Implement Azure AI Content Safety → Implementación de código C#

Provision a *Content Safety* resource

If you don't already have one, you'll need to provision a **Content Safety** resource in your Azure subscription.

1. Open the Azure portal at `https://portal.azure.com`, and sign in using the Microsoft account associated with your Azure subscription.
2. Select **Create a resource**.
3. In the search field, search for **Content Safety**. Then, in the results, select **Create** under **Azure AI Content Safety**.
4. Provision the resource using the following settings:
    - **Subscription**: *Your Azure subscription.*
    - **Resource group**: *Choose or create a resource group.*
    - **Region**: Select **East US**
    - **Name**: *Enter a unique name.*
    - **Pricing tier**: Select **F0** (*free*), or **S** (*standard*) if F0 is not available.
5. Select **Review + create**, then select **Create** to provision the resource.
6. Wait for deployment to complete, and then go to the resource.
7. Select **Access Control** in the left navigation bar, then select **+ Add** and **Add role assignment**.
8. Scroll down to choose the **Cognitive Services User** role and select **Next**.
9. Add your account to this role, and then select **Review + assign**.
10. Select **Resource Management** in the left hand navigation bar and select **Keys and Endpoint**. Leave this page open so you can copy the keys later.

Use Azure AI Content Safety Prompt Shields

In this exercise you will use Azure AI Studio to test Content Safety Prompt Shields with two sample inputs. One simulates a user prompt, and the other simulates a document with potentially unsafe text embedded into it.

1. In another browser tab, open the Content Safety page of Azure AI Studio and sign in.
2. Under **Moderate text content** select **Try it out**.
3. On the **Moderate text content** page, under **Azure AI Services** select the Content Safety resource you created earlier.
4. Select **Multiple risk categories in one sentence**. Review the document text for potential issues.
5. Select **Run test** and review the results.
6. Optionally, alter the threshold levels and select **Run test** again.
7. On the left navigation bar, select **Protected material detection for text**.
8. Select **Protected lyrics** and note that these are the lyrics of a published song.
9. Select **Run test** and review the results.
10. On the left navigation bar, select **Moderate image content**.
11. Select **Self-harm content**.
12. Notice that all images are blurred by default in AI Studio. You should also be aware that the sexual content in the samples is very mild.
13. Select **Run test** and review the results.
14. On the left navigation bar, select **Prompt shields**.
15. On the **Prompt shields page**, under **Azure AI Services** select the Content Safety resource you created earlier.
16. Select **Prompt & document attack content**. Review the user prompt and document text for potential issues.
17. Select **Run test**.

Declaro los namespaces que se utilizarán

```
using System;
using System.Linq.Expressions;
using Azure;
using Azure.AI.ContentSafety;
```

Declaro endpoint y key que se utilizarán. Con ellas construye una instancia de ContentSafetyClient

```
string endpoint = "https://lab-content-safety-guilledev.cognitiveservices.azure.com/";
string key = "BGjyydbr7Vt9uLdtoq5D6Muonwf3ZRgNSavsKenfXfCECv44gYq1JQQJ99BBACYeBjFXJ3w3AAAHACOGbDyU";

ContentSafetyClient client = new ContentSafetyClient(new Uri(endpoint), new AzureKeyCredential(key));
```

Declaro y/o inicializo variables y hago el callout para analizar texto e imágen

```
string userText = "Se encontró a un hombre de 51 años muerto en su coche. Había manchas de sangre en el salpicadero y el parabrisas." +
    "En la autopsia, se encontró un corte profundo, oblicuo y largo en la parte frontal del cuello. Resulta que ha muerto por suicidio.";

var request = new AnalyzeTextOptions(userText);

Response<AnalyzeTextResult> response;
Response<AnalyzeImageResult> responseImage;

response = client.AnalyzeText(request);
responseImage = client.AnalyzeImage(new Uri("https://cdn.apollohospitals.com/health-library-prod/2021/09/Self-injurycutting.png"));
```

Muestro por consola los resultados de los niveles del análisis de los tópicos: Hate, Selfharm, Sexual o Violence

```
Console.WriteLine("\nAnalyse text succeeded:");
Console.WriteLine("Hate severity: {0}", response.Value.CategoriesAnalysis.FirstOrDefault(a => a.Category == TextCategory.Hate)?.Severity ?? 0);
Console.WriteLine("SelfHarm severity: {0}", response.Value.CategoriesAnalysis.FirstOrDefault(a => a.Category == TextCategory.SelfHarm)?.Severity ?? 0);
Console.WriteLine("Sexual severity: {0}", response.Value.CategoriesAnalysis.FirstOrDefault(a => a.Category == TextCategory.Sexual)?.Severity ?? 0);
Console.WriteLine("Violence severity: {0}", response.Value.CategoriesAnalysis.FirstOrDefault(a => a.Category == TextCategory.Violence)?.Severity ?? 0);

Console.WriteLine("\nAnalyse image succeeded:");
Console.WriteLine("Hate severity: {0}", responseImage.Value.CategoriesAnalysis.FirstOrDefault(a => a.Category == ImageCategory.Hate)?.Severity ?? 0);
Console.WriteLine("SelfHarm severity: {0}", responseImage.Value.CategoriesAnalysis.FirstOrDefault(a => a.Category == ImageCategory.SelfHarm)?.Severity ?? 0);
Console.WriteLine("Sexual severity: {0}", responseImage.Value.CategoriesAnalysis.FirstOrDefault(a => a.Category == ImageCategory.Sexual)?.Severity ?? 0);
Console.WriteLine("Violence severity: {0}", responseImage.Value.CategoriesAnalysis.FirstOrDefault(a => a.Category == ImageCategory.Violence)?.Severity ?? 0);
```