BUILDING LLM APPLICATIONS WITH PROMPT ENGINEERING

- BY NVDIA

17th May 2025

LLM :- Large language Model application.

⇓

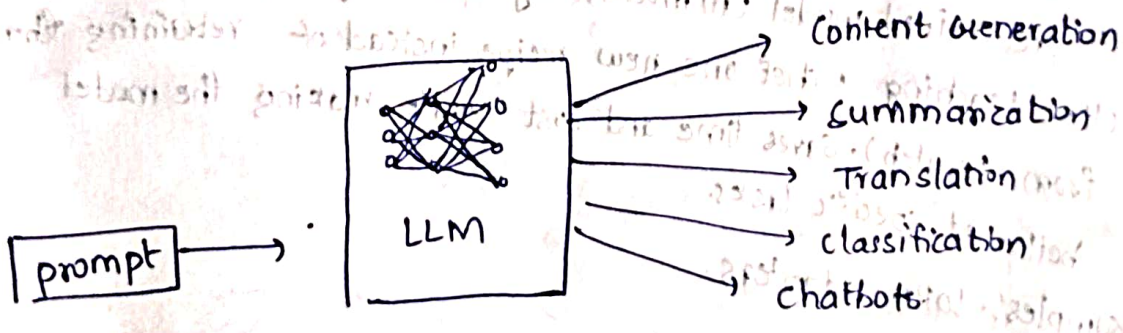
An AI system trained on massive amount of text data to understand and generate human-like language. It predicts and produces text based on patterns it has learned.

Example:- GPT-4, Gemini, claude etc.

-We will host a local large language model on GPU Accelerator.

Course Goals

- Learn to interact programmatically with chat-variant LLMs
- Be capable of using LLMs for a wide variety of application use cases
- Become fluent with fundamental Langchain techniques
- Develop good habits around Iterative prompt-Engineering.



- with python we will be interacting ↑ Via API → directly with a LLM ↕

In this course we are working with Llama version 3.1 8 billion parameter LLM.

Langchain:- a framework for building applications powered by large language models (LLMs). It helps connect LLMs to external data. It helps connect LLMs to external data, tools and workflows, making them more useful for tasks like chatbot, data analysis and automation.

API (Application programming Interface):- A set of rules that leds different software systems communicate with each other . It defines how to request data or services and how they should be delivered.

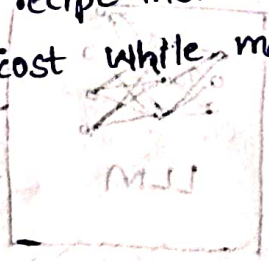
Prompt Engineering in context

1. Prompt Engineering:- Practice of carefully crafting inputs (prompts) to guide large language models (LLMs) to produce the desired outputs. It involves clear, specific instructions or examples to improve the model's responses.

2. RAG :- (Retrieval Augumented Generation):- Supercharging an LLM with real-time data. Before answering, it first searches your documents/ knowledge base (like a librarian fetching books), then generates answers using that fresh info - making responses smarter, fact-based and up-to-date.

3. Fine-tuning (parameter efficient):- A shortcut to customize a pre-trained model (AI model) by updating only a small part of it (like teaching a chef one new recipe instead of retraining them from scratch). Saves time and cost while making the model better at specific tasks.

Examples:- LoRA, adapters.



Prerequisites

- python
- Some prior LLM exposure.

Course content

- NVDIA NIM (slides)
- Intro to prompting (Interactive)
- LCEL chains (Interactive)
- PE techniques w/ messages (Interactive)
- Structured data and document tagging (Interactive)

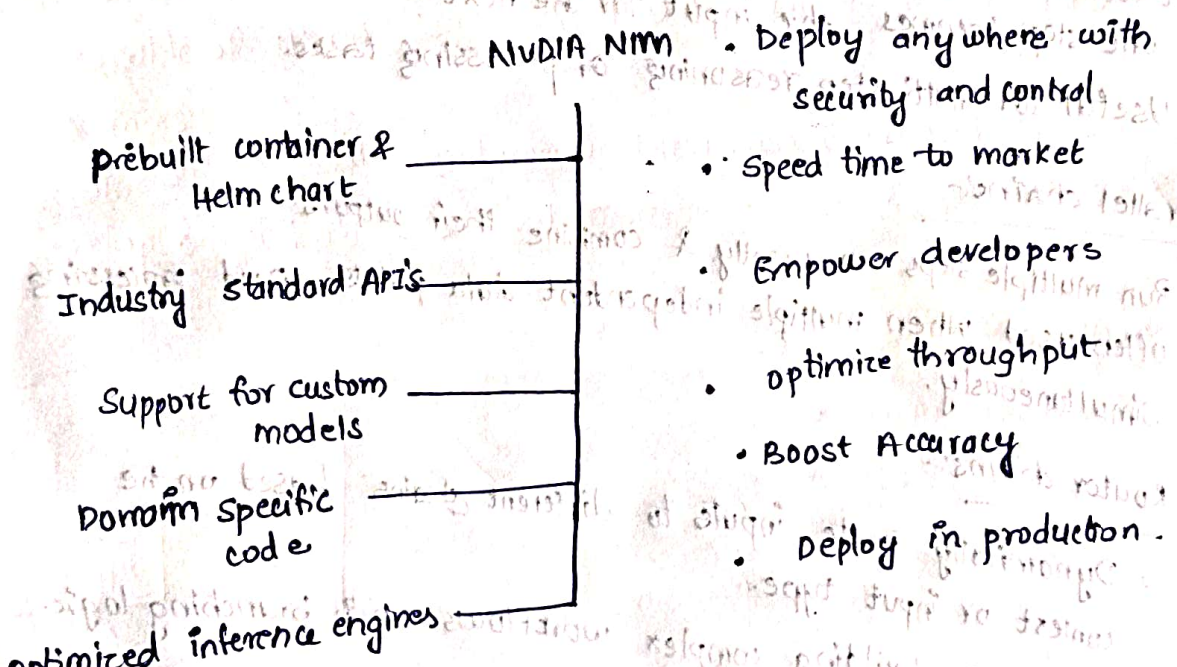
- Tools and Agents (Interactive)
- Course Assessment (Interactive).

NVDIA NIM

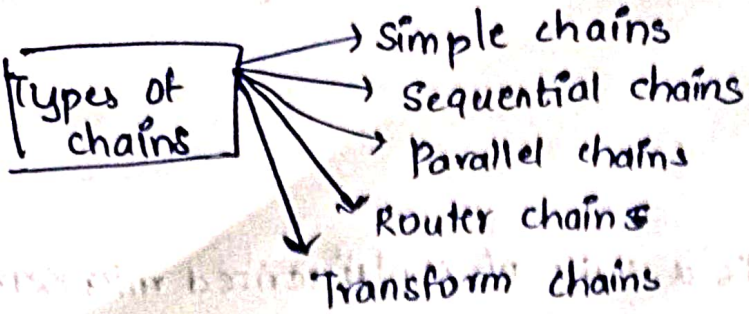
⇓

NVDIA · Inference Microservices :- It's a suite of containerized microservices designed to simplify and accelerate the deployment of AI models across various Infrastructures, including cloud, data centers, workstations and edge devices. These microservices are optimized for NVDIA GPUs and provide standardized APIs, making it easier for developers to integrate AI capabilities into their applications.

⇒ NVDIA NIM is the fastest way to deploy AI models on accelerated Infrastructure across cloud, data center, and PC.



Chain:- A chain refers to a sequence of operations or steps that connect different components like language models, prompts and data sources to create more complex and meaningful outputs. chains are a fundamental building block in Langchain and can be used to combine various language model capabilities, handle inputs and outputs, and incorporate custom logic.



1. Simple chain:-

- Directly connect a single input to a single output.
- Example:- Passing a user query to a large language model and returning the response

2. Sequential chains:-

- Execute multiple steps in sequence, where the output of one step becomes the input for the next.
- Useful for multi-step reasoning or processing tasks.

3. Parallel chains:-

- Run multiple steps concurrently & combine their outputs.
- Often used when multiple independent data points need processing simultaneously.

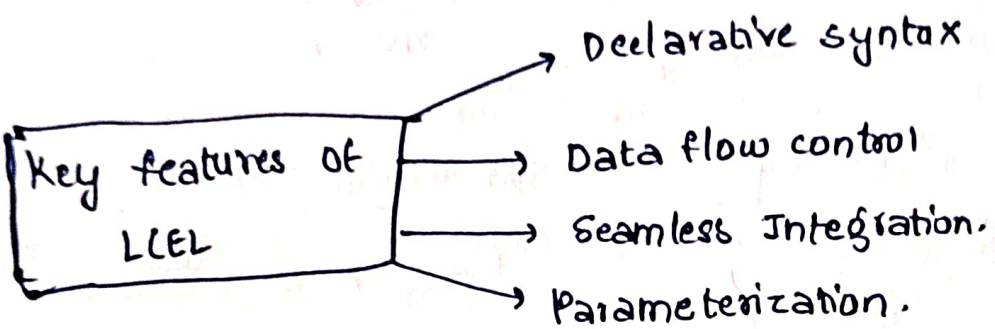
4. Router chains:-

- Dynamically Route inputs to different chains based on the context or input type.
- Useful for building complex workflows with branching logic.

5. Transform chains:-

- Apply custom functions or transformations to inputs or outputs within a chain.
- Example:- Formatting outputs, filtering responses or adding custom metadata.

LCEL (Langchain Expression Language) - is a powerful scripting language. introduced in Langchain to provide a flexible way to define, connect and manipulate components in language model workflows. It aims to make it easier to work with complex chains by providing a concise syntax for describing the flow of data and operations.



1. Declarative Syntax:- LCEL uses a declarative approach, allowing you to describe what should happen, not how it should be implemented.

2. Data Flow control:- LCEL can handle data transformation, conditional logic, and routing without requiring complex python code.

3. Seamless integration:- LCEL is tightly integrated with Langchain's core components like chains, memory, prompts and agents.

4. Parameterization:- Easily pass inputs and outputs between different components without manual wiring.