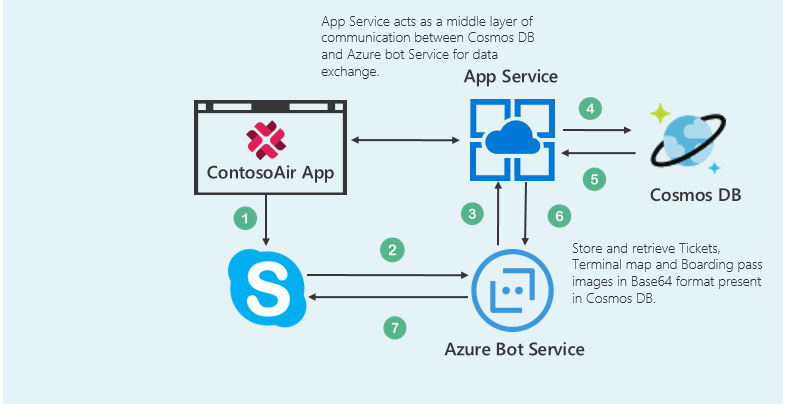
# Azure Bot Service using Skype Bot

## User Story

Consider the user scenario, Kevin is driving towards the airport and due to some reasons, he would get late to catch up his flight. He wants immediate help regarding **flight** **schedule**, **alternate** **flight** **options** and **E** **check-in** to get **boarding** **pass**. He also need **airport** **terminal** **map**, so that he can save his time to find out the route of boarding gate.

Using **ContosoAir** demo application you will come across **Azure Bot Service**, **Cosmos DB** database to get flight assistance. Azure Bot Service creates an intelligent Bot that interacts with user naturally wherever they are.

## Components Used in This Scenario



* The **Azure Bot Service** provides an integrated environment that is purpose-built for bot development, enabling you to build, connect, test, deploy, and manage intelligent bots, all from one place. You can write your bot in C# or Node.js directly in the browser using the Azure editor. [(Azure Bot Service)](https://docs.microsoft.com/en-us/bot-framework/azure-bot-service-quickstart)
* **Azure Cosmos DB** is Microsoft's globally distributed, multi-model database. With the click of a button, Azure Cosmos DB enables you to elastically and independently scale throughput and storage across any number of Azure's geographic regions. It offers throughput, latency, availability, and consistency guarantees with comprehensive [service level agreements](https://aka.ms/acdbsla) (SLAs), something no other database service can offer. ([Azure Cosmos DB](https://docs.microsoft.com/en-us/azure/cosmos-db/))

## What you will learn from this lab

* Creating a skype Bot using **Azure** **Bot Service**.
* Create collections/insert data in **Cosmos** **DB** using **Data** **Migration** **Tool**.
* Playing around code in **Visual Studio 2017** IDE to add some options in the menu list in skype bot.

***Ready? Let’s get started!***

## Scenario 1 - Creating Azure Bot

*We’ll start with accessing* ***Azure Portal.*** *For creating resources, you need to* ***Sign in*** *to* ***Azure Portal.*** *The steps to do the same are given below:*

1. Click on the link [www.portal.azure.com](www.portal.azure.com%20) to open Azure Portal and maximize the browser window.
2. In the **Email** or **Phone** field, enter the given subscribed user Email id or phone number “**user208240@cloudplatimmersionlabs.onmicrosoft.com”**.
3. In the **Password** field, enter the given password **“JuXe9g:\*4jk[”**.
4. Click on **Sign in** button.
5. You may encounter a popup entitled **Welcome to Microsoft Azure** with options to **Start Tour** and **Maybe Later** – **Choose Maybe Later**. [Ignore the step no. 5 if didn’t get the popup message.]

Great! You are now logged in to the Azure Portal.

1. Click on the **Resource groups** option present under icon  and click on the Resource group named as “**BotResourceGrp**” under **Resource groups blade**.
2. Now click on  present in right side panel with **Type** as App Service and **Location** as **West** **US**.
3. Select **C#** from **choose a template** option then click on **Basic** template and click on **Next** button.
4. Click on **Create Microsoft App ID and password** button.  It will redirect you to another tab for generating App ID and password.
5. In the **Email** or **Phone** field, enter the given subscribed user Email id or phone number -> “**user208240@cloudplatimmersionlabs.onmicrosoft.com”**.
6. In the **Password** field, enter the given password -> **“JuXe9g:\*4jk[”**.
7. Click on **Sign in** button to register the bot on **Application** **Registration** **Portal**.
8. Copy **App** **ID** and paste it into the **Notepad** file named as **Credentials** present on the **Desktop** and save the changes.
9. Now again come back to the browser and click on  button to generate the App Password. It will generate the **pop-up** window containing **App** **password**.
10. Copy generated password into the **Notepad** file named as **Credentials** present on the **Desktop** and save the file. Again, come back on the browser and click on **Ok** button.
11. Click on button.  and select **both** the checkboxes  for accepting T**erms** and **Privacy** **Conditions**.
12. Finally click on **Create bot** button to deploy the **Bot** on Azure.

**Note:** Please wait, it takes some time to deploy the Bot.

1. After successful completion of the deployment process you will be redirected to the **BUILD** tab.
2. Click on **Download zip file** button present under **Download source code** section to download the source code of created Bot.
3. To test your Bot, click on the **Test** button provided on right side of the page. 
4. Type your text message in the provided text field and press **Enter** button. The Bot will reply you intelligently.

*Nice work! You have successfully created Azure Bot.*

## Scenario 2 - Creating Cosmos DB collection

### Part A - Adding collection into Cosmos DB

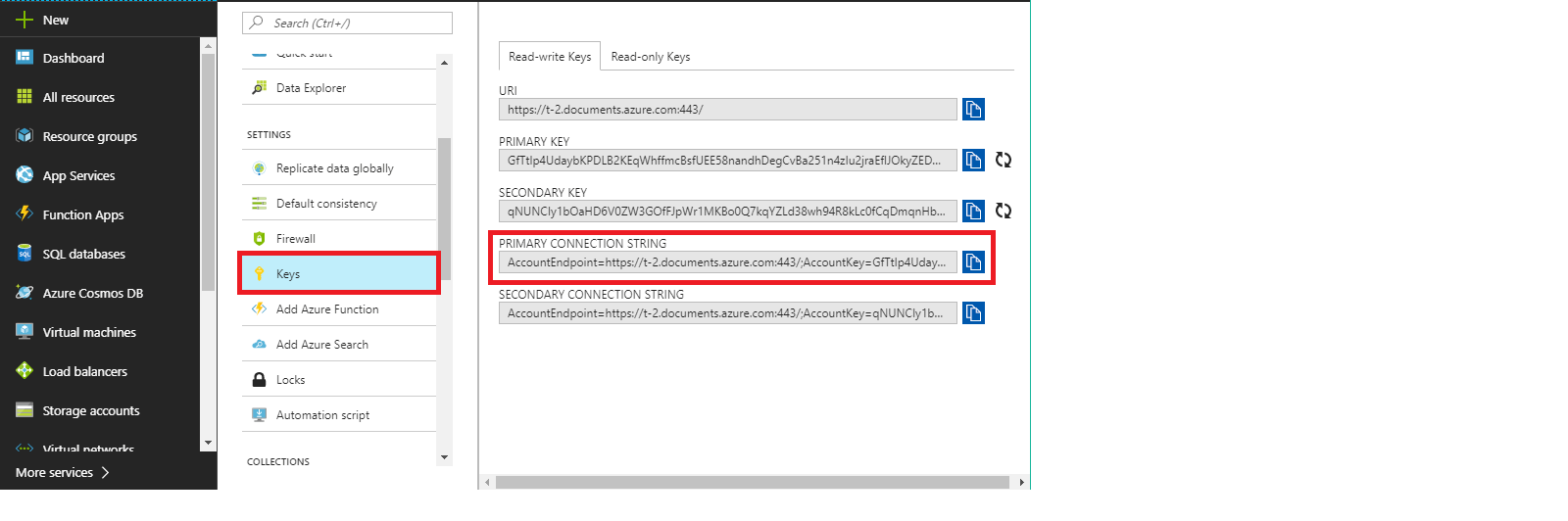
1. Switch to Azure Portal by clicking on the link [poral.azure.com](http://portal.azure.com) as launched in **Scenario 1**.
2. Click on the **Resource groups** option present under  icon and click on the Resource group named as “**BotResourceGrp**” under **Resource groups blade**.
3. Now click on  present in right side panel with **Type** as **Azure Cosmos DB account**.
4. Click on option present on the menu bar.
5. Enter Collection Id as **Bookings** in the Collection Id field present under Add Collection blade. Select the STORAGE CAPACITY as **Fixed** and enter DATABASE as **BotDB** in the DATABASE field and finally click on **OK** button.

### Part B - Inserting data in Cosmos DB using Data Migration Tool

*But, what you will do with Cosmos DB without data? So, let’s insert some data into Cosmos DB using Microsoft’s Data Migration Tool.*

For more details about the migration tool, click on the given link: [Data Migration tool.](https://docs.microsoft.com/en-us/azure/cosmos-db/import-data)

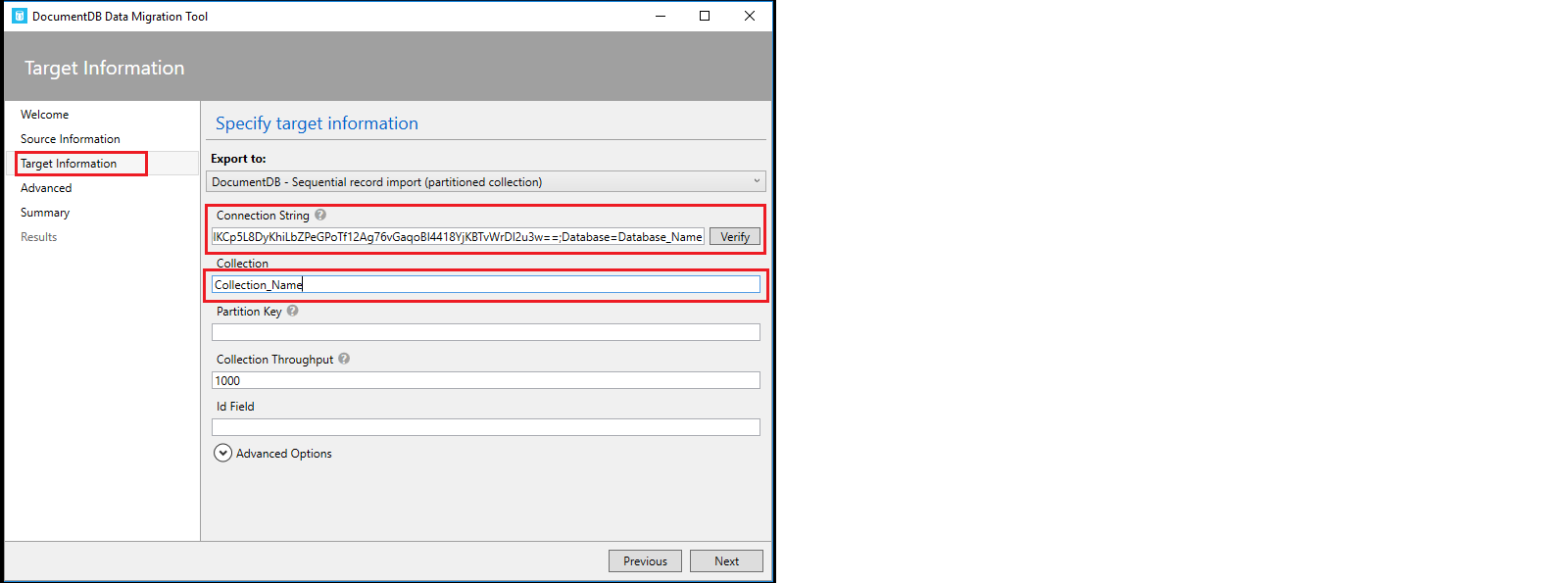
1. Launch the **Data Migration Tool** by double clicking on **dtui.exe** file present on **Desktop**.
2. After the tool is launched, click on **Source Information** menupresent on the left panel**.**
3. Select **JSON file(s)** option from **Import from** dropdown.
4. Click on **Add Files** button.
5. Select the JSON file “**Flights.json**” present at the location **“C:\Json\flights.json”** and click **Open** button.
6. Then click **Next button** to redirect to **Target information** menu.
7. Now, switch to **Azure Portal** in which you are already logged in from **Scenario 1**.
8. Navigate to Azure Portal’s **Resource Group** option present in the favourites menu on the left side panel and select your created Resource “**SoloServiceResourceGrp**” and click on Azure Cosmos DB Account “**SoloServiceCosmosDB**”.
9. Go to **Keys** option under **Azure Cosmos DB account blade** and **copy** the **Primary Connection String**
10. **AccountEndpoint=https://t-2.documents.azure.com:443/;AccountKey=GfTtIp4UdaybKPDLB2KEqWhffmcBsfUEE58nandhDegCvBa251n4zIu2jraEflJOkyZEDwrtxPmKOOftRDqo2Q==;** present on right side of the blade.



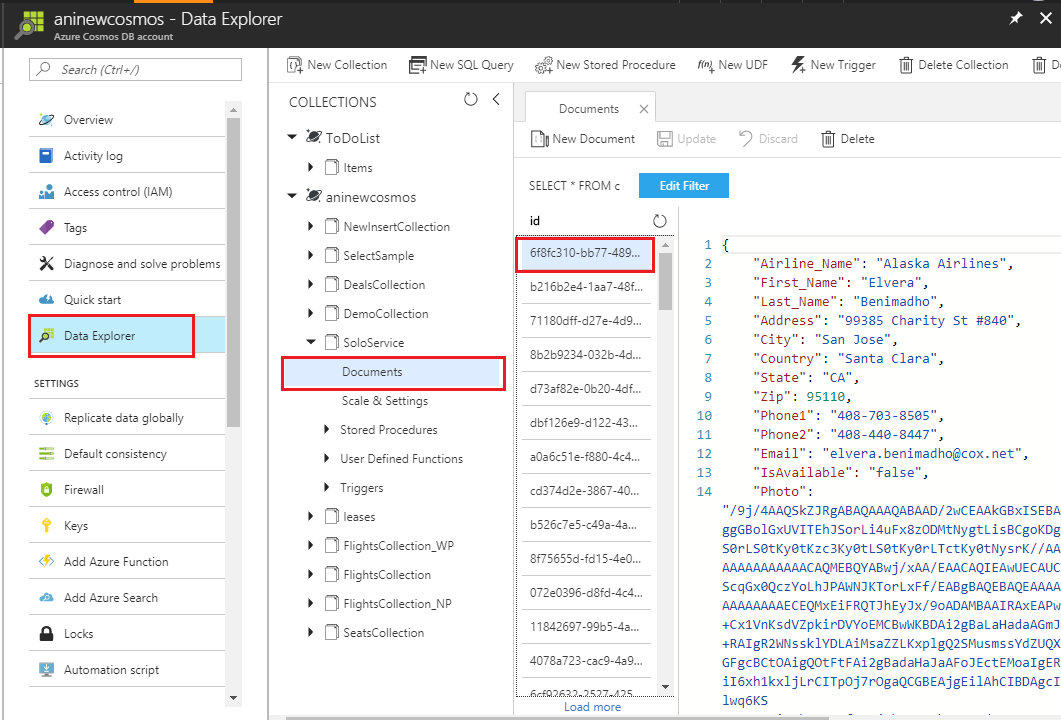
1. Switch back to **Data Migration** **Tool** already launched in step 1.
2. Paste the copied **Primary Connection String** in **Connection String** text box and appended with **Database Name** at the end of the string in **Target information** option.

**For e.g.**

“AccountEndpoint=https://t-2.documents.azure.com:443/;AccountKey=your\_primary\_key==;***Database=*Your Cosmos DB Account Name**”.



1. Once you are done with the Step 11, click on **Verify** button next to **Connection String** field to verify the connection string.
2. On successful verification of Connection string, success pop-up window will display, then click on **OK** button.
3. On verification failure of connection string, a pop-up window will appear prompting the **failure message**, then click on **Ok** button and again verify the connection string till successful verification.
4. Enter the **Collection Name** same as the name of the **Json** file mentioned in **step 5** in the Collection field.
5. Click on **Next -> Next ->** **Import** to import the data present in Json file into **Cosmos DB collection.**
6. This will start the data uploading process. Wait for some time to complete the process and once the process is completed successfully close the **migration tool**.
7. Now, switch to **Azure Portal** in which you are already logged in from **Scenario 1**.
8. Click on Azure Portal’s **Resource Group** option present in the favourites blade in the left side panel and click on **“BotResourceGrp”.**
9. Click on **“BotCosmosDB”** which is your **Cosmos DB Account.**
10. Then click on **Data** Explorer present under Azure Cosmos DB account blade to view the created **collections** which will be fetched from the **Json file** imported through **Data** **Migration** **Tool**.
11. Then click on created collection **“Flights”** to expand. Then, click on **Documents** option under the collection to view the data imported through **Migration tool** into **Cosmos DB.**



1. Repeat the steps **1-18** to import **alternateflights.json** and **bookings.json** files present at the location **“C:\Source\Json”** into Cosmos DB.

*Nice work! You have successfully inserted data in your Cosmos DB.*

1. Click on Keys option present under Azure Cosmos DB account blade in the left panel.



26. Copy the **URI**, **PRIMARY** **KEY** and **PRIMARY** **CONNECTION** **STRING** values and paste it into the Notepad file named as **Credentials** present on **Desktop** and save the file.

## Scenario 3 – Playing around the Visual Studio code

### Part A - Importing the projects into Visual Studio 2017 IDE

1. Go to **Downloads** section. Right click on the downloaded file and click on option **Extract to** for extracting the zip file into the folder.
2. Double click on **Microsoft.Bot.Sample.SimpleEchoBot.sln** file to load the project into **Visual Studio 2017** IDE.
3. Now click on the  button to run the project.
4. It redirects to the browser, copy the **URL** present on browser and paste it into the botframework-emulator present on the desktop by double clicking on the botframework-emulator icon and append **/api/messages** text in the URL, click on Connect button.

**For eg:**

**http://localhost:3984/api/messages**

*Here you are successfully connected the bot with* ***botframework-emulator*** *to see the changes locally. Type text in the provided text box, The Bot will reply you intelligently.*

1. Now go to the **Visual Studio 2017** IDE and right click on the solution window-> **Add** and then click on **Existing Project**.
2. Go to the directory **C:\Source\contosoair\src\ContosoAir.Services,** select the file named **ContosoAir.Services.njsproj** and click on **OK** button.
3. Again, go to **Visual Studio 2017** IDE and right click on the solution window-> **Add** and then click on **Existing Project**.
4. Go to the directory **C:\Source**\**contosoair\src\ContosoAir.Clients.UWP,** select the file named **ContosoAir.Clients.UWP** and click on **OK** button.
5. Double Click on **config.js** file from **ContosoAir.Services** project to open in the code editor window.
6. Replace the values for parameters **DOCUMENT\_DB\_ENDPOINT**, **DOCUMENT\_DB\_PRIMARYKEY** with **URI** and **PRIMARY KEY** present in **Credentials** file on the **Desktop** and save the file.



8. Switch to **Azure Portal** by clicking on the link [portal.azure.com](http://portal.azure.com) as launched in **Scenario 1**.

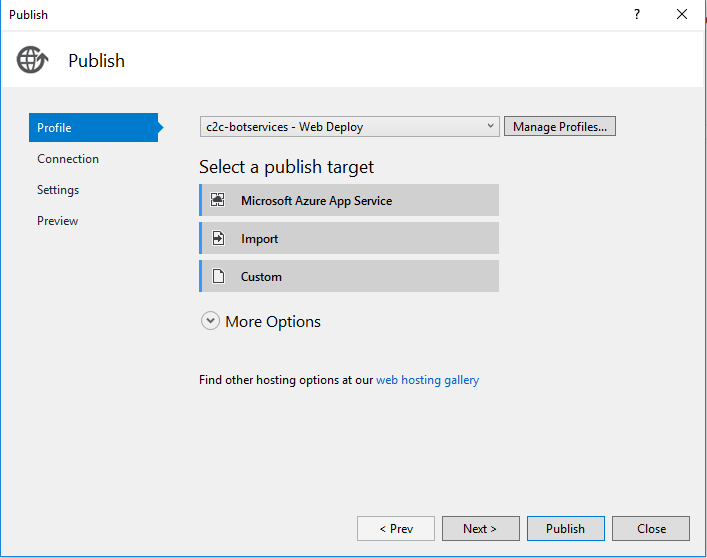
9. Click on the **Resource groups** option present under  icon and click on the Resource group named as “**BotResourceGrp**” under **Resource groups blade**.

10. Now click on the option present in the right-side panel with **Type** as **App Service.**

11. Click on  option present on Menu bar. It will **download** one file.

12. Now go to **Visual Studio 2017** IDE and right click on the project **ContosoAir.Services** from the Solution Explorer window and click on **Publish** option.

13. Click on the **Import** option present under **Profile** Tab. Click on the **Browse** button and select the downloaded file from **Downloads** section and click on **Open**->**OK** button.



14. Click on  button to validate the connection then click on **Next**->**Next**->**Publish** button. After successful publish it will redirect you to the browser.

15. Copy that **URL** and paste it into the file **Credentials** present on the **Desktop**.

### Part B – Publishing the Bot

1. Now go to the **Visual Studio 2017** IDE
2. Create two folders with the following name at the root level of the project named **Microsoft.Bot.Sample.SimpleEchoBot.**

* Forms
* Model

1. Right click on the **Dialogs** folder and click on the **Add** button. Select **Existing** **Item**.
2. Go to the **Documents** under **This** **PC** section, browse the file named as **AlternateFlight.cs** and click on **Add** button.
3. Right click on the **Forms** folder and click on **Add** button. Select **Existing** **Item**.
4. Go to the **Documents** under **This PC** section, browse the file named **FlightOptionForm.cs**, **OptionsForm.cs** and click on **Add** button.
5. Right click on the **Models** folder and click on **Add** button. Select **Existing** **Item**.
6. Go to the **Documents** under **This** **PC** section, select the file named **Options.cs**, **MenuOptions.cs**, **FlightStatus.cs**, **FlightOptions.cs**, **Alternatives.cs** and click on **Add** button.
7. Now right click on the Project **Microsoft.Bot.Sample.SimpleEchoBot** and select **Add** button.
8. Click on **Existing** **Item** and browse **Locale.cs** and **Constants.cs** files from the **Documents** under **This** **PC** and click on **Add** button.
9. Right click on the **Dialogs** folder, select **Add** and click on **Existing** **Item**. Replace the content of **EchoDialog.cs** with the new **EchoDialog.cs** file present at location **Documents** under **This** **PC**.
10. Click on **Existing** **Item** and Select the files named as **Local.cs, Constant.cs** and click on **Add** button.
11. Add using SimpleEchoBot.Dialogs; namespace in the MessagesController.cs file present under Controller folder.

Screenshot.

1. **Global.asax lines need to add.**
2. Copy 3 lines from Web.config.
3. <add key="DocumentDbUrl" value="https://c2cobt.documents.azure.com:443/"/>
4. <add key="DocumentDbKey" value="fKNcATLq4vu74YxVXyFDW3rEUdv17l8nKhEyH0xA9ArXUs1wVfqvSm61GirqS0pjMRRmQRpSU8wqvSpInhdpiA=="/>
5. <add key="APIEndpint" value="http://c2cbotapi.azurewebsites.net/api/bot" />E:\ContosoAir\BotCompleteProject\akkabot\Controllers\
6. Click on  to run the application locally.
7. It will take some time to run the application and after successful redirecting the application it redirects to the link <http://localhost:3984> copy the URL and append **/api/messages** in emulator and test it.

## Conclusion

In this experience, you explored the configuration of **Azure Bot Service, Cosmos DB.** You also played around the code using **Visual Studio – 2017 IDE** to change the Bot source code with the help of demo **ContosoAir App.**

Now you know how to

* Create **Bot** using **Azure** **Bot** **Services**
* Create collections/insert data in **Cosmos** **DB** using **Data Migration Tool**.
* Playing around the code using **Visual** **Studio** **2017** IDE.