The University of Texas at Austin
Machine Learning Laboratory

OLLAMA + RAG

MLL SUMMER ACADEMY

[LAB 2]

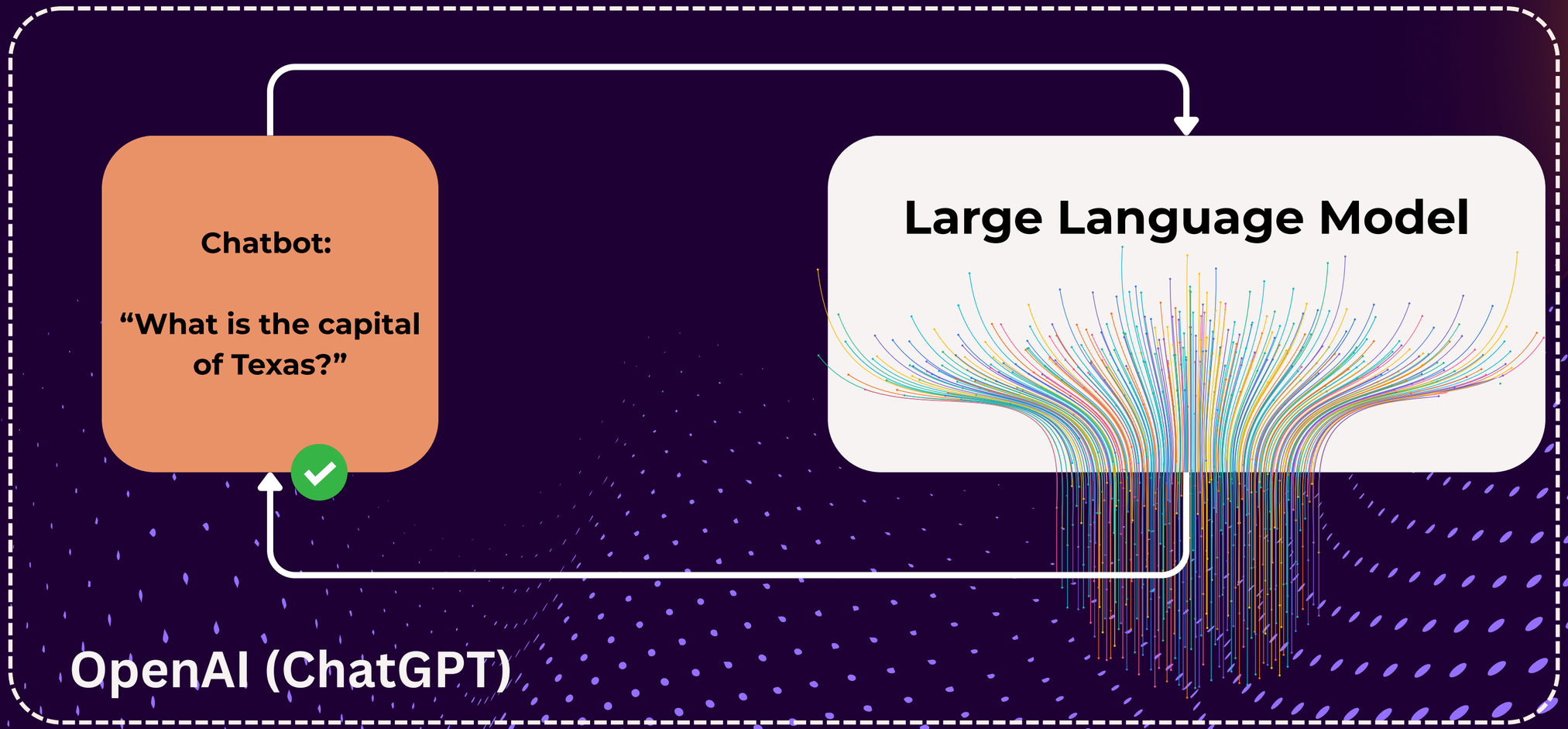
IFML Institute for Foundations of MACHINE LEARNING

JUNE 2025

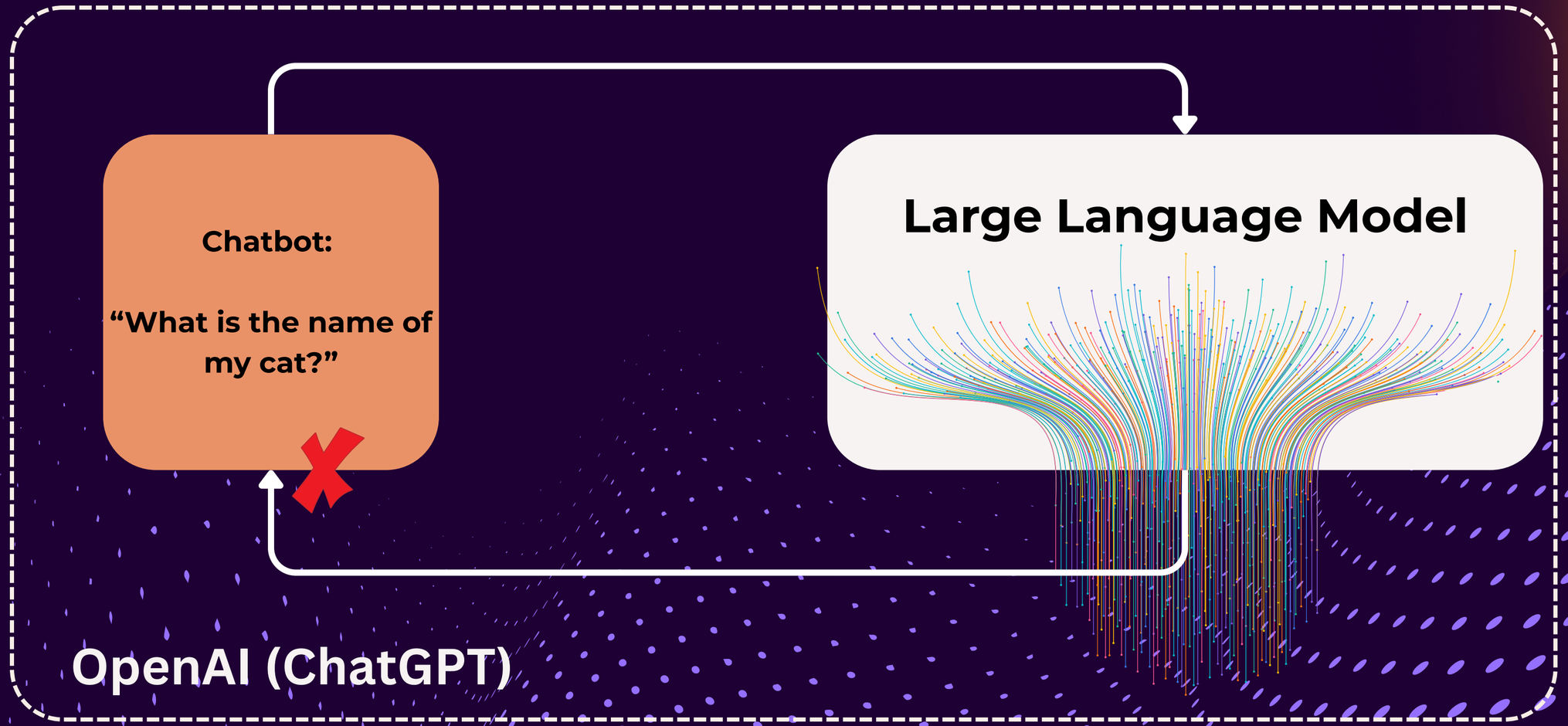
The University of Texas at Austin
Machine Learning Laboratory

RETRIVAL AUGMENTED GENERATION

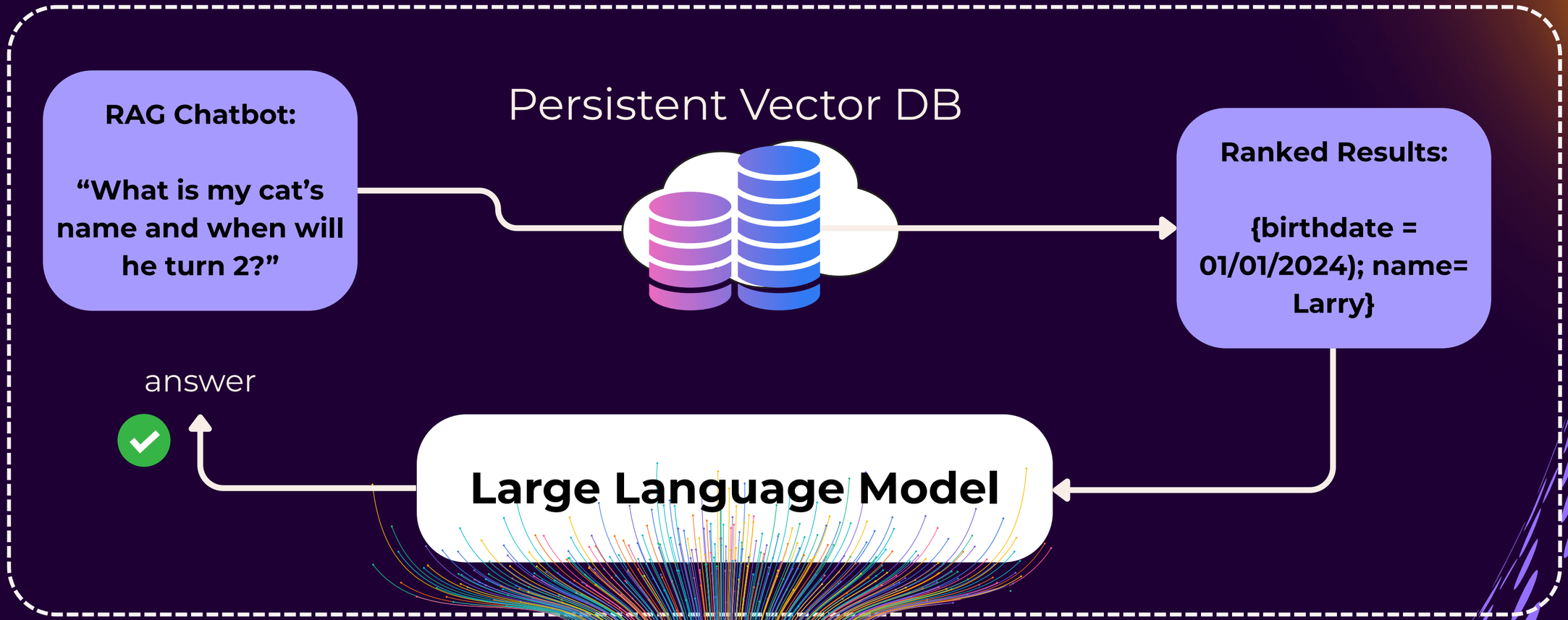
a technique that enhances large language models by retrieving relevant information from external data sources to generate more accurate, relevant, and grounded responses.



RETRIVAL AUGMENTED GENERATION



ADDING CONTEXT & PERSONALIZATION



BUT WAIT... WHAT ABOUT THE COST?

Even in the RAG setup shown, there are costs (via an API) for the LLM to translate the embedding from our RAG database into text for our chat response.

Also what if I don't want to upload all of my personal files for the database either?

01

It's possible to take RAG "offline"

Using open source LLM's and software tools like Ollama, you can run these models on your own hardware!

02

Increase the privacy of what you share

The cloud is just someone elses computer, no matter how large it is. Great for many things, but you should have a choice. Running locally means no internet required!

03

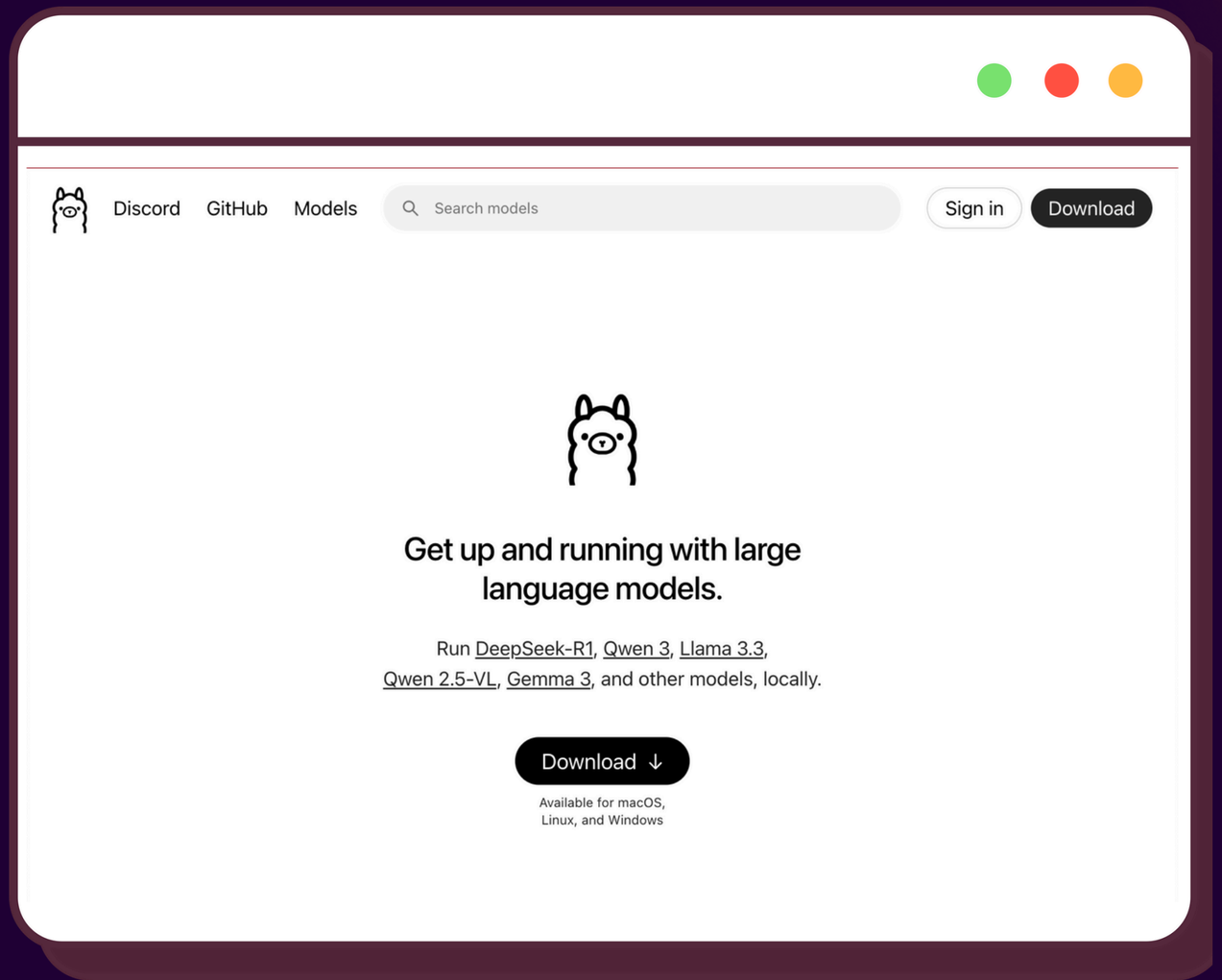
Adapts and improves as you use it

Over time, each interaction you have with your system, and the more files you add, the richer the content and quality of responses will be.

The University of Texas at Austin
Machine Learning Laboratory

OUR LAB

You all have the skills and ability to create this kind of system. And best of all, you can refer back to this guide later and create a system like this on your own computer to use anytime and anywhere.



The University of Texas at Austin
Machine Learning Laboratory

The University of Texas at Austin
Machine Learning Laboratory

OUR BUILD PLAN

We'll work through these steps together as a group, each building upon the next. By the end of the lab, you'll have downloaded 2 different model files, embedded a few PDF's, and have a working UI to ask questions and receive answers about information from those documents.

01

SETUP OLLAMA SERVER & LOAD MODEL FILES

We'll do this through the command line in terminal on your Ubuntu machine in class. We're going to start with the **phi4-mini** model.

02

UPLOAD PDFS OF YOUR CHOICE

Build your library - these could be anyting, but the larger the file the longer it will take for our system to parse and embed the PDF. You have about 2.5 GB to work with on your Lab Machine.

03

REVIEW THE PROGRAM & RUN IT!

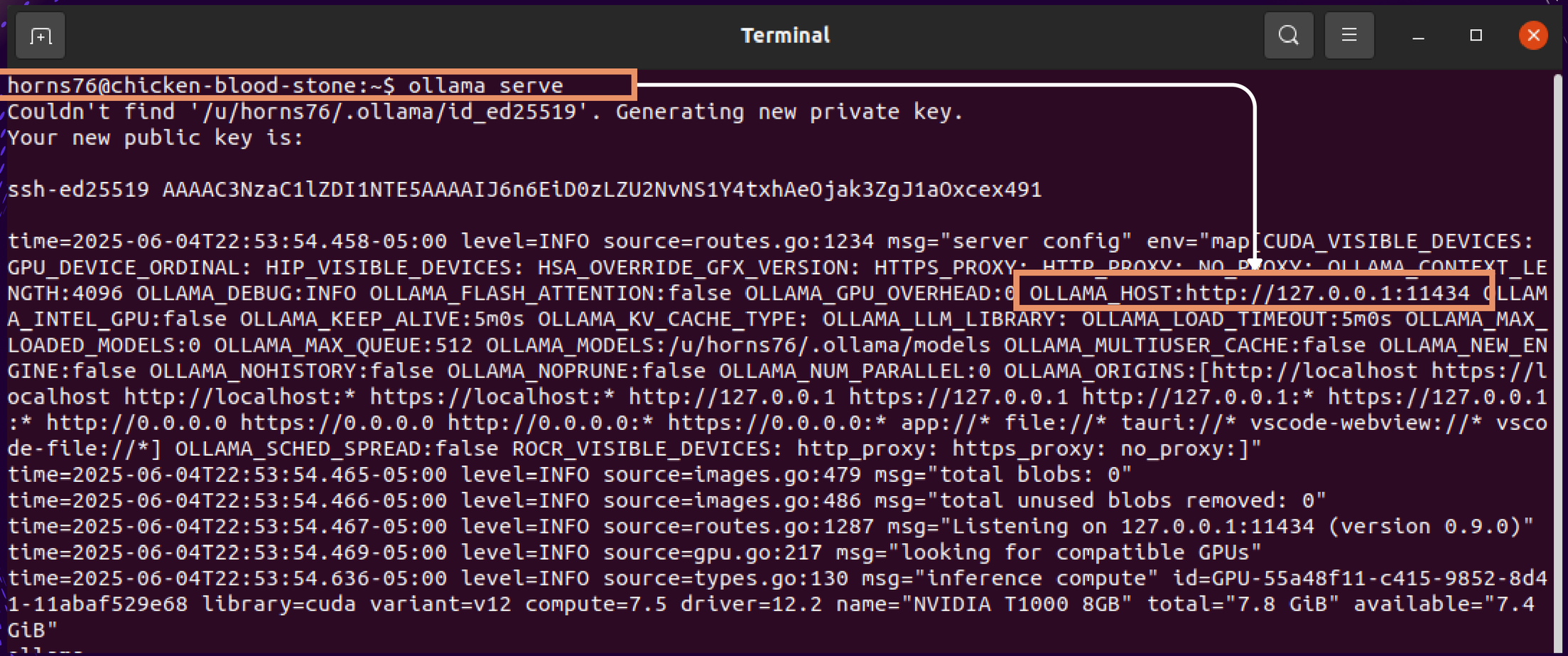
Launch the rag_pipeline.py file and start to chat!

PART 1:

LAUNCH & USE

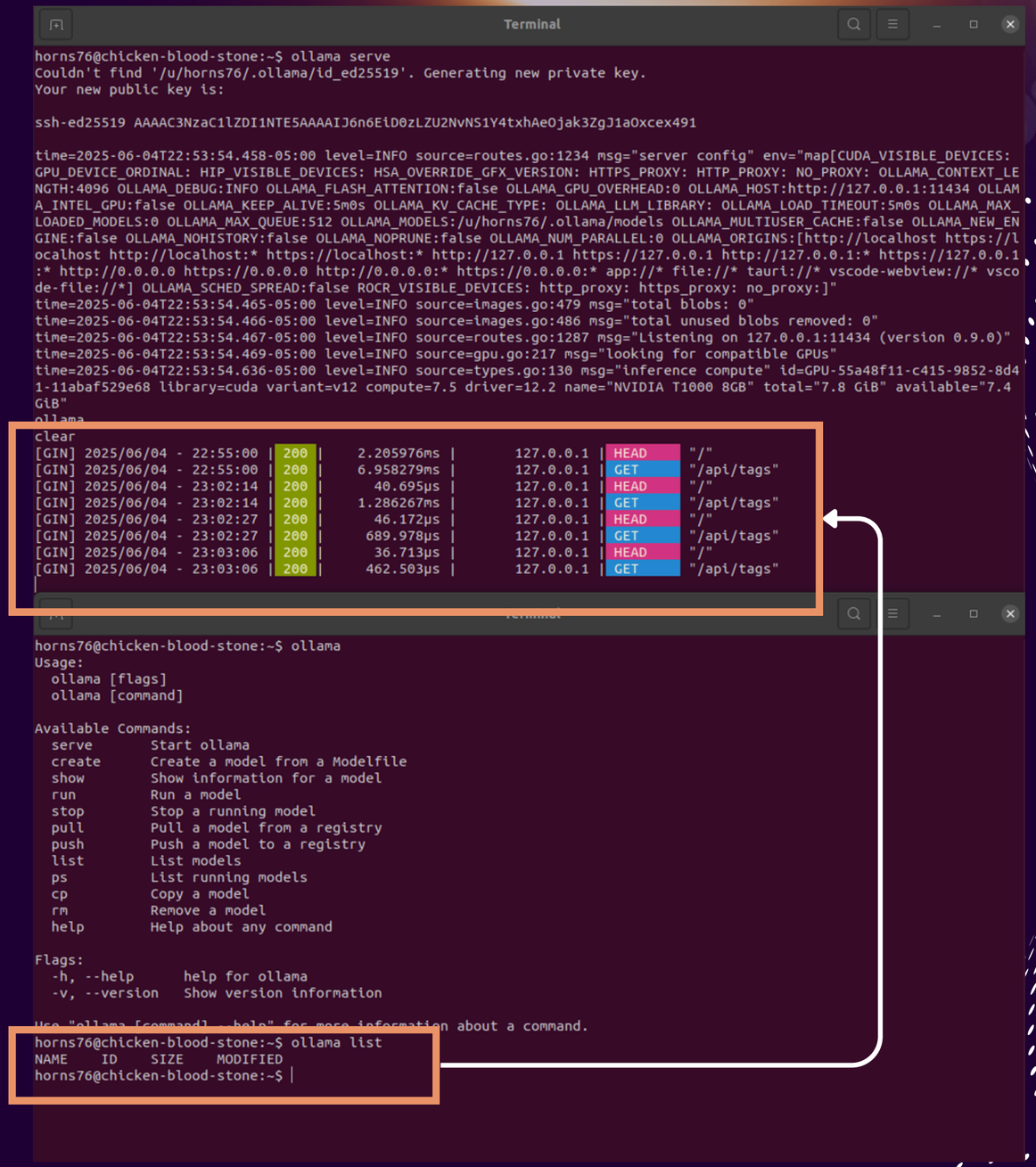
OLLAMA

LAUNCH OLLAMA SERVER



The University of Texas at Austin
Machine Learning Laboratory

VERIFY THE OLLAMA SERVER RUNNING ON YOUR MACHINE

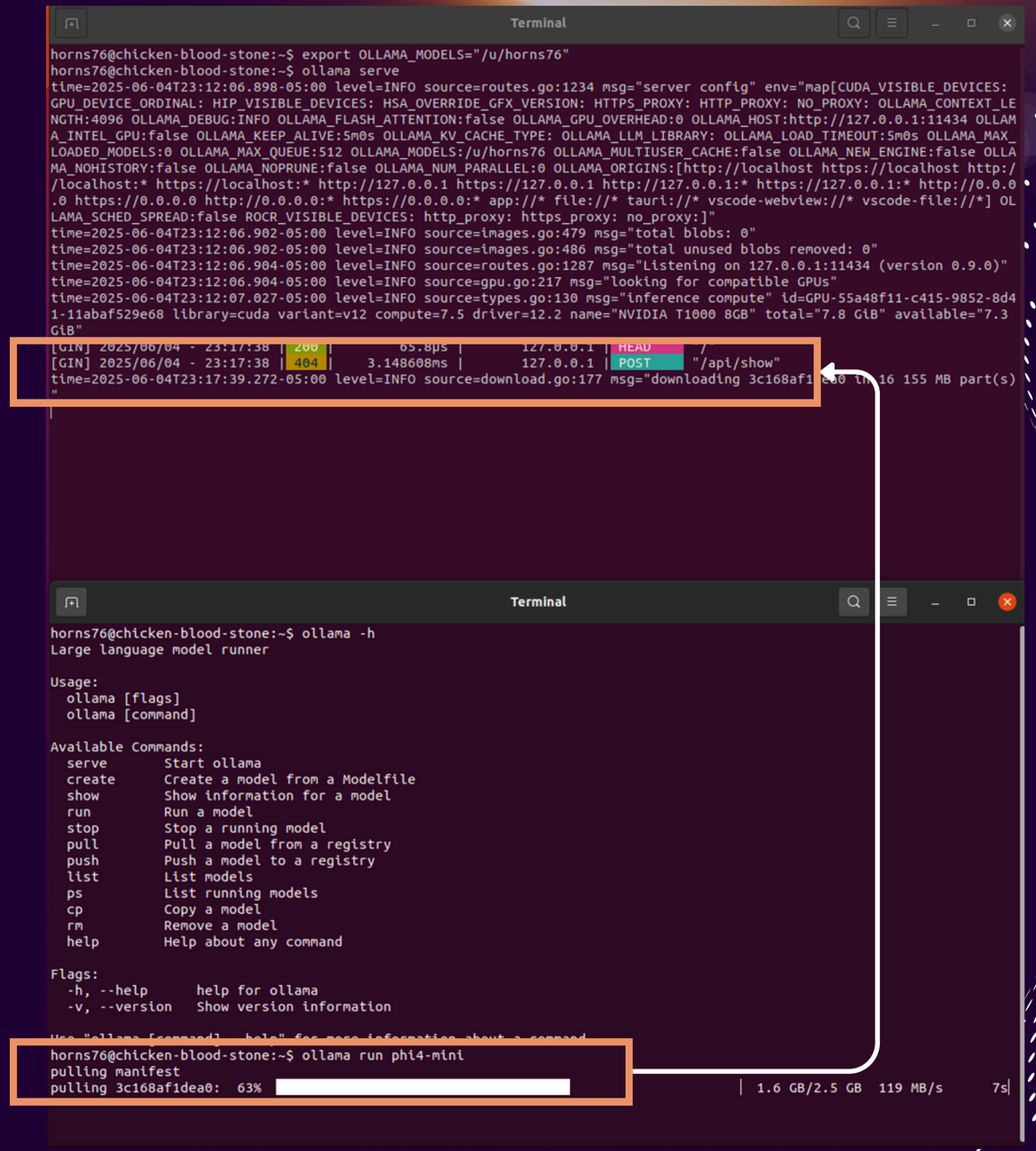


The University of Texas at Austin
Machine Learning Laboratory

INSTALL OUR MODEL:

phi4-mini

Microsoft Research



LOCAL CHAT 🔥

```
horns76@chicken-blood-stone:~$ ollama run phi4-mini
pulling manifest
pulling 3c168af1dea0: 100%                                                2.5 GB
pulling 813f53fdc6e5: 100%                                                 655 B
pulling fa8235e5b48f: 100%                                                1.1 KB
pulling 8c2539a423c4: 100%                                                 411 B
verifying sha256 digest
writing manifest
success
>>> hello, what is your name and what can you do?
Hello! I'm Phi developed by Microsoft. I am an advanced AI assistant designed to help with a wide range of tasks
including answering questions about various topics such as science, history, technology, language translation,
data analysis, creative writing assistance (like generating poems or stories), providing summaries for lengthy
texts like articles and books.

If you have any specific request that you're curious about, feel free to ask!

>>> Send a message (/? for help)
```

To exit chat, type: `/bye`

PART 2:

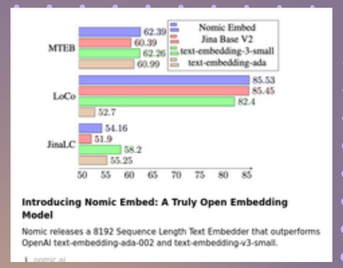
RAG BUILD

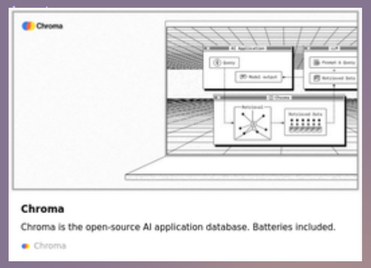
The University of Texas at Austin
Machine Learning Laboratory

OUR SYSTEM COMPONENTS

AI can automate and accelerate the incident response process, enabling faster identification, containment, and recovery from cyberattacks.

references:





01

`nomic-embed-text`

EMBEDDING:

This model is going to embed the parsed text from our RAG PDF files. We'll also use this model to embed our query before searching our vector database

02

Chroma

VECTOR DATABASE:

We're going to use the lightweight and powerful ChromaDB library for our local database.

03

Meta

LLM:

Llama3.1 is an open-source family of reasoning models built using a dataset developed and maintained by Meta.

OUR PLAN

SInce this is going to be a terminal based RAG system, we're going to build each component in sequential order, one after the other.

1: pdf_loader.py	load & chunk PDFs
2: pdf_manager.py	detect & track new PDF's
3: embedder.py	send chunks for embedding
4: retriever.py	ChromaDB storage, retrival, & reranking
5: ollama_runner.py	build prompt, send to LLM
6: chat_loop.py	handle input / output for chat
7: rag_pipeline.py	bring it all together

CLONE THIS REPO:

```
git clone https://github.com/IFML-UT/MLLAcademy-2025_OllamaRAG.git
```