Basic Mobile App with Validation

Create a .NET backend using the Azure portal

You can create a new mobile application right in the [Azure portal](https://portal.azure.com/). You can either follow the steps below, or create a new client and server together by following the [Create a mobile app](https://azure.microsoft.com/en-us/documentation/articles/app-service-mobile-ios-get-started/) tutorial.

1. Log into the [Azure Portal](https://portal.azure.com/).
2. In the top left of the window, click the **+NEW** button > **Web + Mobile** > **Mobile App**, then provide a name for your Mobile App backend.
3. In the **Resource Group** box, select an existing resource group. If you have no resource groups, enter the same name as your app.

At this point, the default App Service plan is selected, which is in the Free tier. The App Service plan settings determine the location, features, cost and compute resources associated with your app. You can either select another App Service plan or create a new one. For more about App Services plans and how to create a new plan, see [Azure App Service plans in-depth overview](https://azure.microsoft.com/en-us/documentation/articles/azure-web-sites-web-hosting-plans-in-depth-overview/)

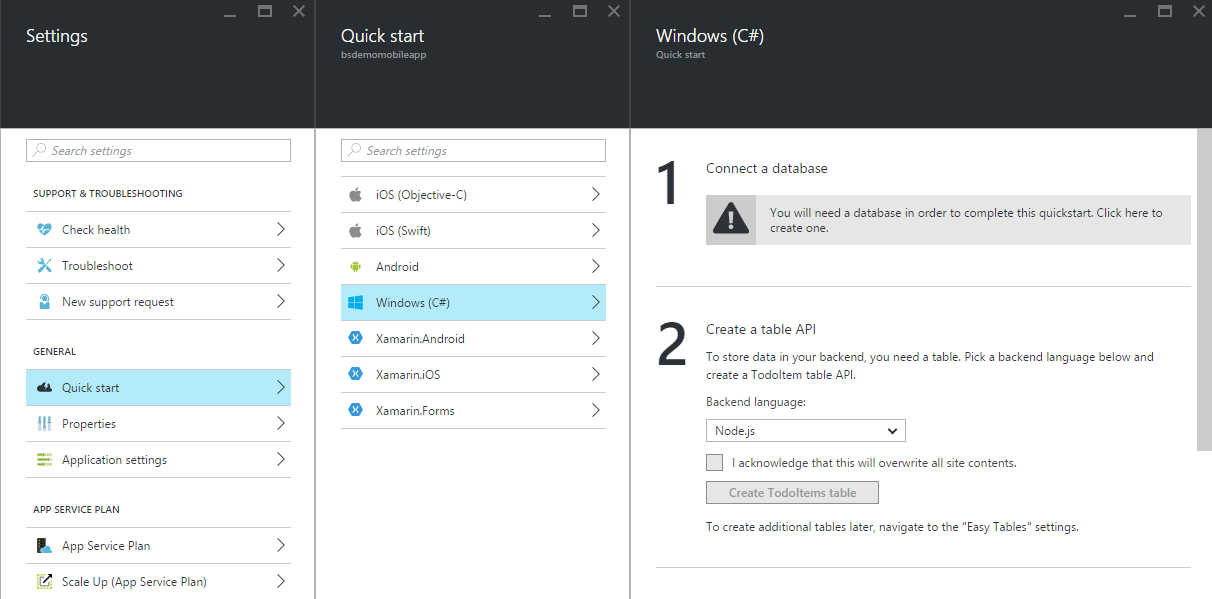
1. Use the default App Service plan, select a different plan or [create a new plan](https://azure.microsoft.com/en-us/documentation/articles/azure-web-sites-web-hosting-plans-in-depth-overview/#create-an-app-service-plan), then click **Create**.

This creates the Mobile App backend. Later you will deploy your server project to this backend. Provisioning a Mobile App backend can take several minutes; the **Settings** blade for the Mobile App backend is displayed when complete. Before you can use the Mobile App backend, you must also define a connection a data store.

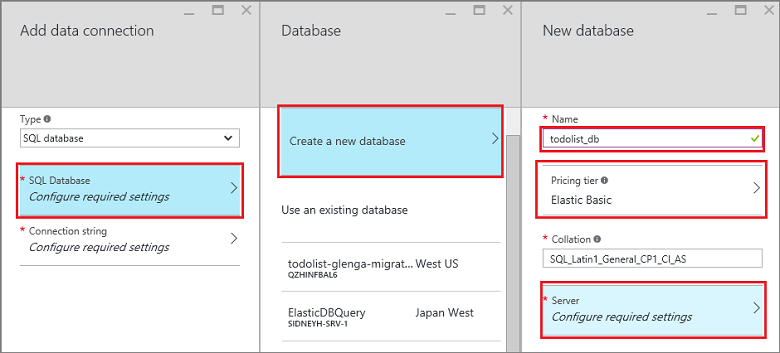
NOTE:

As part of this tutorial, you create a new SQL Database instance and server. You can reuse this new database and administer it as you would any other SQL Database instance. If you already have a database in the same location as the new mobile app backend, you can instead choose **Use an existing database** and then select that database. The use of a database in a different location is not recommended because of additional bandwidth costs and higher latencies. Other data storage options are available.

1. In the **Settings** blade for the new Mobile App backend, click **Quick start** > your client app platform > **Connect a database**.



1. In the **Add data connection** blade, click **SQL Database** > **Create a new database**, type the database **Name**, choose a pricing tier, then click **Server**.



1. In the **New server** blade, type a unique server name in the **Server name** field, provide a secure **Server admin login** and **Password**, make sure that **Allow azure services to access server** is checked, then click **OK** twice. This creates the new database and server.
2. Back in the **Add data connection** blade, click **Connection string**, type the login and password values for your database, then click **OK** twice.

Creation of the database can take a few minutes. Use the **Notifications** area to monitor the progress of the deployment. You cannot continue until the database has been deployed successfully.

1. Back in the Quick Start blade, under **Create a table API**, choose **C#** as your **Backend language**
2. Click Download, extract the compressed project files to your local computer, and open the solution in Visual Studio.

Configure the server project

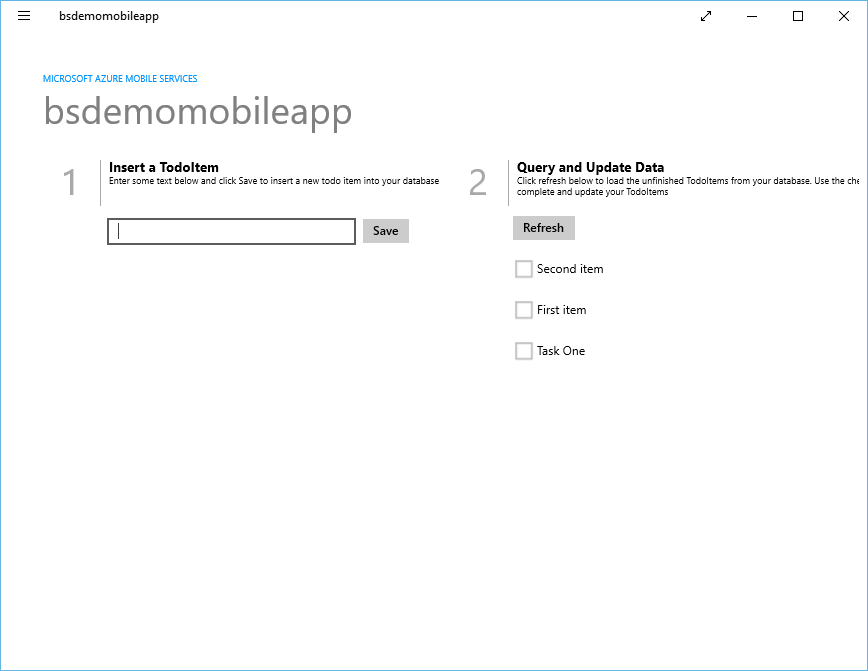
1. Back in the Mobile App backend settings, click **Quick Start** > your client platform.
2. Under **Create a table API**, select **C#** as your **Backend language**,
3. Click **Download**, extract the compressed project files to your local computer, open the solution in Visual Studio, build the project to restore the NuGet packages, then deploy the project to Azure. To learn how to deploy a .NET backend server project to Azure, see [How to: Publish the server project](https://azure.microsoft.com/en-us/documentation/articles/app-service-mobile-dotnet-backend-how-to-use-server-sdk/#publish-server-project) in the .NET backend SDK topic.

You Mobile App backend is now ready to use with your client app.

Download and run the client project

Once you have configured your Mobile App backend, you can either create a new client app or modify an existing app to connect to Azure. In this section, you download a universal Windows app template project that is customized to connect to your Mobile App backend.

1. Back in the **Quick Start** blade for your Mobile App backend, click **Create a new app** > **Download**, then extract the compressed project files to your local computer.
2. (Optional) Add the universal Windows app project to the solution with the server project. This makes it easier to debug and test both the app and the backend in the same Visual Studio solution, if you choose to do so.
3. With the Windows Store app as the startup project, press the F5 key to rebuild the project and start the Windows Store app.
4. In the app, type meaningful text, such as *Complete the tutorial*, in the **Insert a TodoItem** text box, and then click **Save**.



This sends a POST request to the new mobile app backend that's hosted in Azure.

1. Stop debugging, right-click the <your app name>.WindowsPhone project, click **Set as StartUp Project**, and then press F5 again.

Notice that data saved from the previous step is loaded from the mobile app after the Windows app starts.

Add Server Side Validation

1. Switch to Visual Studio solution that contains the mobile service project.
2. In the Solution Explorer window expand the todo list service project and expand **Controllers**. Open the *TodoItemController.cs* file which is part of the mobile service project.
3. Replace the PostTodoItem method with the following method which will validate that the text string is not greater than 10 characters. For items that do have a text length greater than 10 characters, the method returns an HTTP Status code 400 Bad Request with a descriptive message included as content.

public async Task<IHttpActionResult> PostTodoItem(TodoItem item)

{

if (item.Text.Length < 5)

{

return BadRequest("The Item's Text length must be greater than 5.");

}

else

{

TodoItem current = await InsertAsync(item);

return CreatedAtRoute("Tables", new { id = current.Id }, current);

}

}

1. Right click the service project and click **Build** to build the mobile service project. Verify no errors occurred.
2. Right click the service project and click **Publish**.

Update the client

Now that the mobile service is setup to validate data and send error responses for an invalid text length, you need to update your app to be able to handle error responses from validation. The error will be caught as a *MobileServiceInvalidOperationException* from the client app's call to *IMobileServiceTable<TodoItem].InsertAsync().*

1. In the Solution Explorer window in Visual Studio, navigate to the client project and open the MainPage.xaml.cs file. Add the following **using** statement in that file:

using Windows.UI.Popups;

using Newtonsoft.Json.Linq;

1. In MainPage.xaml.cs replace the existing **InsertTodoItem** method with the following code:

private async void InsertTodoItem(TodoItem todoItem)

{

// This code inserts a new TodoItem into the database. When the operation completes

// and Mobile Services has assigned an Id, the item is added to the CollectionView

MobileServiceInvalidOperationException invalidOpException = null;

try

{

await todoTable.InsertAsync(todoItem);

items.Add(todoItem);

}

catch(MobileServiceInvalidOperationException e)

{

invalidOpException = e;

}

if (invalidOpException != null)

{

string strJsonContent = await invalidOpException.Response.Content.ReadAsStringAsync();

var responseContent = JObject.Parse(strJsonContent);

MessageDialog errormsg = new MessageDialog(string.Format("{0} (HTTP {1})",

(string)responseContent["message"],(int)invalidOpException.Response.StatusCode),

invalidOpException.Message);

var ignoreAsyncOpResult = errormsg.ShowAsync();

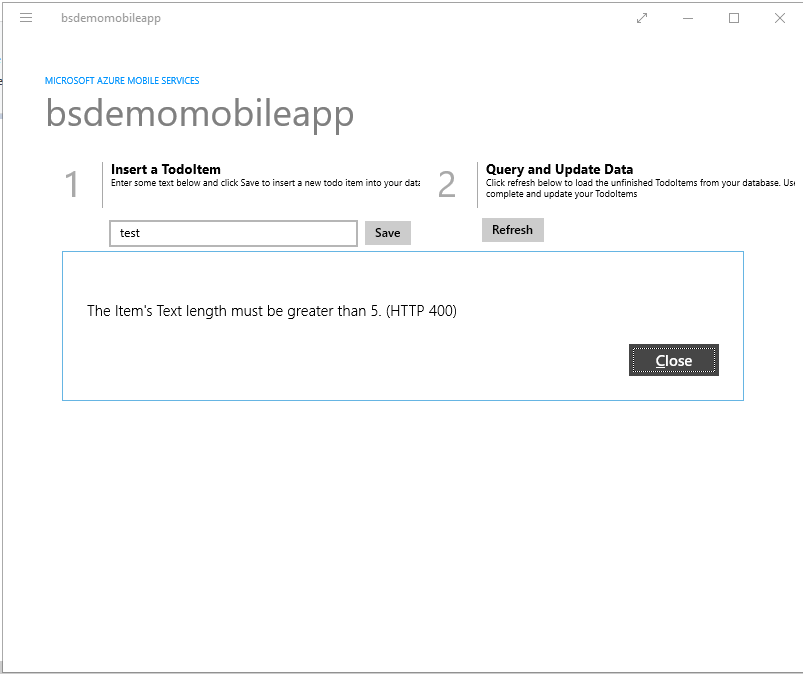
}

}

This version of the method includes error handling for the **MobileServiceInvalidOperationException** that displays the deserialized error message from the response content in a message dialog.

Test Length Validation

1. In Solution Explorer in Visual Studio, right click the client app project and then click **Set as StartUp Project**. Then press the **F5** key to start the app hosting the service locally in IIS Express.
2. Enter the text for a new todo item with a length less than 5 characters and then click **Save**.
3. You will get a message dialog similar to the following in response to the invalid text.



End of the document.