MGMT 803, MBA@RICE, 2026

Thursday, September 24, Morning

Kerry Back
J. Howard Creekmore Professor of Finance

About me

Position

J. Howard Creekmore Professor of Finance and Professor of Economics

At Rice since 2009. Before that, Northwestern, Indiana, Washington University in St. Louis, and Texas A&M

Teaching

- Gen AI and Quant Investments (MBA)
- Applied Finance (MBA)
- Agentic AI, Data, and Decisions (MBA)
- Asset Pricing (PhD)
- Data-Driven Finance (MDS)
- AI and Data Management (ENGI)
- Making Data-Driven Decisions with Agentic AI (Exec Ed)

Four days

	WHEN	TOPICS
1	Thursday morning	How LLMs work · models · computer use · how agents work · doing work with AI · prompting
2	Thursday afternoon	AGENTS.md, SKILL.md · connectors and plugins · harnesses
3	Friday morning	Working with data
4	Saturday morning	Creating docs · creating apps
5	Saturday afternoon	Retrieval augmented generation · open source models
6	Sunday morning	Creating agents · managing security risks

> Every session has hands-on blocks in your own container. Bring a laptop and stay on wifi for four days.

This morning

	TOPIC	WHAT IT COVERS
1	How language models work	Tokens, vectors, training, and why they fail
2	Models and performance	Who makes them, what they cost, how they compare
3	Computer use	Models that click and type, in a browser or on the desktop
4	How agents work	Chatbots, tools, and what is written into the harness
5	Doing work with AI	Where to work, and what is in your lab container
6	Prompting tips	Planning, code for numbers, checking what comes back

WHEN	WHAT
2017	Google researchers publish the transformer (“Attention Is All You Need”)
2020	GPT-3 (OpenAI) shows abilities that appear with scale
2022–2023	Online chatbots. You typed, it typed back
2023–2024	Code execution and web search arrive
2025	The serious versions move to the command line, reading and writing files on your own machine. DeepSeek R1 appears
2025–2026	That command-line capability is wrapped back into an app
2026	Agents that carry out extended work on your files without you living in a terminal

A brief history

Andrej Karpathy: OpenAI co-founder, former Tesla AI chief, now at Anthropic

October 2025

“They just don’t work. They don’t have enough intelligence, they’re not multimodal enough, they can’t do computer use and all this stuff. They don’t have continual learning. You can’t just tell them something and they’ll remember it.”

“I feel like the industry is making too big of a jump and is trying to pretend like this is amazing, and it’s not. It’s slop.”

December 2025

“I really am mostly programming in English now, a bit sheepishly telling the LLM what code to write... in words. Biggest change to my basic coding workflow in ~2 decades.”

February 2026

“It is hard to communicate how much programming has changed due to AI in the last 2 months: not gradually and over time in the ‘progress as usual’ way, but specifically this last December. Coding agents basically didn’t work before December.”

“I don’t think I’ve typed like a line of code probably since December.”

1 · HOW LANGUAGE MODELS WORK

Enough of the mechanism to explain what these models are good at and where they fail. Every failure you meet over the four days has a cause in this block.

Tokenization

Text is cut into pieces from a fixed vocabulary before the model sees any of it.

```
"The cat sat on the mat"  →  ["The", " cat", " sat", " on", " the", " mat"]

"Unbelievable"            →  ["Un", "believ", "able"]
```

Common words stay whole; rare and long words break into pieces. A typical vocabulary runs to about 100,000 tokens.

> Numbers break on the same rules, which have nothing to do with arithmetic. `13,963,378.65` reaches the model as a handful of unrelated pieces.

The whole task

Given the tokens so far, predict the token that comes next. That is the entire operation.

Everything a model appears to do — answering, summarizing, writing code, refusing — is that one prediction, repeated.

> The input can currently run to about a million tokens, roughly two-thirds of the Harry Potter saga.

Vectors

The model does not work with tokens. It works with numbers.

Each token is a list

“king” becomes 4,096 numbers. “queen” becomes 4,096 numbers that are close to them.

Tokens used in similar contexts end up near each other.

The space is mostly empty

There are far more possible vectors than there are tokens, so most points in the space name nothing.

The prediction can land between tokens.

A neural network is a function

THE IDEA

$y = 4x$ is a function. The 4 is a parameter.

Change it to 3 and you have the same structure with different behavior.

THE SCALE

Same idea with hundreds of billions of parameters, arranged as a transformer — the architecture published by Google researchers in 2017.

Training

What is learned

The vector for every token, and every parameter in the network.

Both are adjusted together to reduce prediction error on the training text.

What it takes

Trillions of words. Thousands of GPUs for weeks or months. Tens to hundreds of millions of dollars for one run.

Nothing is looked up or stored as fact. The arrangement captures how words are used, which is why it recalls common things reliably and rare things unreliably.

The pipeline

1. Tokenize the input.
2. Look up the vector for each token.
3. Feed the vectors into the network.
4. The network produces one output vector.
5. Find which token vectors are closest to it.
6. Emit a token, append it to the input, and start again at step 1.

Temperature

The network does not output one token. It outputs a probability for every token in the vocabulary. Temperature controls how that distribution is sampled.

SETTING	BEHAVIOR
Near 0	The highest-probability token every time. Repeatable.
Near 1	Sampled from the distribution. Different each run.
Higher	Low-probability tokens get picked. Text degrades.

> In a chatbot, including Claude Desktop, temperature is fixed near 1 and you cannot change it. Through the API you can.

From prediction to conversation

A model trained only to predict the next token continues your text. It does not answer a question.

1. Role tokens. The text is wrapped in markers: `◁user▷` your question `◁assistant▷`.
2. Instruction tuning. The model is further trained on examples where what follows `◁assistant▷` is a useful reply.
3. Reinforcement learning from human feedback. Raters score replies, and the model is trained toward what raters preferred.

The mechanism did not change. What changed is which continuation is the likely one.

Reasoning models

The same trick applied again: train the model to write out its working before it answers.

How it was trained

Reinforcement learning on problems whose answers can be checked — mathematics, code, puzzles — rewarding the chains of tokens that arrived at a correct answer.

Why it helps

The model holds no state between tokens. The intermediate tokens are the only place a partial result can live, so writing them out is the computation.

> The thinking you see may be a summary of a much longer chain, and you are charged for the whole chain as output tokens.

Demos

TAB	WHAT IT SHOWS	MODEL
Tokens	How text is cut into tokens	Qwen3-1.7B-Base, o200k_base
Next token	Next-token probabilities, and temperature	Qwen3-1.7B-Base
Base vs chat	One prompt, to a base model and a chat model	Qwen3-1.7B-Base, Qwen3-1.7B
Token embeddings	Each token's vector, and its nearest tokens	Qwen3-1.7B-Base
Passage embeddings	One vector per passage, and similarity	all-MiniLM-L6-v2

> Everything runs on this laptop. o200k_base is OpenAI's tokenizer; the rest are open-weight models from Hugging Face.

2 · MODELS AND PERFORMANCE

Who makes the frontier models, what they cost, and how they are compared. Prices and scores are as of September 18, 2026.

The frontier: closed weights

PROVIDER	MODEL	RELEASED	INPUT / OUTPUT, $ PER M TOKENS
Anthropic	Claude Fable 5.1	Sep 2026	10 / 50
Anthropic	Claude Opus 5	Jul 2026	5 / 25
Anthropic	Claude Sonnet 5	Jun 2026	2 / 10
OpenAI	GPT-6 Astra	Sep 2026	10 / 50
OpenAI	GPT-5.6 Sol	Jul 2026	5 / 30 (4 / 20 until Nov 21)
Google	Gemini 3.8 Flash	Sep 2026	1.50 / 7.50 (0.75 / 3.75 until Dec 31)
xAI	Grok 4.6	Aug 2026	2 / 6
Meta	Muse Spark 1.3	Sep 2026	1.25 / 4.25

The frontier: open weights

PROVIDER	MODEL	RELEASED	INPUT / OUTPUT, $ PER M TOKENS
Alibaba	Qwen3.8-Max	Aug 2026	2 / 6
Zhipu (Z.ai)	GLM-5.3	Aug 2026	1.40 / 4.40
Moonshot	Kimi K3	Jul 2026	3 / 15
DeepSeek	DeepSeek V4 Pro	2026	1.32 / 3.96
NVIDIA	Nemotron 3 Ultra	Jun 2026	free
Mistral	Mistral Medium 3.5	Apr 2026	1.50 / 7.50

> Open weights means anyone can download the model and run it on their own hardware. The prices are for the developer’s hosted API; DeepSeek charges half off-peak.

Some benchmark scores

OpenAI's comparison table in its Astra announcement, September 3. All four columns are OpenAI's figures.

BENCHMARK	ASTRA	FABLE 5.1	OPUS 5	GPT-5.6 SOL
Terminal-Bench 4.0	57.7%	55.8%	52.3%	37.3%
Humanity's Last Exam, with tools	57.2%	65.0%	63.6%	—
FrontierMath Tier 4	97.6%	87.8%	73.2%	—
AutomationBench	41.4%	31.4%	26.9%	18.1%

> OpenAI does not say where the Claude numbers came from or which harness they ran in. OpenAI funded FrontierMath and has exclusive access to part of it. Even here, the leader changes from one benchmark to the next.

An aggregation across multiple benchmarks

Artificial Analysis Intelligence Index, v4.3, best setting for each model.

MODEL	SCORE	MODEL	SCORE
Claude Fable 5.1	53	GLM-5.3 (open)	45
GPT-6 Astra	53	Qwen3.8-Max	45
Claude Opus 5	51	Kimi K3 (open)	44
Muse Spark 1.3	48	Grok 4.6	44
GPT-5.6 Sol	47	Gemini 3.8 Flash	41

> The best open model trails the best closed ones by 8 points.

Cost per task

Cost per task is the price per token times the number of tokens the model uses.

MODEL, EFFORT SETTING	INDEX SCORE	COST PER TASK
GPT-6 Astra, max	53	$3.26
Claude Fable 5.1, max	53	$7.63
Claude Opus 5, max	51	$5.86
GPT-5.6 Sol, max	47	$1.99
GPT-6 Astra, low	46	$0.82

> Astra and Fable 5.1 have the same list price, $10 / $50. At max effort Astra writes about 27,000 output tokens per task and Fable 5.1 about 78,000. Artificial Analysis, September 2026.

Who businesses pay

Ramp AI Index, July 2026: the share of US businesses with a paid subscription.

43.5%
ANTHROPIC

39.7%
OPENAI

6.1%
MODEL-SERVING PLATFORMS, WHICH HOST OPEN MODELS

> Many firms pay for more than one. Ramp reports that OpenAI has been growing faster than Anthropic so far in the third quarter.

Chat with open source models

TRY IT

1. Log in at `openrouter.ai`. If you have no account, sign up; it is free.
2. Open the Playground at `openrouter.ai/chat`.
3. Pick a model whose name ends in “(free)”. Today’s list includes GLM 5.2, Nemotron 3 Ultra, DeepSeek V4 Flash, and Qwen3.8 27B.
4. Ask it something you would ask Claude or ChatGPT, and compare the answer.

> Free models allow 50 requests a day on an account that has not bought credits. The Playground can put a second model beside the first to answer the same prompt.

3 · COMPUTER USE

A model that operates a computer the way you do: it looks at the screen, then clicks and types.

What computer use is

It sees

A screenshot of the screen, sent to the model as an image.

It acts

The model replies with an action: click at these coordinates, type this text, scroll, press a key.

It checks

The app carries out the action, takes a new screenshot, and sends it back. The loop repeats until the task is done.

> Each screenshot costs about 1,000 to 1,800 input tokens. A task that takes a hundred steps sends a hundred screenshots.

Browser use and computer use

BROWSER USE

Works inside one web browser.

Reads the page's text and structure as well as screenshots.

Usually a browser extension.

COMPUTER USE

Works across the whole desktop: Excel, Outlook, any app with a screen.

Sees only pixels, so it is slower and misses more.

Needs a desktop app allowed to record the screen and control the mouse.

> Same loop, different reach: the web, or any app on the machine.

What to install

	BROWSER USE	COMPUTER USE	CHEAPEST PLAN
Anthropic	Claude in Chrome extension	Claude Desktop, with computer use turned on in Settings (beta)	Pro, $20 a month
OpenAI	Browser tabs in the ChatGPT desktop app	Computer Use plugin in the ChatGPT desktop app	Plus, $20 a month (Free has limited access)
Google	Auto browse in Chrome (U.S. preview)	Gemini Spark in the Gemini Mac app (beta)	AI Pro, $19.99 for browser; Ultra for desktop

> Each needs a paid plan for regular use. Developers can also call computer use through each vendor’s API and pay only for tokens.

How well it works

OSWorld 2.0: 108 long desktop tasks. The median task takes a skilled person about 1.6 hours.

MODEL	PARTIAL CREDIT	TASK FULLY COMPLETED
Claude Fable 5.1	77.9%	41.7%
Claude Opus 5	75.4%	39.6%
GPT-6 Astra	72.6%	—
Gemini 3.8 Flash	59.0%	—

> Vendor-reported, September 2026; OpenAI used an offline version of the test. On tasks that take a person more than about 2.7 hours, no model fully completes more than 10 percent.

Web pages can give orders

Text on a page can instruct the model. All three vendors warn about this prompt injection and say their defenses are not perfect.

It acts as you

It clicks with your accounts and sees whatever is on your screen. Anthropic advises against using it with financial, legal, or medical data.

It asks first

Permission for each app, and confirmation before purchases, posts, and messages. It is slower than a connector or an API.

What the vendors warn about

4 · HOW AGENTS WORK

A chatbot returns text. An agent is a chatbot with tools it can ask to use.

A chatbot produces text

What it receives

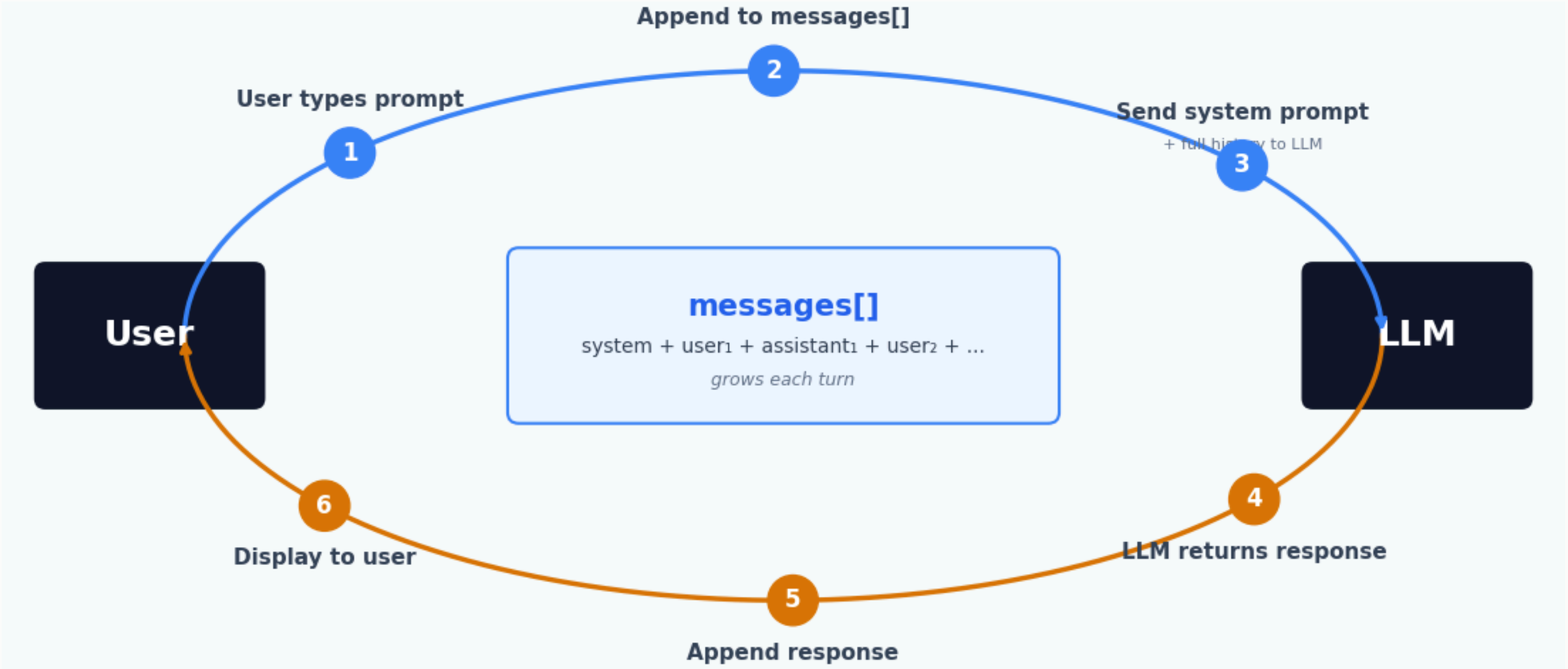
A system prompt, the conversation so far, and your message.

What it returns

More text.

It cannot open your file, run your query, or check its own arithmetic.

Inside a chatbot



An agent has three parts

The model

Decides. Emits text: sometimes an answer, sometimes a request to use a tool.

The harness

Ordinary software wrapped around the model. Runs the tool the model asked for.

The tools

The things that touch files, databases, and other systems.

> The model never runs anything itself. It chooses tools to use. The harness executes, and the harness returns information about the tool outcome to the model.

What the tools are

Read and write files

Documents, spreadsheets, code.

Run code

Python, SQL, the command line.

Search and fetch the web

Run a search, then download and read a page.

Use a screen

Click and type in a browser or a desktop app.

Call a service

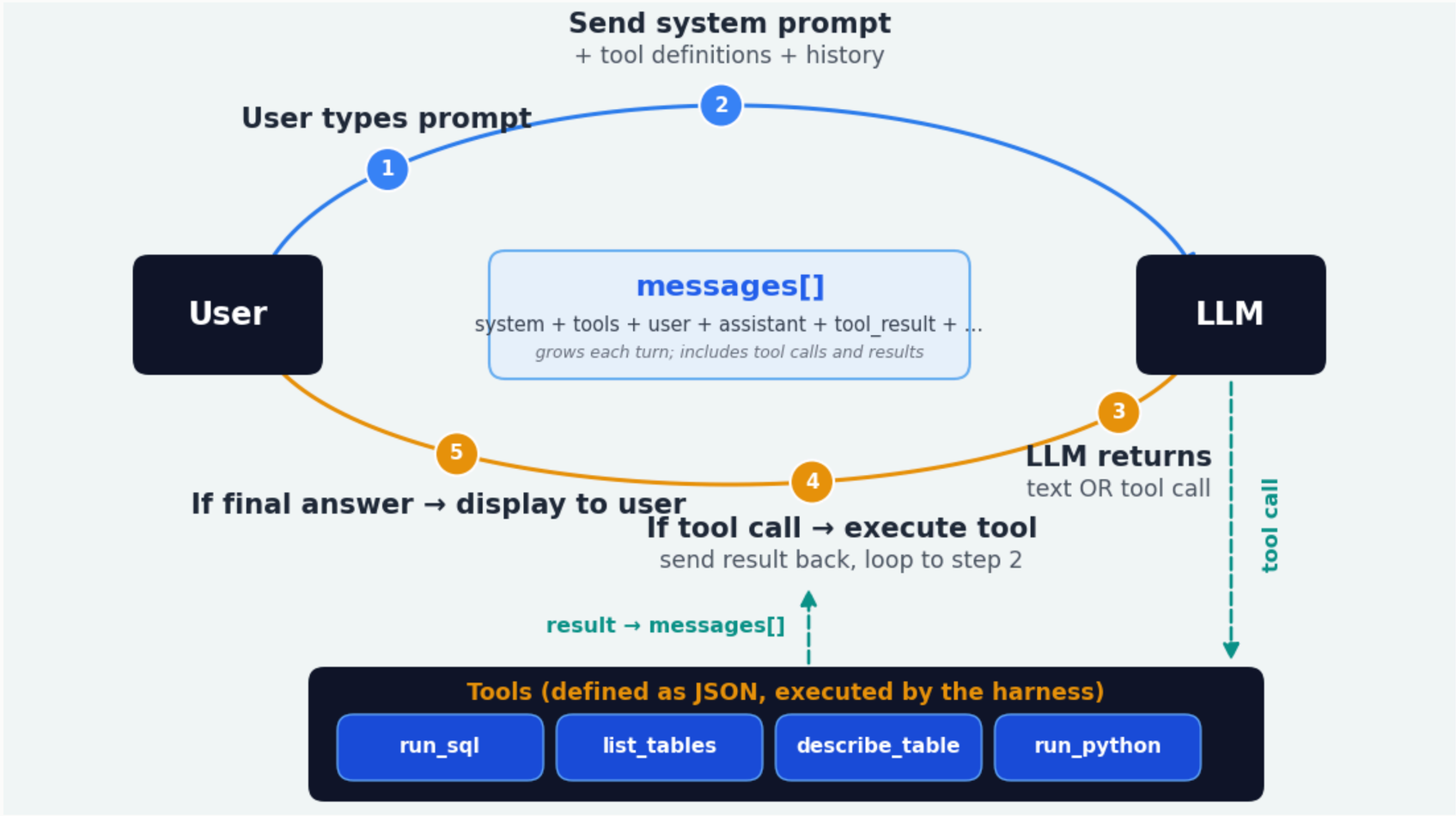
A database, an API, or Gmail and Slack through a connector.

Start another agent

A subagent with its own instructions and tools.

> Smaller ones are common too: ask you a question, save a memory, load a skill, keep a to-do list.

Inside an agent



Prompt and response

PROMPT: HARNESS TO MODEL

```
{
  "model": "claude-sonnet-5",
  "system": "You are a coding agent...",
  "tools": [{
    "name": "bash",
    "description": "Run a command",
    "input_schema": {"type": "object",
      "properties": {
        "command": {"type": "string"}}}
  }],
  "messages": [{"role": "user",
    "content": "How many orders?"}]
}
```

RESPONSE: MODEL TO HARNESS

```
{
  "role": "assistant",
  "content": [{
    "type": "tool_use",
    "id": "toolu_01A",
    "name": "bash",
    "input": {
      "command": "wc -l orders.csv"
    }
  }],
  "stop_reason": "tool_use"
}
```

The loop

1. You ask. One sentence, in ordinary language.
2. It picks a tool. The model emits a tool name and arguments instead of an answer.
3. The harness runs it. Software outside the model checks the call against its rules, then runs it.
4. The result goes back. The harness appends what it keeps; the model reads it and decides what is next.

> Repeated until the model answers instead of asking, or until the harness hits its turn limit.

What the model is sent

The system prompt

Your message

Every prior turn

A description of each tool it may call

The contents of files you attached

The result of every tool call so far

All of it is text, and the model treats all of it the same way.

Five choices written into the harness

What the model reads before it writes

What is dropped when the model's context fills

Which of the model's tool calls run without asking you

What happens when a tool call fails

Whether to generate subagents

> The programmers who wrote the harness made these choices in code, before you typed anything.

	HARNESS	WHERE YOU MEET IT
Anthropic	Claude Code	Terminal, desktop app, IDEs, web. The Claude Agent SDK for agents you build.
OpenAI	Codex	Codex CLI, ChatGPT desktop app, IDEs, cloud. The Agents API (beta, Sep 10) for agents you build.
Google	Antigravity	Desktop app and Antigravity CLI. Replaced Gemini CLI in June 2026.
Open source	OpenCode · Pi · OpenHands · Goose	Any provider’s models, including open-weight models you host.

> Codex CLI is also open source (Apache 2.0) but is built around OpenAI’s models.

Harnesses by name

Try it

1. Install OpenCode 1.18.0 or newer: opencode.ai/docs.
2. Put orders.csv and order_details.csv in a new folder. Then type `opencode` in a terminal opened in that folder:
   Mac: open Terminal, type `cd`, drag the folder onto the window, and press Return.
   Windows: right-click the folder name in File Explorer and choose Open in Terminal.
3. Type `/models` and choose Big Pickle under OpenCode Zen. It is free and needs no account.
4. Ask: *Merge orders.csv and order_details.csv on OrderID and save the result as orders_complete.csv.*
5. Ask: *List the exact names of every tool you have.*

> Free models may train on your prompts. Use nothing confidential.

5 · DOING WORK WITH AI

Where to work

Online chatbot

In a browser. Nothing to install.

Desktop app or CLI

Installed on your computer. Works with files on your computer.

lab803.kerryback.com

A container on a server, with the software already installed.

lab803

Sign in at `lab803.kerryback.com` with your NetID as the username and your Rice student ID (S followed by eight digits) as the password.

A container

Each of you has a container on a Linux server.

Claude Code and Python

Both are installed in each container.

Sonnet 5

Claude Code is connected to Sonnet 5 through an API key.

Right panel

Claude Code. The prompt window is at the bottom.

Left panel

Files (the default), Terminal, and View. Ignore Terminal and View for now.

Three-dots menu in Files

Upload from your laptop and download to your laptop.

Workspace

Important

Refresh

Click Refresh, above the left panel, to see a new file in Files.

No Microsoft Office

Double-clicking a Word, PowerPoint, or Excel file will not open it. Office is not installed.

Download

Download Office files from the three-dots menu and open them on your laptop.

Ask

Ask a question, anything. Type in the prompt window and press Enter.

Upload

Upload a Word doc and ask Claude to read it.

Download

Ask Claude to create a Word doc, then download it and read it.

Explore

Working with Claude Code CLI

STOPPING AND STARTING

To exit Claude Code, press Ctrl-C twice.

- In lab803, press Enter to reconnect. A new conversation starts.
- In other environments, type `claude` and press Enter for a new conversation.

SLASH COMMANDS

Prompts that start with `/`.

`/clear`	Clear the context window
`/resume`	Resume a previous conversation
`/help`	Get help on using Claude Code

Start a new conversation for a new task

AI reads the entire conversation again each time you prompt. Irrelevant stuff consumes tokens and may be confusing.

When a conversation gets very long

Tell AI to write a handoff document so you can start a fresh conversation.

Managing conversations

Try it: build an Excel workbook

> Tell Claude: Build an Excel workbook containing a mortgage amortization table. Include a chart of the amount going to interest and the amount going to principal each month.

Download and investigate

Download the file, open it on your own machine.

1. Click a payment cell. Formula, or a number?
2. Change the rate to 12%. Does the chart move?
3. Change the term to 15 years. Does the table shorten?

> If any answer is no, say so in one sentence and ask Claude to fix that specifically.

1. Ask Claude to add a worksheet containing a plot of the monthly payment as a function of the interest rate.
2. Download and check.

Ask for something else

Try it: data wrangling, charts, docs

In the Claude tab at lab803, the five Northwind spreadsheets are in `data/`.
Elsewhere, download northwind.zip.

1. Ask: *Which countries generated the most revenue?*
2. Ask: *Create a bar chart of revenue by country for the top five countries.*
3. Ask: *Create an Excel workbook containing the revenue table and the chart.*
4. Ask: *Create a Word document containing a one-page summary with the table and the chart.*
5. Ask: *Create a PowerPoint deck containing three slides: the question, the table, and the chart.*

> Open each file before asking for the next. At lab803, download it from the file browser tab.

The xlsx, docx, and pptx skills

An Office file is a zip archive of XML files. Skills tell the AI which tools to use to write it.

SKILL	NEW FILE	EXISTING FILE
xlsx	Python (openpyxl, pandas)	Python (openpyxl) opens the workbook and changes cells and formulas
docx	JavaScript (docx-js)	Unzip, edit the XML with Python, zip again
pptx	JavaScript (PptxGenJS)	Unzip, edit the XML, zip again

> The lab has no Office software, so it cannot display the files it makes. Download them and open them on your laptop.

Claude and ChatGPT in Excel

CLAUDE FOR EXCEL

From Anthropic. Paid Claude plans: Pro, Max, Team, Enterprise.

Cites the cells behind its answers. Edits cells without breaking formulas. Builds pivot tables, sorts, and filters.

Also available for PowerPoint, Word, and Outlook.

CHATGPT FOR EXCEL

From OpenAI. All ChatGPT plans. Also works in Google Sheets.

Builds sheets from a description, writes formulas, and sorts and updates tables.

Asks before it edits the workbook.

> Both work inside Excel on the open workbook. The xlsx skill never opens Excel; it runs Python to write the file. Anthropic warns that an outside workbook can hide instructions the add-in will follow.

6 · PROMPTING TIPS

Plan

> **Prompt**
>
> Plan this and explain your plan to me.

Afterwards

> **Prompt**
>
> `Explain what you did step by step. Did you make any decisions that we have not discussed? Write this as an HTML doc.`

HTML is easy for AI to write and easy to read.

Numbers need code

If a number appears in prose and no code produced it, it was predicted.

> **Prompt**
>
> `Compute this in Python and show me the code and the output. Do not report a number you did not compute.`

The same rule applies to counts, percentages, dates, and anything summed across rows.

Verify claims

Ask what would falsify it

“What would have to be true for this to be wrong, and how would I check?”

Make it show its source

The file, the row count, the line of code.

Generator–critic

1. Get the first answer.
2. Start a fresh conversation and give it the answer without the reasoning that produced it.
3. Ask it to find what is wrong with it.
4. Take the objections back to the first conversation.

The second conversation has no stake in the first answer.

Most ‘prompt engineering’ techniques from a few years ago are now either useless or actually counterproductive.

Tell it what you want, not what to do

I want … How can we do it?

If something doesn’t work

Try again.

> Treat AI as an infinitely patient colleague with a poor memory.

Good enough prompting

Try it: planning

Three rigs, twenty wells, minimize lost production: workover.md. At lab803 it is in `data/`.

1. Upload workover.md and ask: *Find the best routes for the rigs.*
2. Start a new conversation (`/clear` in the Claude tab). Upload the file again and ask: *Plan how to do this and share your plan.* Read the plan, then ask it to proceed.

> Did it use the same method both times?

Try it: explain and critique

1. Ask: *Write an HTML doc explaining what you did, step by step.*
2. Start a new conversation. Tell it to read the HTML doc and ask: *Read this and critique the method.*