# Rubicon Global Azure Bootcamp 2018

Version 20180406 - 1

# LAB 2: Step 1 – Create a LUIS Bot in Azure Bot Services

We start by creating a new Bot Service in Azure. These steps will also create a new LUIS app for us for step 2.

1. Login to your azure portal (<https://portal.azure.com>)
2. Click on ‘+Create new resource”
3. Search for “bot service”
4. In the search results, look for “Web App Bot” and click on it
5. Click Create
6. Give the bot a name “GAB2018BOT-<YOURINITIALS>”  
   *(Replace <YOURINITIALS> with your initials!)*
7. In location select “West Europe”
8. In pricing tier, select “F0” as your pricing tier
9. Click on bot template
10. Click on “Language understanding” box (SDK Language is default set on C#)
11. Click Select
12. Check “I confirm I have read and understand the notice below”
13. Click “Pin to dashboard” for easy access
14. Click Create

# LAB 2: Step 2 – Create a LUIS App & Train LUIS

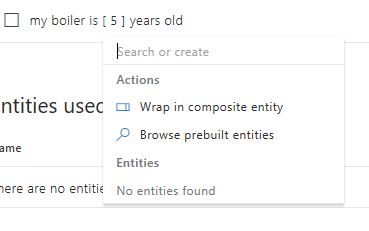
We have created the bot, now we are going to build the vocabulary we need with LUIS and train some language understanding for our AI part of the chatbot.

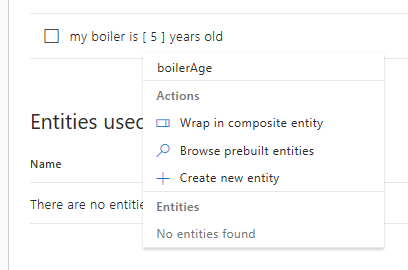
1. Open your web browser and surf to https://www.luis.ai
2. Click on Sign In
3. Sign in with the same Azure credentials as the one used in Step 1
4. Step 1 has already created a Luis application for you
5. Select the application (in our example “GAB2018BOT-<key>”)

You will now get to your applications default page. You can choose to "Create new intent" or "Add prebuild domain intent"

1. The template used in step 1 has already created a number of Intents (Cancel, Greeting, Help and None)
2. Click "Create new intent"
3. Fill in a descriptive name for your intent. i.e. "Warranty"
4. Now add the sentences that will be typical to trigger this event. Add the following sentances:
   1. is my warranty still good
   2. when will my warranty expire
   3. is this repair still covered by my warranty
   4. how much will this repair cost me
   5. do i have to pay to get this fixed

Let’s create a second Intent

1. Select Intents
2. Click [Create new intent]
3. Type “BoilerAge”
4. Click [Done]
5. Type the following sentence in the first input field: “My boiler is 5 years old”
6. Click on “[5]”  
   
7. Enter the name for this entity “boilerAge” and click Create new entity’

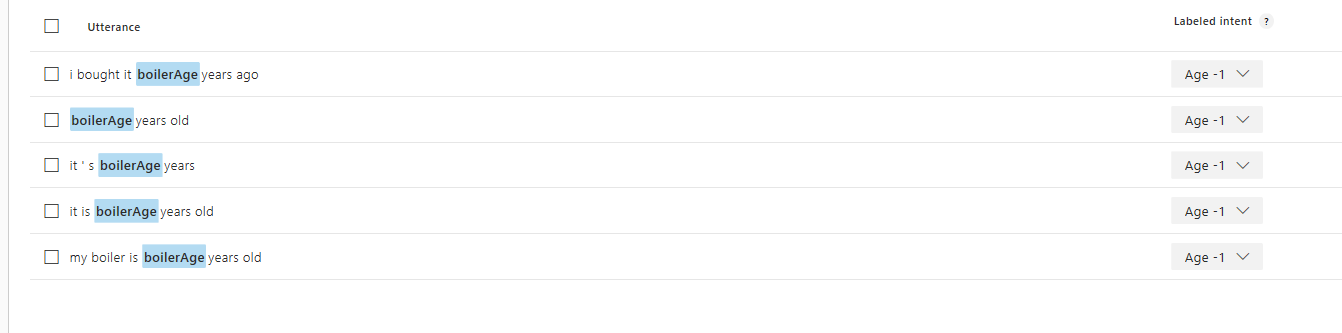


1. In the popup, select “Simple” as entity type
2. Click Done

Repeat steps 16 to 18 for the following 4 sentances. Each time selecting the value “5” and selecting ‘boilerAge’ from the entities:

* i bought it 5 years ago
* 5 years ago
* it’s 5 years old
* it is 5 years old

1. Your intent should look like this:



We can now train our app.

1. Click [Train] on the top menu
2. After training is done.
3. Click on [Publish] on top menu
4. Select ‘Production’ in the Publish To option.
5. Leave other settings default.
6. Click [ Publish to product slot]

You can now test your app

1. Click on [Test] in the top menu.
2. Type a sentence (different from the once you used to train)

# Lab 2: Step 3 – Customize BOT

We have built the base of our bot. Trained and published our AI using a LUIS app. Now let’s go ahead and customize our bot.

1. Login/return to the Azure portal
2. Select the app you buildt in step 1
3. Click on ‘Build’ (below BOT MANAGEMENT section)
4. Click on ‘Download zip file’ to get the sourcecode
5. Wait until the zip file is created and click on the button [Download zip file]
6. Extract the Zip file to a folder (your development folder of choice)
7. Start Visual Studio
8. Click File -> Open -> Project/solution
9. Select \*.sln file in the directory you have extracted the solution

*THE NEXT STEPS ARE ONLY NECESSARY IF YOU WANT TO TEST LOCALLY. YOU CAN SKIPP THESE AND CONTINUE WITH STEP 17.*

You need to add the following keys to the web.config for local build and run.

1. Open the web.config file.
2. Add the following keys to <appSettings>

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

<add key="LuisAPIKey" value="" />

<add key="LuisAppId" value="" />

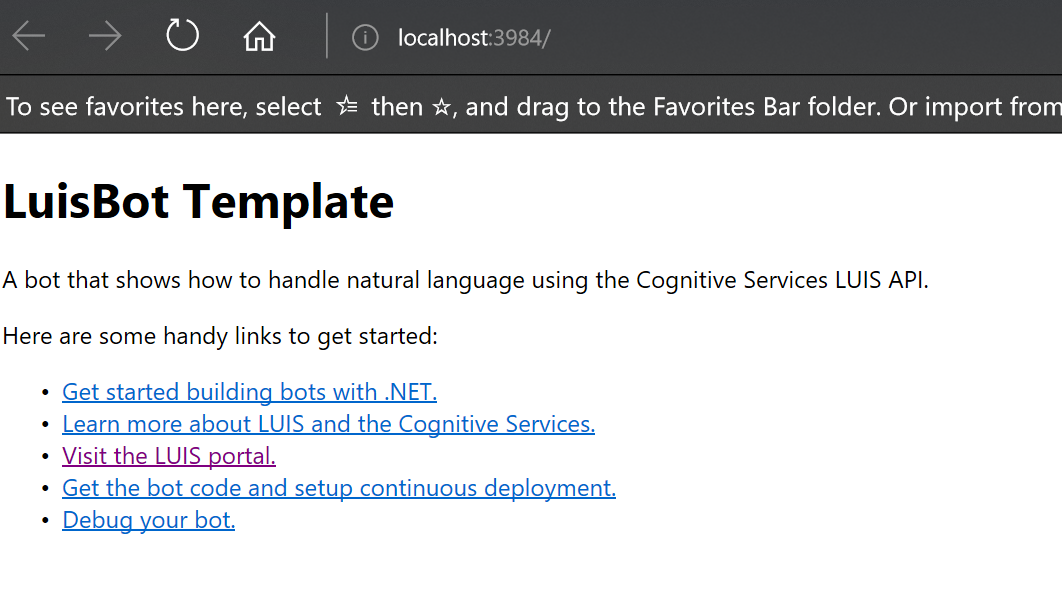
<add key="LuisAPIHostName" value="" />

1. To get the values for the keys, go to the Bot app in your Azure portal.
2. Click on Application Settings
3. Copy the values from the settings page to your web.config keys.

Lets run the application to see if everyting is working.

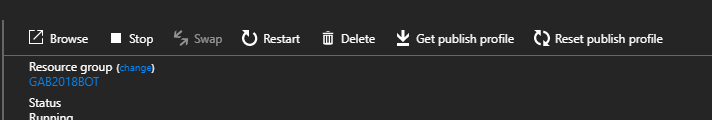
1. Click run
2. If everything goes well your default browser is opened to <http://localhost:3984>

You should see something like this

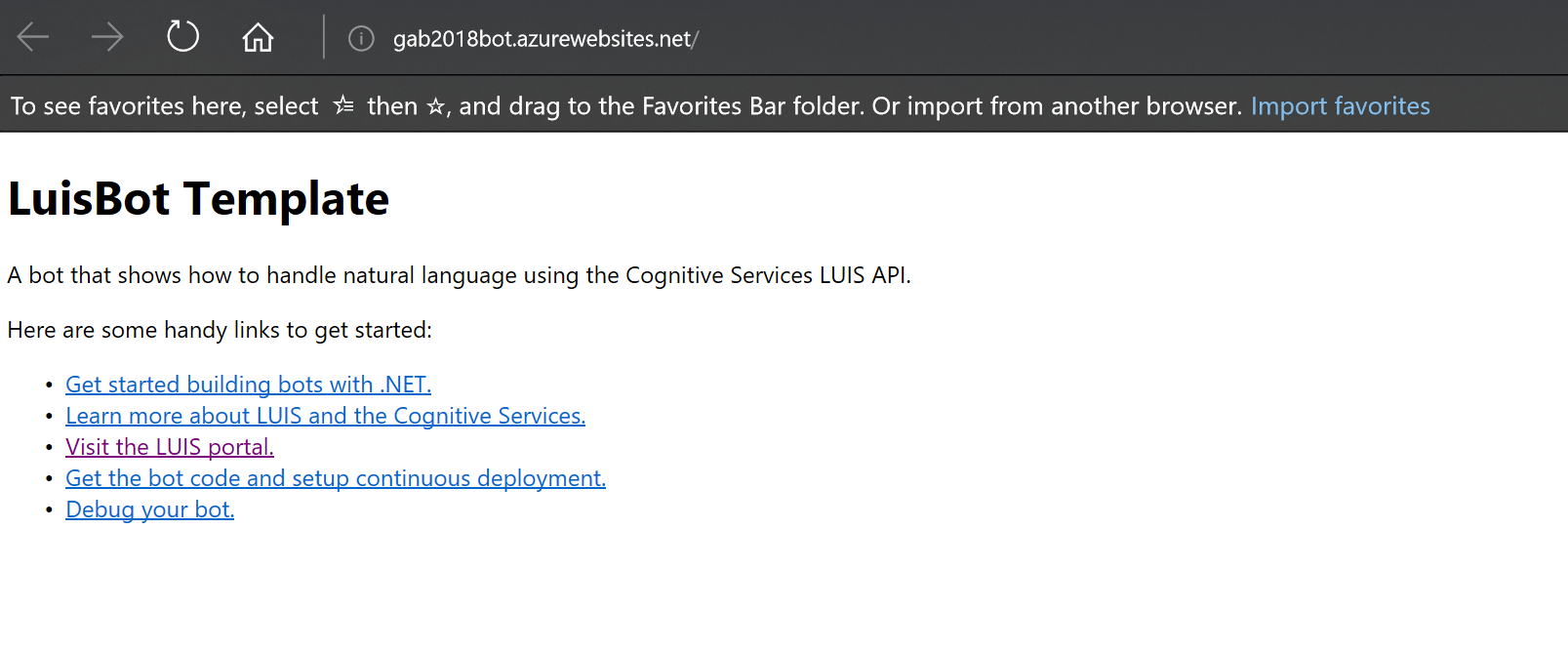


Let’s publish to Azure for a first time. For this you need to get the publish profile from Azure.

1. Go to the Azure portal
2. Select the resource group created for you Bot (GAB2018BOT)
3. Select the App Service (gab2018bot)
4. In the overview you can download “Get publish profile” from the top menu



1. Click Get publish profile to download the file
2. Open Visual Studio
3. Select the project overview
4. Click Publish
5. Below the drop box, select Create new profile
6. Select Import profile
7. Click Publish
8. Select the downloaded file
9. The project should build and publish to Azure and the following website will be opened in a new browser window.



Let’s test the bot

1. Go to the Azure portal
2. Open your bot
3. Click “Test in web chat”
4. Try a sentence in the chat window
5. The respons should be “You have reached <INTENT>. You said: Help”  
   The <INTENT> should be the intent that matches your sentence.

Next we are going to customize the dialog

1. Open “BasicLuisDialog.cs” in the Dialogs folder
2. Study this file and try to understand how LuisDialog works. A dialog specialized to handle intents and entities from LUIS. You can find more info about dialogs in bots here: <https://docs.microsoft.com/en-us/azure/bot-service/nodejs/bot-builder-nodejs-dialog-manage-conversation-flow>
3. Try to add the intent for Warranty by copying one of the other ones.
4. Publish your bot
5. Test it in Azure

You should have added something like this:

[LuisIntent("Warranty")]

public async Task WarrantyIntent(IDialogContext context, LuisResult result)

{

await this.ShowLuisResult(context, result);

}

Lets build on this.

1. In the class BasicLuisDialog, add the following

private bool \_hasWarranty = false;

1. Replace the code you added with the code below.   
   Go through the code line by line and try to understand what it is doing. Finish the missing code to get it working.

[LuisIntent("Warranty")]

[LuisIntent("BoilerAge")]

public async Task WarrantyIntent(IDialogContext context, LuisResult result)

{

if (!\_hasWarranty)

{

\_hasWarranty = true;

<TODO: ADD CODE HERE TO RESPOND WITH "Hello, when did you purchase the device?”>

context.Wait(MessageReceived);

return;

}

if (\_hasWarranty)

{

var boilerAge = result.Entities.FirstOrDefault(x => x.Type == "boilerAge")?.Entity;

var boilerBuildYear = result.Entities.FirstOrDefault(x => x.Type == "boilerBuildYear")?.Entity;

int age;

if (!int.TryParse(boilerAge, out age))

{

await context.PostAsync("I'm a very simple bot, I only understand numbers... Please improve me!!!");

return;

}

< ADD CODE HERE TO RESPONG WITH "Good news, your device has warranty!" IF THE AGE IS 2 YEARS OR LES, ELSE RESPOND WITH "We are sorry, but you warranty has expired. The warranty is 2 years.">

}

}

1. Publish your bot
2. Test your bot with the following dialog:
   1. You: Do I have warranty
   2. Bot: Hello, when did you purchase the device
   3. You: 5 years ago
   4. Bot: We are sorry, but you warranty has expired. The warranty is 2 years.

This is the end of this LAB. Do try to play with your bot.

* Try more sentences
* Add sentences to the LUIS app and retrain and publish it
* Add more intents you can use in the dialog, for example ask what color the boiler has

You can find the complete project solution in GIT as part of this lab.

# LAB 2: Bonus step

You are a senior developer and found this easy as pie and have some time to the next LAB. Try these steps on your own:

* Connect your bot to the Skype channel and test it using skype.