## Legal Contract Assistant Installation manual

## Add-In installation

1. Create a shared folder on your PC or Network, for example [\\MY-PC\Shared](file:///\\MY-PC\Shared). Make sure this shared folder is accessible by all users intending to use add-in.
2. Copy manifest.xml to the shared folder.
3. In WORD, go to File – Options – “Trust-Center” – “Trust-Center-Settings” – “Trusted Add-In Catalogs”
4. Add a new catalog using network share you have created. Don’t forget to check “Show in Menu” option.

A screenshot of a computer

Description automatically generated

1. Click Ok and restart WORD
2. In WORD menu, click on “Add-ins”, select “More Add-ins” and choose “Shared Folder”
3. Select “Legal Contract Assistant” and click Add.

A screenshot of a computer

Description automatically generated

1. Legal Contract Assistant should appear in the menu.

A screenshot of a computer

Description automatically generated

## Add-In Usage

1. Open Terms of Use\_example.docx
2. Click on “Legal Contract Assistant” add-in. Ignore debugging dialog (click Ok) – we will remove it later.
3. Click Login. Currently it is not connected, but the intend is that only users who registered and have license can use add-in. Read disclaimer please. The purpose of the disclaimer is to get the user’s consent to upload the content of the current file to OpenAI storage. This content is needed to create Open AI Assistant and will be deleted at the end of the session…
4. Choose Excel file with annotations – Excel Example.xlsx
5. The screen will show an annotation map grouped by article. There could be multiple annotations for the same article, and you can choose which one to feed to OpenAI by selecting/unselecting checkboxes.
6. Clicking on the article link will navigate you to the same article in the document and will highlight the header. (IMPORTANT: When you apply OpenAI suggestion by clicking on “Approve and replace” button, the header must be highlighted, since it indicates modified text placement location.)
7. Click on “Suggest changes”. The first time will take approximately 10 seconds, because the system is creating OpenAI assistant in background and asking the first question. Consequently, clicking on “Suggest changes” should be faster.

A screenshot of a computer

Description automatically generated

1. OpenAI showing changes in Yellow and additions in Green
2. Review the changes and click on “Approve and Replace” button. This action will remove the original text from WORD and replace the ENTIRE article with suggestions. This is in HTML format, so highlighted text will also make it to WORD. Replacement action will remove all existing formatting from WORD paragraph. (I think this is the most complex improvement, because users will have to re-apply existing formatting on OpenAI suggestion).

A screenshot of a computer

Description automatically generated

1. You can click on “Suggest Changes” in the same article multiple times – OpenAI will rephrase suggestion

## Excel format

Excel file format, although simple, is very strict. It must have exactly two columns, one called “Article” and another called “Annotation”. There should not be empty cells – for each annotation proper article is required. If multiple annotations exist for the same article, they should be placed one under another. Even more important, Article names in Excel MUST match Articles in WORD document. The only difference that is allowed is casing. For example, Excel article name “Payments and billing” will successfully find WORD article name “Payments and Billing”, but will not be able to find WORD article “Payment and Billing” (missing “s” at the end of Payment)

A screenshot of a computer

Description automatically generated

Although it sounds complicated, but I think it is very natural process to copy articles’ names from Word to Excel – just need a discipline.

## Developer notes

1. This add-in uses native OpenAI API not Azure OpenAI API. The reason is that OpenAI, unlike Azure OpenAI, does not have limits and allows you to spend a lot of tokens. The process of suggesting changes for entire article using the whole document as context is very resource intensive. Likely for us, Open AI still cheap, each session will probably cost approximately 10 cents (although it all depends on the size of document and number of changes).
2. Another reason for using OpenAI is that Microsoft seems to be diverting from OpenAI in favor of Co-Pilot, which seems radically different (at least from a programming standpoint)
3. The major challenge in processing WORD documents is not OpenAI itself, but a vast difference in the way people create WORD documents. For example, “Terms of use example” is not a classic word document at all. It is probably a copy from the Internet, so it does not have numbered lists for article headers, like, for example, I would create. So, before this tool becomes a PROD quality it would require a massive amount of testing and fine turning.
4. Keeping the original paragraph format is another massive challenge, otherwise it creates unfavorable user experience.
5. Distribution of this Add-in is another challenge. If we want to use Microsoft Store, Add-In must be certified for use on multiple platforms and implement various features like Single sign-on. Likely we always have the option to distribute it using manual Manifest (The process I have described it this document), but it will require building our own Logging, Registration and Licensing system. In my opinion, the latter is a better option, because it is easy to understand and gives you ultimate control.
6. We also need to create generic site with description of add-in and its features. On the same site users can register and buy licenses. Currently I am using Syngraphus.com as a domain.