Every tech blog nowadays seems to feature at least some analysis of AI-related topics. This article stems from my personal need for a reliable tool to assist with daily tasks at my desk. The comparison criteria are based on my personal requirements, and the evaluation is entirely empirical and subjective. This means that I actually downloaded and installed each one of mentioned tools and tested them against my own needs.

You might notice that Ollama is not included. While I use Ollama, I prioritized tools with user-friendly UX suitable for semi-technical individuals who prefer to avoid console-based or low-level management.

Here are the key areas I focused on:

- ease of installation and management
- work with locally downloaded models
- ability to process documents
- act as a hub for other different applications
- be stable and have nice and easy UI/UX

If you would want to rephrase that, you would say that I wanted to find software that utilizes most of the model capabilities without going into programming or very advanced configuration. First of all, I am describing the functionalities of those tools and then somewhere near the end of this article you will find a comparison table.

If you would like me to add some aspect to these tools and compare them, or send your tool that you would like to see tested against those, feel free to reach me via contact form under this article.

Testing LLM Software for Running Local LLM

Finding suitable candidates for comparison is challenging. I do imagine that reason is that either they are known by AI developers so they don’t need to ask anyone, or if someone is non-technical then he is just using commercial online versions of those – such as ChatGPT. However, I wanted to tackle topic from “semi-technical” point of view. So it is possible that I missed some good tools – if you think so, contact me and I can add it to comparison.

AnythingLLM – featureful solution for non-technicals

- Website: https://anythingllm.com/
- Docs: https://docs.anythingllm.com/
- Code: https://github.com/Mintplex-Labs/anything-llm
- Version tested: 1.6.9
- Commits: 1k, Stars: 27.4k, Forks: 2.8k
- Organization/Person/Owner: Mintplex Labs Inc.

LM studio – developer friendly AI environment

- Website: https://lmstudio.ai/
- Docs: https://lmstudio.ai/docs
- Code: https://github.com/lmstudio-ai
- Version tested: 0.3.5
- Commits (main repo only): 0.1k, Stars: 1.7k, Forks: 0.1k
- Organization/Person/Owner: Element Labs, Inc.

Jan – KISS is the best rule

- Website: https://jan.ai/
- Docs: https://jan.ai/docs
- Code: https://github.com/janhq/jan
- Version tested: 0.5.8
- Commits: 3.8k, Stars: 23.6k, Forks: 1.4k
- Organization/Person/Owner: Homebrew Computer Company

GPT4All – simple, yet effective solution

- Website: https://GPT4All.io/
- Docs: https://docs.GPT4All.io/
- Code: https://github.com/nomic-ai/GPT4All
- Version tested: 3.4.2
- Commits: 2.1k, Stars: 70.7k, Forks: 7.7k
- Organization/Person/Owner: Nomic, Inc

OpenWebUI – familiar UI, unknown capabilities

- Website: https://openwebui.com/
- Docs: https://docs.openwebui.com/

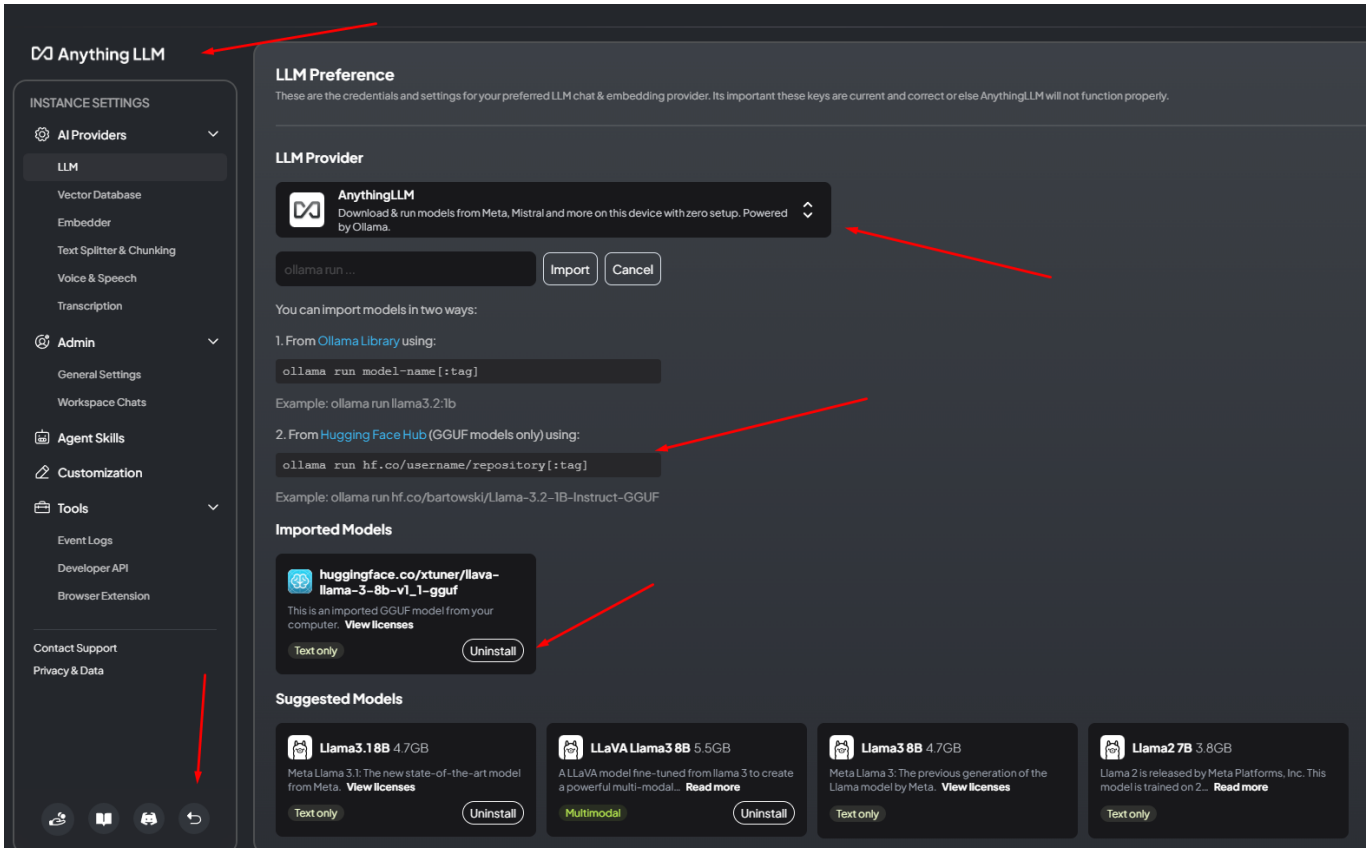
- Code: https://github.com/open-webui/open-webui
- Version tested: 0.3.35
- Commits: 7.2k, Stars: 47.6k, Forks: 5.8k
- Organization/Person/Owner: Tim J. Baek (see https://docs.openwebui.com/team)

AI Model Management and Download Interface

First thing you need to do is to install UI – which is straightforward for all interfaces except OpenWebUI. Next thing you need to do is to download AI model to local machine so you can actually run it. And later on you should of course be able to manage downloaded models with ease.

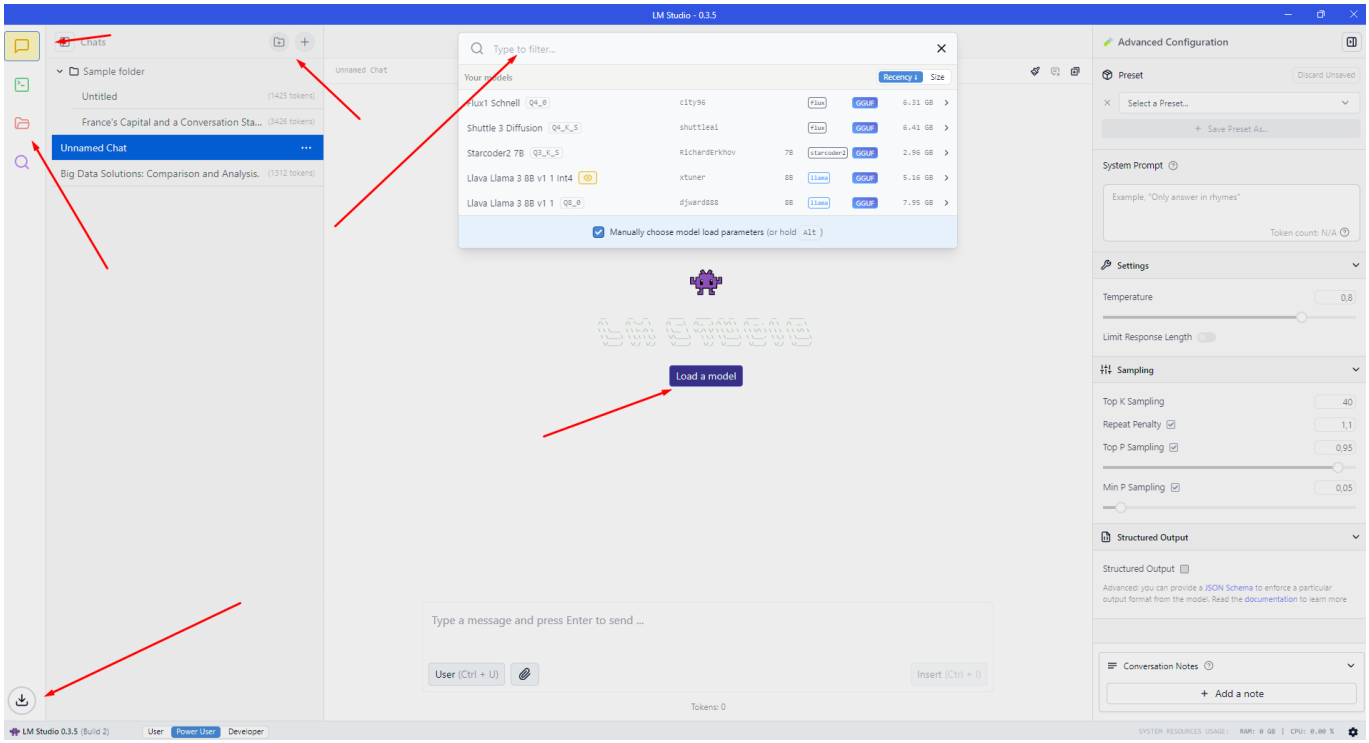
AnythingLLM

With **AnythingLLM** first thing you need to do is to actually find where settings are – there are at the left bottom of the software. Next, navigate to “All Providers” and select one from the dropdown menu or download a suggested model via the console. In the drop down you will notice a lot of different providers so it is nice that they support a lot of them. This not only includes HuggingFace, but also locally run Ollama and different commercial providers such as OpenAI, Azure, Antropic and so on. There is no “Search for models” option inside program, so you need to search them online and then run a command from **AnythingLLM** interface to download it. The interface won’t help you with different model sizes or filtering them, but after you download it you will see if model is text only or multi-modal. Then you can simply go back and open new thread, which can be organized in folders and each folder can have different LLM provider settings. Such organization is quite useful if you want to test different models (or model settings) against same queries.



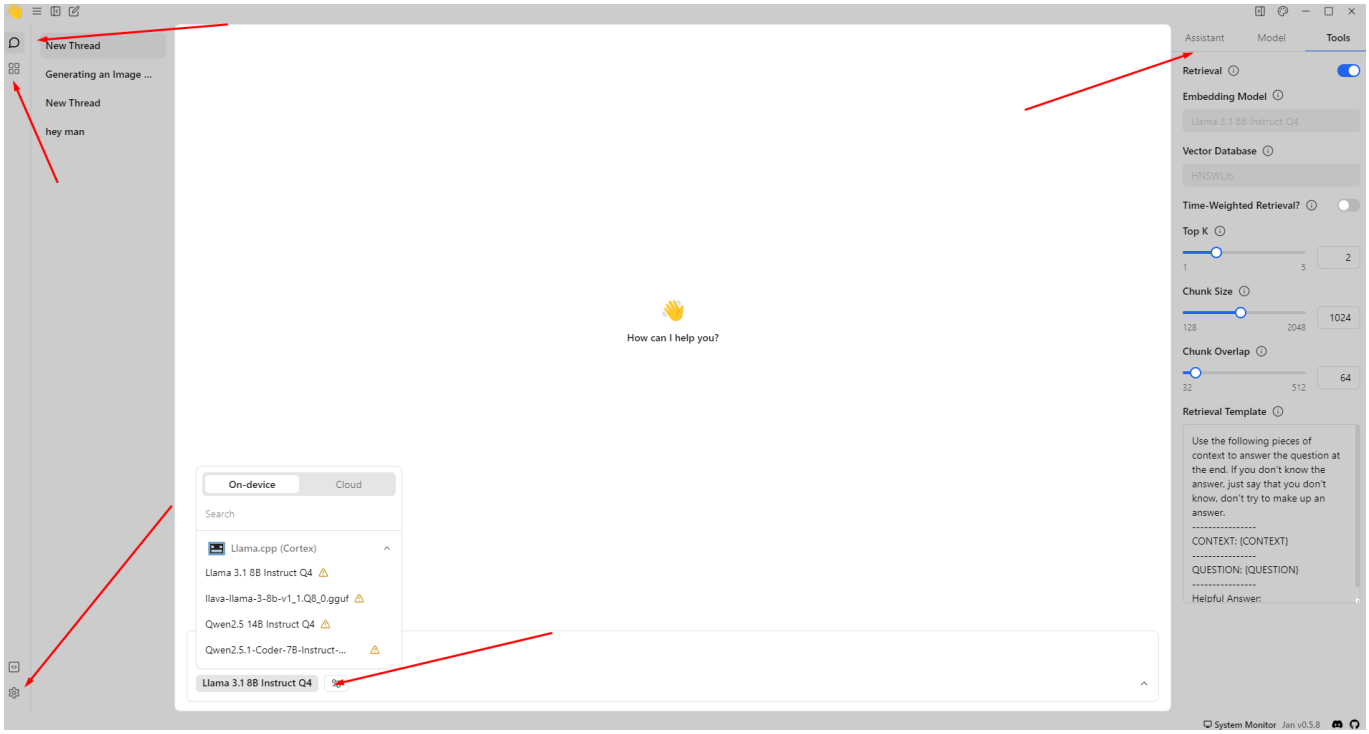
LM Studio

With **LM Studio** it is not obvious where to download models aswell. On the left hand you will have for icons of which third one is “My Models” but you won’t find it there. You need to type something in search input at the top search bar. Then you will see that new window opens where you can actually see results from HuggingFace and find different models. That window allows you to select model version or size which is very useful if repository have different versions of the same LLM. When you go back to “Chat”, you will notice that you can organize chats in folders as you can in AnythingLLM.



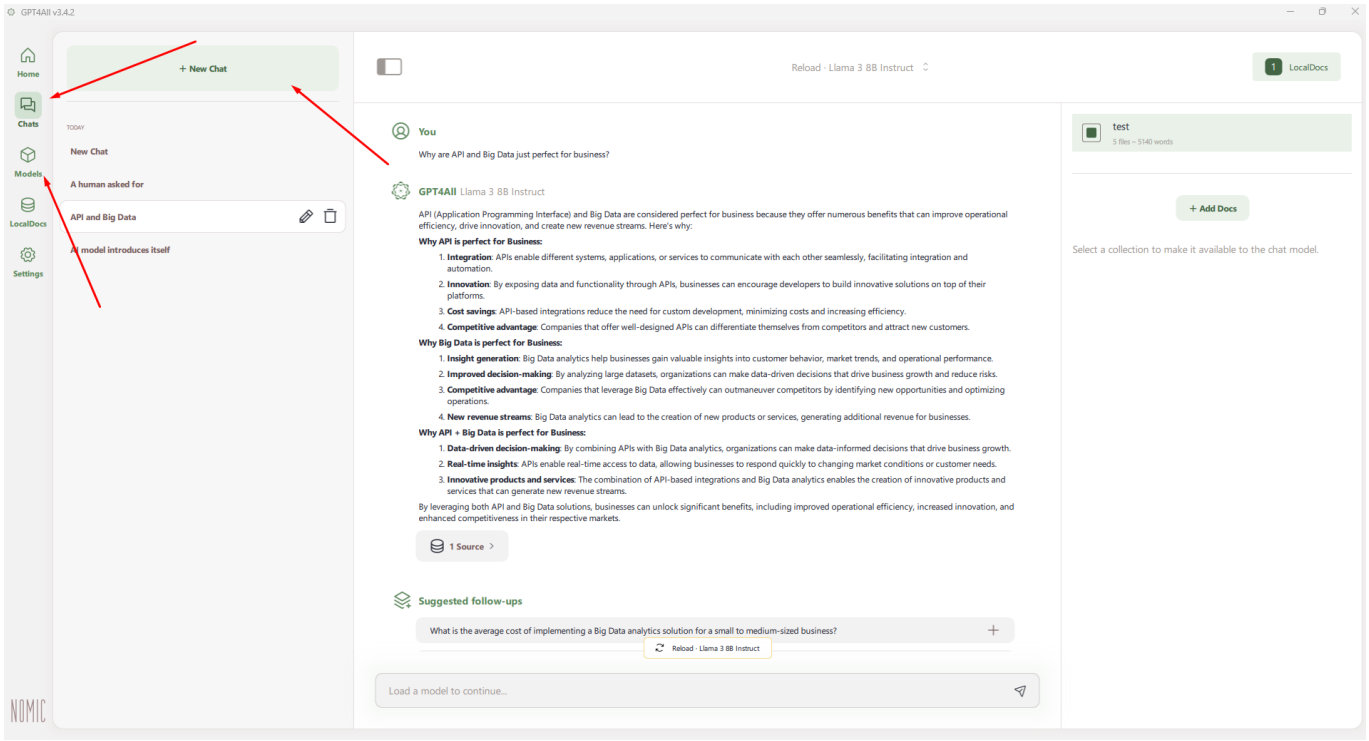
Jan

When using **Jan**, your first action should be navigating to the Hub. In the Hub you will have different models which are tagged if they can be slow on your device or inoperative or if you should be fine to use them. This is very nice and useful feature especially if you are a newbie. That helps you to determine which models you actually want to try out locally. In the settings at the very bottom, you can see that you are able to connect to many different commercial providers aswell, such as: Anthropic, MistralAI, OpenAI and so on. After downloading model, if you go back to chats, you will notice that you cannot organize chats in folders. Although interface is intended to be very easy so I understand this UI decision.



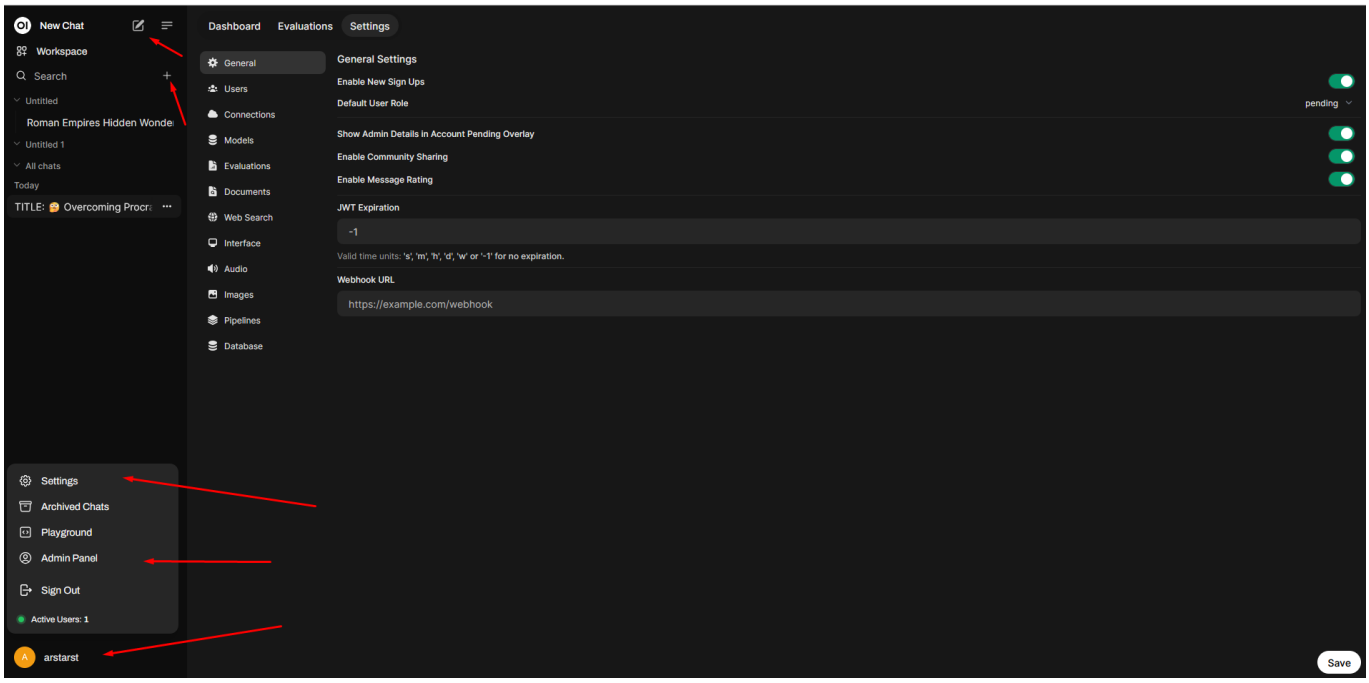
GPT4All

With **GPT4All** first thing you want to do is to go to models on the left where you can find model you would want to download. It has also very useful interface because you get parameters, such as file size, RAM required, parameters, quantization and type. You will get at the beginning propositions of the models but you can use search function to find anything from HuggingFace. After downloading model you obviously go back to chats where you can start your conversation. Same as in Jan you won’t get folders, only threads where you can use specific models.



OpenWebUI

With all of the above options **OpenWebUI** is only exception where you won't get installer. Instead, you need to install Python and then serve application. This is a major drawback if you are a non-console person. Although I decided to include it because of functionalities available, and because installation is actually just running two commands from console. By default **OpenWebUI** is using Ollama as a source for models and you are not getting any interface for searching and downloading models from admin panel. For connecting commercial services, you will need to enter OpenAI compatible API URL and key. At the start, you will be prompted to create an account – since it is a local account, you can use dummy details without concern (I am aware that you can disable this when using Docker installation). You can organize conversations in folders and chats and UI is very ChatGPT-like so such combination would be easiest choice if you would want to run **OpenWebUI** online.



Processing Files Inside Local AI Chat

Ability to analyze files that I will provide to model is one of the most important things that I require from interface. Without it, I could actually just use command line with Ollama instead of installing full interface. I aimed to verify whether the interface could process uploaded documents, pass them to the model, and return meaningful information. While essentially for simple text files it was MOSTLY not an issue, it is worth noting that I was looking for images as well (for example for OCR purposes) and voice transcription. None of the tools was able to actually transcribe audio out of the box: AnythingLLM couldn’t download Whisper model, Jan and GPT4All couldn’t upload such files, LM Studio was always interpreting input as text. For OpenWebUI there exists Whisper tool, but I was not able to make it work – although there is open issue to support it natively: https://github.com/open-webui/open-webui/issues/

You should be aware that the software I tested is not designed to create graphics, but mostly for LLM using GUFF format (graphic creation LLMs are most commonly using “safetensors” format) – if you are looking for UI that is able to create images, you would want to look for example at ComfyUI, which is not analyzed in this article, because it is not UI for simple LLM use. I know that you can connect OpenWebUI

with ComfyUI, but again this would be outside scope of this comparison.

One of the things that are advertised about **AnythingLLM** is that it allows you to process many different document types. The problem is – at least in my case – that **AnythingLLM** likes to hang. It stops processing the document if it encounters any error and does not give you information about what happened, but instead you need to restart whole application to get it working again. Additionally, keep in mind that any uploaded document is added to a collection that **AnythingLLM** can access for future chats. I found it sometimes counter intuitive, but I do understand RAG logic in this. After selecting proper model it was able to properly describe images uploaded as well.

With **LM Studio**, I did not have any of such issues. Anything that I provided to **LM Studio** was processed properly (or understandable error was displayed). Upon uploading, you will get a notification that document will be added to RAG – but RAG is a topic I will cover later on. This software also processed images I uploaded without any problems and allowed LLM to describe what was on the image.

With **Jan** I had problems with uploading documents, but it is mentioned on the website that this functionality is experimental. Essentially, there is an issue on GitHub in progress (I do suspect that it is connected with problems I had – https://github.com/janhq/jan/issues/4023), so I would expect it to be fixed in next version. But at the moment of writing article document interpretation does not work at all. Of course because of this I couldn't test image uploading and interpretation.

GPT4All was to be honest the strangest case of uploading document functionality of them all. It only allows you to upload XLSX files. I have no idea why is that and why I cannot upload TXT files or DOCX files or actually anything else. This is a major drawback for this software. As a result, I could not test its ability to pass images to the LLM and receive a proper response.

I am not sure why, but for **OpenWebUI** uploading document was slowest one but documents were interpreted correctly. Since by running LLM locally I do agree that it sometimes can work slowly, so this is an acceptable case for me. It processed images without any problems or unnecessary waiting.

Using RAG, knowledge, databases and external information

Comparing Documents with Locally Hosted LLMs

One thing that I wanted to try was to force AI to compare five to ten different documents I provided to them. But essentially, it rendered impossible in all cases I have tested, probably because of that all documents were either treated as a context or added to RAG. With context-treatment I just hit context limits of LLMs, and with RAG not whole documents were considered, but only parts of them. I do understand that this is possible with some LLMs, especially commercial ones, but for the sake of this article I will just omit this aspect in comparison, because I was not able to achieve this aim with considered tools. Although, this opens up topic: how tested interfaces utilize RAG abilities?

To explain this aspect better, I must first describe what is RAG in the context of Large Language Models (LLMs), like GPT (Generative Pre-trained Transformer). “RAG” stands for Retrieval-Augmented Generation. RAG is a methodology that combines the capabilities of pre-trained language models with retrieval mechanisms to enhance the generation of responses. This model architecture helps improve the relevance and factual accuracy of the outputs produced by LLMs.

How RAG Works

- **Retrieval Mechanism:** When a query or a prompt is presented, the RAG system retrieves relevant documents or pieces of information from a large corpus or database.
- **Augmentation:** The retrieved documents are then used to augment the prompt, providing additional context or directly relevant information to the language model.
- **Generation:** The augmented prompt is fed into the transformer-based language model, which generates a response based on both the original query and the newly retrieved information.

Benefits of RAG

- **Improved Accuracy:** By fetching relevant external information, RAG can produce more accurate and contextually appropriate responses.

- **Knowledge-based Responses:** It allows the model to provide answers based on a broader knowledge base beyond what it was initially trained on.
- **Flexibility:** The combination of retrieval and generation offers a balanced approach that can be crucial for applications requiring high-quality, reliable outputs.

The conclusion would be that if RAG functionality is available, I might as well try to use it. So below you will find comparison of what each of considered tools is capable in context of using RAGs, knowledge bases, searching information online that could extend knowledge, connecting to external data sources such as MySQL/MariaDB, GitHub/GitLab, Postgres, and so on.

Of course using RAG systems and AI agents is a perfect case for integrations with your SaaS business. See some options below:

LLM Tools in Context of Using External Information

AnythingLLM utilizes feature called “Agent Skills” to connect to different data sources. Essentially, in chat you call “@agent” and provide command that agent should conduct (those skills are not used by default in the chat if you just call LLM, but if you call agent directly it works like a charm). This can be either considered advantage or disadvantage, depending on how you rather to use this feature. For the agent skills you have available features such as generating and saving files, generating charts, web search using DuckDuckGo or a Google search or Bing or Serply (and more), and SQL connector which allows to connect to MySQL or Postgres or SQL Server. I must say, I was not able to properly run generating charts, but web search worked flawlessly. I had a little bit of problems with SQL connector, but I was able to get it to work. However, interpreting data from the database did not work as expected, likely because the database was treated as contextual data that you can pick elements from, not as a “monolith” that can return exact results. Possibly using another model (or different temperature settings) would produce better outcomes, but essentially conclusion is that **AnythingLLM** connected to the database and used it as knowledge source. One more thing about documents: when you upload document in the chat, it is added to document management system, where you can pull it right or left, depending if you want to use this document in current context. If you want the document to be interpreted precisely in the chat, you also should use a pin. But even developers are saying that this can produce, but does not have to produce expected results (for more information visit

https://docs.useanything.com/llm-not-using-my-docs). Besides that it is fairly easy and intuitive document management system. It is also worth noting that this is the only tool considered that provides you natively (when you go to the “Documents” and “Data connectors” tab within RAG) with ability to connect to GitHub repository, GitLab repository, YouTube transcripts, link scrapper and Confluence.

For **LM Studio**, you have this functionality, I would say, partially available – in every chat you can upload documents that would be used within this chat, but there is no central point where you could manage all the documents or all connections. I haven’t found also any other capabilities of connecting to external data sources such as SQL databases or search engines. This could probably be added by using developer tab, but since I would want to use tool as semi-technical user, writing custom script would not be an acceptable solution at this moment.

I have not found any document management system in **Jan** as well. Maybe because they want to implement it later on by using plugin system, although at this moment it is non existing. There is also no ability to connect to search engines or databases.

GPT4All covers well RAG functionality – they have pretty good system of uploading and managing documents within knowledge base, and within chat you can select which datasets you want to use. But there are no possibilities to connect to external databases or search engines, so you can only work on text-based documents.

With **OpenWebUI** you have actually pretty advanced settings for the documents processing in context of the specific chat. I think it provides most settings for document processing of all of the compared interfaces. Web search is also available – you need to mark checkbox inside admin panel, select “web search” and select engine. Then use “+” when creating new prompt to get it working. I had some problems with DuckDuckGo by default, but other services worked properly. In terms of using other data connectors that I was interested in, **OpenWebUI** does not provide them. But when you go to the workspace at the top, you have tabs, models, knowledge, prompts, tools and functions. And when you click discover a tool or any other prompt, you will be redirected to the page where you can download plugins made by community. Although I have not found plugins that I needed (Git and SQL) so I would treat this case same as in LM Studio – if coding is needed, then it’s not provided. But there are other plugins in there that you can find useful – and for the easy installation I do recommend this link
https://docs.openwebui.com/features/plugin/

Useful Integrations and OpenAI Compatible API

Here I would like to compare what other capabilities does each user interface provide. There is not much to add in here, but there are few surprising elements that might be for some people a game changer. I was trying to focus on integration with web browser, Visual Studio Code and Office package and Thunderbird.

For any of those integrations you either need to connect via OpenAI compatible API provided by the interface or direct connection to Ollama service. And while every tool that I tested provides OpenAI compatible API there is a problem on another side of that equation – not many plugins even try to utilize this API and if I found they tried, they mostly did not work (for example I saw in Jan logs that calls were made but were invalid and non-OpenAI API compatible). So while OpenAI API implementations do not work (which was the solution I wanted), we can still use Ollama connection – for now only solution that works without extra complications.

As said, every interface I tested provides OpenAI compatible API. In this aspect AnythingLLM provides non-standard endpoint (but it is worth noting their API provides additional endpoints that you can use to connect with AnythingLLM service), LM Studio is doing just good work, Jan has log viewer that I liked most of all, and GPT4All and OpenWebUI do not provide easly-accessible logs via UI.

For the web browser connection, only AnythingLLM provides something that is ready out of the box. It is a plugin for chromium-based browsers. No other interface provides such, but you are able to connect to any of those by using Page Assist plugin that is available for both chromium-based and Firefox browsers https://github.com/n4ze3m/page-assist – I would say my great discovery of this research, I liked it a lot.

For the Visual Studio Code the only native integration I have found was for LM Studio by using CodeGPT plugin. All the other plugins I have found were either not connecting to API or had some other problems, but some of them worked properly with Ollama. I will not recommend any of VSC plugins because there are many of such and you can find the one that suits best your needs. Although you can also test some of the plugins I tested: CodeGPT, Privy, vscode-openai, llm-vscode, Local AI Pilot, and few with “Ollama” in name. So in my opinion for VSC – do not bother with OpenAI API compatibility, but connect to Ollama.

For the Office package I have found TextCraft addon

(https://github.com/suncloudsmoon/TextCraft). But besides that every other plugin I have tried is either paid or requires registration or both, they are not working locally, so they do not comply with configuration standard I wanted to achieve. And problem with TextCraft is that it just did not work with my configuration at all – I am not sure why – so maybe it will work with your configuration.

Last addon I can recommend is ThunderAI – https://addons.thunderbird.net/en-US/thunderbird/addon/thunderai/ – it is an addon for Thunderbird which is able to connect directly with Ollama and server standard operations such as: rewrite, summarize or translate. What is great about it is that you can add new prompts in your language so even if you have mailboxes run in different languages (a I am) you can easily make it work for you. To make it work just remember to set OLLAMA_ORIGINS as described in “Important information” section – without it for me it couldn’t connect to Ollama.

UX, Stability and Licensing of LLM Interfaces

AnythingLLM just looks to be user friendly, but it somehow fails. It seems at the very beginning that it is simple to use. But when you try some advanced configs, they do tend to be described counterintuitive – and this also goes with button placement and saving. I do imagine that it is a matter of getting used to it, but if you add to it that AnythingLLM is the only program that for me crushed on regular basis (especially when uploading files), it is getting really annoying. For developer it should be understandable, although if anything fails there are little logs – sometimes this program fails without leaving any trace. License seemed to be MIT license, which is very good.

LM Studio might seem overwhelming on the start, but you can switch to client view aswell, which is a bit simpler. It creates this feeling that it should be used by more advanced developers, and I think this is what they want to achieve with the end product. You can access more logs but a bit concerning thing is that you have many mixed licenses to use this software and terms of service that can change at any time. Because of this I wouldn’t say that this is 100 percent solution open source friendly even though you have libraries available on GitHub.

Then you have Jan, which is in my opinion easiest to use, and very stable, but not very developer oriented. I would say, even though you have readable server logs, it aims to be simplest to use by non-technical users. License is also very permissive,

as it is AGPL.

GPT4All has very nice aesthetics and is easy to use – not as easy as Jan but still very easy. For developer friendliness, it does not have as easily accessible logs as it could. Besides that, they are using MIT license, which is very good and doesn't constrain you in anything.

OpenWebUI is hardest to use of all that I tested – that is to set up and configure. But once it is set up fully, it is fairly easy to use and powerful. You do not have access to logs via ui, but it is definitely developer oriented, which is especially visible in plugin's functionalities. It uses MIT license, so again, very good for you.

Which local LLM Software Should I Use?

	AnythingLLM	LM Studio	Jan	GPT4all	OpenWebUI
MODEL MANAGEMENT					
Installer Available	Yes	Yes	Yes	Yes	No
Integrations with model providers	Multiple providers, but no actual search UI	Just HuggingFace, but you get nice results view	Multiple providers and very user friendly search	Easy to find and nicely presented model parameters	No, just Ollama
Integrations with commercial providers	Yes, multiple	No	Yes, multiple	No	OpenAI compatible
Easy folders and chats management	Yes	Yes	Chats without groups	Chats without groups	Yes
IN-CHAT FILES					
Images output	No – not found	No – not found	No – not found	No – not found	Yes, by ComfyUI or commercial DALL-E

	AnythingLLM	LM Studio	Jan	GPT4all	OpenWebUI
Accepting text documents	It sometimes hangs and you need to restart	No problems encountered	Uploading documents is currently totally broken	Just XLSX files, I'm not sure why	Sometimes slowly, but works
Interpreting images and possible OCR	Yes	Yes	Cannot load into chat	Cannot load into chat	Yes
Audio transcription	No	No	Cannot load into chat	Cannot load into chat	Should be possible via tools
OUTSIDE INFORMATION					
Comparision analysis of different documents	No, limit by LLM capabilities	No, limit by LLM capabilities	No, limit by LLM capabilities	No, limit by LLM capabilities	No, limit by LLM capabilities
Multiple documents that can be treated as RAG	Yes, not bad document management	No	No	Yes, very user friendly and with chat context	Yes
Can search information online	Yes	No	No	No	Yes
Another connections available	Yes, natively	No	No	No	Yes, some, with plugins
INTEGRATIONS AND API					
Ability to serve OpenAI compatible API	Yes, non-standard endpoint	Yes	Yes, with nice logging	Yes, but hidden logs	Yes, but hidden logs
Integrate with web browser	Chromium dedicated plugin	No, but Page Assist exists	No, but Page Assist exists	No, but Page Assist exists	No, but Page Assist exists

	AnythingLLM	LM Studio	Jan	GPT4all	OpenWebUI
Visual Studio Code	Not via OpenAI API, but to Ollama	Native using CodeGPT	Not via OpenAI API, but to Ollama	Not via OpenAI API, but to Ollama	Not via OpenAI API, but to Ollama
Integrate with Office	No	No	No	No	No
OTHER ELEMENTS					
UX and stability	Tries to be user friendly but is not, buggy, not very stable	Might seem overwhelming on start, but there is “client view” aswell	Easiest to use, stable	Easy to use and nice aesthetics	Not easy to setup and config, but easy to use once ready
Developer friendliness	Some functionalities are counter intuitive, but there is extended documentation. Little logs	More logs and for sure developer oriented solution	Not very developer oriented, but readable server logs	No easily accessible logs	No logs via UI, but definitely developer oriented
License	MIT	Mixed + ToS	AGPL	MIT	MIT

A one-sentence summary for every tested software from me would be:

- The most concerning thing about LM Studio is their license and that’s reason I wouldn’t use it – besides that it looks like a very good software.
- If you are looking for something that is as easy as possible to use – I would recommend Jan, It will just work.
- If you don’t like Jan for some reason – go with GPT4All – it seems that after fixing few things it can be a pretty strong competitor.
- AnythingLLM is the one that is packed with features, but there is serious problem with stability and downloading models could be improved – so it is a good choice if you are feeling adventurous or if it works stable in your environment.
- If none of the above satisfies you and you are not afraid of a little bit coding from time to time (or you are looking for some LLM development) – go with OpenWebUI and it will pay you back with extensibility.

hope you liked this comparison and that I put a little light to complex word of using offline LLM assistants (as if the LLM world itself wasn’t complex enough). Maybe this will help you not just understand better how LLMs and AIs work, but also put them to better use, even offline – either for business commercial use or for your own workflow improvement. And if you are looking to implement AI solution into your SaaS software – then you just found a company that can help you with it – just use contact form below and get in touch!