

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

CSC4801
AI-assisted Software Engineering

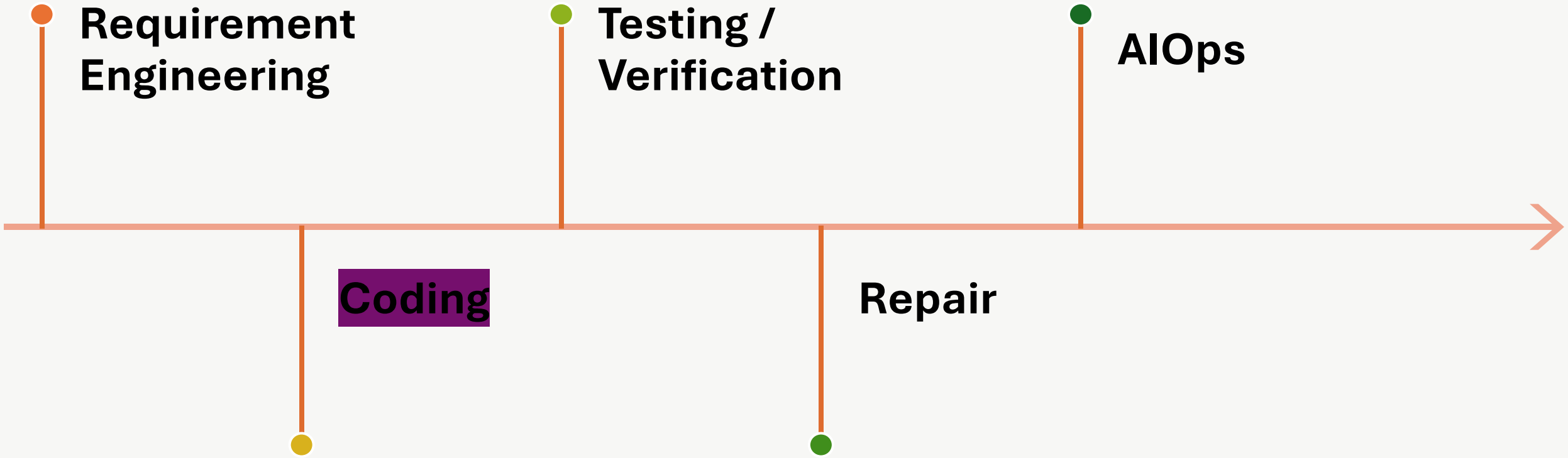
Lecture 10: Team Collaboration

Instructor: Jinsheng Ba | 2026 Fall

Quiz I

- 10/22 Thursday
- 8:30-9:00 Lecture
- 9:00-10:00 Quiz
  - BRING your own pens
  - NO electric devices
  - NO cheat sheet

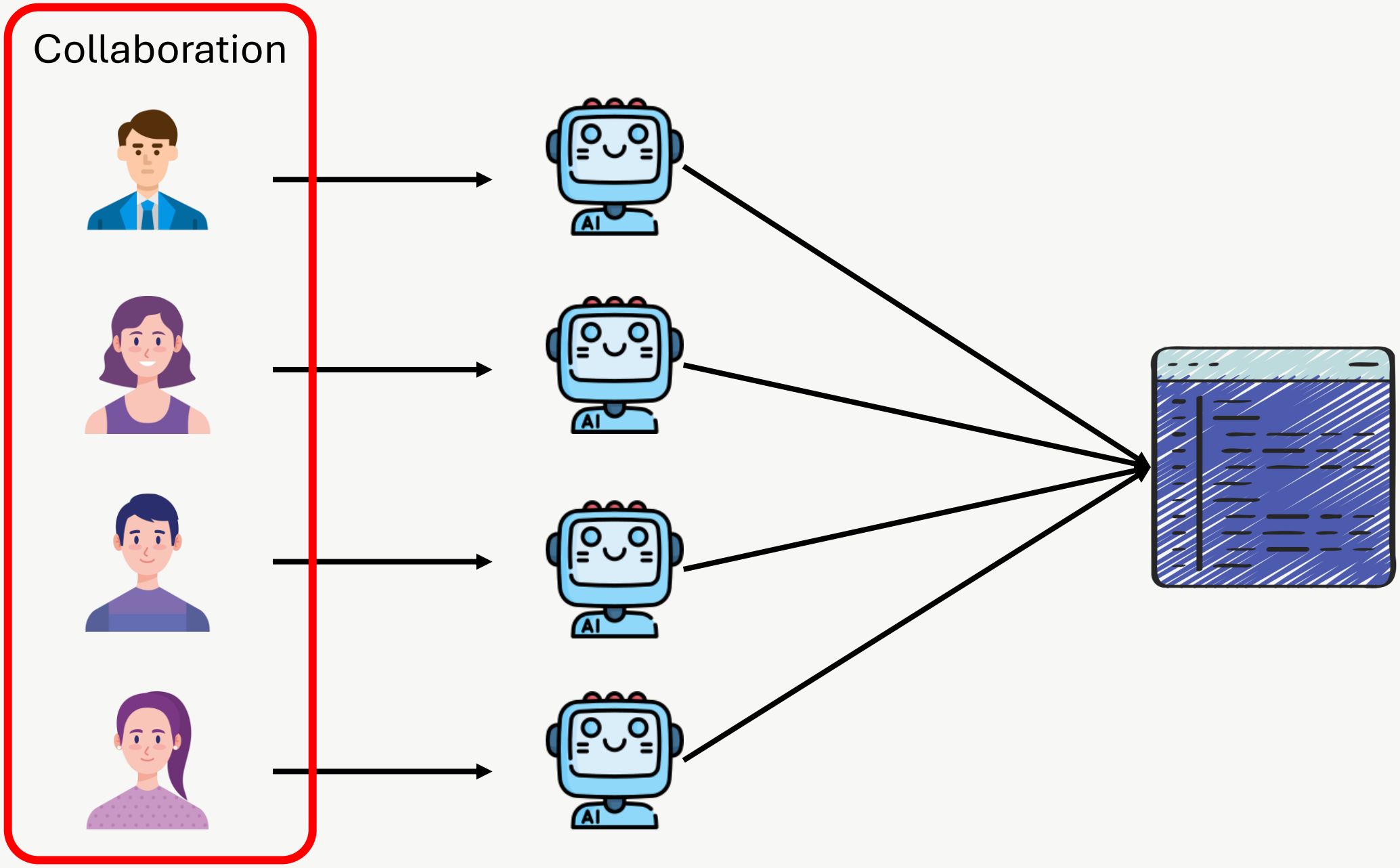
AI-Assisted Software Engineering

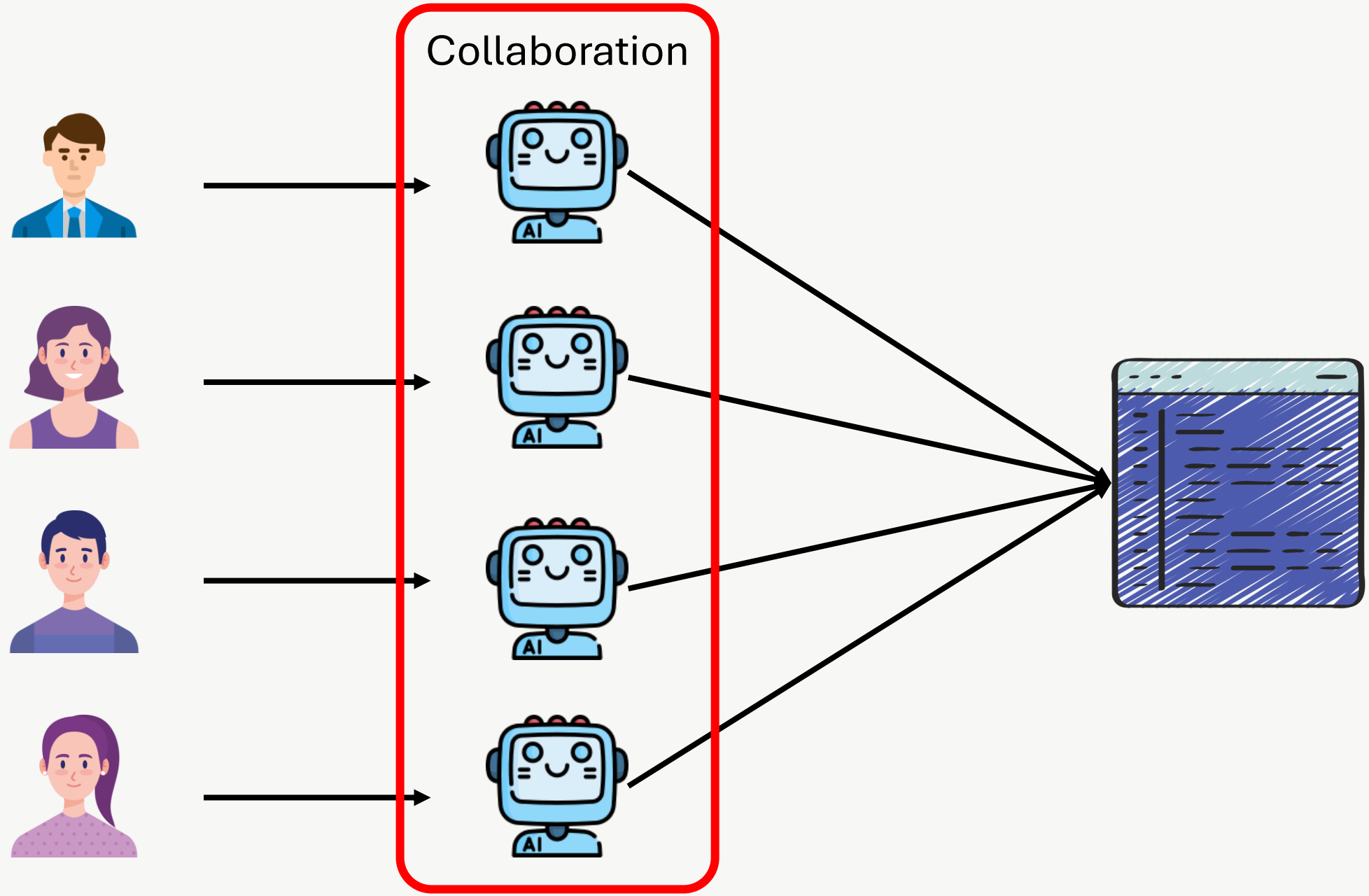


We have introduced a lot of techniques to write code.

How to work in a team?



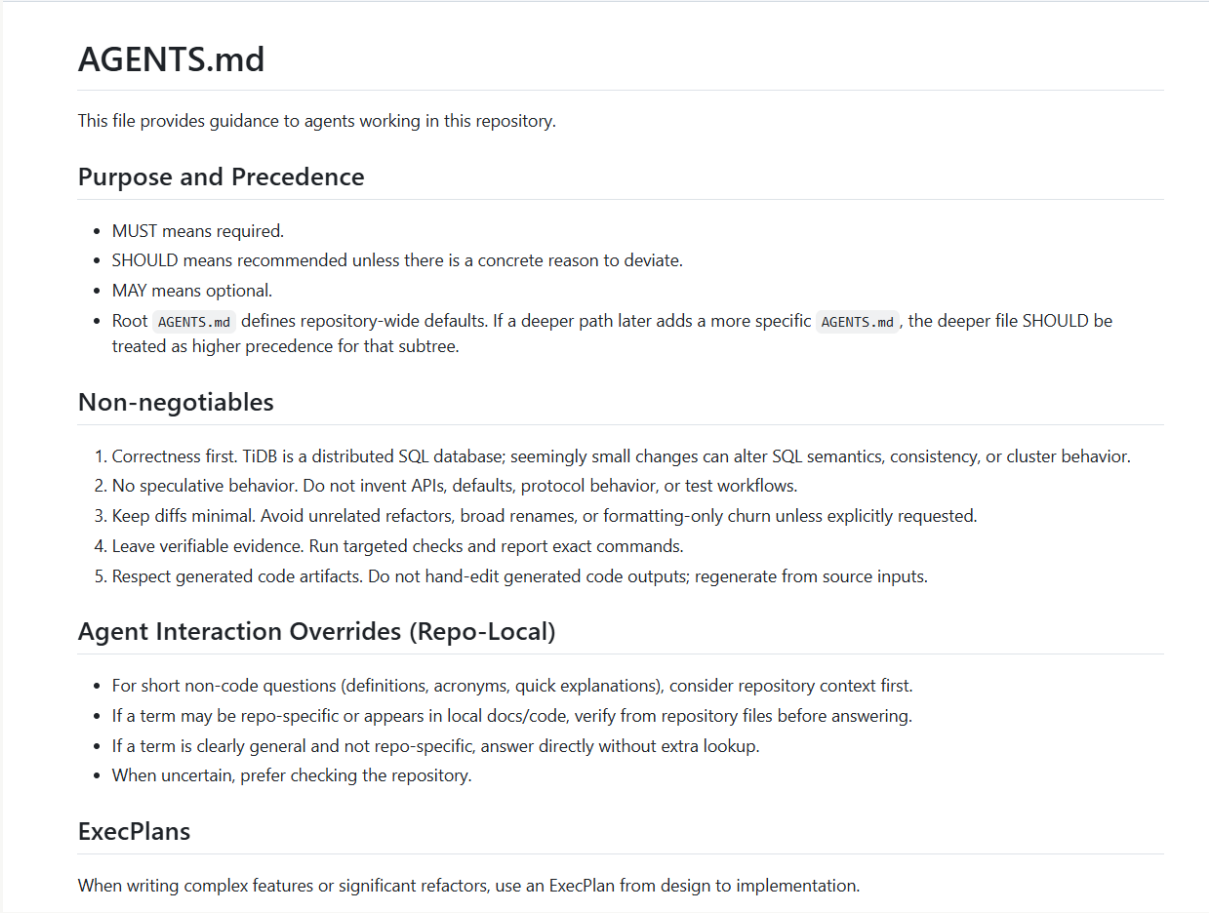

Collaboration
AI
AI
AI
AI


Collaboration
AI
AI
AI
AI

Agent Sharing

Shared Memory

Agents.md is project-specific memory file in the root of the project (If you use claude code or similar agents).



AGENTS.md

This file provides guidance to agents working in this repository.

Purpose and Precedence

- MUST means required.
- SHOULD means recommended unless there is a concrete reason to deviate.
- MAY means optional.
- Root `AGENTS.md` defines repository-wide defaults. If a deeper path later adds a more specific `AGENTS.md`, the deeper file SHOULD be treated as higher precedence for that subtree.

Non-negotiables

1. Correctness first. TiDB is a distributed SQL database; seemingly small changes can alter SQL semantics, consistency, or cluster behavior.
2. No speculative behavior. Do not invent APIs, defaults, protocol behavior, or test workflows.
3. Keep diffs minimal. Avoid unrelated refactors, broad renames, or formatting-only churn unless explicitly requested.
4. Leave verifiable evidence. Run targeted checks and report exact commands.
5. Respect generated code artifacts. Do not hand-edit generated code outputs; regenerate from source inputs.

Agent Interaction Overrides (Repo-Local)

- For short non-code questions (definitions, acronyms, quick explanations), consider repository context first.
- If a term may be repo-specific or appears in local docs/code, verify from repository files before answering.
- If a term is clearly general and not repo-specific, answer directly without extra lookup.
- When uncertain, prefer checking the repository.

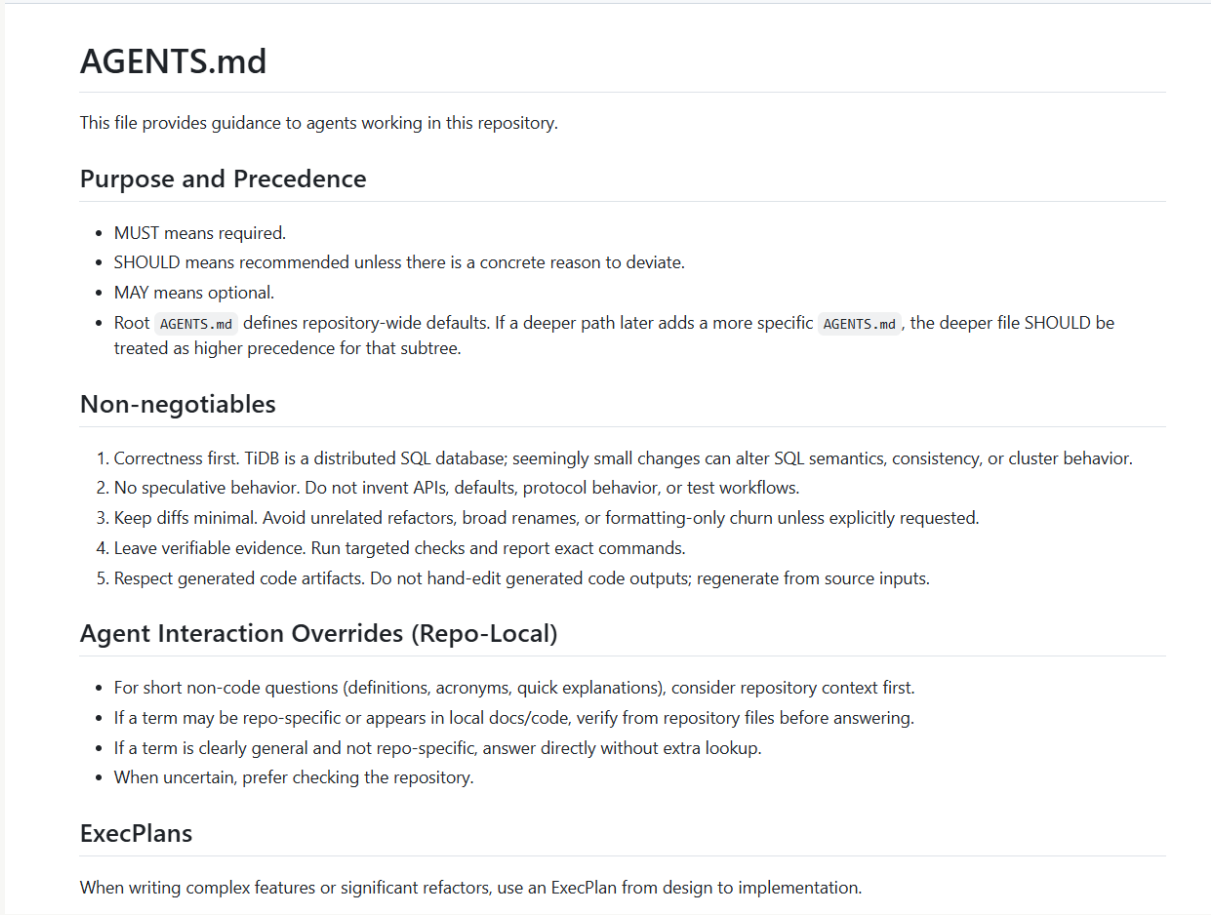
ExecPlans

When writing complex features or significant refactors, use an ExecPlan from design to implementation.

https://github.com/pingcap/tidb/blob/master/AGENTS.md

Shared Memory

In your team, you could write your code style, interface specifications, and so on.



AGENTS.md

This file provides guidance to agents working in this repository.

Purpose and Precedence

- MUST means required.
- SHOULD means recommended unless there is a concrete reason to deviate.
- MAY means optional.
- Root `AGENTS.md` defines repository-wide defaults. If a deeper path later adds a more specific `AGENTS.md`, the deeper file SHOULD be treated as higher precedence for that subtree.

Non-negotiables

1. Correctness first. TiDB is a distributed SQL database; seemingly small changes can alter SQL semantics, consistency, or cluster behavior.
2. No speculative behavior. Do not invent APIs, defaults, protocol behavior, or test workflows.
3. Keep diffs minimal. Avoid unrelated refactors, broad renames, or formatting-only churn unless explicitly requested.
4. Leave verifiable evidence. Run targeted checks and report exact commands.
5. Respect generated code artifacts. Do not hand-edit generated code outputs; regenerate from source inputs.

Agent Interaction Overrides (Repo-Local)

- For short non-code questions (definitions, acronyms, quick explanations), consider repository context first.
- If a term may be repo-specific or appears in local docs/code, verify from repository files before answering.
- If a term is clearly general and not repo-specific, answer directly without extra lookup.
- When uncertain, prefer checking the repository.

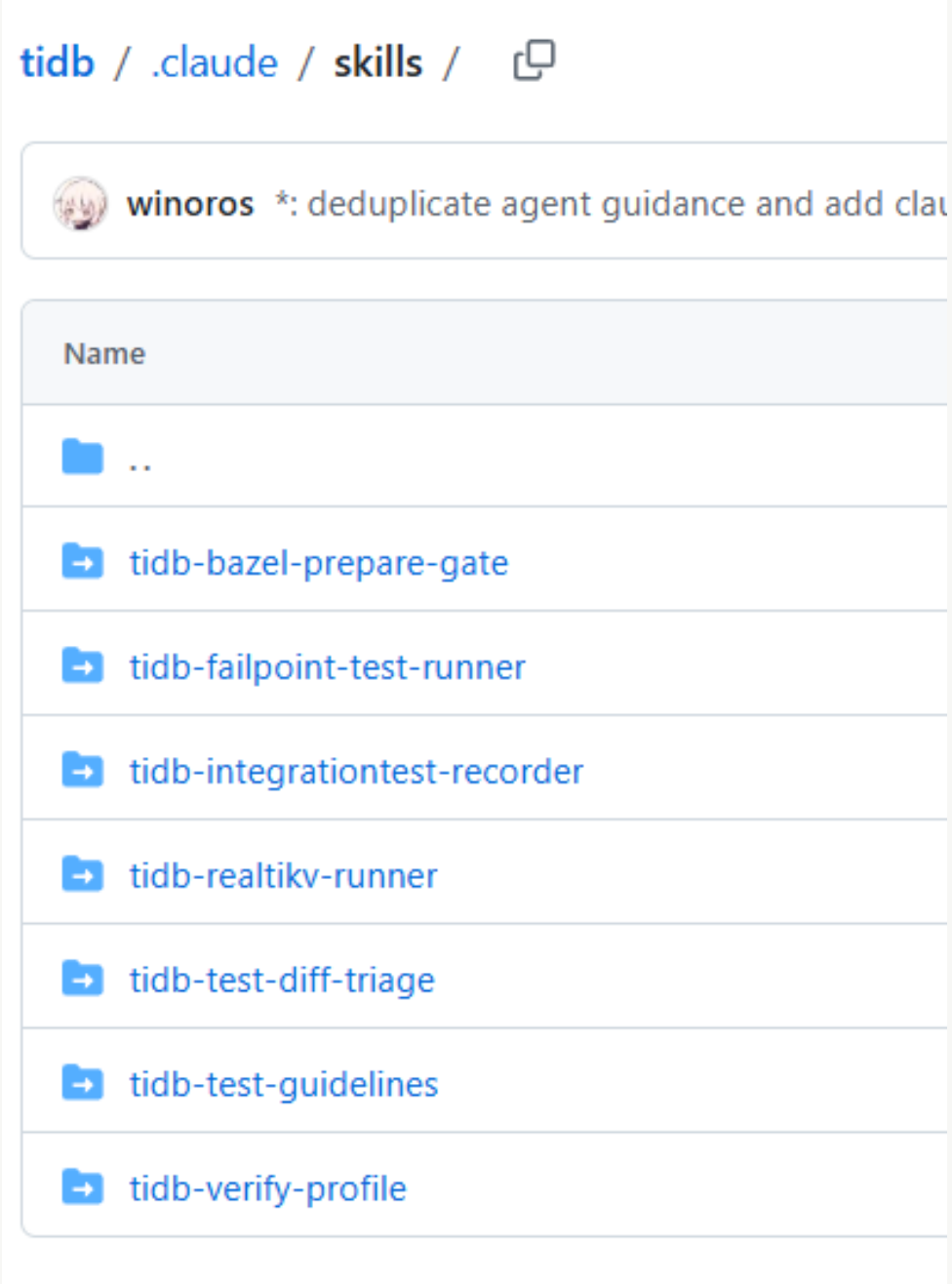
ExecPlans

When writing complex features or significant refactors, use an ExecPlan from design to implementation.

https://github.com/pingcap/tidb/blob/master/AGENTS.md

Shared Skills

You could also put project-specific skills here.



https://github.com/pingcap/tidb/tree/master/.claude/skills

Version Control

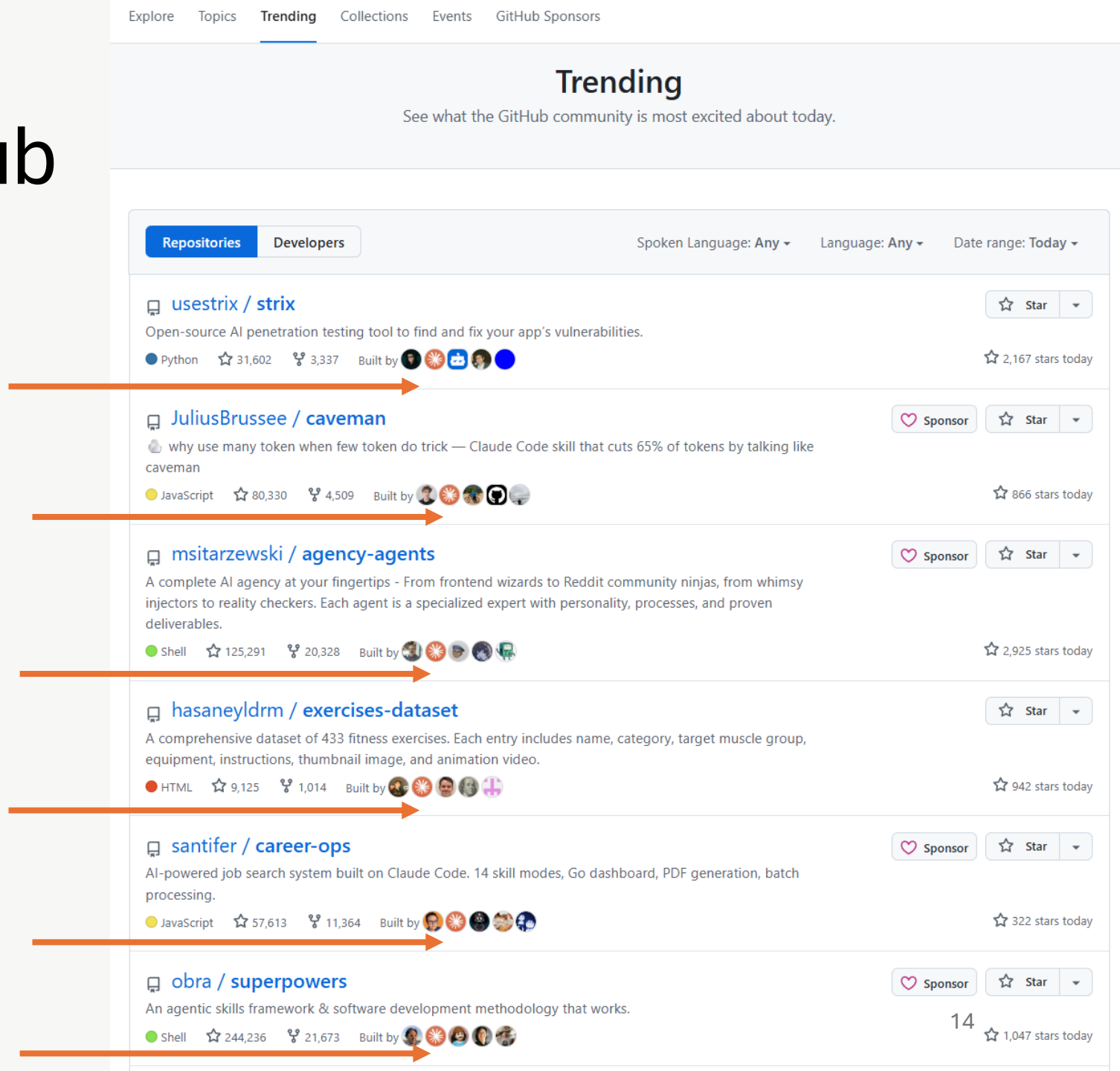
Git

- Git is necessary in using LLMs for SE.
- Coding agents frequently submit new commits.



Projects On GitHub

https://github.com/trending



Projects On GitHub

Code agents are co-authors of commits



https://github.com/usestrix/strix/commit/f342808d2b4af551a17e47af744d2f19dee3c443

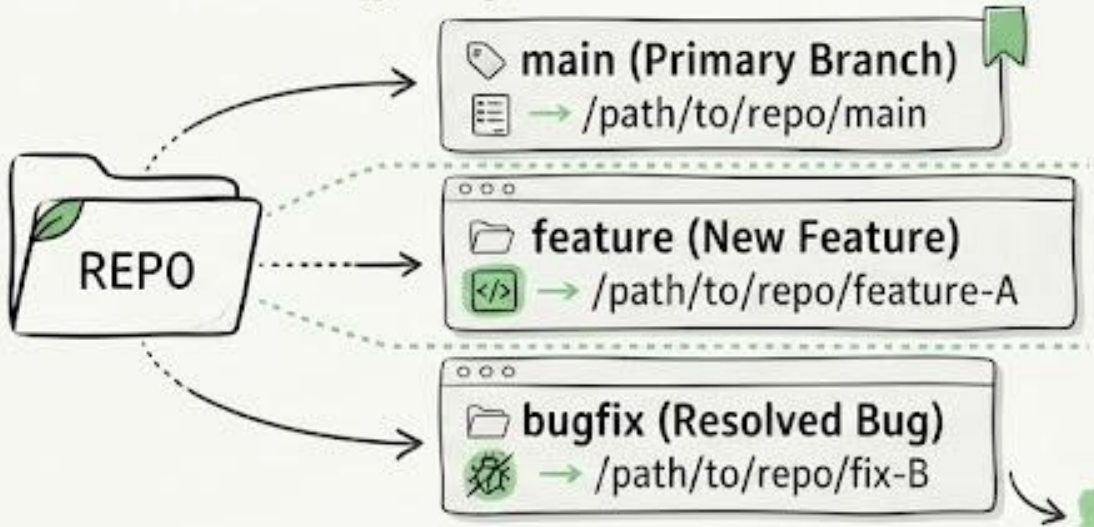
Parallel

Git Worktree

High Efficiency | Parallel Development

『git worktree: Transforming 'Branch Switching' into 'Isolated Workspaces'』

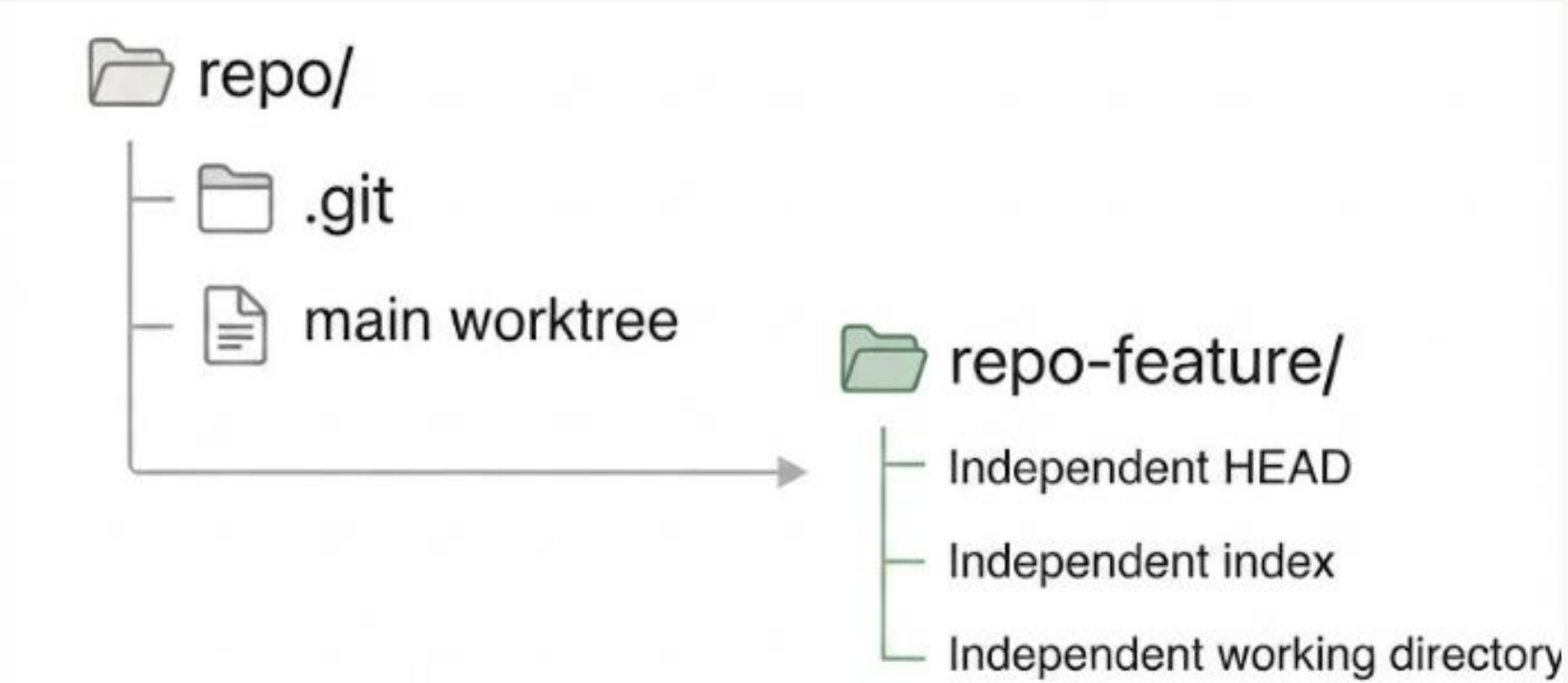
Especially suitable for Vibe Coding / Agentic Parallel Development




```
$ git worktree add <new-branch-name> <path>
```

Quick Context Switching, Zero Interference

Say Goodbye to Repetitive Stash/Checkout



- ✅ Shared object database
- ✅ Shared refs
- ✅ Isolated current file state

```
git worktree add -b fix ../worktree-fix main
```

Directly Using Worktrees in Agents

Start Claude in a worktree

Pass `--worktree` or `-w` to create an isolated worktree and start Claude in it. By default, the worktree is created under `.claude/worktrees/<value>/` at your repository root, on a new branch named `worktree-<value>`:

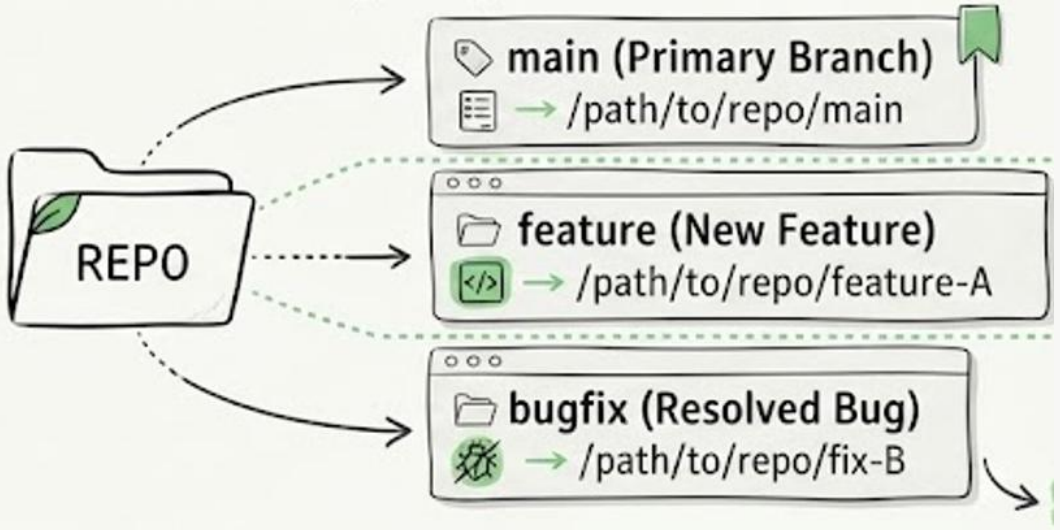
```
claude --worktree feature-auth
```

To put worktrees somewhere else, configure a `WorktreeCreate` **hook**. Run the command again with a different name in another terminal to start a second isolated session:

```
claude --worktree bugfix-123
```

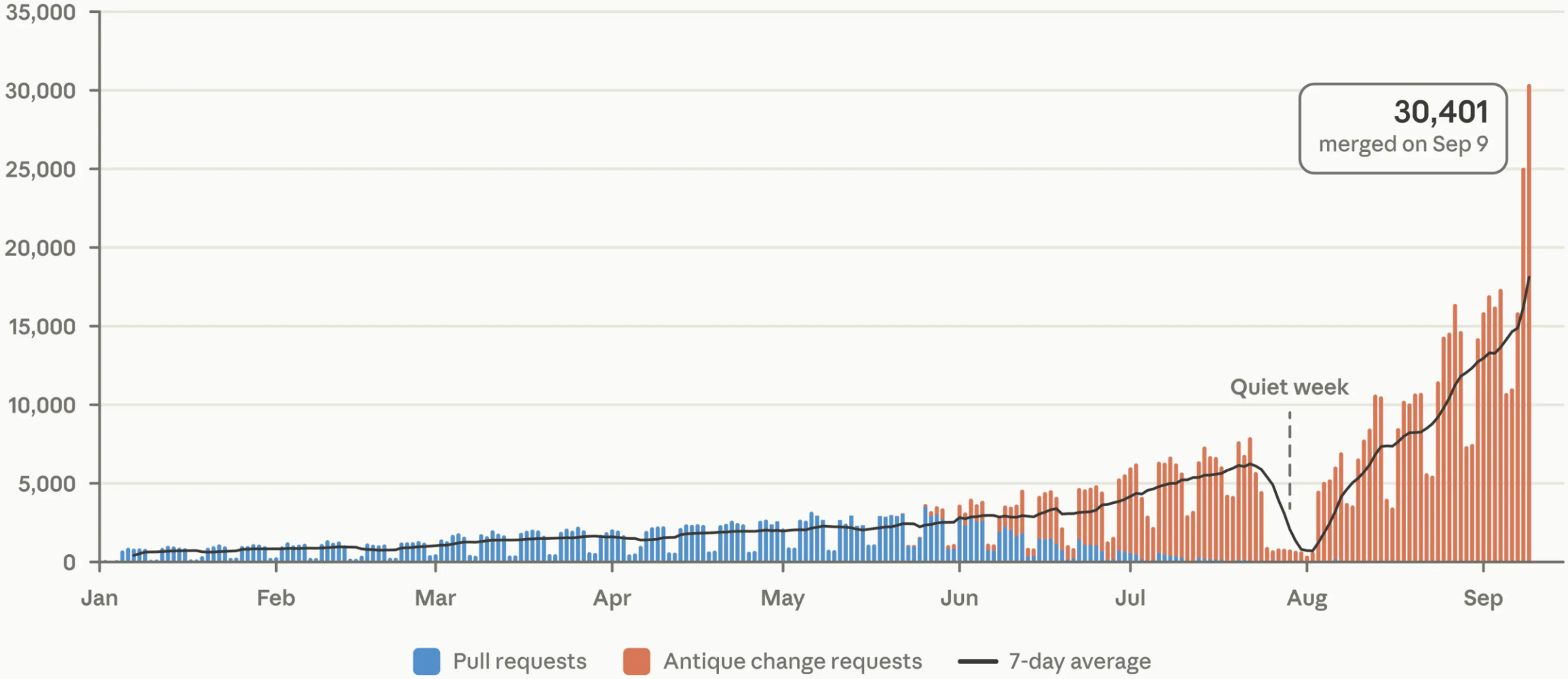
https://code.claude.com/docs/en/worktrees

While your agent may submit **thousands** of commits in an hour



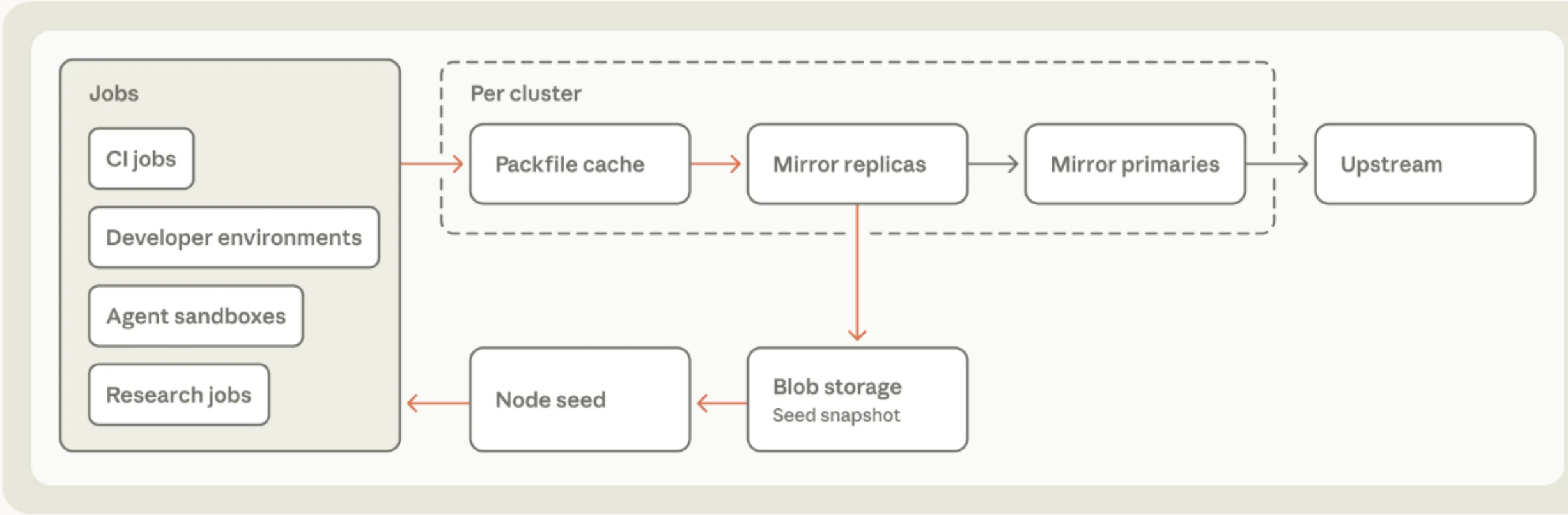
Changes merged per day

Anthropic monorepo, January–September 2026



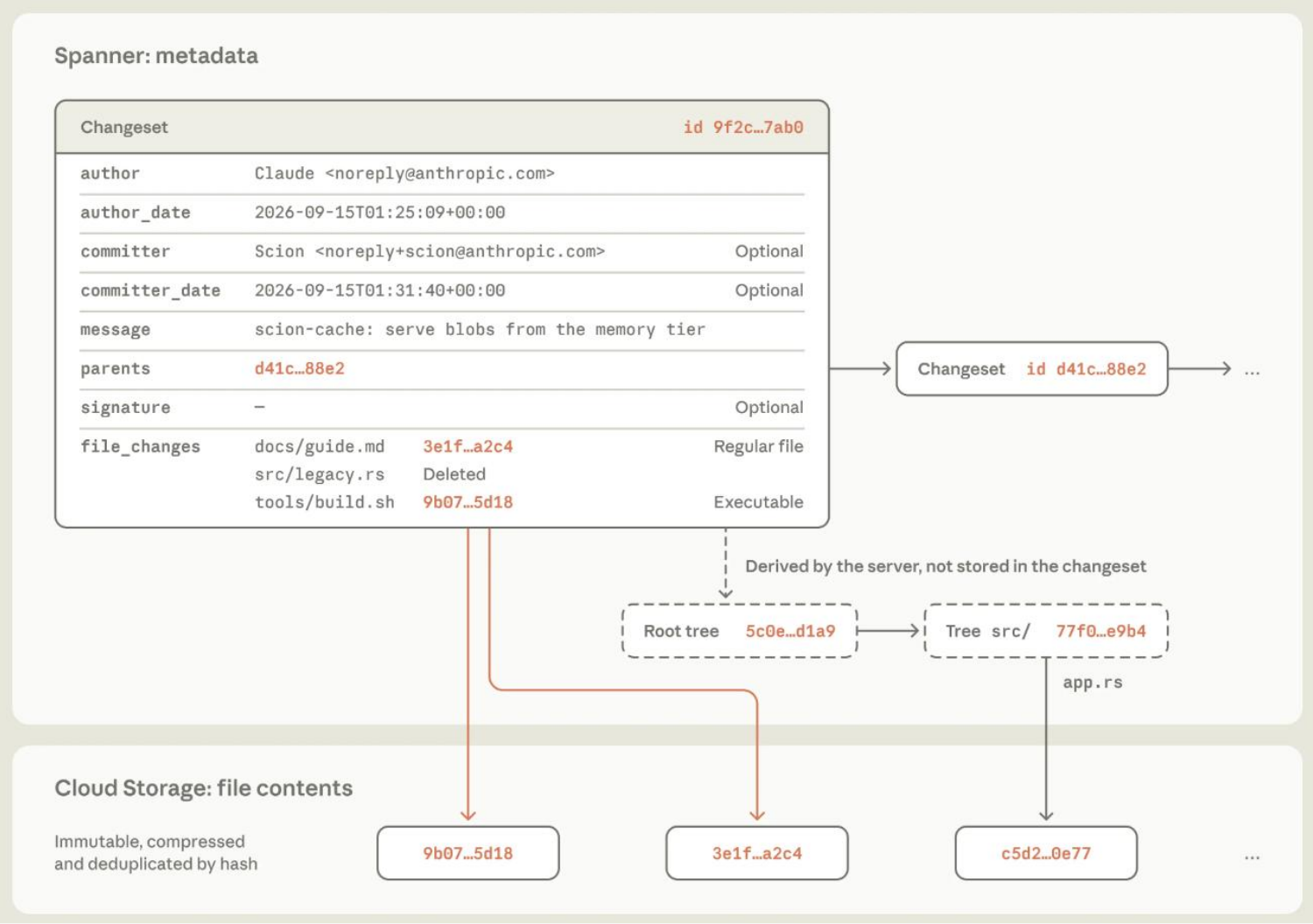
https://www.anthropic.com/engineering/redesigning-source-control

Performance Bottleneck

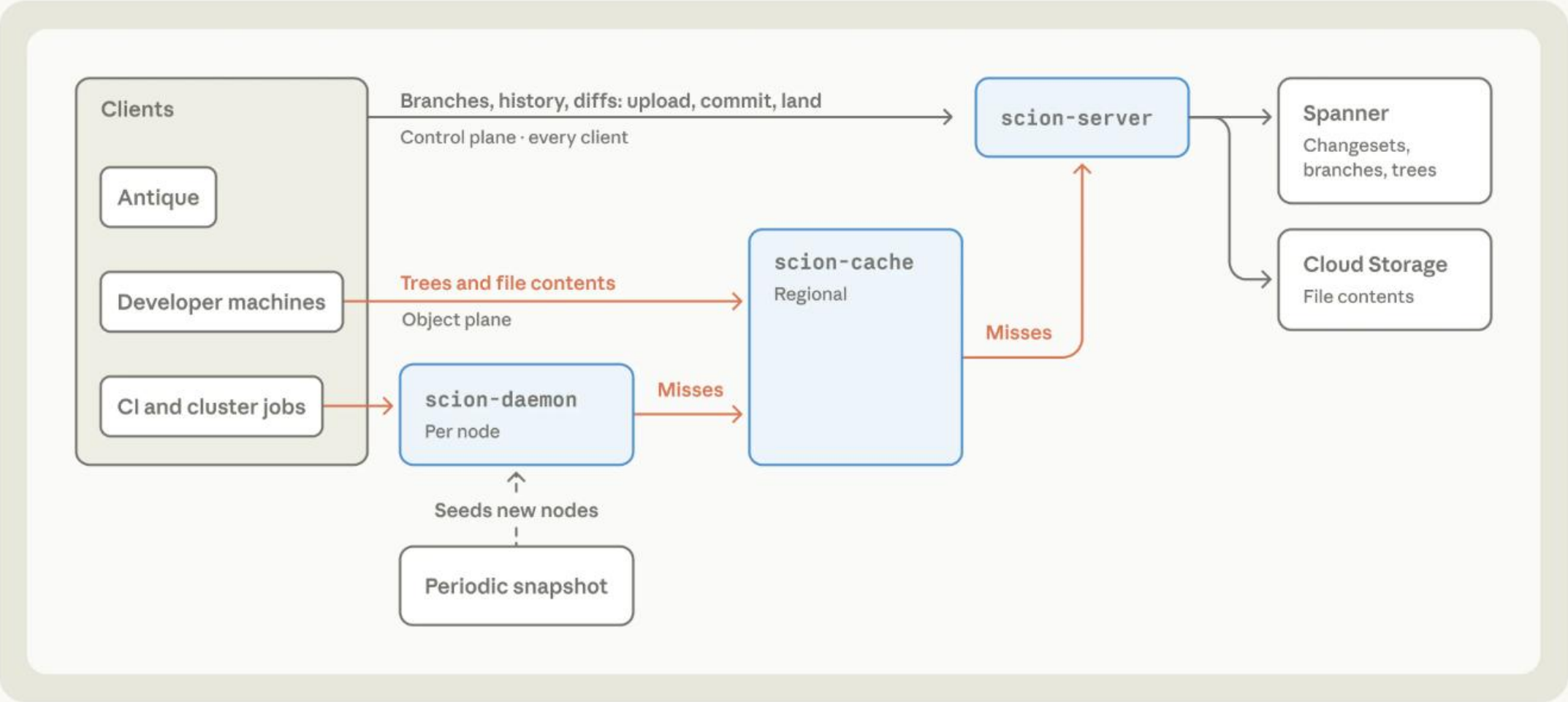


The typical setup for scaling git checkouts.

What to Git Push

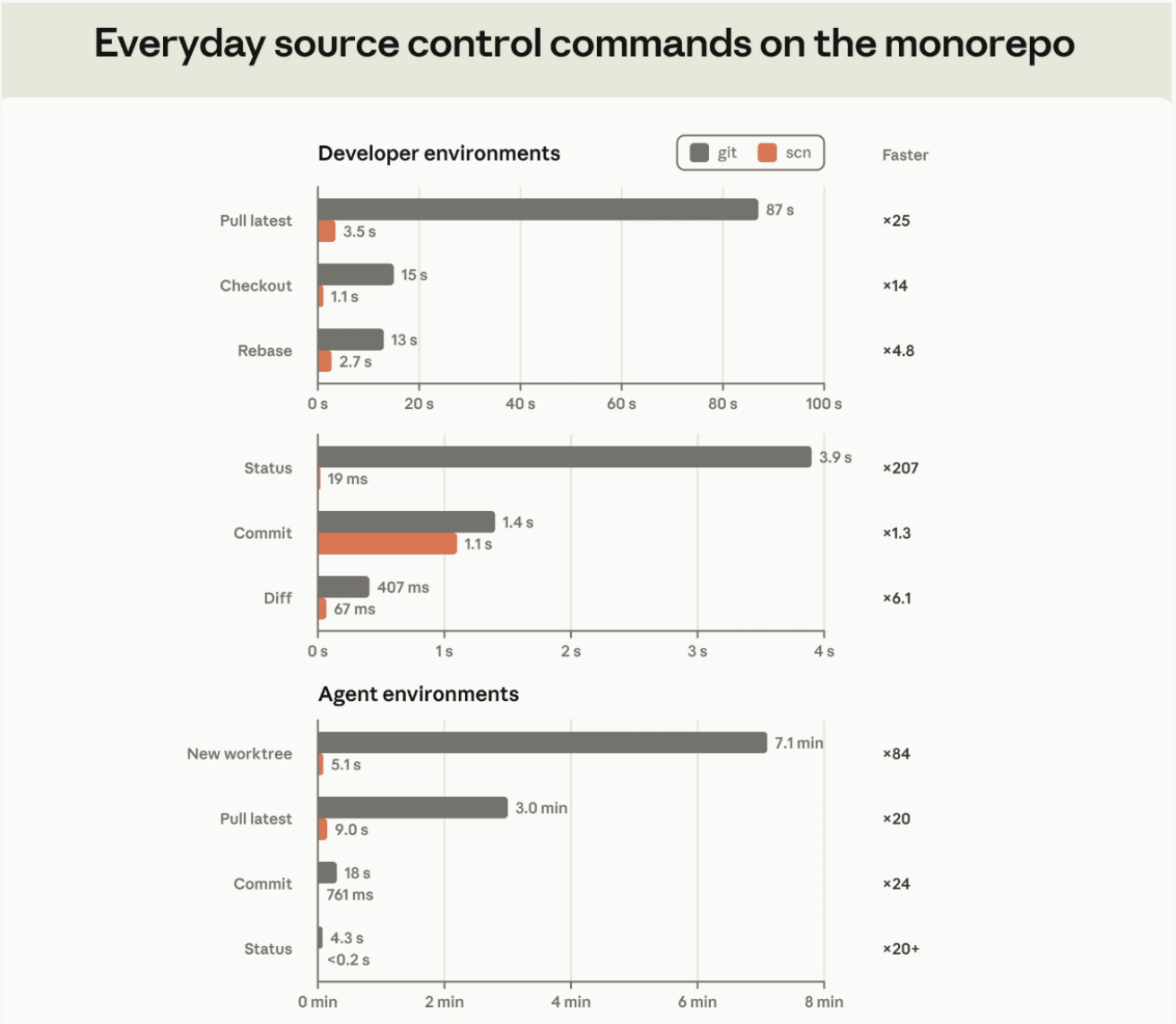


Checkout Architecture

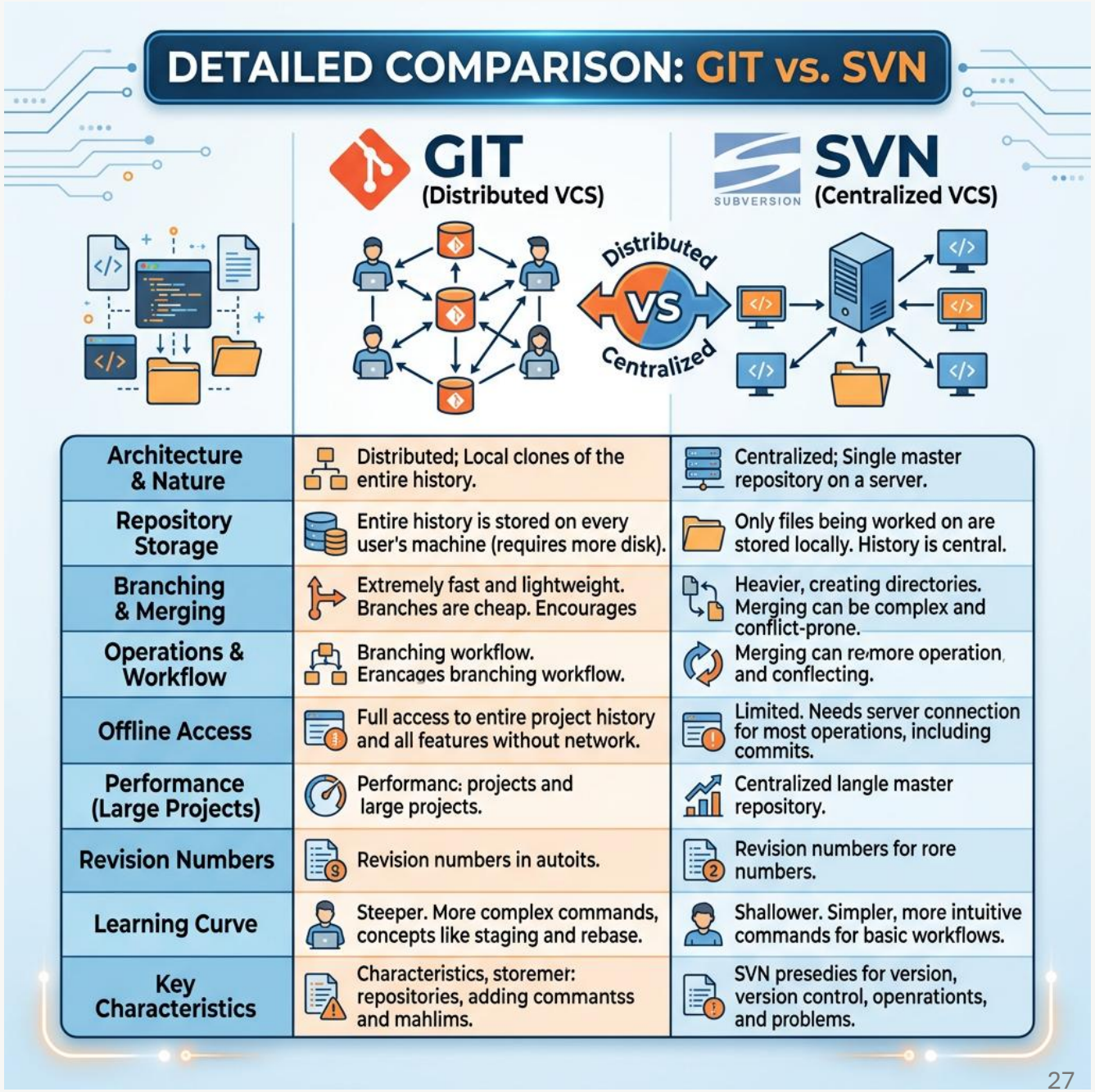


Scion's checkout architecture.

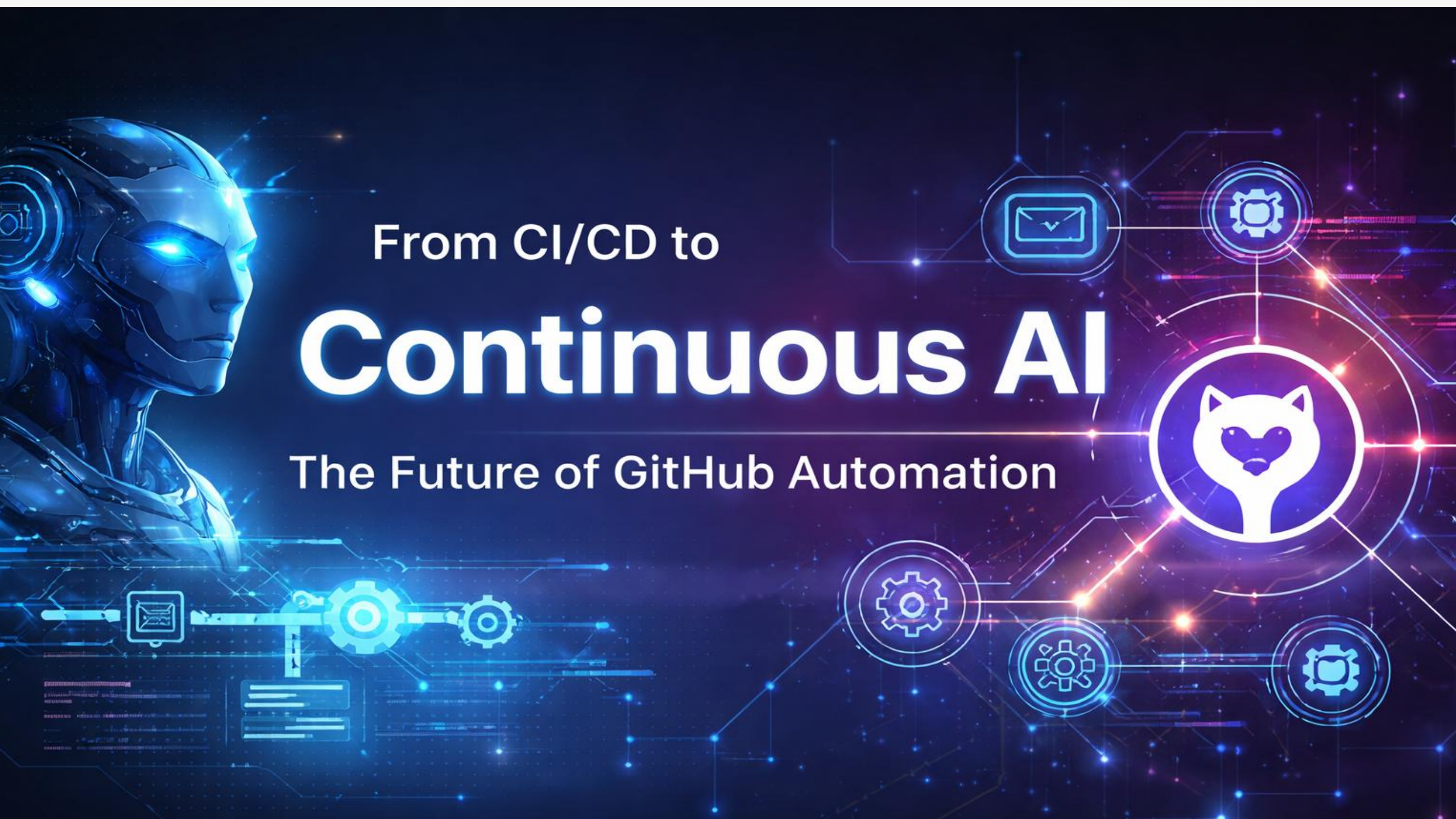
Results


Everyday source control commands on the monorepo
Developer environments
git
scn
Faster
Pull latest
87 s
3.5 s
×25
Checkout
15 s
1.1 s
×14
Rebase
13 s
2.7 s
×4.8
0 s
20 s
40 s
60 s
80 s
100 s
Status
3.9 s
19 ms
×207
Commit
1.4 s
1.1 s
×1.3
Diff
407 ms
67 ms
×6.1
0 s
1 s
2 s
3 s
4 s
Agent environments
New worktree
7.1 min
5.1 s
×84
Pull latest
3.0 min
9.0 s
×20
Commit
18 s
761 ms
×24
Status
4.3 s
<0.2 s
×20+
0 min
2 min
4 min
6 min
8 min

Discussion



Architecture & Nature	Distributed; Local clones of the entire history.	Centralized; Single master repository on a server.
Repository Storage	Entire history is stored on every user's machine (requires more disk).	Only files being worked on are stored locally. History is central.
Branching & Merging	Extremely fast and lightweight. Branches are cheap. Encourages	Heavier, creating directories. Merging can be complex and conflict-prone.
Operations & Workflow	Branching workflow. Erancages branching workflow.	Merging can re·more operation, and conflecting.
Offline Access	Full access to entire project history and all features without network.	Limited. Needs server connection for most operations, including commits.
Performance (Large Projects)	Performanc: projects and large projects.	Centralized langle master repository.
Revision Numbers	Revision numbers in autoits.	Revision numbers for rore numbers.
Learning Curve	Steeper. More complex commands, concepts like staging and rebase.	Shallower. Simpler, more intuitive commands for basic workflows.
Key Characteristics	Characteristics, storemer: repositories, adding commantss and mahlims.	SVN presedies for version, version control, openrationts, and problems.

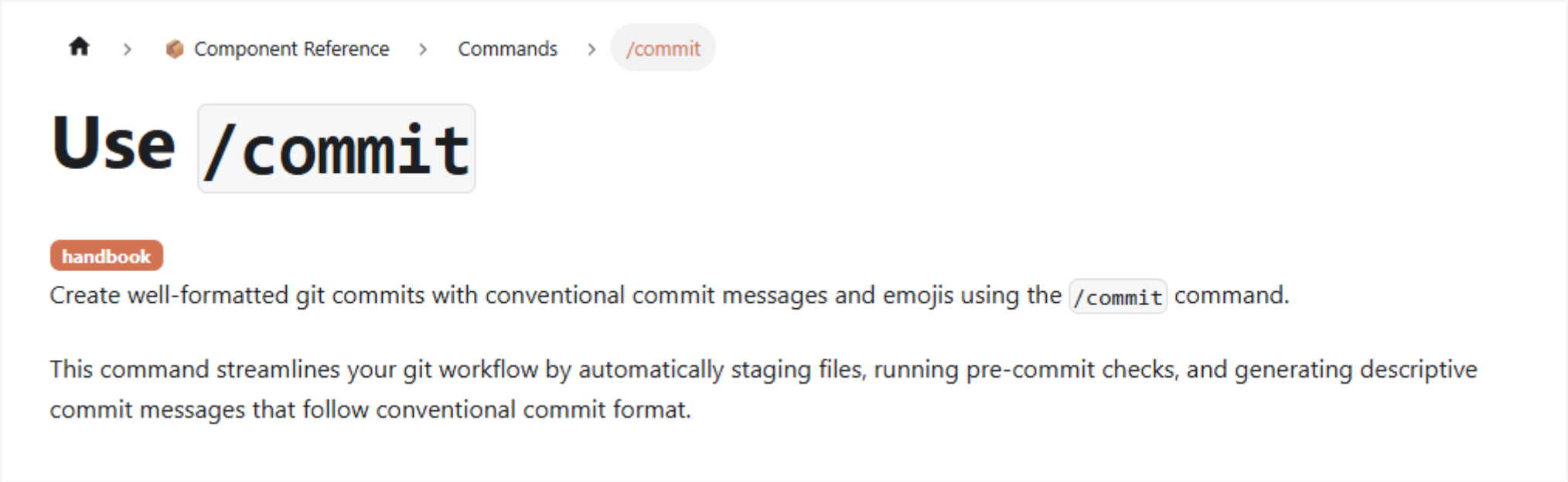

From CI/CD to
Continuous AI
The Future of GitHub Automation

From CI/CD to Continuous AI

- CI (Continuous Integration)
  - Developers merge code frequently
  - Automated builds and tests validate changes
- CD (Continuous Deployment)
  - Code automatically deployed to production
  - Reduced manual intervention

https://techcommunity.microsoft.com/blog/azuredevcommunityblog/from-cicd-to-continuous-ai-the-future-of-github-automation/4507378

Git Commit Message



https://nikiforovall.blog/claude-code-rules/component-reference/commands/commit/

Git Commit Message Examples

```
✨ feat(auth): add OAuth2 authentication support

Implement OAuth2 flow to support third-party login providers.
This replaces the custom token-based authentication system.
```

```
🐛 fix(api): handle null values in user profile endpoint

Prevent crashes when user profile data contains null fields.
Fixes #123
```

```
📝 docs: update installation guide with Windows instructions

Add step-by-step instructions for Windows users including
PowerShell-specific commands and common troubleshooting tips.
```

https://nikiforovall.blog/claude-code-rules/component-reference/commands/commit/

Code Review



https://x.com/iamdevloper/status/397664295875805184

Motivation

Make sure that the overall code health

Identify issues from your teammates

Effectiveness

Study of errors on programs without/with code review reported 4.5
→ 0.82 errors/100 lines

https://blog.codinghorror.com/code-reviews-just-do-it/

What to Catch

- Logic and correctness errors

```python
# ❌ Buggy: accidentally uses `is` for comparison
if status is "ok":
    process()

# ✅ Correct
if status == "ok":
    process()
```

What to Catch

- Logic and correctness errors
- Readability and maintainability

```python
# ❌ Hard to read
x = [i for i in range(100) if i % 7 == 0 and i % 5 != 0 or i % 11 == 0]

# ✅ Clearer with helper functions
def is_valid(n):
    return (n % 7 == 0 and n % 5 != 0) or n % 11 == 0

x = [i for i in range(100) if is_valid(i)]
```

What to Catch

- Logic and correctness errors
- Readability and maintainability
- Performance

```python
# ❌ O(n^2) – unnecessary repeated lookups
for user in users:
    if user in big_list:
        process(user)

# ✅ Better – O(n) using a set
big_set = set(big_list)
for user in users:
    if user in big_set:
        process(user)
```

What to Catch

- Logic and correctness errors
- Readability and maintainability
- Performance
- Security

```python
# ❌ Dangerous: executing user input
eval(user_input)

# ✅ Safe alternative
import ast
safe_expr = ast.literal_eval(user_input)
```

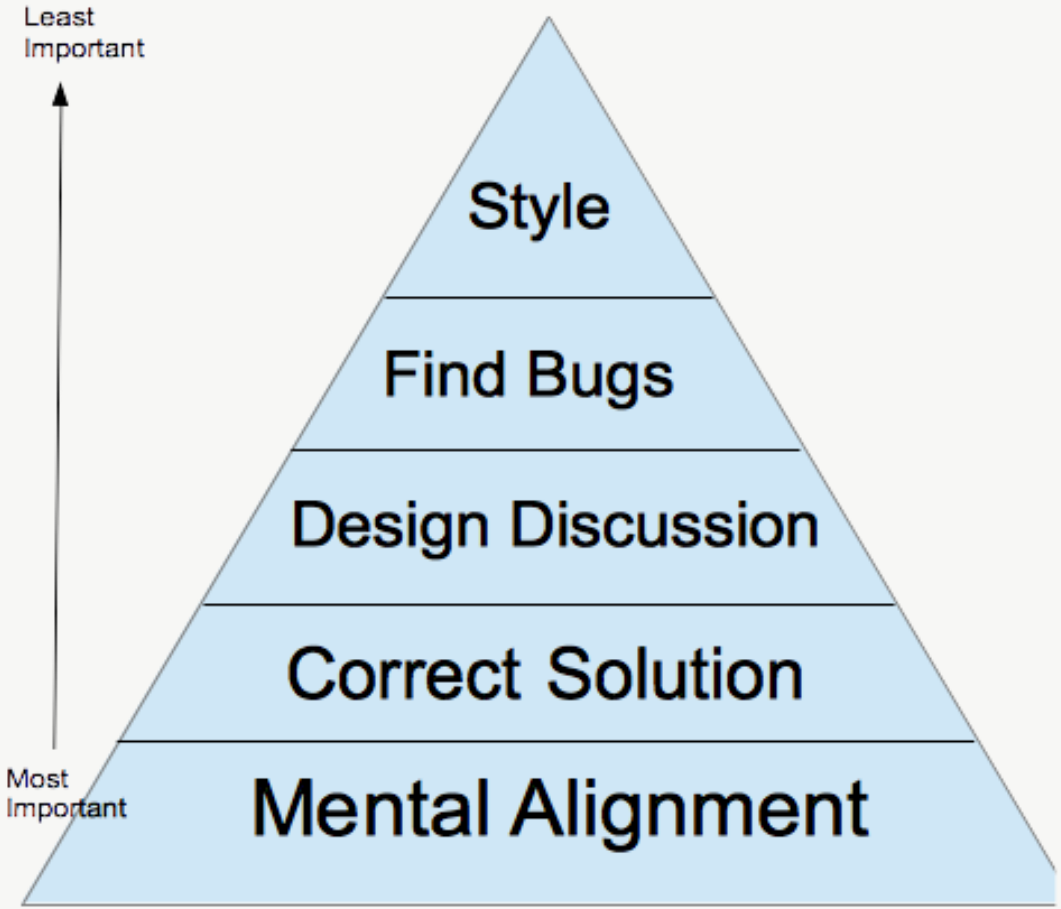
What to Catch

- Logic and correctness errors
- Readability and maintainability
- Performance
- Security
- Best practices

```
# ❌ Non-standard pattern: accessing by username
user = users_by_username["alice123"]
send_email(user["email"])

# ✅ Standard pattern: accessing by user_id
user = users_by_id[user_id]
send_email(user["email"])
```

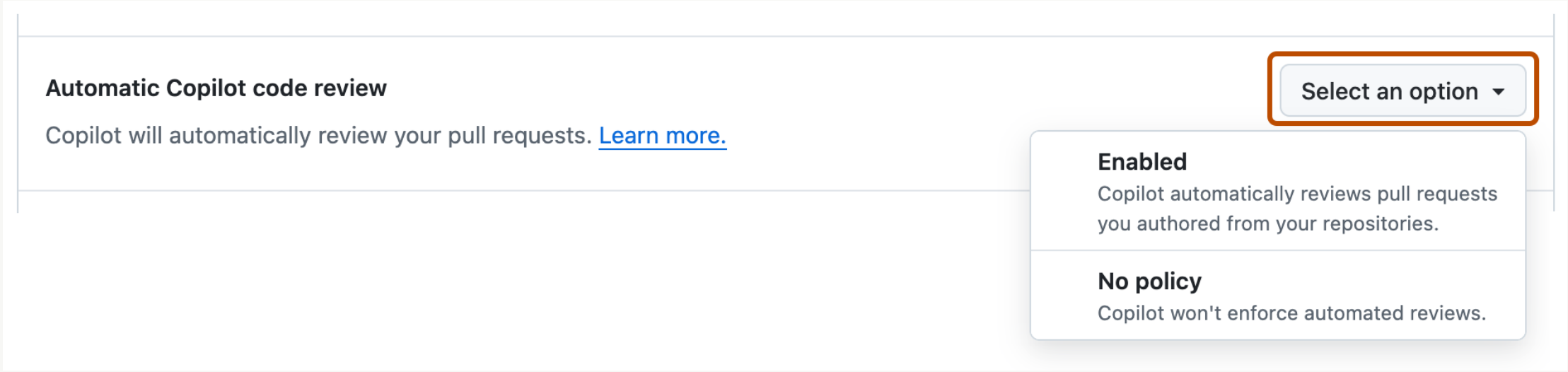
Code Review: Hierarchy of Needs

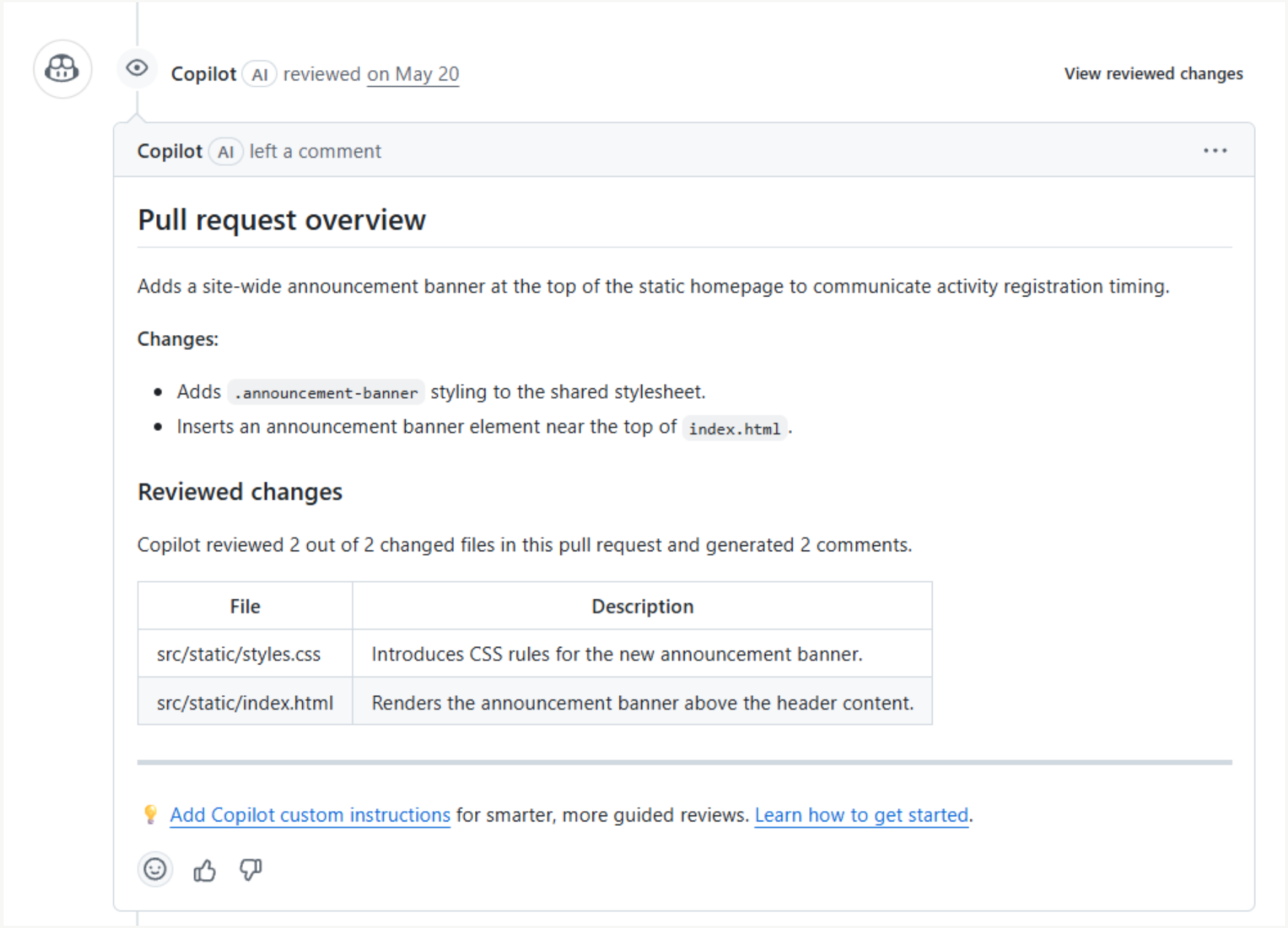


https://blakesmith.me/2015/02/09/code-review-essentials-for-software-teams.html

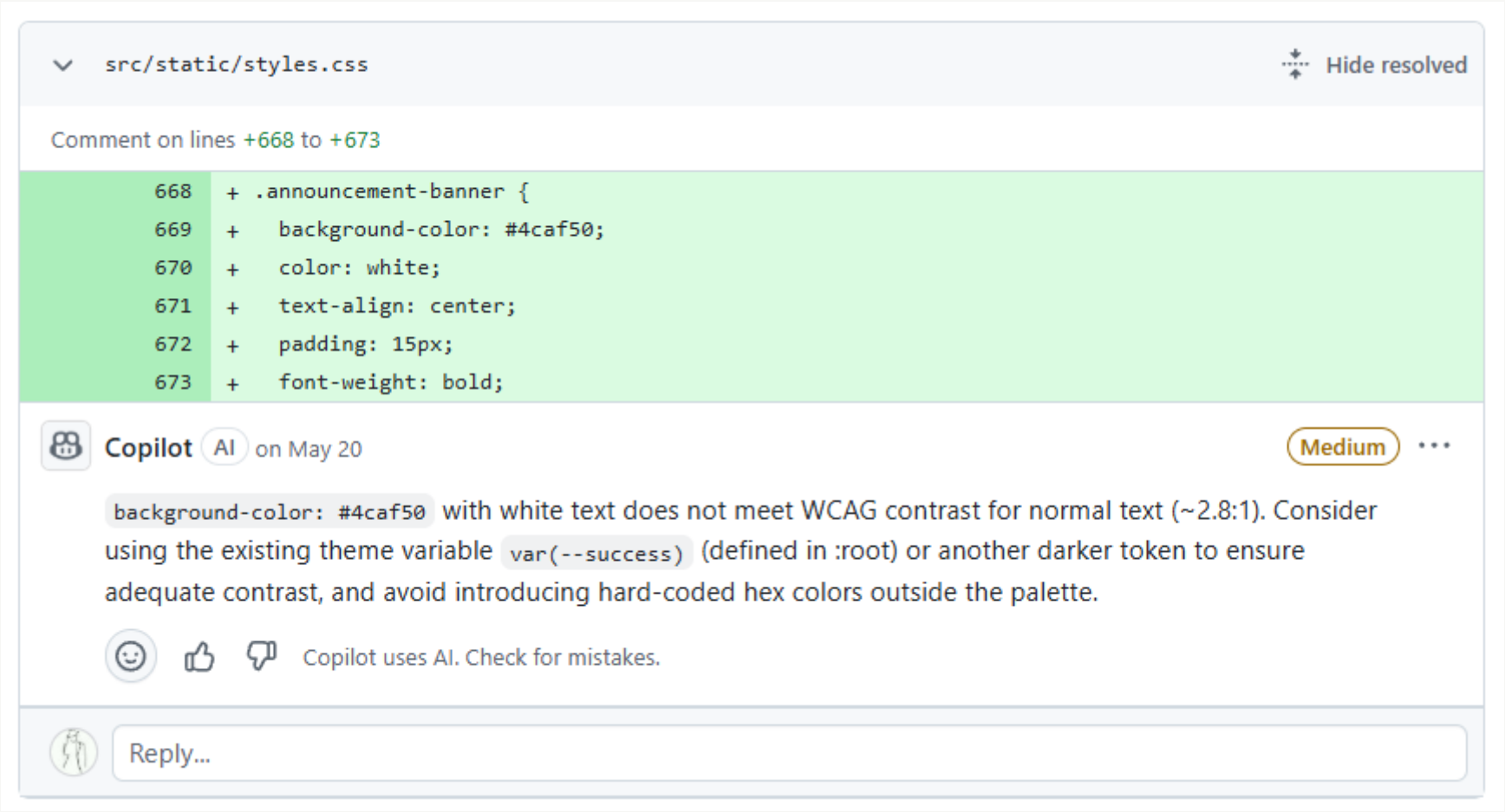
Existing Productions

- GitHub Copilot


Automatic Copilot code review
Copilot will automatically review your pull requests. Learn more.
Select an option
Enabled
Copilot automatically reviews pull requests you authored from your repositories.
No policy
Copilot won't enforce automated reviews.



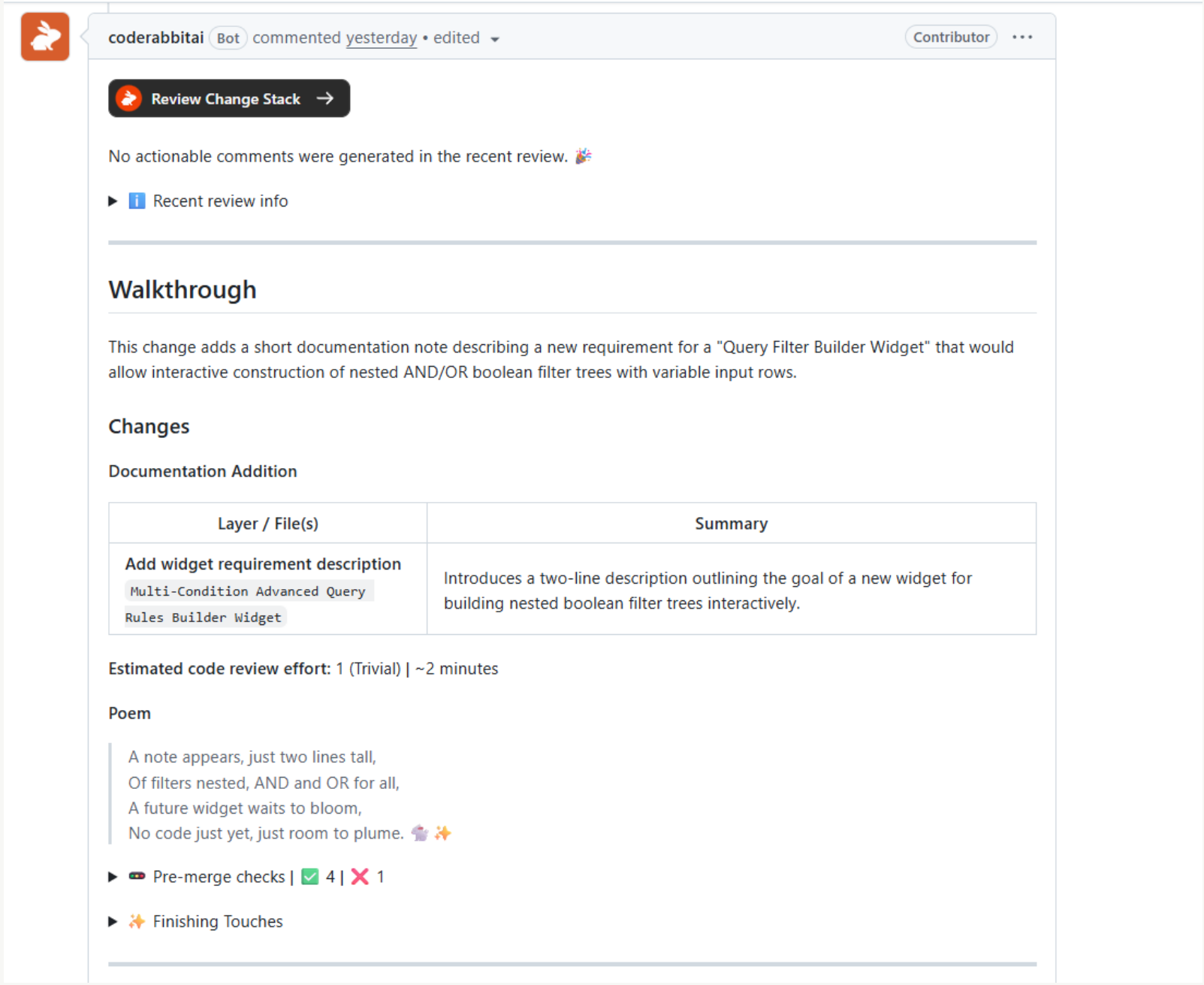
https://github.com/EficodeDemoOrg/copilot-code-review-demo/pull/3



https://github.com/EficodeDemoOrg/copilot-code-review-demo/pull/3

Existing Productions

- GitHub Copilot
- CodeRabbitai



coderabbitai Bot commented yesterday • edited Contributor

Review Change Stack →

No actionable comments were generated in the recent review. 🎉

▶ ℹ️ Recent review info

Walkthrough

This change adds a short documentation note describing a new requirement for a "Query Filter Builder Widget" that would allow interactive construction of nested AND/OR boolean filter trees with variable input rows.

Changes

Documentation Addition

Layer / File(s)	Summary
Add widget requirement description `Multi-Condition Advanced Query Rules Builder Widget`	Introduces a two-line description outlining the goal of a new widget for building nested boolean filter trees interactively.

Estimated code review effort: 1 (Trivial) | ~2 minutes

Poem

> A note appears, just two lines tall,
> Of filters nested, AND and OR for all,
> A future widget waits to bloom,
> No code just yet, just room to plume. 🪻 ✨

▶ 🕶 Pre-merge checks | ✅ 4 | ❌ 1

▶ ✨ Finishing Touches

https://github.com/appsmithorg/appsmith/pull/41950

Can you build a GitHub Action using MiMO?

```yaml
name: AI PR Review

on:
  pull_request:
    types: [opened, synchronize, reopened]

permissions:
  contents: read
  pull-requests: write

jobs:
  review:
    # 不在 fork PR 上使用仓库 secrets，避免泄露
    if: github.event.pull_request.head.repo.full_name == github.repository
    runs-on: ubuntu-latest

    steps:
      - name: Checkout base repository
        uses: actions/checkout@v4
        with:
          ref: ${{ github.event.pull_request.base.sha }}
          persist-credentials: false

      - name: Set up Python
        uses: actions/setup-python@v5
        with:
          python-version: "3.12"

      - name: Install dependencies
        run: pip install requests

      - name: Run review
        env:
          MIMO_BASE_URL: ${{ secrets.MIMO_BASE_URL }}
          MIMO_API_KEY: ${{ secrets.MIMO_API_KEY }}
          MIMO_MODEL: ${{ secrets.MIMO_MODEL }}
          GITHUB_TOKEN: ${{ github.token }}
          REPO: ${{ github.repository }}
          PR_NUMBER: ${{ github.event.pull_request.number }}
          PR_HEAD_SHA: ${{ github.event.pull_request.head.sha }}
        run: python .github/scripts/mimo_review.py
```



香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Questions

Next class: Coding Exercise & Quiz I