Hands-On Exercise

Let's build! To-do list app

1. /model haiku
2. run git init
3. Build a simple web-based to-do app. Users should be able to: add a task, mark a task as complete, delete a task.
4. How many lines of code did my app require?

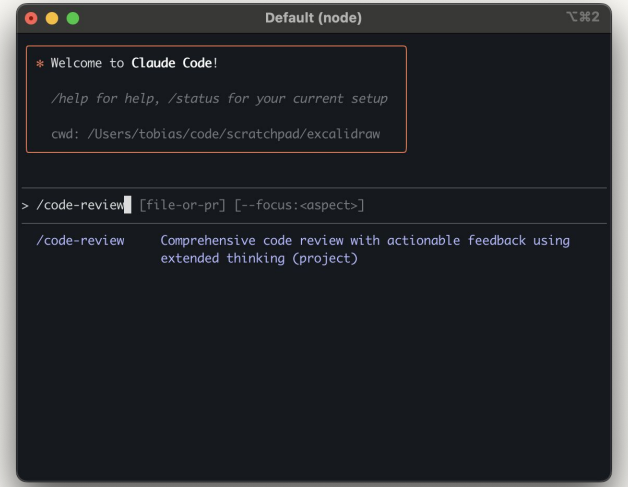
Advanced mode: Have Claude add support for multiple todo lists

Hands-On Exercise

Understanding our codebase with /init

What did we just build? Let's analyze and understand our current to-do list application.

1 `/init`



Advanced mode: "write a README explaining this application: the tech stack, and how to run it locally."

Hands-On Exercise

Build a PRD

1 `/clear`

2 `/model sonnet`

Plan mode on

3 **Write a product requirements document for a todo app. Ask me questions to understand the requirements. Once you're done, save it as PRD.md.**

- Now, work with claude to refine and improve the PRD
  - E.g. "change section XYZ - we don't want X, we want Y"
  - E.g. "add a detailed section describing how we'll measure success."

Accept edits on

4 **Update the code to reflect the PRD** (this will take about 2m30s)

Advanced mode: give Claude an existing Chubb PRD and ask it to use that PRD's format when writing the todo app PRD.

Hands-On Exercise

Build A Command

1. Review this codebase for security vulnerabilities

2. ! mkdir -p .claude/commands/ && echo "Review this codebase for security vulnerabilities:" > .claude/commands/security-review.md

3. /security-review this app

Advanced mode: edit the security-review.md to take $ARGUMENTS

Hands-On Exercise

Building A Subagent

1. **/agents**
   Select 'Create new agent', then choose location 1. Project, then for Creation Method choose: 1. Generate with Claude

2. **A root-cause-analysis-agent that uses the "Five whys" to understand why an error in a log file occurred.**

3. *(optional - can use existing codebase)*: **in a subfolder called code-dummy, generate a dummy codebase with an error and its corresponding log file**

4. **Run root cause analysis on application.log in the code-dummy project** (or as appropriate for your codebase)

Advanced mode: Adjust the toy example above to run your specific root cause analysis / debugging workflow (edit .claude/agents/root-cause-analyzer.md)

Hands-On Exercise

Consensus

Ask claude to check with other experts that all have their stance and own memory through agents

Example prompt:

Come to a consensus between experts on this PR.

1. review PR changes and consider up to 7 experts of different expertise and point of views who could bring unique, highly critical opinion
2. Converse with each expert through an agent and have one also defend the current implementation
3. throughout the process, assemble their opinions and arguments

Final step: propose a recommendation

Advanced Mode
Create a slash command with scope and type of experts as arguments

Hands-On Exercise

Testing a web page with playwright MCP

1. **Make me a web page for my widget factory and use a browser to make sure that there's a cool fireworks effect when I hover over a button**

2. **/plugin install**
   Scroll down and select Playwright. Press ‘i’ to install.

3. Restart Claude Code

4. **Use playwright to test that this widget app works by clicking on the buttons and checking that the fireworks show**

Advanced mode: Setup the Atlassian mcp server and use it to list and complete tickets

Hands-On Exercise

‘The Critic’ Exercise (Challenge pattern)

Validate with adversarial agent whose job it is to find flaws

1 `/clear`

2 **As a senior lead, review the changes and spot latent issues, second and third order unintended implications. Be extremely critical.**

Advanced Mode: move into a Chubb repo you work in, and ask Claude to run the analysis against the last 5 commits

Hands-On Exercise

Consensus

Ask claude to check with other experts that all have their stance and own memory through agents

Example prompt:

Come to a consensus between experts on my work.

1. review my changes and consider up to 7 experts of different expertise and point of views who could bring unique, highly critical opinion

2. Converse with each expert through an agent and have one also defend the current implementation

3. throughout the process, assemble their opinions and arguments

Final step: propose a recommendation

Advanced Mode
Create a slash command with scope and type of experts as arguments

Hands-On Exercise

Verification Loop

Test-driven grounding: Tests define truth, code must pass

Example prompt:

- **With an agent: run {TOOL} and fix any newly introduced issues that were not yet present on main**

  Where {tool} can be biome for typescript or error prone for java or ruff for python

Advanced Mode
Leverage tools to automatically add the verification after code generation

Hands-On Exercise

Security Review

Ensure that you're building with a Security first mindset

Example prompts:

- **Dedicated security agent: Check for injection vulnerabilities, auth bypasses, data exposure, dependency risks**
- Pattern: Security review as mandatory gate before merge
- Integration with **/hooks**: Auto-trigger on sensitive file changes

Advanced Mode
Anthropic released a AI powered github action to conduct reviews and propose solutions
https://github.com/anthropics/claude-code-security-review