# Gathering Supplies – Preliminary Set Up

1. Ollama

Fortunately, Ollama is just a simple [download](https://ollama.com/). Once, downloaded and through the setup wizard, I accessed Ollama via the terminal.

The first command than I ran was *ollama run mistral* which was supposed to start up the minstral model. However, I did not have minstral downloaded, so that was first automatically downloaded. Once running, the model was fully functional. The only complaint that I have is that it seemed to run quite slow, but through preliminary research this seems to be a product of both hardware limitations and the model file that I am using. I can’t realistically solve the hardware problem, but I do think that optimizing the model file will be something to look into in the future.

A black screen with white text

Description automatically generated

Mistral running in the terminal via Ollama

1. Docker

Docker wasn’t as simple. Initially, I thought it was just an [install](https://www.docker.com/products/docker-desktop/) and setup wizard, but it wouldn’t work. It seems that because I have Windows 11 Home rather than Windows 11 Pro, Docker will only run Linux containers. To resolve this issue, I had to download WSL first before trying to utilize docker. I also noticed that upon doing this, docker had many more options within its desktop app which I took to be a sign that it was working (at least better than before).

1. Ollama Web UI

[Ollama Web UI](https://github.com/open-webui/open-webui) was easy to set up (once the Docker issues were resolved). It was as simple as running the following command:

*docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main*

Since I didn’t already have Web Gui installed for me, it automatically downloaded it and then set up the Docker container with Ollama running inside.

# Ollama in the Browser

With all these prerequisites complete, I should in theory have Ollama in my browser.

The first launch seemed processing as that local host port now had a sign in page for Open WebUI.

A screenshot of a login form

Description automatically generated

Landing page on first launch

After logging in, it was fully functional despite being in the browser. All that’s required is selecting a model and then asking a question. It appears that since it is all local and not connected to the internet, the only models that you can select are those which are already downloaded. This is one difference between in the terminal and in the browser that makes the terminal marginally better. However, once you have the models you want set up, this benefit goes away and the ease of the Web UI makes it basically completely superior to the terminal version for the task of just a chat bot. A screenshot of a computer

Description automatically generated

First question in the browser

# Beyond Local – Access Outside the Network

# Apple Integration – Enchanted LLM

# More Models – Existing and Created

## Existing

Using existing models is as simple as downloading them and then refreshing to web UI to access to newly available models. I found the easiest way to do this was to first run Ollama in the terminal which would automatically download the requested model. Then I would move over to the web UI for the actual use of the model.

codellama

The first model that I wanted to explore was [codellama](https://ollama.com/library/codellama) because I have found one of the main uses for Chat-GPT in my current workflow is quickly generating short code snippets to plug into existing programs. Having a locally hosted way to do this would be very helpful for when I am working on projects without access to the internet because I am on the go or whatever (like at a coffee shop that has an open public network that I don’t want to connect to for security reasons). I tried both the general use codellama model and the Python specific version. I didn’t really see a difference, but I would assume that the Python specific version is just more capable at genearting advanced responses due to its specialty in Python.A screenshot of a computer program

Description automatically generated

A function generated by the model to calculate factorials

## Created