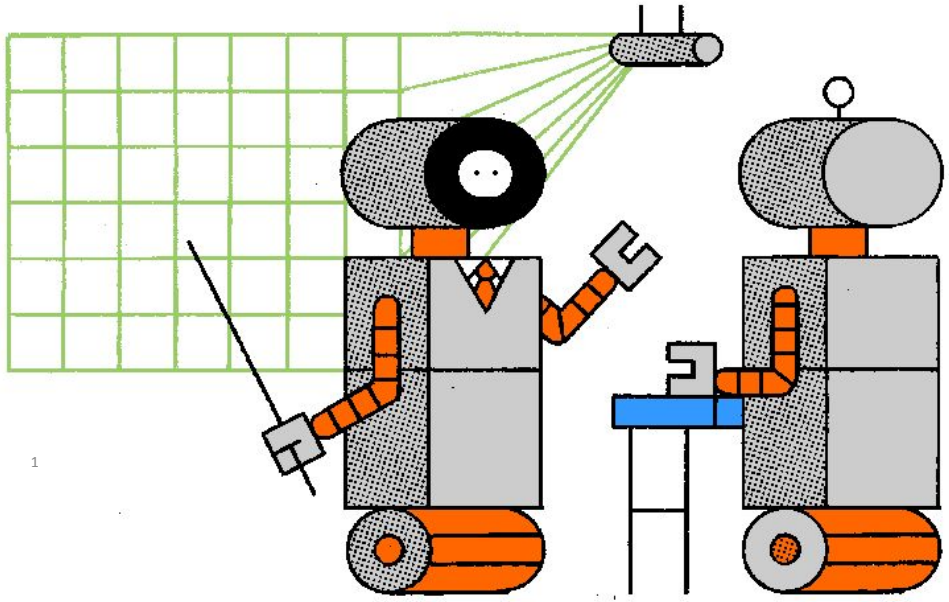
traversaal.ai

Let's build Agents

(a.k.a Automated Workflows)

Module 2

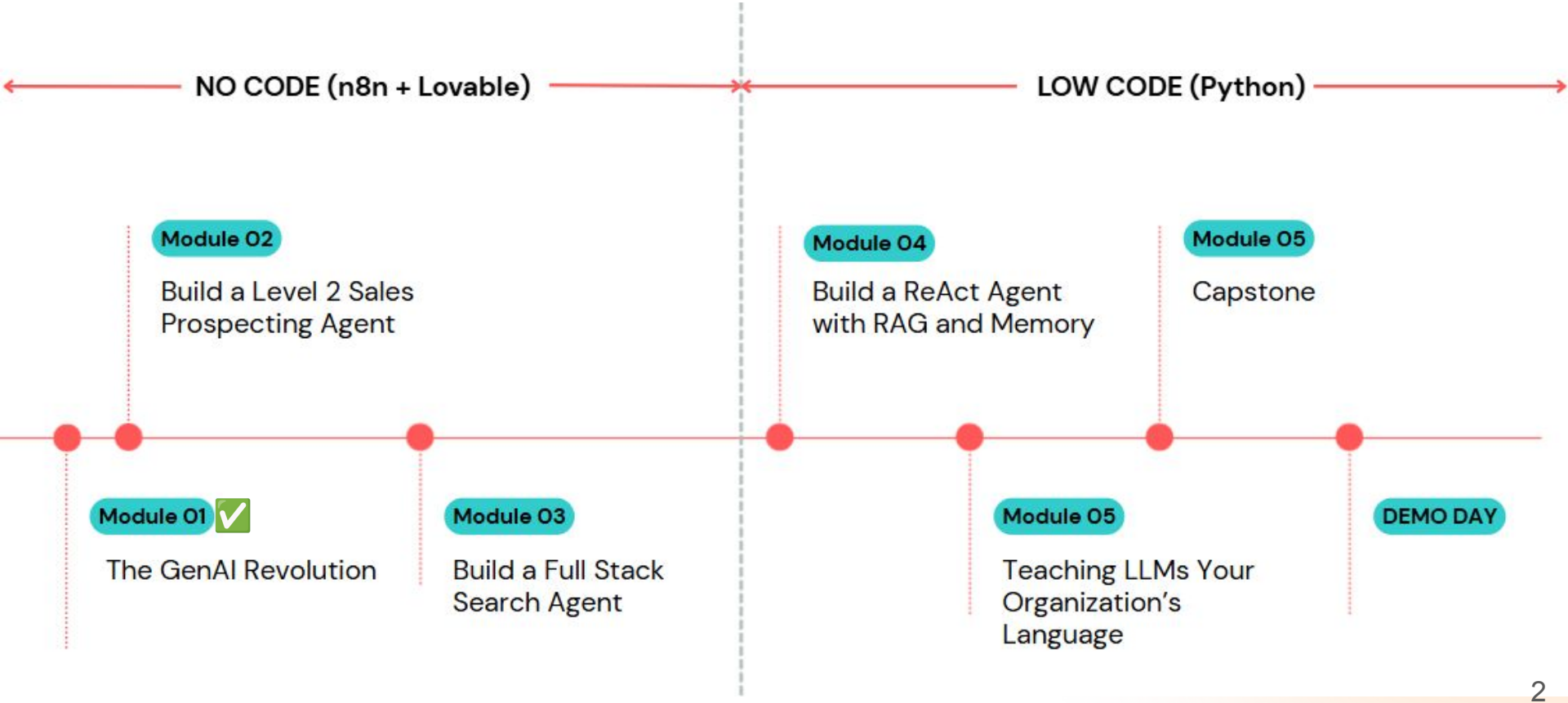


Hamza Farooq

Founder & CEO

Course OverLook

Building Gen AI Agents for Enterprise: Beyond the Hype



Recap from Module 01

- The GenAI Inflection Point: Understanding the Fundamental Shift
- Large Language Models (LLMs): From Foundation to LLMs
- GenAI in Organizations
- Future of GenAI: Agents and Agentic Architectures

Learning Outcomes for Module 02

- **Prompt Engineering: Role, Format, Style**
  - Learn how to craft prompt stacks that guide behavior and enable better agent control.
- **Context Engineering: Chunking, Windowing, Input Design**
  - Understand how to inject relevant information into agents effectively using input design techniques.
- **Evaluating Agent Behavior in Real Time** (in lessons)
  - Explore how to test and monitor agent performance using qualitative and quantitative signals

Learning Outcomes for Module 02

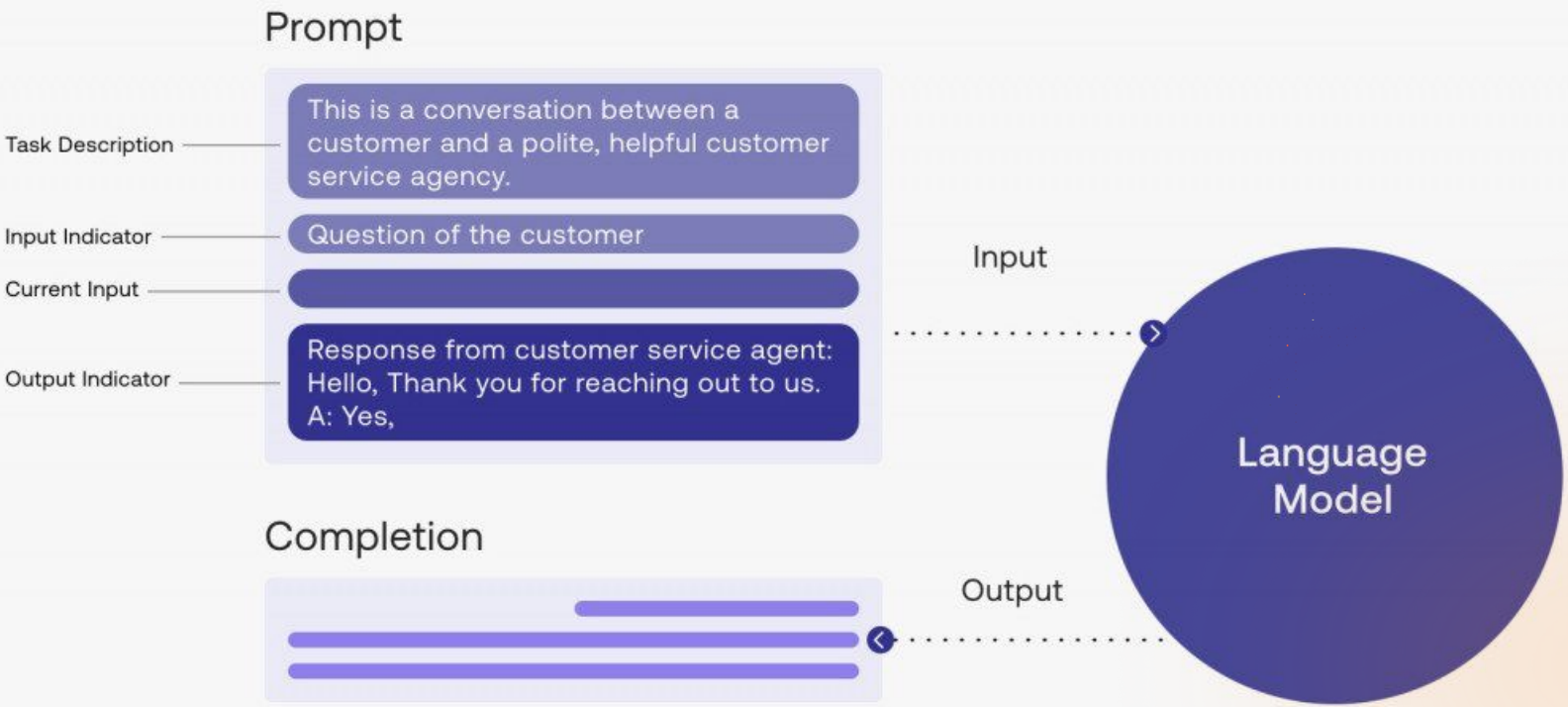
- **Tools Deep Dive: n8n and No-Code Workflows**
  - Get familiar with n8n as the core orchestration platform for chaining together inputs, logic, and APIs.
- **Sales Prospecting Agent Architecture**
  - Use a concrete use case to scaffold understanding of how real-world vertical agents are structured.
- **Build and Test Your Agent in n8n**
  - Assemble your first working prototype that demonstrates autonomous workflow execution.

01: Prompt Engineering

Prompt Engineering

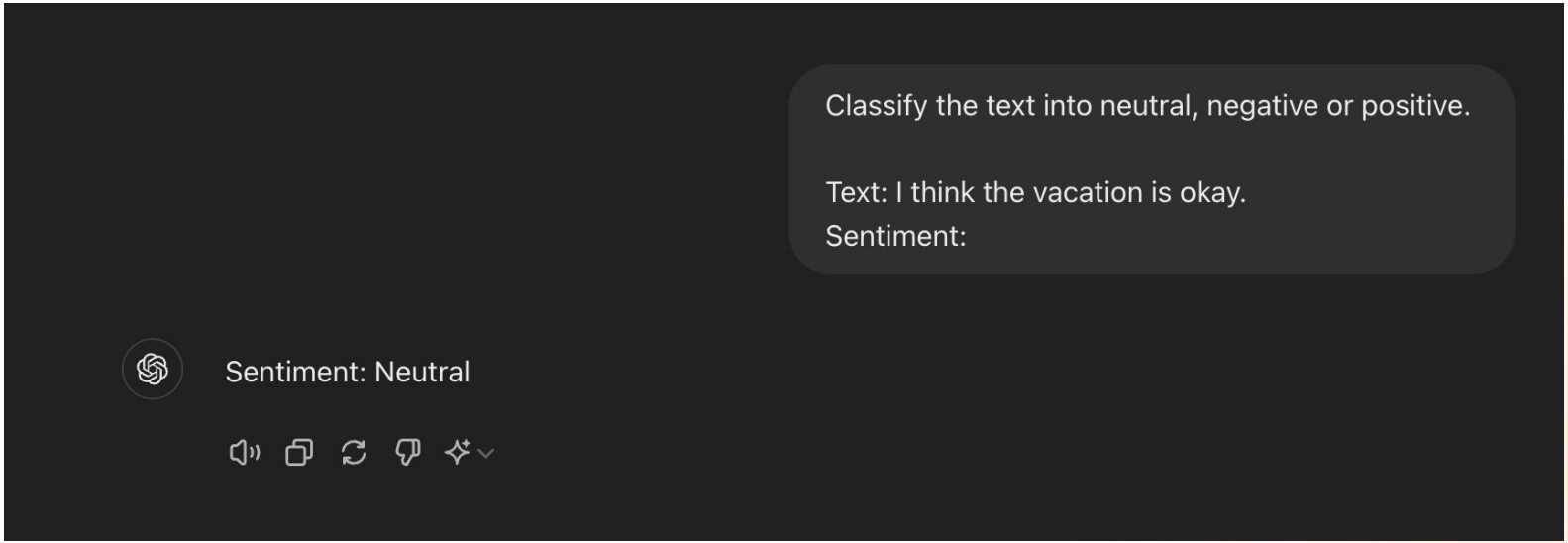
Prompt engineering involves designing and refining language model prompts to achieve specific desired outputs. It includes crafting prompts that provide clear instructions, context, or constraints to guide the model's responses.

Prompt Engineering



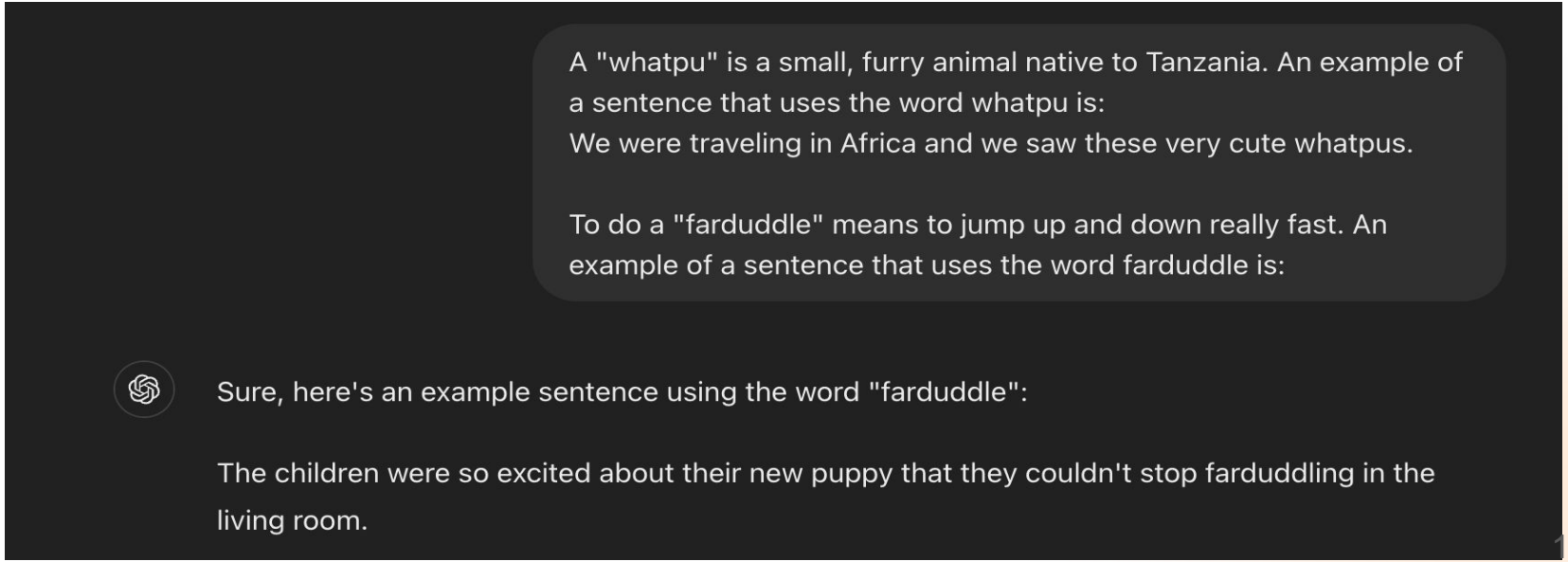
Zero Shot Prompting

- Direct instruction given to LLM without any examples or demonstrations.
- Model relies on its pre-trained knowledge to perform the task.
- Suitable for straightforward tasks.



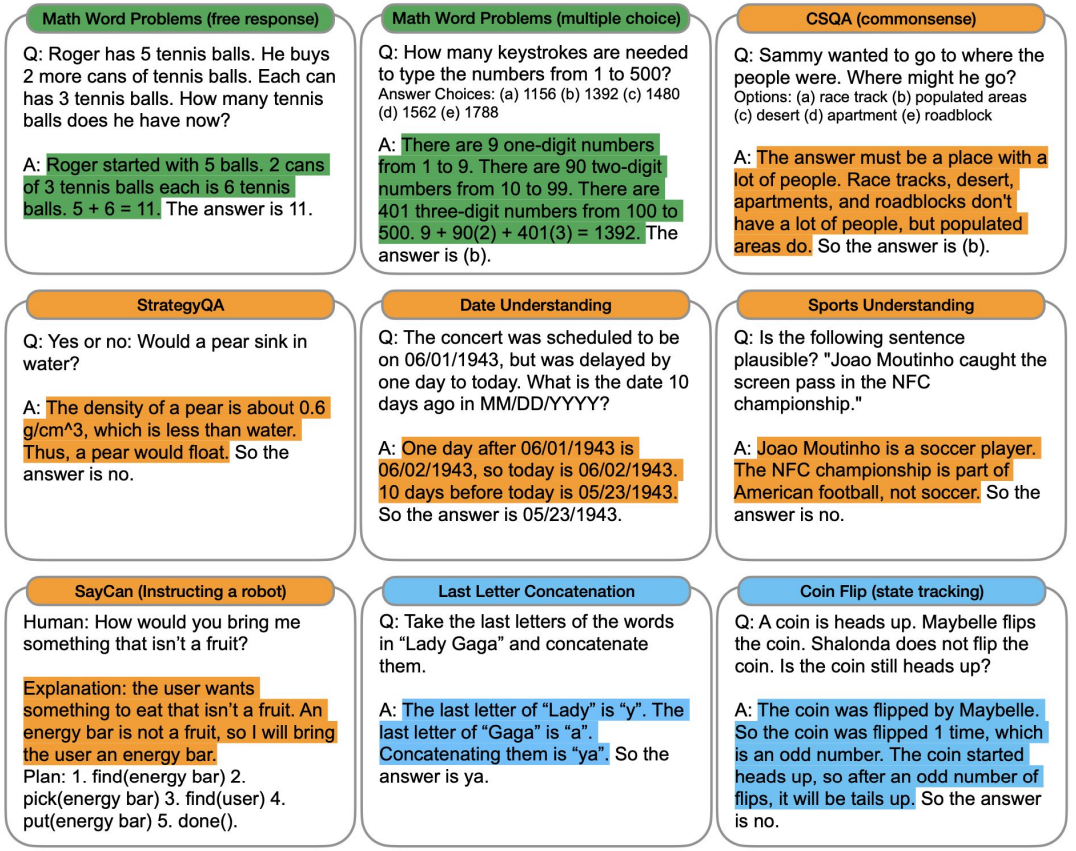
Few Shot Prompting

- Provides demonstrations within the prompt to enable in-context learning.
- Steers the model towards better performance for subsequent inputs.
- Demonstrations serve as conditioning to guide the model's response.



Chain of Thought Prompting

Chain-of-thought (CoT) prompting enhances the reasoning capabilities of LLMs by encouraging them to break down their reasoning into a series of intermediate steps

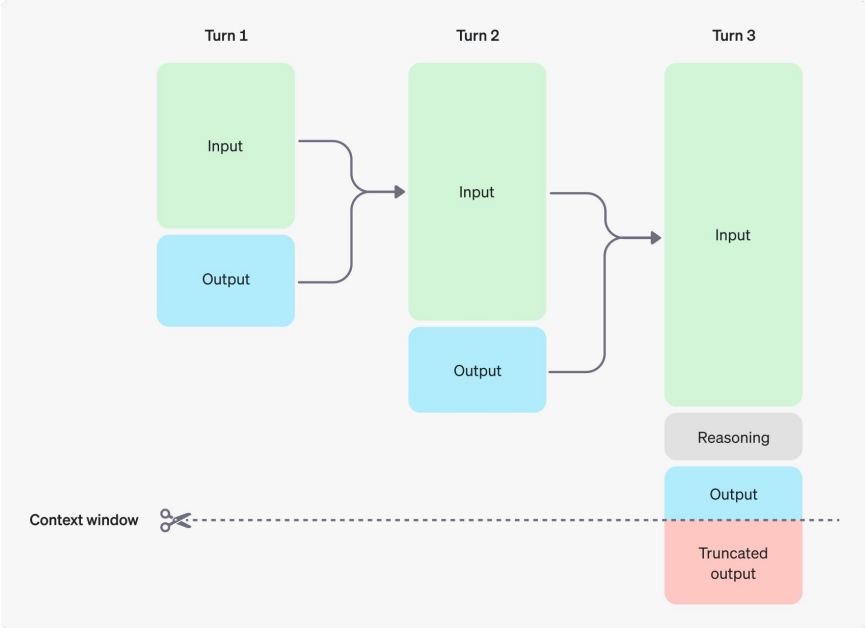


Source

Reasoning Models

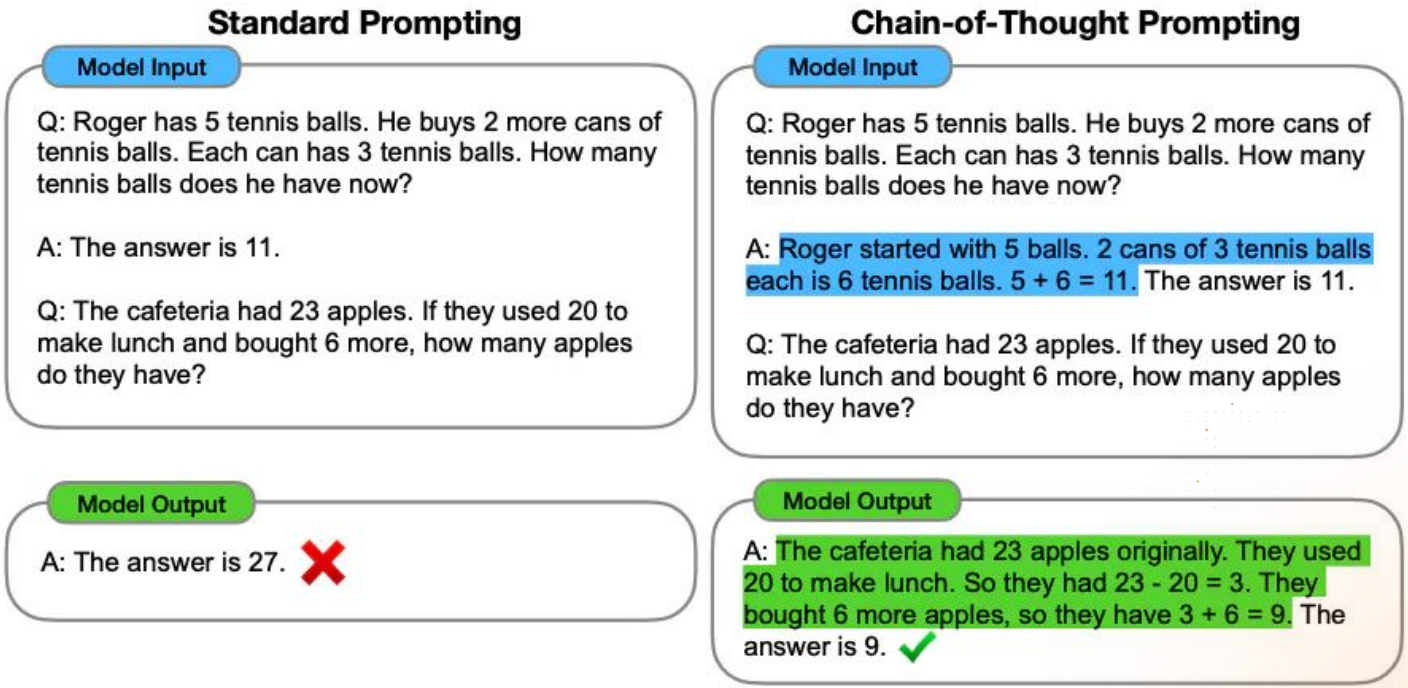
These models introduce internal "reasoning tokens" to break down prompts and explore solutions step-by-step.

Built on existing LLMs, they leverage reinforcement learning and Chain-of-Thought (CoT) to guide their processing.



Source

Chain of Thought Prompting



Source

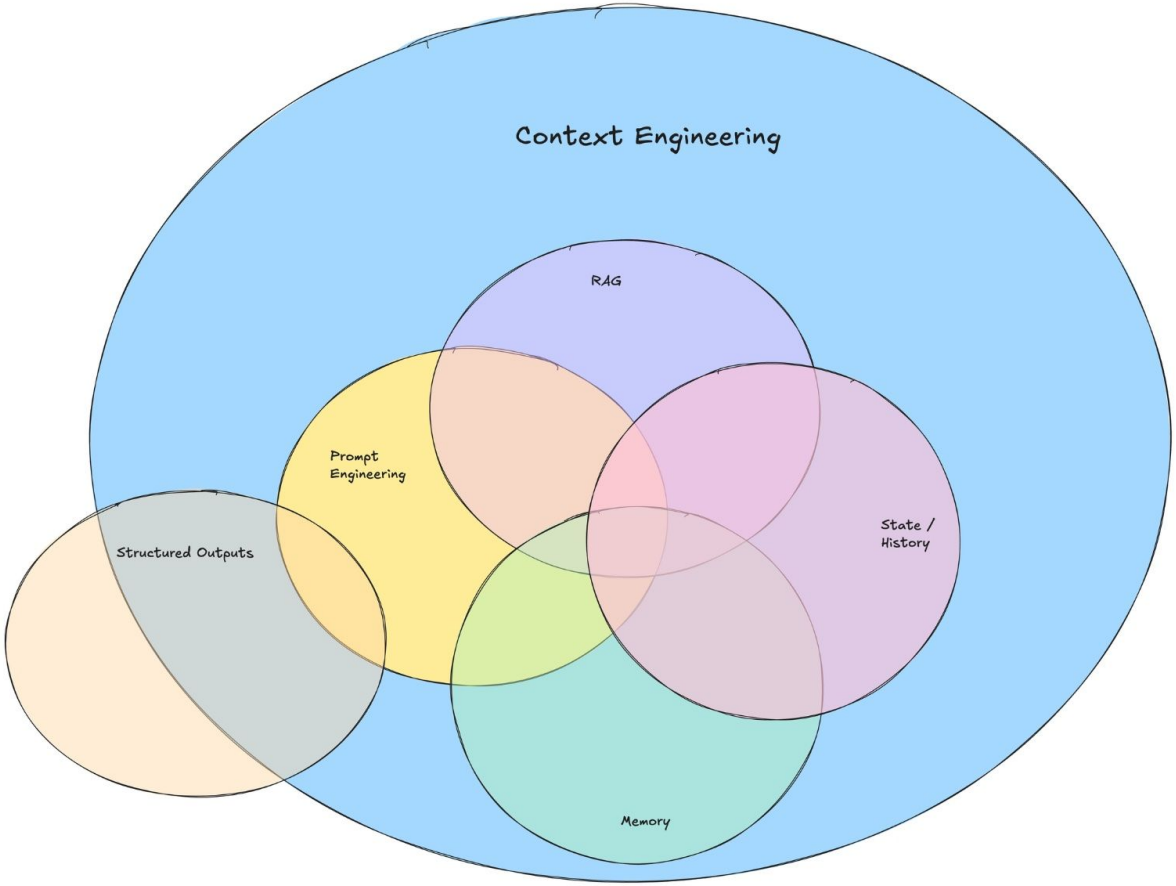
02: Context Engineering

Context Engineering

Context engineering involves designing dynamic systems that manage how information and tools are presented to the LLM over time.

This shift is crucial as applications increasingly rely on multi-agent setups, tool integrations, and long-term interactions, requiring adaptive context management rather than static prompt design.

Context Engineering



Source

Context engineering vs Prompt Engineering

- **Prompt Engineering**: Focuses on *how you instruct* the LLM.

- **Context Engineering**: Focuses on *what information you provide* to the LLM.

In short, Prompt Engineering is a subset of Context Engineering and Prompt lives inside the container that context builds.

Types of Context Engineering

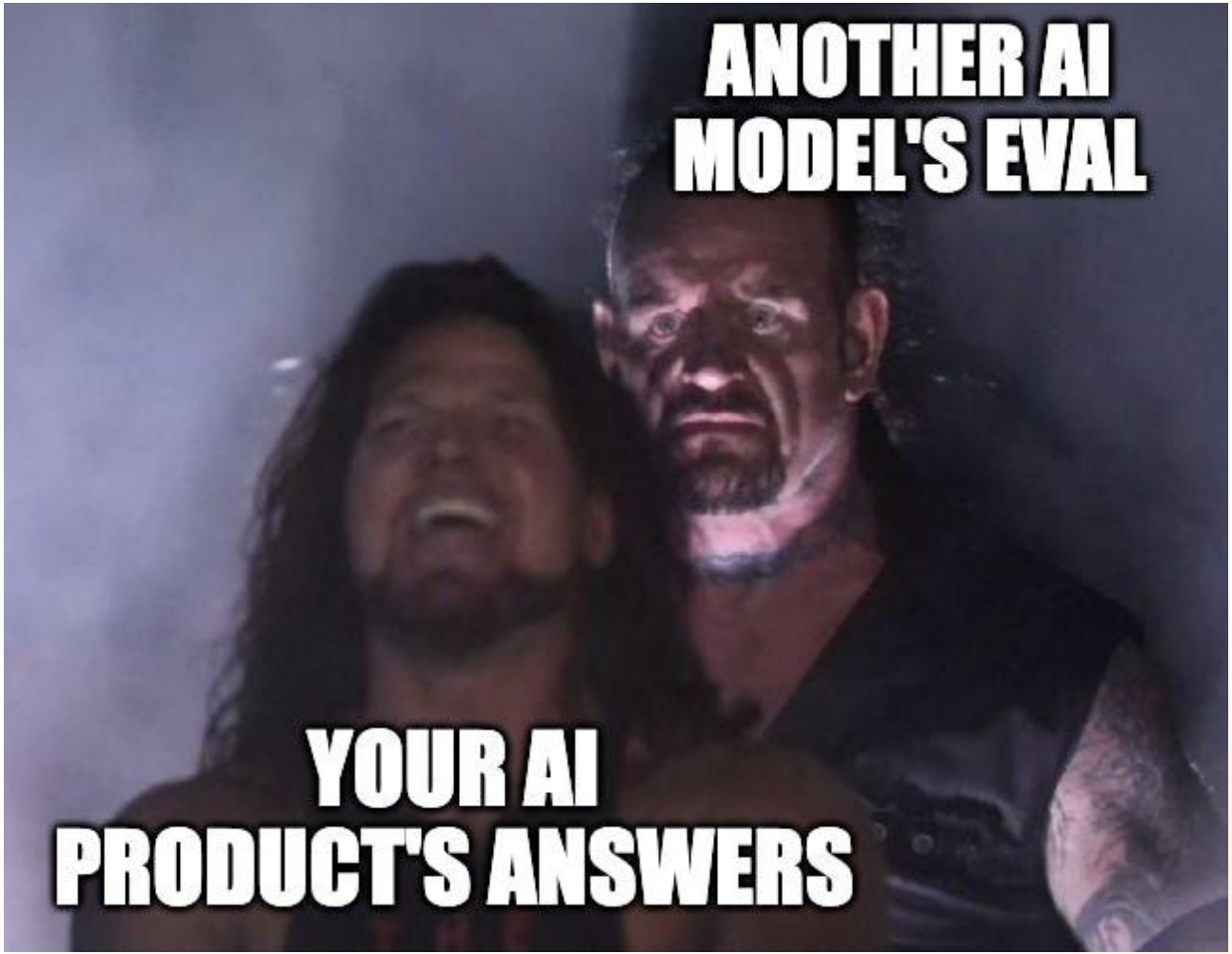
Context Engineering optimizes an LLM's environment by:

- **Refining Instructions:** Crafting all prompts, system-level to specific tasks and examples.
- **Managing Knowledge & Memory:** Integrating external data (RAG) and handling agent memory.
- **Enabling Agent Workflows:** Structuring inputs/outputs and defining external tool use.

Importance of Context Engineering

- When LLM-powered agents make mistakes, it's rarely the underlying model's inherent flaw.
- More often than not (especially with improving models), issues stem from the LLM not being passed the appropriate context to generate a good output.
- "Bad context" primarily means two things: Missing Information or Poor Formatting.
- Ultimately: Mastering context engineering is paramount for building reliable, intelligent, and consistently high-performing AI agents.

03: Evaluating Agent Behaviour


ANOTHER AI
MODEL'S EVAL
YOUR AI
PRODUCT'S ANSWERS

Evaluating LLMs and Agents: Why?

- Critical for building dependable AI systems in real-world applications.
- Provides data-driven insights to refine prompts, context, and agent logic.
- Identify and mitigate biases, hallucinations, or unexpected behaviors early.
- Directly links agent performance to key business objectives and user satisfaction.

Evaluating LLMs and Agents: What?

Stages of evaluation

1. **Overall Response Metrics**

    a. Is the final response of the agent or agentic system correct and complete?
    b. Does the output align with or contradict the expected "ground truth"?

1. **Tool Level Metrics**

    b. Did the tool give the correct output ?
    c. Did the search tool produce the correct results ?
    d. Is the code tool's output correct ?

Metrics for LLM Evaluation

How should we test the inference speed?

- **Time To First Token (TTFT):** Time required to process the prompt and then generate the first output token
- **Inter-token latency (ITL):** Average time between consecutive tokens
- **End-to-end Latency:** Total time taken to generate the entire response
  ~ = Average output length of tokens * Inter-token latency
- **Throughput:** Number of output tokens per second

Metrics for LLM Evaluation

How should we test the response quality?

- **Relevance:** Is the output directly on-topic and addressing the user's need?
- **Coherence & Fluency:** Does the response flow naturally, is it grammatically correct, and easy to understand?
- **Completeness:** Does it provide all necessary information without being overly verbose?
- **Tone & Style:** Does the response match the desired persona, brand voice, or interaction context (e.g., formal, friendly, concise, empathetic)?
- **Conciseness:** Is the information delivered efficiently and to the point?

Evaluation Metrics for Agents

Common Metrics used

- Accuracy
- Precision
- Recall
- F1 Score

Other Metrics

- Step Count : Number of steps taken to solve a query
- Tool Call Efficiency : Ratio of useful vs redundant tool calls.
- Time to Completion: Total time taken to respond to the query

Tool -level metrics used

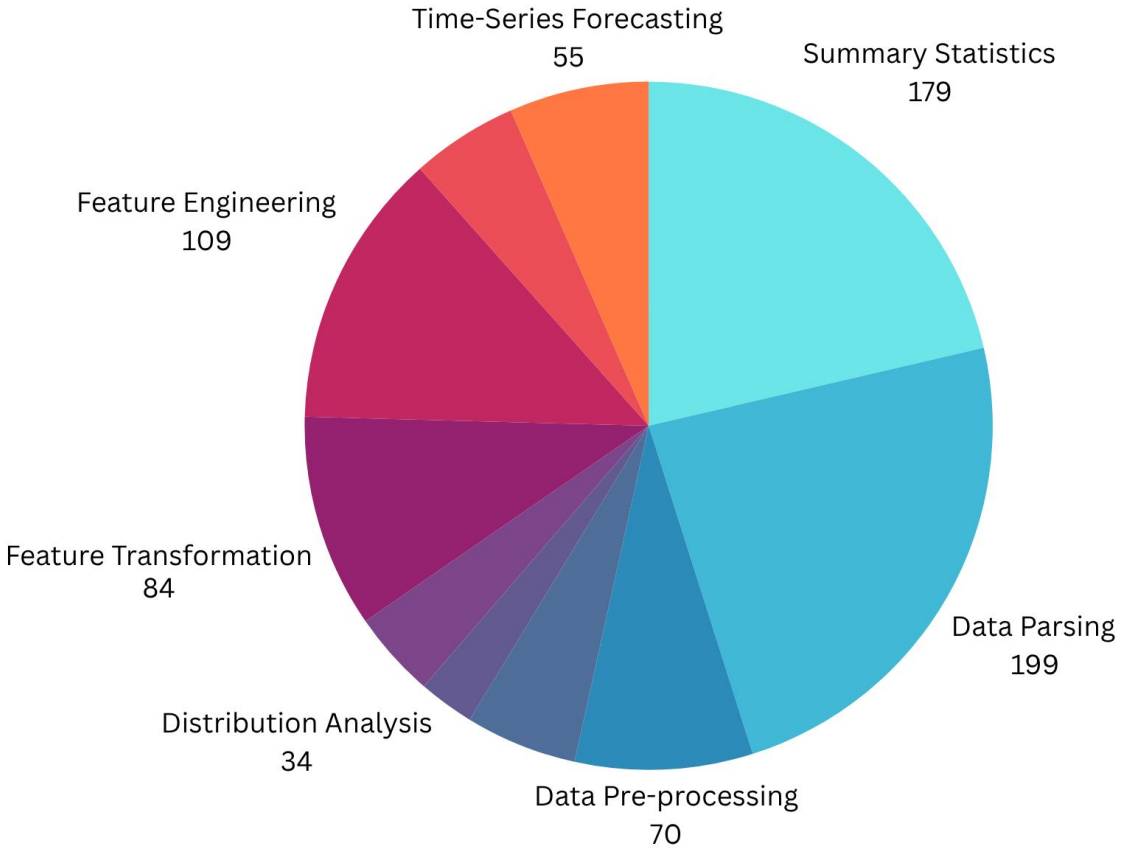
Pass@K	Out of "K" number of attempts, is atleast one of the responses correct ?	All queries are scored as 0 or 1	useful as it highlights a tool's potential to eventually succeed with repeated attempts
Pass^K	Out of "K" number of attempts, is the response correct in all attempts ?	All queries are scored as 0 or 1	useful in critical systems such as medical diagnosis tools where failure isn't an option
Pass#K	Out of "K" number of attempts, how frequently is the response correct ?	queries are scored partially based on success rate	useful in scenarios where both accuracy as well as efficiency need to be taken into account

Paper Introducing Pass@K
https://arxiv.org/abs/2107.03374

Paper Introducing Pass^K
https://arxiv.org/abs/2406.12045

Pass#K is a new metric we are introducing in an upcoming paper

Benchmarking Performance of Traversaal.ai Data Science Agent



Accuracy by AgentPro

92.1 %

Accuracy by OpenAI o3

59.1 %

Accuracy by Claude Sonnet 4

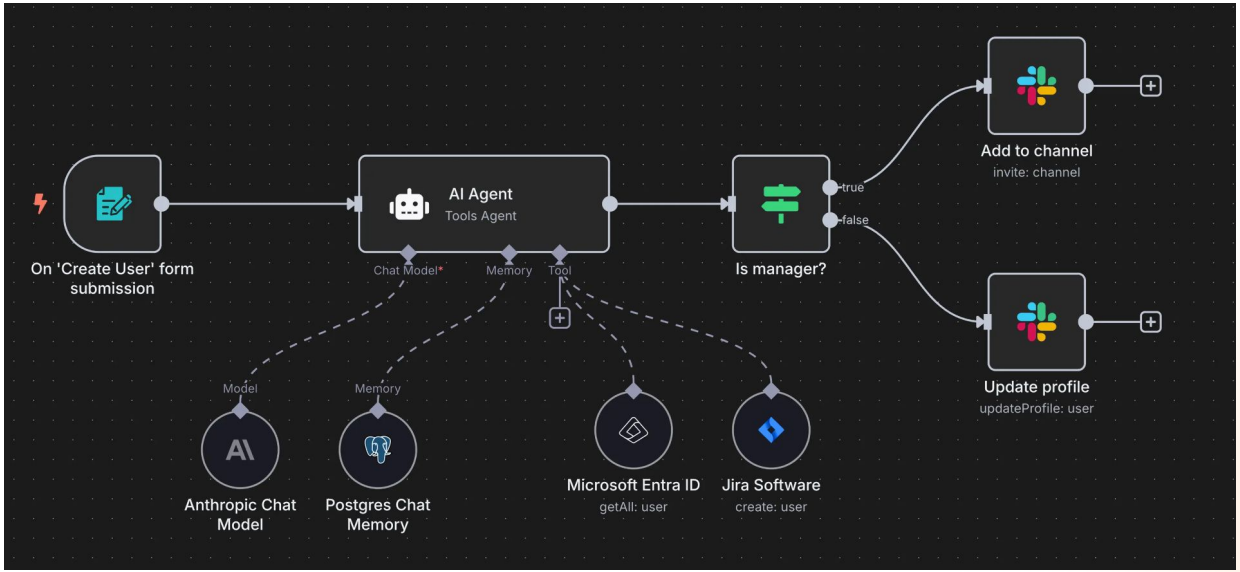
60.5 %

03: n8n and No-Code Workflows

Introducing n8n: Your Automation Companion

What is n8n?

A workflow automation tool that lets you build automated processes without writing a ton of code. Think of a digital assistant that glues your apps together, handling tasks like data syncing, alerts, and even multi-step business processes automatically.



Source

Key Concepts

1. **Workflows**
   A workflow is the complete, logical sequence of steps that defines your automation process. It's the visual blueprint for how data moves and actions are performed.

1. **Nodes**
   Nodes are the fundamental, modular workhorses that execute specific actions or integrate with external services within your n8n workflows.

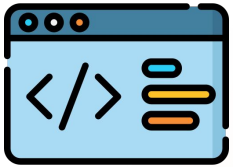
Types of Nodes

Trigger Nodes

Action Nodes

Utility Nodes

AI Nodes



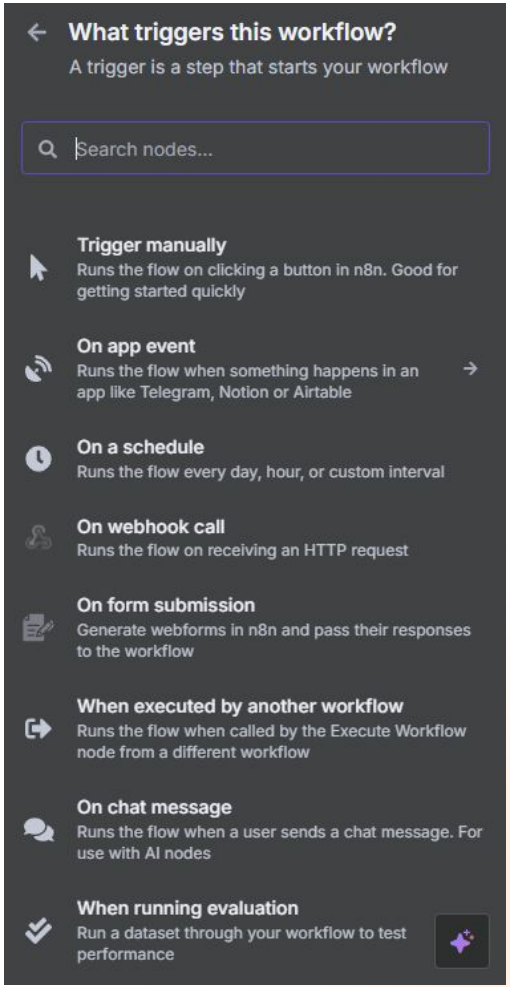
Code Nodes

Trigger Nodes

Trigger Nodes initiate workflows. Think of them as the “entry point” or the “watchers” that detect when something interesting happens.

Popular Trigger Nodes:

- Webhook: Starts the workflow when an external HTTP request is received.
- Email Trigger: Watches for incoming emails.
- RSS Feed Trigger: Monitors for updates to a feed.
- n8n Form Trigger: Gets a form input from the user

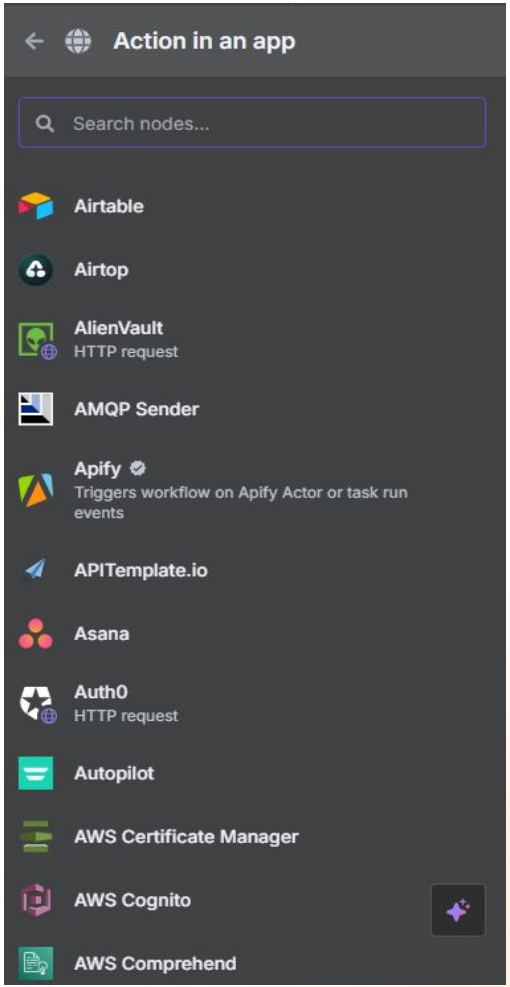


Action Nodes

Action Nodes interact with external services or applications to create records, send messages, or post data.

Examples of Action Nodes:

- HTTP Request: Sends customizable HTTP calls to any API.
- Airtable: Create/update/find records.
- Send Email: Dispatches an email via SMTP or third-party email providers.
- Discord: Posts messages to a channel or DMs.

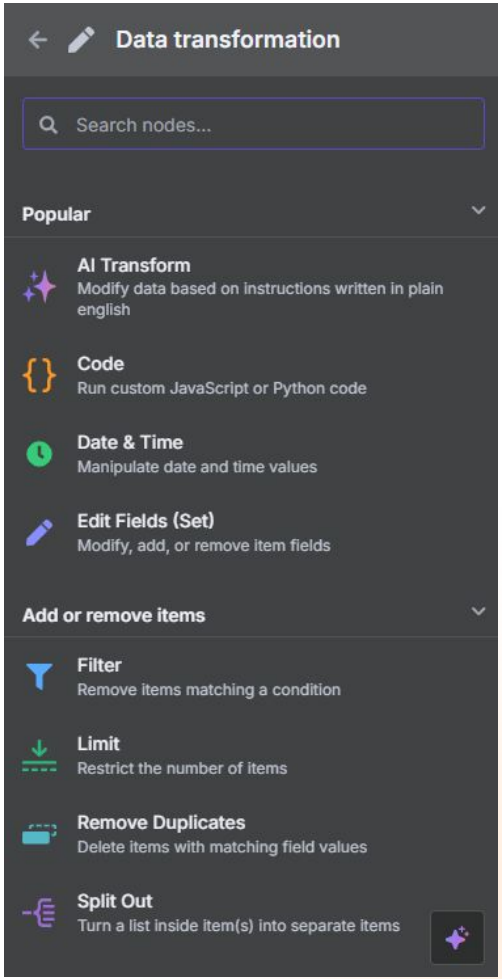


Utility Nodes

Utility Nodes don't talk to external apps. Instead, they manipulate data inside the workflow — changing structure, format, flow, or visibility. They are essential for shaping data to fit the requirements of other nodes.

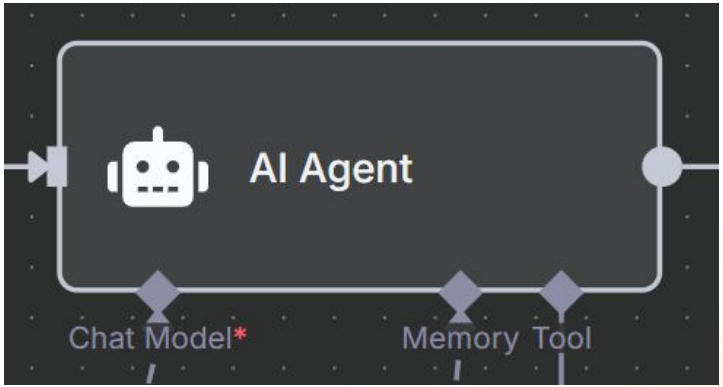
Key Utility Nodes:

- IF: Implements conditional logic (like if/else).
- Set: Creates or modifies fields in the data.
- Merge: Combines multiple inputs into one output.
- Filter: Filters data on the based of a condition



AI Nodes

The AI/LLM Node provides a straightforward way to integrate powerful LLMs into your n8n workflows. This node empowers your automation to interpret, understand, and generate text-based content from unstructured data like documents, emails, or chat inputs..

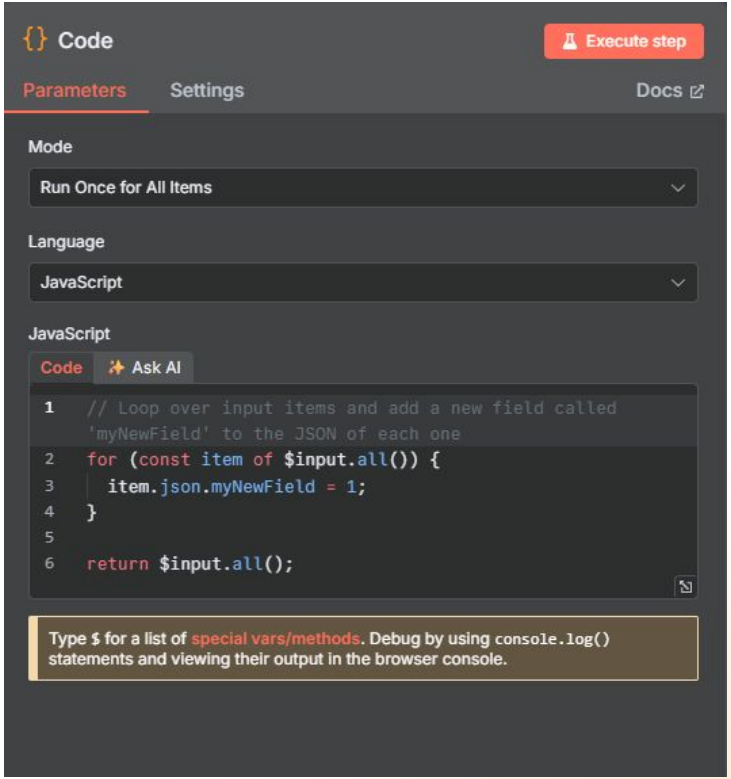


Practical Use Cases:

- Summarizing long emails or reports.
- Creating social media captions from product descriptions.
- Making decisions dynamically (e.g., choosing a reply or routing logic).

Code Node

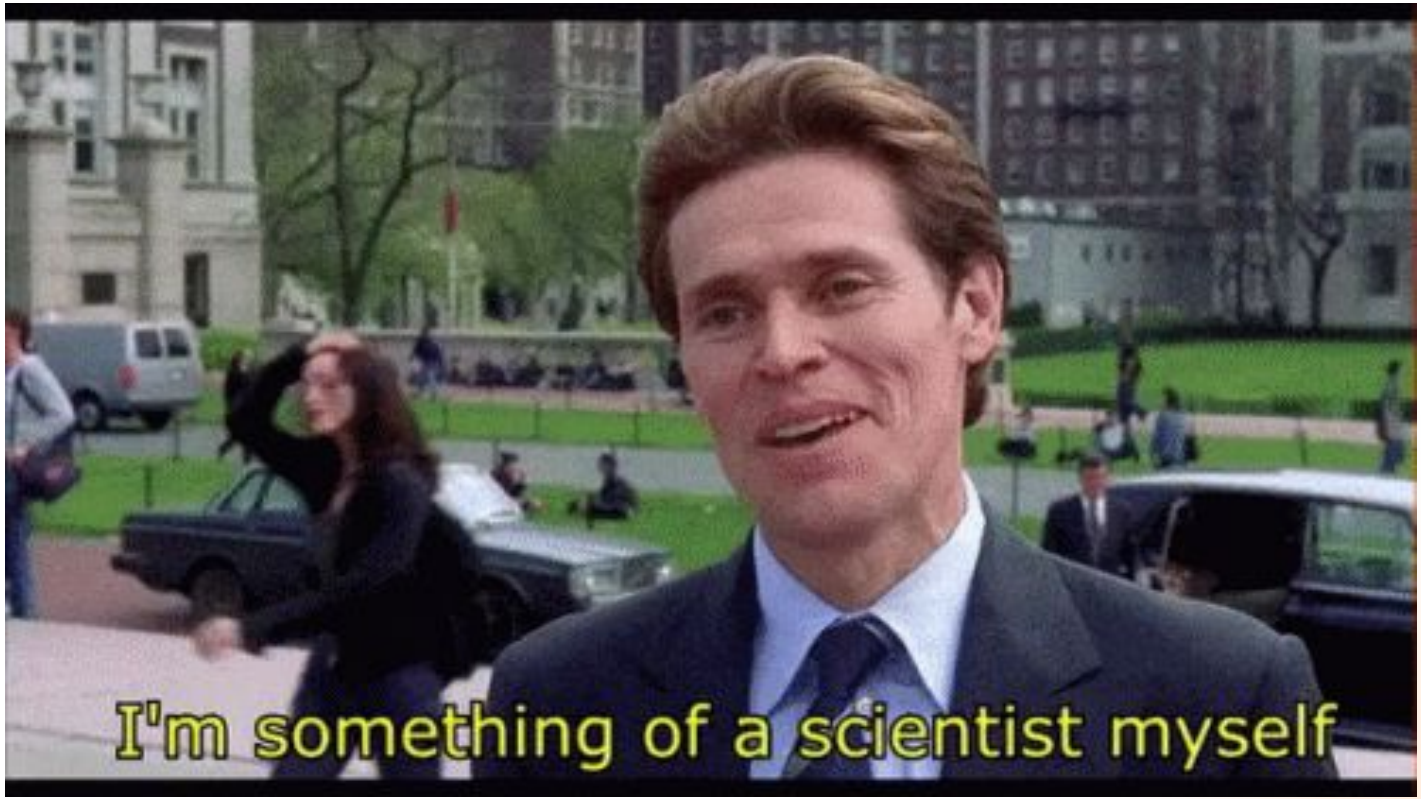
For advanced users and developers, Code Nodes allow direct JavaScript execution within the workflow. These nodes are a bridge between n8n's visual programming style and traditional scripting — providing flexibility without abandoning structure.



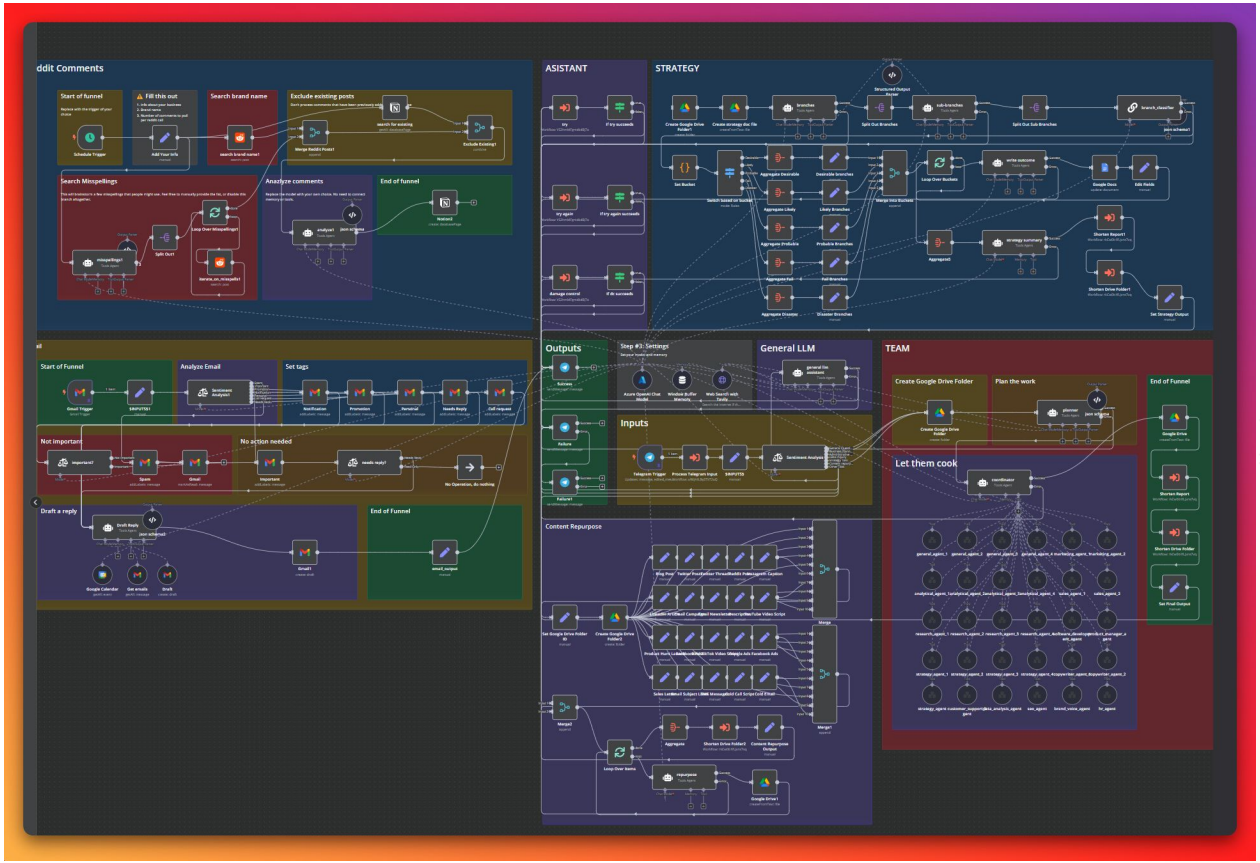
When to Use:

- Perform custom data manipulation not covered by standard nodes.
- Integrate APIs that lack dedicated n8n integrations.
- Add fallback logic, validations, or transformations.

It is safe to say now…



but please don't do this...

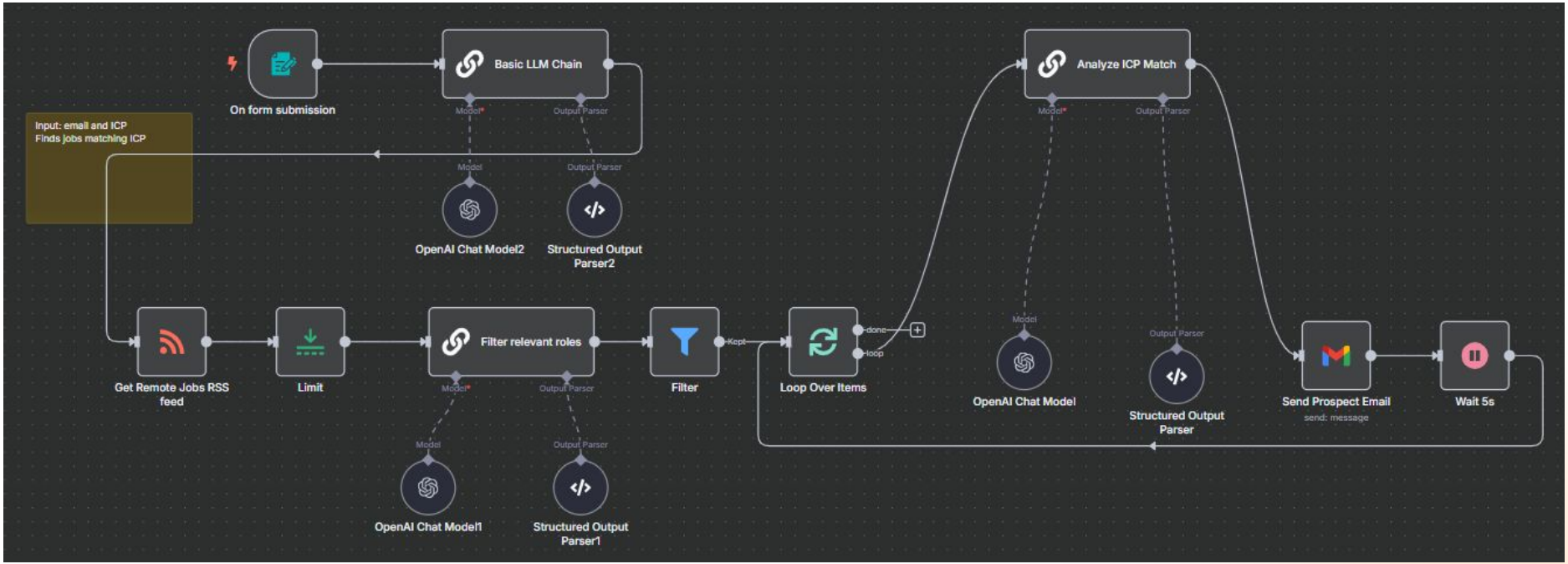


Source

04: Sales Prospecting Agent

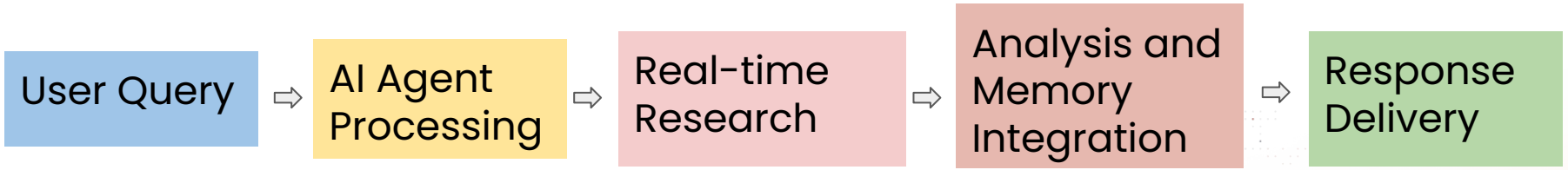
How to load json file

Architecture



Build Your first Agent

Use n8n to build an agent


User Query
AI Agent Processing
Real-time Research
Analysis and Memory Integration
Response Delivery

Thank you!

Hamza Farooq
hamza@traversaal.ai