**Tech Summit 2018**

**Hand­s-on lab**

Microsoft Bot Build Lab

****

Contents

[Microsoft Bot Build Lab 1](#_Toc526155015)

[Overview 3](#_Toc526155016)

[System requirements 4](#_Toc526155017)

[Exercises 4](#_Toc526155018)

[Exercise 1: Create Azure Service 5](#_Toc526155019)

[Exercise 2: Setup local bot project with Bot Framework SDK 21](#_Toc526155020)

[Exercise 3: Use the Bot Framework Emulator to test your local Bot 22](#_Toc526155021)

[Exercise 4: Deploy your local bot on Azure 25](#_Toc526155022)

[Exercise 5: Use Web Chat to test your Azure bot 26](#_Toc526155023)

Overview

Microsoft Bot Framework is a comprehensive offering to build and deploy high quality bots for your users to enjoy in their favorite conversation experiences. Developers writing bots all face the same problems: bots require basic I/O; they must have language and dialog skills; they must be performant, responsive and scalable; and they must connect to users – ideally in any conversation experience and language the user chooses. Bot Framework provides just what you need to build, connect, manage and publish intelligent bots that interact naturally wherever your users are talking – from text/sms to Skype, Slack, Facebook Messenger, Kik, Office 365 mail and other popular services. This lab is to setup a collaboration reserve Azure bot with language understanding, Q&A, translator and spell check service. User could chat with bot with more than 60 languages e.g. Chinese, English, Japanese etc. to reserve the restaurant and check reviews.



**System requirements**

* 1. You must have the following to complete this lab:
  + Azure Subscription
  + Windows 10
  + Microsoft Visual Studio 2017 Community (latest update)
  + Bot Framework Emulator (latest update)

**Exercises**

* 1. All of sources code could be found at
  2. <https://github.com/alezhao/Bot-Luis-QnA-Translator-Spell>
  3. This Hands-on lab includes the following exercises:
  4. Create Azure Service
  5. Setup local bot project with Bot Framework SDK
  6. Use the Bot Framework Emulator to test your Bot
  7. Deploy your local bot on Azure
  8. Use web chat to verity your Azure bot service
  9. Estimated time to complete this lab:  **50 minutes**.

Exercise 1: Create Azure Service

* 1. Login Azure portal with provide Azure subscription
  2. Create a notepad TXT file, copy and paste below notes into TXT file, save as config.txt at C:\lab\config.txt.

*<appSettings>*

*<!-- update these with your BotId, Microsoft App Id and your Microsoft App Password-->*

*<add key="BotId" value="" />*

*<add key="MicrosoftAppId" value="" />*

*<add key="MicrosoftAppPassword" value="" />*

*<!-- Azure bing spell check -->*

*<add key="IsSpellCorrectionEnabled" value="true" />*

*<add key="BingSpellCheckApiKey" value="" />*

*<!-- Azure translator -->*

*<add key="AzureTranslatorApiKey" value="" />*

*<!-- Azure bot storage to trigger message -->*

*<add key="AzureWebJobsStorage" value="" />*

*<!-- QnA maker -->*

*<add key="QnAmakerUriBase" value="" />*

*<add key="KnowledgeBaseId" value="" />*

*<add key="AuthorizationKey" value="" />*

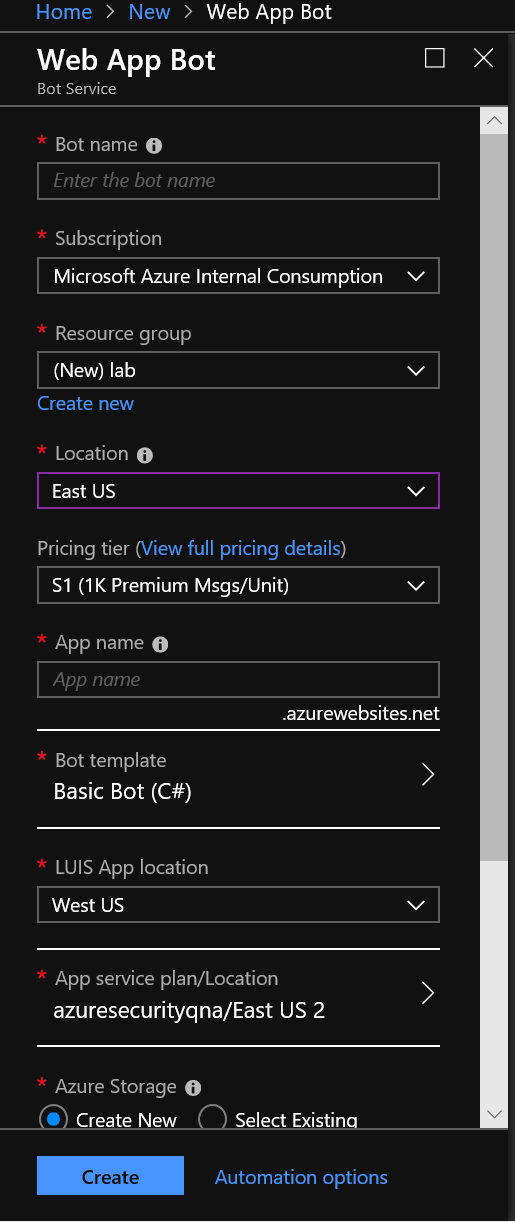
*<!-- Luis -->*

*<add key="LuisApplicationID" value="" />*

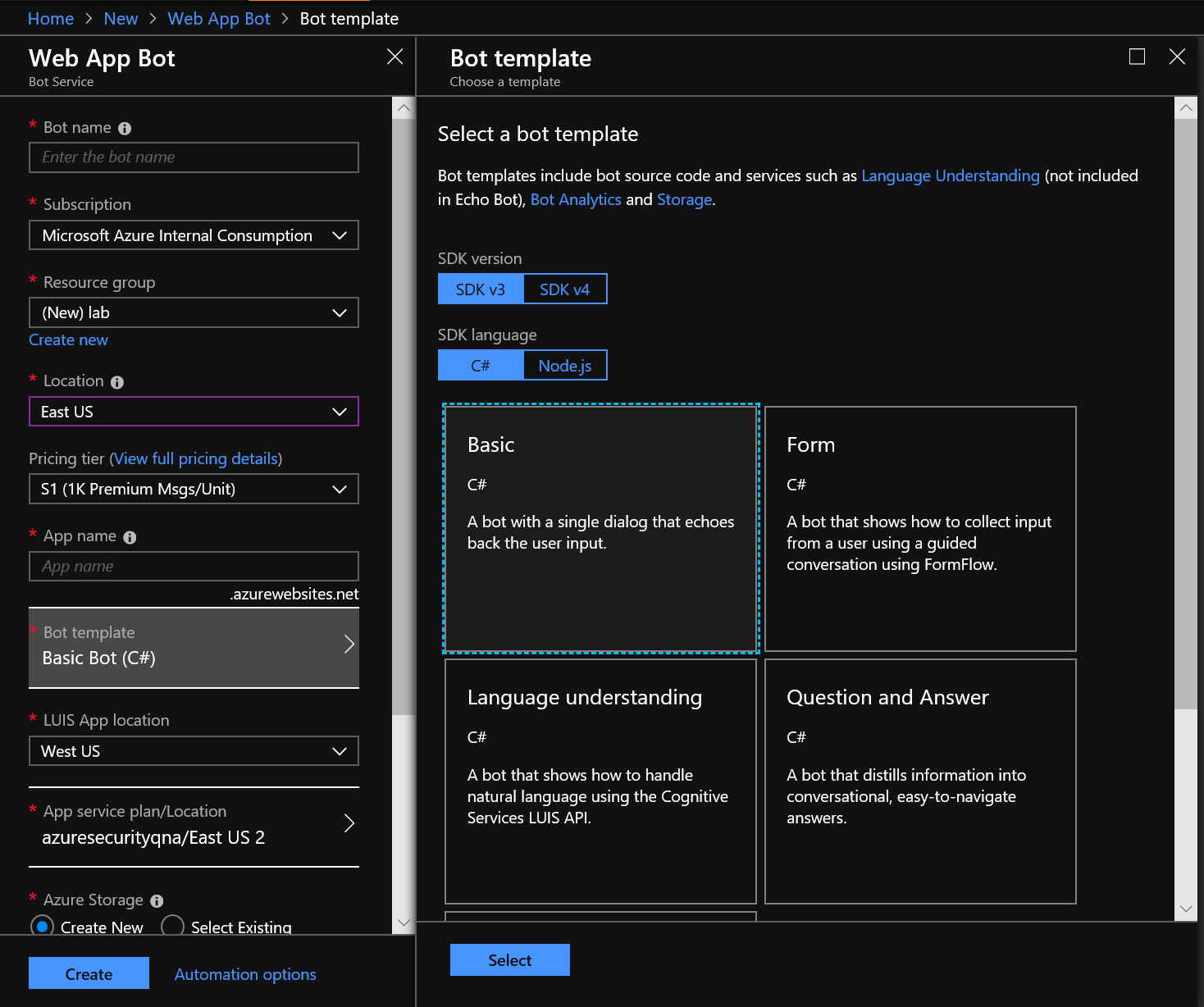
*<add key="LuisAuthoringKey" value="" />*

*</appSettings>*

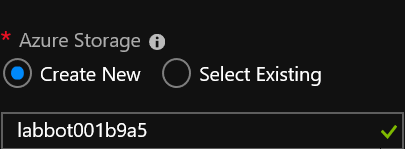
* 1. Create a Web App Bot Service
     1. Create a new **Resource Group**, click **Add** at top panel. (remember **Resource Group Name**, please create all of services with this **Resource Group**)
     2. Then entry **Web App bot**. A new blade will open with information about the **Web App Bot**.
     3. In the **Bot Service** blade, provide the requested information about your bot as specified in the table below the image.



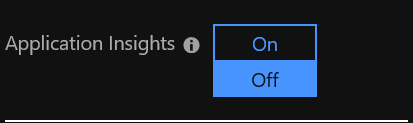
Entry your **[Bot Service Name],** entry **[App Name]** in **App name** section, choose lab provided **[Azure Subscription]** in **Subscription** section, select **East Aisa** at **Location** section, click **Bot template** section, select **SDKv3** with **Basic C#** template, click **Select.** Click and create new app service plan in **App service plan/Location** section, select **East Asia** at **Location** section at output panel.



Create New **[App Storage]** in **App Storage** section

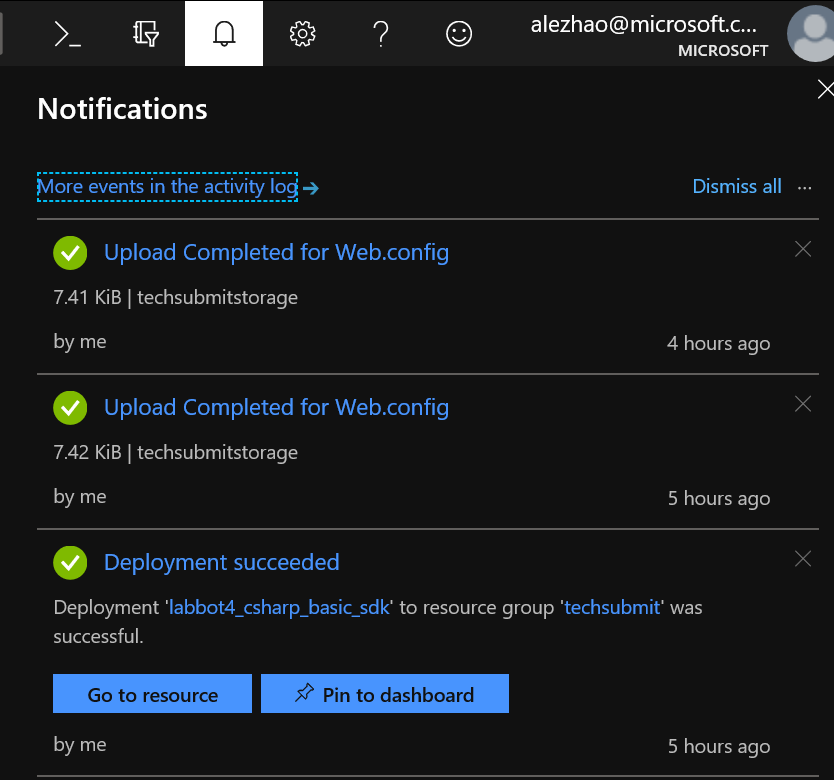


Unable **Application Insights** currently

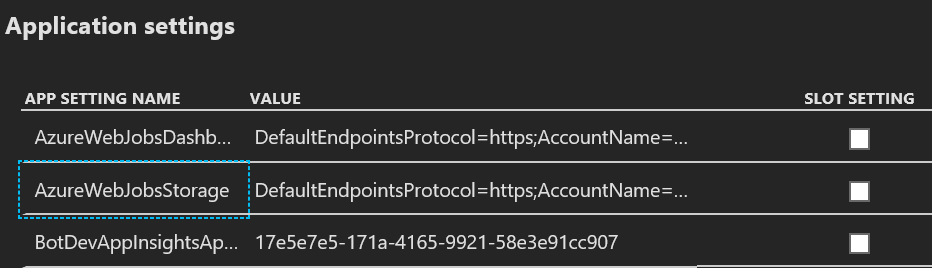


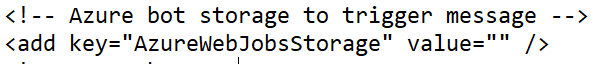
Click **Create**

* + 1. Click upper **Notification** link of right-hand, if resource created successfully, click **To to resource** to open service you just created.

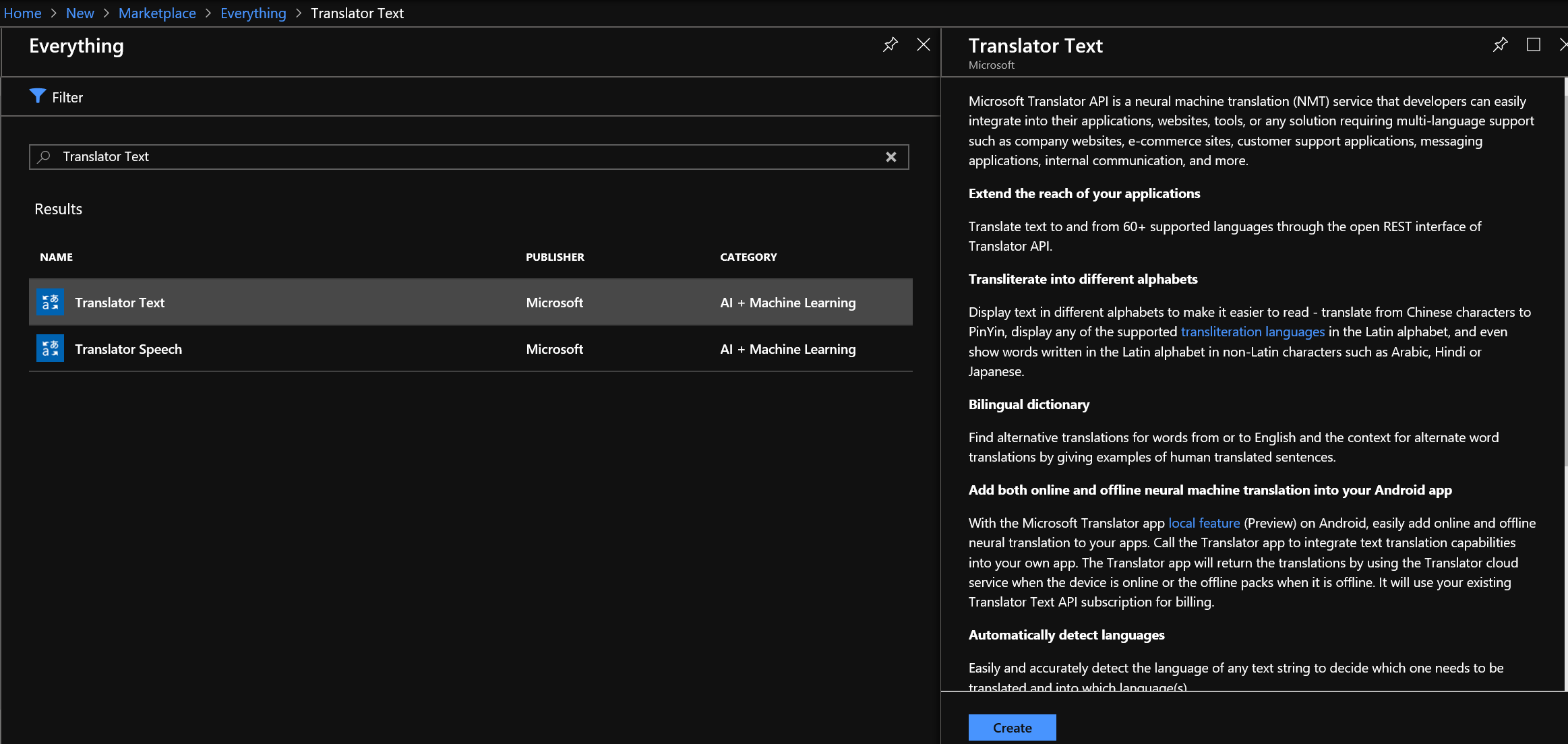


* + 1. Click **Application Settings** under **App Service Settings** section, copy value of **AzureWebJobsStorage** to C:\lab\config.txt

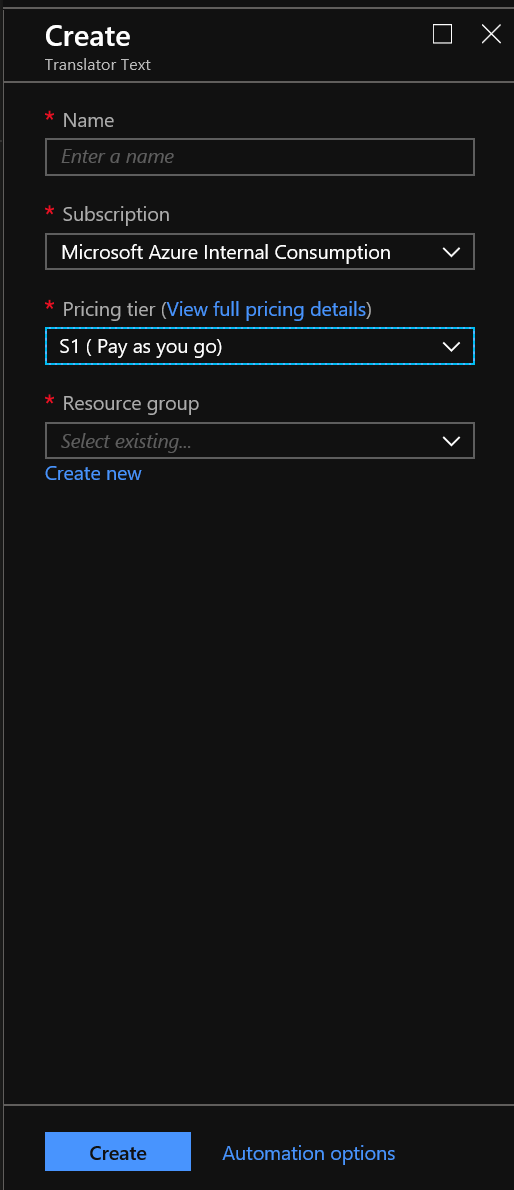




* 1. Create a Translator Text service
     1. Click **Create new resource** link found on the upper left-hand corner of the Azure portal, entry **Translator Text** in filter panel, click **enter** keyboard, select **Translator Text** service, click **Create**



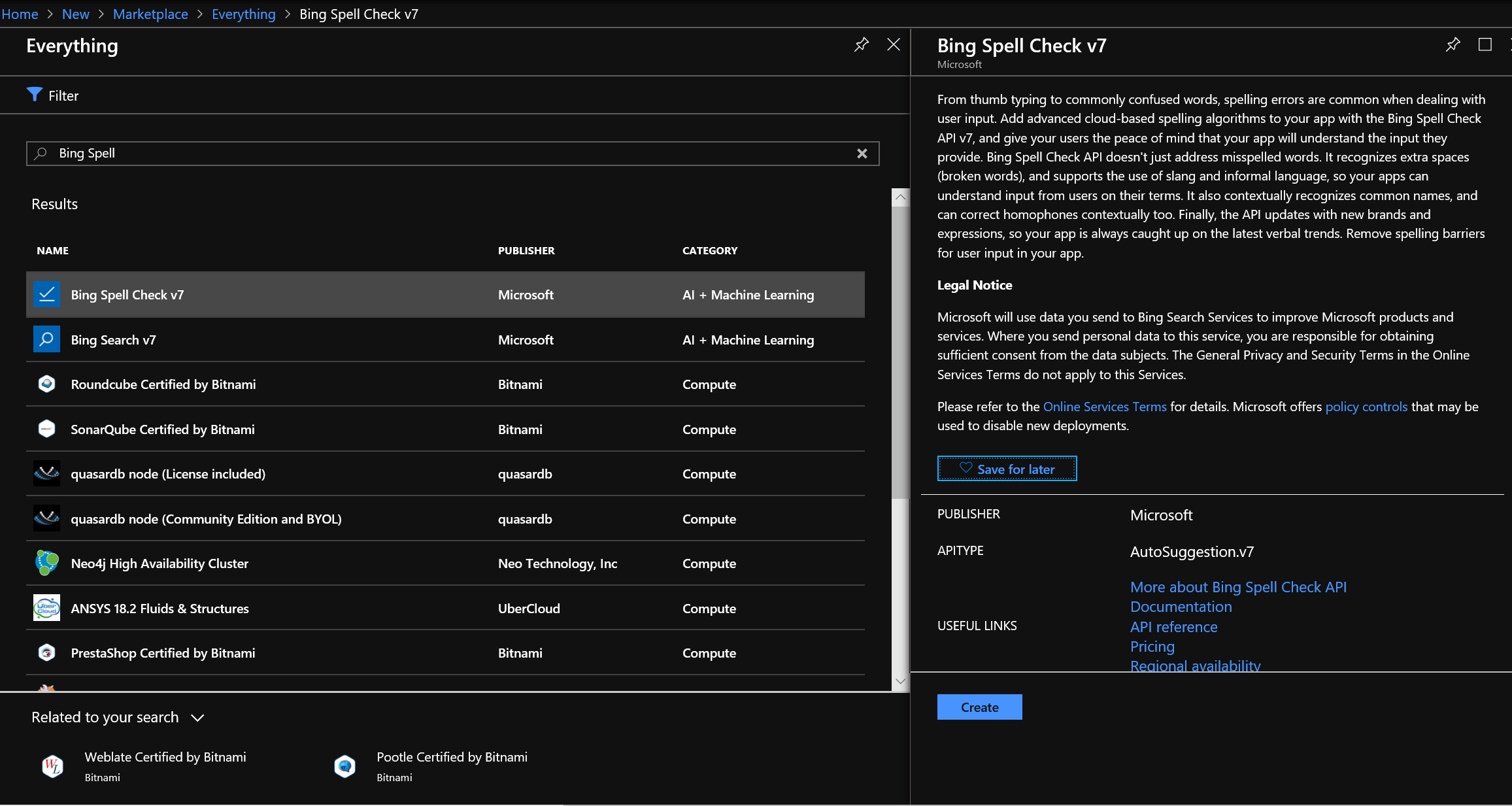
* + 1. Entry **[Translator Service Name]** of your service, choose lab provided **[Azure Subscription]** in **Subscription** section, select **S1** in Pricing tier section, select **techXXX** at **Resource group** section



* + 1. Click **Create**
    2. Click upper **Notification** link of right-hand, if resource created successfully, click **Go to resource** to open service you just created.
    3. Click **Keys** in **RESOURCE MANAGEMENT** section, copy KEY 1 into C:\lab\config.txt

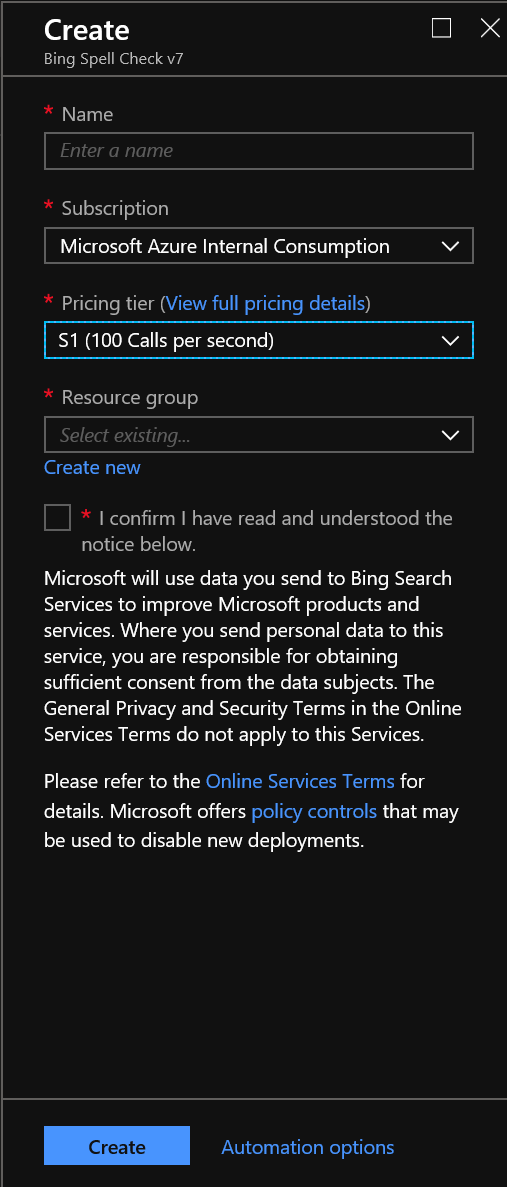


* 1. Create a Bing Spell Check service
     1. Click **Create new resource** link found on the upper left-hand corner of the Azure portal, entry **Bing Spell** in filter panel, click **enter** keyboard, select **Bing Spell Check v7** service, click **Create**

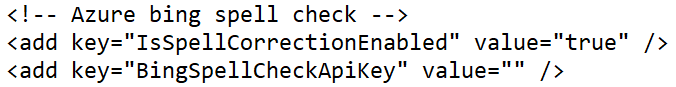


* + 1. Entry **[Spell Check Service Name]** of your service, Entry **[Translator Service Name]** of your service, select **S1** in **Pricing tier** section, select **techXXX** at **Resource group** section

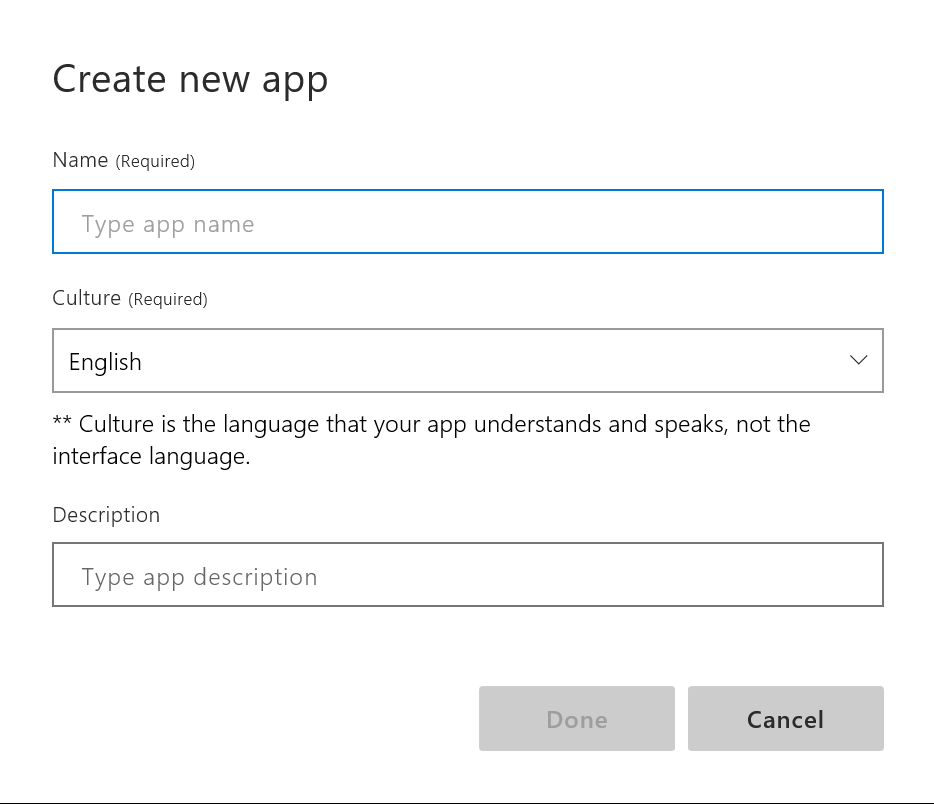
select **S1** in Pricing tier section, select **techXXX** at **Resource group** section, Check on **Confirm** condition.



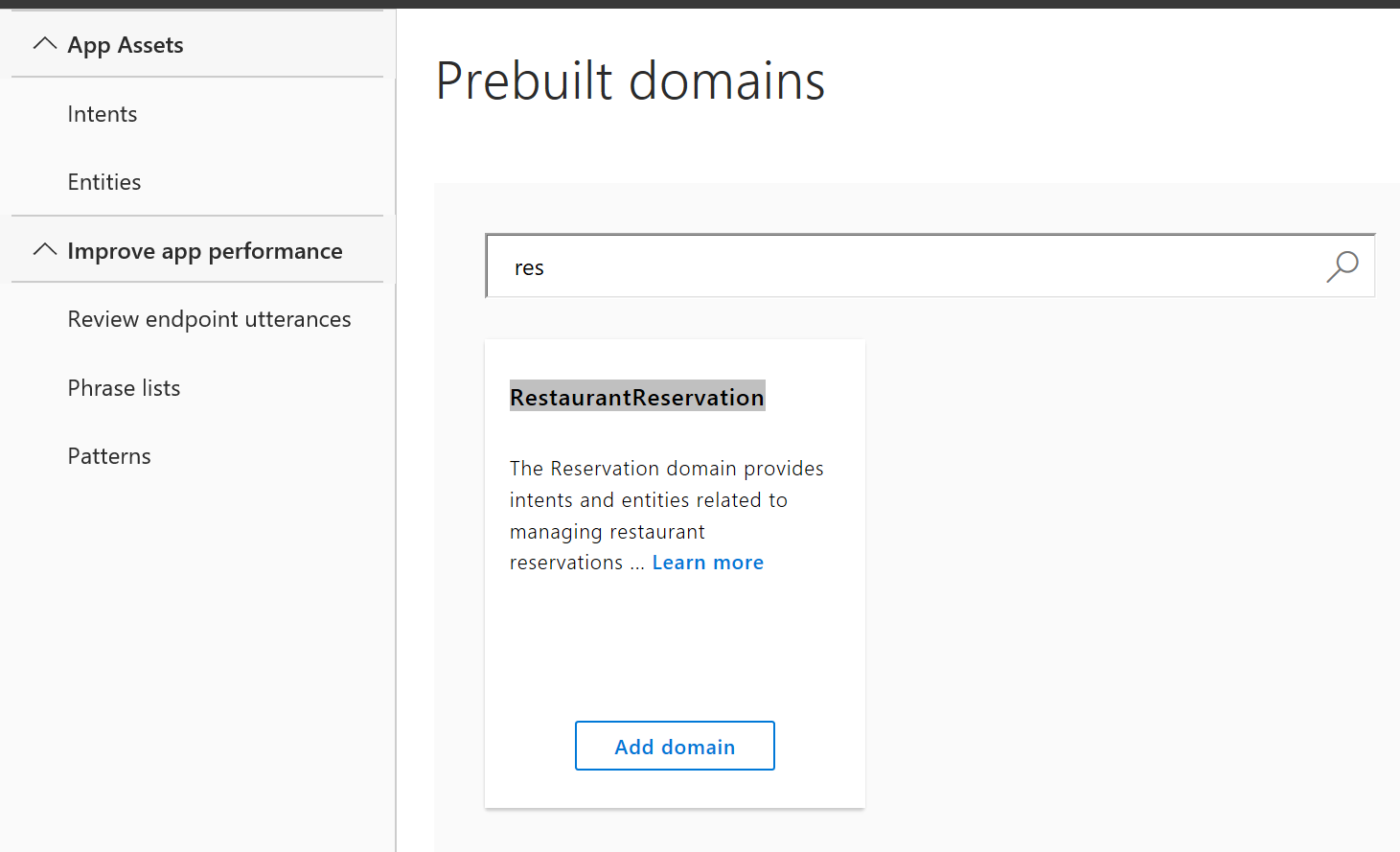
* + 1. Click **Create**
    2. Click upper **Notification** link of right-hand, if resource created successfully, click **To to resource** to open service you just created.
    3. Click **Keys** in **RESOURCE MANAGEMENT** section, copy KEY 1 into C:\lab\config.txt



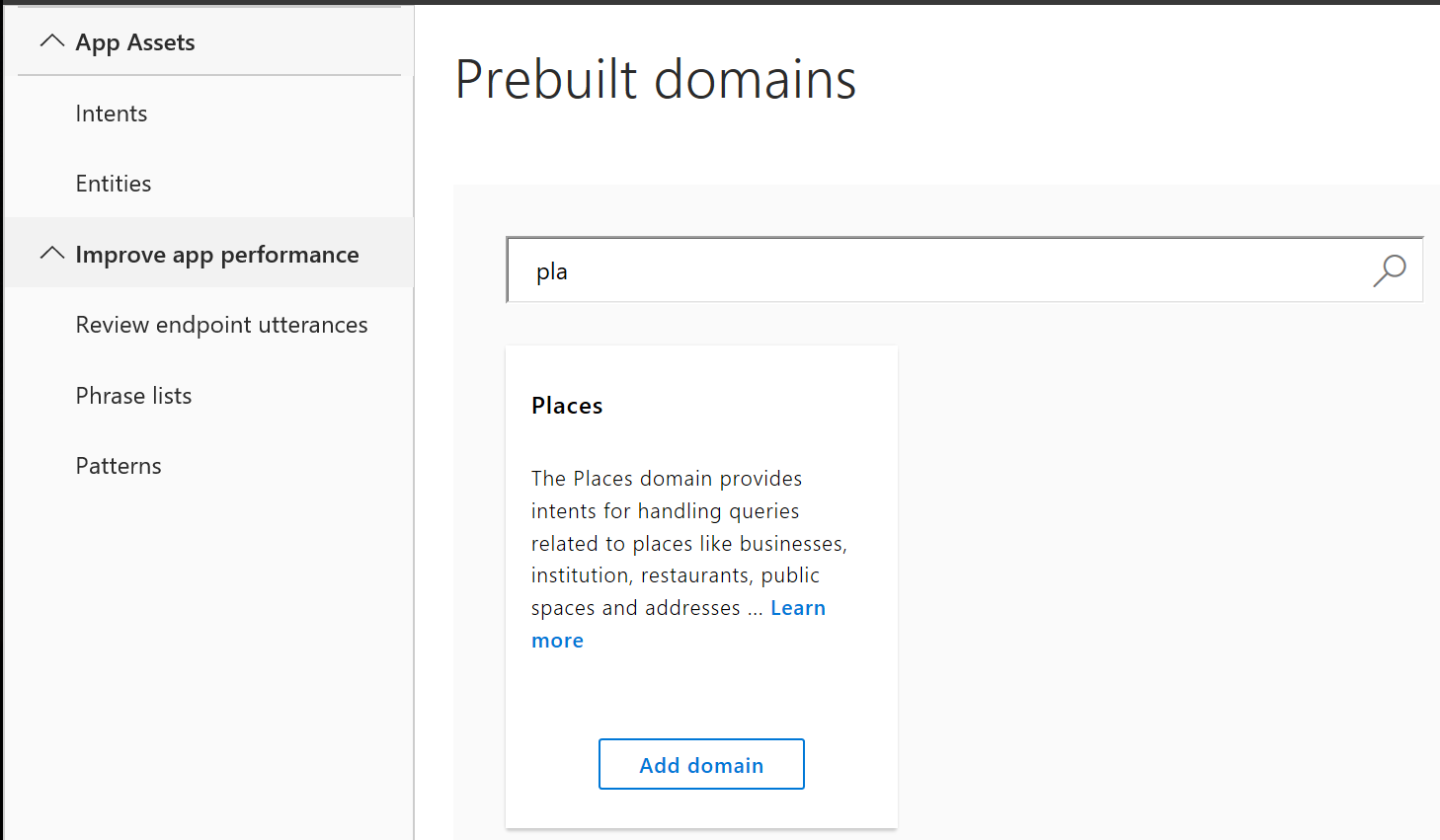
* 1. Create a LUIS service
     1. Open LUIS portal <http://www.luis.ai>, click **Login,** entry lab provided **[Azure Subscription]** and **[Password]** to login
     2. Click **Create new app** link found on the upper left-hand corner of the LUIS portal, entry **[LUIS Name]** in **Name** section, Click **Done**



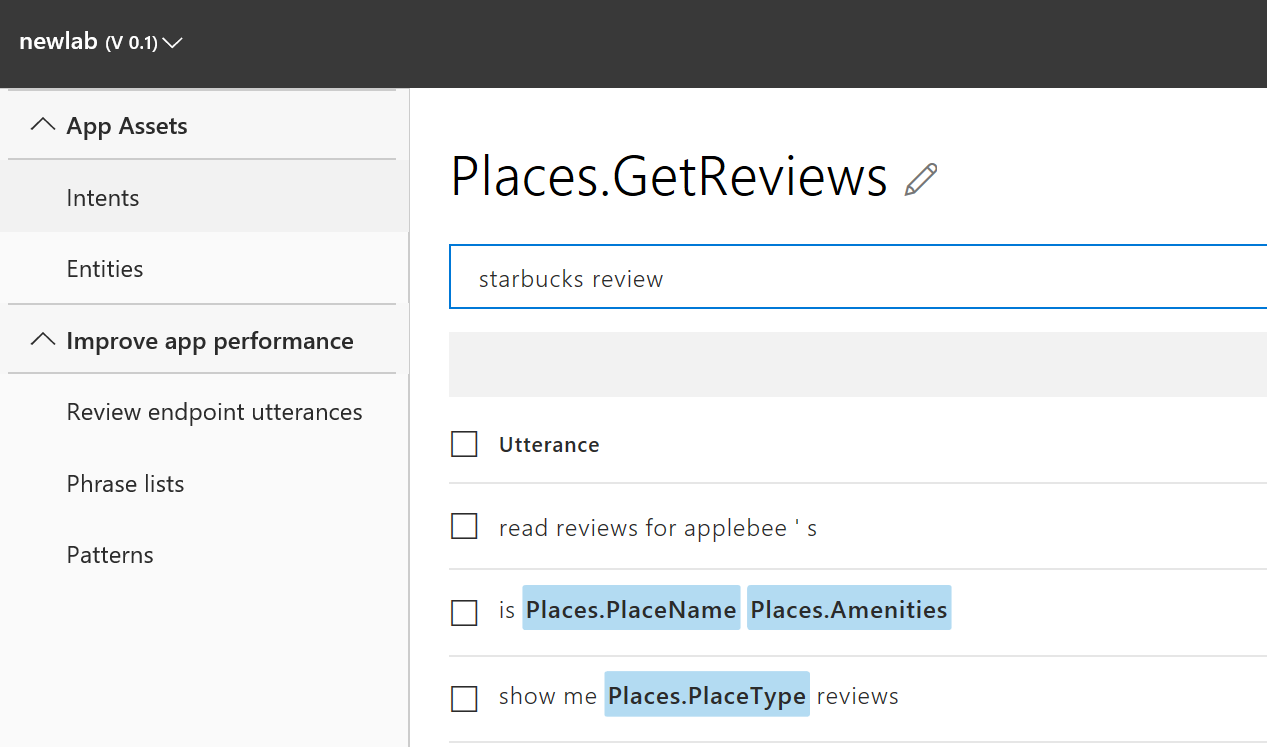
* + 1. Click **Prebuilt domains** link found on the bottom left-hand corner of the LUIS portal, entry **res** in the filter panel, click **Add Domain** of output **RestaurantReservation** domain



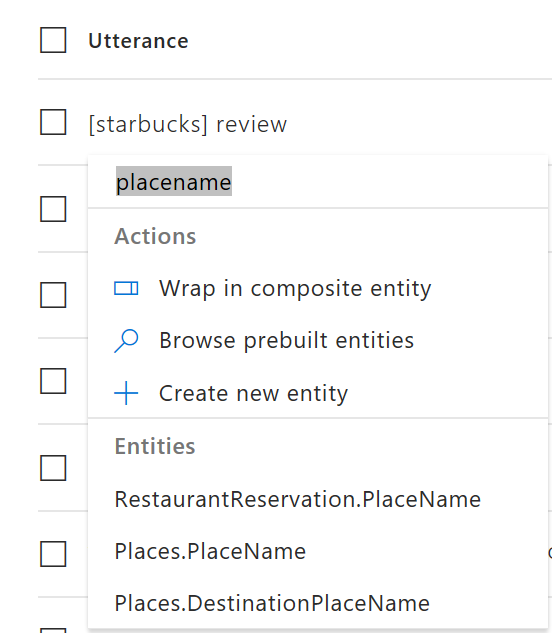
Entry **pla** in the filter panel, click **Add Domain** of output **Places** domain



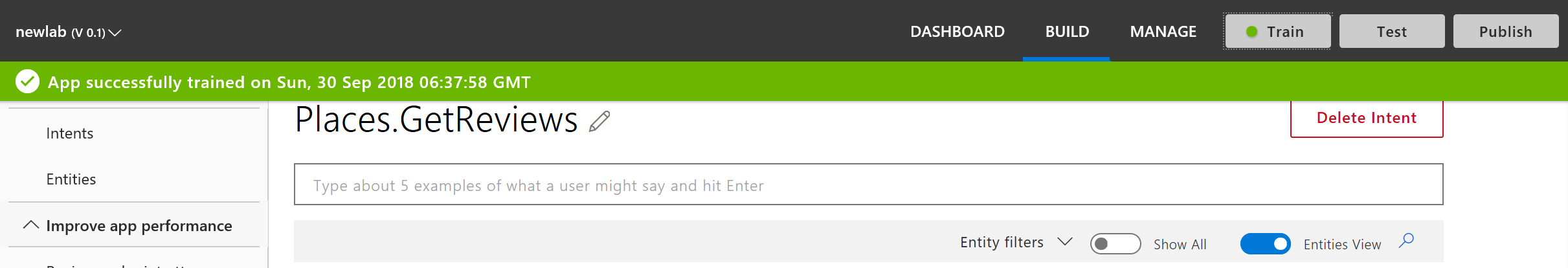
* + 1. Click **Intents** link found in left panel, find **Places.GetReviews** and click, entry **starbucks reviews** at utterance entries to add new utterance, click **enter** keyboard



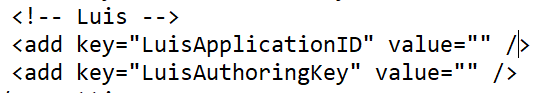
* + 1. Click **starbucks** word in new utterance added, entry **placename** at pop-up window filter panel, click **Places.PlaceName**



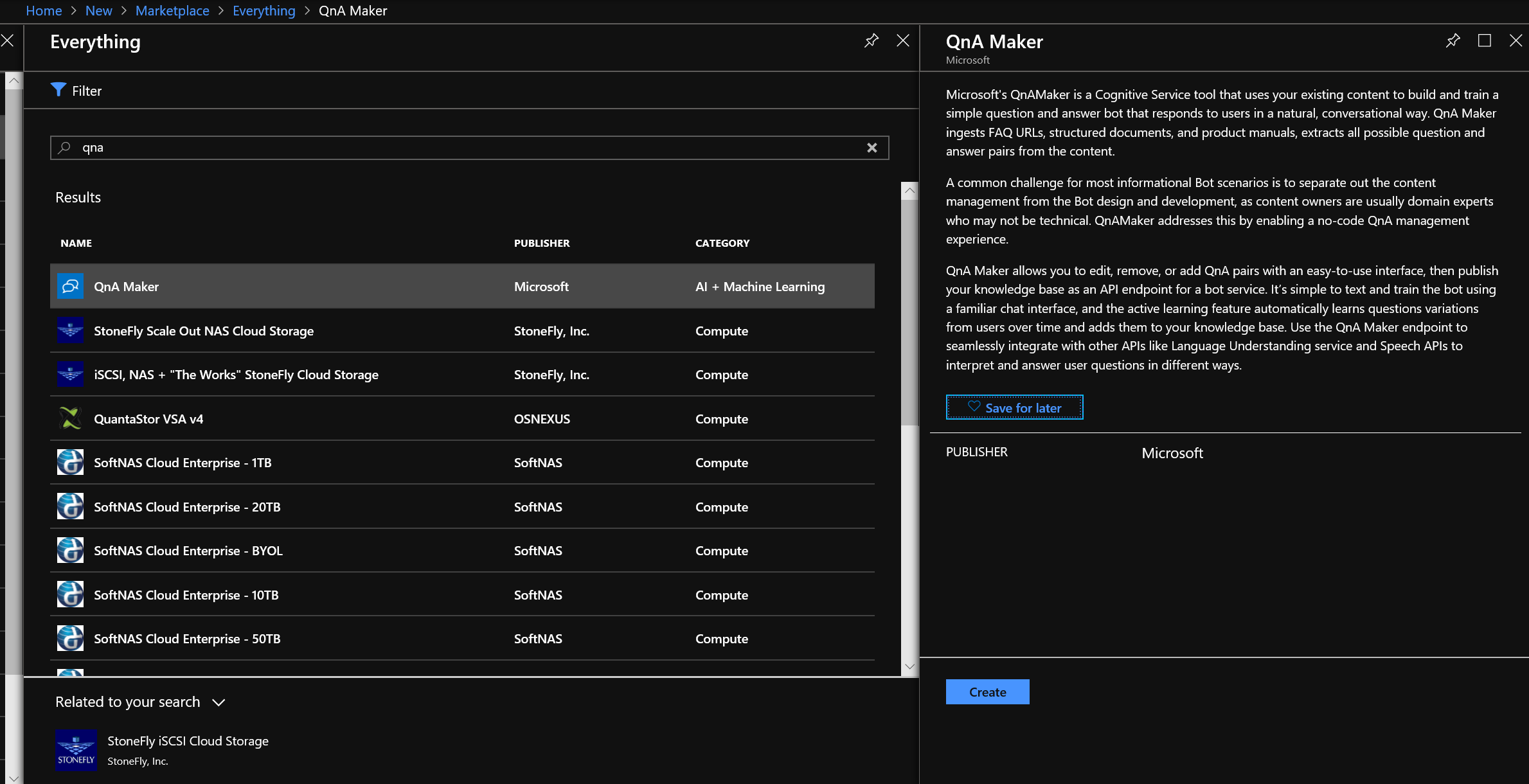
* + 1. Click **Train** link found on the upper right-hand corner of the LUIS portal, once status shows complete, click **publish** to on the upper right-hand corner to publish LUIS app to production



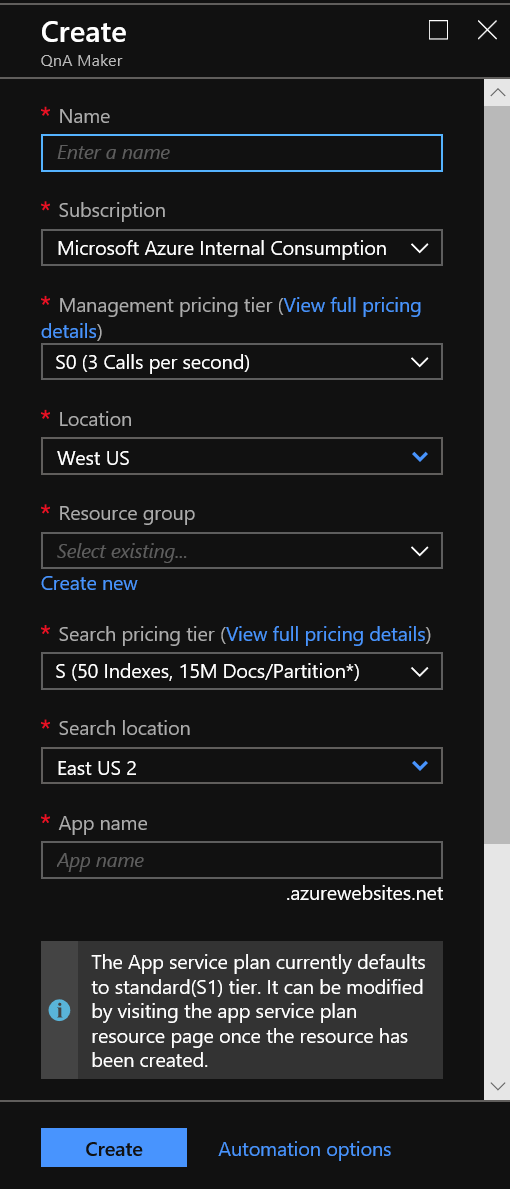
* + 1. Create a notepad, save as config.txt.
    2. Click MANAGE tab at upper panel, save **Application ID** at left-hand **Application Information** panel to C:\lab\config.txt, save **Authoring Key** at left-hand **Keys and Endpoint** panel to C:\lab\config.txt



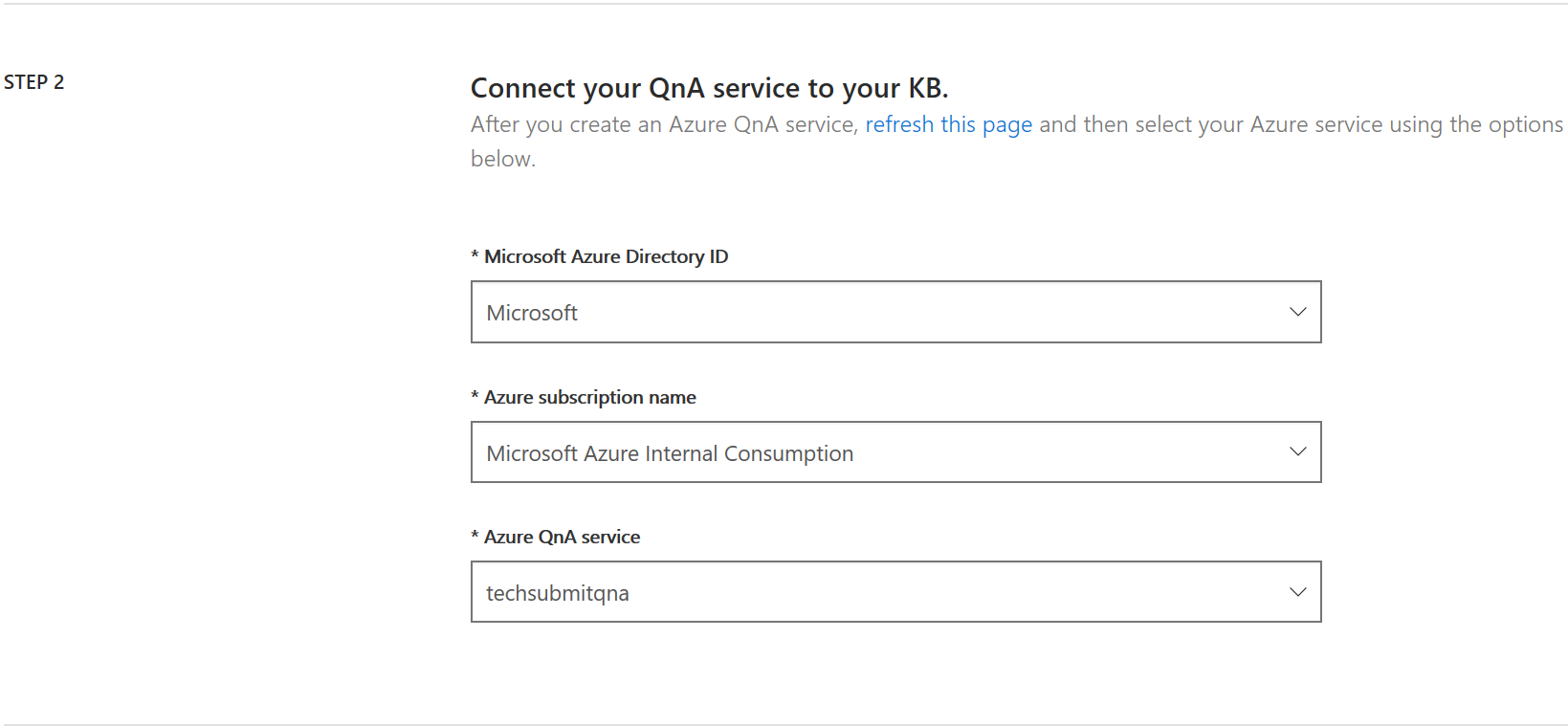
* 1. Create a QnA maker service
     1. Click **Create new resource** link found on the upper left-hand corner of the Azure portal, entry **QnA** in filter panel, click **enter** keyboard, select **QnA Maker** service, click **Create**



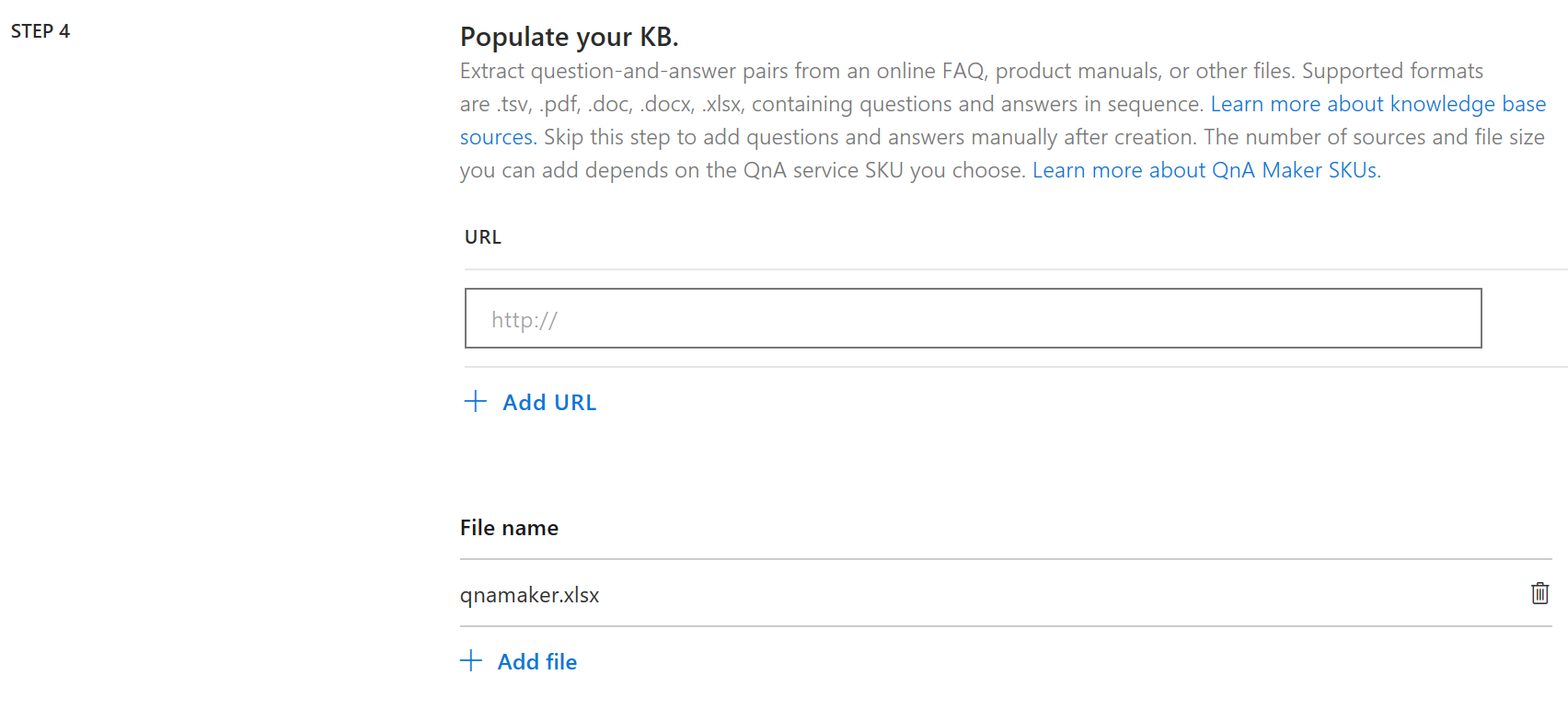
* + 1. Entry **[QnA Maker Service Name]** in **Name** section, choose lab provided **[Azure Subscription]** in **Subscription** section, select **West US** in **Location** section, select **S0** in **Management price tier** section, select **S** in **Pricing tier** section, entry **[App Name]** in **App name** section, select **West US** at **Search location** section.



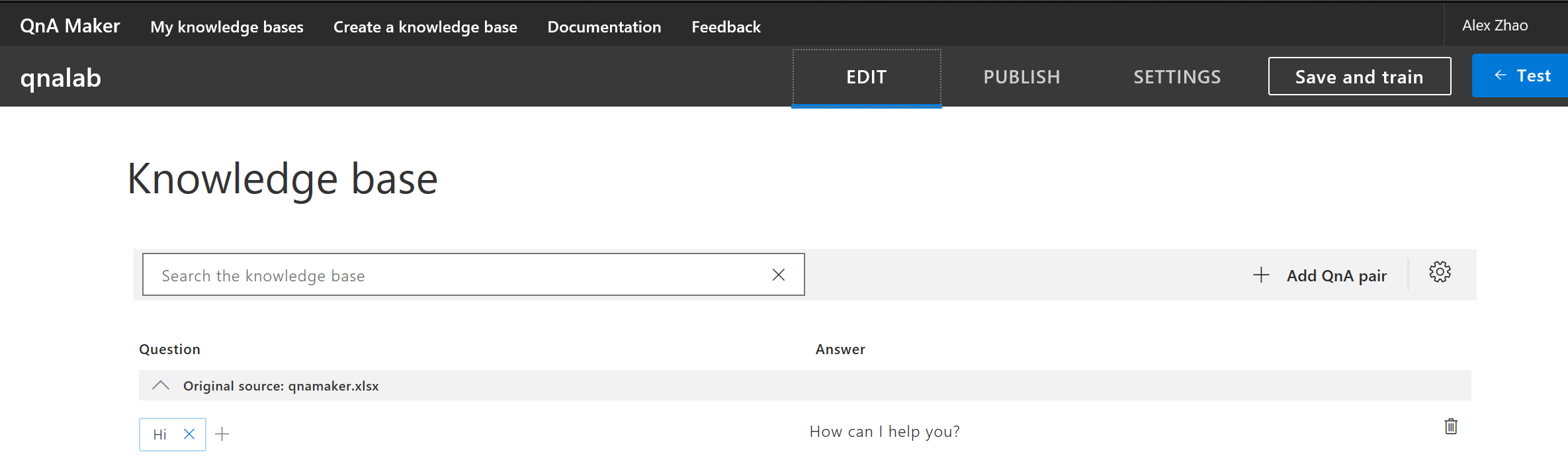
* + 1. Click **Create**
    2. Open QnA maker portal <https://www.qnamaker.ai/>, click Create a knowledge base at upper tab.
    3. Under section **Step 2**, select **Microsoft** in **\* Microsoft Azure Directory ID** section, choose lab provided [Azure Subscription] in **\* Azure subscription name** section, select **[QnA maker Service Name]** which created by pervious step in **\* Azure QnA service** section.



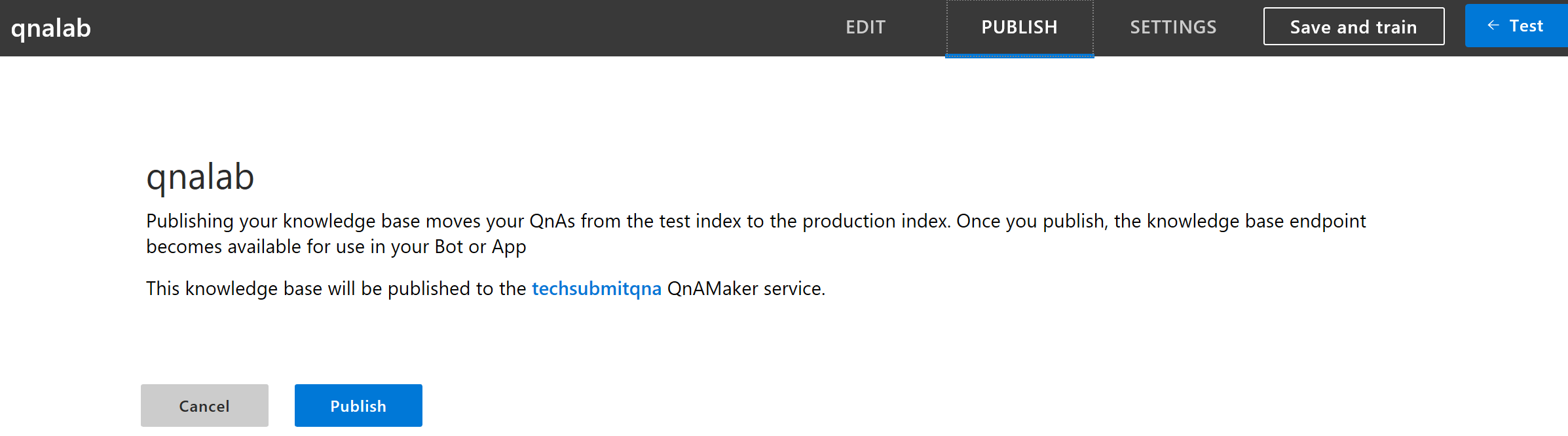
* + 1. Under section **Step 3**, entry your **[Knowledge Base Name]** in **Name your KB** section.
    2. Under section **Step 4**, click c, select **C:\lab\qnamaker.xlsx** to upload



* + 1. Click **Create your KB** at the bottom.
    2. Under your created QnA Maker App, click **Save and train** at upper of right-hand corner.



* + 1. After train completed, select **PUBLISH** at upper panel, click **Publish**



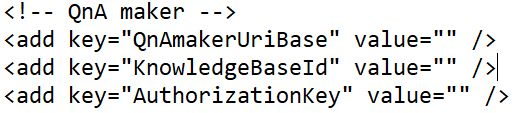
* + 1. After publish completed, go to SETTING panel, scroll down to the bottom, copy **[Knowledge Based ID]**, **[QnAmakerUriBase]** and **[AuthorizationKey]** found at

POST /knowledgebases/**[Knowledge Based ID]**/generateAnswer

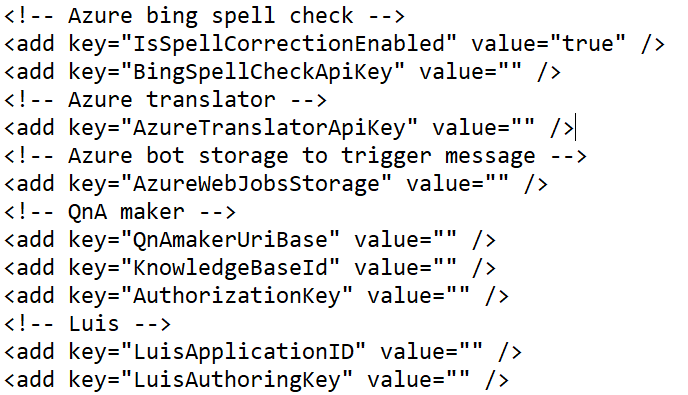
Host: **[QnAmakerUriBase]**

Authorization: **[AuthorizationKey]**

into C:\lab\config.txt, save config.txt

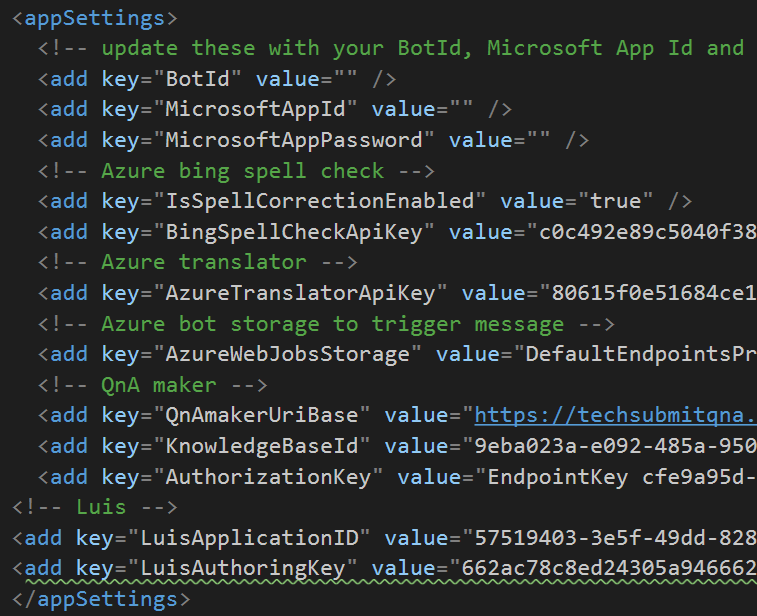


* 1. Verify C:\lab\config.txt, to check if below keys are all setup.

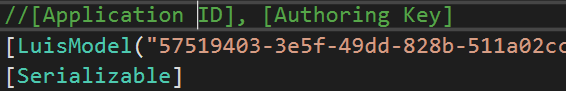


Exercise 2: Setup local bot project with Bot Framework SDK

* 1. Double click **Microsoft.Bot.Sample.SimpleEchoBot.sln** in C:\lab\labbot\, open Bot solution with Visual Studio 2017 community.
  2. Double click Web.config under project **Microsoft.Bot.Sample.SimpleEchoBot,** replace < appSettings >…</ appSettings> with values in C:\lab\config.txt



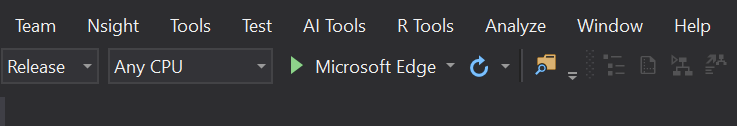
* 1. Entry **Root** at upper of right-hand filter, double click **RootLuisDialog.cs** from filter result, replace [LuisModel("[Application ID]", "[Authoring Key]")] with values in C:\lab\config.txt, Ctril+S to save, clear filter.



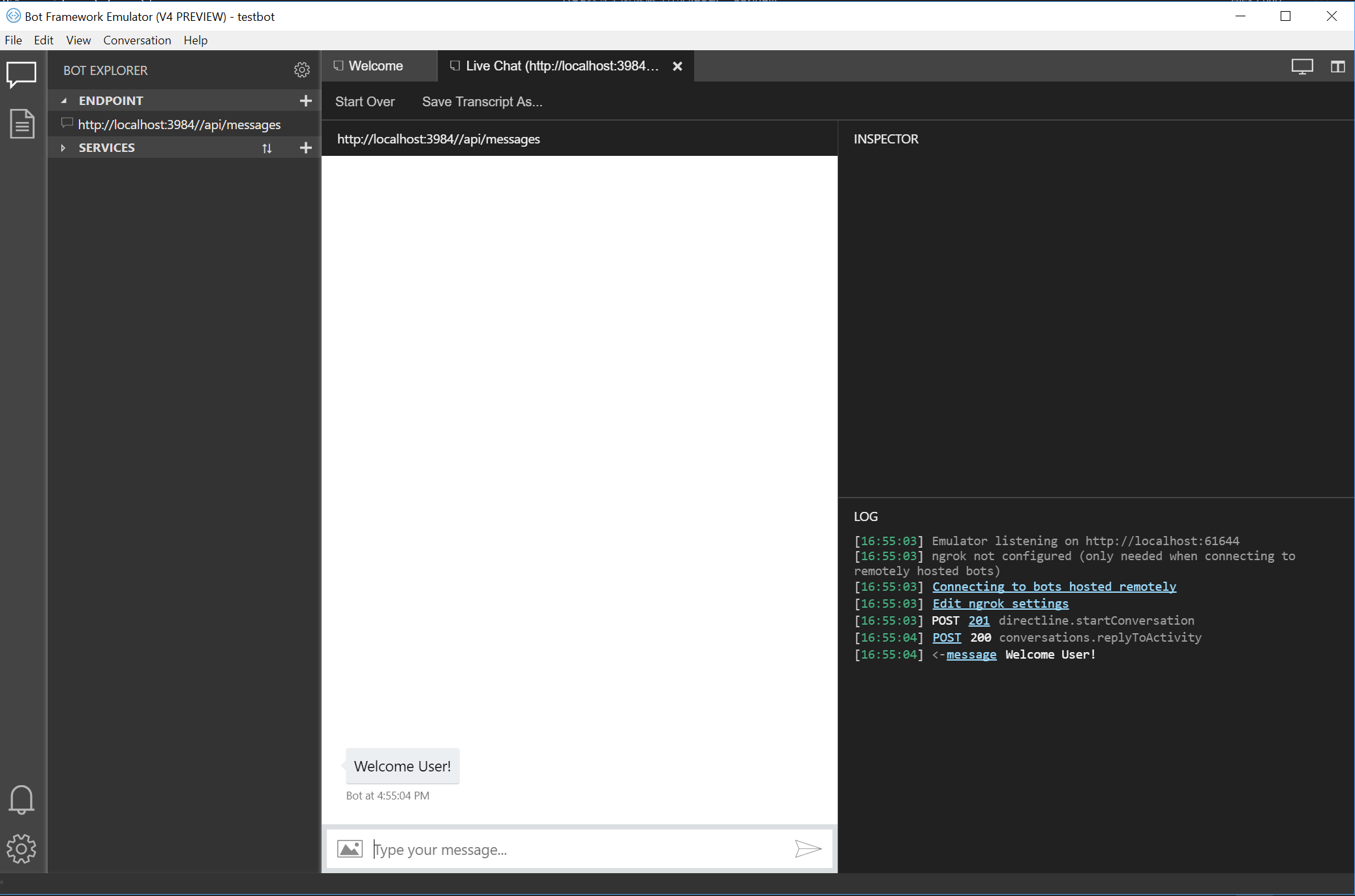
* 1. Right click project name **Microsoft.Bot.Sample.SimpleEchoBot** at upper of right-hand corner, select **Build**

Exercise 3: Use the Bot Framework Emulator to test your local Bot

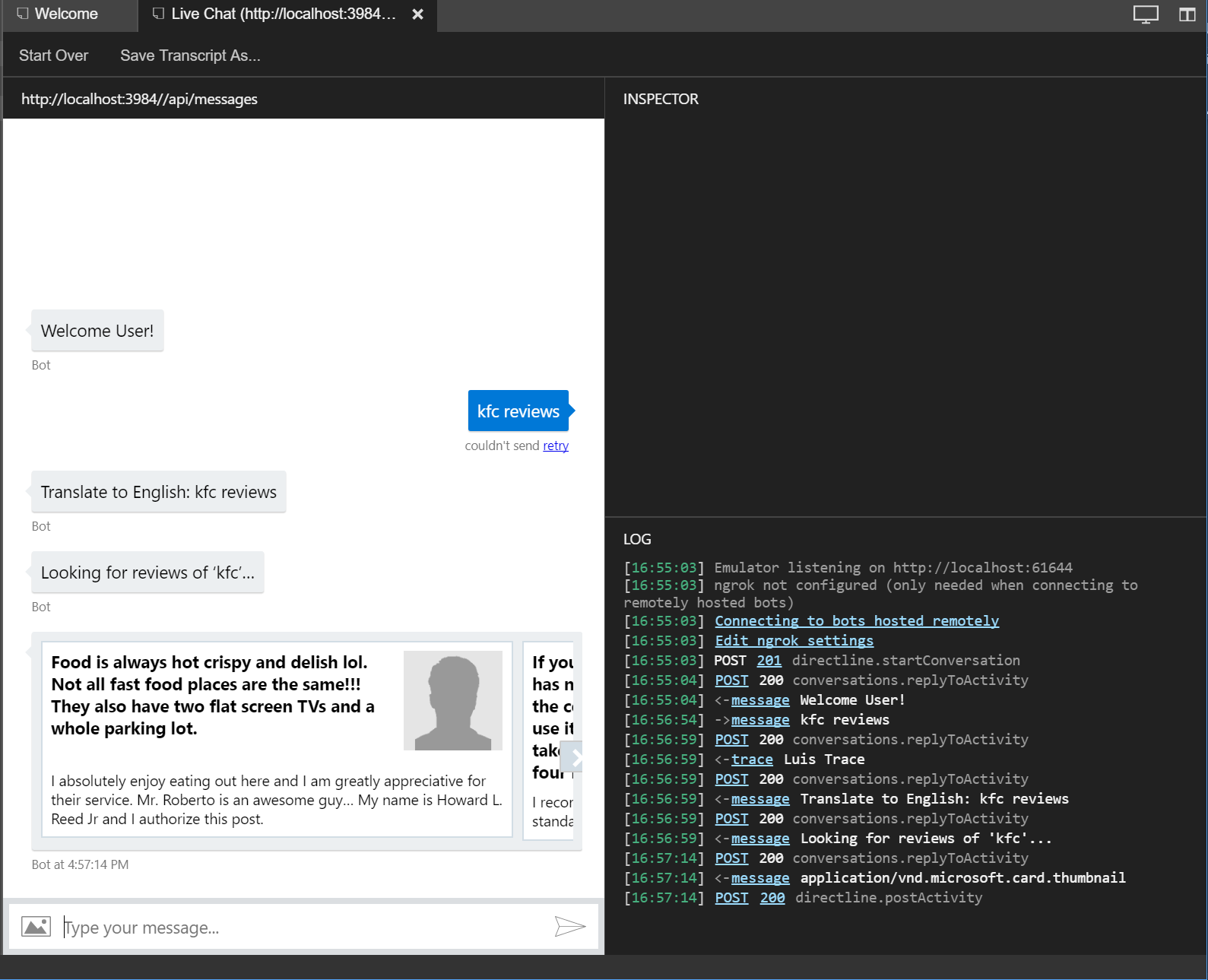
* 1. Click debug with **IIS Express (Microsoft Edge)** upper of middle, copy opened **[Local URL]** with port e.g. <http://localhost:3984/>



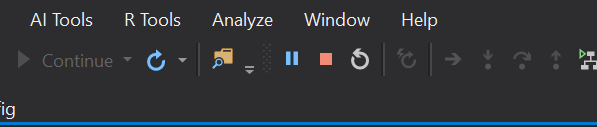
* 1. Open **Bot Emulator** at Decktop, click **Create a new bot configuration,** entry **[Bot Name],** entry **[Local URL]/api/messages** in Endpoint URL section. E.g <http://localhost:3984//api/messages>, click **Save and connect**, save configuration file under C:\lab\[Bot Name].bot
  2. Verify if Bot Emulator return 200 at LOG section with **Welcome User!**



* 1. Entry some utterance to check e.g. KFC reviews. If return like below then success. Now your local bot is working, and connected to LUIS, translator, spell check and QnA maker. Next step we need deploy your local bot onto Azure.

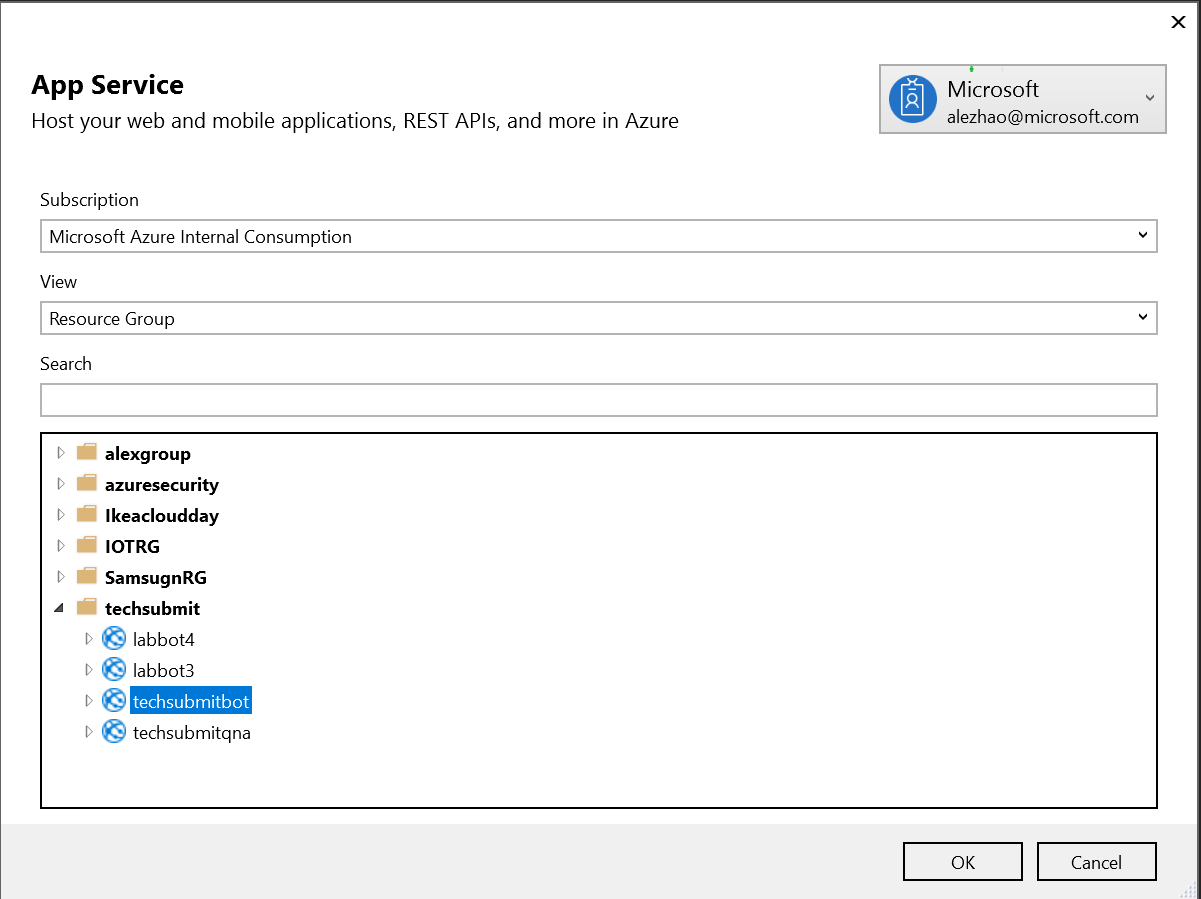


* 1. Stop debug by click **Stop** **Debugging** at middle of upper panel.

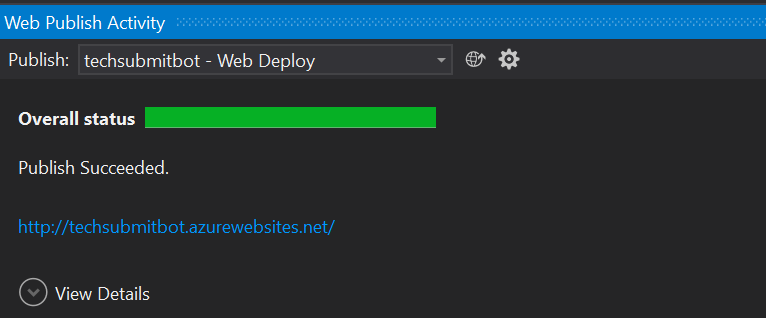


Exercise 4: Deploy your local bot on Azure

* 1. Right click project name **Microsoft.Bot.Sample.SimpleEchoBot** at upper of right-hand corner, select **Publish**
  2. Select **Microsoft Azure App Service** in left panel **Publish** section, select **Select Existing,** click **Publsih,** login with lab provided **[Azure Subscription]** and **[Password],** select **[Azure Subscription],** select **techXXX**, click **[Bot Service Name]** created in pervious step, click **OK**



* 1. Verify if publish succeeded.



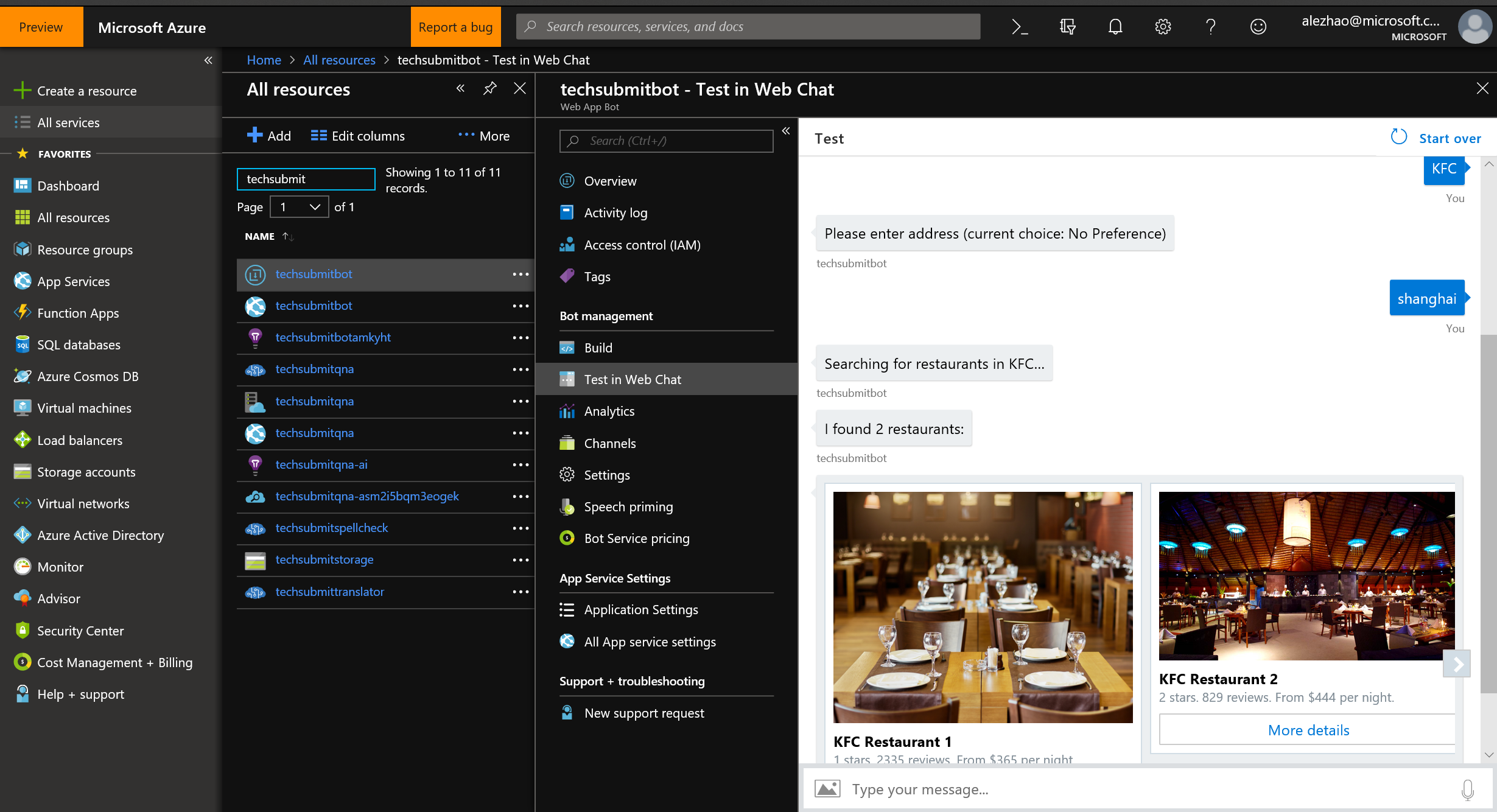
Exercise 5: Use Web Chat to test your Azure bot

* 1. Login Azure portal with provide Azure subscription
  2. Click **All resources** at left-hand panel, entry **[Bot Service Name]** to find your bot service and click, select **Test in Web Chat** under **Bot management** section.
  3. Entry test utterance (multiple languages) to verify your bot service on Azure

e.g.

“I wang to book a room at KFC”

“KFC reviews”



Click **Channels** under **Bot management** section, click **Edit** of **Web Chat,** copy first **Secret Keys** at the top and paste to a new notepad, copy and paste URL value in src of **Embed code** to notepad, replace **YOUR\_SECRET\_HERE** with **Secret Keys** just saved



Open **Microsoft Edge**, put whole URL to URL section and search, you will find a web chat bot is ready, it also could integrated as a frame.

* 1. Entry test utterance (multiple languages) to verify your web chat bot.

