Fundamentals of Artificial Intelligence: Ready to Move Beyond AI Hype? Build Real Skills That Matter!

What it is, what it can do, and how to start using it.

May 2026

Lida Gharibvand
Professor, Loma Linda University
lgharibvand@gmail.com

Today's Agenda — Four Sections

01

Foundations

What AI, ML, deep learning, and LLMs actually are

02

Tools

ChatGPT, Claude, Gemini, Copilot, Perplexity, and more

03

Applications & Prompting

Prompts, workflows, and real day-to-day use cases

04

Advanced Concepts & Responsible AI

RAG, agents, reasoning models, risks, governance, what's next

THE BIG IDEA

ChatGPT is one room. AI is the whole building.

AI that SEES

Vision systems read images and video — faster than people, all day, every day.

AI that DECIDES

Recommenders, anomaly detectors, and reinforcement learners pick what happens next.

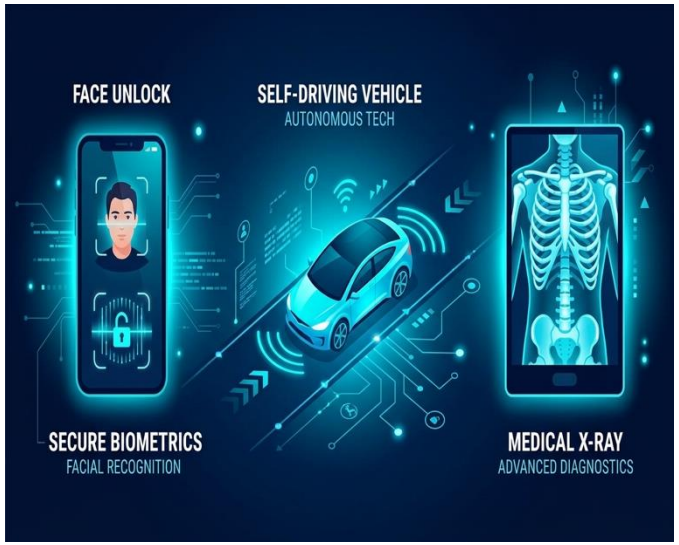
AI that ACTS

Robots, agents, and control systems turn predictions into real-world motion.

The most powerful AI doesn't talk. It watches, ranks, flags, and acts.

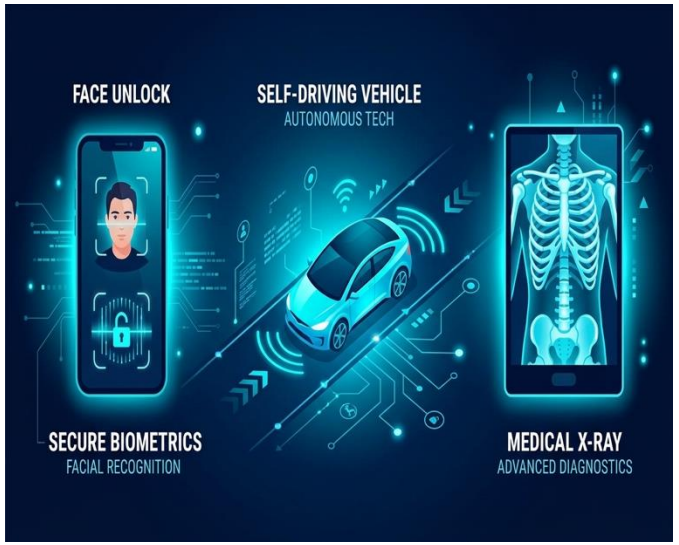
Section opener — "The Big Idea": framing what AI is and why it matters now.

Where computer vision is already running.



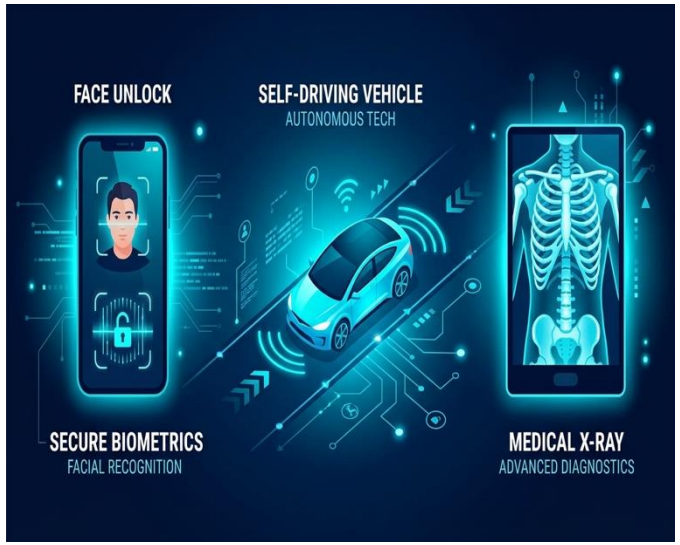
Face Unlock

Maps your face into a vector and matches it in milliseconds — every time you pick up your phone.



Self-Driving

Real-time detection of lanes, cars, pedestrians, and signs from many cameras at once.



Medical Imaging

Radiology models flag tumors, fractures, and anomalies in CT, MRI, and X-ray scans.

VISION IN THE WILD — Where computer vision is already running.

None of these need a chat interface. They just see and decide.

AI that knows what you want next.

01

Learns from your behavior

Clicks, watches, skips, and dwell time are training data.

02

Predicts your taste

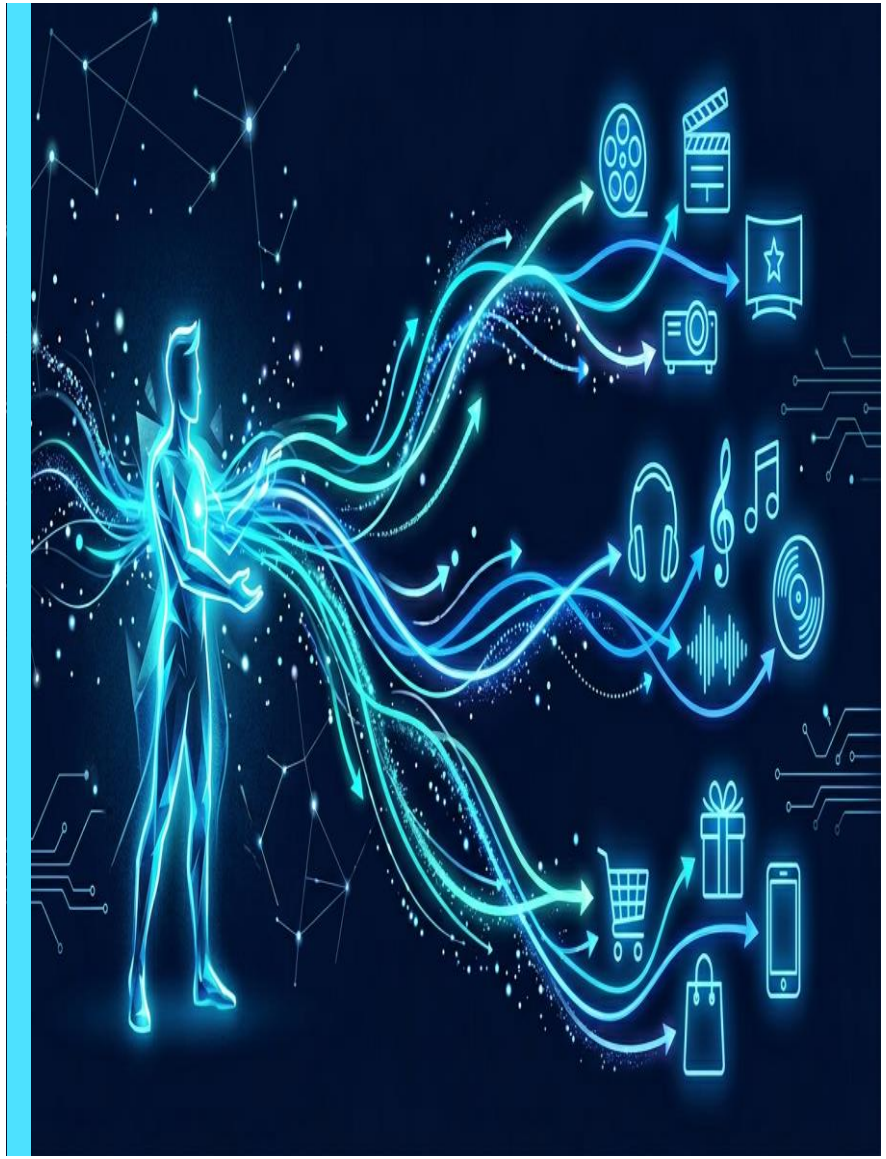
Embeds users and items in the same space; ranks candidates by score.

03

Drives most of what you consume

Up to ~80% of Netflix watches and ~35% of Amazon revenue come from recommendations.

The feed is an AI. The shelf is an AI. The home page is an AI.



AI that knows what you want next. — Learns from your behavior

Three you already trust with your time.

Netflix

Personalizes rows, artwork, and even the order you see episodes in.

Amazon

“Customers also bought” + ranked search results = revenue engine.

Spotify

Daily mixes and Discover Weekly are pure recommendation models.

RECOMMENDERS IN THE WILD — Three you already trust with your time.

Your taste is partly an output of these models — not just an input.

ROBOTICS

AI that ACTS in the physical world.

01

Sense → Plan → Act

Combines vision, prediction, and control into one closed loop.

02

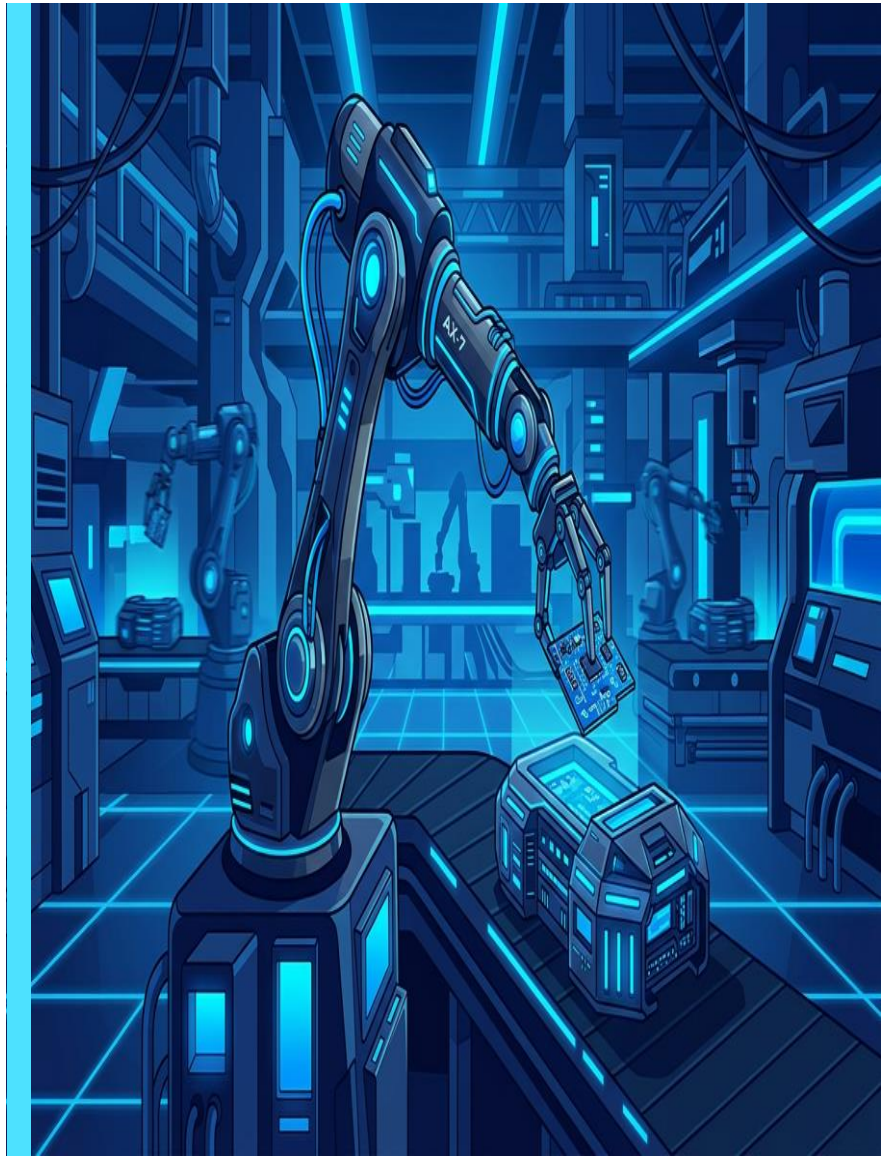
Industrial today, general-purpose tomorrow

Warehouse pickers, surgical assistants, autonomous drones.

03

Hard parts: dexterity & safety

Reasoning is easier than reliably grabbing a coffee cup.



The body is the bottleneck — not the brain.

HEALTHCARE

AI in healthcare — beyond the chatbot.

01

Disease prediction

Risk scoring from EHRs, labs, and wearables — long before symptoms.

02

Medical imaging

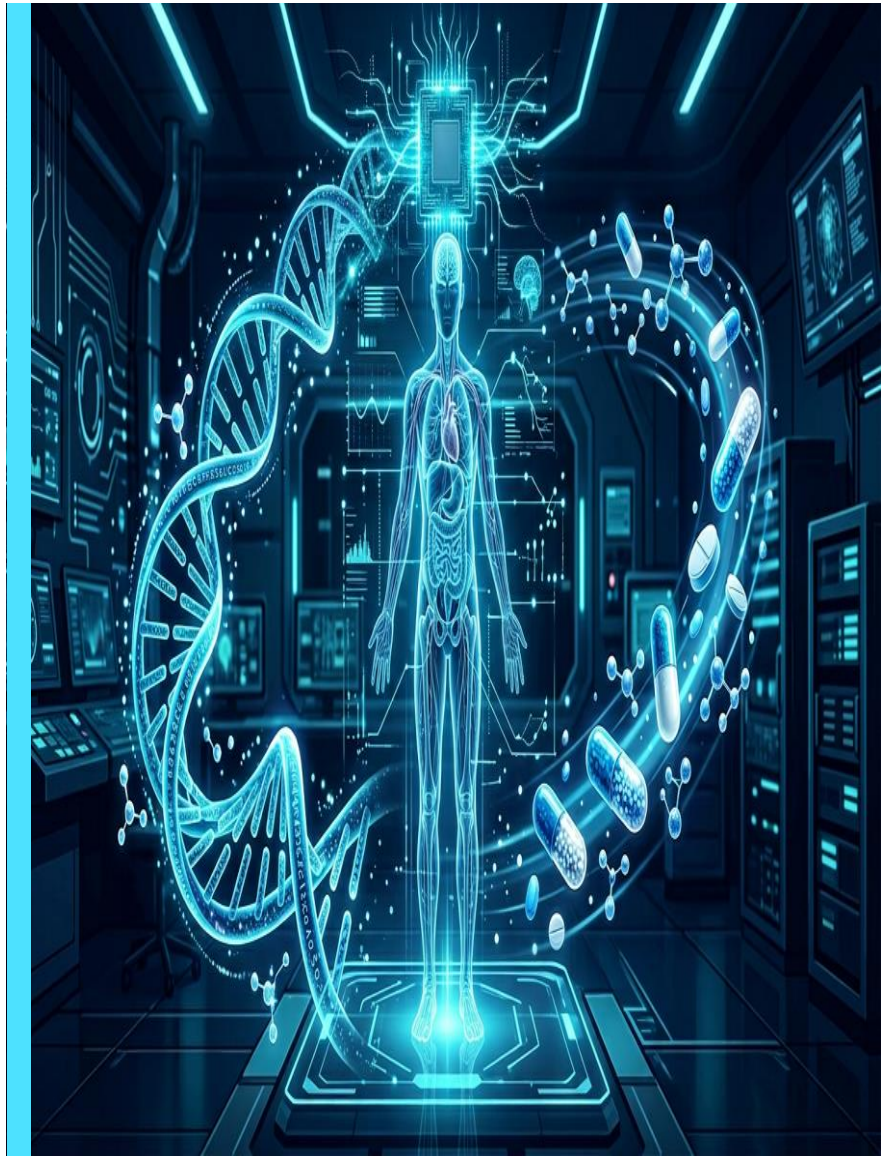
Detection of tumors, retinopathy, fractures with radiologist-level accuracy on benchmarks.

03

Drug discovery

Protein structure (AlphaFold), molecule generation, and clinical trial matching.

Most healthcare AI is a quiet specialist model — not a friendly assistant.



REINFORCEMENT LEARNING

AI that learns by trial and reward.

01

Reward, not labels

Agent acts → environment responds → reward signal updates the policy.

02

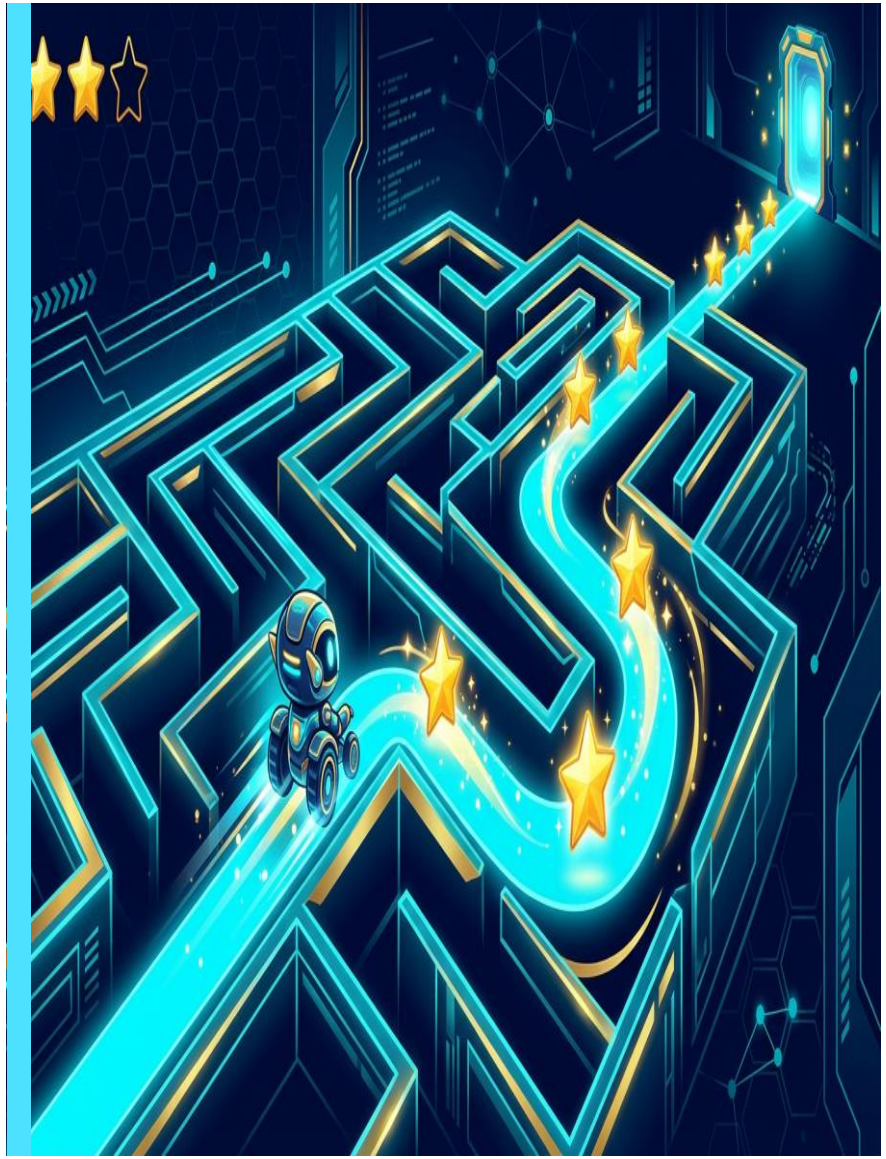
Superhuman in narrow worlds

AlphaGo, Dota OpenAI Five, StarCraft AlphaStar, chip floorplanning.

03

Powering modern LLMs too

RLHF & RLAIF — yes, RL is also how ChatGPT got polite.



No teacher, just a scoreboard. That's often enough.

SPEECH

AI that listens — and speaks back.

01

Speech-to-Text (ASR)

Whisper-class models transcribe across dozens of languages, even noisy audio.

02

Text-to-Speech (TTS)

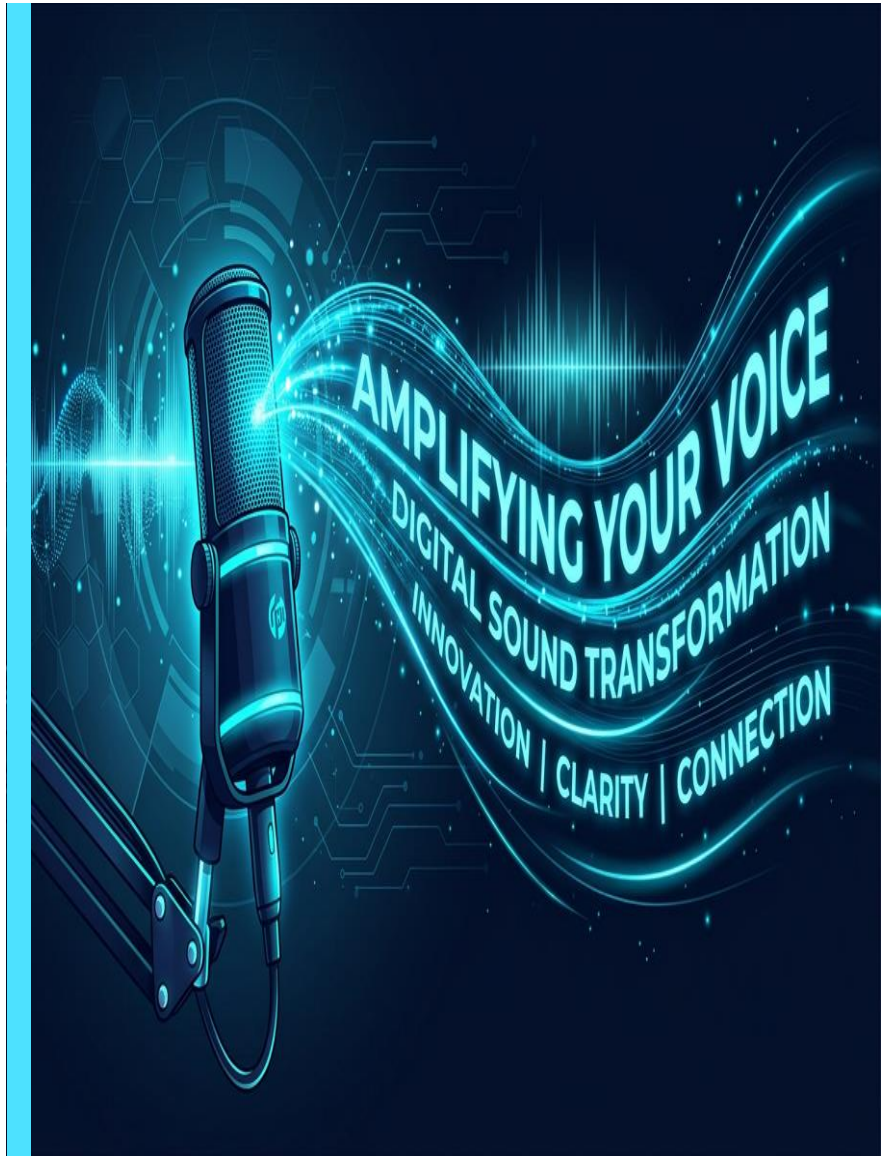
Neural voices that sound human — and can be cloned from minutes of audio.

03

Voice agents

Call centers, dictation, accessibility, real-time translation.

When you 'talk to AI,' a stack of speech models runs before the LLM ever sees a word.



GRAPHS

AI that understands relationships.

01

Networks, not tables

Operates on nodes & edges — people, accounts, molecules, roads.

02

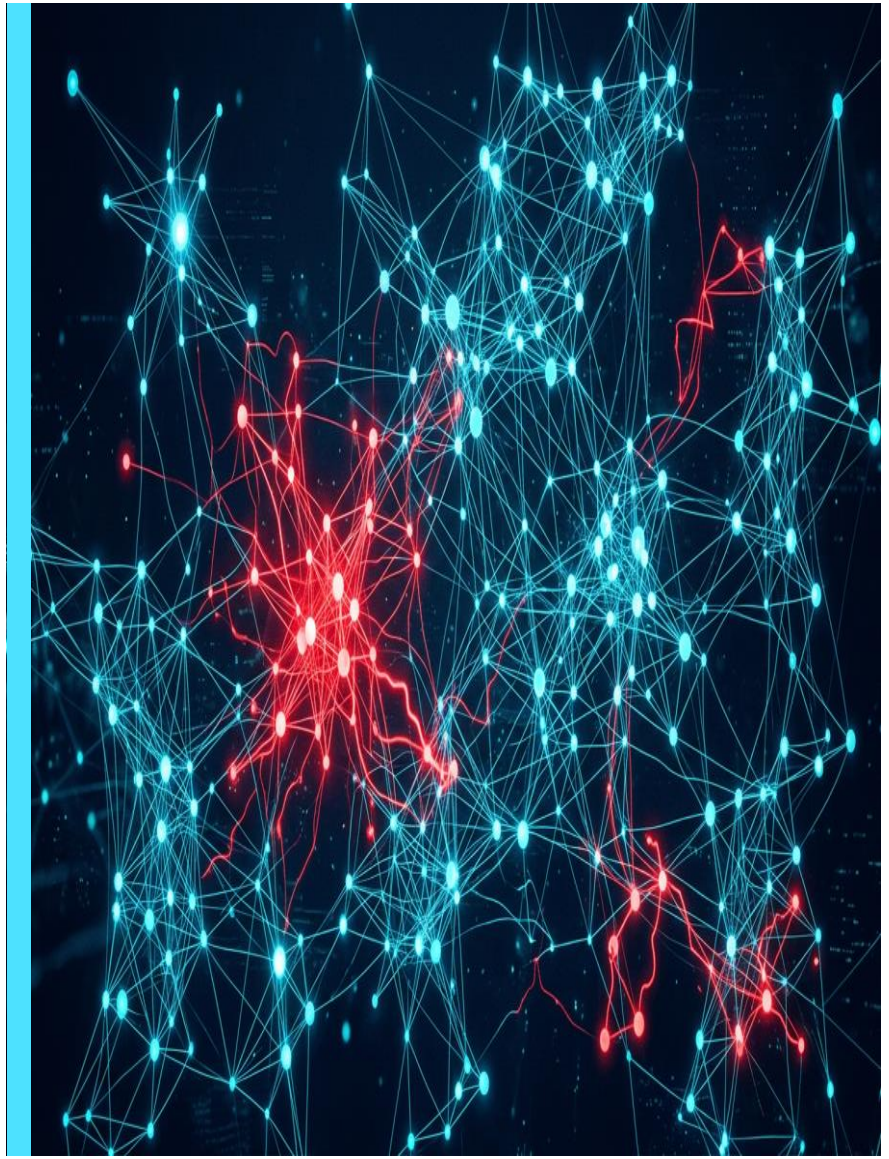
Catches the invisible

Fraud rings, money laundering, drug-target interactions.

03

Quiet workhorse

Recommends friends, routes packages, predicts protein interactions.



Some patterns only show up as a shape, not a row.

AI is a tool — not magic.

It doesn't think. It predicts. How you guide it is the work.

Section opener — "Big Idea": framing the conceptual core.

AI vs. ML vs. DL — same family, different jobs

Artificial Intelligence (AI)

Any system that does something we'd call "smart" if a person did it.

Examples: Spam filters, GPS rerouting, voice assistants.

Machine Learning (ML)

A way to build AI by showing examples instead of writing rules.

Examples: Netflix recommendations, fraud detection.

Deep Learning (DL)

ML using neural networks with many layers — great for images, audio, text.

Examples: Face unlock, voice-to-text, ChatGPT.

AI paradigms — how models learn

Four main flavors of machine learning. The flavor decides what data you need.

Supervised learning

Train on labeled examples (input, correct answer). Goal: predict the label on new inputs.

Examples: spam filter, image classification, credit-risk scoring, churn prediction.

Needs: lots of clean labels. The label budget is the bottleneck.

Unsupervised learning

Find structure in unlabeled data. No 'correct answer'.

Examples: customer segmentation, anomaly detection, topic modeling, embeddings for search.

Needs: a lot of raw data. Outputs are exploratory — you must interpret them.

Reinforcement learning

An agent takes actions in an environment and gets rewards. Learn the policy that maximizes reward.

Examples: AlphaGo, robotics, ad bidding, RLHF for chat models (the 'HF' part).

Needs: a simulator or live environment, a clean reward signal.

Self-supervised learning

Make labels from the data itself: predict the next word, the missing patch, the next frame.

Examples: GPT-style pretraining, BERT, CLIP, modern audio and vision foundation models.

Needs: huge amounts of raw text/images/audio. This is how today's foundation models are trained.

AI paradigms — how models learn — Four main flavors of machine learning. The flavor decides what data you need.

How does a model "learn"?

Analogy: a kid learning fruit

- Show 100 photos labeled "apple" and "banana".
- The kid notices: round + red = apple, long + yellow = banana.
- On a new photo, they guess based on what they noticed.
- Wrong guesses → they update their mental rule.
- That's it. ML does this with numbers instead of intuition.

Three steps every model goes through

1

Train

Feed labeled examples; the model adjusts internal numbers ("weights") until its guesses match.

2

Test

Show new examples it has never seen and measure accuracy.

3

Deploy

Use it on real data. Keep watching — performance can drift.

How does a model "learn"? — Analogy: a kid learning fruit

Major deep-learning architectures — a cheat sheet

Six families you will hear named. Each one is a different shape of neural network for a different shape of problem.

MLP — Multilayer Perceptron

Plain stacked layers of neurons. The original deep network. Good for tabular data when you don't need spatial or sequence structure. Foundation for everything else.

CNN — Convolutional Neural Net

Slides small filters across grids. Built for images: edges → shapes → objects. Powers face unlock, medical imaging, defect detection. Examples: ResNet, EfficientNet.

RNN / LSTM — Recurrent

Reads inputs in order and carries a memory. Built for sequences: speech, time series, older translation. Largely replaced by transformers, still used in tiny on-device models.

Transformer

Attention over all tokens at once. Powers every modern LLM (GPT, Claude, Gemini, Llama). Strength: long-range context. Cost: quadratic in sequence length.

Diffusion model

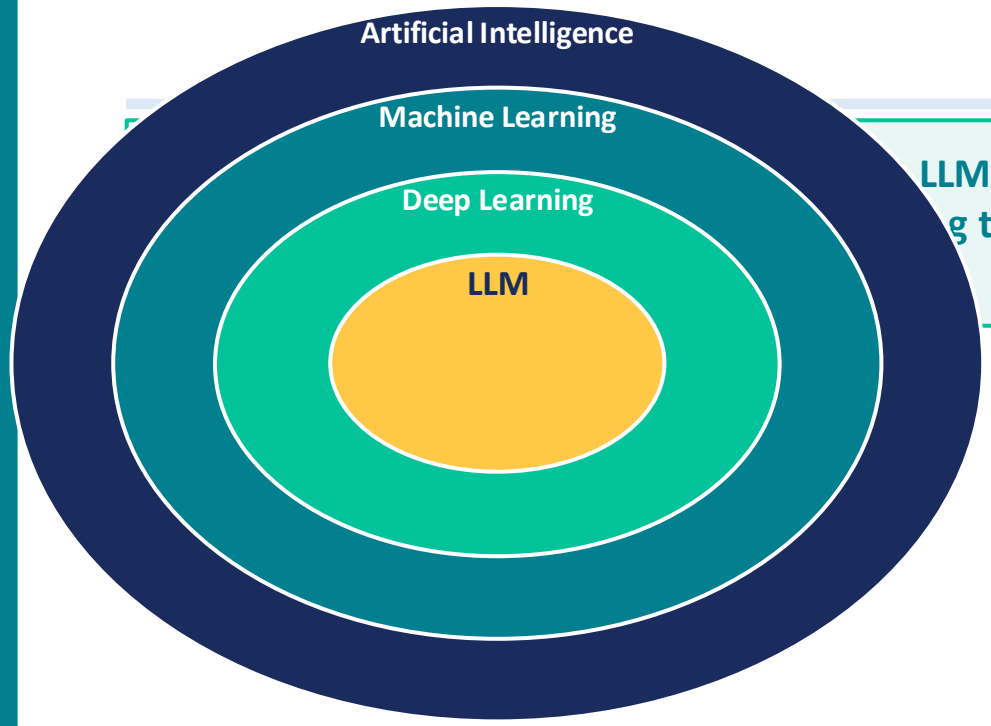
Learns to reverse noise step-by-step into a clean output. Powers most image and video generators (Stable Diffusion, Imagen, Sora-style models).

MoE — Mixture of Experts

Many small expert sub-networks; router picks a few per token. Cheaper inference for huge models. Used by Mixtral, GPT-5 class systems, DeepSeek.

Major deep-learning architectures — a cheat sheet — Six families you will hear named. Each one is a different shape of neural network for a different shape of problem.

What is a Large Language Model (LLM)?



LLM is a deep-learning model trained on massive text corpora (trillions of tokens)
g the next token. Generative AI extends this idea to images, audio, video, and

Popular Models (May 2026)

GPT-5.5 / o3

OpenAI – multimodal, agent mode, reasoning

Claude 4.5 Opus

Anthropic – long docs, computer use, coding

Gemini 3 Pro

Google – 1M context, video, Deep Research

Llama 5

Meta – open weights, self-hostable

Copilot

Microsoft 365 (GPT-5.5 backend)

Mistral

Mistral Large 3 — EU residency, open weights

What is a Large Language Model (LLM)? — AI ⊃ ML ⊃ DL ⊃ Generative AI ⊃ LLM. An LLM is a deep-learning model trained on massive text corpora (trillions of tokens) that generates human-like text by pre…

Core idea: it predicts the next word

An advanced prediction engine, not a thinking mind.

- Trained on a massive amount of text from the internet, books, and code.
- Given a prompt, it predicts the most likely next word — over and over.
- No reasoning, beliefs, or memory beyond the conversation in front of it.
- Implication: the quality of your input shapes the quality of the output.

Core idea: it predicts the next word — An advanced prediction engine, not a thinking mind.

How LLMs Work — Token Prediction at Scale

Key Concepts

Tokenization

Text split into tokens (~4 chars each). GPT-5 context: 400K tokens ≈ 100K words.

Attention Mechanism

Model learns which tokens in context matter most — the 'Transformer' breakthrough.

Temperature

Controls randomness. Low (0.1) = focused & deterministic. High (0.9) = creative & varied.

Hallucination

LLMs can generate plausible-sounding but incorrect info — always verify critical outputs.

How LLMs Work — Token Prediction at Scale — "The color\nof the sky is"

Why is AI everywhere right now?

Three things lined up at once.

Data

The internet gave models trillions of words and billions of images to learn from.

Compute

GPUs got ~1000× cheaper per useful operation since 2012.

Methods

The Transformer (2017) lets models scale efficiently — bigger really did mean better.

Recent milestones

2017	2022	2023	2024	2025–26
Transformer paper	ChatGPT launches	GPT-4, image gen mainstream	Multimodal + agents emerge	GPT-5.5, Claude 4.5 Opus, Gemini 3, agents mainstream

Why is AI everywhere right now? — Three things lined up at once.

What today's AI can and can't do

Can do well

- Summarize and rewrite text
- Translate between languages
- Generate first drafts (emails, code, designs)
- Answer questions over documents you provide
- Extract structured data from messy text
- Hold a fluent conversation
- Describe what's in an image or screenshot

Can't (reliably) do — yet

- Guarantee facts — it can confidently make things up ("hallucinate")
- Know things that happened after its training cutoff (unless given tools)
- Do precise math without a calculator tool
- Truly understand what it says — it predicts plausible text
- Replace human judgment in high-stakes decisions
- Keep your data private if you paste into a public chat

What today's AI can and can't do — • Summarize and rewrite text • Translate between languages • Generate first drafts (emails, code, designs) • Answer questions over documents you provide • Extract s...

Four ways a model learns

Pick the paradigm by the shape of the data and the goal.

Supervised

What it is

Labeled examples → predict labels.

Example

Email → spam/not. X-ray → tumor/clean.

Unsupervised

What it is

No labels → find structure.

Example

Cluster customers. Detect anomalies.

Reinforcement

What it is

Agent acts, gets reward, adapts.

Example

Robotics, game-playing, RLHF for LLMs.

Self-supervised

What it is

Model creates its own labels from raw data.

Example

How GPT/BERT/Llama actually pretrain.

Most modern LLMs combine three: self-supervised pretraining → supervised fine-tuning → RL from human feedback (RLHF).

Four ways a model learns — Pick the paradigm by the shape of the data and the goal.

Deep learning architectures, by job

Different shapes for different data. Transformers ate the world after 2017.

MLP / Feedforward	Tabular data. Stack of fully-connected layers. The original neural net.	*Credit scoring, churn prediction*
CNN — Convolutional	Images. Filters slide over pixels and detect edges → shapes → objects.	*Face unlock, medical imaging, ResNet*
RNN / LSTM / GRU	Sequences. Reads token by token, carries hidden state. Now mostly retired.	*Old machine translation, speech-to-text*
Transformer	Sequences again — but with self-attention so every token sees every other token in parallel. The base of every modern LLM.	*GPT, Claude, Gemini, Llama, BERT*
Diffusion / VAE / GAN	Image and video generation. Learn to denoise random noise into a sample from the data distribution.	*Stable Diffusion, FLUX, Imagen, Sora*
Mixture of Experts (MoE)	Sparse: only a few 'expert' sub-networks fire per token. Bigger models, same compute.	*Mixtral, GPT-4, DeepSeek-V3*

Deep learning architectures, by job — Different shapes for different data. Transformers ate the world after 2017.

How an LLM actually works

It predicts the next token. Repeat. That's the whole engine.

INPUT

Your prompt becomes a sequence of tokens (≈ word pieces).

EMBED

Each token → vector. Position is added.

ATTENTION × N

Self-attention layers let every token weigh every other. Many layers stacked.

PREDICT

Output: probability over the next token. Sample one. Append. Repeat.

TOKENS — what the model sees

```
"ChatGPT writes well"
```

Tokenizer splits this into:

```
[ "ChatGPT" ] [ " writes" ] [ " well" ]
```

Why it matters

- Pricing is per token (≈ 3/4 of an English word).
- Context window = max tokens it can see at once. GPT-4o ≈ 128K, Claude Sonnet ≈ 200K, Gemini Pro ≈ 1–2M.
- When the window fills, the chat silently forgets — start a new chat with a one-line summary.
- Output tokens cost 3–5× input tokens, so cap output length.

How an LLM actually works — It predicts the next token. Repeat. That's the whole engine.

Generative AI — what it can produce now

One mental model: AI that creates, instead of just classifying or predicting.

TEXT

ChatGPT, Claude, Gemini, Llama, Mistral

Drafts, summaries, code, analysis

IMAGES

Midjourney, FLUX, Imagen 4, Stable Diffusion

Marketing visuals, mockups, art

AUDIO

ElevenLabs, Suno, OpenAI Voice

Voiceover, music, dubbing, podcasts

VIDEO

Veo 3, Runway Gen-4, Sora 2, Kling

Short clips, explainer animations

CODE

GitHub Copilot, Cursor, Claude Code, Codex

Autocomplete, refactor, full features

MULTIMODAL

GPT-4o, Gemini 2.5, Claude Sonnet 4

Read PDFs, charts, images; speak; see

Generative AI — what it can produce now — One mental model: AI that creates, instead of just classifying or predicting.

SECTION 01

01

What AI actually is

Three terms, one Russian doll: AI ⊃ ML ⊃ DL ⊃ Generative AI. AI is the umbrella — any system that does something we'd call intelligent.

Section opener — Section 01: Foundations of AI.

AI for research

Three modes: open-web search, your-own-docs, and deep multi-source synthesis.

Open-web search

Perplexity, Gemini Deep Research, ChatGPT Search

Fast factual lookup with inline citations.

```
"What changed in EU AI Act enforcement in early 2026? 5 bullets, link official sources."
```

Your-own-docs

NotebookLM, ChatGPT custom GPTs, Claude Projects

Upload PDFs/notes/audio. The model only answers from your sources.

```
"From these 12 papers, list every methodology section and group them by approach."
```

Deep research

ChatGPT Deep Research, Gemini Deep Research, Perplexity DR

Long-running agent that searches dozens of sources and writes a report.

```
"Build a competitive landscape of GLP-1 drug pipelines as of [date]. Include trial phase, sponsor, primary endpoint."
```

Always click at least one cited source. AI summaries can mis-quote even when the URL is real.

AI for research — three modes: open-web search, your-own-docs, and deep multi-source synthesis.

The tools to know (May 2026)

Seven assistants worth knowing — each card has a logo cue, a one-line role, and a prompt to paste.

ChatGPT

OpenAI · GPT-5.5 family

Everyday writing, brainstorming, coding help.

Try this prompt

```
"Act as my editor. Rewrite this paragraph for a busy exec. Under 80 words: …"
```

★ Voice, Canvas, file uploads, image gen, ChatGPT Images 2.0, agent mode.

Claude

Anthropic · Opus 4.5 / Sonnet 4.5

Long documents, careful reasoning, code review.

Try this prompt

```
"40-page contract attached. List 5 clauses likely to hurt the buyer, with quotes."
```

★ Projects, extended thinking, computer use for browser/desktop tasks.

Gemini

Google · Gemini 3 Pro / 3 Flash

Search, video & image, Workspace, Deep Research.

Try this prompt

```
"Summarize this 1-hour YouTube video into 10 bullets and 3 quotes I can share."
```

★ Reads images and video natively · 1M-token context · Deep Research mode.

Microsoft Copilot

Microsoft · 365 Copilot + Studio

People who live in Word, Excel, Outlook, Teams.

Try this prompt

```
(Excel) "Add a column flagging rows where revenue dropped >10% MoM, and explain the formula."
```

★ Runs GPT-5.5 Thinking · Copilot Studio builds custom agents · Copilot+ PCs.

Perplexity

Perplexity · Deep Research + Comet

Quick researched answers with sources.

Try this prompt

```
"What changed in EU AI Act enforcement in early 2026? 5 bullets and link the official sources."
```

★ Comet browser, persistent memory, picks GPT-5.5 / Claude 4.5 / Gemini 3.

Mistral · Le Chat

Mistral AI · Large 3 · EU-based

EU data residency, image gen, web search, agents.

Try this prompt

```
"Translate this German email into polite English and suggest a 2-line reply."
```

★ Open-weights (Mixtral, NeMo, Small 3) can run on your own server.

Llama 5 (Meta)

Meta · Llama 5 · open-weights (600B+)

Self-hosted or via Groq/Together for low cost.

Try this prompt

```
"Run Llama on your laptop with Ollama: ollama run llama5, then ask a question."
```

★ Free to download. Try via meta.ai, Groq Cloud, or Hugging Face.

Pick by use case

Match the job to the tool.

Write & edit	**ChatGPT**
Long PDFs	**Claude**
Image / video	**Gemini**
Office & Excel	**Copilot**
Sourced facts	**Perplexity**
EU / French	**Mistral**
Self-hosted	**Llama**

The tools to know (May 2026) — Seven assistants worth knowing — each card has a logo cue, a one-line role, and a prompt to paste.

Compare LLMs the way you'd compare cars

Stop picking models by vibe. Use the same prompt, same rubric, same scoring across models.

The 5-step comparison method

1 Pick one task

One concrete task like 'summarize this contract into 5 risks.' Not 'which is smarter.'

2 Write one frozen prompt

Same prompt text, same role, same format spec. Save it. Don't tweak per model.

3 Run on 3+ models

ChatGPT, Claude, Gemini at minimum. Add Mistral, Llama, Copilot for breadth.

4 Score on a rubric

Rate each output 1–5 on accuracy, completeness, format compliance, tone, and verifiability.

5 Spot-check facts

For any claim that matters, click through to the source or verify in the original document.

Example rubric (1 = bad · 5 = great)

Criterion	ChatGPT	Claude	Gemini
Accuracy of facts	4	5	3
Completeness	5	4	4
Format compliance	4	5	5
Tone match	4	4	3
Verifiable citations	3	4	5
Total / 25	**20**	**22**	**20**

Numbers above are illustrative. Always rerun on your own task — winners flip by domain.

Compare LLMs the way you'd compare cars — Stop picking models by vibe. Use the same prompt, same rubric, same scoring across models.

When to use Copilot vs ChatGPT vs Claude vs Gemini

There is no single winner. Pick the tool by the job and the context.

Job	Best default	Why	Avoid when
Productivity inside Word, Excel, PPT, Teams	**Microsoft Copilot**	Grounded in your live M365 data. Excel formulas + meeting summaries are unmatched in-app.	You don't have an M365 license.
Long PDF analysis · 100k+ token docs	**Claude**	Largest practical context window for free / low-tier users. Strong at faithful quoting.	You need cited web sources.
Sourced web research with live citations	**Perplexity / Gemini**	Both surface URLs and recent results. Perplexity is more aggressive about citation hygiene.	Privacy of the query matters.
Code in your IDE	**GitHub Copilot / Cursor**	Inline ghost-text completion + chat that sees your repo. Not a separate website.	You're not in a project — use Claude or ChatGPT for one-offs.
Brainstorming, drafting, general chat	**ChatGPT**	Default fluency. Strong tool ecosystem (GPTs, Code Interpreter, image gen).	You need fully sourced facts.
Open-source / self-hosted needed	**Llama / Mistral**	Run locally or on your VPC. No data leaves your network. Quality close to closed peers.	You don't have infra to host.

When to use Copilot vs ChatGPT vs Claude vs Gemini — There is no single winner. Pick the tool by the job and the context.

Hands-on workshop: 11 stations

Pick any 3 stations. Same prompt structure on every one. Bring your own real (non-confidential) text.

How each station works

Goal	What you'll have at the end of the station — concrete and shareable.
You need	Free account (or local install) and a real piece of text/data of your own.
Steps 1–4	Click-by-click. Same shape every time so you can predict the next move.
The prompt	Paste-ready. Coral box. Edit the [bracketed] parts only.
Success check	How you know you're done — not just 'it produced output.'
Gotcha	The mistake everyone makes their first time.

Score your prompt (3-point rubric)

Use this on every station. Aim for 3/3 before moving on.

1. **Role + audience set**
   `e.g. "Act as a contract reviewer for a non-lawyer buyer."`
2. **Output shape pinned**
   `e.g. Bullets, table, word count — anything precise.`
3. **Verification hook**
   `e.g. Quotes, citations, page numbers, or a 'say uncertain if unsure'.`

Deliverable

One Slack/email message you can send today. Paste the prompt + the output.

~~1. ChatGPT~~ ~~2. Claude~~ ~~3. Gemini~~ ~~4. Microsoft Copilot~~ ~~5. Perplexity~~ ~~6. Mistral~~ ~~7. Llama (Meta)~~

Hands-on workshop: 11 stations — Pick any 3 stations. Same prompt structure on every one. Bring your own real (non-confidential) text.

Station 1 — ChatGPT

GOAL Turn a draft email into a clean, action-oriented version.

You need

- A free chatgpt.com account
- A real (non-confidential) draft email or use the sample below

Steps

1. Open chatgpt.com → New chat.
2. Paste this prompt and your draft.
3. Read the rewrite. Ask: "Make it 30% shorter and add a clear ask in the first line."
4. Iterate two more times. Stop when you would actually send it.

PASTE THIS PROMPT

```
Act as my executive editor. Rewrite the email below for a busy reader.
Rules: under 90 words, one clear ask in line 1, friendly but direct, no
filler.
Return 2 versions: (a) formal, (b) casual.

Email draft:
[paste your draft]
```

Sample input → Hi team, just wanted to follow up on the thing we discussed last week about maybe possibly looking into the Q4 budget situation if you have time...

✓ SUCCESS CHECK

Output is ≤90 words, has a clear ask in line 1, and you would actually send it.

⚠ GOTCHA

Don't paste anything containing customer data or NDA material into a free tier.

Station 2 — Claude

GOAL Surface the 5 riskiest clauses in a 20-40 page document, with quotes.

You need

- claude.ai free account
- Any long PDF (lease, policy, T&Cs, paper). Use a sample contract if you don't have one.

Steps

1. Open claude.ai → Start a new chat.
2. Drag the PDF into the message box.
3. Paste the prompt below and send.
4. Ask follow-ups: "Quote clause 3 verbatim and propose a redline."

PASTE THIS PROMPT

```
You are a careful contract reviewer. Read the attached PDF.
Return:
1) The 5 clauses most likely to hurt the buyer, with the page number and a
1-sentence reason.
2) For each, a verbatim quote (max 30 words).
3) A suggested redline in plain English.
If you are unsure, say "uncertain" instead of guessing.
```

Sample input → Sample sources: any SaaS Master Services Agreement template (Google: "MSA template PDF").

✓ SUCCESS CHECK

You get exactly 5 clauses, each with a real quote and a page number.

⚠ GOTCHA

If quotes look paraphrased, ask: "Quote clause X word-for-word from the PDF."

Station 3 — Gemini

GOAL Make Gemini read an image and pull 3 takeaways + 1 question worth asking.

You need

- gemini.google.com (free Google account)
- A screenshot of any dashboard or chart (Excel, Looker, news graphic).

Steps

1. Open Gemini → click the paperclip → upload your image.
2. Paste the prompt and send.
3. Follow up: "What would I look at next to confirm takeaway #2?"
4. Try the same image in ChatGPT for a side-by-side comparison.

PASTE THIS PROMPT

```
Here is a screenshot of a dashboard.
1) Plain English: what are the 3 most important things to know?
2) What looks suspicious or missing?
3) Give me 2 questions to ask the analyst before I forward this to my
boss.
Don't invent numbers — only use what is visible in the image.
```

Sample input → Sample image: any public sales/marketing dashboard screenshot (search "sales dashboard screenshot").

✓ SUCCESS CHECK

The 3 takeaways match what you can actually see in the image.

⚠ GOTCHA

Models still hallucinate small numbers — verify any specific figure you plan to quote.

Station 4 — Microsoft Copilot

GOAL Get Copilot to write, explain, and apply an Excel formula on a sample table.

You need

- Microsoft 365 with Copilot (or copilot.microsoft.com free)
- A small Excel sheet with sales numbers — sample columns: Month, Region, Revenue.

Steps

1. Open Excel → put your data in a Table (Ctrl+T).
2. In Excel: Home tab → Copilot → "Add a column...".
3. Paste the prompt. Approve the suggested formula.
4. Ask Copilot: "Explain the formula step by step."

PASTE THIS PROMPT

```
Add a column called "MoM Drop" that is TRUE when Revenue dropped >10%
versus the same Region in the previous month.
Write the Excel formula, explain it line by line, and tell me one edge
case it does not handle.
```

Sample input → Sample table: 12 rows, columns Month (Jan-Dec), Region (EU/US), Revenue.

✓ SUCCESS CHECK

You get a working formula, a plain-English explanation, and one named edge case.

⚠ GOTCHA

Copilot in the free web version cannot edit your local file — it only suggests. You paste the formula yourself.

Station 4 — Microsoft Copilot — Get Copilot to write, explain, and apply an Excel formula on a sample table.

Station 5 — Perplexity

GOAL Get a 5-bullet answer with citations you can actually open and trust.

You need

- perplexity.ai (free, no login needed)
- A real question you would normally Google for 10 minutes.

Steps

1. Open perplexity.ai → switch to "Pro" search if available (free has limited Pro queries).
2. Paste the prompt below.
3. Click each citation. If the linked page does not say what was claimed, mark it ❌ in your notes.
4. Ask the follow-up: "What is the strongest counter-argument and who makes it?"

PASTE THIS PROMPT

```
Question: [your real question]
Return:
1) A 5-bullet answer.
2) Each bullet must have at least one citation linking to an authoritative
source (gov, peer-reviewed, named publisher — not Reddit/SEO blog).
3) A 1-line note on freshness: when was the most-recent source published?
```

Sample input → Sample question: "What changed in EU AI Act enforcement obligations during 2025?"

✓ SUCCESS CHECK

Every bullet has a clickable citation that actually supports the claim.

⚠ GOTCHA

Perplexity sometimes mis-attributes — at least one citation per session usually does not say what the bullet claims.

Station 5 — Perplexity — Get a 5-bullet answer with citations you can actually open and trust.

Station 6 — Mistral · Le Chat

GOAL Translate a short business message and make the tone fit the target culture.

You need

- chat.mistral.ai (free)
- A short message in any language. EN ↔ FR/DE/ES is where Mistral shines.

Steps

1. Open chat.mistral.ai → New chat.
2. Paste the prompt and your text.
3. Ask: "Suggest 2 alternative tones — one warmer, one more formal."
4. Spot-check with a native speaker if it is going to a real client.

PASTE THIS PROMPT

```
Translate the following from German to English.
Then:
1) Suggest a polite 2-sentence reply I can send.
2) Note any cultural conventions an American reader might miss.
3) Flag any phrase that does not translate cleanly and explain why.

Text:
[paste]
```

Sample input → Sample text: any 3-line German business email; templates are easy to find online.

✓ SUCCESS CHECK

The reply is one you would actually send, and you understand at least one cultural nuance you did not know.

⚠ GOTCHA

For high-stakes legal/medical translation, always have a human reviewer.

Station 6 — Mistral · Le Chat — Translate a short business message and make the tone fit the target culture.

Station 7 — Llama (Meta)

GOAL Run Llama 4.1 locally with Ollama, then get the same answer from a hosted version (Groq) for speed.

You need

- A laptop (8 GB RAM min for the 8B model)
- ollama.com (free, ~700 MB install)
- Optional: groq.com free API key for fast hosted inference.

Steps

1. Install Ollama → open a terminal.
2. Run: ollama run llama3.1 (downloads on first launch, ~5 GB)
3. Type your question. No internet needed once downloaded.
4. For speed: paste the same question into chat at groq.com (Llama 4.1-70B, ~500 tokens/s).

PASTE THIS PROMPT

```
You are a helpful assistant. In ≤6 bullets, explain what an open-weight
model is and why a small business might want to run one locally instead of
using a hosted API. Be concrete about cost and privacy.
```

Sample input → Sample command: ollama run llama3.1 → then paste the prompt at the >>> cursor.

✓ SUCCESS CHECK

You get a coherent answer locally and one from Groq, and you can name 1 trade-off between them.

⚠ GOTCHA

Local 8B models are weaker than ChatGPT/Claude on long reasoning. Use them for privacy-first or cost-first jobs.

Station 7 — Llama (Meta) — Run Llama 4.1 locally with Ollama, then get the same answer from a hosted version (Groq) for speed.

Station 8 —— GitHub Copilot

GOAL **Use Copilot to write a function from a docstring, then have it generate the tests until they pass.**

You need

- VS Code (or JetBrains) with the GitHub Copilot extension signed in
- A Python file open. A pinned terminal with pytest installed

Steps

1. Open a new file rolling.py. Type the docstring shown on the right and press Enter.
2. Accept Copilot's inline suggestion. Use Tab to accept, Esc to reject, Alt+] to cycle alternatives.
3. Open Copilot Chat. Ask it to generate pytest tests for rolling_zscore.
4. Run pytest. If a test fails, paste the failure into Copilot Chat: "Fix the function so this test passes." Repeat until green.

PASTE THIS PROMPT

```
def rolling_zscore(values: list[float], window: int) ->
list[float]:
    """Return the rolling z-score over `window` items.

    For index i < window-1 return None.
    Handle constant windows (std=0) by returning 0.
    """
    # Copilot: complete this function.
```

After accepting: ask Copilot Chat → "Generate edge-case pytest tests including empty list, window>len(values), and constant input."

✓ SUCCESS CHECK

All Copilot tests pass. The function returns None for the warm-up region and 0 (not NaN) for constant windows.

⚠ GOTCHA

Don't accept code blindly. Read every diff. Copilot will sometimes import a library you don't have or invent a helper that doesn't exist.

Station 8 —— GitHub Copilot

GOAL **Use Copilot to write a function from a docstring, then have it generate the tests until they pass.**

- VS Code (or JetBrains) with the GitHub Copilot extension signed in
- A Python file open. A pinned terminal with pytest installed

Steps

1. Open a new file rolling.py. Type the docstring shown on the right and press Enter.
2. Accept Copilot's inline suggestion. Use Tab to accept, Esc to reject, Alt+] to cycle alternatives.
3. Open Copilot Chat. Ask it to generate pytest tests for rolling_zscore.
4.

PASTE THIS PROMPT

```
def rolling_zscore(values: list[float], window: int) ->
list[float]:
    """Return the rolling z-score over `window` items.

    For index i < window-1 return None.
    Handle constant windows (std=0) by returning 0.
    """
    # Copilot: complete this function.
```

After accepting: ask Copilot Chat → "Generate edge-case pytest tests including empty list, window>len(values), and constant input."

✓ SUCCESS CHECK

All Copilot tests pass. The function returns None for the warm-up region and 0 (not NaN) for constant windows.

⚠ GOTCHA

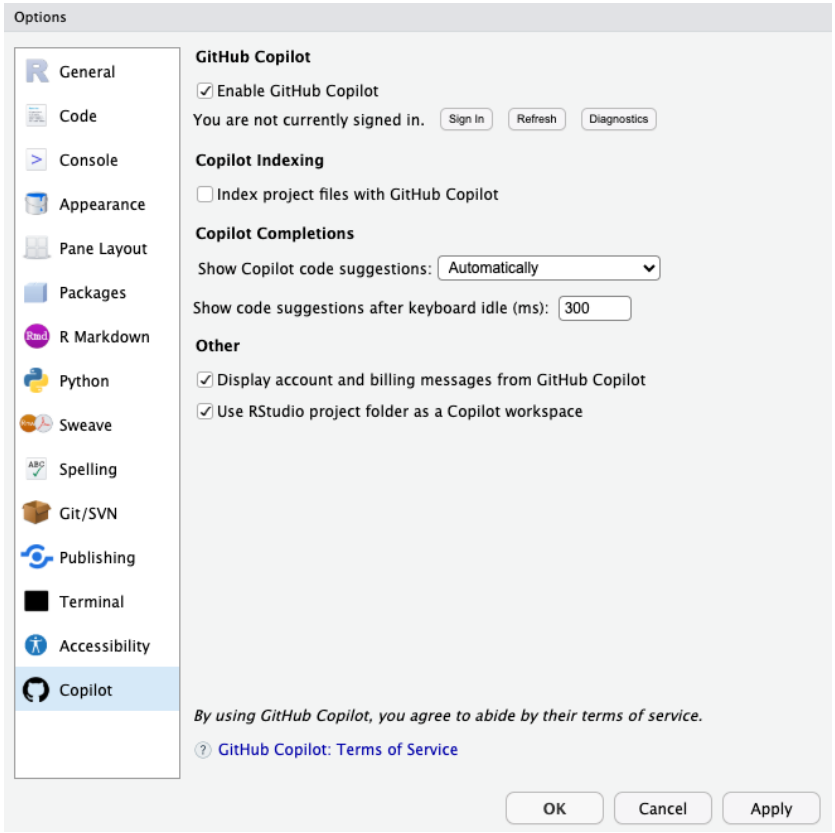
Don't accept code blindly. Read every diff. Copilot will sometimes import a library you don't have or invent a helper that doesn't exist.

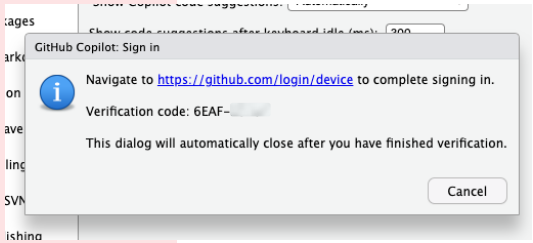
Station 8 —— GitHub Copilot

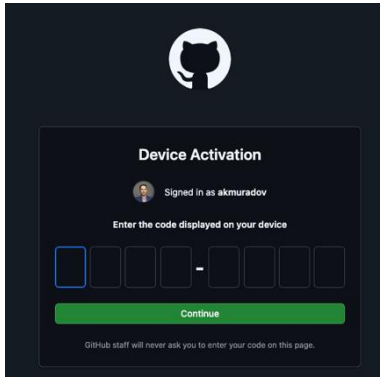
Using Copilot in RStudio

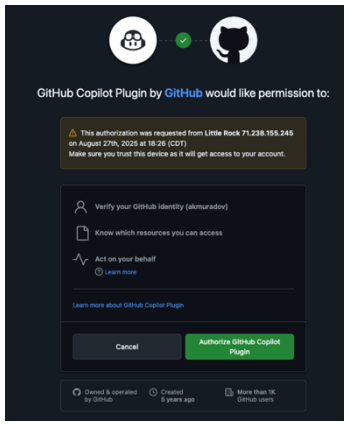
- Install or update RStudio Desktop (2023.09.0 or later)
- Create or sign in to a GitHub account
- Subscribe to GitHub Copilot
- Open RStudio and connect GitHub Copilot

RStudio Setup
Go to Tools → Global Options → Assistant
Choose GitHub Copilot
Sign in to GitHub when prompted
Open an .R file and start typing









Station 8 —— GitHub Copilot_x000B_Option 1 — Using Copilot in RStudio Install or update RStudio Desktop (2023.09.0 or later) Create or sign in to a GitHub account Subscribe to GitHub Copilot Open RStudio a...

Station 8 —— GitHub Copilot

Using Copilot in VS Code

- Install Visual Studio Code
- Install the R extension
- Install GitHub Copilot extension
- Sign in to GitHub
- Open an .R file to start coding

VS Code Workflow
Type comments describing your task
Copilot predicts and generates code
Accept suggestions with TAB
Use Copilot Chat for explanations and debugging

Station 8 —— GitHub Copilot

Advantages of Copilot in R

- Faster coding
- Reduces repetitive tasks
- Helps beginners learn R syntax
- Useful for teaching and research workflows

Limitation and Risk:
AI-generated code may contain errors
Always validate statistical outputs
May generate inefficient code
Do not paste confidential data into public AI systems

Example: SAS to R Conversion
Prompt: Convert SAS PROC MEANS into R code
Copilot can generate dplyr and ggplot workflows
Helpful for SAS programmers transitioning to R

Station 8 —— GitHub Copilot_x000B_Option 2 — Using Copilot in VS Code Install Visual Studio Code Install the R extension Install GitHub Copilot extension Sign in to GitHub Open an .R file to start coding…

Station 9 — Cursor

GOAL **Refactor a messy script with Cmd+K, then drive a multi-file change with Composer.**

You need

- Cursor installed (cursor.com), free tier is fine
- Any small Python project, or use the messy_orders.py sample below

Steps

1. Open Cursor → open the file → press Cmd+K (Ctrl+K on Win/Linux).
2. Type the inline prompt on the right. Review the diff. Accept or reject.
3. Press Cmd+I to open Composer. Drag in 2–3 related files (or @-mention them).
4. Ask Composer for a multi-file refactor: split logic, add CLI, update README, add tests.
5. Review every diff in the changes panel. Accept selectively. Run the tests.

PASTE THIS PROMPT

```
// Cmd+K (inline edit):
Refactor this file: extract the price-calculation logic
into a pure function calculate_total(line_items, tax_rate),
add type hints, and write a short docstring with one example.

// Cmd+I (Composer, multi-file):
Using @messy_orders.py as the source of truth, create a CLI
in @cli.py that accepts a CSV path and prints totals.
Add @README.md usage. Add @test_orders.py with 3 cases.
```

Sample input → a single 80-line script with mixed parsing, math, and printing all in one main() function.

✓ SUCCESS CHECK

Tests pass. Each new file is < 60 lines. README has a one-line install and a runnable example.

⚠ GOTCHA

Composer can edit many files at once. Always click through every diff before accepting — never bulk-accept changes you haven't read.

Station 10 — Kaggle

Free GPU notebooks + the AI assistant

GOAL **Open the Titanic dataset in a Kaggle notebook, run AI-assisted EDA, and train a baseline classifier with cross-validation.**

You need

- Free Kaggle account (kaggle.com)
- A web browser. Nothing installed locally

Steps

1. Go to kaggle.com/competitions/titanic → Code tab → New Notebook.
2. Click Run All to load the starter cell. Confirm train.csv and test.csv are attached.
3. Open the AI assistant (right panel). Ask the prompt on the right. Read each cell before running.
4. Add a final cell: train RandomForestClassifier with 5-fold CV; print mean accuracy and std.
5. Save Version → Submit Predictions to leaderboard. Note your score.

PASTE THIS PROMPT

```
// Kaggle AI assistant prompt:
Load train.csv. Show shape and dtypes.
Then:
1) Plot survival rate by Sex and by Pclass.
2) Show missing-value counts as a table.
3) Build a features dataframe with: Pclass, Sex (encoded),
   Age (median-filled), SibSp, Parch, Fare (log1p), Embarked (one-
hot).
4) Train a RandomForestClassifier with 5-fold CV.
   Print mean accuracy ± std.
Use seaborn for plots. Keep each step in its own cell.
```

Sample task → predict who survived the Titanic. Public leaderboard target: > 0.77 accuracy on test.csv.

✓ SUCCESS CHECK

Mean CV accuracy ≥ 0.80 on train. Submission appears on the leaderboard.

⚠ GOTCHA

The AI assistant doesn't always pick the best-validated approach. Treat it as a fast first draft, then iterate the features yourself.

Station 11 — Microsoft Copilot — an AI layer inside the apps you already use

Not a separate website you visit. It lives inside Word, Excel, PowerPoint, and Teams.

Big idea

Copilot works best when you already have data or content.

Open a real document, spreadsheet, or meeting.
Then ask Copilot to summarize, restructure, calculate, draft, or extract.

It is much weaker as a blank-page tool than ChatGPT or Claude. It is much stronger when grounded in your live Microsoft 365 data.

Source: Microsoft Copilot — Quick Tutorial (Business Use), §1.

Word

Rewrite, summarize, draft.

Excel

Analyze trends, write formulas, find anomalies.

PowerPoint

Word doc → slides; design clean-up; add structure.

Teams

Meeting summaries, action items, decisions.

Station 11 — Microsoft Copilot — an AI layer inside the apps you already use — Not a separate website you visit. It lives inside Word, Excel, PowerPoint, and Teams.

Microsoft Copilot in Word

Best when grounded in an existing draft, document, or email thread.

What to use it for

- Rewrite content for a different audience or length
- Summarize long documents into bullet points
- Create first drafts from rough notes

PRO TIP

Always give context.

Example: "This is for a board presentation. Keep it formal and concise. Do not invent any numbers — only use what's in the document."

EXAMPLE PROMPTS

"Rewrite this for an executive audience under 100 words."

"Summarize this document into 5 bullet points."

"Make this more persuasive."

Microsoft Copilot in Word — Best when grounded in an existing draft, document, or email thread.

Microsoft Copilot in Excel

The most powerful Copilot surface for non-developers. Treat it as a junior analyst.

What to use it for

- Analyze trends in a dataset
- Generate formulas you can read and verify
- Find anomalies, outliers, and missing data

PRO TIP

You don't need to know Excel formulas — Copilot writes them for you. But you do need to verify them.

When Copilot returns a formula, ask: "Explain this formula step by step and tell me one edge case where it would give the wrong answer."

EXAMPLE PROMPTS

"Highlight rows where revenue dropped more than 10% month-over-month."

"Create a formula to calculate growth rate."

"Summarize key insights from this dataset."

Example formula Copilot might generate:

```
=IF(B2=0, "N/A", (C2-B2)/B2)
```

Microsoft Copilot in Excel — The most powerful Copilot surface for non-developers. Treat it as a junior analyst.

Microsoft Copilot in PowerPoint

Best as a layout assistant on top of content you already wrote.

What to use it for

- Turn a Word document into a slide deck
- Make existing slides more visual and less text-heavy
- Add structure: agenda, section dividers, conclusion

PRO TIP

Workflow: write the story in Word first, then ask Copilot in PowerPoint to "turn into presentation."

The document structure (headings, bullets) becomes the slide structure. This works far better than asking Copilot to invent slides from a one-line prompt.

EXAMPLE PROMPTS

“Create a 5-slide executive summary from this report.”

“Make these slides more visual and less text-heavy.”

“Add a conclusion slide with key takeaways.”

Microsoft Copilot in Teams

Turn meetings from a fog into a record. Best with recording + transcript on.

What to use it for

- Summaries of long meetings
- Action items by person
- Capture decisions and who said what

PRO TIP

Set the meeting up for Copilot success.

Turn on recording. Use a clear meeting title. End with a 60-second verbal recap. Copilot's summary is only as good as the audio it ingests — so if the meeting was a mess, the summary will be too.

EXAMPLE PROMPTS

"Summarize this meeting in 5 bullets."

"List all action items by person."

"What decisions were made and what's still open?"

Workshop wrap-up: share, compare, repeat

Five minutes per station, then a 10-minute group share. Pattern beats novelty.

What worked

Which app gave the cleanest output? Why? Note the prompt structure that won — most likely role + format + verification.

What didn't

Where did the model hallucinate, miss context, or refuse? Was it the model, or your prompt? Iterate the prompt before blaming the model.

Next prompt

Pick one prompt you'll save into your notes app today. Add the {input} placeholder. Re-run it next week with new content.

Three habits to take home

1. Always set a role in line 1.
2. Always pin the output shape (bullets, table, word count).
3. Always ask for a verification hook (quotes, citations, 'say uncertain').

Next slide: anatomy of a good prompt — formalizes what you just practiced.

Workshop wrap-up: share, compare, repeat — Five minutes per station, then a 10-minute group share. Pattern beats novelty.

AI for coding

Three layers, from autocomplete to autonomous agents inside your editor.

Inline autocomplete

GitHub Copilot, Cursor Tab, Windsurf

Suggests the next line as you type. Best for boilerplate, tests, repetitive patterns.

TIP — Always read before accepting. Tab → accept, Esc → reject.

Chat / edit-with-context

Cursor (Cmd-K, Cmd-L), Copilot Chat, Continue, Claude Code

Highlight code → ask for refactor, explanation, tests, bug fix. Model sees your file or repo.

TIP — Pin the right files. The model can only help with what it can see.

Agentic coding

Cursor Composer / Agent, Claude Code, Devin, Aider, OpenAI Codex

Multi-step: reads repo, edits multiple files, runs tests, iterates until green.

TIP — Run on a branch. Always diff and review before merge.

AI for coding — Three layers, from autocomplete to autonomous agents inside your editor.

AI for data exploration

Stop fighting formulas. Start asking questions.

Copilot in Excel

Built into Microsoft 365 Excel. Sees your table.

- Write formulas you don't know
- Highlight anomalies and trends
- Add computed columns from a description
- Explain a formula line by line

TRY THIS

"Add a column flagging rows where Revenue dropped >10% versus the same Region in the previous month."

ChatGPT / Claude (Code Interpreter)

Upload CSV/Excel/Parquet. Runs Python in a sandbox.

- Plot distributions and time series
- Run regressions, t-tests, clustering
- Pivot, join, dedupe
- Returns the .py and the result file

TRY THIS

"Load the CSV, find the 5 customers with highest churn risk based on usage drop. Plot them."

Kaggle Notebooks

Free GPU/TPU. Public datasets. AI assistant inside the notebook.

- Free compute (limits apply)
- 50K+ public datasets one click away
- Built-in code suggestions
- Public competitions for practice

TRY THIS

"Train a baseline gradient-boosted model on the Titanic dataset; report cross-val accuracy."

AI for data exploration — Stop fighting formulas. Start asking questions.

AI sees, hears, and speaks

Modern assistants take more than text. Stop limiting yourself to typing.

Image in

Photo of a whiteboard → typed minutes. Photo of a dish → recipe. Screenshot of an error → diagnosis.

PDF / chart in

Drop a 40-page contract → 5 risky clauses with quotes. Drop a chart image → describe trend, get the numbers.

Voice in

Talk through a thought → get a clean draft back. Faster than typing for brainstorming.

Voice out

Read a long email reply aloud while you walk. Turn a study guide into a podcast (NotebookLM).

Video in

Paste a YouTube URL into Gemini → 10-bullet summary + timestamps.

Image out

Generate diagrams, mockups, social images directly in chat.

RAG — AI that reads your documents

Retrieval-Augmented Generation: combine an LLM with your own knowledge base. Reduces hallucinations.

1	2	3	4	5
LOAD	**SPLIT**	**EMBED**	**RETRIEVE**	**GENERATE**
Ingest PDFs, web pages, notes, transcripts.	Break documents into chunks (~500–1000 tokens).	Convert each chunk to a vector. Store in a vector DB.	User query → vector search → top-k relevant chunks.	Feed retrieved chunks + query to the LLM. Grounded answer.

USE RAG WHEN

- Your data is not on the public internet (internal wiki, contracts, support tickets).
- Facts change often — you'd rather update a document than retrain a model.
- You need source citations — RAG can return the chunk it used.

TOOLS THAT DO RAG FOR YOU

- NotebookLM — drop in sources, ask questions, get cited answers.
- ChatGPT custom GPTs / Claude Projects — attach files, persistent context.
- LangChain, LlamaIndex, Vercel AI SDK — build your own with code.

RAG — AI that reads your documents — Retrieval-Augmented Generation: combine an LLM with your own knowledge base. Reduces hallucinations.

02

Using AI today

Prompts that work, and the most common ways people get more done with AI.

Anatomy of a good prompt

Four pieces, every time. The framework that beats every clever trick.

ROLE

Who is the model pretending to be?

"Act as a senior contract reviewer for a non-lawyer buyer."

TASK

What is the single, specific thing you want done?

"Flag clauses that hurt the buyer and explain each one."

CONTEXT

Audience, goal, constraints, source material.

"Audience: a small-business owner. Tone: plain English. Source: contract attached."

FORMAT

How should the answer be shaped?

"Return a 5-row table: Clause | Risk | Fix in plain English."

The daily template: "Act as ___, do ___, for ___, in ___ format."

Anatomy of a good prompt — Four pieces, every time. The framework that beats every clever trick.

Why prompts matter

Think of it as instructions to a very smart but very literal intern.

- AI follows what you write — not what you meant.
- Bad input → bad output. No amount of model capability fixes a vague prompt.
- Specificity beats cleverness: clear beats clever, every time.
- You are the director. The model is the actor.

Why prompts matter — Think of it as instructions to a very smart but very literal intern.

Prompt Engineering — The Art of Talking to AI

💡 **Think of it this way:** ChatGPT is a brilliant new employee. How clearly and specifically you brief them determines the quality of their output. Vague brief → generic results. Precise brief → exceptional results.

Bad Prompt → Bad Output

Write an email.

Summarize this.

Write SAS code.

Good Prompt → Great Output

I'm a pharma sales rep. Write a 100-word cold email to an oncologist introducing our new targeted therapy for HER2+ breast cancer. Tone: professional yet warm.

Summarize this clinical protocol in 5 bullet points, focusing on primary endpoints, patient population, and key exclusion criteria. Max 20 words per bullet.

Write SAS code to merge ADSL and ADAE datasets on USUBJID, keeping only treatment-emergent adverse events (AETR='Y'), sorted by subject then AESTDTC.

Anatomy of a good prompt

A prompt is just text input. But four ingredients make it work better.

Role	Tell the model who it is.	*"You are a patient math tutor for 12-year-olds."*
Task	Say exactly what you want.	*"Explain fractions using a pizza analogy."*
Context	Give relevant background.	*"The student knows whole numbers but is new to fractions."*
Format	Specify the shape of the answer.	*"Reply in 3 short paragraphs, then 2 practice questions."*

Anatomy of a good prompt — A prompt is just text input. But four ingredients make it work better.

Anatomy of a good prompt

Four pieces, every time. The framework that beats every trick.

Role

Who is the model pretending to be? "Act as a senior editor…"

Task

What is the single, specific thing you want done?

Context

Audience, goal, constraints, and any source material.

Format

How should the answer be shaped — bullets, table, length?

Anatomy of a good prompt — Four pieces, every time. The framework that beats every trick.

Common mistake

What a vague prompt looks like — and what you get back.

❌ VAGUE PROMPT

"Write something about AI."

WHAT YOU GET

A generic, lukewarm essay. No audience, no angle, no length, no tone. Good for nothing in particular.

Common mistake — What a vague prompt looks like — and what you get back.

Better approach

Specific. Constrained. Goal-aware.

✅ **CLEAR + SPECIFIC**

"Write a 200-word LinkedIn post for marketers about how AI changed copywriting in 2026. Friendly tone, end with a question."

WHY IT WORKS

Audience, format, length, tone, and a call-to-action are all pinned down. The model has fewer choices to make — and makes the right ones.

Better approach — Specific. Constrained. Goal-aware.

Bad prompt vs. good prompt

DEMO 2

TRY THIS

1. Type: "Write something about productivity."

2. Read the result. It will feel generic.

3. Now: "Act as a productivity coach. Write a 150-word LinkedIn post for busy parents about one practical morning habit. Friendly tone, end with a question."

4. Compare side by side.

WHAT TO NOTICE

Same model, completely different output. The difference is your prompt — not the AI.

Bad prompt vs. good prompt — 1. Type: "Write something about productivity."

7 Ways to Write Better Prompts

1 Be Specific

"Write a 100-word email to a cardiologist about drug X's safety profile"

2 Add Constraints

"...within $200 budget, for 10 people, in bullet format, max 5 bullets"

3 Add Context

"I'm a new SAS programmer. Explain this proc mixed output as if I'm a statistician."

4 Set a Role

"You are an FDA regulatory expert. Review this ICF for compliance issues."

5 Give Examples

"Like this existing SOP section [paste example], write the section on data backup."

6 Probe Further

"Tell me more about point 3 above." / "Expand the second paragraph."

7 Iterate & Refine

After each response: "Make it shorter." / "More formal." / "Add a call to action."

7 Ways to Write Better Prompts' — "Write a 100-word email to a cardiologist about drug X's safety profile"

Prompt Engineering Techniques

Zero-Shot

Ask directly with no examples. Best for simple, well-defined tasks.

```
Classify this text as positive, negative, or neutral:
"The drug showed minimal side effects."
```

→ **Positive**

Few-Shot

Provide 2–5 examples to teach the pattern before asking the real question.

```
Headache after dose → Safety Issue
Patient tolerated well → No Safety Issue
Dizziness reported →
```

→ **Safety Issue**

Chain-of-Thought

Instruct the model to reason step-by-step before giving the final answer.

```
Let's think step by step:
1. Is there an adverse event?
2. Is there patient info?
3. Is there drug info?
If all yes → Safety Issue
```

→ **Structured reasoning + answer**

Reflection

Ask a 'Helper Agent' to critique and improve the AI's first response.

```
ChatGPT: [first answer]
Helper Agent: Identify weaknesses in the above.
ChatGPT: Revised answer incorporating feedback.
```

→ **Higher-quality output**

Prompt Engineering Techniques – Zero-Shot: Ask directly with no examples. Best for simple, well-defined tasks.

The TICG Prompt Template Framework

# Identity Who are you?	I am a SAS programmer / clinical data manager / regulatory writer.
# Task What do you need?	*Create SAS code to map raw EDC data to SDTM DM domain per CDISC IG v3.4.*
# Context Background details	*This is a Phase 3 oncology study. Data comes from Medidata Rave export as SAS datasets.*
# Guidance Examples or rules	*Follow this existing mapping spec format: [paste example]. Output only the SAS code.*

The TICG Prompt Template Framework — I am a SAS programmer / clinical data manager / regulatory writer.

💡 *Tip: The more structured your template, the more consistent and repeatable your AI outputs.*

Tips for prompting

Three habits that turn 'meh' answers into useful ones.

01

Be detailed and specific

Give context. Spell out the task. Tell the model what 'good' looks like.

02

Guide the model to think through its answer

Break the request into steps. Ask for intermediate reasoning before the final answer.

03

Experiment and iterate

There's no perfect prompt. Try, evaluate, refine — repeat.

Tips for prompting — Three habits that turn 'meh' answers into useful ones.

Be detailed and specific

Give the model enough context to do the task well.

✗ Vague prompt

"Help me write an email asking to be assigned to the legal documents project."

Why it falls short: no context about you, no audience, no tone.

✓ Detailed prompt

"I'm applying for a job on the legal documents project, which will check legal documents using LLMs. I have ample experience prompting LLMs to generate accurate text in a professional tone. Write a paragraph of text explaining why my background makes me a strong candidate to this project and advocate for my candidacy."

Context + task + tone → much more useful output.

Be detailed and specific — Give the model enough context to do the task well.

Guide the model to think through its answer

Break a request into steps and the answers get sharper.

One-shot prompt

"Brainstorm 5 names for a new cat toy."

What you usually get: a flat list, hit-or-miss.

Stepwise prompt

Step 1: Come up with 5 fun, joyful words that relate to cats.

Step 2: For each word, come up with a rhyming name for a toy.

Step 3: For each toy name, add a fun, relevant emoji.

Asking for intermediate steps gives the model room to reason.

Guide the model to think through its answer — Break a request into steps and the answers get sharper.

Guide the model to think — model response

The output you get when you ask for the steps.

Step 1: 5 fun cat words	Step 2: Rhyming toy name	Step 3: Add an emoji
Purr	Purr-Twirl	Purr-Twirl 🌀
Whisker	Whisker-Whisper	Whisker-Whisper 🤫
Feline	Feline-Beeline	Feline-Beeline 🐾
Pounce	Pounce-Bounce	Pounce-Bounce ⚽
Meow	Meow-Wow	Meow-Wow 😺

The structured prompt produces a structured answer — and one you can iterate on.

Guide the model to think — model response — The output you get when you ask for the steps.

Experiment and iterate

There is no perfect prompt for every person or situation. Build a process instead.

v1

Prompt

Help me rewrite this: […]

↓ evaluate output, refine prompt

v2

Prompt

Correct any grammatical and spelling errors in this: […]

↓ evaluate output, refine prompt

v3

Prompt

Correct any grammatical and spelling errors in this, and rewrite in a tone appropriate for a professional resume: […]

Experiment and iterate — There is no perfect prompt for every person or situation. Build a process instead.

Iteratively improving your prompt

A simple loop you can run in your head every time.

Prompting process

1. Be clear and specific in your prompt.
2. Send it. Read the response carefully.
3. Ask: why isn't this giving the desired output?
4. Refine the prompt — add context, examples, constraints.
5. Repeat until it's good enough for the job.

Tip: don't overthink the first try. Ship a draft prompt, then improve it.

⚠ Caveats

Confidential information

Don't paste passwords, customer data, secrets, or anything you wouldn't email to a stranger.

Trust the output?

LLMs can sound confident and still be wrong. Verify facts, numbers, citations, and code before using them.

Mental model

Treat the LLM as a fast, eager intern: useful, but you check the work.

Iteratively improving your prompt — A simple loop you can run in your head every time.

Image generation

Generative AI applications — making pictures, not just words.

What is image generation?

Models that turn a text prompt into a brand-new image. Trained on huge collections of images and captions, they learn visual patterns and can compose new pictures on demand.

How a prompt looks

"A watercolor of a calico cat reading a book on a windowsill, soft morning light, cozy mood."

Describe subject, style, lighting, mood — like briefing an illustrator.

Common uses & tools (May 2026)

Use cases	Marketing visuals, mockups, storyboards, slide art, concept art, social posts.
Popular tools	Midjourney, DALL·E (in ChatGPT), Stable Diffusion, Adobe Firefly, Google Imagen / Nano Banana.
Watch-outs	Image rights, deepfakes, stylistic copying, brand consistency, hands and text still glitchy.

Image generation — Generative AI applications — making pictures, not just words.

Five prompting tricks worth knowing

1

Be specific

Vague: "Make this email better." → Specific: "Rewrite to be 30% shorter, friendly, and end with a clear next step."

2

Show, don't just tell

Give 2–3 examples of the style or output you want. Models copy patterns very well.

3

Ask for steps

"Think step by step" or "List your reasoning before the answer." Helps with logic problems.

4

Iterate, don't restart

Reply "Make it shorter," "Use a warmer tone," "Add a counterargument." Build on the last answer.

5

Have it critique itself

"Now act as a strict editor and find three weaknesses in the above."

Five prompting tricks worth knowing — Vague: "Make this email better." → Specific: "Rewrite to be 30% shorter, friendly, and end with a clear next step."

Example: assigning a role

A role narrows the model's behavior fast.

PROMPT

"Act as a marketing expert reviewing a landing page. Read the copy below and give me three specific changes that will lift conversion. Be blunt."

Roles set tone, vocabulary, and standards. The same task without a role is generic.

Example: assigning a role — A role narrows the model's behavior fast.

Add context

Context is what separates "AI wrote this" from "this works."

- Audience — who is going to read this? Beginners, executives, engineers?
- Goal — what should the reader think, feel, or do next?
- Constraints — length, tone, format, things to avoid.
- Source material — paste the article, transcript, or data it should use.

Add context — Context is what separates "AI wrote this" from "this works."

Pin down the output format

Tell the model exactly how to respond.

- Bullet points for scannable lists.
- Table for comparisons (specify columns).
- Numbered steps for procedures.
- Short summary first, details after — "TL;DR then long version."

Pin down the output format — Tell the model exactly how to respond.

Show, don't just tell

Few-shot prompting — give the model examples of what "good" looks like.

PATTERN

"Here are 3 examples of subject lines I love: [paste them]. Now write 10 more in the same style for [topic]."

Examples teach style, tone, and format faster than any description ever will.

Constraints make output sharper

Limits force decisions. Decisions make writing concrete.

- Word or character limits — "under 80 words."
- Tone — "plain, no marketing speak."
- Style — "like Hemingway, short sentences."
- Banned words — "no clichés, no 'delve,' no 'in today's world.'"

Constraints make output sharper — Limits force decisions. Decisions make writing concrete.

Always ask for options

Don't pick the first answer — ask for several and choose.

- "Give me 5 versions: punchy, formal, casual, contrarian, funny."
- "Now combine the best parts of versions 2 and 4."
- Comparison is faster than perfection on the first try.
- You'll notice patterns the model defaults to — and you can avoid them.

Always ask for options — Don't pick the first answer — ask for several and choose.

Tone is a parameter

Naming the tone gets you 80% of the way there.

- Formal — for legal, academic, executive contexts.
- Casual — for social posts, friendly emails, internal Slack.
- Professional — for client work, proposals, reports.
- Try unusual ones: "like a tired expert," "like a smart friend."

Tone is a parameter — Naming the tone gets you 80% of the way there.

Tip 1: use roles deliberately

Pick the role to match the standard you want applied.

- "Act as a strict editor" — for tightening writing.
- "Act as a hostile reviewer" — for stress-testing arguments.
- "Act as a curious beginner" — for finding gaps in your explanation.
- Switch roles in the same chat to get different angles fast.

Tip 1: use roles deliberately — Pick the role to match the standard you want applied.

Tip 2: break tasks into small pieces

The model performs better on a small, sharp ask than a sprawling one.

- Outline → draft → edit, as separate prompts.
- Brainstorm → critique → pick → expand.
- Each turn is short, easy to verify, and easy to redo.
- You stay in control of every checkpoint.

Tip 2: break tasks into small pieces — The model performs better on a small, sharp ask than a sprawling one.

Tip 3: ask for alternatives

Default to plurals. Single answers hide the design space.

- "Give me 5 versions across different tones."
- "Now give me 3 unconventional takes."
- "Combine the best of #2 and #4."
- Cheap to generate. Easy to compare. Better to choose from.

Tip 3: ask for alternatives — Default to plurals. Single answers hide the design space.

Tip 4: combine outputs

Stitching beats picking. Use the model to merge its own drafts.

- "Take the structure of A and the tone of B."
- "Use the examples from version 1 and the conclusion from version 3."
- "Now smooth the seams so it reads as one piece."
- You get something none of the individual drafts contained.

Tip 4: combine outputs — Stitching beats picking. Use the model to merge its own drafts.

Tip 5: build workflows, not prompts

A workflow is a small pipeline you reuse across many inputs.

- Save a prompt template for emails, summaries, and reviews you do often.
- Chain steps the same way every time — predictability is the win.
- Note the prompts that fail and refine them next time.
- Treat your prompt library like a personal toolset.

Tip 5: build workflows, not prompts — A workflow is a small pipeline you reuse across many inputs.

Tip 6: practice daily

Five minutes a day beats one big training session.

- Run one real task through ChatGPT every day for a month.
- Compare "with AI" vs. "without AI" on the same task.
- Keep the prompts that worked. Throw out the rest.
- Skill compounds — a year of small reps is enormous.

Tip 6: practice daily — Five minutes a day beats one big training session.

Ask → refine → improve.

First answers are drafts. Treat the conversation as the work, not the question.

Workflow theme card — building repeatable AI workflows, not one-off prompts.

Use follow-ups

The second prompt is usually the one that earns its keep.

- "Make it shorter." / "Make it punchier."
- "Now do the same for a different audience."
- "What did you assume? List your assumptions."
- "Critique your own answer and rewrite it."

Use follow-ups — The second prompt is usually the one that earns its keep.

Chain prompts

Break a big task into a sequence the model can actually do well.

- Step 1 — Brainstorm 10 angles for a blog post.
- Step 2 — Pick the best 3 and outline each.
- Step 3 — Draft the chosen outline into a full post.
- Step 4 — Edit for tone, length, and SEO.

Chain prompts — Break a big task into a sequence the model can actually do well.

The daily prompt template

Save this. Use it every time you don't know where to start.

TEMPLATE

"Act as ___, do ___, for ___, in ___ format."

Example: "Act as a startup advisor, review my pitch, for a seed-stage SaaS investor, in 5 bullet points."

The daily prompt template — Save this. Use it every time you don't know where to start.

Repeat tasks → AI.

If you do it more than twice the same way, write a prompt template for it.

Automation theme card — turning prompts into multi-step automations.

Multi-step tasks

Don't ask for the whole cake. Ask for the recipe, then bake step by step.

- Outline first, then expand each section in its own turn.
- Each step is shorter — easier for the model, easier for you to check.
- Catch errors early before they compound across the whole answer.
- You stay in the driver's seat — the model assists, doesn't decide.

Multi-step tasks — Don't ask for the whole cake. Ask for the recipe, then bake step by step.

Combine ChatGPT with other tools

ChatGPT plus one other tool is usually stronger than ChatGPT alone.

- ChatGPT + a spreadsheet — generate, then paste, then verify.
- ChatGPT + a search engine — draft, then fact-check links.
- ChatGPT + a code editor — generate, paste, run, iterate.
- ChatGPT + your notes app — turn raw thoughts into clean outputs.

Combine ChatGPT with other tools — ChatGPT plus one other tool is usually stronger than ChatGPT alone.

Three ways to put AI into your work

Pick the simplest one that solves the problem. Most jobs stop at level 1.

PROMPTING	RAG	FINE-TUNING	AGENTS
Just write better prompts. No code.	Give the model your own documents, then ask questions.	Train the model on your style or task.	Let the model choose and execute multi-step actions.
Best for: drafts, brainstorms, one-off tasks.	Best for: internal Q&A, support, research over your library.	Best for: consistent format/tone you can't get with prompting.	Best for: research, browser tasks, repeatable workflows.
Effort: minutes	*Effort: hours–days*	*Effort: days–weeks; needs ≥100 examples*	*Effort: high; supervise carefully*
Tools: any chat (ChatGPT, Claude, Gemini, Copilot)	Tools: NotebookLM, Custom GPTs, LangChain, LlamaIndex	Tools: OpenAI/Anthropic fine-tune APIs, open-weight LoRA	Tools: Cursor Agent, ChatGPT Agent, Claude Code, n8n + LLM

Three ways to put AI into your work — Pick the simplest one that solves the problem. Most jobs stop at level 1.

Don't fine-tune before you've maxed out prompting.

Demo 1 — Turn a messy meeting into clean notes (ChatGPT)

Five minutes. Works the same in Claude or Gemini.

1 Paste raw input

Drop in the meeting transcript, voice memo, or your bullet notes. Don't clean it up.

2 Tell it the role

"Act as my chief of staff." This sets tone and detail level.

3 Ask for the exact format

Decisions, action items (with owners), risks, follow-ups. Specify bullets.

4 Iterate, don't restart

"Make the action items shorter." "Add a 2-line email to the team."

⚡ **Tip: never paste sensitive data into a free tier you can't verify.**

PROMPT

```
Act as my chief of staff. Below is a raw transcript of a 30-minute team
meeting.

Give me:
1) Decisions made (one bullet each)
2) Action items as a table: Owner | Task | Due
3) Open questions
4) A 4-line email I can send to the team tonight.

Transcript: <<<paste here>>>
```

WHAT YOU GET BACK (sample)

```
Decisions
• Launch date pushed to Mar 14 to allow QA buffer.
• Pricing stays at €49/mo for the pilot cohort.

Action items
Owner   | Task                          | Due
Sara    | Send updated roadmap          | Fri
Marco   | Draft pricing FAQ             | Wed
Lina    | Confirm QA capacity           | Mon

Open questions
• Who owns customer comms on launch day?

Email
"Team — quick recap: launch moves to Mar 14, pricing holds at €49. Sara
owns the roadmap update by Friday. Reply if I missed anything. — M"
```

Demo 1 — Turn a messy meeting into clean notes (ChatGPT) — Five minutes. Works the same in Claude or Gemini.

Demo 2 & 3 — Long PDFs, charts, and quick research

Pick the tool that fits the input you have.

DEMO 2

Claude — read a long PDF

When: 30+ page contract, paper, or report.

PROMPT

```
You are a careful contract reviewer.

Read the attached 42-page MSA. Then return:
1) The 5 clauses that most favor the vendor
(quote them).
2) The 3 clauses I should push back on,
with suggested redlines.
3) Any defined terms used inconsistently.

If you are unsure about a clause, say so —
do not guess.
```

WHY THIS WORKS

- Role ("careful contract reviewer") sets standards.
- Numbered output is easy to copy into a memo.
- "Quote them" forces grounding in the document.
- Permission to say "unsure" reduces hallucinated clauses.

DEMO 3

Gemini — explain a chart

When: you have a screenshot, dashboard, or messy slide.

PROMPT (with image attached)

```
Here is a screenshot of our Q3 sales
dashboard.

1) In plain English, what are the 3 most
important takeaways?
2) What looks suspicious or worth double-
checking?
3) Draft 2 questions I should ask the
analyst who built it.

Speak to a non-technical executive.
```

TIP

- Gemini reads images natively — no transcribing needed.
- Asking "what looks suspicious" surfaces outliers it noticed.
- "Speak to a non-technical executive" controls the tone.
- Works the same on a photo of a whiteboard.

BONUS

Perplexity — researched answer

When: you need facts with sources, fast.

PROMPT

```
Compare the EU AI Act and the US AI
Executive Order as of early 2026.

Give me:
1) A 5-row table of what each regulates.
2) The 3 biggest differences for a SaaS
company.
3) The official source URL for each row.
```

CHECK BEFORE TRUSTING

- Click every citation — Perplexity sometimes mis-attributes.
- Watch the publish date on each source.
- Re-run the same query in ChatGPT or Claude to cross-check.

Try it now: rewrite this email

Paste the gray box into ChatGPT, Claude, or Gemini. Compare the results.

PROMPT

```
You are a friendly office assistant. Rewrite the email below so it is 30% shorter, polite, and ends with a
clear question. Keep the meaning intact. After the rewrite, list two changes you made and why.

EMAIL:
Hi team, just wanted to circle back on the report from last week — I haven't heard anything and I'm wondering
if anyone has had a chance to look it over yet because the deadline is sort of coming up soon and I want to
make sure we are aligned. Thanks!
```

TIP If the answer is too formal, reply "Make it warmer." If it's too long, reply "Cut it in half." Don't start over — refine.

Generate an email or report

DEMO 3

TRY THIS

1. Tell ChatGPT the situation in plain English.

2. Specify recipient, goal, tone, and length.

3. Ask for two versions: short and long.

4. Pick one, then ask for one targeted edit.

WHAT TO NOTICE

Three minutes, draft done. The win is not the first answer — it's how fast you can iterate to a usable one.

Generate an email or report — 1. Tell ChatGPT the situation in plain English.

Brainstorm business ideas

DEMO 4

TRY THIS

1. "Give me 20 business ideas at the intersection of AI and [industry you know]."
2. Pick the 3 most interesting.

3. "For each, list 3 reasons it could fail."

4. "Now pick the strongest one and outline a 30-day validation plan."

WHAT TO NOTICE

Quantity first, then critique, then plan. The model is great at all three — and embarrassing if you stop after step 1.

Brainstorm business ideas — 1. "Give me 20 business ideas at the intersection of AI and [industry you know]."

Summarize an article

DEMO 5

TRY THIS

1. Paste a long article into ChatGPT.

2. "Summarize in 5 bullets, then list the 3 weakest claims."

3. Then: "What's missing from this article that an expert would notice?"

4. Save the summary. Skip rereading the article later.

WHAT TO NOTICE

This is the workflow that pays for ChatGPT all by itself for most knowledge workers.

Summarize an article — 1. Paste a long article into ChatGPT.

Iterate to a usable answer

DEMO 6

TRY THIS

1. Take any output you got earlier.

2. Ask: "What's weak about this?"

3. Ask: "Rewrite using your own critique."

4. Ask: "Now cut 30% without losing meaning."

WHAT TO NOTICE

Self-critique loops are the cheapest quality upgrade you'll ever apply. Two extra prompts, dramatically better output.

Iterate to a usable answer — 1. Take any output you got earlier.

Use Case: Coding & Data Analysis

Example: Code Conversion (SAS → Python)

PROMPT: Convert this SAS code to Python using pandas. Explain any differences.

```
data dm; merge demo rand; by usubjid; run;
```

SAS Input

```
data dm;
  merge demo rand;
  by usubjid;
run;
```

Python Output (AI Generated)

```
import pandas as pd

dm = pd.merge(demo, rand,
              on='usubjid',
              how='outer')
```

Example: Inline Data Analysis

Filter

Select patients where age >= 40 and race = 'White' from the Patient Data below.

✓ 2 patients returned

Sort

Sort Patient Data by age ascending. Show only patient ID and age.

✓ 01-001 (25), 01-002 (28)...

Summarize

Count patients by race from the dataset. Present as a small table.

✓ White: 2, Asian: 1, Black: 1

Use Case: Clinical & Healthcare Applications

Safety Case Classification

Prompt:

Classify: Does this narrative contain an AE + patient info + medication? If all yes → Safety Issue.

"Patient 002 experienced severe headache 3hrs after taking Drug A." → SAFETY ISSUE ✓

Protocol Synopsis Generation

Prompt:

Write a clinical trial synopsis for: 'Phase 3, randomized, double-blind, placebo-controlled study evaluating Drug A in schizophrenia.'

Generates: Objectives, design, population, endpoints, statistical methods, ethics — in seconds.

PANSS Mental Health Scoring

Prompt:

Given patient's answers below, assign PANSS Anxiety score (Absent/Minimal/Mild/Moderate/Severe/Extreme). Final score = most severe.

Q4: 'I worry a lot' → Moderate. Q5: 'Heart racing' → Moderate Severe. Final: Moderate Severe

Patient Recruitment Screening

Prompt:

From this EHR excerpt, determine if patient meets inclusion criteria: Age 40–65, HbA1c > 7.5%, no prior GLP-1 use.

Screens thousands of records vs manual review — saving weeks of coordinator time.

Scientific Literature Summary

Prompt:

Summarize the key findings, limitations, and clinical implications of the attached Phase 2 trial results in 5 bullets.

Instant synthesis of papers that would take hours to read — with key stats highlighted.

Medical Content Generation

Prompt:

Write patient-friendly instructions for taking metformin: dosing, timing, side effects to watch for, and when to call the doctor.

Multilingual-ready content at the right health literacy level — in seconds.

Use Case: Clinical & Healthcare Applications — Safety Case Classification

Where AI actually saves time today

Writing

Draft emails, summaries, meeting notes, slide outlines.

Analysis

Explain a spreadsheet, suggest formulas, write Python for data.

Learning

Explain a topic at your level, generate quizzes, create flashcards.

Coding

Write boilerplate, debug errors, port between languages.

Translation

Translate plus adapt tone (formal/casual, region).

Brainstorming

Generate 20 names, headlines, or angles you can edit down.

Where AI actually saves time today — Draft emails, summaries, meeting notes, slide outlines.

AI sees, hears, and speaks (multimodal)

Modern assistants take more than text. Try uploading these.

📷 A photo of a receipt

"Extract items, prices, and total. Return as a table."

📈 A chart screenshot

"What story is this chart telling? What's misleading about it?"

🎙️ A voicemail / audio file

"Transcribe and summarize action items."

📄 A 40-page PDF

"Find every mention of 'liability' and quote them with page numbers."

🗺️ A whiteboard photo

"Turn this into a clean diagram description and a meeting summary."

🎬 A short video clip

"Describe what happens in this clip, scene by scene."

Use case: writing

ChatGPT as a fast, tireless first-draft assistant.

- Emails — internal, sales, polite-no's, awkward follow-ups.
- Reports — executive summaries, status updates, post-mortems.
- Content — blog posts, newsletters, social captions.
- Editing — tighten, simplify, change tone, cut by 30%.

Use case: writing — ChatGPT as a fast, tireless first-draft assistant.

Use case: brainstorming

Quantity first. The model never runs out of ideas — you pick.

- Ideas — "give me 30 angles for a post on remote work."
- Names — products, projects, characters, domain names.
- Strategies — pricing options, GTM angles, growth experiments.
- Counter-arguments — "argue against my plan, find the weakest part."

Use case: brainstorming — Quantity first. The model never runs out of ideas — you pick.

Use case: research

Speed up learning — but verify before you cite.

- Summaries — long articles, PDFs, transcripts in one screen.
- Explanations — "explain this concept like I'm a smart 12-year-old."
- Comparisons — frameworks, vendors, approaches side by side.
- Always check facts when stakes are real.

Use case: coding

Especially powerful for developers — and useful for non-developers too.

- Generate code — small functions, scripts, boilerplate.
- Debug — paste an error, paste the code, ask why.
- Translate between languages — Python to JS, SQL to pandas.
- Explain code — "walk me through what this does, line by line."

Use case: coding — Especially powerful for developers — and useful for non-developers too.

Use case: business

Where teams get the biggest ROI right now.

- Marketing — copy variants, campaign ideas, audience research.
- Customer service — draft replies, summarize tickets, suggest fixes.
- Sales — personalize outreach, prep call notes, draft proposals.
- Ops — turn meeting transcripts into action items.

Use case: business — Where teams get the biggest ROI right now.

Use case: personal

Quietly the highest-value category for most people.

- Planning — trips, meals, study schedules, workouts.
- Learning — explain a topic, quiz me, find gaps.
- Productivity — turn a brain dump into a prioritized list.
- Decisions — pros/cons, devil's advocate, frameworks to choose.

Use case: personal — Quietly the highest-value category for most people.

Custom GPTs & Projects: reusable AI helpers

Save your context once, reuse it forever. Three platforms, same idea: an assistant that already knows your stuff.

ChatGPT — Custom GPTs

Build it from chat: instructions + uploaded files + tool toggles. Share by link. Use for: "Reply in our brand voice" assistants.

Claude — Projects

Drop in style guides, glossaries, and reference docs. Every chat in the project starts with that context loaded.

Gemini — Gems

Same pattern inside Google. Pin instructions and files; reuse across Workspace.

Good first build

An "Email cleaner" GPT: instructions = your tone rules, files = 5 sample emails. Paste a draft, get it polished.

What to put in instructions

Role, audience, tone, format, hard rules ("never invent numbers"), and one example of good output.

What to put in files

Style guide, FAQ, glossary, past good answers. Keep under ~20 docs — more isn't better.

When NOT to build one

If you'd only use it twice, just paste the context. Custom GPTs pay off when you'll reuse weekly.

Custom GPTs & Projects: reusable AI helpers — Save your context once, reuse it forever. Three platforms, same idea: an assistant that already knows your stuff.

NotebookLM: turn your sources into an AI tutor

Google's tool for thinking with documents. You give it sources, it answers only from them — with citations.

What it does

Upload PDFs, slides, Google Docs, websites, YouTube links. Ask questions; every answer cites the source passage.

Why it's different

It refuses to answer outside your sources. That kills hallucinations on the topic and keeps you grounded in real text.

The party trick

Audio Overview: turns your sources into a 10-minute podcast with two AI hosts discussing them. Surprisingly good.

Best for

Studying a textbook, prepping for a meeting from 10 PDFs, onboarding from internal docs, reading a long report.

Limits

Up to 50 sources per notebook (free tier). Doesn't browse the web. Free for personal Google accounts.

Try this

Drop a 100-page report in. Ask: "What are the 3 biggest risks the authors flag, with quotes?"

Privacy note

Personal NotebookLM data isn't used to train models (per Google), but treat any cloud upload as "left the building".

NotebookLM: turn your sources into an AI tutor — Google's tool for thinking with documents. You give it sources, it answers only from them — with citations.

Five common pitfalls (and the fix)

Treating it as a search engine

Don't ask for facts you can't verify. Use Perplexity, or ask the model to cite sources and then check them.

Hallucinations

Confidently wrong answers. Ask: "What sources support this? What might be wrong?"

Privacy slips

Never paste secrets, passwords, or sensitive client data into a public chat. Use enterprise versions or local models.

One-shot expectations

First answers are rarely the best. Plan to iterate 2–4 times.

Letting it decide

Use AI to draft and explore, not to decide. You stay accountable.

Five common pitfalls (and the fix) — Treating it as a search engine

AI as a workflow.

Stop thinking about prompts. Start thinking about pipelines.

Mindset theme card — habits and posture for productive AI use.

Ask anything — and read the answer carefully

DEMO 1

TRY THIS

1. Pick a topic you actually know well.

2. Ask ChatGPT a basic question about it.

3. Read the answer slowly.

4. Mark anything that's vague, generic, or wrong.

WHAT TO NOTICE

You'll spot patterns: hedging, listy structure, safe takes, occasional fabrications. This is what to watch for in topics you don't already know.

Ask anything — and read the answer carefully — 1. Pick a topic you actually know well.

03

ChatGPT Mastery

Practical guide · Prompts · Workflows · Tools · Mistakes — use ChatGPT, don't just play with it.

Why this matters

Most people use ChatGPT at a very basic level. Small changes lead to dramatically better results.

- AI is everywhere — work, school, home, creative projects.
- Most people use it wrong: vague prompts, no context, one-shot answers.
- Small changes in how you prompt → outsized improvements in output.
- The skill gap is widening between casual users and effective users.

Why this matters — Most people use ChatGPT at a very basic level. Small changes lead to dramatically better results.

What you'll learn

Practical usage, not theory.

- Prompting — how to write instructions ChatGPT can actually follow.
- Workflows — how to chain prompts into real work.
- Tools — features beyond plain chat (files, web, code, images).
- Mistakes — what goes wrong, and how to spot it.

What you'll learn — Practical usage, not theory.

The extended prompt template

When the daily template isn't enough — five labeled sections for complex jobs.

# Identity	The role the AI should play. *"I am an SOP developer."*
# Task	The primary objective. *"Write an SOP for TFL development."*
# Subtasks	Step-by-step instructions. *"1) Define scope. 2) List inputs. 3) ..."*
# Context	Background information. *"Audience: pharma stats team. Standard: CDISC."*
# Guidance / Examples	Reference materials, formats, tone. *"Match the tone of the attached SOP_v3.pdf."*

FILLED-IN EXAMPLE

```
# Identity
I am a clinical biostatistician.

# Task
Write a 1-page synopsis of a Phase II NSCLC trial.

# Subtasks
1) State indication and population.
2) Define endpoints and comparator.
3) Summarize statistical plan in plain English.

# Context
EGFR+, comparator docetaxel, primary endpoint PFS.
Audience: lay readers (ICF reuse).

# Guidance
Tone like the attached ICF_template.pdf.
Reading level: 8th grade.
```

The extended prompt template — When the daily template isn't enough — five labeled sections for complex jobs.

Five tricks worth knowing

Steal these. They compound.

01

Be specific

Generic prompt → generic answer. Define the exact output you need: length, format, audience.

02

Show, don't just tell

Paste 2–3 examples of the output style you want. Few-shot beats explanation.

03

Ask it to think

Break the request into steps. 'Plan first, then answer.' Reasoning models do this internally.

04

Add constraints

'Under 100 words.' 'JSON only.' 'Cite quotes.' Constraints sharpen output.

05

Iterate

Refine, don't restart. 'More concise.' 'Now in a table.' 'Make it sound less corporate.'

Five tricks worth knowing — Steal these. They compound.

Bad → Better → Best

Same goal. Three prompts. Watch the output quality climb.

BAD

PROMPT

Analyze this.

WHY

Vague. No role, no audience, no format. Output will be generic.

BETTER

PROMPT

Analyze this Q1 sales data and give 3 risks and 3 opportunities.

WHY

Has a task and a count. Still missing role, audience, format.

BEST

PROMPT

Act as a CFO advisor. Analyze this Q1 sales data. Identify trends, anomalies, and 3 actionable recommendations for leadership. Output as a 1-page memo: 100-word exec summary + 3-row risk table + 3-row recommendation table.

WHY

Role + task + audience + format. Structured, scannable, executive-ready.

Bad → Better → Best — Same goal. Three prompts. Watch the output quality climb.

Structure = Task + Context + Output format

Structured LLM comparison

Same prompt. Multiple models. Score against a fixed rubric. Pick the winner per task.

1. Fix the task

One real, representative job (not toy).

2. Fix the prompt

Identical prompt across every model — no per-model tweaking.

3. Fix the rubric

Decide scoring axes before you see any output.

4. Run blind

Hide model names while scoring. Or use a colleague.

5. Score & log

Save prompt + output + score. Re-run when models update.

RUBRIC — score 1–3 per axis

Accuracy	Are facts/numbers/quotes correct? Verifiable?
Faithfulness	Did it follow your role/format/constraints?
Reasoning	Did it justify steps where needed, or just assert?
Tone fit	Right voice for the audience?
Speed / cost	Latency and tokens. Sometimes the cheaper model wins.
Refusal / safety	Refused appropriately, or over-refused?

Tip: Run the same prompt in 3 chat tabs (ChatGPT, Claude, Gemini). Paste outputs into a single doc. Score with the rubric above.

Structured LLM comparison — Same prompt. Multiple models. Score against a fixed rubric. Pick the winner per task.

Leading LLM platforms — when to use which

Pick the assistant for the job. None of them is best at everything.

Platform	Strengths	Best for	Watch out for
ChatGPT (OpenAI)	Conversational, image gen, voice, file uploads, agent mode	Daily writing, brainstorming, coding help	Confidently wrong on niche facts
Claude (Anthropic)	Long context (~200K), careful reasoning, code review	40-page contracts, research synthesis	Refuses borderline-but-fine requests
Gemini (Google)	Native image/video, ~1–2M context, Workspace, Deep Research	YouTube/PDF summarization, research	Output style varies between versions
Microsoft Copilot	Inside Word/Excel/PPT/Teams; Copilot Studio for agents	Office productivity — esp. Excel & PPT	Free web tier can't edit local files
Perplexity	Search-native, sources every claim, Deep Research mode	Quick fact-finding with citations	Source quality varies — still verify
Mistral / Le Chat	EU data residency, open-weights available (Mixtral, Small)	EU-regulated workflows, self-host	Frontier capability slightly behind
Llama (Meta)	Open-weights, run locally with Ollama or hosted on Groq	Privacy-first, cost-first, on-device	8B local models weak on long reasoning

Leading LLM platforms — when to use which — Pick the assistant for the job. None of them is best at everything.

Build workflows, not single prompts

A workflow is a small pipeline you reuse across many inputs. That's where the leverage is.

EXAMPLE — Weekly customer-feedback review

Step 1 · CLEAN

Paste raw support tickets. 'Strip signatures, headers, and PII.'

Step 2 · CLASSIFY

'Tag each ticket: bug | feature request | praise | churn risk.'

Step 3 · CLUSTER

'Group similar tickets. Name each cluster in 5 words.'

Step 4 · SUMMARIZE

'Top 5 themes this week, ranked by ticket count, with one example quote each.'

Step 5 · EXEC NOTE

'Now write a 5-bullet email to the head of product. Plain English.'

WHY WORKFLOWS BEAT PROMPTS

- One job per prompt = better output. The model performs better on small, sharp asks.
- Each step is auditable — you can spot exactly where it went wrong.
- Reusable. The pipeline runs every Monday with new input.
- Mix tools: step 1 in Excel + Copilot, step 2 in ChatGPT, step 3 in Perplexity.
- Save the prompts in a doc or as a custom GPT / Project — never type them from scratch again.

DAILY TEMPLATE — pin this

"Act as ___, do ___, for ___, in ___ format."

Build workflows, not single prompts — A workflow is a small pipeline you reuse across many inputs. That's where the leverage is.

Enterprise rollout — practical roadmap

From 'someone tried it on their laptop' to 'the company runs on it'.

01	**Cross-functional team**	IT, Legal, Security, Privacy, plus business owners. Don't let one team drive alone.
02	**Policies & guidelines**	Data security: GDPR, HIPAA, GxP, SOC 2. Define what data can/can't go to public models. Approved-tools list.
03	**Proof of concept**	One use case. Real data (or realistic). Measure accuracy, bias, time saved, ROI in a controlled environment.
04	**Training & enablement**	Prompt engineering, limits, ethics, tool-specific tips. Office hours. A shared prompt library.
05	**Deploy on enterprise tier**	ChatGPT Enterprise, Copilot for M365, Claude for Work, or self-hosted open-weight. Secure, private, auditable.
06	**Monitor & iterate**	Usage dashboards. Quality reviews. Drift checks. Sunset what isn't working. Promote what is.

Enterprise rollout — practical roadmap — From 'someone tried it on their laptop' to 'the company runs on it'.

Integrate AI into your week — not just your prompts

The compounding gain is in workflows, not in single questions.

1. Build personal workflows

- Pick three tasks you do weekly: meeting prep, weekly report, code review notes.
- Write a frozen prompt template for each — Role · Context · Format · Constraints.
- Save them in a notes app or as Custom GPTs / Projects.
- Review monthly: what's saving time, what's slop, what to drop.

2. Build team workflows

- Agree on one default tool per job (use slide 138).
- Share a team prompt library. One owner.
- Define data hygiene: what can never go into a free / public model.
- Run a 30-minute monthly demo: "what new prompt or tool earned its keep this month."

3. Build the verification habit

- Apply slide 129's 7-step check on anything that leaves your org.
- Treat AI output as a junior analyst's draft — never as a final answer.
- Keep a running log of hallucinations you catch. Pattern-spotting matters.
- When stakes are high, escalate to a human expert. Always.

Skill compounds: a year of small reps is enormous. Start one workflow this week — not three.

Integrate AI into your week — not just your prompts — The compounding gain is in workflows, not in single questions.

SECTION 04

04

Building with AI

From a chat window to something you ship: apps, RAG, fine-tuning, agents.

Section opener — Section 04.

Under the hood: input → output

An LLM is a function that predicts the next "token" (≈ a word piece) over and over.

1. Your prompt

Text becomes a sequence of tokens. "Hello world" → 2 tokens.

2. The model

Billions of parameters predict the most likely next token, given everything before it.

3. The answer

Tokens are generated one at a time and turned back into text. That's what you see streaming.

Token

A chunk of text (~3/4 of a word). Pricing and limits are measured in tokens.

Context window

How many tokens a model can "see" at once. Modern models: 200K–2M tokens.

Temperature

How random the output is. 0 = focused, 1 = creative.

Parameters

The internal numbers learned during training. Frontier models have 100B–2T+.

Under the hood: input → output — An LLM is a function that predicts the next "token" (≈ a word piece) over and over.

Tokens & context windows: why they matter

Tokens are the unit of pricing, memory, and "why did it forget?" — three things every beginner hits.

1. What's a token?

≈ 3/4 of a word in English. "ChatGPT" = 1 token. Numbers, code, and non-English text use more. Try platform.openai.com/tokenizer.

2. Pricing is per token

API plans bill per million in/out tokens. Output usually costs 3–5× input. Small models (Haiku, Flash) are ~30× cheaper than frontier.

3. Context = how much it sees

GPT-4o ~128K, Claude Sonnet ~200K, Gemini Pro ~1–2M. 1M tokens ≈ a small book. When the window fills, the chat silently forgets.

Trim before pasting

Long PDFs = big input bill + slower replies. Paste only the parts that matter, or use a long-context model.

Cap your output

"5 bullets, max 80 words" beats "detailed 5-page response". Output tokens are the expensive ones.

Start fresh on long chats

If a chat gets weird after many turns, the window is overflowing. New chat + a one-line summary fixes it.

Reasoning models think (and bill)

o1/o3 and "extended thinking" generate hidden reasoning tokens you still pay for. That's why they're slower & pricier.

Tokens & context windows: why they matter — Tokens are the unit of pricing, memory, and "why did it forget?" — three things every beginner hits.

Three ways to add AI to your work

Pick the simplest one that solves your problem.

Prompting

Just write better prompts. No code.

Best for: one-off tasks, drafts, brainstorming, learning.

Effort: minutes

RAG (retrieval)

Give the model your own documents, then ask questions over them.

Best for: company knowledge bases, customer support, internal Q&A.

Effort: days

Fine-tuning

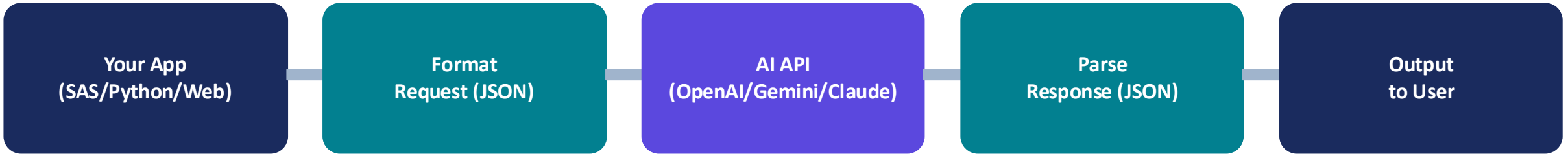
Continue training the model on your examples to bake in a style or domain.

Best for: specialized formats, tone, or tasks prompts can't capture.

Effort: weeks

Three ways to add AI to your work — Pick the simplest one that solves your problem.

Building Applications with Gen AI APIs



Python — OpenAI ChatGPT API

```python
import openai

openai.api_key = "sk-YOUR_KEY"

def ask_ai(prompt):
    response = openai.chat.completions.create(
        model="gpt-4o",
        messages=[
          {"role":"user", "content":prompt}
        ],
        temperature=0.3
    )
    return response.choices[0].message.content

print(ask_ai("Explain SDTM DM domain"))
```

SAS — Gemini API

```
%let api_key = YOUR_GEMINI_KEY;
%let url = https://generativelanguage
  .googleapis.com/v1beta/models/
  gemini-1.5-flash:generateContent;

filename prompt '/tmp/prompt.txt';
filename resp '/tmp/resp.json';

proc http method='POST'
  url="&url.?key=&api_key."
  ct='application/json'
  in=prompt out=resp;
run;

libname r JSON fileref=resp;
data result; set r.content_parts; run;
```

RAG & LangChain — AI That Knows Your Data

RAG (Retrieval-Augmented Generation): Connect any LLM to your own documents, databases, or SOPs — so it answers based on YOUR data, not just its training.

When to Use RAG

Clinical Protocol Q&A

Upload trial protocols → ask: 'What are the inclusion criteria for Study X?' — instant precise answer with citation.

SOP Compliance Check

Load all company SOPs → ask: 'Does this process violate any existing guidelines?' — with traceability.

Regulatory Submission

Upload all study documents → ask: 'Summarize safety findings for Module 2.7' in regulatory language.

Literature Review

Upload 100 papers → 'What are the main mechanisms reported for drug resistance in NSCLC?' in minutes vs weeks.

RAG: "AI that reads your documents"

Retrieval-Augmented Generation — fancy name, simple idea.

Analogy: a librarian + a writer

- You ask a question.
- The librarian (retriever) finds 3-5 relevant pages from your files.
- The writer (LLM) reads those pages and writes the answer.
- Result: answers grounded in YOUR data, with citations.

What you need to build it

Documents

PDFs, web pages, transcripts — whatever you want it to know.

A vector database

Stores chunks of your docs as searchable embeddings (Pinecone, Weaviate, pgvector, Chroma).

An LLM

OpenAI / Anthropic / open-source — anything with an API works.

A glue framework (optional)

LangChain, LlamaIndex, or just plain Python.

RAG: "AI that reads your documents" — Retrieval-Augmented Generation — fancy name, simple idea.

Retrieval-Augmented Generation (RAG)

Give the model your documents at answer time, not training time.

What it is

RAG = Retrieve relevant chunks from your own documents → feed them into the prompt → the LLM answers using that context. The model isn't retrained; it's grounded.

1. Retrieve

Search your knowledge base (PDFs, wiki, tickets) for the most relevant chunks for the user's question.

2. Augment

Stuff those chunks into the prompt as context, alongside the question.

3. Generate

The LLM answers using that context — and can cite the source.

Why use it: keeps answers current, reduces hallucinations, lets you point at fresh or private data without fine-tuning.

Retrieval-Augmented Generation (RAG) — Give the model your documents at answer time, not training time.

Example — General chatbot (no RAG)

Answers come from the model's training data only.

How it works

User asks

"What is our company's parental-leave policy?"

LLM answers from training

Uses only what it learned during pre-training. It has never seen your HR handbook.

Result

A generic answer, or a confident-sounding made-up answer (a hallucination).

Sample exchange

User: What is our parental-leave policy?

Bot: "Most US companies offer 12 weeks under FMLA…"
(Generic. Doesn't know YOUR policy.)

⚠ **No grounding in your documents → prone to hallucination and out-of-date info.**

Example — General chatbot (no RAG) — Answers come from the model's training data only.

Example — Chatbot with RAG

Same question, but the bot first looks up your actual handbook.

How it works

User asks

"What is our company's parental-leave policy?"

Retrieve from your docs

System searches HR handbook, policy PDFs, intranet → finds the 2 most relevant chunks.

Augment the prompt

Prompt = user question + retrieved chunks + "answer using only the context above."

LLM generates grounded answer

Quotes the policy, can cite the page, says "I don't know" if the docs don't cover it.

Sample exchange

User: What is our parental-leave policy?

Retrieved (HR Handbook, p. 14)

"Eligible employees receive 16 weeks paid parental leave, taken any time within 12 months of birth or adoption…"

Bot: You get 16 weeks of paid parental leave, usable within 12 months of birth or adoption. Source: HR Handbook, p. 14.

Example — Chatbot with RAG — Same question, but the bot first looks up your actual handbook.

Examples of RAG applications

Where a 'chat with your data' pattern actually pays off.

Internal knowledge assistant

Ask the wiki, HR handbook, IT runbooks. Cites the page so people trust the answer.

Customer support copilot

Drafts replies grounded in product docs and past tickets. Agents review and send.

Legal & contract Q&A

Search across contracts and policies. "Show me every clause about auto-renewal."

Clinical reference

Ground answers in current guidelines and the patient's own chart, with citations.

Finance / earnings analysis

Chat with a 10-K or a stack of analyst reports. Quotes lines back with page numbers.

Research literature search

Pull in PubMed / arXiv abstracts and let the LLM synthesize — links back to each paper.

Examples of RAG applications — Where a 'chat with your data' pattern actually pays off.

Tool use — food order taking

An LLM can do more than chat: it can call tools.

Conversation

User: "Send me a burger!"

Bot: "Ok, it's on the way!"

Looks like chat — but underneath, the LLM made a real call to the restaurant's ordering system.

Tool use — food order taking — An LLM can do more than chat: it can call tools.

Tool use — under the hood

Same conversation, but you can see the LLM calling a tool.

1 · User message

"Send me a Coffee!"

The LLM parses intent and figures out it needs to call the ordering tool, with parameters: item, user, address.

2 · Tool call (hidden)

```
ORDER(
  item: "Coffee",
  user_id: 0001,
  qty: 1,
  address: "1234 My Home"
)
```

The LLM emits a structured call. The app actually runs it against the restaurant's API.

3 · Reply to user

"Ok, it's on the way!"

Optional confirmation: "Is this correct? If yes, your card will be charged." — keep humans in the loop for actions with consequences.

Tool use — under the hood — Same conversation, but you can see the LLM calling a tool.

Tools for reasoning — without a tool

LLMs are not great at precise math.

"How much would I have after 8 years if I deposit $100 into a bank account that pays 5% interest?"

"After compounding $100 at 5% interest for 8 years, you would have $147.04."

Wrong. The correct value is $100 \times 1.05^8 \approx \$147.74$. The LLM did the arithmetic in its head — and slipped.

Tools for reasoning — without a tool — LLMs are not great at precise math.

Tools for reasoning — with a calculator tool

Let the LLM call a calculator instead of guessing.

"How much would I have after 8 years at 5% on $100?"

```
CALCULATOR(
  100 * 1.05^8
)
```

External calculator program

$$100 \times 1.05^8 = 147.74$$

"After compounding $100 at 5% interest for 8 years, you would have $147.74." ✓

Tools for reasoning — with a calculator tool — Let the LLM call a calculator instead of guessing.

Agents: AI that takes actions

Instead of just answering, an agent can plan, use tools, and try again until it finishes the job.

👀 Observe

Read the current state — chat, file, webpage, screenshot.

🧠 Think

Pick the next step. Reasoning models shine here.

🛠️ Act

Call a tool: search, run code, click a button, send an email.

🔁 Repeat

Loop until the goal is met — or it asks you for help.

What's new in 2026

Computer-use agents (Claude, OpenAI Operator) can drive a browser. "MCP" (Model Context Protocol) is becoming the standard plug for tools and data.

Agents: AI that takes actions — Instead of just answering, an agent can plan, use tools, and try again until it finishes the job.

Agents

An LLM that chooses and carries out a sequence of actions on its own.

What is an agent?

Use an LLM to choose and execute multi-step actions toward a goal — search, click, read, write, call APIs. Cutting-edge area of AI research.

Example — "Help me research BetterBurgers' top competitors"

Plan	1. Search top competitors 2. Visit each website 3. Summarize each homepage.
SEARCH	SEARCH("BetterBurgers competitors") → fastburger.com, burgerworld.com, …
VISIT	VISIT(http://fastburger.com) VISIT(http://burgerworld.com) …
SUMMARIZE	For each page: "Summarize the following text…" → bullet summary per competitor.
Deliver	Hand the user a clean comparison: who they are, what they sell, what stands out.

Agents — An LLM that chooses and carries out a sequence of actions on its own.

Fine-tuning: teaching the model your style

When fine-tuning makes sense

- You want a very specific output format every time.
- You want a consistent tone (your brand voice).
- Prompts have grown to several pages and still aren't enough.
- You have ≥100 high-quality input/output examples.
- Latency or cost matters: smaller fine-tuned models can replace bigger general ones.

When you should NOT fine-tune

- You're trying to add facts — use RAG instead.
- Your data changes often — fine-tune is static; RAG updates with documents.
- You haven't tried serious prompting yet — start cheaper.
- Your dataset is tiny or inconsistent — garbage in, garbage out.
- Privacy bans sending data out — consider open-source + local fine-tuning.

Fine-tuning: teaching the model your style — When fine-tuning makes sense

How models learn manners (RLHF)

Reinforcement Learning from Human Feedback — why ChatGPT feels helpful instead of weird.

1

Pretrain

Read the internet; learn to predict the next word. The result is fluent but unhelpful.

2

Supervised tuning

Human writers show "a good answer looks like this." The model imitates.

3

Reward model

Humans rank pairs of answers. A second model learns "answer A is better than B".

4

Reinforcement

The main model gets nudged to produce answers the reward model rates highly.

Newer twist (2025–2026): some labs use AI feedback (RLAIF/Constitutional AI) to scale this up.

How models learn manners (RLHF) — Reinforcement Learning from Human Feedback — why ChatGPT feels helpful instead of weird.

Reasoning models: thinking before answering

A 2024–2025 shift. Models can now spend extra compute thinking through hard problems.

Classic LLM

- Streams an answer immediately.
- Fast, cheap.
- Struggles on multi-step logic and math.
- Great for chat, drafts, summaries.
- Examples: GPT-4o, Claude 4, Gemini 2.5.

Reasoning model

- Pauses to think ("chain-of-thought") before replying.
- Slower, more expensive.
- Much better at math, code, planning, hard logic.
- Best for thorny problems, not quick chat.
- Examples: o3 / o4-mini, Claude 4.5 "thinking", Gemini 2.5 Pro reasoning.

Reasoning models: thinking before answering — A 2024–2025 shift. Models can now spend extra compute thinking through hard problems.

Closed-source vs. open-source models

Closed-source (API)

- Use via API: OpenAI, Anthropic, Google.
- Frontier capability, low setup.
- Pay per token.
- Your data leaves your machine — check the privacy terms.
- Risk of vendor lock-in and pricing changes.

Open-source (download)

- Llama 4.x / 4, Qwen 2.5/3, Mistral, DeepSeek, Gemma.
- Run locally (Ollama, LM Studio) or self-host.
- Full data privacy and control.
- Lags frontier closed models by 3–9 months on quality.
- You manage updates, hardware, and security.

Closed-source vs. open-source models — • Use via API: OpenAI, Anthropic, Google. • Frontier capability, low setup. • Pay per token. • Your data leaves your machine — check the privacy terms. • Risk o...

Hallucinations — when AI sounds confident but is wrong

The model invents details that look entirely plausible. Fluency is the trap.

What hallucinations look like

- Made-up citations, URLs, and case law
- Plausible-sounding but wrong numbers
- Quotes "from" people who never said them
- Fake API methods or function names
- Wrong dates and historical details
- Confidently wrong math without a tool
- Smooth, well-written, completely false

Why it happens

- An LLM is a next-token predictor, not a fact database.
- Training data is finite and dated — if it doesn't know, it still has to predict something.
- The training objective rewards plausibility, not truth.
- Long context fills up — the model forgets and improvises.
- Reasoning chains compound errors at each step.
- Most-suspicious zones: names, numbers, dates, quotes, code APIs.

Hallucinations — when AI sounds confident but is wrong — The model invents details that look entirely plausible. Fluency is the trap.

Verify before you trust

A 60-second checklist. Run it on every output that could embarrass you, mislead a customer, or land in a legal document.

01	**Plausibility scan**	Read once. Does anything feel off? Trust that instinct.
02	**Spot-check facts**	Pick one number, one name, one date. Verify each against the original source.
03	**Click cited links**	Does the page exist? Does it actually say what was quoted? Don't skip this.
04	**Cross-check elsewhere**	Confirm in a second source — Wikipedia, primary doc, or a colleague who knows.
05	**Use grounded tools**	For high-stakes work, prefer NotebookLM, RAG, or a search-grounded model (Perplexity, Gemini).
06	**Ask the model to double-check**	Paste it back: 'Are you sure? List anything you're uncertain about.' Often it self-corrects.
07	**Human in the loop for high stakes**	Legal, medical, financial, security. A human reviews before it ships.

Verify before you trust — A 60-second checklist. Run it on every output that could embarrass you, mislead a customer, or land in a legal document.

Privacy — what to never paste

Treat a public chat like a public forum. Once it's in, you can't get it out.

DO NOT PASTE

- Customer names, emails, addresses
- Salaries, performance reviews, HR docs
- Health records or anything HIPAA-adjacent
- Source code under NDA
- Contracts and legal docs (unless cleared)
- API keys, passwords, internal URLs
- Anything you wouldn't email a competitor

SAFER PATHS

- Enterprise tier (ChatGPT Enterprise, Copilot for M365, Claude for Work) — data not used to train.
- Self-hosted open-weight models (Llama, Mistral) — data never leaves your network.
- Redact before pasting: replace names, IDs, and amounts with placeholders.
- Use Projects/GPTs with explicit data-use settings turned off.
- Check vendor terms: data retention, training opt-out, regional residency (EU).

Privacy — what to never paste — Treat a public chat like a public forum. Once it's in, you can't get it out.

Limits, bias, and ethics

What today's AI can't do well — and the risks of pretending otherwise.

Knowledge cutoff

Training data has an end date. Without a search tool, it can't know what happened after.

Confident errors

It says 'I'm not sure' less often than it should. Calibration is improving but not solved.

Recent events

Without web access, anything in the last 6–12 months is unreliable.

Decisions you'd own

Don't use AI to decide hiring, lending, medical, or legal outcomes without a human reviewer and an audit trail.

Bias in, bias out

Models reflect their training data. Test outputs across groups before deploying anything user-facing.

Math without tools

Plain chat can't reliably do precise arithmetic. Use code interpreter, calculator tool, or a reasoning model.

Long memory

It does not remember past chats unless you turn on memory or paste context. Don't assume continuity.

Copyright & attribution

AI-generated text and images sit in a gray legal zone. Don't claim authorship for important external work without disclosure.

How do you know it's working?

"It looks good" is not an evaluation.

Golden examples

Hand-pick 30–100 inputs and the answers you'd accept. Re-run after every change.

Side-by-side

Compare two prompt versions on the same inputs. Pick the better one — or have an LLM judge it.

Production logs

Sample 1% of real traffic each week and review by hand. The most underrated technique.

Task-specific metrics

Faithfulness for RAG, exact-match for extraction, ROUGE for summaries, code-tests pass rate for code.

How do you know it's working? — "It looks good" is not an evaluation.

SECTION 05

05

Risks & next steps

Limits, ethics, and how to keep going after this class.

Section opener — Section 05: Risks and responsible use.

Risks, Concerns & Responsible AI Use

Data Privacy — HIGH

Never input PHI, PII, or confidential trial data into public AI tools. Use enterprise versions with data privacy guarantees (ChatGPT Enterprise, Claude for Enterprise).

Hallucination — HIGH

LLMs confidently generate incorrect information. Always verify regulatory, medical, or legal AI outputs with subject matter experts before use.

Bias — MED

Training data reflects historical biases. Review AI outputs for demographic, clinical, or ethical bias — especially in patient-facing content.

Regulatory Gap — MED

AI use in clinical submissions is evolving. Follow FDA AI/ML guidance, GDPR, HIPAA, GxP requirements. Document AI involvement in regulated processes.

IP & Copyright — MED

AI outputs may contain copyrighted content. Don't input proprietary competitor data. Review enterprise data use agreements carefully.

Over-reliance — LOW

AI is an assistant, not a decision-maker. Maintain human oversight, especially for safety-critical clinical decisions. Keep humans in the loop.

Risks, Concerns & Responsible AI Use — Never input PHI, PII, or confidential trial data into public AI tools. Use enterprise versions with data privacy guarantees (ChatGPT Enterprise, Claude for Ente...

Honest limitations

Hallucinations

Models sometimes generate confident, well-formed nonsense. Always verify important facts.

Bias

Trained on human text — they reflect human biases. Spot-check outcomes for protected groups.

Knowledge cutoff

Without a search tool, the model's knowledge ends at its training date. Ask: "When is your knowledge from?"

Context limits

Long inputs get truncated or quality drops. Summarize, chunk, or use longer-context models for long docs.

Cost & latency

Frontier models are expensive at scale. Smaller models or fine-tunes often suffice.

Reproducibility

Same prompt can produce different answers. Use temperature=0 if you need consistency.

Honest limitations — Models sometimes generate confident, well-formed nonsense. Always verify important facts.

Hallucinations: when AI sounds confident but is wrong

What they look like

- Made-up citations, URLs, and case law
- Plausible-sounding but wrong numbers
- Quotes "from" people who never said them
- Fake API methods or function names
- Wrong dates and historical details
- Confidently wrong math without a tool
- Smooth, well-written, completely false

How to catch them

- Ask for sources — then click them
- Use a tool that grounds in your docs (NotebookLM, RAG)
- Cross-check facts in 1 other place
- Be most suspicious of names, numbers, dates, quotes
- Watch for too-perfect, too-specific answers
- Use reasoning models for math; let them call a calculator
- If stakes are high: a human checks before it ships

Hallucinations: when AI sounds confident but is wrong — • Made-up citations, URLs, and case law • Plausible-sounding but wrong numbers • Quotes "from" people who never said them • Fake API methods or...

Hallucinations

The model invents details that look entirely plausible.

MISTAKE

AI makes things up — fluently.

WHY IT BURNS YOU

Fabricated quotes, fake citations, invented APIs, wrong dates. The fluency is the trap. Cross-check anything that could embarrass you, mislead a customer, or land in a legal document.

Hallucinations — The model invents details that look entirely plausible.

The biggest mistake

If you remember nothing else from this talk, remember this.

MISTAKE

✕ Trusting AI blindly.

WHY IT BURNS YOU

ChatGPT sounds confident even when it's wrong. Confidence ≠ accuracy. Always verify anything that has real consequences — facts, numbers, names, citations, code that runs in production.

The biggest mistake — If you remember nothing else from this talk, remember this.

Lack of context

The single biggest reason your output is mediocre.

MISTAKE

Vague prompts → vague answers.

WHY IT BURNS YOU

If you didn't tell it the audience, the goal, the tone, and the constraints, it's filling those blanks itself — and it picks the most generic option every time.

Not iterating

Stopping after the first answer leaves most of the value on the table.

MISTAKE

First draft is not the final draft.

WHY IT BURNS YOU

The first answer is a baseline, not a result. Two follow-up prompts ("critique it," "rewrite it shorter") usually take you from 60% to 90% quality.

Not iterating — Stopping after the first answer leaves most of the value on the table.

Overcomplicating

More words ≠ better prompts. Sometimes it's worse.

MISTAKE

Walls of instructions confuse the model.

WHY IT BURNS YOU

If your prompt is 400 words of overlapping rules, the model will quietly ignore half of them. Be specific, not exhaustive. One job per prompt is usually enough.

Wrong expectations

It's a tool, not a colleague. Set expectations accordingly.

MISTAKE

AI ≠ human intelligence.

WHY IT BURNS YOU

It doesn't understand. It doesn't have judgment. It doesn't know your business. Use it for speed and breadth — keep humans in the loop for judgment, accuracy, and accountability.

Using AI responsibly

Privacy

Don't paste personal data, secrets, or client info into public chats. Use enterprise tiers, on-prem, or data-redacted prompts.

Attribution & honesty

Disclose when AI helped produce something. Don't pass off generated work as fully original.

Fairness

Test outputs on diverse inputs. AI in hiring, lending, or healthcare needs explicit fairness checks.

Copyright & data rights

Generated content may resemble copyrighted material. Check licensing before commercial use of images/code.

Environment

Training and inference use real energy. Default to smaller models when they suffice.

Human oversight

AI suggests, humans decide — especially for medical, legal, financial, or safety calls.

Using AI responsibly — Don't paste personal data, secrets, or client info into public chats. Use enterprise tiers, on-prem, or data-redacted prompts.

Privacy: what to never paste

Treat the chat like a public forum

- Customer names, emails, addresses
- Salaries, performance reviews, HR docs
- Health records or anything HIPAA-adjacent
- Source code under NDA
- Contracts and legal docs (unless cleared)
- API keys, passwords, internal URLs
- Anything you wouldn't email a competitor

Safer paths

- Toggle off "use my chats for training" in settings
- Use enterprise/Team tiers — they don't train on your data by default
- Use the API with a no-retention agreement
- Run an open model (Llama, Mistral) locally for sensitive work
- Redact names → "Customer A" before pasting
- Check your employer's AI policy first
- When in doubt: don't paste it

Privacy: what to never paste — Treat the chat like a public forum

Verify before you trust

A 60-second check that catches most errors

- Read it once for plausibility — does anything feel off?
- Spot-check 1 number, 1 name, 1 date against the source
- Click any cited link — does the page actually say that?
- Ask the model: "What's the weakest part of this answer?"
- Re-prompt with: "List anything you weren't sure about."
- For code: actually run it on a small input
- For decisions: a human signs off, not the bot

When to verify hardest

- Anything legal, medical, or financial
- Numbers in a report, deck, or email to leadership
- Names, titles, and quotes attributed to real people
- Citations, case law, and URLs (these hallucinate often)
- Code that touches production or money
- Translations into a language you don't read
- Anything you'll be on the hook for

Verify before you trust — A 60-second check that catches most errors

Lifecycle of a real AI project

Most failures happen because step 1 was skipped.

1

Scope

Pick a narrow problem. Define what "good" looks like with examples.

2

Prototype

Cheap end-to-end with prompting first. Test on real (but small) data.

3

Evaluate

Build an eval set of 30–100 cases. Measure quality, not just vibes.

4

Improve

Better prompts → RAG → fine-tune → smaller model. Stop when good enough.

5

Deploy

Add guardrails, logging, fallbacks. Roll out to a small group first.

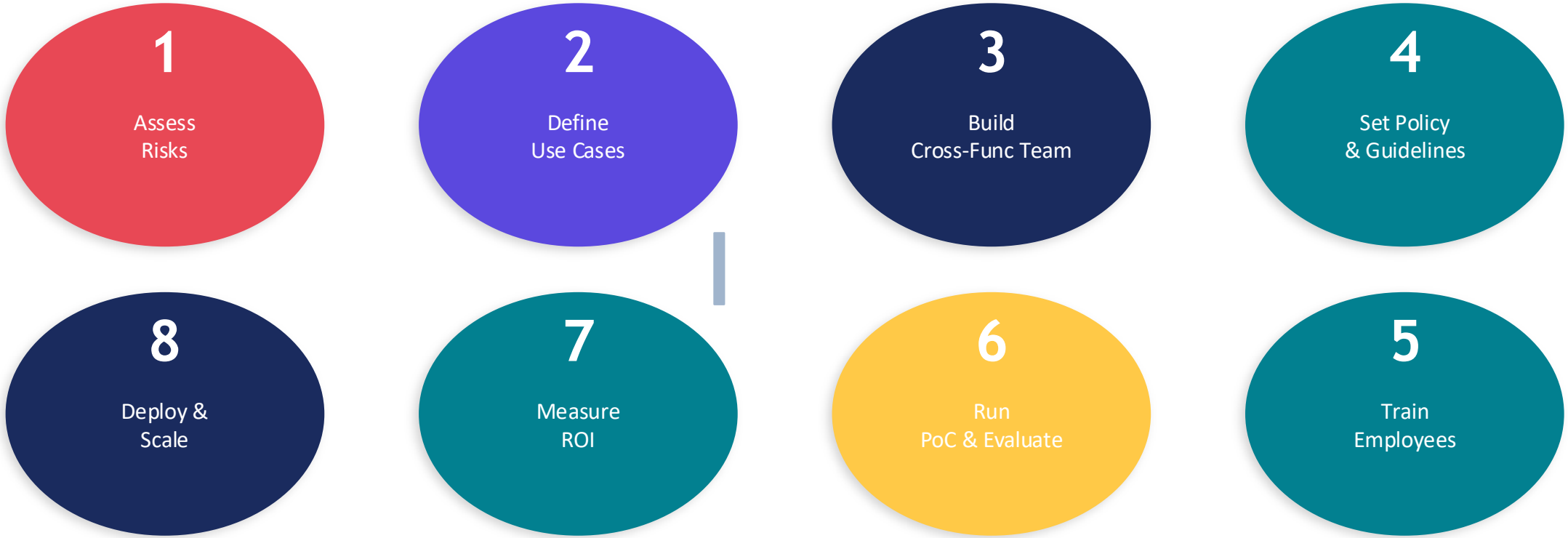
6

Monitor

Track quality, cost, and user feedback. Models drift; users change.

Lifecycle of a real AI project — Most failures happen because step 1 was skipped.

Enterprise AI Implementation Roadmap



Key Stakeholders

IT & Security | Legal & Compliance | Data Privacy | Biometrics/Statistics | QA | Regulatory Affairs | Leadership

Enterprise AI Implementation Roadmap — Build Cross-Func Team

WHERE THIS GOES NEXT

AI everywhere.

Embedded in your inbox, your editor, your phone, your tools.
The skill you build today is the floor — not the ceiling.

Start now. The compounding starts the day you do.

06

Going further

From Predictive to Generative

The AI Evolution: From Predictive to Trustworthy GenAI

Classical Predictive	Early GenAI (Isolated)	Enterprise-Grade GenAI
Approach		
Deterministic scoring on structured CAS data	Standalone LLM API calls with no data context	RAG-grounded LLM with live CAS table retrieval
Hallucination Risk		
N/A - deterministic output	~18% error rate on numeric claims	<2% with 3-layer validation framework
Accuracy		
>99% on structured analytical tasks	Plausible but statistically unverified	>99% - PROC SQL cross-checks every claim
Governance		
Full SAS audit trails and RBAC	None - black-box API responses	Full lineage: CAS to chunk to LLM to audit log

The AI Evolution: From predictive analytics to trustworthy generative AI.

Why Isolated LLMs Fail - and What the Numbers Prove

The problem is statistical illiteracy. The solution is SAS Viya.

⚠ The Problem with Isolated LLMs

Statistically Illiterate
LLMs cannot run PROC SQL, PROC MEANS, or verify numeric claims against your CAS data.

~18% Error Rate
Unvalidated GenAI outputs contain numerical errors at rates unacceptable for enterprise reporting.

No Governance Trail
Isolated API calls produce zero audit log, no RBAC, no lineage from data to answer.

Stale Knowledge
Models trained on past data - not your live CAS tables, not your latest SAS metadata.

✓ Proven Results with SAS Viya

85-99%
Efficiency Gains

>99%
Numerical Accuracy

<2%
Hallucination Rate

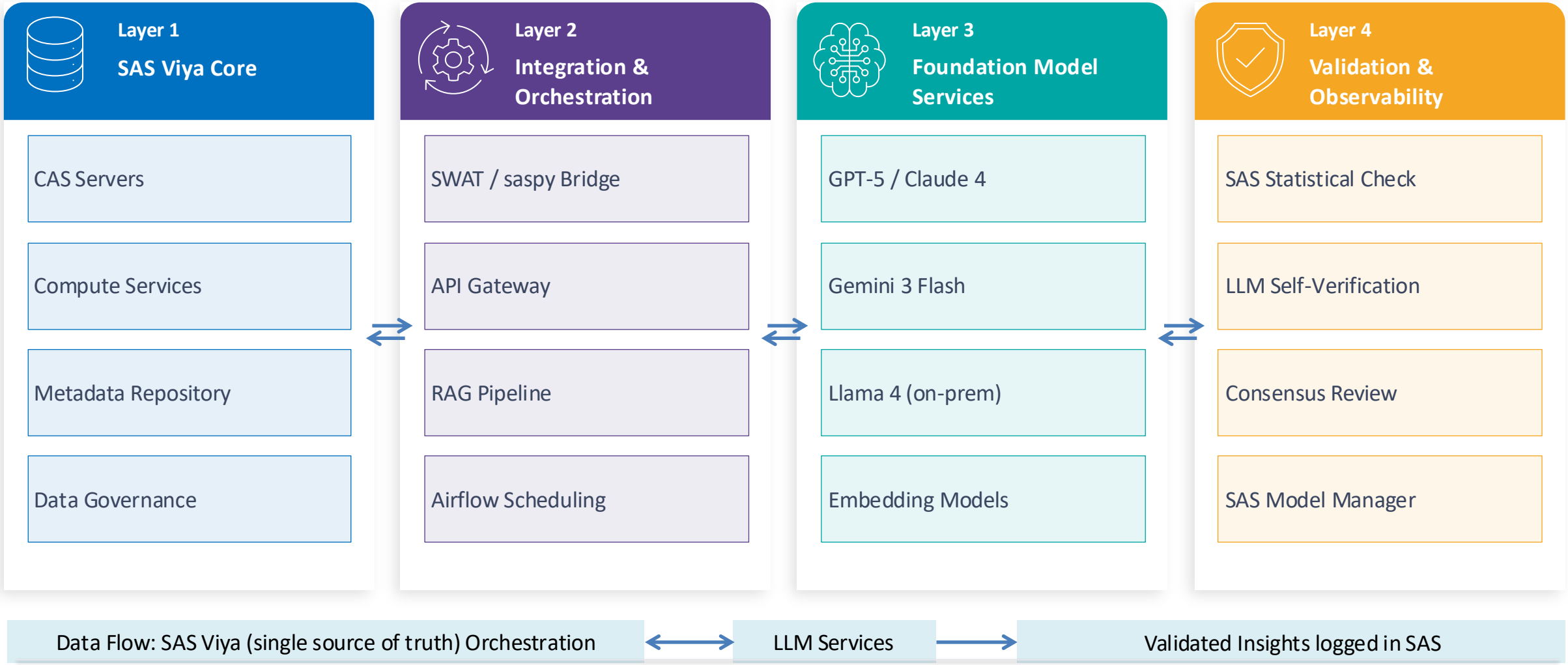
71%
Cost Reduction

These results are only possible when SAS Viya is the statistical ground truth - and GenAI is the narrative layer.

SAS Viya as statistical ground truth — generative AI handles the narrative; SAS supplies the proof.

SAS Viya GenAI Integration Architecture

Four coordinated layers from data to validated insight



Architecture diagram — Four coordinated layers from data to validated insight.

Enterprise Case Studies: Measured Impact

Real-world SAS Viya GenAI deployments across three industries

Financial Services

Challenge:

25+ analyst hrs/week writing compliance report narratives

Solution:

RAG + LangGraph agent queries CAS → generates → validates → submits

Time Saved

Before: 25 hrs → After: 3 hrs (88%)

Accuracy

Before: 94% → After: >99.5% (+5.5pp)

Healthcare

Challenge:

2–3 day delays for medical reviewers needing ad-hoc CAS queries

Solution:

NL query interface: medical English → SAS SQL → CAS → plain-English summary

Query Speed

Before: 2–3 days → After: <2 min (99.9%)

Self-Service

Before: 0% → After: 78% (+78pp)

Retail

Challenge:

Weekly batch insights too late for same-day merchandising decisions

Solution:

Streaming ESP pipeline + on-prem Llama 4 — zero PII egress

Insight Latency

Before: 48 hrs → After: <15 min (99.5%)

Campaign Lift

Before: Baseline → After: +41% (+41%)

Consistent pattern: 85–99% efficiency gains · SAS statistical cross-validation ensures accuracy in all deployments

Enterprise case studies — measured impact across deployments.

Limitations & Future Directions

Honest assessment — and the research agenda ahead

⚠ Current Limitations

Multimodal Validation Gaps

Image/chart outputs from LLMs cannot yet be cross-checked against SAS PROC results — human review remains mandatory for visual outputs.

Agentic Orchestration Risks

LangGraph agents may loop or exhaust context in long analytical sessions. Robust max-iteration limits and fallback to HITL are essential but add latency.

On-Prem LLM Quality Trade-offs

Llama 4-70B reduces cost and PII risk but lags GPT-5/Claude 4 on complex reasoning tasks. Benchmark on your domain before substituting.

RAG Freshness Latency

Incremental vector indexing pipelines introduce 5–15 min lag for hot CAS tables. Near-real-time workloads require additional streaming architecture.

Future Research Directions

Agentic SAS Workflow Automation

Full LangGraph agent suites that autonomously plan, code, execute PROC statements, and deliver validated reports — extending SAS programmers' reach.

Multimodal SAS VA Integration

LLMs that interpret VA charts, compare visual trends against CAS tables, and flag anomalies — enabling natural language QA over dashboards.

Federated RAG for Regulatory Data

Cross-institution RAG that retrieves from distributed SAS environments without centralizing sensitive data, meeting HIPAA/GDPR at query time.

AutoML + GenAI Co-pilot

LLM that interprets SAS AutoML results, recommends feature engineering, and generates interpretable model narratives for model risk review.

Current limitations — caveats and known failure modes.

From Predictive to Generative AI

Isolated LLM Calls vs. Framework-Augmented Responses
Clinical Adverse Event Analysis — Live Demo

Study: CDISCPILOT01 | Drug: Xanomeline (54 mg / 81 mg)
Population: 254 subjects with mild-to-moderate Alzheimer's disease
Data: 32 CDISC SDTM/ADaM domains + clinical documents
Model: x-ai/grok-code-fast-1

FRAMEWORK LAYERS

- Layer 1 — SAS Viya Core (Data + Governance)
- Layer 2 — Integration & Orchestration (RAG Pipeline)
- Layer 3 — Foundation Model Services (LLM)
- Layer 4 — Validation & Observability (Statistical cross-check)

Section divider.

Lida Gharibvand | Loma Linda University | SAS Innovate 2026

The Query

HARD

Safety / Adverse Events | ID: ae_019

What was the mean duration (in days) of adverse events by severity (AESEV)? Use ADAE.ASTDY and AENDY.

Ground truth answer: `{'MILD': 24.8, 'MODERATE': 23.9, 'SEVERE': 19.2}`

Formula: `ADAE['AE_DUR'] = ADAE['AENDY'] - ADAE['ASTDY'] + 1 | grouped by AESEV`

Difficulty: Hard | Dataset: ADAE | Variables: ASTDY, AENDY, AESEV | Computed directly from clinical datasets

Category: Safety / Adverse Events | Difficulty: Hard

Worked example — the user query (safety / adverse events).

Isolated LLM vs. Framework-Augmented

Isolated LLM Call

Data summary dump · No tools · No RAG · No validation

The summary does not contain aggregated statistics such as mean duration by AESEV. It provides column types, unique values and sample rows, but not the calculated means of (AENDY - ASTDY) or ADURN grouped by AESEV.

- Cannot compute — no query execution
- No verification layer — may hallucinate
- No audit trail or data lineage
- 41,596 prompt tokens consumed

5.46s	41,596	950
Time	Prompt tokens	Completion

VS

Framework-Augmented

RAG + Domain Graph + SAS Execution + Validation

The mean duration of adverse events by severity:
MILD: 24.77 days (N=770)
MODERATE: 23.88 days (N=378)
SEVERE: 19.16 days (N=43)

- SAS PROC-validated from actual datasets
- Domain graph for structural navigation
- Full audit trail — 1 tool call, 2 rounds
- 91% fewer prompt tokens (3,459 vs 41,596)

5.61s	3,459	856
Time	Prompt tokens	Completion

Worked example — isolated LLM versus framework-augmented results.

Accuracy vs Ground Truth

MILD (N=770)

Framework answer

24.77 days

Ground truth **24.8**

Delta: 0.03 days (0.1%) — within 2% threshold

MODERATE (N=378)

Framework answer

23.88 days

Ground truth **23.9**

Delta: 0.02 days (0.1%) — within 2% threshold

SEVERE (N=43)

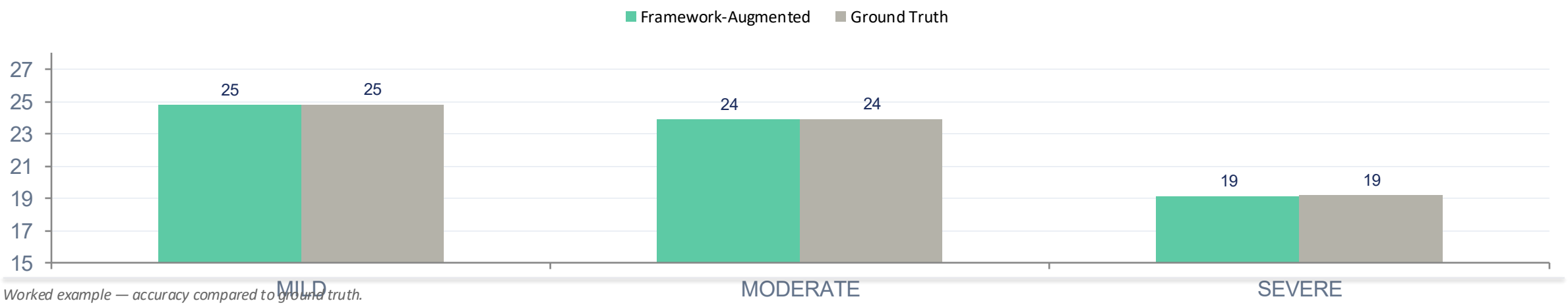
Framework answer

19.16 days

Ground truth **19.2**

Delta: 0.04 days (0.2%) — within 2% threshold

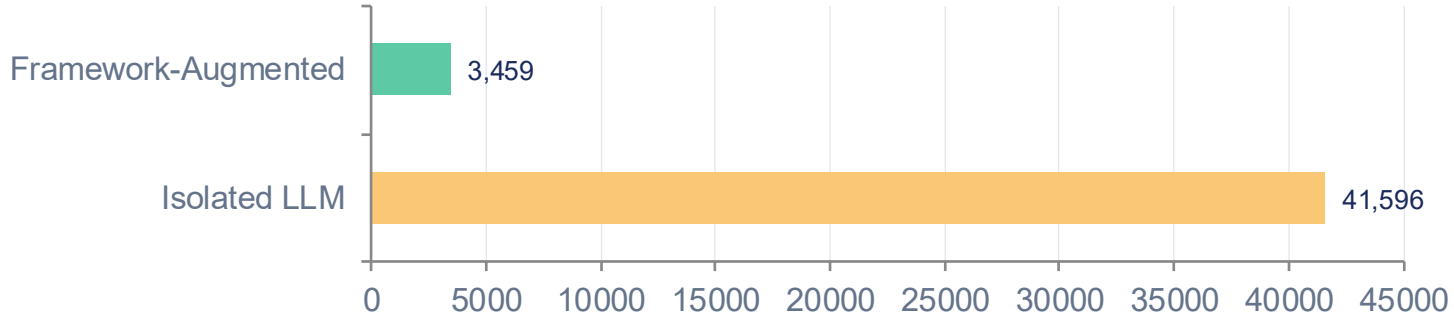
Mean duration by severity (days)



Worked example — accuracy compared to ground truth.

Token Efficiency & Audit Trail

Prompt token usage — 91% reduction with framework



91%

Fewer prompt tokens
(3,459 vs 41,596)

Audit trail — 3-step provenance chain

1

RAG Retrieval — domain graph navigation

```
Sources resolved: ADAE.ASTDY, ADAE.AENDY, ADAE.AESEV, ADAE.ADURN
```

2

SAS Execution — PROC MEANS via CAS

```
ADAE['AE_DUR'] = ADAE['AENDY'] - ADAE['ASTDY'] + 1;  grouped by AESEV
```

3

Statistical Validation — cross-check vs ground truth

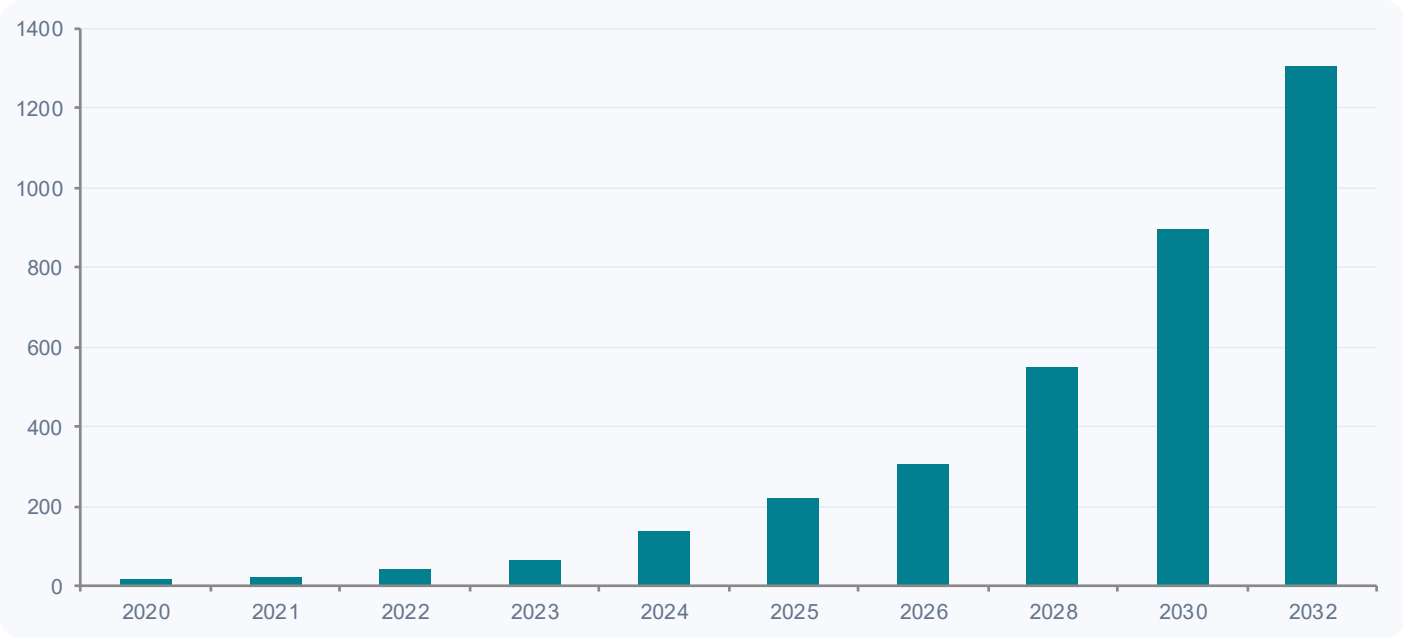
```
Max delta: 0.04 days (0.2%) — within tolerance threshold (<2%)
```

Worked example — token efficiency and audit trail.

Sources verified: ADAE.ASTDY, ADAE.AENDY, ADAE.AESEV, ADAE.ADURN

The Future of Gen AI — Where Is This Going?

Bloomberg Intelligence Gen AI Market Forecast ($B)



Key Trends to Watch

Agentic AI

AI systems that autonomously plan and execute multi-step tasks (coding, research, scheduling).

Multimodal AI

Models that process text, images, audio, video, and code simultaneously.

Smaller Models

Efficient local models (Llama, Phi, Mistral) running on laptops — no cloud required.

Gen AI is the new electricity — not a replacement for expertise, but an amplifier of it.

The question is no longer IF your organization will use AI — but HOW WELL.

The future of generative AI — where this is going next.

The New Standard for Trustworthy AI

Honest assessment — and the research agenda ahead

[Math]

Math is the Anchor

GenAI provides the narrative - SAS Viya provides the proof. Never sacrifice statistical rigor for conversational ease. Every number must trace back to a CAS table.

[Lock]

Governance is Non-Negotiable

Moving from Experimental AI to Enterprise AI requires the audit trails and RBAC already built into your SAS infrastructure. Compliance is not a feature - it is the foundation.

[Chart]

Efficiency Without Compromise

71% cost reduction and >99% accuracy are not mutually exclusive. They are the result of engineered deployment: semantic caching, model tiering, and on-prem open-source LLMs.

85–99%
Efficiency Gains

>99%
Numerical Accuracy

<2%
Hallucination Rate

71%
Cost Reduction

"The goal isn't just to build AI that talks-it's to build AI you can trust with your business."

Three pillars — Math is the anchor, governance is non-negotiable, validation is the differentiator.

Homework: 3 small experiments this week

1

Replace one task

Pick one weekly task (a recurring email, a meeting summary, a checklist). Hand it to AI for a week. Track time saved and where it slipped.

2

Build a tiny tutor

Tell ChatGPT/Claude: "You are my [topic] tutor. Quiz me daily for 5 minutes, get harder when I'm right, easier when I'm wrong." Run it for 5 days.

3

Read a 50-page PDF in 10 minutes

Upload any PDF, ask: "Outline this in 8 bullets, then list the 3 weakest claims and what evidence is missing." Verify one of the claims by hand.

Homework: three small experiments to try this week.

Where to keep learning

Free first. Pay only when you know what you need.

Courses

DeepLearning.AI "AI for Everyone" (Andrew Ng); "Generative AI for Everyone"; Anthropic Academy; Google AI Essentials.

Free / cheap, 4–10 hours each.

Hands-on

Just use ChatGPT, Claude, or Gemini for everything for two weeks. Notice where it helps and where it fails.

0$. Most useful thing on this slide.

Code path

Hugging Face course; LangChain / LlamaIndex docs; build a tiny RAG over your own PDFs.

Requires basic Python.

Stay current

Newsletters: Import AI, The Batch, Latent Space. Skim once a week, don't drown.

15 min/week.

Communities

r/LocalLLaMA, r/ChatGPT, Hugging Face forums, local AI meetups.

Best for problem-solving.

Papers (later)

When you're ready: "Attention Is All You Need," InstructGPT, RAG, RLHF surveys.

Optional. Don't start here.

Where to keep learning — paths, communities, and references.

SIGNATURE IDEA

AI is powerful — but guided.

The best users aren't the ones who know the most prompts. They're the ones who keep steering.

Signature idea — single takeaway slide.

Learn by doing.

Reading about ChatGPT teaches you nothing. Using it for one real task today teaches you everything.

Mindset takeaway slide.

TAKEAWAY

AI is bigger than ChatGPT. LLMs are just one piece.

SEE

Vision, speech, graphs — AI that perceives.

DECIDE

Recommenders, anomaly, RL — AI that chooses.

ACT

Robotics, agents, control — AI that does.

Next time someone says "AI," ask: which kind?

Final takeaway slide.

Workshop

I have created this website using multiple Ai packages:

https://claude.ai/public/artifacts/ba3c2135-6fca-4978-b6f2-017b24a1f64a

For homework, try to do the something!

Workshop closing slide.

Questions, demos, follow-ups — let's talk.

Thank you!

Lida Gharibvand

Professor, Loma Linda University

lgharibvand@gmail.com

LinkedIn. www.linkedin.com/in/lidagharibvand

Thank-you / Q&A slide.