

# GETTING STARTED WITH LANGCHAIN




# LangChain



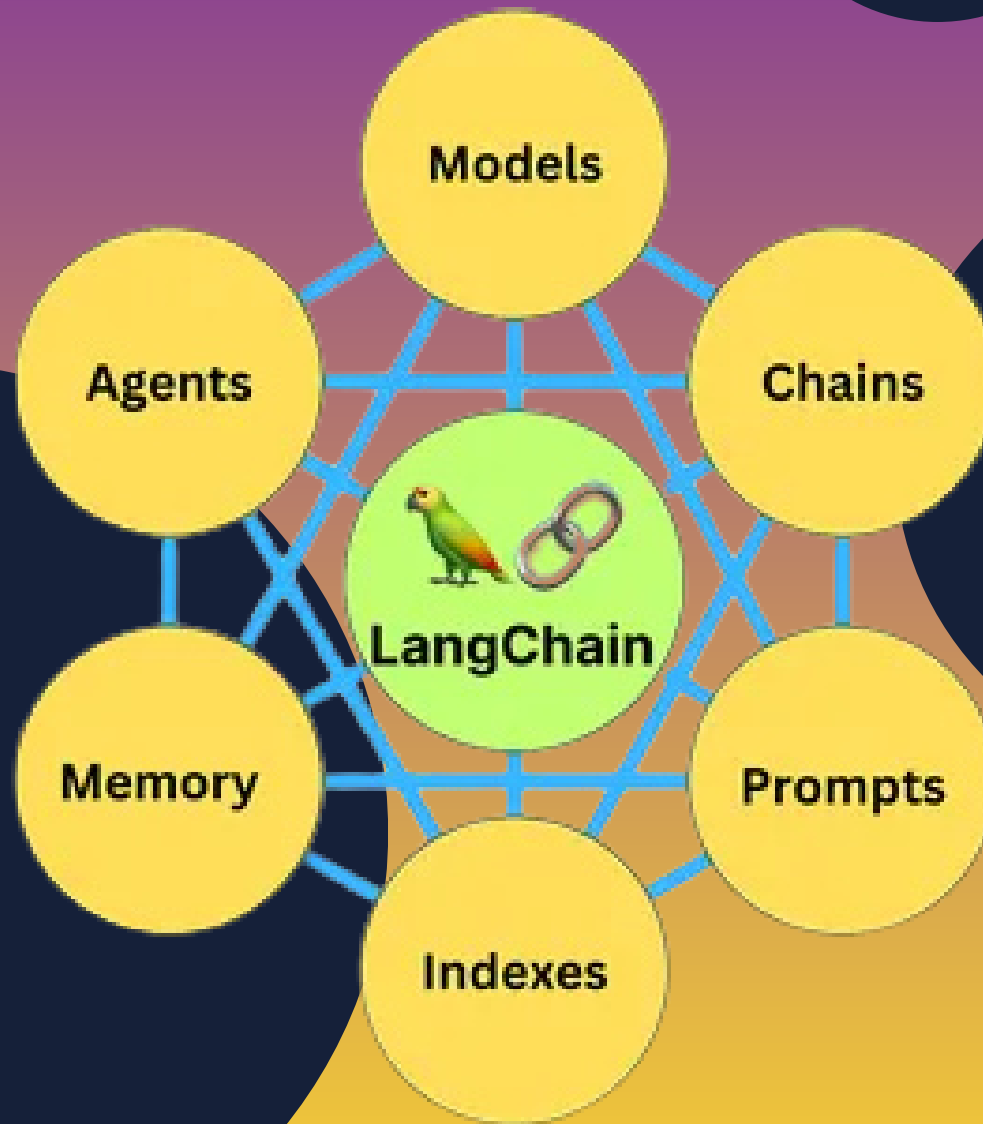
Ahmad Sana Farooq

# WHAT IS LANGCHAIN?

- - A Python framework to build apps powered by LLMs
- - Helps with prompting, retrieving documents, and chaining tools
- - Makes it easy to build chatbots, assistants, and RAG systems
-  Think of LangChain as a 'Toolkit' for working with LLMs

# WHY LANGCHAIN?


- - Handles complex backend LLM logic
- - Modular: Plug in different models (OpenAI, Gemini, LLaMA, etc.)
- - Integrates with tools like Vector DBs, Embeddings, APIs
- - Saves time – no need to reinvent the wheel!



# LANGCHAIN CORE CONCEPTS

- - PromptTemplates → Format prompts for LLMs
- - Chains → Combine multiple steps (Prompt → LLM → Output)
- - Tools/Agents → LLMs decide which tools to use
- - Retrievers → Search documents (used in RAG apps)
- - Memory → Keep chat history for conversational AI

# BUILD A RAG APP (SIMPLE OVERVIEW)

- 1. Load documents (PDFs, TXT, etc.)
  - 2. Split into chunks
  - 3. Embed and store in Vector DB
  - 4. Query + retrieve similar chunks
  - 5. LLM answers using context
- 
-  Powered by: TextLoader, Chroma, Embeddings, RetrievalQA

# DEMO USE CASES

-  AI Tutor
-  PDF Q&A bot
-  Code Explainer
-  Customer Support Bot
-  Mental Health Assistant
-  Research & Writer AI Agent

# TIPS FOR BEGINNERS

- - Start with Prompt + LLM
- - Add Chains step-by-step
- - Use Chroma for local vector store
- - Try LangChain Expression Language (LCEL)
- - Don't get overwhelmed — learn by building! 💪



# LET'S BUILD WITH LANGCHAIN!

- LLMs are powerful.
- LangChain makes them usable.
- Start with simple ideas.
- Keep experimenting.
- Have fun! 🌟