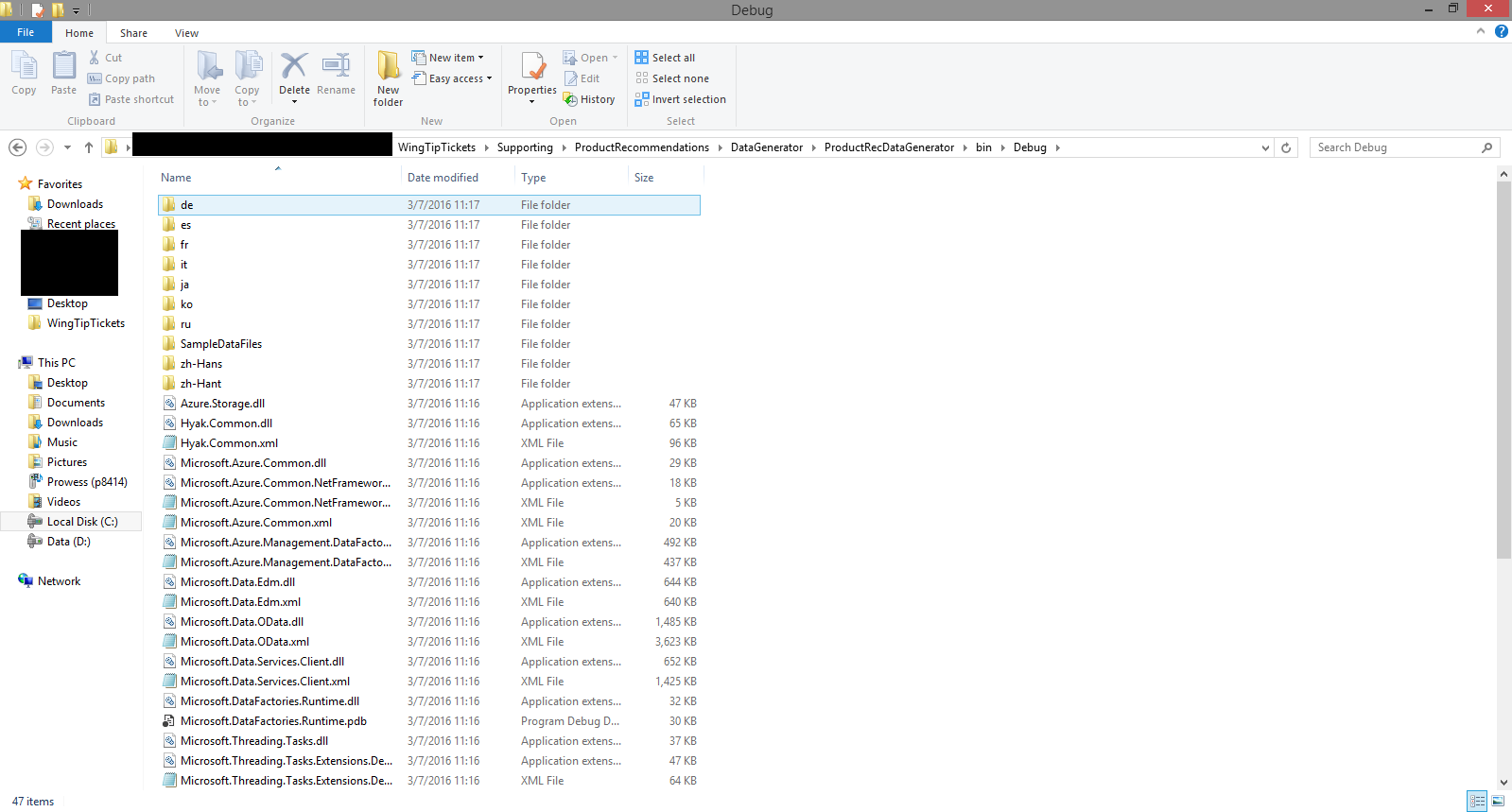
***Figure 18*** *Find the location of published zip file*

1. Copy the new file to **\PowerShell\Packages\**.
2. Rename the published file to **SecondaryPackage.zip**.
3. Copy the renamed **SecondaryPackage.zip** file to **\PowerShell\Packages\ProductRecDataGenerator**.
4. Launch Visual Studio 2013/2015.
5. Click **File** > **Open Project/Solution**.
6. Browse to **\Supporting\ProductRecommendations\DataGenerator**.
7. Select **ProductRecDataGenerator.sln**.



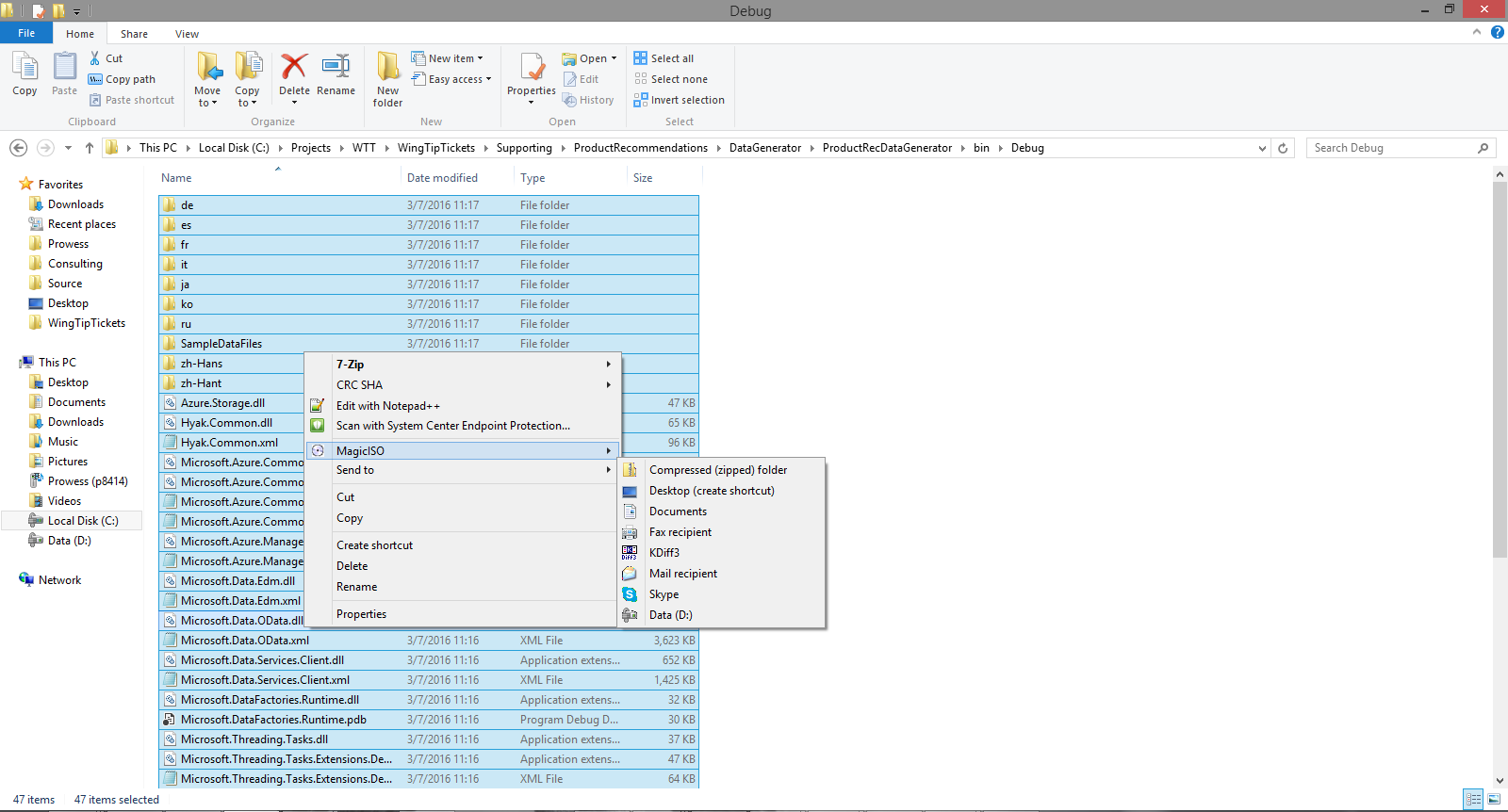
***Figure 19*** *Open the ProductRecDataGenerator.sln project solution*

1. Right-click **ProjectRecDataGenerator**, and then select **Build**.
2. Browse to the location of the build files.



***Figure 20*** *Find the location of the ProductRecDataGenerator build files*

1. Select all the files.
2. Right-click, select **Send To**, and then select the compressed (zipped) folder.



***Figure 21*** *Compress the ProductRecDataGenerator build files*

1. Rename the folder to **ProductRecDataGenerator.zip**.
2. Copy the zip file to **\PowerShell\Packages**.

# WingTip Tickets Services Deployed for Each PowerShell Script

Contents

[New-WTTEnvironment.ps1 17](#_Toc450891633)

[Get-WTTSqlDatabaseServerV12RegionAvailability.ps1 18](#_Toc450891634)

[Get-WTTAzureSqlDatabaseServerRegionCapabilities.ps1 19](#_Toc450891635)

[New-WTTAzureResourceGroup.ps1 19](#_Toc450891636)

[New-WTTAzureStorageAccount.ps1 19](#_Toc450891637)

[New-WTTAzureDocumentDb.ps1 20](#_Toc450891638)

[New-WTTAzureSqlDatabaseServer.ps1 20](#_Toc450891639)

[Deploy-DBSchema.ps1 21](#_Toc450891640)

[Populate-DBSchema.ps1 22](#_Toc450891641)

[Populate-Tickets.ps1 23](#_Toc450891642)

[New-WTTAzureSearchService.ps1 23](#_Toc450891643)

[Deploy-WTTWebApplication.ps1 25](#_Toc450891644)

[New-WTTAzureTrafficManagerProfile.ps1 25](#_Toc450891645)

[Add-WTTAzureTrafficManagerEndpoint.ps1 25](#_Toc450891646)

[Deploy-WTTAzureDWDatabase.ps1 26](#_Toc450891647)

[New-WTTADFEnvironment.ps1 27](#_Toc450891648)

[New-WTTPowerBI.ps1 28](#_Toc450891649)

[Set-WTTEnvironmentWebConfig.ps1 29](#_Toc450891650)

[Test-WTTAzureSQLConnection.ps1 30](#_Toc450891651)

New-WTTEnvironment.ps1  
Command:  
New-WTTEnvironment –WTTEnvironmentApplicationName <Environment Name> -WTTEnvironmentPrimaryServerLocation <Primary Server Location> -AdminUserName <Developer> -AdminPassword <P@ssword1> -AzureSqlServerVersion <12.0> -AzureSqlDatabaseName <Customer1> -AzureSqlDWDatabaseName <CustomerDW> -WebAppPackagePath <C:\wingtiptickets> -webAppPrimaryPackageName <primarypackage.zip> -webAppSecondaryPackageName <secondarypackage.zip> -tenantEventType <pop, rock, classical>   
What’s Deployed:  
This script is the main script that runs through and deploys all the services that are needed to run the WingTip Tickets demo.  
Parameters:

**Table 4** New-WTTEnvironment parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| –WTTEnvironmentApplicationName | datacamp*xx* *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -WTTEnvironmentPrimaryServerLocation | Datacenter location (such as West US, East US, West Europe, or East Europe) | Primary Azure SQL Server datacenter location |
| -adminUserName | Developer | Azure SQL Server administrator user name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |
| -AzureSqlServerVersion | v2.0 or v12.0  v12.0 is default | Default Azure SQL Server version |
| -AzureSqlDatabaseName | Customer1 | Azure SQL Database name |
| -AzureSqlDWDatabaseName | CustomerDW | Azure SQL Data Warehouse name |
| -WebAppPackagePath | Path to files | Path to where the PowerShell scripts and supporting files are stored |
| -webAppPrimaryPackageName | PrimaryPackage.zip | Name of the web-application primary package |
| -webAppSecondaryPackageName | SecondaryPackage.zip | Name of the web-application secondary package |
| -tenantEventType | Pop, rock, or classical | Web-application theme |

Get-WTTSqlDatabaseServerV12RegionAvailability.ps1  
Command:  
Get-WTTSqlDatabaseServerV12RegionAvailability –azureResourceGroupName $azureResourceGroupName

What’s Deployed:  
This script goes through the list of available datacenters and verifies if the datacenter is available for deployment. It also assists in automating the selection of a datacenter in the new-wttenvironment.ps1 script. This helps in that the end user does not have to provide the –WTTEnvironmentPrimaryServerLocation parameter. Both the primary and secondary Azure SQL Database locations are selected through this script.  
Parameters:

**Table 5** Get-WTTSQLDatabaseServerV12RegionAvailability parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| –azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | The resource group used throughout the PowerShell scripts to name the services that are deployed |

Get-WTTAzureSqlDatabaseServerRegionCapabilities.ps1  
Command:  
Get-WTTAzureSqlDatabaseServerRegionCapabilities

What’s Deployed:  
This script is used in the Get-WTTSqlDatabaseServerV12RegionAvailability.ps1 PowerShell script. It gets the list of locations that are available for SQL deployment and verifies that the location is available for deployment.  
Parameters:  
No parameters needed for this PowerShell script.

New-WTTAzureResourceGroup.ps1  
Command:  
New-WTTAzureResourceGroup -AzureResourceGroupName $azureResourceGroupName -AzureResourceGroupLocation $primaryServerLocation   
What’s Deployed:  
This script deploys the Azure resource group that all the Azure resources will be deployed to.  
Parameters:

**Table 6** New-WTTAzureResourceGroup parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -AzureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure resource Group |
| -AzureResourceGroupLocation | Global | Location of the Azure Resource Group, does not necessarily have to be in the same location as the rest of the resources. |

New-WTTAzureStorageAccount.ps1  
Command:  
New-WTTAzureStorageAccount -azureResourceGroupName $azureResourceGroupName -AzureStorageAccountName $azureStorageAccountName -AzureStorageAccountType "Standard\_GRS" -AzureStorageLocation $primaryServerLocation

What’s Deployed:  
This script deploys the Azure Storage Account that is used to deploy the web applications and Azure Data Factory services. It also hosts the Azure SQL Database audit logs.  
Parameters:

**Table 7** New-WTTAzureStorageAccount parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure resource group |
| -AzureStorageAccountName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure storage account |
| -AzureStorageAccountType | Standard\_GRS | Azure storage-account type; Standard\_GRS is default |
| -AzureStorageLocation | Global | Azure storage-account location; same location as the Azure SQL Database primary server |

New-WTTAzureDocumentDb.ps1  
Command:  
New-WTTAzureDocumentDb -azureResourceGroupName $azureResourceGroupName -WTTDocumentDbName $azureDocumentDbName -WTTDocumentDbLocation $WTTDocumentDbLocation

What’s Deployed:  
This script deploys the DocumentDB service used in the web application.  
Parameters:

**Table 8** New-WTTAzureDocumentDB parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure resource group |
| -WTTDocumentDbName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the DocumentDB service |
| -WTTDocumentDbLocation | Datacenter location (such as West US, East US, West Europe, or East Europe) | Location to deploy the DocumentDB service to |

New-WTTAzureSqlDatabaseServer.ps1  
Command:  
New-WTTAzureSqlDatabaseServer -azureSqlServerName $azureServerPrimaryName -azureSqlServerLocation $primaryServerLocation -adminUserName $adminUserName -adminPassword $adminPassword -AzureSqlServerVersion $AzureSqlDatabaseServerVersion –azureResourceGroupName $azureResourceGroupName  
What’s Deployed:

This script deploys the Azure SQL Server. Run it twice to set up a primary Azure SQL Server in the primary region and a secondary Azure SQL Server in the secondary region. The primary Azure SQL Server hosts the Customer1, Customer2, Customer3, Recommendations, and CustomerDW databases.  
Parameters:

**Table 9** New-WTTAzureSQLDatabaseServer parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureSqlServerName | datacamp*xx*primary  datacamp*xx*secondary *xx* is the datacamp number or deployment name | Azure SQL Server primary or secondary server names |
| -azureSqlServerLocation | Datacenter location (such as West US, East US, West Europe, or East Europe) | Azure SQL Server location for the primary or secondary server |
| -adminUserName | Developer | Azure SQL Server administrator name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |
| -AzureSqlServerVersion | 12.0 | Azure SQL Server version to deploy, v12.0 is default |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment | Name of the Azure resource group |

Deploy-DBSchema.ps1  
Command:  
Deploy-DBSchema -azureResourceGroupName $azureResourceGroupName

-azureSqlServerName $azureSqlServerPrimaryName -DatabaseEdition "Basic" -adminUserName $adminUserName -adminPassword $adminPassword -azureSqlDatabaseName $AzureSqlDatabaseName

What’s Deployed:  
Customer1, Customer2, and Customer3 database schema is deployed to the database.  
Parameters:  
  
***Table 10*** *Deploy-DBSchema parameters*

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -azureSqlServerName | datacamp*xx*primary *xx* is the datacamp number or deployment name | Azure SQL Server primary name |
| -DatabaseEdition | Basic,  Standard Premium | Azure SQL Database edition |
| -adminUserName | Developer | Azure SQL Server administrator name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |
| -azureSqlDatabaseName | Customer1 Customer2 Customer3 | Azure SQL Database name |

Populate-DBSchema.ps1  
Command:  
Populate-DBSchema -azureResourceGroupName $azureResourceGroupName -azureSqlServerName $azureSqlServerPrimaryName -adminUserName $adminUserName -adminPassword $adminPassword -AzureSqlDatabaseName $AzureSqlDatabaseName

What’s Deployed:

This script populates the Customer1, Customer2, and Customer3 databases with the content needed for the WingTip Tickets application.  
Parameters:

**Table 11** Populate-DBSchema parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureResourceGroupName | datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -azureSqlServerName | datacamp*xx*primary *xx* is the datacamp number or deployment name | Azure SQL Server primary name |
| -adminUserName | Developer | Azure SQL Server administrator name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |
| -AzureSqlDatabaseName | Customer1, Customer2, or Customer3 | Azure SQL Database containing content for the WingTip Tickets application |

Populate-Tickets.ps1  
Command:  
Populate-Tickets -azureResourceGroupName $azureResourceGroupName -adminUserName $adminUserName -adminPassword $adminPassword -AzureSqlDatabaseName $AzureSqlDatabaseName -AzureSqlServerName $azureSqlServerPrimaryName

What’s Deployed:  
This script populates the tickets table with data to display tickets purchased in the Seat Chart Map shown in the venue.  
Parameters:

**Table 12** Populate-Tickets parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -AzureSqlServerName | datacamp*xx* *xx* is the datacamp number or deployment name | Azure SQL Server primary name |
| -adminUserName | Developer | Azure SQL Server administrator name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |
| -AzureSqlDatabaseName | Customer1 | Azure SQL Database containing content for the WingTip Tickets application |

New-WTTAzureSearchService.ps1  
Command:  
new-wttazuresearchservice -wttenvironmentapplicationname $wttenvironmentapplicationname -azureResourceGroupName $azureResourceGroupName -azuresearchservicelocation $primaryServerLocation -AzureSqlServerName $azureSqlServerPrimaryName -adminUserName $adminUserName -adminPassword $adminPassword -AzureSqlDatabaseName $AzureSqlDatabaseName

What’s Deployed:  
This script deploys the Azure Search service. It checks if there are any free search services currently in the Azure subscription and, if any are found, it deploys a standard Azure Search service. This PowerShell script also uses a series of REST commands to set up the index on the Customer1 database.  
Parameters:

**Table 13** New-WTTAzureSearchService parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -wttenvironmentapplicationname | datacamp*xx* *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure resource group |
| -azuresearchservicelocation | Datacenter location (such as West US, East US, West Europe, East Europe) | Azure Search service location |
| -AzureSqlServerName | datacamp*xx* *xx* is the datacamp number or deployment name | Azure SQL Server primary name |
| -adminUserName | Developer | Azure SQL Server administrator name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |
| -AzureSqlDatabaseName | Customer1 | Name of the Azure SQL Database used in the Azure Search service |

Deploy-WTTWebApplication.ps1  
Command:  
Deploy-WTTWebApplication -azureStorageAccountName $azureStorageAccountName -azureResourceGroupName $azureResourceGroupName -Websitename $azureSqlServerPrimaryName -WebAppPackagePath $WebAppPackagePath -webAppPackageName $webAppPrimaryPackageName

What’s Deployed:  
This script deploys the web-application package to the web application that is created through the new-wttenvironment.ps1 PowerShell script.  
Parameters:

**Table 14** Deploy-WTTWebApplication parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure resource group |
| -Websitename | datacamp*xx*primary  datacamp*xx*secondary *xx* is the datacamp number or deployment name | Name of the web app, either primary or secondary |
| -WebAppPackagePath | Path to files | Path to where the PowerShell scripts and supporting files are stored |
| -webAppPackageName | PrimaryPackage.zip or SecondaryPackage.zip | Name of the web-application package |

New-WTTAzureTrafficManagerProfile.ps1  
Command  
New-WTTAzureTrafficManagerProfile -AzureTrafficManagerProfileName $wTTEnvironmentApplicationName -AzureResourceGroupName $azureResourceGroupName

What’s Deployed:  
This script deploys Azure Traffic Manager to show fault tolerance with Azure web applications.  
Parameters:

**Table 15** New-WTTAzureTrafficManagerProfile parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -AzureTrafficManagerProfileName | datacamp*xx* *xx* is the datacamp number or deployment name | Azure Traffic Manager name |
| -AzureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure resource group |

Add-WTTAzureTrafficManagerEndpoint.ps1  
Command:  
Add-WTTAzureTrafficManagerEndpoint -AzureTrafficManagerProfileName $wTTEnvironmentApplicationName -azurePrimaryWebAppName $azureSqlServerPrimaryName -azureSecondaryWebAppName $azureSqlServerSecondaryName -AzureTrafficManagerEndpointStatus "Enabled" -AzureResourceGroupName $azureResourceGroupName

What’s Deployed:  
Adding the web applications to Azure Traffic Manager.  
Parameters:

**Table 16** New-WTTAzureTrafficManagerEndpoint parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -AzureTrafficManagerProfileName | datacamp*xx* *xx* is the datacamp number or deployment name | Azure Traffic Manager name |
| -azurePrimaryWebAppName | datacamp*xx*primary *xx* is the datacamp number or deployment name | Primary web-app name |
| -azureSecondaryWebAppName | datacamp*xx*secondary *xx* is the datacamp number or deployment name | Secondary web-app name |
| -AzureTrafficManagerEndpointStatus | Enabled or Disabled | Status of the web application in Azure Traffic Manager |
| -AzureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | Azure Traffic Manager resource-group name; usually the same as -WTTEnvironmentApplicationName |

Deploy-WTTAzureDWDatabase.ps1  
Command:  
Deploy-WTTAzureDWDatabase -azureResourceGroupName $azureResourceGroupName -azureSqlServerName $azureSqlServerPrimaryName -DatabaseEdition "DataWarehouse" -adminUserName $adminUserName -adminPassword $adminPassword -azureDWDatabaseName $AzureSqlDWDatabaseName

What’s Deployed:  
Azure SQL Data Warehouse with CustomerDW data-warehouse database. This database is also populated with the schema and data needed for the HOL. The data-warehouse data is loaded via Polybase from a preconfigured storage account.  
Parameters:

**Table 17** Deploy-WTTAzureDWDatabase parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -azureSqlServerName | datacamp*xx*primary *xx* is the datacamp number or deployment name | Azure SQL Server primary name |
| -DatabaseEdition | DataWarehouse | Azure SQL Database Data Warehouse edition parameter; this specifies that we are deploying an Azure SQL Data Warehouse database and not a normal Azure SQL Database |
| -adminUserName | Developer | Azure SQL Server administrator name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |
| -azureDWDatabaseName | CustomerDW | Name of the Azure SQL Data Warehouse database |

New-WTTADFEnvironment.ps1  
Command:  
New-WTTADFEnvironment -ApplicationName $WTTEnvironmentApplicationName -azureResourceGroupName $azureResourceGroupName -azureSqlServerName $azureSqlServerPrimaryName -azureSQLDatabaseName "Recommendations" -DatabaseEdition "Basic" -adminUserName $adminUserName -adminPassword $adminPassword

What’s Deployed:  
This script deploys Azure Data Factory, the recommendations database for Azure Data Factory, the necessary Azure storage account blob storage, and the datasets, pipelines, and linked services for Azure Data Factory data processing.  
Parameters:

**Table 18** New-WTTADFEnvironment parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -ApplicationName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure Data Factory environment |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure resource group |
| -azureSqlServerName | datacamp*xx*primary *xx* is the datacamp number or deployment name | Azure SQL Server primary name |
| -azureSQLDatabaseName | Recommendations | Azure SQL Database name to process Azure Data Factory |
| -DatabaseEdition | Basic | Azure SQL Database edition |
| -adminUserName | Developer | Azure SQL Server administrator name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |

New-WTTPowerBI.ps1  
Command:  
New-WTTPowerBI -azureResourceGroupName $azureResourceGroupName -AzurePowerBIName $azurePowerBIWorkspaceCollection -azurePowerBILocation $azurePowerBILocation -AzureSqlServerName $azureSqlServerPrimaryName -adminUserName $adminUserName -adminPassword $adminPassword -AzureSqlDatabaseName $AzureSqlDatabaseName -azureDWDatabaseName $AzureSqlDWDatabaseName

What’s Deployed:  
This script deploys the Azure Power BI Embedded service, including the workspace. This script also uploads the created reports view in the web application and sets the data source to the correct Azure SQL Server.  
Parameters:

**Table 19** New-WTTPowerBI parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -AzurePowerBIName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Power BI workspace collection service |
| -azurePowerBILocation | Datacenter location (such as West US, East US, West Europe, East Europe) | Azure Power BI Embedded service location |
| -AzureSqlServerName | datacamp*xx*primary *xx* is the datacamp number or deployment name | Azure SQL Server primary name |
| -adminUserName | Developer | Azure SQL Server administrator name |
| -adminPassword | P@ssword1 | Azure SQL Server administrator password |
| -AzureSqlDatabaseName | Customer1 | Azure SQL Database containing content for the WingTip Tickets application |
| -azureDWDatabaseName | CustomerDW | Azure SQL Data Warehouse database containing content for the WingTip Tickets application; this data does not directly reflect any ticket purchases or customers from the Customer1 database |

Set-WTTEnvironmentWebConfig.ps1  
Command:  
Set-WTTEnvironmentWebConfig -WTTEnvironmentApplicationName $wTTEnvironmentApplicationName -azureResourceGroupName $azureResourceGroupName -Websitename $azureSqlServerPrimaryName -SearchName $searchName -SearchServicePrimaryManagementKey $searchServicePrimaryManagementKey -AzureSqlServerPrimaryName $azureSqlServerPrimaryName -AzureSqlServerSecondaryName $azureSqlServerSecondaryName -azureDocumentDbName $azureDocumentDbName -documentDbPrimaryKey $documentDbPrimaryKey -powerbiSigningKey $powerbiSigningKey -powerbiWorkspaceCollection $powerbiWorkspaceCollection -powerbiWorkspaceId $powerbiWorkspaceId -seatMapReportID $seatMapReportID -TenantEventType $TenantEventType

What’s Deployed:  
This script sets the app settings in the deployed web applications.  
Parameters:

**Table 20** New-WTTEnvironmentWebConfig parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -WTTEnvironmentApplicationName | datacamp*xx* *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -azureResourceGroupName | datacamp*xx* *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -Websitename | datacamp*xx*primary  datacamp*xx*secondary *xx* is the datacamp number or deployment name | Name of the web app, either primary or secondary |
| -SearchName | datacamp*xx* *xx* is the datacamp number or deployment name | Name of the Azure Search service |
| -SearchServicePrimaryManagementKey | xxxx-1234-xxxx-1234-xxxx | Azure Search Service primary access key |
| -AzureSqlServerPrimaryName | datacamp*xx*primary *xx* is the datacamp number or deployment name | Primary Azure SQL Server name |
| -AzureSqlServerSecondaryName | datacamp*xx*secondary *xx* is the datacamp number or deployment name | Secondary Azure SQL Server name |
| -azureDocumentDbName | datacamp*xx* *xx* is the datacamp number or deployment name | DocumentDB name |
| -documentDbPrimaryKey | xxxx-1234-xxxx-1234-xxxx | DocumentDB primary access key |
| -powerbiSigningKey | xxxx-1234-xxxx-1234-xxxx | Azure Power BI Embedded service primary access key |
| -powerbiWorkspaceCollection | datacamp*xx* *xx* is the datacamp number or deployment name | Azure Power BI Embedded service workspace collection name |
| -powerbiWorkspaceId | xxxx-1234-xxxx-1234-xxxx | Azure Power BI Embedded service workspace ID |
| -seatMapReportID | xxxx-1234-xxxx-1234-xxxx | Azure Power BI Embedded service seat-mat report ID |

Test-WTTAzureSQLConnection.ps1  
Command:  
Test-WTTAzureSQLConnection -ServerName $ServerName -UserName $UserName -Password $Password -DatabaseName $DatabaseName -WTTEnvironmentApplicationName $WTTEnvironmentApplicationName

What’s Deployed:  
This script tests that the Azure SQL Server is available and accessible.  
Parameters:

**Table 21** Test-WTTAzureSQLConnection parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Description |
| -ServerName | datacamp*xx*primary *xx* is the datacamp number or deployment name | Primary Azure SQL Server name |
| -UserName | Developer | Azure SQL Server administrator name |
| -Password | P@ssword1 | Azure SQL Server administrator password |
| -DatabaseName | Customer1 | Azure SQL Database containing content for the WingTip Tickets application |
| -WTTEnvironmentApplicationName | datacampxx *xx* is the datacamp number or deployment name | The name used throughout the PowerShell scripts to name the services that are deployed |
| -TenantEventType | Pop, rock, classical | Web-application theme |