One issue, one review decision

Keep the Copilot cloud agent within the issue

Issue-to-review workflow

```
issue contract
    ↓
repository setup
    ↓
proposed change
    ↓
test evidence
    ↓
human decision
```

Keep the same issue and acceptance criteria at every step.

Access policy

The lab requires:

- approved Copilot cloud agent access;
- a training repository with synthetic data;
- a named reviewer;
- an issue with a file scope and stop condition.

If any requirement is missing, stop the lab. Do not bypass policy.

The fixed issue

```
Reject blank task titles
```

Allowed files:

- `src/app.js`
- `tests/app.test.js`

Required command:

```
npm test
```

No dependency changes. No unrelated refactor.

Checkpoint 1: Issue contract

The reviewer must answer:

- Which behavior changes?
- Which behavior stays the same?
- Which files may change?
- What proves completion?
- What is out of scope?

Do not assign the issue until every answer is clear.

Repository instructions

Tell the agent how to work in this repository:

- project shape;
- coding conventions;
- allowed file scope;
- test command;
- prohibited changes.

Instructions support the issue. They do not repair a vague issue.

Copilot setup steps

The workflow file belongs at:

```
.github/workflows/copilot-setup-steps.yml
```

It uses one job named:

```
copilot-setup-steps
```

Review permissions, commands, and secrets before merging it to the default branch.

Checkpoint 2: Setup

Record:

- baseline test result;
- repository instructions location;
- setup workflow location;
- allowed files;
- reviewer and stop condition.

Stop when you cannot explain the baseline test result or setup requires unapproved access.

Start the session

- assign the fixed issue through an entry point allowed by repository policy;
- watch the session log;
- steer only within the issue;
- stop on scope drift.

Checkpoint 3: Proposed change

Inspect before running tests:

- changed files;
- validation behavior;
- response shape;
- valid request behavior;
- dependency and configuration changes.

A small diff can still be wrong.

The expected behavior

```
if (typeof title !== "string" || title.trim() === "") {
  return res.status(400).json({ error: "title is required" });
}
```

The code may differ, but it must preserve the required behavior and constraints.

Checkpoint 4: Tests

Required cases:

Case	Expected
`""`	HTTP 400
`"   "`	HTTP 400
`" Weekly plan "`	HTTP 201 with preserved response shape

Run the full supplied suite and record the command output.

Keep review feedback within the issue

Good:

```
The whitespace case is missing. Add a focused test in tests/app.test.js.
Keep the current response shape and do not change other endpoints.
Run npm test and report the result.
```

Bad:

```
Clean up the API while you are there.
```

Checkpoint 5: Human decision

Choose:

- **approve** when criteria, scope, and tests pass;
- **request changes** when a correction within the issue remains;
- **pause** when access, test output, or ownership is unresolved.

The agent does not make this decision.

Stop conditions

Stop the lab when:

- the requested files expand;
- a new dependency appears;
- credentials or unapproved data are requested;
- repeated test failures have no recovery plan;
- the issue changes during implementation;
- the reviewer is no longer available.

Trainer demo recovery

The solution folder contains:

- the final issue contract;
- reviewed setup workflow;
- proposed diff;
- runnable solution project;
- test output;
- completed human decision.

Use it only when the trainer demo is delayed. Learners still need a live cloud-agent run for lab completion.

Lab handoff

Keep the same issue through all five checkpoints. Record the reviewer's decision.