docker

Build and Deploy AI-powered Apps locally using Docker GenAI Stack

8-Nov 2024
Docker Singapore Meetup

About Speaker

- DevRel at Docker
- Former Docker Captain
- Docker Community Leader
- Distinguished Arm Ambassador
- Worked at Dell EMC, VMware, Redis

COLLABNIX

Ajeet Singh Raina

Today’s Agenda

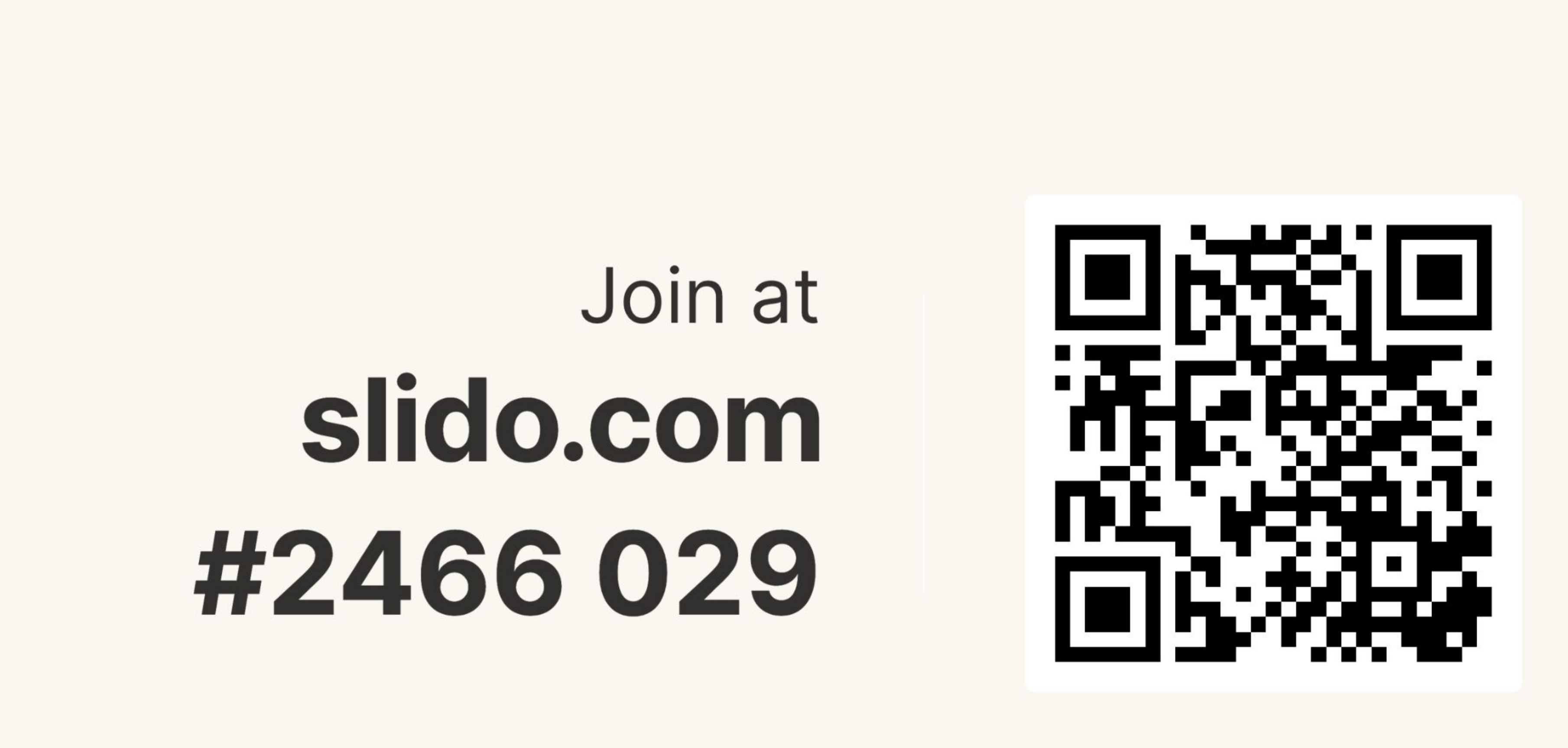
Introduction to GenAI

- **The Rise of LLMs/GenAI**: Understand why large language models are gaining popularity and their impact on development.
- **Traditional ML vs. Foundation Models**: Explore the differences and benefits of foundation models.
- **LLM Hallucinations**: Learn about the limitations of LLMs and how to address them.
- **Overview of GenAI Stack**: Get an introduction to the components of the GenAI stack and how they integrate with Docker.

Practical Applications with Docker GenAI Stack

- **Getting Started with Docker GenAI Stack**: Hands-on sessions to get you started.
- **Use Cases**:
  - **Chat with PDF**: Implementing chat interfaces using PDF data.
  - **StackOverflow Loader**: Integrating Stack Overflow data.
  - **CodeExplorer**: Building a code exploration tool.
  - **Support Agent App**: Creating support agent applications.

What are some of the latest AI tools you're aware of?



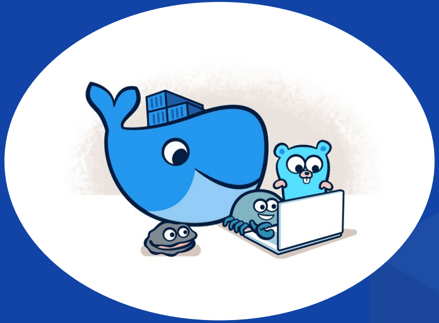
Join at

slido.com

#2466 029

docker

The Rise of LLM



The Large Language Model (LLM) Market was valued at 10.5 Billion USD in 2022 and is anticipated to reach 40.8 Billion USD by 2029 ~ Valuates Reports

LLM Infra Stack – Market Landscape

A work in progress

CriteriaVentureTech

Data Layer

Vector Database & Search

- MongoDB
- elastic
- Pinecone
- zilliz
- Weaviate
- NEON
- supabase
- vectara
- SUPERLINKED
- qdrant
- chroma
- pgvector
- nuclia
- marqo
- deeplake
- upstash
- Vespa
- HYPERSPACE
- NHOST
- LanceDB

Data Management

- UNSTRUCTURED.IO
- mindsdb
- NOMIC
- datologyai
- Automorphic
- Kern
- MANTIUM
- TAYLOR
- lettria
- vue.ai
- Lilac
- Indexify
- .txt

Data Labeling & Annotation

- scale
- Labelbox
- Argilla
- Toloka
- SuperAnnotate
- HumanSignal
- Kili
- prodigy
- Watchful
- datasaur.ai
- refuel
- Anote
- M47

Data Pipeline & Orchestration

- Fivetran
- matillion
- ASTRONOMER
- WarpStream
- Airflow
- Airbyte
- PREFECT
- ZenML
- MAGE
- dagster
- SuperDuperDB
- Flyte
- kedro

Synthetic Data

- gretel
- TONIC
- hazy
- MOSTLY·AI
- scale
- Argilla
- HAVEN
- tuna
- Syntheticus

Model Layer

Fine Tuning & Training

- Argilla
- scale
- Snorkel
- surge
- HAVEN
- arcee.ai
- GigaML
- ANARCHY
- Predibase

RAG

- jina
- LiteLLM
- CREDAL
- Ragas
- EMBEDCHAIN
- nuclia
- SciPhi
- NEUM

Orch. Frameworks

- LlamaIndex
- DUST
- LangChain
- FIXIE
- deepset
- SuperDuperDB
- griptape

Experiment Tracking & Prompt Eng.

- W&B
- comet
- neptune.ai
- DVC
- DagsHub
- AimStack
- HumanFirst
- vellum
- agenta
- Humanloop
- Baseplate
- orq
- HAVEN
- LOG10
- Autoblocks
- Radiant
- HegelAI
- PromptLayer
- KLU
- eppo

Quality Testing & Evaluation

- Galileo
- trulens
- GANTRY
- arize
- deepchecks.
- Cleanlab
- Giskard
- Bespoken
- context
- Patronus AI
- Toloka
- vectorview
- ATHINA
- BRAINTRUST
- RagaAI
- AgentOps
- promptfoo
- Confident AI
- inspeq ai
- Chatter
- UpTrain
- GOLDENSET
- Quotient
- phospho

Federated Learning

- Flower
- FedML
- Opaque
- Devron
- sherpa.ai
- DynamoFL
- integrate.ai
- APHERIS
- TUNE INSIGHT

Deployment Layer

- deepspeed
- anyscale
- Modular
- OctoAI
- run:ai
- Kubeflow
- deci.
- cerebras
- SELDON
- NEURAL MAGIC
- CentML
- Paperspace
- baseten
- Modal
- RunPod
- Outerbounds
- Martian
- replicate
- cerebrium
- BentoML
- mystic
- zeet
- Titan
- Alpa
- MULTIVERSE
- orkes
- fireworks.ai
- BANANA
- Lepton AI
- HippoML
- beam
- Floom
- E2B
- commonbase
- PoplarML
- nCompass
- dstack

Monitoring Layer

Observability

- DATADOG
- fiddler
- WHYLABS
- aporia
- arize
- mona
- helicone
- portkey
- EVIDENTLY AI
- HoneyHive
- Langfuse
- Parea
- Superwise
- traceloop
- LogSpend
- exxa
- Flowstack
- LOG10
- BricksAI

App/User Analytics

- Nebuly
- sentify
- Autoblocks
- context

Enterprise Considerations

Security & Privacy

- Synack
- praetorian
- Nightfall AI
- ROBUST INTELLIGENCE
- Arthur
- HIDDENLAYER
- PROTECT AI
- TROJ.AI
- Giskard
- LAKERA
- Lasso
- Sarus
- Guardrails AI
- CALYPSOAI
- skyflow
- MITHRIL SECURITY
- harmonic
- preamble
- Prompt:
- Deepkeep
- PRIVATE AI
- ADVERSA
- CRANIUM
- MINDGARD
- cadea
- AIShield
- PRIVATE GPT
- PROTOPIA
- Cape Privacy
- Automorphic
- Kobalt Labs
- PromptArmor

Governance, Compliance & Risk

- 2021.AI
- credo ai
- Holistic AI
- QUANTPI
- MONITAUR
- Advai
- Deasie
- KomplyAi
- Verta
- Vera
- calvinrisk
- Impact AI
- alinia
- TRUSTIBLE
- DEEPLOY
- LUMENOVA
- ARMILLA
- enzai
- anch.AI
- FAIRLY
- PureML
- Arklow
- fairnow
- Saidot
- Mission Control
- Suzan AI

E2E Approach

E2E LLMOps Platform

- aws
- databricks
- DataRobot
- Hugging Face
- Vertex AI
- clarifai
- haystack
- Lightning AI
- METAFLOW
- deepset
- iguazio
- Dataloop
- CLEAR ML
- Qwak
- LangChain
- ANARCHY
- Adaptive ML
- EXPLOSION
- Domino
- Valohai
- LAMINI
- together.ai
- Kern
- truefoundry
- GPT
- baserun
- Steamship
- Chainlit

No/Low code AI App builder

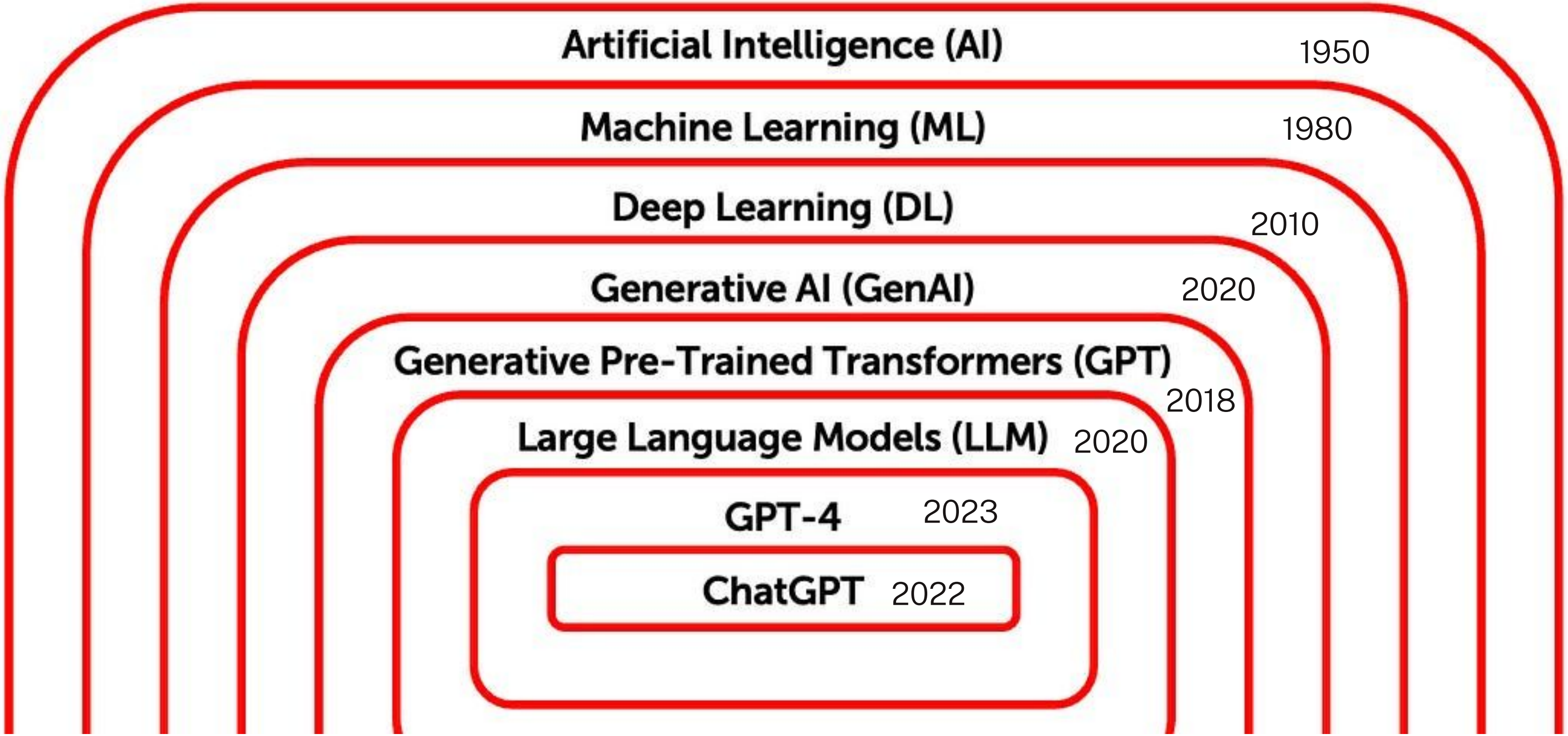
- DUST
- FIXIE
- ONE AI
- Continual
- stack
- gradient
- yurts
- re·tune
- Dify.
- Mind
- Langflow
- Prompt Studio
- FinetuneDB
- Relevance AI
- Mano AI
- GRADIENTJ

Why are LLMs so popular?

Automate data retrieval tasks such as finding and summarizing info from large datasets	OpenAI GPT-3 Hugging Face’s BART
Improve customer service experiences - Creating Chatbots	Rasa Dialog Flow CX Microsoft Bot Framework Amazon Lex
Expedite reading, understanding, and summarizing	HF Summarization Pipeline Google Cloud’s Natural Language API Amazon Web Service’s Comprehend API
Content & code generation	Amazon Bedrock GitHub Copilot (Code Gen) Copy.ai (Content Gen) Jasper.ai (Content Gen) CodeGen (Code Gen) Tabnine (Code Gen)

AI Terminology

@AndreRetterath
datadrivenvc.io



Artificial Intelligence (AI) 1950

Machine Learning (ML) 1980

Deep Learning (DL) 2010

Generative AI (GenAI) 2020

Generative Pre-Trained Transformers (GPT) 2018

Large Language Models (LLM) 2020

GPT-4 2023

ChatGPT 2022

What is Generative AI?

Generative AI

- A branch of AI
- Generate Content(text, images, 3d models)
- Based on LLMs

Large Language Model(LLM)

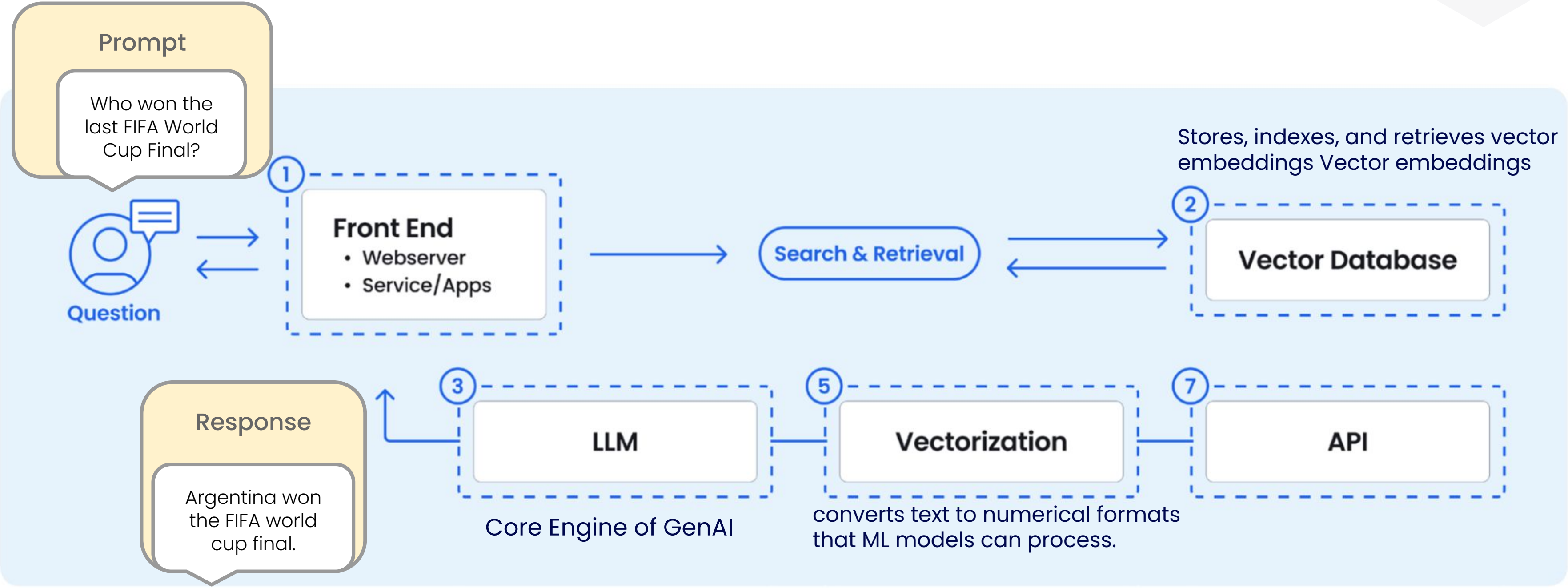
- Core Engine of GenAI
- Neural network
- Trained on massive amounts of data
- Human-like interactions

Use cases

- Chatbots, virtual assistant (customer queries)
- Content creation tools
- Music, image..generation tools
- Code generation tools
- Analyzing Medical reports
- Recruiting, New Employee Onboarding
- Patent and Trademark examination

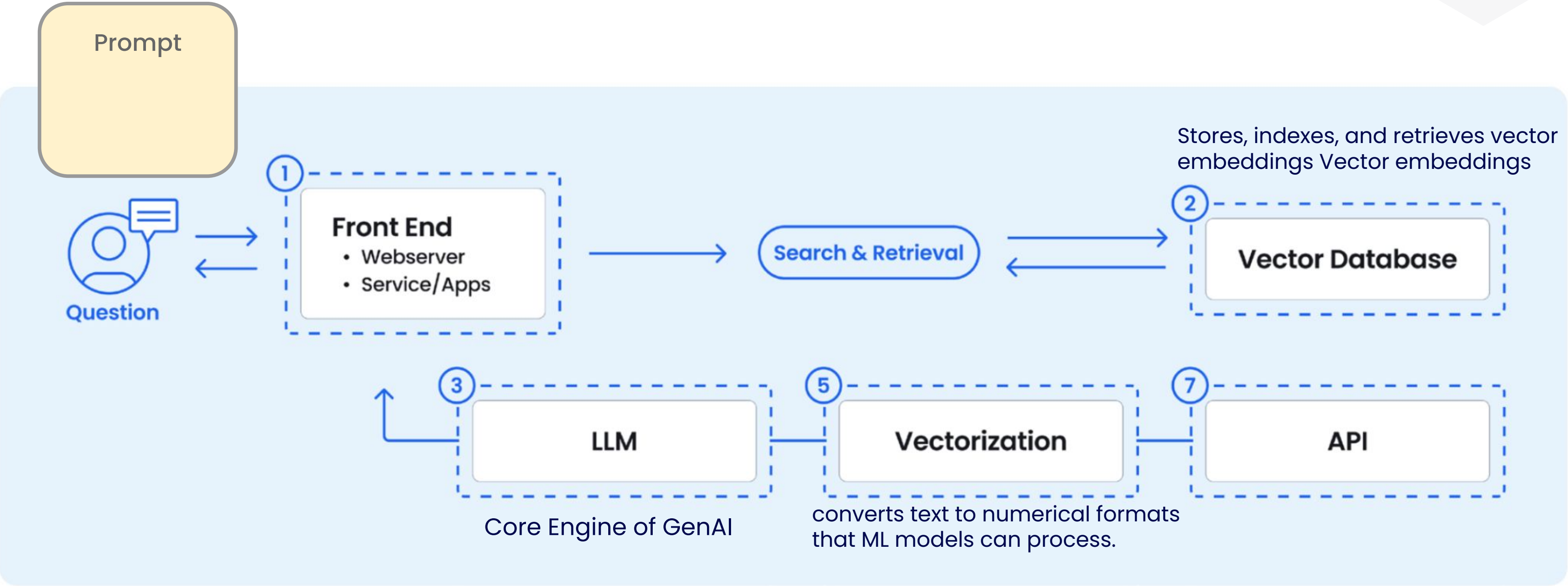
A Typical GenAI architecture

Generative AI empowers users to submit various **prompts** to produce new content, including **text, images, videos, sounds, code, 3D designs, and more**. It acquires knowledge by training on existing digital content and documents found online.


Prompt
Who won the last FIFA World Cup Final?
Question
1
Front End
• Webserver
• Service/Apps
Search & Retrieval
Stores, indexes, and retrieves vector embeddings Vector embeddings
2
Vector Database
Response
Argentina won the FIFA world cup final.
3
LLM
5
Vectorization
7
API
Core Engine of GenAI
converts text to numerical formats that ML models can process.

A Typical GenAI architecture

Generative AI empowers users to submit various **prompts** to produce new content, including **text, images, videos, sounds, code, 3D designs, and more**. It acquires knowledge by training on existing digital content and documents found online.

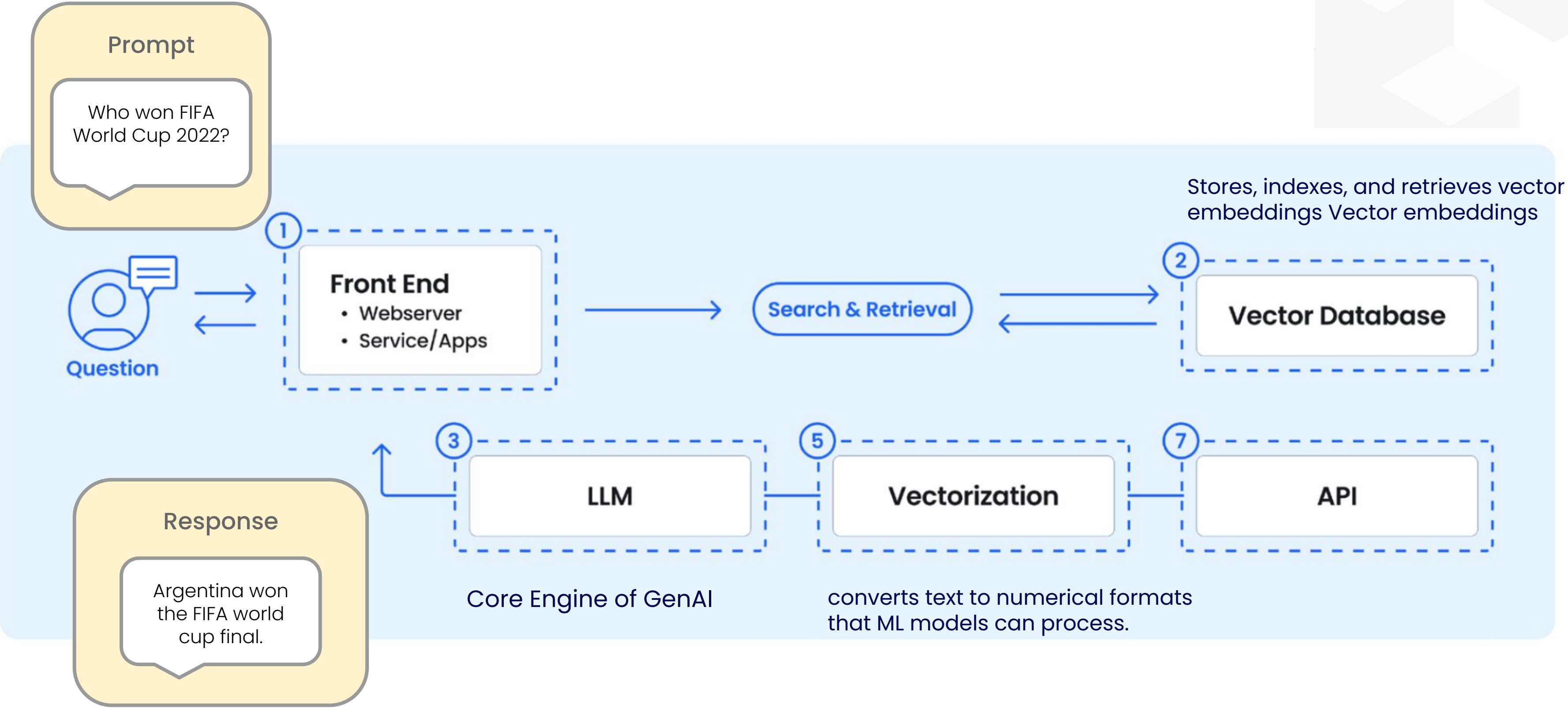


Stores, indexes, and retrieves vector embeddings Vector embeddings

Core Engine of GenAI

converts text to numerical formats that ML models can process.

A Typical GenAI architecture



Prompt

Who won FIFA World Cup 2022?

Question

1 Front End

- Webserver
- Service/Apps

Search & Retrieval

Stores, indexes, and retrieves vector embeddings Vector embeddings

2 Vector Database

3 LLM

5 Vectorization

7 API

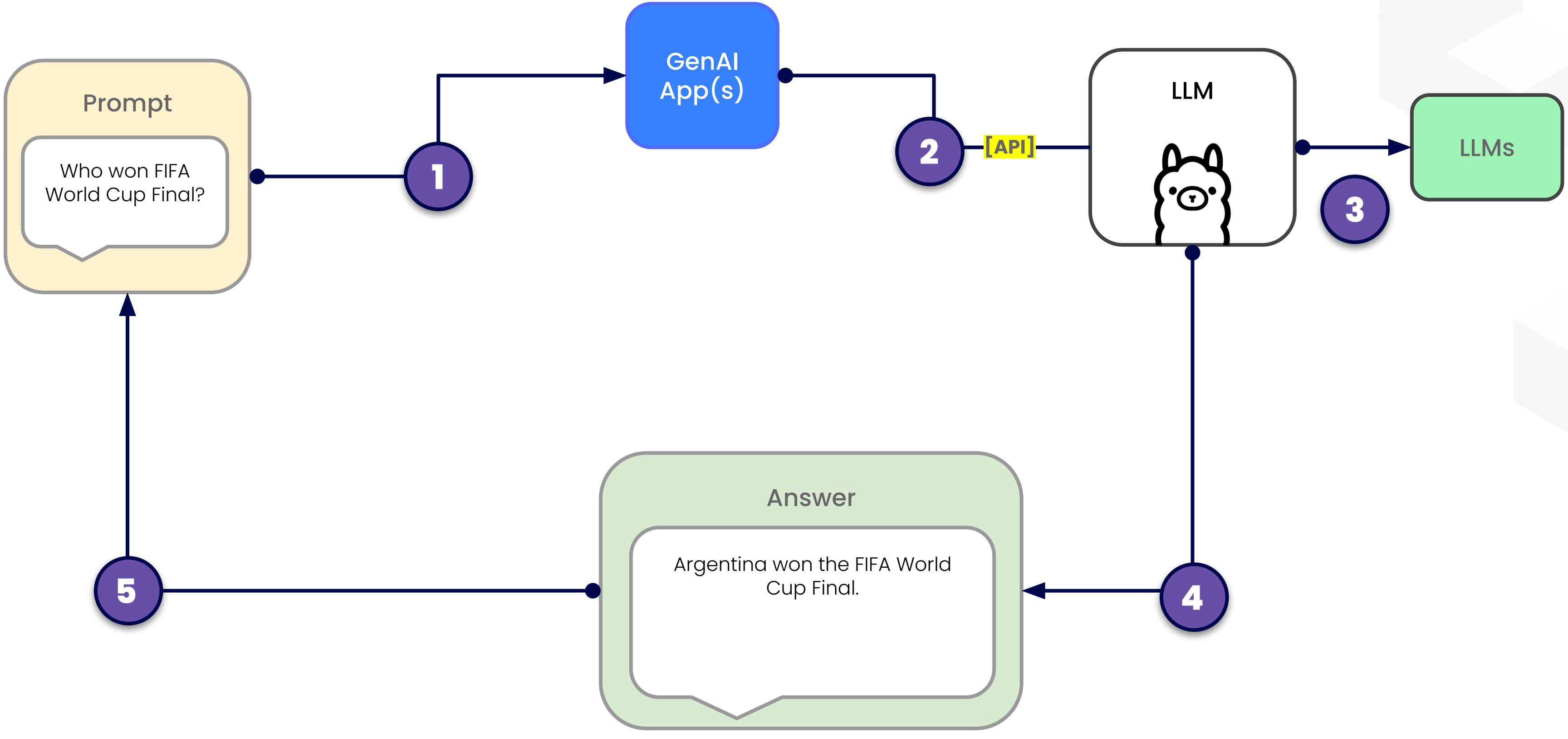
Core Engine of GenAI

converts text to numerical formats that ML models can process.

Response

Argentina won the FIFA world cup final.

A Typical GenAI architecture


Prompt
Who won FIFA World Cup Final?
1
GenAI App(s)
2
[API]
LLM
3
LLMs
4
Answer
Argentina won the FIFA World Cup Final.
5

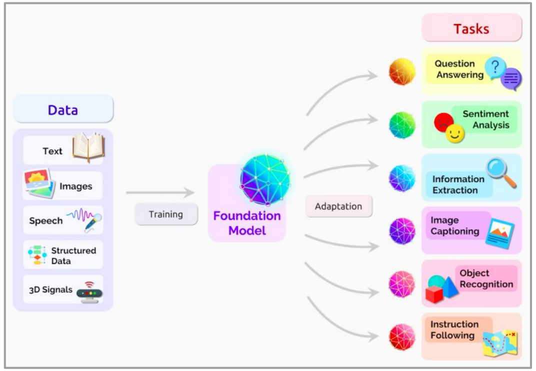
Scaling AI/ML with Foundation Models

Traditional AI/ML

- Models trained for narrow specific tasks
- Poor transferability
- Difficult to get enough training data
- Challenging to scale
- Require specialized expertise

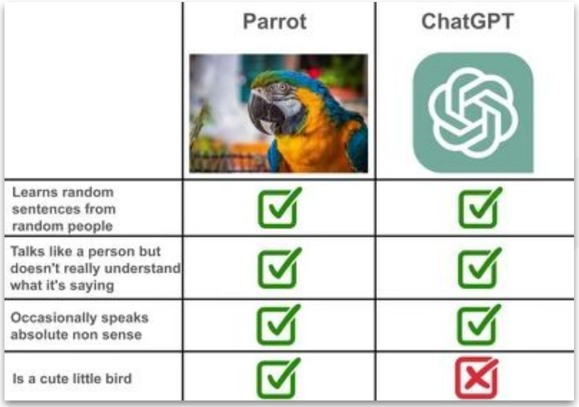
Foundation Model AI Paradigm

- Train a large “Foundation Model”
- Adapt to many different use cases through fine tuning or other forms of augmentation
- Scalable & transferable
- Accessible to developers and other non-experts



Top 5 GenAI Challenges

1. Mostly trained to generate human-like language, not to make accurate inferences with that language
2. Training data comes from publically available corpuses which can contain false, bias, or contradicting information
3. Knowledge cut-off due to resource intensive training (i.e. ChatGPT only trained on data up to Sep 202X)
4. Lack of enterprise domain knowledge
5. Inability to verify or attribute sources



	Parrot	ChatGPT
Learns random sentences from random people	☑	☑
Talks like a person but doesn't really understand what it's saying	☑	☑
Occasionally speaks absolute non sense	☑	☑
Is a cute little bird	☑	☒

Result:

- Great language understanding
- Issues with factual accuracy and consistency

Therefore:

Companies cannot rely on LLMs alone for mission critical data and decisions, instead, they must rely on private, factual information.

LLM Hallucinations

Definition: Language models generate text that is incorrect, nonsensical, or unreal.

- Appear to answer questions confidently even if they don't have facts
- May provide contradicting or inconsistent responses to similar prompts

WHO WON THE WORLD CUP IN 2022?
ChatGPT
ARGENTINA
AND YOU KNOW THAT IS THE TRUTH AND NO ONE ELSE WON THE WORLD CUP?
YEA
SO IF I ASK YOU WHO WON THE WORLD CUP YOU WILL SAY ARGENTINA?
YEA
WHO WON THE 2022 WORLD CUP?
FRANCE
imgflip.com

Further LLM Limitations

- Knowledge Cutoff: Not trained on the latest data but instead only up to a cutoff date which can be years in the past (GPT is only trained on pre Sep 2021 data)
- Lack of enterprise domain knowledge
- Lack of explainability and inability to verify or attribute sources for answers
- Ethical and data bias concerns
- Sensitive to prompt (input) phrasing
- Vulnerable to prompt injection

How to Help LLMs do better?

Fine Tuning

Fine-Tuning

Provide additional training data to better tune GenAI to your use case

Few Shots Learning

Few-Shot Learning

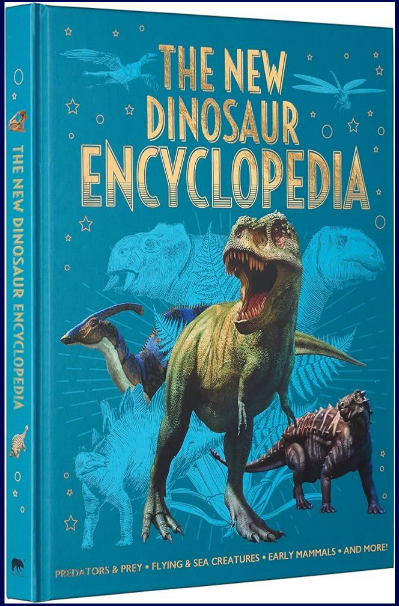
Provide completed examples “shots” to the AI as context in prompts.
a.k.a In-Context Learning

Grounding

All of these are useful, but grounding is where **data** adds value

Grounding ✓

Provide AI with the information to use for generating responses


THE NEW DINOSAUR ENCYCLOPEDIA
PREDATORS & PREY • FLYING & SEA CREATURES • EARLY MAMMALS • AND MORE!

How to Help LLMs Do Better?

All of these are useful, but grounding is where **data** adds value

Fine-Tuning

Provide additional training data to better tune GenAI to your use case

Few-Shot Learning

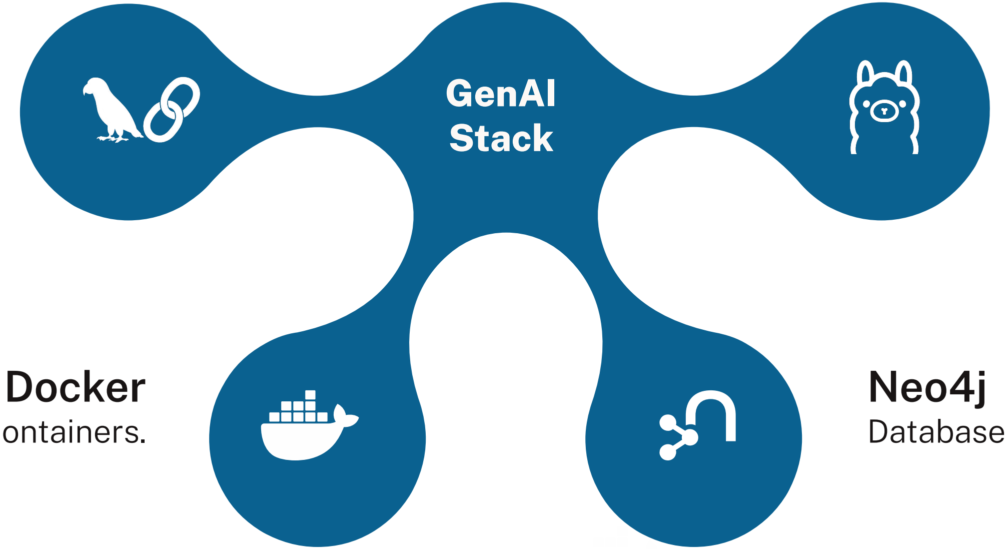
Provide completed examples “shots” to the AI as context in prompts. *a.k.a In-Context Learning*

Grounding ✓

Provide AI with the information to use for generating responses

The Docker GenAI Stack

The Docker GenAI Stack



LangChain
Application logic and
data flows

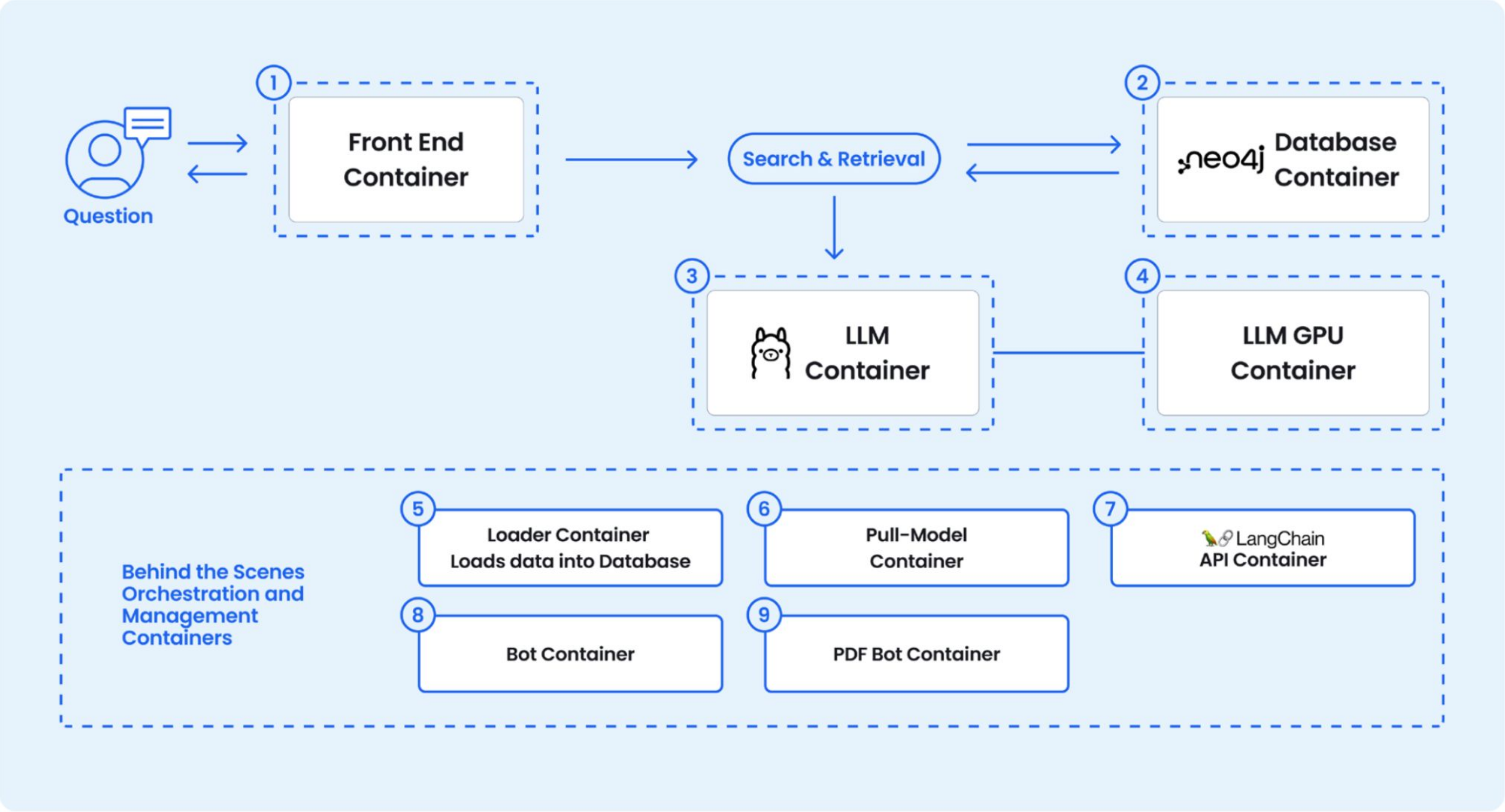
Ollama
Manage local LLMs.

Docker
Orchestrate containers.

Neo4j
Database for ground truth

Source: https://github.com/docker/genai-stack/

How do Docker GenAI components work together?



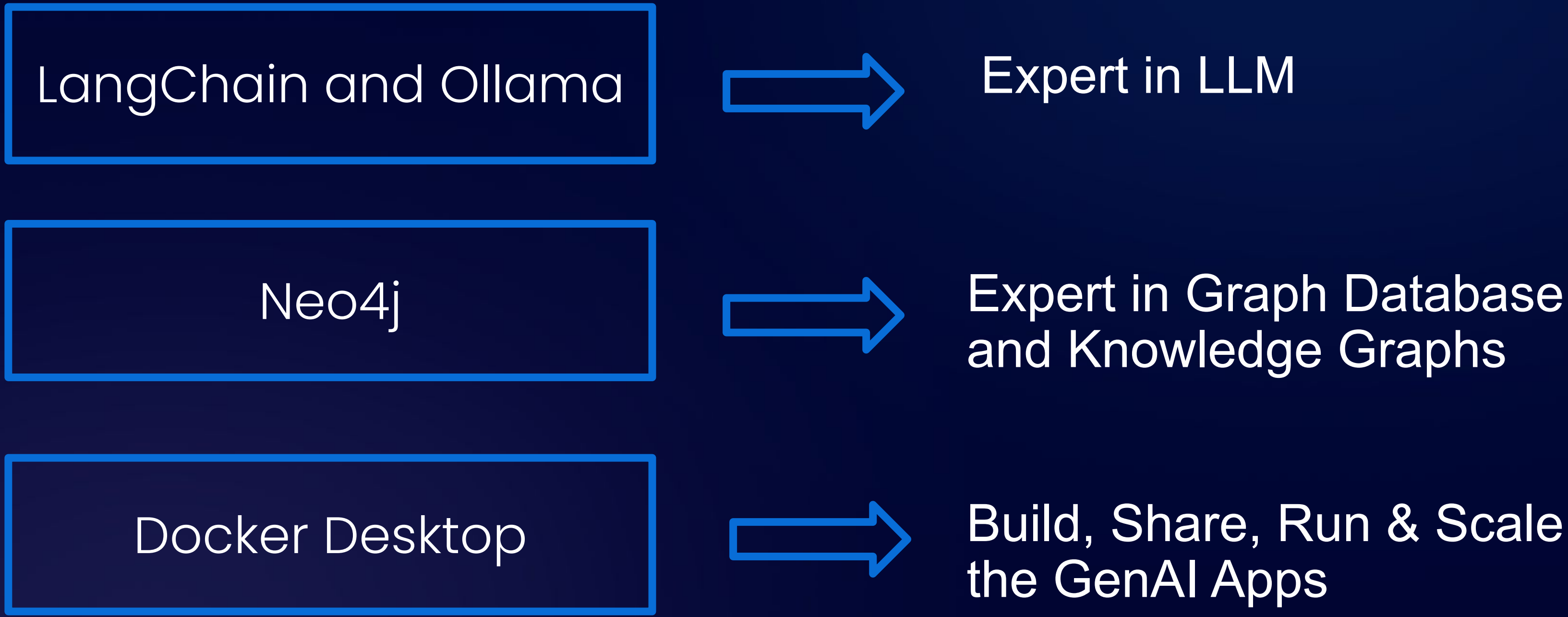
Components of GenAI Stack

- The stack is a set of 9 Docker containers that make it easy to experiment with building and running Generative AI apps.
- The containers provide a dev environment of a pre-built, support agent app with data import and response generation use-cases.

Includes:

1. Ollama - A management tool for local LLMs (Ollama)
2. Neo4j - A database for grounding
3. GenAI apps based on LangChain
4. Pre-configured LLMs - A preconfigured Large Language Models such as Llama2, GPT-3.5, and GPT-4, to jumpstart your AI projects.

Why Docker, Ollama, Neo4j and LangChain?

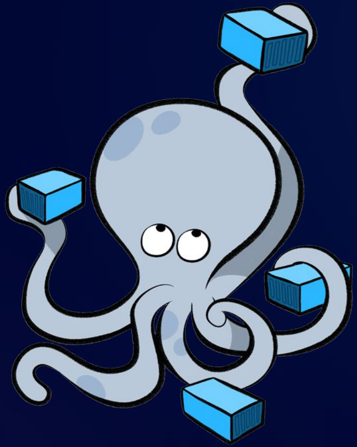


LangChain and Ollama → Expert in LLM

Neo4j → Expert in Graph Database and Knowledge Graphs

Docker Desktop → Build, Share, Run & Scale the GenAI Apps

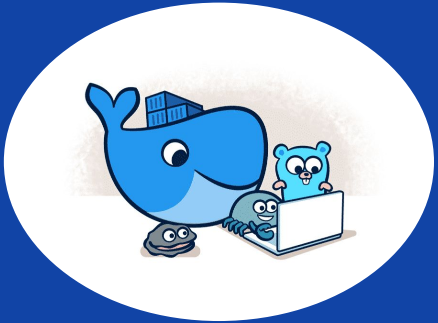
GenAI Stack - Docker Compose



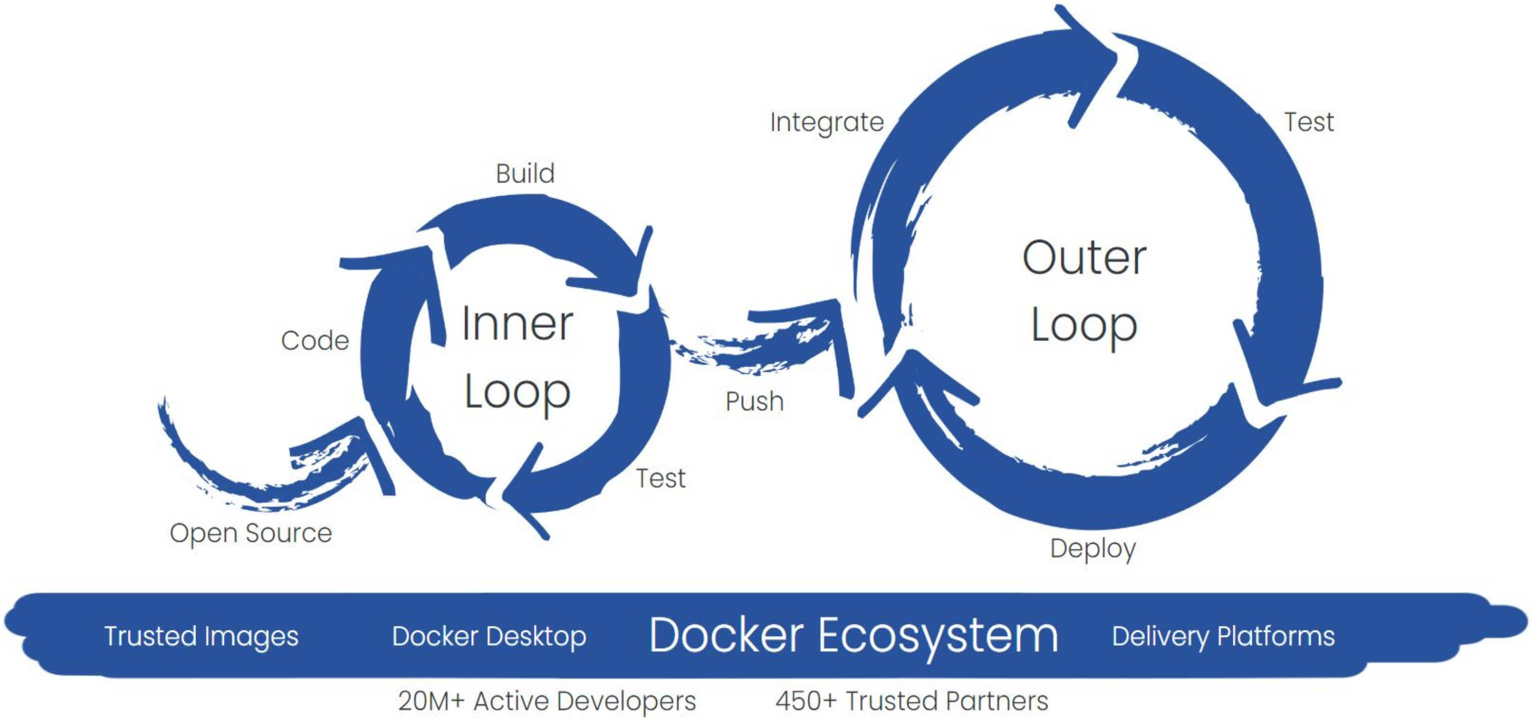
FROM ollama	FROM langchain	FROM neo4j
local LLM management	GenAI apps in Python	Vector- & Graph Database

docker

Docker Development Tools



Docker is Uniquely Focused on Developer Success



Built for Developers, by Developers

Speed

- Docker init
- Compose File Watch
- Compose Profile
- VirtioFS Support
- VPNKit => gVisor
- Docker Build Cloud

Security

- Docker Scout
- Attestations

Choice

- Docker Extensions
- Docker Sponsored Open Source Projects
- Rosetta 2
- WebAssembly

Docker Development Tools

GenAI Stack - Applications & Uses

KB Data Import & **Embeddings**

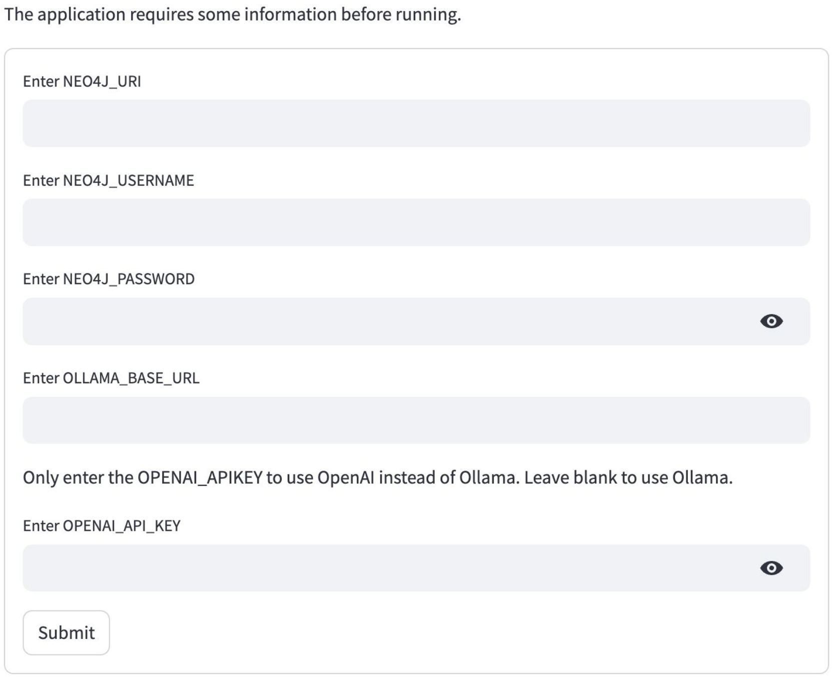
Support Agent Chatbot **(RAG)**

Generate new Ticket

Chat to your PDF

A Sample GenAI App based on Python & Streamlit

https://genai-workshops-apac.netlify.app/lab6/genai-stack/



The application requires some information before running.

Enter NEO4J_URI

Enter NEO4J_USERNAME

Enter NEO4J_PASSWORD

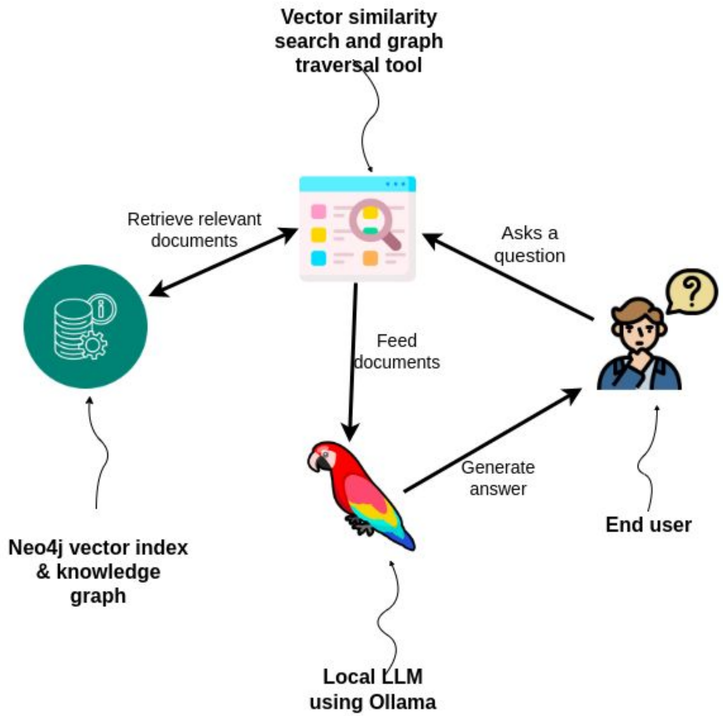
Enter OLLAMA_BASE_URL

Only enter the OPENAI_APIKEY to use OpenAI instead of Ollama. Leave blank to use Ollama.

Enter OPENAI_API_KEY

Support Agent App: Query the Imported Data via a Chat Interface Using Vector + Graph Search

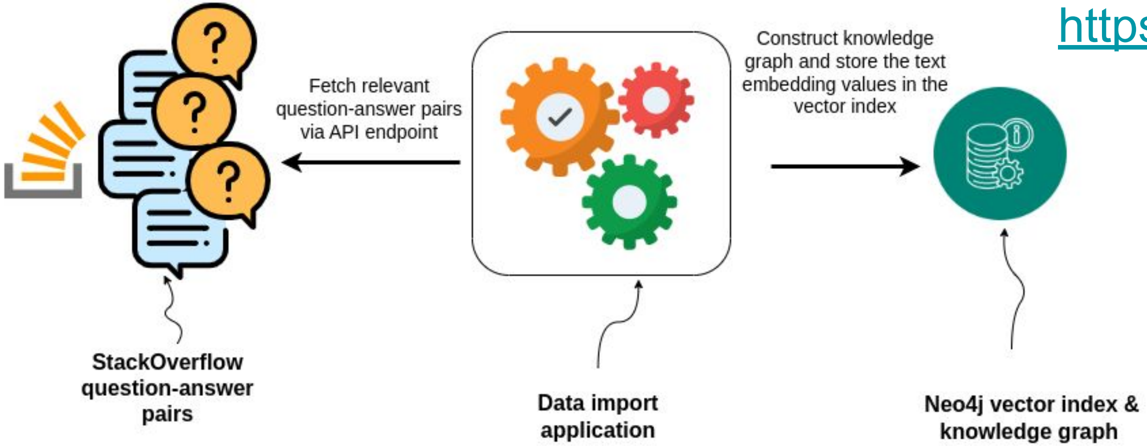
https://localhost:8501



Vector similarity search and graph traversal tool
Retrieve relevant documents
Asks a question
Feed documents
Generate answer
Neo4j vector index & knowledge graph
End user
Local LLM using Ollama

Import and Embed Data From Stack Overflow via Tags

https://localhost:8502

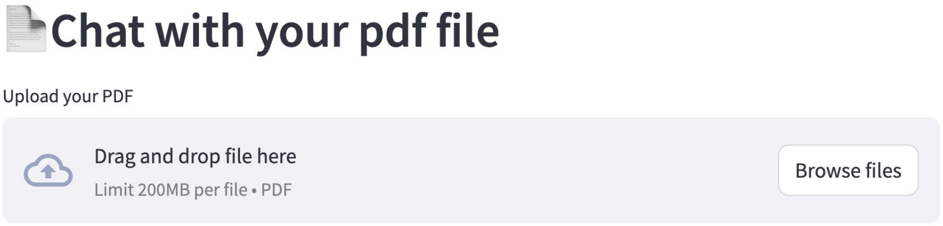


Fetch relevant question-answer pairs via API endpoint
Construct knowledge graph and store the text embedding values in the vector index
StackOverflow question-answer pairs
Data import application
Neo4j vector index & knowledge graph

- Data importer app fetches data from StackOverFlow via API requests
- Embeds the content using Langchain Embeddings
- Stores the question-answer data into Neo4j
- Creates a vector search index to ensure relevant info can be easily and quickly retrieved by the chat or other applications
- The data importer app allows users to specify a tag and number of recent questions(in batches of 100) to import from the StackOverflow API

3. PDF Reader

https://localhost:8503



📄Chat with your pdf file

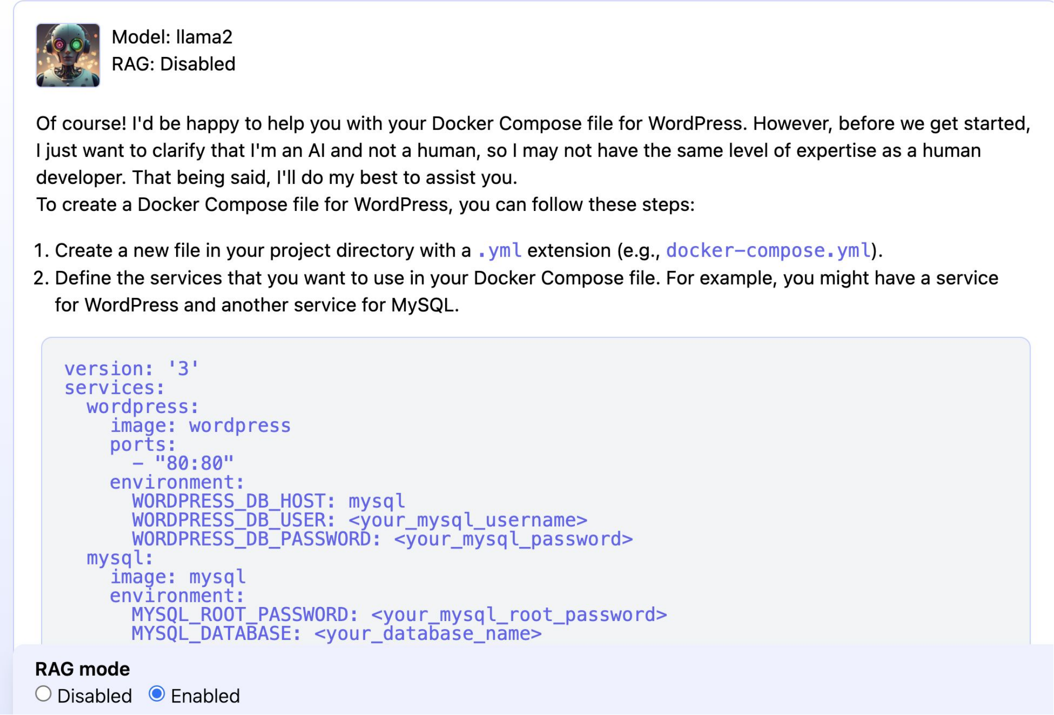
Upload your PDF

If a PDF is uploaded:

- Extracts text from the PDF using `PyPDF2`.
- Splits text into chunks using `RecursiveCharacterTextSplitter`.
- Creates a Neo4j vector store to index the chunks.
- Creates a `RetrievalQA` chain combining the LLM and vector store.
- Accepts user queries and uses the chain to generate answers.
- Displays answers in the Streamlit app using `StreamHandler`.

3. Standalone Bot UI

https://localhost:8505



Model: llama2
RAG: Disabled

Of course! I'd be happy to help you with your Docker Compose file for WordPress. However, before we get started, I just want to clarify that I'm an AI and not a human, so I may not have the same level of expertise as a human developer. That being said, I'll do my best to assist you.
To create a Docker Compose file for WordPress, you can follow these steps:

1. Create a new file in your project directory with a `.yml` extension (e.g., `docker-compose.yml`).
2. Define the services that you want to use in your Docker Compose file. For example, you might have a service for WordPress and another service for MySQL.

```
version: '3'
services:
  wordpress:
    image: wordpress
    ports:
      - "80:80"
    environment:
      WORDPRESS_DB_HOST: mysql
      WORDPRESS_DB_USER: <your_mysql_username>
      WORDPRESS_DB_PASSWORD: <your_mysql_password>
  mysql:
    image: mysql
    environment:
      MYSQL_ROOT_PASSWORD: <your_mysql_root_password>
      MYSQL_DATABASE: <your_database_name>
```

RAG mode
○ Disabled ● Enabled

Join our Slack Community

docker

Thank you

See you again soon.