DAY 02 MODULE

# The Brain: LLMs & Prompts

Mastering the Engine of AI: Models, Economics, and Engineering.

# Guidelines

- Attendance is mandatory for all 5 sessions
- Hands on activity is mandatory
- 15 min break at 10:30PM
- QnA session at the end (10-15 min)
- Feel free to drop your questions in chat
- There will be quizzes in-between, drop your answers in chat

# 5 day roadmap

(2-5PM)

✓ 1

**Shift**

Agentic Thinking vs. Chatbots

2

**Brain**

LLM Types & Prompting

3

**Hands**

Function Calling & Tools

4

**Memory**

RAG & Vectors

🏁

**Build**

End to end pipeline & Capstone

# Today’s Roadmap

01 **The Landscape**

Proprietary (GPT-4o) vs. Open Source (Llama/Gemma).

02 **Economics**

Tokenization, Costs, and Latency.

03 **Prompt Engineering**

Deep dive into the R.O.L.E.S. Framework.

04 **Advanced Reasoning**

Chain of Thought (CoT) and Few-Shot techniques.

## Hands on

# Day 1 recap: quiz

## Which architecture is used by LLMs

# Day 1 recap: quiz

## Smallest unit that can be processed by LLMs

________

# Module 1
# The LLM Landscape

Choosing the right brain for the job.

# Quick concept check

## What is PII?

- Aadhar number
- Person Name (Private Information)

# Quick check

## Batch vs Real-time processing?

- Current weather
- Historic data analysis

# The Great Divide

### Closed Source (API)

**Models:** GPT-4o, Claude 3.5, Gemini 1.5.

**Pros:** Maximum intelligence, zero maintenance, massive context windows.

**Cons:** Data privacy risks, cost at scale, vendor lock-in.

### Open Source (Local)

**Models:** Llama 3, Mistral, Gemma.

**Pros:** Total data privacy, free (hardware only), fine-tunable.

**Cons:** Requires GPU management, slightly lower logic capabilities.

# The Heavyweight: GPT-4o ( Gemini 1.5 pro)

### ”Omni” Model

GPT-4o is the current state-of-the-art (SOTA) for general reasoning.

- **Multimodal:** Understands text, audio, and images natively.
- **Reasoning:** Exceptional at complex instruction following.
- **Use Case:** Complex coding, creative writing, analyzing messy data.

4o

# The Challenger: Llama 3

## The Open Standard

Released by Meta, Llama 3 performs shockingly well for its size.

- **8B Version:** Fast enough to run on a laptop. Great for summarization.
- **70B Version:** Rivals GPT-4 in text tasks but requires server-grade GPUs.
- **Use Case:** Private internal tools, high-volume classification.

# Quick Check: Data Privacy

**Which Industry is more likely to have private data?**

Health care

Marketing

Ecommerce

# Critical Factor: Data Privacy

### The ”PII” Rule

If your data contains **Personally Identifiable Information (PII)** like medical records or financial data, you usually cannot send it to a public API like OpenAI.

# Activity: Match the Model

### Scenario A

Analyzing highly sensitive legal contracts for a bank.

### Scenario B

Building a generic travel planner chatbot for a public website.

### Scenario C

Summarizing 10,000 news articles per day cheaply.

# How to access open source models?

Download Ollama

Huggingface

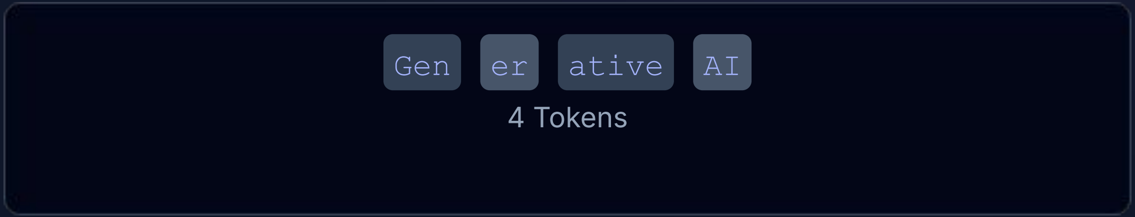
# Module 2
# The Economics of AI

Tokens, Costs, and Latency.

# How LLMs Read: Tokenization

LLMs do not see words. They see **Tokens** (chunks of characters).

- **Rule of Thumb:** 1,000 Tokens ≈ 750 Words.
- **Common Word:** "Apple" = 1 Token.
- **Complex Word:** "Ingeniously" = 3 Tokens [In, gen, iously].



Gen er ative AI

4 Tokens

# Input vs. Output Cost

API providers charge differently for reading vs. writing.

| Model | Input Cost (Per 1M Tokens) | Output Cost (Per 1M Tokens) |
|---|---|---|
| GPT-4o | $5.00 | $15.00 |
| GPT-3.5 Turbo | $0.50 | $1.50 |
| Llama 3 (API) | ~$0.10 | ~$0.10 |

*Prices are illustrative approximations.

# Latency (Speed)

### Time to First Token (TTFT)

How long user waits before text starts appearing. Crucial for chatbots.

### Tokens Per Second (TPS)

How fast the text generates. Higher is better for large document generation.

**Trade-off:** Smarter models (GPT-4o) are usually slower than smaller models (GPT-3.5/Llama 8B).

# Quiz: Economics

### Question 1:

Which operation is typically more expensive in API costs?

A Input (Reading user prompt)

[✓ B] Output (Generating the answer)

C They are always the same cost

# Module 3
# Prompt Engineering

Programming with English.

# Garbage In, Garbage Out (GIGO)

LLMs are not mind readers. They are **Pattern Matchers**.

### Vague Prompt

"Write an email."

*(Model guesses context, likely generic.)*

### Engineered Prompt

"Write a polite decline email to a vendor selling paper, referencing our budget freeze."

*(Model has constraints to follow.)*

# Zero-Shot vs. Few-Shot

## Zero-Shot

Asking the model to do something without examples.

```
"Classify this: 'I loved it!' -> ?"
```

## Few-Shot (The Cheat Code)

Giving examples to guide the pattern.

```
"Hate it -> Negative" "Okay -> Neutral" "Loved it
-> ?"
```

# Module 4
# The R.O.L.E.S. Framework

The Standard Operating Procedure for Prompts.

# R = Role

### Why it matters

Giving the AI a persona primes specific subsets of its training data.

### Example

❌ "Explain Quantum Physics."

✅ "Act as a Kindergarten Teacher. Explain Quantum Physics using analogies about toys."

# O = Objective

State the specific goal clearly. Use strong action verbs.

### Draft

Create new content.

### Summarize

Condense information.

### Analyze

Find patterns or errors.

# L = Limitations

Constraints breed creativity and precision.

- **Length:** "Max 3 sentences."
- **Exclusions:** "Do not use technical jargon."
- **Scope:** "Focus only on the financial aspects."

# E = Examples

Show, don't just tell. This is "Few-Shot" prompting implemented.

```
User: Create a catchphrase. Example Input: Nike Example Output: Just Do It. Input: McDonald's Output: I'm Lovin' It.
Input: [Your Brand]
```

# S = Style / Structure

### Style (Tone)

Professional, Witty, Sarcastic, Empathetic.

### Structure (Format)

Markdown table, JSON, Bullet points, Python list.

# Activity: The Fix-It Lab

**Bad Prompt:** **”Write a blog about AI.”**

How do we apply R.O.L.E.S to fix this? (Discuss)

Hint: Role=Tech Journalist, Limitation=500 words...

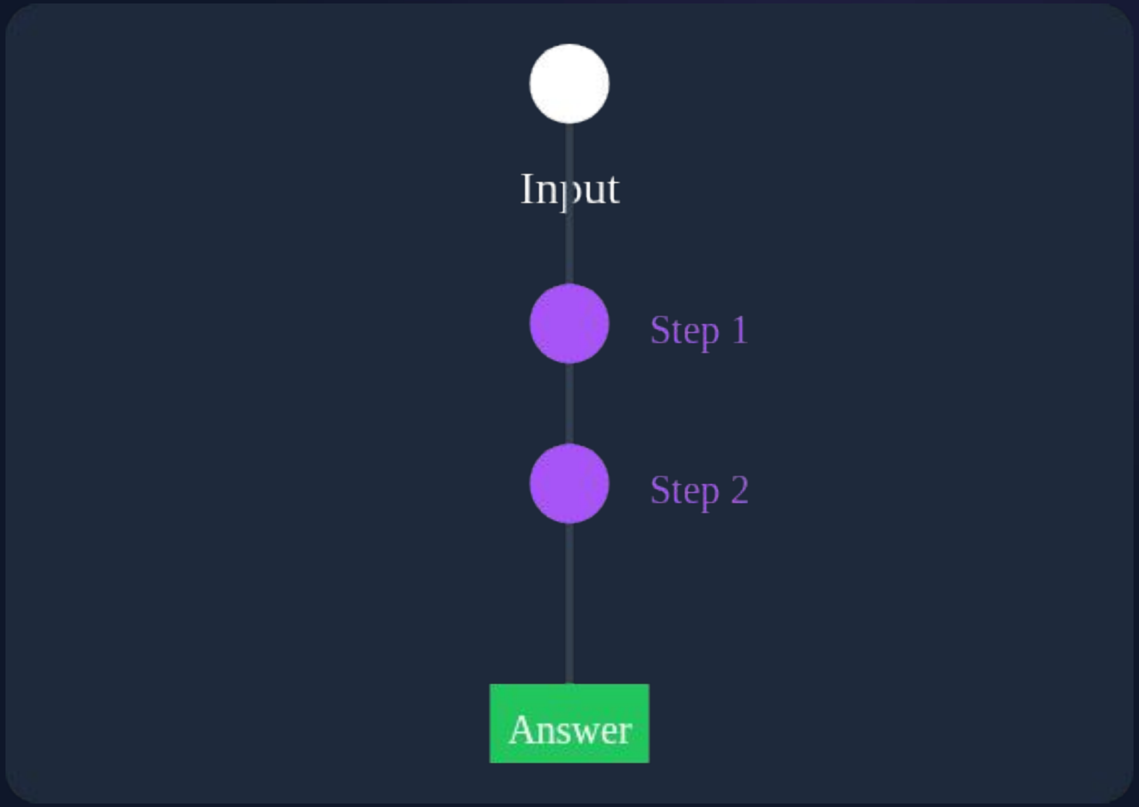
# Module 5
# Advanced Reasoning

Getting the model to "Think".

# Chain of Thought (CoT)

By asking the model to **"think step-by-step"**, you force it to generate intermediate reasoning tokens.

This improves accuracy on math and logic problems significantly.



# Self-Consistency

### ”The Council of Elders”

Instead of asking once, ask the model to generate **3 different reasoning paths**.

If 2 out of 3 paths lead to the same answer, pick that one.

```
Path A -> Answer: 42 Path B -> Answer: 42 Path C -> Answer: 40 Result: 42 (Majority Vote)
```

# Module 6
# Hands-On Lab

Re-engineer and Compare.

# The 5 Scenarios

| # | Task Type | Bad Prompt |
|---|---|---|
| 1 | Creative | "Write a poem about dogs." |
| 2 | Coding | "Write python code for data." |
| 3 | Business | "Email my boss about being late." |
| 4 | Summary | "Summarize this." (Paste article) |
| 5 | Logic | "Solve this riddle." |

# Day 2 Summary

### LLM Types

API vs Open Source

### Economics

Tokens & Costs

### Engineering

R.O.L.E.S Framework

### Reasoning

Chain of Thought

# Day 2 Complete

Tomorrow: The Tools (Function Calling).

Q & A