

Where the harness files built in the course go

Location	What to write	Built in the course	Caution
.github/copilot-instructions.md	Instructions on how to write, read in every conversation	Created on Day 1. On Day 2, the line "Do not split the file" is rewritten as 2 lines	About 20 lines. Fix it when the assumptions change
AGENTS.md (directly under tpro-work)	Project premise: structure, how to run it, what you may touch	Created in the Day 1 extra exercise	Write facts only. Do not copy content that is the same as copilot-instructions.md
.github/instructions/tests.instructions.md	Rules that apply only when you work with files matching applyTo	Day 3 Exercise 8. Take the expected values from testcases.md	Without applyTo: tests/** at the top, it is read in all conversations
.github/skills/test-design/SKILL.md	Steps for test case design. Column definitions are in references/columns.md	Built in Day 3 Exercise 6 and improved by 1 line in Exercise 9	Use the same lowercase English letters for name and the folder name
.github/skills/regress/SKILL.md	Puts the pasted result next to the previous one and returns only the lines that changed	Day 3 Exercise 7	A skill does not open the browser. Paste the result first, then call it
.github/agents/security-reviewer.agent.md	A read-only reviewer. Limit tools to read-only ones	Day 3 Exercise 6. Used in the practical exam to review the part for the new format	Copy model exactly as the picker shows it, without changing even one character

Official commands (the ones used in the course)

/create-skill Extracts steps into a skill. It can also build one from the flow of a conversation

/create-instructions Creates instructions that apply only to specific files (applyTo)

Chat: New Custom Agent From the Command Palette, creates an agent narrowed to a single role

/clear For when you move to different work. Stops the earlier conversation from carrying over

/compact When the conversation gets long and the premises start to be forgotten, condense it to the key points and continue

/init Drafts the overall instructions file from the contents of the folder

/skills, /agents Check the list of installed skills and agents

/troubleshoot When a skill or agent is not being read, have it read the log

Ask, Agent and Plan

Mode	File	When to use
Ask	Unchanged	Ask about contents, have it explain, compare options
Agent	Not final until you Keep	Create, fix, add. Undo if it is wrong
Plan	Unchanged	Check the plan before a big change

If a CSV in a changed format arrives next month

- 1 Import** Open index.html and drop the CSV onto the app. Check the count in the warning area
- 2 Unit** Open tests.html and copy the SUMMARY line
SUMMARY PASS=n FAIL=n HOLD=n ...
- 3 Full run** Open index.html?e2e and check the E2E result
- 4 Compare** Paste the SUMMARY line and the E2E result, then run /regress
/regress

- 5 Redesign** When the format changes, give /test-design the specification to get new cases, then fix worklog-core.js and worklog-config.js
/test-design #new-spec.md
- 6 Review** Have security-reviewer check the part you added
Run security-reviewer as a subagent and have it review the function I added.

Call patterns

- Build from the conversation** Lock in the flow that worked, as it is
/create-skill Make a skill from the flow so far. Name it regress
- Create instructions** Decide the scope first
/create-instructions Instructions that apply only when working with files in the tests folder
- Return side by side** Have it return the result in a section separate from the parent's reply
Run security-reviewer as a subagent and return the result in its own section.

Tips and cautions

If Agent rewrites the test files to match the implementation, do not Keep that diff. An implementation that differs from the specification remains PASS.

Take expected values from testcases.md and the specification, and do not copy them from the contents of worklog-core.js. If you write this in the instructions for tests, it still applies when you forget to say it.

A .prompt.md file that saves a single prompt has been announced to move to SKILL.md. Build a skill even for a single prompt.

A subagent does not run on a model above its parent. Make the parent the higher model and the reviewer the lighter one.

Before you import a skill from outside, read all of SKILL.md and the scripts. Skills with scripts often need Python or Node.

Sources (official and primary information / click to open)

1. [VS Code built-in commands](#)
2. [VS Code Custom instructions](#)
3. [VS Code Agent Skills](#)
4. [VS Code Custom agents](#)
5. [VS Code Tools reference](#)
6. [vscode-copilot-chat /create-*](#)