🚀 First Day of Quarter 4 — Prompt Engineering  
  
**1. What is Prompt Engineering?**  
Prompt engineering is the process of structuring, wording, and formatting your input (prompt) to guide an AI model’s response.  
 In simple terms, it’s how you “talk” to AI so that it understands what you want and gives a precise, useful answer.  
  
**Example:**  
 ❌ Bad Prompt: “Write about marketing.”  
 ✅ Good Prompt: “Write a 150-word LinkedIn post about digital marketing trends in 2025, in a friendly and professional tone.”  
The second one is prompt engineered — it’s clear, specific, and has context.  
  
**2. How Does It Work?**  
AI models like ChatGPT are Large Language Models (LLMs) trained on huge amounts of text.  
  
 They generate responses by predicting what word (or token) comes next — based on your input and their training data.  
  
**Prompt engineering works by:**  
Providing context → so the AI knows who, what, and why.  
Setting instructions → clearly defining the goal, tone, or format.  
Constraining output → using parameters like length, style, temperature, or role.  
Iterating → refining prompts based on the model’s response until you get the desired output.  
  
**3. Common Techniques in Prompt Engineering**  
1)Role Prompting: “You are an expert software engineer. Explain APIs simply.”  
2) Few-shot Prompting: "“Example 1: … Example 2: … Now write another example.”  
3) Chain-of-Thought (CoT): “Think step by step before answering.”  
4) Context Injection: “Using the company’s mission statement below, draft a proposal…”  
5) Parameter Control: “temperature=0.2 → more factual; temperature=0.8 → more creative.”  
  
**4. Why Prompt Engineering Matters**  
Boosts accuracy and relevance of responses  
Saves time and tokens (cost)  
Helps build AI-driven apps and agents  
Makes AI work for specific goals (teaching, coding, design, business, etc.)  
  
prompt ka matlab hai agent sai baat karna, agent what to do, how to do.  
clear instruction provide karna. tools & Guardrails when to call.  
  
song composing in 🧠 Prompt Engineering.   
  
chatgpt sab nai use ki   
agent sai sau sawaal the  
ki gayein sawaal tere  
agent, hum kamaal the  
  
instruction clear daina iss keyaal main  
jaao output milga tumera her marzi ka sawaal pe  
  
tera prompt tha yaa mera context  
sawaal samjh gaya agent her instruction main.  
sab hai kubool, sab instruction hai manzoor  
gar tumhaari clear instruction hai context main.  
   
tera prompt tha yaa mera context  
sawaal samjh gaya agent her instruction main.  
  
agent se sawaal kar bolae, tasveer mein hamaari dekh kar bolae,  
instruction main clear context kyun? user batao naa  
ye sab pata tha tumko, instruction main context bhool jaaoge daina humko  
instruction ko achay sai craft nai kia kyun? user batao naa  
  
phenka context se bhi humko yun nikaal ke  
jaao ab agent dai ga answer baar pyaar sai.  
tera prompt tha yaa mera context  
sawaal samjh gaya agent her instruction main.

What is context engineering?  
  
Context Engineering is the process of designing, managing, and optimizing the context provided to a large language model (LLM) — such as ChatGPT — to improve the accuracy, relevance, and quality of its responses.  
  
 **1. What “context” means.**  
When you prompt an AI model, you don’t just give it one question — you give it context:  
Background information, Instructions or goals, Examples Role or tone (e.g., “You are a teacher”) This context window helps the AI understand what you want and how you want it.  
  
 **2. Why Context Matters.**  
Language models don’t “understand” in a human sense — they predict text based on context. The context includes:  
The prompt (your direct input)  
Any conversation history, External data or examples provided in the input System instructions or constraints By shaping this context effectively, we control how the model thinks and responds.  
  
 **3. What context engineering does.**  
Context engineering is about carefully building and managing that context so the model behaves in a predictable, high-quality way.  
  
Role prompting: “You are an expert data scientist.”  
Few-shot prompting: Show 2–3 examples before asking for a new result.  
Context layering: Feed in user’s goal → data → example → question.  
Dynamic context injection: Summarize related notes or user history.  
Compression: “Summarize the last 5 messages in 100 tokens.”  
  
4. **Core Principles of Context Engineering.**1. Context Framing: Set the role or purpose of the model.  
 Example: “You are a financial advisor helping small businesses with budgeting.”  
2. Information Injection: Feed relevant background data into the prompt so the model has everything it needs.  
 Example: “Here’s a customer’s purchase history: [data]. Recommend new products.”  
3. Instruction Precision  
Use clear, structured directions to control output tone, format, or reasoning.  
 Example: “Summarize this article in three bullet points, using plain English.”  
4. Memory and Continuity  
Maintain consistent context across messages to build coherent multi-turn conversations.  
**5.Constraint Design**  
Apply rules to limit or guide responses (e.g., word limits, tone, style).  
6. Dynamic Context Updating  
Modify or add context mid-conversation as new information appears.  
  
**context: to provide background or relevant information to LLM.**

**I am study in Quarter 4 in GIAIC on 10 October 2025.**

**Context hai Quarter 4, 10 October**

**Provide the data of the 10 top class student in GIAIC**

**Context hai Top 10, student**

**Programmer no need to use AI effectively.**

**Means if doctor is not effectively from AI its doesn’t need.**

**Tool calling for CV**

**Tool calling for gmail**

**F = Open (“demo.text”) “r” for read**

**Print (F.read())**

Song: Don't Worry Be Happy 1989   
  
Agent ko Hai Problem  
instruction Mein Hai Problem  
Aage Piche Aaju Baju  
Problem Hi Problem  
Dont Worry, Be Happy 😂 👍 😁 😆   
  
Prompt chhota kahe ko,  
Context toda kahe ko,  
Role aur data clear karo,  
AI ko samjha do!  
Dont Worry, Be Happy  
  
Context Engineering karo,  
Prompt ko thoda socho,  
Output perfect aayega,  
Model khush ho jaayega!   
Dont Worry, Be Happy  
  
Instruction likho seedha,  
Format bhi dikhla do,  
Jo chaho woh likhwa lo,  
LLM ko samjha do!   
Dont Worry, Be Happy

What is Nano Banana?  
  
Google Nano Banana (like MidJourney, DALL·E, or Stable Diffusion) responds to highly descriptive prompts. The more structured, visual, and contextual your input, the closer the output matches your intent.  
  
Nano means Something very small, minimal, or fine-grained.  
Banana means A chunk of data (text, token, or piece of context) that an LLM “eats” to understand meaning.  
  
🧠 In Prompt Engineering Terms:  
Nano Banana = Token or micro-context unit.  
It’s the smallest semantic chunk the model reads at a time.  
Each “banana” (token) is part of the model’s context window — the memory area the model uses to generate the next word.  
  
🧩 Example:  
This sentence is broken down into Nano Bananas (tokens):  
 → [Write] [a] [poem] [about] [the] [moon] [.]  
The model digests these tiny bananas to understand your intent.  
  
💡 Why it Matters:  
Understanding Nano Bananas helps you:  
Optimize prompts (shorter, clearer = fewer tokens = faster response).  
Manage token limits (context length).  
Write efficient prompts that don’t “overfeed” the model.  
  
Prompt 1: Bold and Heroic Portrait in Dramatic Light  
Prompt 2: Casual Outdoor Lifestyle Portrait  
Prompt 3: Elegant Formal Portrait with Suit  
Prompt 4: Artistic Black and White Portrait  
Prompt 5: Adventure and Travel Portrait  
Prompt 6: Modern Street Style Portrait  
Prompt 7: Cinematic Close-Up with Moody Lighting  
Prompt 8: Festive Indian Attire Portrait  
Prompt 9: Fitness and Gym Lifestyle Portrait  
Prompt 10: Professional Studio Headshot with Subtle Smile

Core Principles  
1. Specificity Over Generality  
2. Visual Hierarchy  
3. Professional Photography Language  
  
nano banana link  
[**https://nanobanana.ai/**](https://nanobanana.ai/)  
  
Gemini AI Photo Editing Prompts   
[**https://lnkd.in/d2jfu6pc**](https://lnkd.in/d2jfu6pc)  
  
GPT-5 nano  
Rate limits ensure fair and reliable access to the API by placing specific caps on requests or tokens used within a given time period. Your usage tier determines how high these limits are set and automatically increases as you send more requests and spend more on the API.  
  
Modalities Text - Image  
Endpoints Chat Completions, Responses, Batch  
Features Streaming, Function calling, Structured outputs  
Tools File search, Image generation, Code interpreter, MCP.

What sophisticated autocomplete system?  
  
LLMs don't "understand" in the human sense—they're sophisticated autocomplete systems. Your prompt sets up the context for what should come next.  
  
A sophisticated autocomplete system is an advanced AI-powered text prediction system — like the one used in ChatGPT, Google Smart Compose, or code editors like GitHub Copilot — that goes far beyond simple word completion.  
  
At its core, an autocomplete system predicts the next word (or words) you’re likely to type, based on what you’ve already written.  
  
**Simple vs. Sophisticated**  
Simple Autocomplete  
Suggests based on dictionary or frequency (no context).  
You type “hel” → suggests “hello”  
  
Sophisticated Autocomplete  
Uses machine learning (LLMs) to understand context, tone, intent, and grammar — not just spelling.  
You type “Can you write an email to” → suggests “the HR department about my leave?”  
  
How a Sophisticated Autocomplete System Works  
Context Understanding:  
 It reads not only your last few words but the entire conversation or paragraph to guess what fits best.  
(Example: It knows “apple” in “I want to buy an apple” vs. “Apple released a new iPhone.”)  
  
LLM Prediction (Language Model):  
 It uses transformer models like GPT or BERT to predict the next token (word or part of word) with the highest probability.  
  
Temperature, Top-K, and Top-P:  
 These parameters control how creative or focused the autocomplete is.  
Low temperature: precise and predictable.  
High temperature: more creative and varied.  
  
Continuous Feedback:  
 It learns from user corrections and preferences to improve over time.  
  
Real-World Examples  
ChatGPT: Finishes your thoughts or expands ideas in conversation.  
  
Gmail Smart Compose: Suggests full sentences while you type emails.  
  
GitHub Copilot / VS Code: Suggests entire code functions based on a few words or comments.  
  
Search Engines: Predict your full query intelligently based on trends and your history.

What is “Temperature” in Prompt Engineering?

In AI models like ChatGPT, **“temperature”** is a parameter that controls **how random or creative** the model’s output is.

It affects **the balance between accuracy and creativity**.

**🔸 Low Temperature (e.g., 0 to 0.3)** Temperature = **0.2** (Focused & Factual)

Prompt: Write a short paragraph about Quaid-e-Azam Muhammad Ali Jinnah

Quaid-e-Azam Muhammad Ali Jinnah was the founder of Pakistan. He led the All-India Muslim League and played a key role in the creation of Pakistan in 1947. He is respected for his leadership, vision, and commitment to the rights of Muslims in the Indian subcontinent.

**Style:** Clear, historical, and factual.  
**Use case:** School report, exam answer, or factual article.

🔸 Medium Temperature (e.g., 0.5 to 0.7) **0.6** (Balanced & Expressive)

Quaid-e-Azam Muhammad Ali Jinnah was not only a political leader but also a man of unshakable will. His determination and vision turned the dream of Pakistan into reality in 1947. To millions, he symbolized hope, dignity, and unity.

**Style:** Slightly emotional, but still factual.  
**Use case:** Magazine article, motivational speech.

🔸 High Temperature (e.g., 0.8 to 1.0+) Temperature = **1.0** (Creative & Imaginative)

Quaid-e-Azam was a storm in silence — a man whose calm words moved mountains. With his pen and purpose, he carved a nation from dreams and determination. His legacy still whispers in the green of our flag and the hearts of those who believe in justice and faith.

**Style:** Poetic and emotional, less factual.  
**Use case:** Speech, song, poetry, or storytelling.

| **Temperature** | **Style** | **Description** | **Best Use** |
| --- | --- | --- | --- |
| 0.2 | Serious | Pure facts, no emotion | Exams, reports |
| 0.6 | Balanced | Factual + slightly emotional | Essays, speeches |
| 1.0 | Creative | Imaginative, poetic | Poetry, songs, inspiration |

1. **Top-K (K = “number of choices”)**

**Top-K** controls **how many possible next words** the model considers **before choosing one**.  
It limits the *pool of randomness*.

* **Low K (e.g., 10):** Model considers only the top 10 most likely next words → more **focused**.
* **High K (e.g., 100 or 200):** Model can consider more possibilities → more **creative or random**.

**Prompt:**

Write a short line about **Quaid-e-Azam**.

**Top-K = 10 (Focused):**

Quaid-e-Azam was a great leader who founded Pakistan.

**Top-K = 100 (Creative):**

Quaid-e-Azam was the architect of a dream that became a green reality.

*Higher K allows rare or creative word choices.*

1. ***Top-P (a.k.a. Nucleus Sampling or Probability Threshold)***

**Top-P** picks words based on **probability mass**.  
It keeps adding likely words until the cumulative probability reaches *P* (e.g., 0.9).

* **Low P (0.3):** Only most predictable words are used → factual, plain.
* **High P (0.9):** Model can use diverse and imaginative words.

**Prompt:**

* Describe Quaid-e-Azam’s leadership.

**Top-P = 0.3 (Focused):**

* Quaid-e-Azam’s leadership was disciplined, honest, and determined.

**Top-P = 0.9 (Creative):**

* Quaid-e-Azam’s leadership was like a storm guided by calm — firm, fearless, and filled with destiny.

Higher P allows more expressive, poetic responses.

1. **Output Length**

This defines **how long** the model’s answer should be — in **tokens** (each token ≈ 1 word or part of a word).

* **Short length (e.g., 50 tokens):** Concise answers.
* **Long length (e.g., 300 tokens):** Detailed paragraphs.

**Prompt:**

Write about Quaid-e-Azam’s vision for Pakistan.

**Output Length = 50 tokens (Short):**

Jinnah dreamed of a nation built on faith, unity, and discipline — where justice and equality guide every citizen.

**Output Length = 300 tokens (Long):**

Quaid-e-Azam envisioned Pakistan as a land of peace, tolerance, and opportunity... *(continues with more detail and examples).*

*You control how deep or brief the response is.*

**Prompt:**

Write about Quaid-e-Azam’s vision for Pakistan.

**Output Length = 50 tokens (Short):**

Jinnah dreamed of a nation built on faith, unity, and discipline — where justice and equality guide every citizen.

**Output Length = 300 tokens (Long):**

Quaid-e-Azam envisioned Pakistan as a land of peace, tolerance, and opportunity... *(continues with more detail and examples).*

*You control how deep or brief the response is.*

1. **Token Limit**

Every AI model has a **maximum token capacity** (prompt + response combined).  
If your input is too long, it reduces how much output the model can generate.

For example:

* GPT-4 Turbo → ~128,000 tokens
* GPT-5 (you’re using) → even higher limits

💡 If your prompt = 5,000 tokens, and model limit = 8,000 → only 3,000 tokens remain for output.

**long Quaid-e-Azam biography** (say 6,000 tokens), the model can only reply briefly — maybe 2,000 tokens — before hitting the limit.

*So, keep your prompt concise if you want longer answers.*

| **Parameter** | **Controls** | **Low Value** | **High Value** | **Example Use** |
| --- | --- | --- | --- | --- |
| **Top-K** | How many word options | Focused | Creative | Factual vs poetic |
| **Top-P** | Probability diversity | Predictable | Imaginative | Essay vs poem |
| **Output Length** | Response size | Short summary | Long explanation | Tweet vs article |
| **Token Limit** | Model’s total space | Input-heavy (less output) | Balanced input/output | Long documents |