# Calling the service

* The first thing we need is the information to call the prediction service. We can find this in the Custom Vision Portal at <https://www.luis.ai> with our project open. In the Manage tab with the Azure Resources section selected, get the following information:
  + The Endpoint URL for the Azure prediction resource
  + The primary key for the Azure prediction resource
  + The endpoint URL for the Azure authoring resource
  + The primary key for the Azure authoring resource
  + The Application Id under the Application Information section
* In Visual Studio open the Lab4 project in the project folders.
* Open the Constants\AzureML.cs file. Paste the four items collected above into the following variables respectively:
  + PredictionAPIKey
  + PredictionEndpoint
  + AuthoringAPIKey
  + AuthoringEndpoint
  + AppId
* Also, in the Constants\AzureML.cs file set the AuthoringVersion to “Current” and Environment to “Staging”.
* Ensure the following Nuget packages have been added:
  + Microsoft.Azure.CognativeServices.Language.Luis.Runtime
  + Microsoft.Azure.CognativeServices.Language.Luis.Authoring
  + Netwonsoft.Json
* Implement the following code in the GetEvaluationResult method of the HomeController class. This takes in a string for the location of an image and returns information from our model.

// Create a client object to communicate with our service

var credentials = new Runtime.ApiKeyServiceClientCredentials(AzureML.PredictionAPIKey);

var luisClient = new Runtime.LUISRuntimeClient(credentials, new System.Net.Http.DelegatingHandler[] { });

luisClient.Endpoint = AzureML.PredictionEndpoint;

// Create a request object

var predictionRequest = new RuntimeModels.PredictionRequest

{

Query = checkText

};

// Call the service to get the intent information

var result = await Runtime.PredictionOperationsExtensions.GetSlotPredictionAsync(luisClient.Prediction,

AzureML.AppId,

AzureML.Environment,

predictionRequest);

// Convert the results to a JSON string

var jsonResults = JsonConvert.SerializeObject(result);

// return back our model

return new IndexModel

{

TextToSearch = checkText,

ResultJson = jsonResults

};

* Try running the project, enter some text into the top textbox and press the submit button

Notice that the returned JSON has a topIntent property. If you search for “I want to book a car” notice that it not only identifies the intent but also is able to pick out the reservation type.

# Reinforcement Learning

What if it incorrectly identifies the intent? For example, type in, “Do I have any car reservations?” It might return that the intent is a “Create Booking” intent. How can a user tell us this was wrong? We are going to have a way for the user to say a message should have been part of the “None” intent.

* Open up the Lab4 project.
* Implement the following code in the AddToNoneUtterance method of the HomeController class. This takes in a string for utterance that is being used and tells our service that it is part of the “None” intent.

// Create a client object to communicate with our service

var credentials = new Authoring.ApiKeyServiceClientCredentials(AzureML.AuthoringAPIKey);

var luisClient = new Authoring.LUISAuthoringClient(credentials, new System.Net.Http.DelegatingHandler[] { });

luisClient.Endpoint = AzureML.AuthoringEndpoint;

// Add a new example for an intent using our utterance

// Notice we define the intent by text name

var newExample = new AuthoringModels.ExampleLabelObject()

{

IntentName = "None",

Text = checkText

};

// Add this utterance as an example of the none intent

var result = Authoring.ExamplesExtensions.AddAsync(luisClient.Examples, AzureML.AppId, AzureML.AuthoringVersion, newExample);

//Return back a model for display

return new IndexModel

{

TextToSearch = checkText,

ResultJson = resultJson

};

* Run the project and use the “Do I have any car reservations?” utterance again.
* Check the LUIS portal to see if your utterance is an example under the None intent

If you try to classify the phrase again what is the result? If it still didn’t show up as “None” what would you need to do?