Build and use a planning canvas

Session M7-04 | Product and Delivery Teams | Intermediate

Use visible state when chat stops being enough

A canvas helps you review several records and make repeated updates through named actions. Keep that view available after the conversation ends.

The source is the live Session M7-03 parent issue and its linked child issues, not the previous chat.

Keep GitHub state and planning information distinct

GitHub records issue state. The canvas groups that state with planning information that helps people decide what to do next.

GitHub fact	Canvas planning view
Open issue	Ready for discovery
Assigned issue	Ownership confirmed
Blocked issue	Risk, dependency, and next decision
Closed issue	Planning result recorded

Label the two kinds of state differently. Otherwise the team cannot tell whether they are seeing a decision or a source-system fact.

Keep fields that change a decision

Field	Keep it when	Remove it when
Risk	It changes sequencing or approval	It only repeats the issue body
Owner	The responsible person is unclear	GitHub already makes it clear
Next action	A team needs a follow-up	It is only a status synonym

The canvas should expose important work without becoming a second backlog.

Review canvas updates

```
Observe source state
      ↓
Propose canvas update
      ↓
Review the affected record
      ↓
Accept, revise, or reject
```

Use this loop to catch stale data and updates that lack supporting sources.

Start from live GitHub issues

The canvas should show:

- parent objective;
- issue and planning status;
- owner or ownership gap;
- dependency and risk;
- decision and next action.

Create it in plain language

```
/create-canvas Create a delivery-planning canvas for the approved GitHub issues.
People can update planning status and record decisions. Copilot can summarize
and propose updates. Do not close issues, change assignees, or publish content.
```

This is a no-code review. Inspect generated files and capabilities, then ask Copilot for specific changes.

Review before use

Ask:

1. Does each field support a real decision?
2. Can every action be explained?
3. Are external writes excluded?
4. Can the owner recover?
5. When should the canvas be retired?

Simpler is better

Remove generated fields and actions that do not help the team decide or act.

Load live GitHub issues

Use GitHub MCP to retrieve the Session M7-03 issues.

Copilot can move the approved fields into the canvas. GitHub remains the source for issue state.

Label planning state separately

The canvas may show `ready for review` while the GitHub issue remains open.

That is acceptable when the canvas labels the planning state and source-system state separately.

Check three views

Compare:

- the visible canvas;
- Copilot's report;
- a fresh GitHub MCP read.

Resolve or explain every mismatch.

Create visible drift

1. Preview and approve a GitHub issue comment through MCP.
2. Retrieve the changed issue in a fresh request.
3. Do not refresh the canvas.
4. Compare the new GitHub state with the visible canvas.

The canvas should show that its last refresh is older than the source change.

Refresh the canvas

Use the approved refresh capability.

Check:

- the blocker appears in planning state;
- the next action names the owner decision;
- GitHub state and planning state stay distinct;
- the last-refresh time changes.

Critique the live state

Run `/context`, then use a fresh session when needed.

Ask `/rubber-duck` to find stale claims, mixed state labels, unsafe actions, and missing owners.

The live canvas and linked issues are authoritative for Session M7-05.

Lab result

- Learner-created canvas
- Reviewed state and actions
- Live GitHub issue data
- A detected and reconciled state mismatch
- Owner, refresh, recovery, and retirement rules
- Four linked status-visibility workstream candidates

Stop if live canvas creation is unavailable.