**Solution explanation.**

As usual, we start creating the object in order to control browser. Secondly, I have use method ***Navigate().GoToUrl*** to browse the URL we want to cover the testing.

Next step is to use an explicit wait to make sure the website is loaded in time before starting to find the element too early, avoiding an exception error.

We do need to create the object for selecting language we would like to translate from/to. *Note: I have the code structure clear, but I am not able to find Id/ClassName/Name for these particular language-menu-button using HTML Inspection.*

This will be a common part for all the following solutions below.

**Use case #1 – Copy & Paste translation from English to Danish**

For writing the text to be translated, I find the element by name and by using method ***SendKeys****,* I type manually a text example for the input text in English.

As the web application translates automatically without clicking in any button, directly next step is, as before, finding the right text box element by name. See in the code the process of copy/paste.

Finally, I would like to check that we copied correctly from our web app so I have created a file to copy translated text.

**Use case #2 – Document translation for a simple Word file (.docx) from English to Danish**

Finding element “mat-button-wrapper” we use method .***SendKeys*** with attribute the path where the file is located. After that due to app design we need to click on *Translate to Danish* and for this action first we need to locate the button and after use method ***.Click***

At the end I used ***Thread.Sleep(5000)*** for giving the tool enough time to translate the doc.

**Use case #3 – Language swap (using the swap button in the middle of the panel)**

For writing the text to be translated, I find the element by name and by using method ***SendKeys****,* I type manually a text example for the input text in English.

As the web application translates automatically without clicking in any button, directly next step is identify the swap button and use method ***.Click***

**How to set up and run the solutions.**

Automated tests are divided in 3 different projects compressed in one .zip

You will only need to change in the code for solution 2 the path where the project is located in your local machine. If you use Visual Studio, directly open the .sln file, make sure ***Selenium.Webdriver*** is added from *Nuget packages* and click on execute.