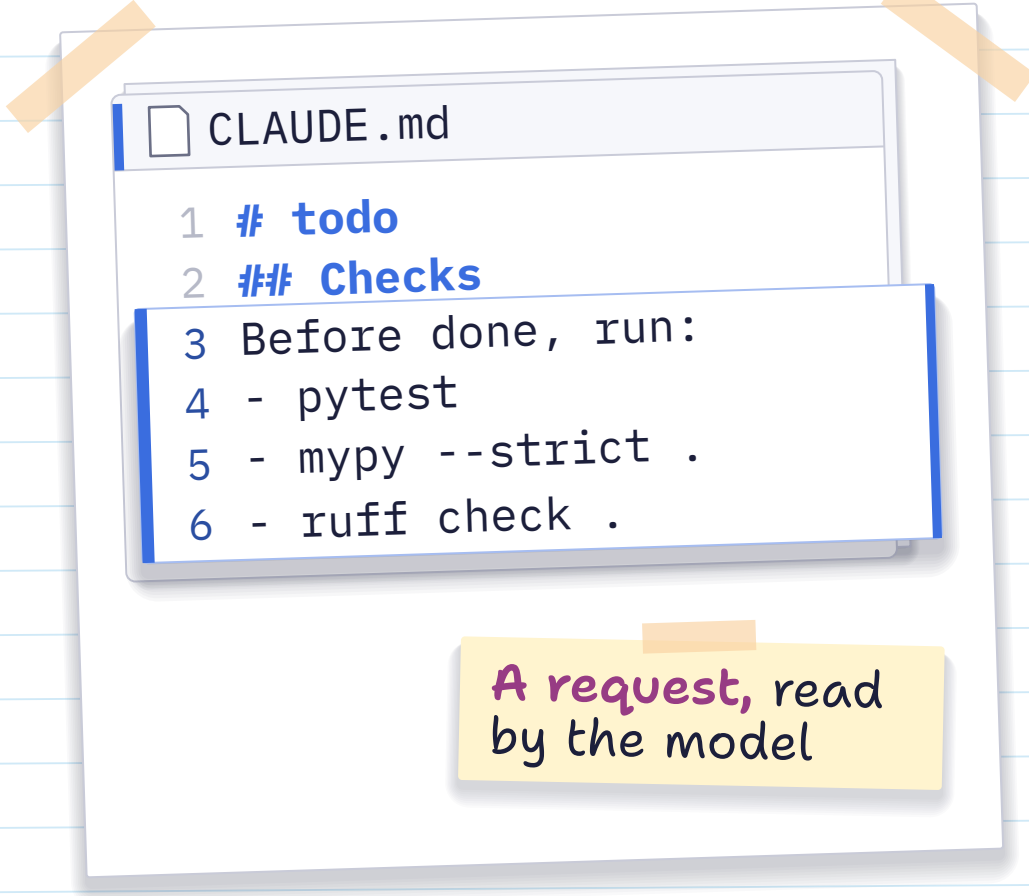
Prof. Aviv Zohar

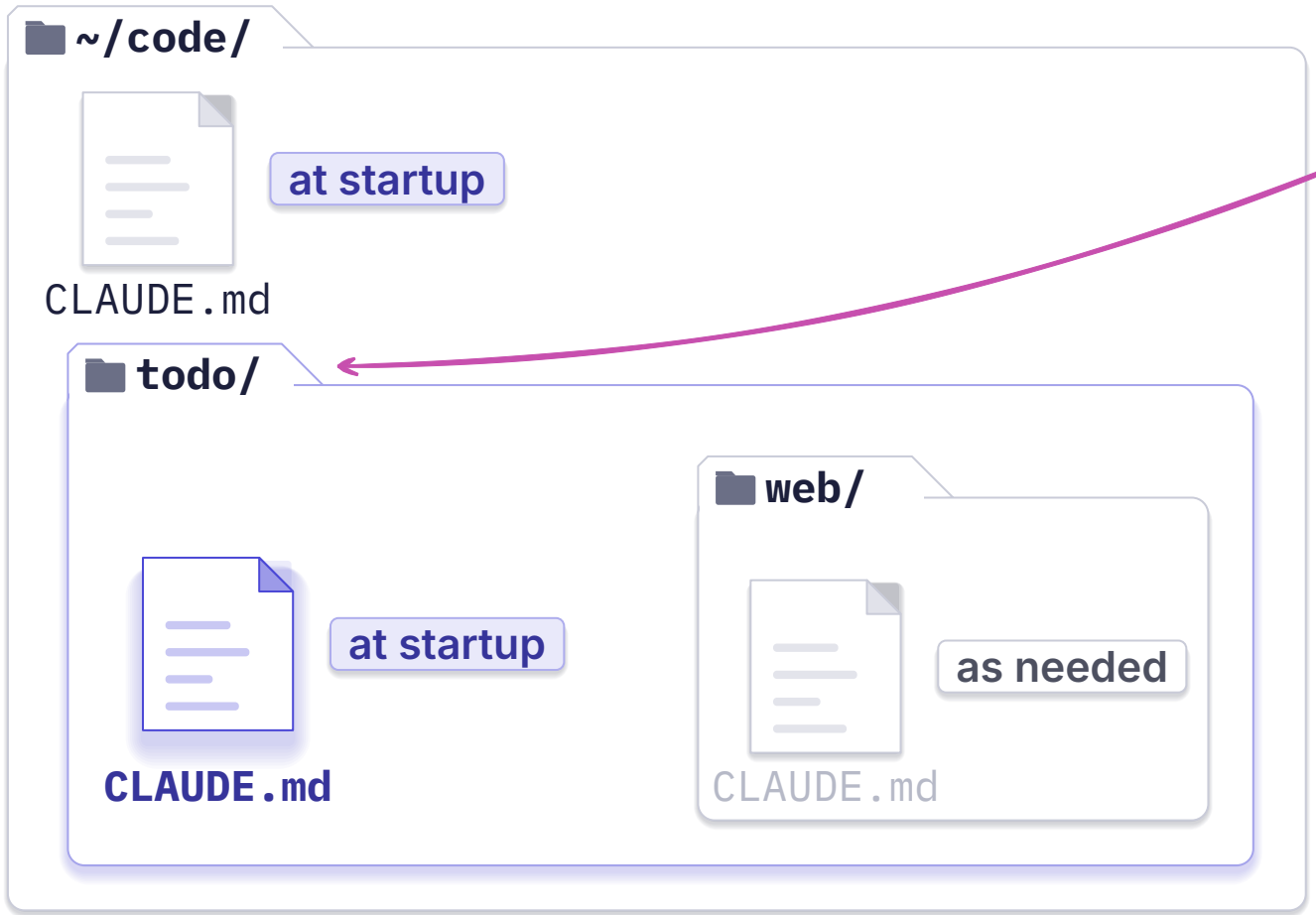
Semester A, 2026–7

Writing CLAUDE.md

Useful instructions, and what a written rule cannot do



The project's instruction file loads at startup



shell

```
~/code/todo $ claude
```

Start Claude Code in the project.

Codex reads `AGENTS.md`: the same kind of file under another name.

`/init` already wrote a first draft. **What belongs in it?**

Record the exact check commands

CLAUDE.md 3 lines

```
1 # todo
2 ## Checks
3 - Make sure everything works.
```

Which checks? Run when?

The agent has to guess.

Record the exact check commands

CLAUDE.md 6 lines

```
# todo
## Checks
Before reporting a task as done, run:
- pytest
- mypy --strict .
- ruff check .
```

when to run them

the tests
type checking
linting

This example: a Python project with three tools configured.

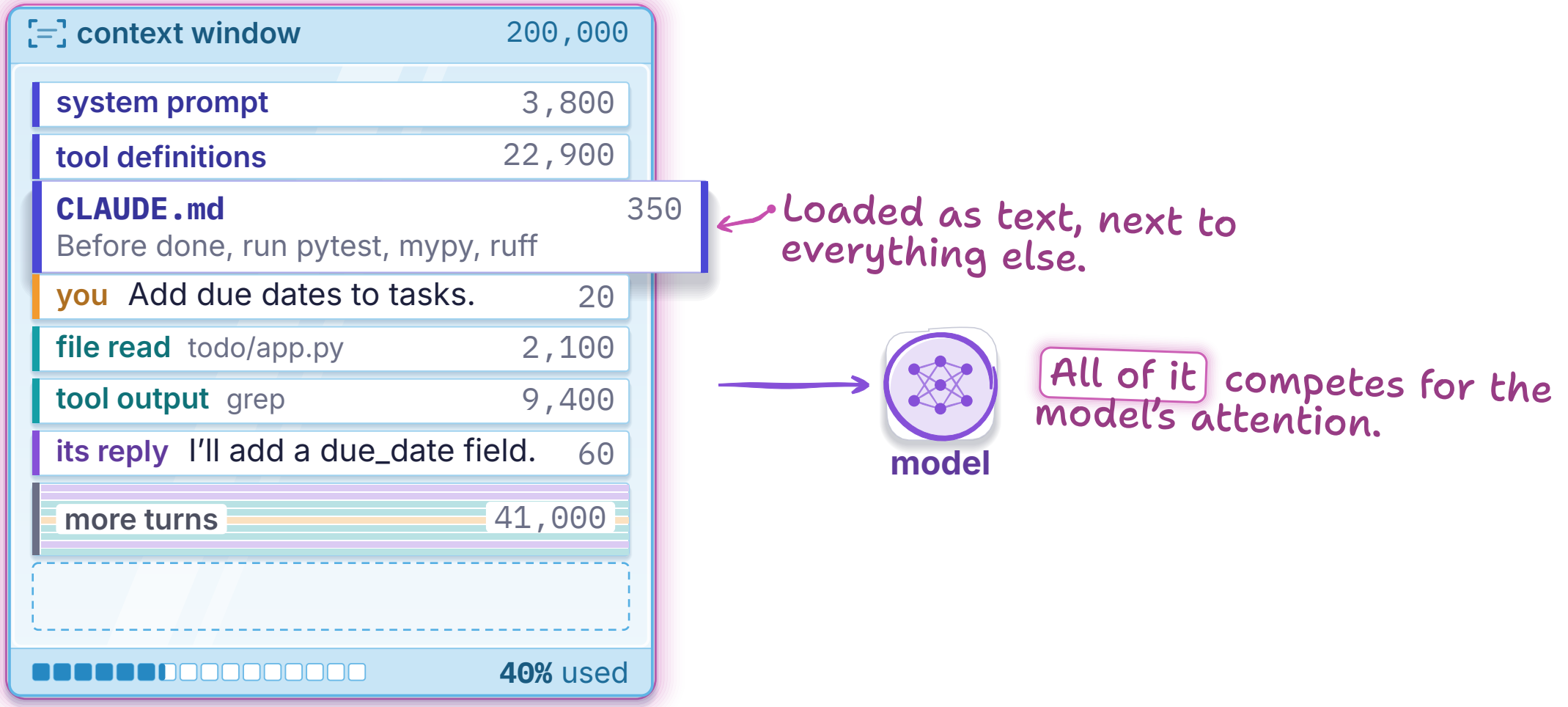
Record the exact check commands

CLAUDE.md 12 lines

```
# todo
## Checks
Before reporting a task as done, run:
- pytest
- mypy --strict .
- ruff check .
## Don't run
- git push: I push after review.
## Conventions
- Log with logging, never print.
## Where things are
- Database code: todo/storage.py
```

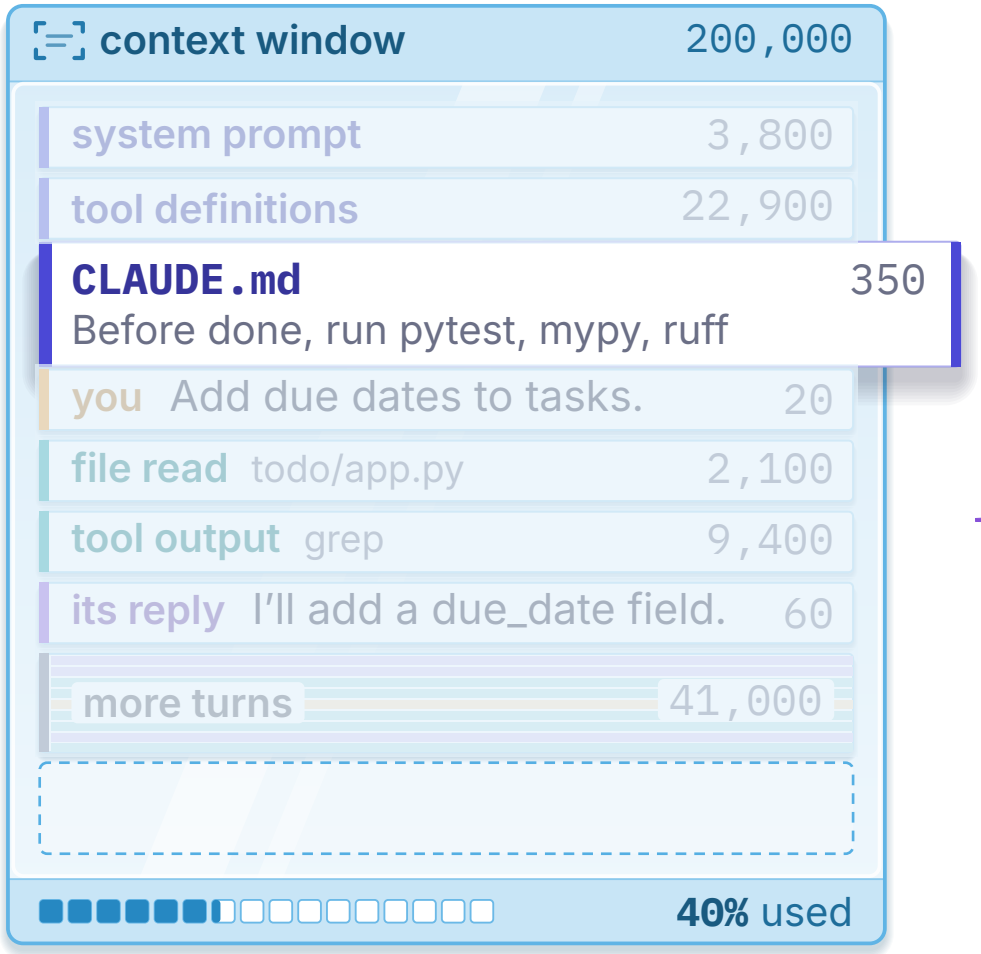
A line earns its place when it records a project fact the agent would otherwise guess, or ask about again in another session.

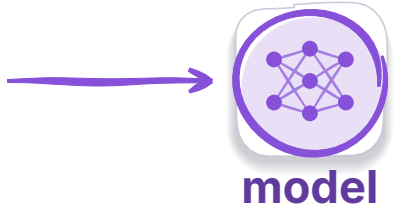
A rule in the instruction file is a request



illustrative values

A rule in the instruction file is a request





The same rule, three sessions

session 1	✓ ran all three
session 2	✗ skipped them
session 3	✗ ran only pytest

A fuzzy tool: it can follow the rule, overlook it, or misapply it.

illustrative values

When it reports the checks pass, look for the run

agent session ~/code/todo

tool call and its output

Bash pytest

```
12 passed in 0.41s
```

tool call and its output

Bash mypy --strict .

```
Success: no issues found in 6 source files
```

tool call and its output

Bash ruff check .

```
All checks passed!
```

the report

agent written by the model

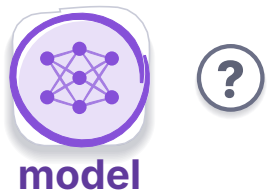
Due dates added. All checks pass.

Look for the run: each command the agent ran, followed by its output.

A rule that must hold needs a hook

CLAUDE.md

```
## Checks
Before reporting done, run:
- pytest
```



Read by the model, which may follow it.

inside the harness

`Edit todo/app.py`

after every edit

pytest

exit 0

built in upcoming videos

Run by the harness, whether or not the model remembers.

Keep the instruction file focused

CLAUDE.md

```
# todo
Write clean, readable code.
Tests: python -m unittest
Never edit generated/. Builds overwrite it.
Keep the code clean and readable.
Tests live in tests/, one file per module.
```

Every line is loaded, in every session.

Long instruction files reduce adherence, Claude Code's guidance warns.

Claude Code docs, code.claude.com/docs/en/memory

Keep the instruction file focused

CLAUDE.md

```
1  # todo
-  Write clean, readable code.                    ⓐ
-  Tests: python -m unittest                      ⓑ
4  Never edit generated/. Builds overwrite it.
-  Keep the code clean and readable.              ⓐ
6  Tests live in tests/, one file per module.
```

ⓐ They repeat each other, and say nothing about this project. Both go.

ⓑ Stale: the project moved to `pytest`. Correct it.

Keep the instruction file focused

CLAUDE.md

```
1 # todo
- Write clean, readable code.
+ Tests: pytest
4 Never edit generated/. Builds overwrite it.
- Keep the code clean and readable.
6 Tests live in tests/, one file per module.
```

Builds overwrite it: edits there are lost.

This project's choice, which the agent cannot guess.

Add a rule after a mistake you have seen. **Record** known constraints early. **Review** for stale or repeated lines.

Recap



Keep it focused

Project facts the agent cannot guess. Cut repeated, generic and stale lines.

pytest

The exact check commands, and when to run them

Then check the session to see that they ran.



Still a request

A rule that must hold needs a hook, which runs the check on its own.

Recap



Keep it focused

Project facts the agent cannot guess. Cut repeated, generic and stale lines.



The exact check commands, and when to run them

Then check the session to see that they ran.



Still a request

A rule that must hold needs a hook, which runs the check on its own.

Reusable checks: less to verify by hand after each change.
The tests you keep go on checking earlier requirements as the project grows.