AI MADE EZ · LEVEL 2 SUMMARY

How AI kitchens work

11 lessons · about 67 minutes in all · free

01 How AI kitchens work **02** The order window **03** House rules on the wall **04** Appliances
05 Plug your apps into AI **06** The recipe binder **07** The head chef **08** The chef in your own kitchen
09 Taste tests **10** Plan your own kitchen **11** Everything is a wrapper

01 How AI kitchens work.

- Claude is a model.
- It's an LLM: a large language model.
- Your order is a prompt.

Model = the chef (Claude)
LLM = a chef that predicts the next piece of text
Prompt = your order
Token = the pieces it counts and charges by
Context window = the counter: all it can see

Say it: "Claude is a model, an LLM. It sees only what's in its context window, and it charges by tokens."

02 The order window.

- The API is the order window.
- The API key is your account card.
- The window remembers nothing.

API = the window apps use to send Claude tickets
API key = your account card: bills your tab (keep it secret)
The app = the waiter carrying tickets and plates
Request = the ticket · Response = the plate
Messages = the list of user and assistant turns

Say it: "Apps talk to Claude through the API. Each request carries the whole chat, because the API remembers nothing."

03 House rules on the wall.

- The system prompt is the house rules on the wall.
- Prompt engineering is writing clear tickets.
- Hallucination: Claude sounds sure but is wrong.

System prompt = rules on the wall, written by the builder
Prompt engineering = clear tickets: who, example, shape
Jailbreak = a user talking around the rules
Examples = a few finished plates to learn from
XML tags = labeled boxes: <facts>, <rules>, <example>

Say it: "The system prompt sets rules, hard limits belong in the app, and Claude can still hallucinate."

04 Appliances.

- Tools are appliances.
- The chef asks; the app runs it.
- Each appliance has a label.

Tool = an appliance Claude can ask for
Tool definition = the label: name, what it does, inputs
Tool call = Claude asks to use it
Tool result = what the app sends back
The app runs your tools; Claude only asks

Say it: "With tool use, Claude asks for an action and the app actually runs it."

06 The recipe binder.

- RAG: look it up, then cook.
- Nothing gets trained.
- The binder is indexed by meaning.

RAG = find the right pages, then Claude answers from them
Chunking = cutting documents into findable pages
Embeddings = number codes for meaning, so similar pages can be found
Vector database = finds the closest-meaning pages
Citations = which page the answer came from

Say it: “RAG looks up the right pages and hands them to Claude. Nothing gets trained.”

07 The head chef.

- An agent is a chef running the whole dinner.
- A workflow is a prep checklist.
- Anthropic’s advice: start simple.

Agent = Claude picks its own steps and tools until done
Agent loop = think → tool → check → repeat
Workflow = a prep checklist: fixed steps set by the builder
Orchestrator = head chef splitting the job
Subagent = sous-chef with its own counter, reports a summary

Say it: “An agent picks its own steps in a loop. Start with one request, and keep a person on risky actions.”

05 Plug your apps into AI.

- Out of the box, Claude can’t see your stuff.
- MCP is a standard plug.
- Plugged in, Claude can look things up and do things.

Out of the box, Claude can’t see your calendar, email or files
MCP = Model Context Protocol, a standard plug so apps can connect to AI
MCP server = the plug made for one app (Google Calendar, Drive, Slack…)
Connector = the Claude app’s name for an app you plugged in
Plugged in, Claude can read things and do things. Keep risky actions on “always ask”

Say it: “Claude can’t see your stuff until you plug it in. MCP is the standard plug; in the Claude app, a plugged-in app is a connector. Only plug in what you trust.”

08 The chef in your own kitchen.

- Claude Code works inside your project.
- CLAUDE.md is the note on your fridge.
- Skills are recipe cards.

Claude Code = an agent that works in your project files
CLAUDE.md = the fridge note, read every session (requests)
Skill = recipe card (SKILL.md), picked by its description
/command = a shortcut you type; your own ones are made as skills
Hook = a bell that rings at set moments (enforced, like permissions)

Say it: “Claude Code works inside a project. CLAUDE.md is a request; hooks and permission rules are enforced.”

09 Taste tests.

- An eval is a taste test with a scorecard.
- “It looked fine” is a vibe check.
- Graders do the tasting.

Eval = same questions, same scorecard, every change
Success criteria = what good means, measurable
Test set = the fixed questions (keep some unseen)
Grader = code for exact checks, LLM-as-judge for fuzzy ones
Regression = a fix that broke something else

Say it: “Before we ship a change, we re-run the same test set and compare scores. That’s an eval.”

10 Plan your own kitchen.

- An architect picks the pieces.
- Every piece has a price.
- Guardrails are the safety rails.

Chef (model):
Order window (API):
House rules (system prompt):
Appliances (tools):
Outlets (MCP):

Say it: “Start with the smallest setup that works, test it, and add pieces only when a real need shows up.”

11 Everything is a wrapper.

- AI is layers.
- A wrapper is a layer on top of someone else’s layer.
- Each layer charges for what it adds.

You → apps → models → chips → electricity
Wrapper = an app built on top of someone else’s model
Model = the chef: the trained AI that writes (Claude, GPT, Gemini)
Thin wrapper = a screen over one prompt. Easy to copy
Good wrapper = adds your data, design or workflow

Say it: “Most AI apps are wrappers: they sit on a model and add their own part. The question is always what they add.”

Prompts to try

Change the parts in [brackets] to fit you. One a day is plenty.

01 · How AI kitchens work.

ORDER #001 copy · paste · send

```
I work on writing emails. Give me a
one-page cheat sheet I can keep:
1. One real task from my work, and
roughly how many tokens it would use
(mark it as an estimate).
2. When my chat gets long enough to
start a fresh one, in one line.
3. Is Sonnet the right Claude model
for this? One line each comparing it
to Haiku, Opus, Fable.
```

02 · The order window.

ORDER #002 copy · paste · send

```
I don't code. In plain words first,
explain what a marketing tool sends
to Claude to write ad ideas, using
Sonnet. Then show the actual request
with YOUR_KEY_HERE as the key.
1. Add a plain-words note next to
each part (model, messages, reply
length).
2. Show a made-up sample response
(label it a sample) and point out
the token counts.
3. Explain why the app must resend
earlier messages for a follow-up.
No real API key, use a placeholder.
```

03 · House rules on the wall.

ORDER #003 copy · paste · send

```
Act as the help assistant for a
bakery. Your system prompt (house
rules):
- Under 80 words
- If unsure, say so

Then: 1. Answer a tricky customer
question you make up. 2. Show the
same answer with no rules. 3. Tell
me in 2 lines which rule helped
most, and one rule I should add.
4. Give me the final rules as one
block I can paste into a Claude
Project’s instructions.
```

04 · Appliances.

ORDER #004 copy · paste · send

```
For a restaurant ordering app, write
the tool definition for "check
what’s in stock" that I could hand
to a developer.
1. Name, what it does, what inputs
it needs: in plain words, then as
JSON (the format apps read).
2. Walk through one example: user
question → tool call → tool result →
final answer.
3. Say who actually runs the tool
(Claude or my code) and one way it
could go wrong.
```

05 · Plug your apps into AI.

06 · The recipe binder.

ORDER #005 copy · paste · send

```
Make me a one-page connector plan I
can show IT. I use: Google Calendar,
Gmail.
A table with one row per service:
one useful thing Claude could do if
connected · read-only or can change
things · one safety rule.
Then explain in 3 lines what MCP, an
MCP server and a connector are,
using a wall-outlet comparison.
```

ORDER #006 copy · paste · send

```
Make me a setup plan: I want Claude
to answer customer questions. My
documents: a 10-page FAQ.
1. Do I need RAG, or can it all just
be pasted in or uploaded to a Claude
Project? Say why in 2 lines.
2. One line each: at what size would
I need RAG, and what would chunking,
embeddings and a vector database do
for me then?
3. How do I make answers cite the
page they used, and what should it
do when no page fits?
4. If it fits in a Claude Project:
list the files to upload and write
the project instructions I can
paste.
```

07 · The head chef.

ORDER #007 copy · paste · send

```
Make me a one-screen plan for this
job: reply to every customer email.
1. Would you build this as one
request, a fixed workflow, or an
agent? Why, in 2 lines.
2. Draw the steps as a short list
(think / tool / check).
3. Where must a human approve before
it acts? 4. Would subagents help?
One line.
```

08 · The chef in your own kitchen.

ORDER #008 copy · paste · send

```
(Paste this into Claude Code, in an
empty practice folder. If you only
have the Claude app, paste it there
and it will show you the files
instead.)

Set up my small business website for
working with you:
1. Create CLAUDE.md (under 20 lines)
with these rules:
   - Explain changes in plain words
   - Ask before deleting anything
2. Create one small skill at
.claude/skills/plain-words/SKILL.md
that rewrites any text in plain
words (name and one-line description
at the top).
3. Show me both files, then tell me
which rule should be a hook or a
permission rule instead, because it
must always hold.
```

09 · Taste tests.

10 · Plan your own kitchen.

ORDER #009 copy · paste · send

```
Help me build a tiny eval for a
help-desk bot. What good means:
Correct facts, Short answers.
1. Turn those into 3 specific,
measurable checks.
2. Write 8 test questions, including
2 tricky edge cases.
3. For each check, say if code or an
LLM judge should grade it, and write
the judge's instructions.
4. Give me a blank score table to
fill in after I try these questions
on the real tool.
```

ORDER #010 copy · paste · send

```
Be my AI architect. I want to build
a helper for my small business. I
think it needs: Tools, My documents,
Taste tests.
Give me a one-screen plan with:
model, system prompt (3 rules),
tools, MCP connectors, documents
(paste in or RAG?), workflow or
agent, the eval, and guardrails.
Then: which piece I picked that I
don't need yet, the simplest version
I could build this week, and a rough
cost per 100 uses, marked as an
estimate (prices change, so tell me
where to check them).
```

11 · Everything is a wrapper.

ORDER #011 copy · paste · send

```
Explain the layers under an ai
website builder in plain words, for
someone who doesn't code.
1. Which layer is it: an app that
wraps an AI model, or a model
itself? If you don't know for sure,
say so.
2. What does it add on top of the
model (its questions, design, data,
workflow)?
3. What could I do with a general AI
chat instead, and what would I lose?
Don't guess facts about the company.
Only say what's public, and tell me
where to check.
```


Made by Alexandra Vargas · not affiliated with Anthropic