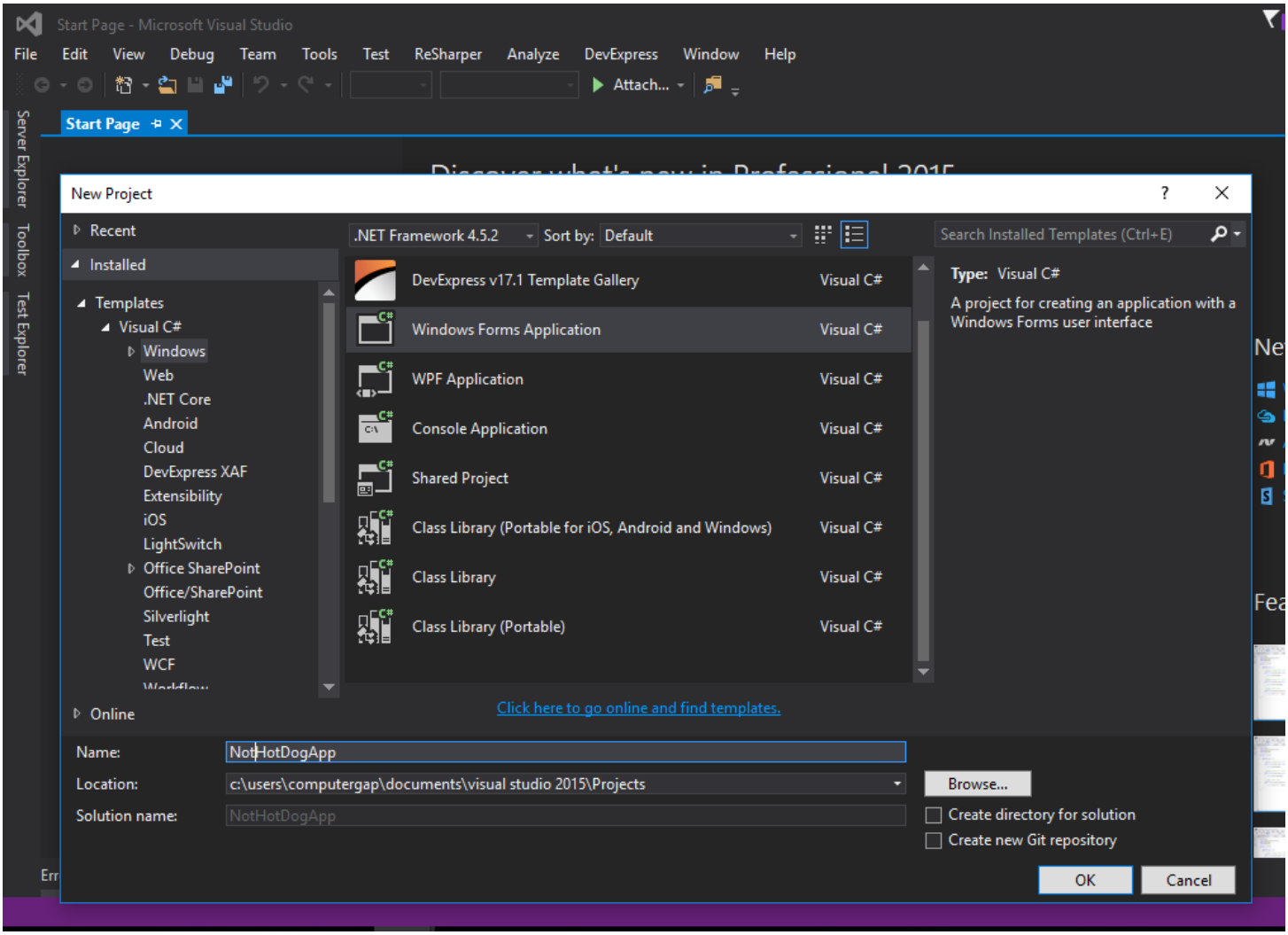
Ejemplo de Servicios Cognitivos – Computer Vision API

Hace unos días viendo un programa de televisión, uno de los protagonistas presento una aplicación que se denominaba “Not Hot Dog” y el concepto era bastante simple. Se tomaba una imagen y la app indicaba si era un perro caliente o no. Aparte de lo gracioso de las escenas respecto a esa aplicación, a mí me quedo la inquietud de cómo se podría hacer algo así. Investigando un poco encontré que Microsoft ofrece un conjunto de librerías que ayudan a realizar análisis sobre imágenes y que resulta relativamente sencillo realizar un app estilo “Not hot dog”. Hagamos un ejemplo.

1. **Creemos una aplicación de escritorio en Visual Studio 2015**

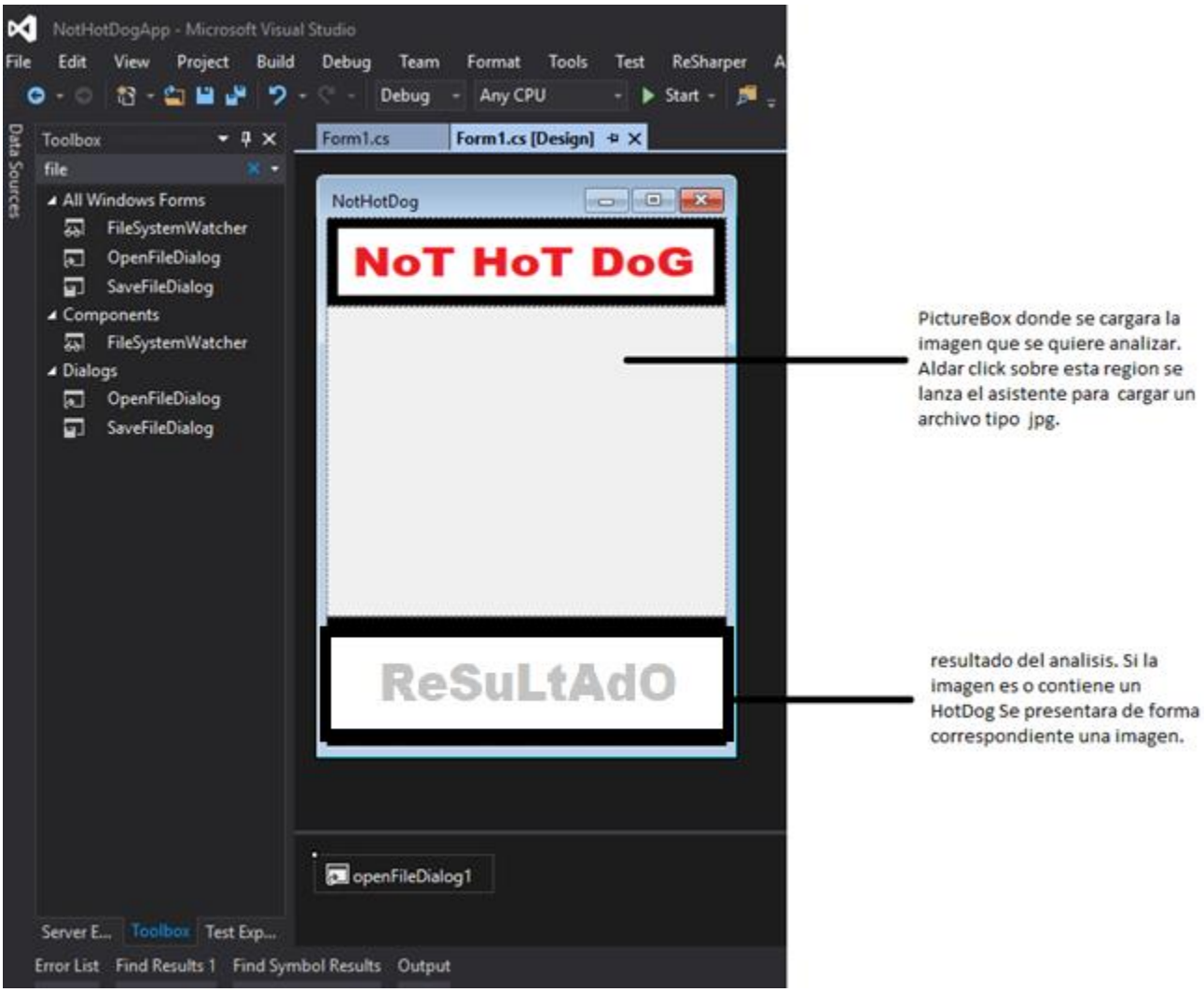
Nota: El código completo se puede bajar de GitHub (https://github.com/lisandro777/NotHotDogExample)

Para el caso del ejemplo mi aplicación la denomine **NotHotDogApp** y será una aplicación tipo **Windows Forms** que se escribirá usando C#.

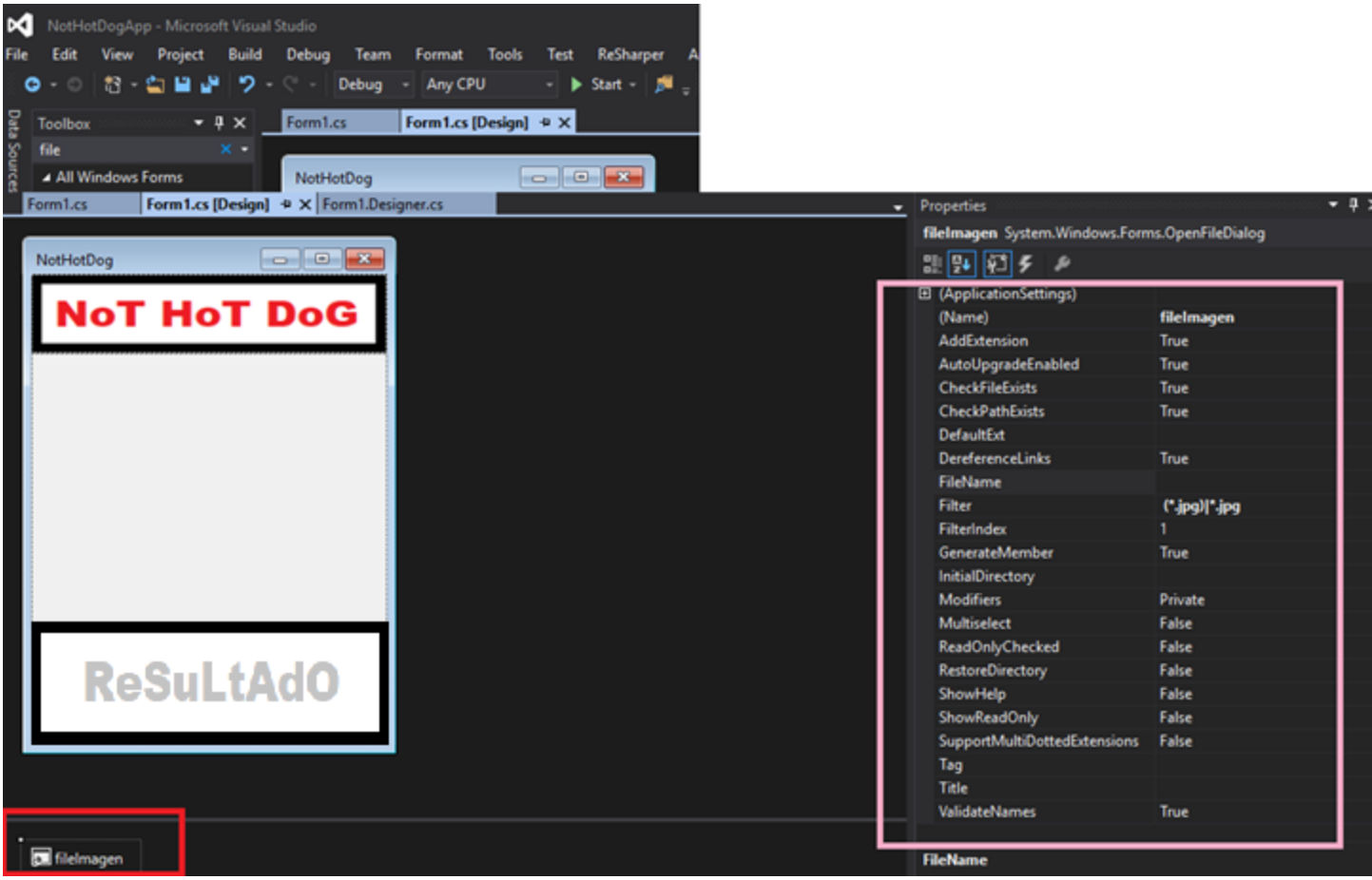


2. **Construir un formulario que permita cargar una foto formato** “.jpg” (es una restricción del API que se va a usar) y que después de analizarla indique si contiene un Hot Dog.

   En mi caso cree un formulario simple, con tres paneles y dos controles tipo PicturBox. Uno para mostrar la imagen que se desea analizar y otro para presentar una imagen como resultado.



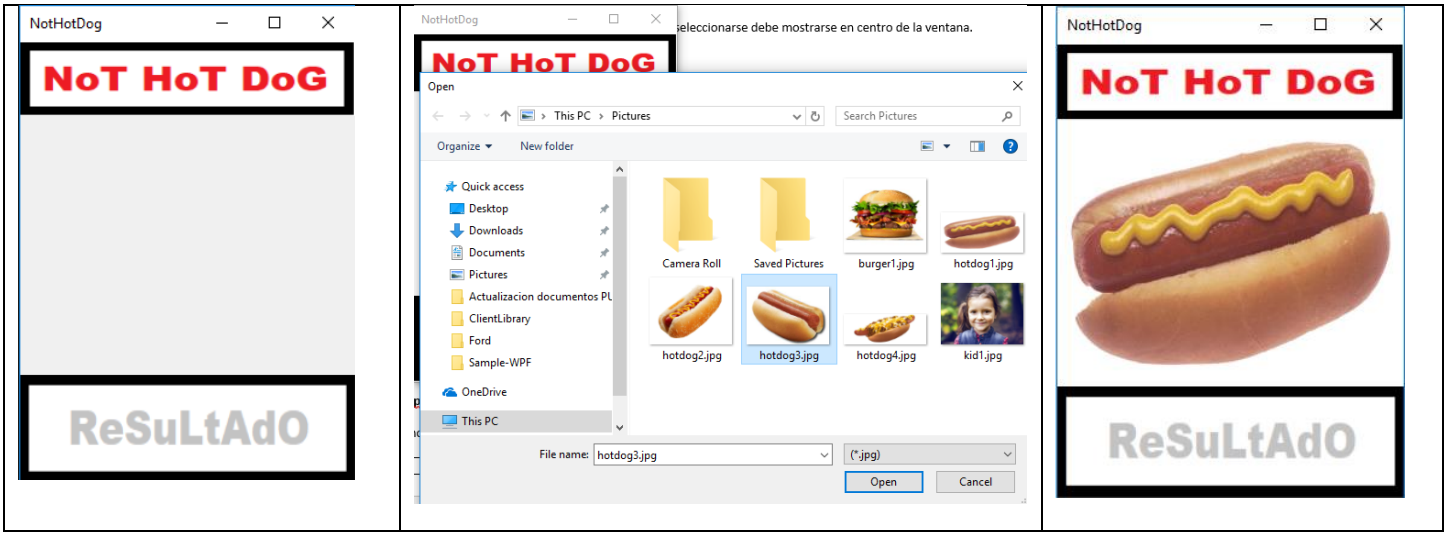
El control tipo ***Open File Dialog*** tiene la siguiente parametrización



Para cargar la imagen que se quiere analizar se programa el evento “Click” del PictureBox que carga la imagen. Al dar click se hace uso de un control tipo *Open File Dialog* para seleccionar el archivo desde una ubicación en el equipo.

```
private void picImagen_Click(object sender, EventArgs e)

      {

          fileImagen.ShowDialog();


          var ruta = fileImagen.FileName;


          if (string.IsNullOrWhiteSpace(ruta)) return;


          picImagen.Image = new Bitmap(ruta);

          picImagen.SizeMode = PictureBoxSizeMode.StretchImage;


      }
```

En este punto, al ejecutar la aplicación si se da click debe lanzarse el cuadro de dialogo que permite seleccionar una imagen y al seleccionarse debe mostrarse en centro de la ventana.



Tenemos la aplicación corriendo, pero sin ningún análisis sobre la foto. ¿Cómo sabemos si la foto es un Hot Dog? Hagamos uso de los Servicios Cognitivos de Microsoft Azure.

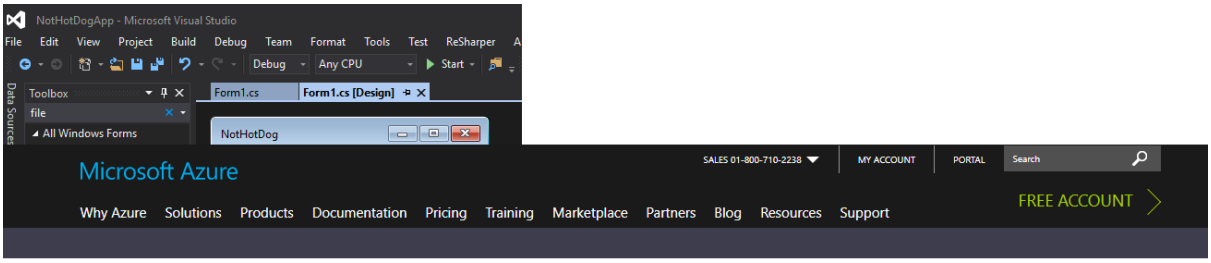
3. **Configurando y haciendo uso de los servicios Cognitivos de Azure (Computer Vision API)**

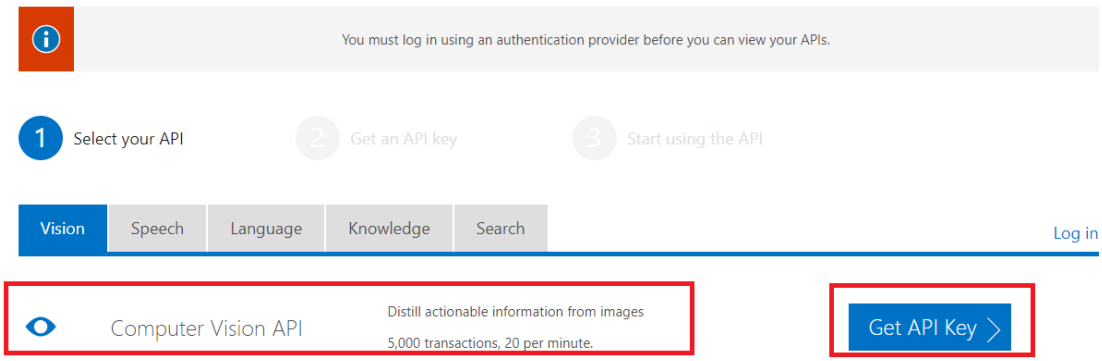
   Para esto necesitamos en primer lugar una cuenta de Azure. Si no tenemos una, podemos crearla de forma gratuita durante un tiempo limitado (30 días).

   https://azure.microsoft.com/en-us/free

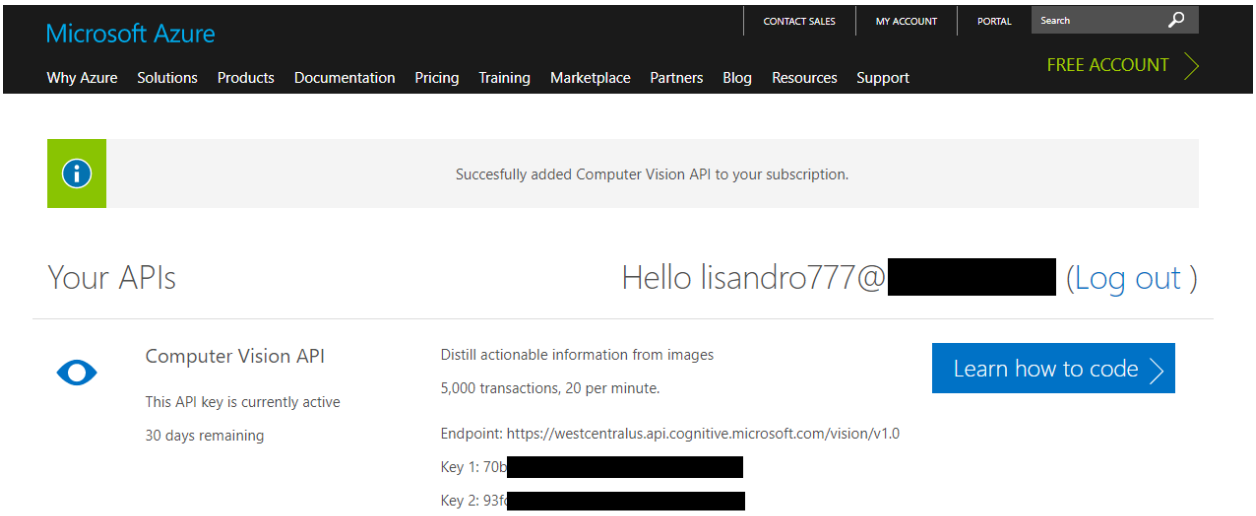
   Después de tener una cuenta y de ingresar a Azure, necesitamos una solicitar una llave (Key) para poder hacer uso del API. Las Llaves se puede solicitar en la siguiente página:

   https://azure.microsoft.com/en-us/try/cognitive-services/



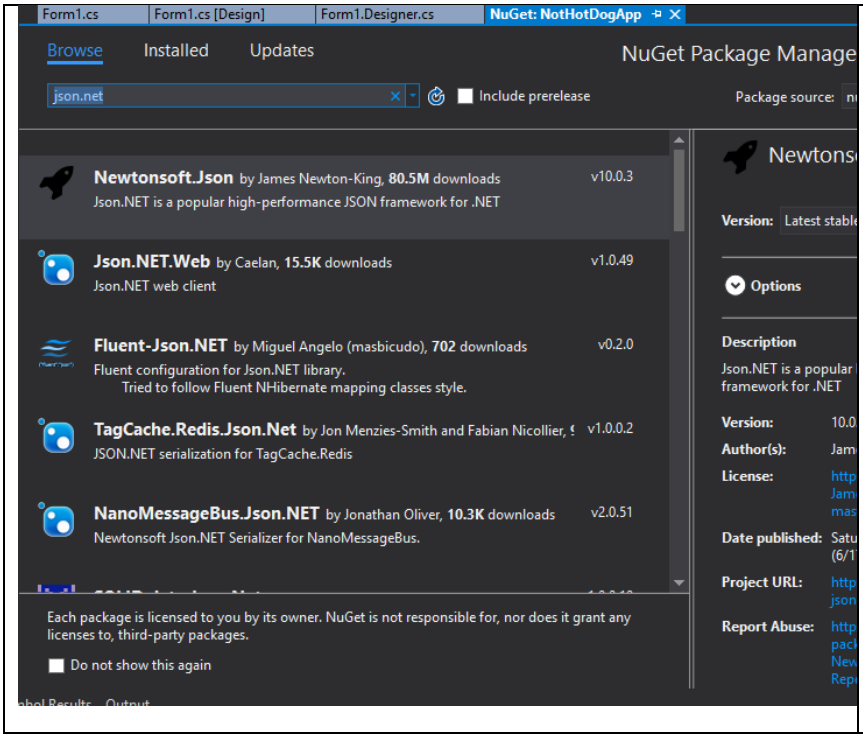


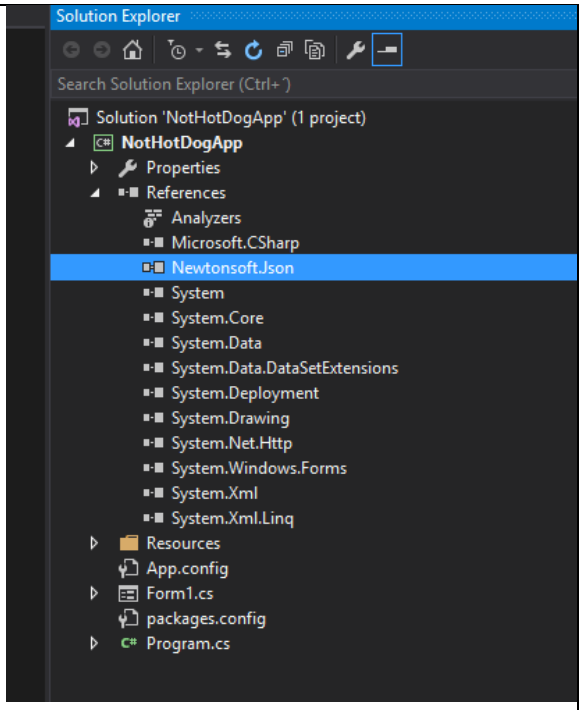
Una vez presionado el botón “Get API Key” y seguir el asistente, Azure entrega una URL y dos llaves para poder usar el API. La pantalla que obtuve en mi caso es la siguiente:



4. Crear un método para realizar el llamado al API.

Antes de hacer el llamado a la página del servicio debemos contar con un mecanismo que nos facilite el procesamiento del resultado a cualquier solicitud. Tengamos en cuenta que la respuesta del servicio de Análisis de imágenes de Microsoft es un JSON con una estructura especifica que veremos más adelante. Para manejar JSON en nuestra aplicación vamos a adicionar la librería Json.Net haciendo uso de Nuget.





a. Creemos un nuevo método llamado ***AnalizarImagen*** este nuevo método debe ser ***async*** (Asíncrono) ya que haremos dentro del método un llamado a un servicio web tipo REST y debemos esperar por la respuesta de este servicio (***await***).

```
private async void AnalizarImagen(string ruta)
{

}
```

b. Declaramos variables auxiliares con la información requerida para acceder al API

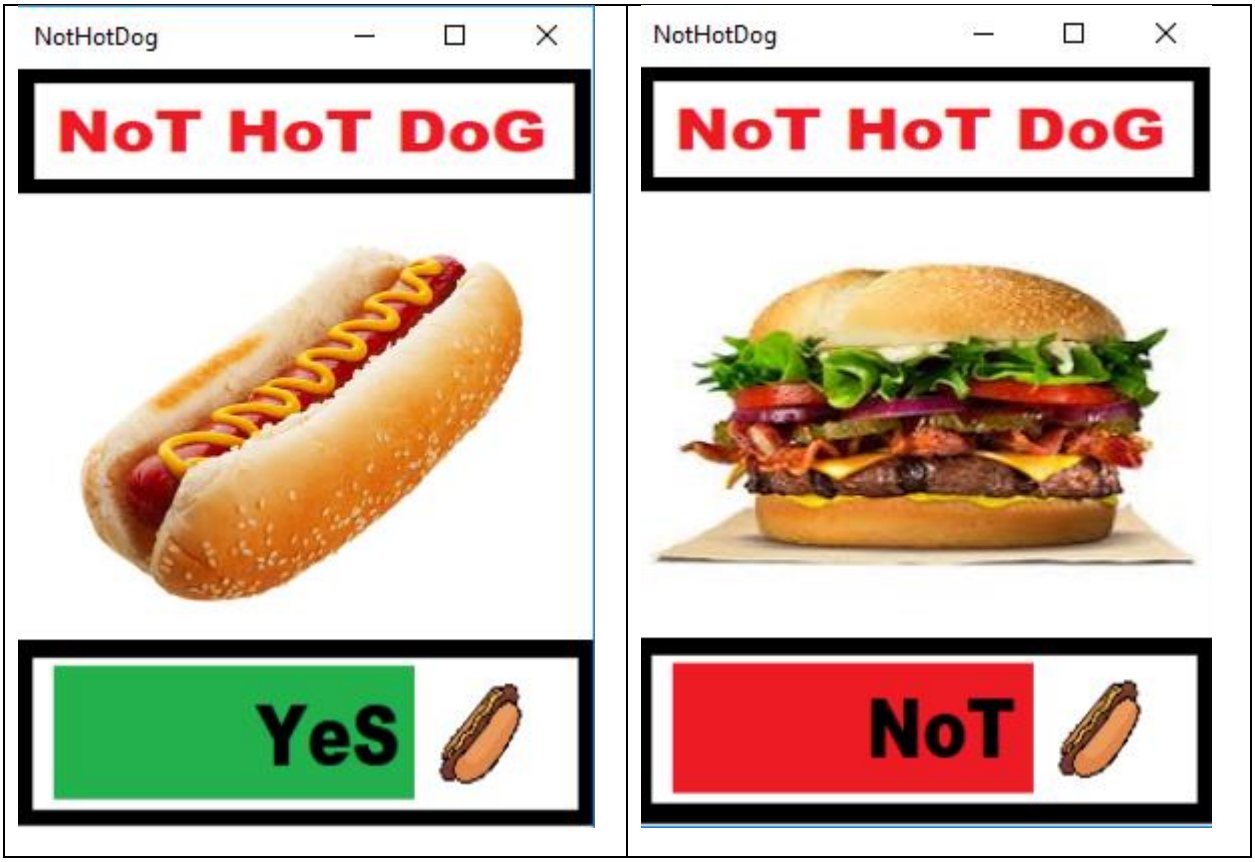
```
private async void AnalizarImagen(string ruta)
{
    //Una de las dos Llaves obtenidas desde la página de Azure
    string llave = "70b9XXXXXXXXXXXXXXXXXX";

    //La url ofrecida por el servicio de Azure.
    string urlServicio = "https://westcentralus.api.cognitive.microsoft.com/vision/v1.0/analyze";

    HttpClient cliente = new HttpClient();

    // Encabezado de la solicitud.
    cliente.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", llave);

    //Parametros que se envian como parte de la URL.
    string requestParameters = string.Format("visualFeatures=Description&subscription-key={0}", llave);

    // Url completa de la solicitud
    string uriCompleta = urlServicio + "?" + requestParameters;
}
```

c. Adicionamos la creación del cuerpo de la solicitud y hacemos el llamado al servicio.

```
private async void AnalizarImagen(string ruta)
{
    //Una de las dos Llaves obtenidas desde la página de Azure
    string llave = "70b9XXXXXXXXXXXXXXXXXX";

    //La url ofrecida por el servicio de Azure.
    string urlServicio = "https://westcentralus.api.cognitive.microsoft.com/vision/v1.0/analyze";

    HttpClient cliente = new HttpClient();

    // Encabezado de la solicitud.
    cliente.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", llave);

    //Parametros que se envian como parte de la URL.
    string requestParameters = string.Format("visualFeatures=Description&subscription-key={0}", llave);

    // Url completa de la solicitud
    string uriCompleta = urlServicio + "?" + requestParameters;


    HttpResponseMessage response;

    // Se convierte la imagen a un conjunto de bits para ser enviado como cuerpo de la solicitud
    FileStream fileStream = new FileStream(ruta, FileMode.Open, FileAccess.Read);
    BinaryReader binaryReader = new BinaryReader(fileStream);
    byte[] bitImagen = binaryReader.ReadBytes((int)fileStream.Length);

    using (ByteArrayContent content = new ByteArrayContent(bitImagen))
    {
        //Se especifica el tipo de contenido de la solicitud, en este caso datos binarios.
        content.Headers.ContentType = new MediaTypeHeaderValue("application/octet-stream");

        // Se ejecuta el  llamado al API
        response = await cliente.PostAsync(uriCompleta, content);
```

```
        // Se obtiene la respuesta.
        string contentString = await response.Content.ReadAsStringAsync();

        dynamic dynJson = JsonConvert.DeserializeObject(contentString);
        foreach (var item in dynJson)
        {
            string valor = ((Newtonsoft.Json.Linq.JContainer)item).First.ToString();

            if (valor.ToUpper().Contains("HOTDOG") || valor.ToUpper().Contains("HOT DOG"))
            {
                this.picResultado.Image = global::NotHotDogApp.Properties.Resources.YesHDResult;
                return;
            }
        }

        this.picResultado.Image = global::NotHotDogApp.Properties.Resources.NotHDResult;
    }

}
```

d. Para determinar si la imagen contiene un Hot Dog, se hace una comparación básica y elemental del contenido del resultado. Existen mejores formas de hacer uso del resultado y de validar la respuesta, sin embargo el objetivo del ejemplo es servir como base para entender el uso de los servicios cognitivos específicamente de análisis de imágenes. Al ejecutar la aplicación deberíamos ver el resultado de la validación



e. Si necesitamos obtener más información relacionada a la Imagen que estamos analizando, lo único que tenemos que hacer es indicarlo en los parámetros de la solicitud. Actualmente solamente estamos usando el parámetro "***Description***" sin embargo tenemos a disposición más parámetros:

https://westcentralus.api.cognitive.microsoft.com/vision/v1.0/analyze?visualFeatures=Adult,Categories,Color,Description,Faces,ImageType,Tags

Como vimos, usando los APIS disponibles de Azure para análisis de imágenes resulta ser muy sencillo y practico. Si se requiere más información se puede referir a la siguiente página:

https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/quickstarts/csharp#prerequisites