



PEOPLE
IDEAS
CODE
A BETTER
TOMORROW

From Planning and Sprint Design to Development, Release, Production, and Continuous Improvement



This 10-part series maps where and how AI can add value across the software development life cycle (SDLC) using GitHub Copilot and the broader ecosystem.

- ✓ A practical, end-to-end view of the SDLC
- ✓ Key use cases and real examples
- ✓ Tools, integrations, and patterns
- ✓ Tips, best practices, and lessons learned



GitHub Copilot works across your tools, data, and workflows.



Monitoring

Observe, troubleshoot,
and learn (MCP, Agents).

1. Discovery & Product Planning

Clarify the problem before building the solution.

Turn ideas, user needs, and business goals into a clear, actionable plan for what to build, why it matters, and how to measure success.



Why It Matters

A strong discovery phase reduces waste, aligns stakeholders, and sets the foundation for a successful SDLC. Clear goals and well-defined requirements lead to better decisions and faster delivery down the line.

1

What happens in this phase

Bring together people, context, and data to understand the problem and define the plan.



Define product vision

Why are we building this?



Understand user needs

Who are the users and what do they need?



Align business goals

How does this create value?



Identify constraints

Time, budget, technical, regulatory, etc.



Set success metrics

How will we measure success?

2

Where AI helps

Use AI to turn scattered information into clear insights and actionable plans.



Summarize stakeholder notes

Turn meeting notes into key themes and actions.



Synthesize research

Find patterns across user interviews, surveys, and existing data.



Generate personas and epics

Create user personas and draft epics from research.



Turn vague ideas into clearer requirements

Refine and structure ideas into actionable requirements.



Identify risks and dependencies

Highlight potential risks, assumptions, and dependencies early.

IDEAS
PEOPLE
INSIGHTS

PLANS
ALIGNMENT
REAL IMPACT

3

GitHub Copilot stack

A connected set of tools to support planning and discovery.



Copilot Chat in IDE

Get help brainstorming, refining requirements, and exploring options.



Copilot App for issue discussions

Use Copilot in GitHub to summarize discussions and capture action items.



RAG over docs and wiki

Ground answers in your product docs, wikis, and research.



MCP access to knowledge systems

Connect to tools like Jira, Confluence, and other internal systems.



Skills for reusable planning templates

Use custom skills for personas, epics, and requirements templates.

4

Outputs & benefits

A clear plan, shared understanding, and a strong foundation for delivery.



Product vision statement

A clear and compelling vision.



Defined scope

What's in scope (and out of scope).



Epic list

High-level features and user stories.



Early roadmap

Sequenced plan with key milestones.



Risk register

Known risks, assumptions, and dependencies.



Better alignment and faster discovery

Less time on analysis, more time building the right solution.



Good AI prompts here

Try prompts like:



"Summarize these user interview notes and extract the key insights."



"Draft a list of epics based on these requirements and research."



"List the main assumptions in this plan and what we should validate."



"What requirements might be missing based on this information?"

2. Sprint Planning & Backlog Shaping

Turn priorities into a workable sprint plan.

1 What happens in this phase

Refine and shape the backlog so the team can plan a realistic and valuable sprint.

- ✓ Refine backlog items
- ✓ Split epics into smaller stories
- ✓ Define acceptance criteria
- ✓ Estimate effort
- ✓ Map dependencies
- ✓ Align on a clear sprint goal



2 Where AI helps

Use AI to speed up backlog refinement and create a clear, actionable sprint plan.

- ✓ Split work into smaller tasks
- ✓ Draft user stories in a consistent format
- ✓ Suggest acceptance criteria
- ✓ Detect duplicate or similar items
- ✓ Expose potential blockers and dependencies
- ✓ Propose estimation considerations
- ✓ Summarize previous sprint lessons

4 Outputs & benefits

A well-shaped backlog and a clear plan for a successful sprint.

- ✓ Clear and shared sprint goal
- ✓ Prioritized and refined backlog
- ✓ Well-defined acceptance criteria and Definition of Done
- ✓ Cleaner issue structure
- ✓ Faster, more focused planning sessions

3 GitHub Copilot stack A connected set of tools to help you plan and shape the backlog.



GitHub CLI

Create and manage issues, labels, and PR workflows from the command line.



GitHub Copilot CLI

Get terminal-based planning help, generate issues, and analyze your backlog.



Copilot Chat

Discuss ideas, refine stories, and get suggestions in natural language.



Skills

Use backlog and issue templates to keep structure and consistency.



Hooks

Enforce issue hygiene (e.g., required fields, labels, or templates).



Agents

Coordinate multi-step planning tasks across issues, dependencies, and documentation.



Useful prompts

Try these prompts with GitHub Copilot to speed up your sprint planning.



"Break this epic into stories."



"Draft acceptance criteria."



"Find dependencies."



"Create a sprint checklist."

3. Architecture & Solution Design

PEOPLE
IDEAS
CODE
A BETTER
TOMORROW

Design systems with clarity, trade-offs, and reusable patterns.

Use AI and GitHub Copilot to explore options, design secure and scalable architectures, and document decisions for your software solution.

1 What happens in this phase

Turn requirements into a well-architected solution.

- ✓ Choose the right architecture style (e.g., monolith, modular, microservices, serverless).
- ✓ Define components and interfaces.
- ✓ Model data and storage needs.
- ✓ Identify integrations with internal and external systems.
- ✓ Evaluate constraints, trade-offs, and risks (performance, security, cost, compliance).

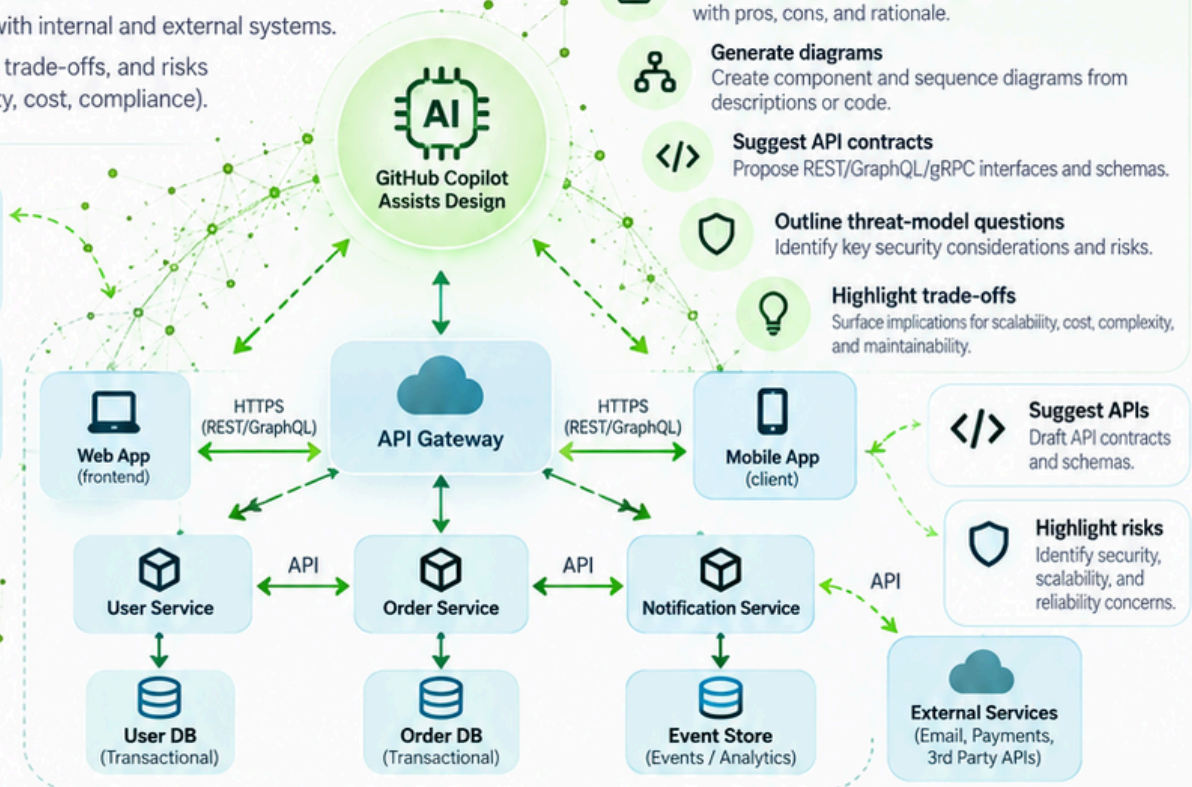
2 Where AI helps

Get faster insights and higher-quality designs.

- Compare design options**
Evaluate patterns and recommend alternatives based on your context.
- Draft ADRs**
Create structured Architecture Decision Records with pros, cons, and rationale.
- Generate diagrams**
Create component and sequence diagrams from descriptions or code.
- Suggest API contracts**
Propose REST/GraphQL/gRPC interfaces and schemas.
- Outline threat-model questions**
Identify key security considerations and risks.
- Highlight trade-offs**
Surface implications for scalability, cost, complexity, and maintainability.

AI suggests Components
Propose a modular architecture based on requirements.

Generate diagrams
Create component and sequence diagrams.



3 GitHub Copilot stack

Use the right tools together for architecture work.

- Copilot Chat in IDE**
Discuss options, get architecture guidance, and refine designs in context.
- RAG over architecture docs and standards**
Ground answers in your internal docs, standards, and reference architectures.
- MCP to internal knowledge bases**
Connect to enterprise systems, wikis, and design libraries.
- Skills for ADR and diagram templates**
Use reusable templates and structured prompts for ADRs, diagrams, and reviews.
- SDK for custom architecture assistants**
Build tailored agents for your architecture domain.
- HydraFusion for orchestrated reasoning**
Combine multiple AI agents to explore options, validate designs, and produce artifacts.

4 Outputs & benefits

From designs to decisions, faster.

- Architecture Decision Records (ADRs)**
Clear, structured decisions with rationale.
- API outline**
Endpoints, schemas, and interface definitions.
- Component map**
Visual diagrams of services and interactions.
- Data model**
Entities, relationships, and storage strategy.
- Clearer decisions**
Better visibility into trade-offs and risks.
- Faster design exploration**
Try more options and converge quickly on the right solution.

Artifacts AI can draft

Get started with ready-to-review artifacts that you can refine.

- Architecture Decision Record (ADR)**

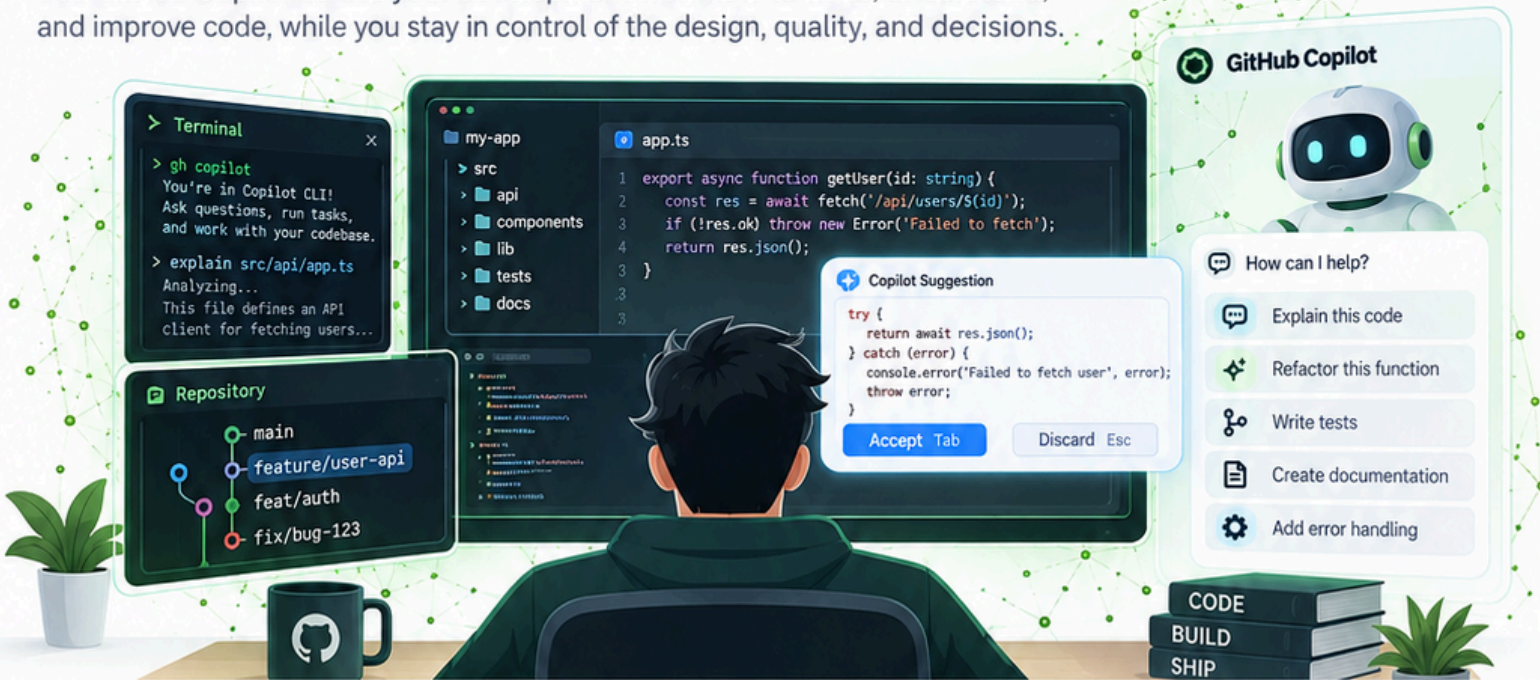
Context
Decision
Consequences
- Sequence diagram**
- Data model**
- Integration checklist**
 - ✓ External system and purpose
 - ✓ Authentication method
 - ✓ Rate limits and error handling
 - ✓ Data mappings
 - ✓ Security and compliance considerations

PEOPLE
IDEAS
CODE
A BETTER
TOMORROW

4. Development & AI Pair Programming

Build faster while keeping humans in control.

Use GitHub Copilot across your development workflow to write, understand, and improve code, while you stay in control of the design, quality, and decisions.



1

What happens in this phase

Turn plans and designs into working software.



Implement features

Write and complete code for new functionality.



Refactor code

Improve structure, readability, and maintainability.



Write documentation

Add inline comments and project docs.



Create scripts

Automate repetitive tasks.



Connect APIs

Integrate with internal and external services.



Fix bugs

Find, understand, and resolve issues.

2

Where AI helps

Get tailored suggestions, explanations, and ready-to-use code.



Explain unfamiliar code

Summarize what existing code does.



Generate boilerplate

Create common patterns and templates.



Scaffold endpoints

Build API routes, handlers, and tests.



Propose refactors

Suggest cleaner, safer, more idiomatic code.



Write regex/SQL/snippets

Create and refine complex expressions.



Draft docs

Turn code into clear documentation.



Decompose TODOs

Break down tasks into smaller steps.



Best use cases

Where Copilot shines in development work.



Legacy code explanation

Understand unfamiliar systems and codebases.



Refactoring

Modernize and improve existing code.



Test-first coding

Generate tests and implement features iteratively.



Docs generation

Create README files, API docs, and inline comments.

3

GitHub Copilot stack

A set of tools that work together across your workflow.



GitHub Copilot in IDE

Real-time code suggestions in your editor.



Copilot Chat

Ask questions, explain code, and get guidance.



Copilot CLI

Use Copilot in your terminal and automate tasks.



RAG over codebase & docs

Get answers from your code, docs, and internal knowledge.



Skills

Add custom instructions and workflows for your team.



Agents

Tackle multi-file tasks across your codebase autonomously.

4

Outputs & benefits

Higher productivity and higher quality.



Faster implementation

Turn ideas into working code sooner.



Fewer context switches

Get help in your flow, without leaving your tools.



Better onboarding

Understand new codebases more quickly.



More consistent patterns

Follow your team's standards and best practices.



PR-ready code

Higher quality, better tested, and easier reviews.

5. Testing & Quality Engineering

Use AI to expand coverage, reveal edge cases, and speed up feedback.

1 What happens in this phase

-  Write unit, integration, and end-to-end tests.
-  Validate business rules and acceptance criteria.
-  Reproduce and diagnose bugs.
-  Review test failures and improve tests and code.

MORE
COVERAGE

HIGHER
QUALITY

End-to-End Tests
User journeys and critical flows

Integration Tests
Services, APIs and contract tests

Unit Tests
Functions, components and business logic

```
describe('User login', () => {
  it('should allow valid user', () => {
    // ...
  })
});
```

- ✓ Unit tests
- ✓ Integration tests
- ✓ End-to-end tests

 1 failing test

CI Results

✓ Unit tests	124/124
✓ Integration tests	36/36
✓ End-to-end tests	20/20
✗ Security tests	1/20

 Test failed

3 GitHub Copilot stack

Use the full Copilot ecosystem to accelerate quality engineering.

-  **Copilot Chat**
Get help writing, fixing, and explaining tests.
-  **Copilot CLI**
Run local test workflows, generate tests, and analyze failures.
-  **Agents**
Automate repair loops, triage failures, and suggest fixes.
-  **Hooks**
Run quality gates in PRs and CI (check tests, coverage, linters).
-  **RAG**
Use company quality standards, testing guidelines, and past incidents as context.

AI can generate

Examples of what Copilot can create to support your testing workflow.

-  **Unit tests**
Tests for functions, components, and edge cases.
-  **Factories**
Test data factories for consistent and realistic data.
-  **Fixtures**
Sample data, mocks, and environments.
-  **Negative cases**
Tests for invalid input, error states, and abuse cases.
-  **Bug reports**
Clear, reproducible bug reports with steps, expected vs. actual results, and suggested fixes.

4 Outputs & benefits

Higher confidence, fewer regressions, and more reliable releases.

-  **Broader coverage**
More unit, integration, and end-to-end tests, including edge cases.
-  **Faster test authoring**
Generate and maintain tests more quickly with AI assistance.
-  **Earlier bug discovery**
Find issues sooner in development and in CI pipelines.
-  **More reliable releases**
Higher quality software with fewer regressions and production issues.

6. Code Review, Security & Compliance

Review changes faster without lowering standards.

1 What happens in this phase

Turn a pull request into a high-quality, secure, compliant merge.

- ✓ Review pull requests
- ✓ Check architecture fit
- ✓ Validate coding standards
- ✓ Assess security and compliance
- ✓ Prepare for merge

2 Where AI helps

Augment reviewers with context, analysis, and guidance.

- Summarize diffs
- Explain code changes
- Draft review comments
- Flag risky areas
- Map changes to secure coding guidelines
- Identify policy gaps and missing tests



Pull request #482

Open

Add user authentication flow

+120 -18

Conversation 4

Commits 3

Checks 6

Files 8



AI Summary

Adds OAuth authentication, refactors session handling, and updates error handling.



Security Check

Potential sensitive data exposure. Consider using environment variables.



AI Review Comment

Consider extracting this logic into a separate function for better testability.



Compliance Check

Maps to secure coding guideline AUTH-01. No policy violations found.

3 GitHub Copilot stack

Combine GitHub Copilot and the broader ecosystem to power smarter, more secure reviews.

- Copilot App for PR discussions
- GitHub Copilot on repo workflows
- GitHub CLI
- RAG over security and compliance docs
- Hooks for policy enforcement
- SDK or MCP-backed review bots



copilot reviewed 2 hours ago

AI

3 comments



security-bot reviewed 3 hours ago

1 issue



alice approved 5 hours ago

Approved



bob changes requested 6 hours ago

2 comments

Merge pull request



Human in the loop

AI provides analysis, suggestions, and checks, but engineers make the final decisions.

Review AI suggestions critically.

Use domain knowledge and context.

You approve, request changes, or merge.

4 Outputs & benefits

Higher quality code, lower risk, and faster delivery.

- Better review quality**
Clearer context and more consistent feedback.
- Faster approvals**
Summaries and AI insights reduce review time.
- Safer merges**
Security and compliance issues caught earlier.
- Stronger auditability**
Linked guidelines, policies, and review history.



A Connected Toolchain

GitHub Copilot works across your tools, data, and workflows.



IDE

Review diffs and get AI insights.



Terminal

Use GitHub CLI and Copilot CLI.



GitHub Repo

Pull requests, issues, and docs.



Documents

Security guidelines, compliance policies (RAG).



Deployment

Policy checks in CI/CD.

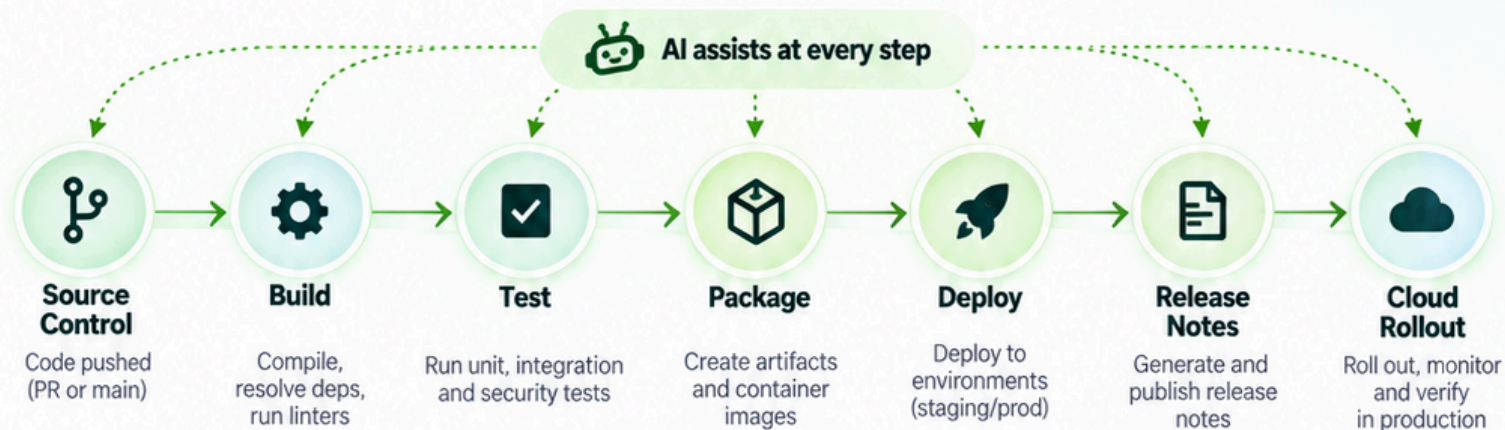


Monitoring

Track compliance and review metrics.

7. CI/CD, Release & Deployment

Automate delivery and reduce release friction.



1 What happens in this phase

Turn validated code into a reliable release.



Prepare CI workflows

Set up workflows, triggers, and quality checks.



Validate builds

Run tests, security scans, and verify artifacts.



Manage environments

Configure and maintain dev, staging, and production.



Package releases

Build artifacts, container images, and version them.



Deploy changes

Automate deployments with approvals and gates.



Communicate release details

Share release notes and update stakeholders.

2 Where AI helps

Use AI to automate, troubleshoot, and document.



Generate workflow YAML

Create and refine GitHub Actions workflows.



Fix pipeline errors

Diagnose failures and suggest fixes.



Draft release notes

Summarize changes from commits, PRs, and issues.



Prepare rollout and rollback plans

Suggest safe rollout strategies and rollback steps.



Compare environment configs

Find and explain differences between environments.



Create deployment scripts and checklists

Generate scripts and step-by-step runbooks.

3 GitHub Copilot stack

A connected toolchain for delivery.



GitHub CLI

Run workflows, manage releases, and automate tasks.



GitHub Copilot CLI

Use natural language to run and manage DevOps tasks.



Copilot Chat in IDE

Get help writing workflows, scripts, and configurations.



Hooks for release quality gates

Automate checks (tests, security, compliance) before deploy.



MCP to infrastructure documentation

Access live infrastructure docs, runbooks, and APIs.



Agents and HydraFusion

Orchestrate multi-step release tasks across tools and systems.

4 Outputs & benefits

More reliable, faster, and safer releases.



Cleaner pipelines

Consistent, maintainable CI/CD workflows.



Faster releases

Less manual work and quicker time to production.



Better release communication

Clear, accurate, AI-generated release notes.



Safer deployments and rollbacks

Planned rollouts, automated checks, and quick recovery when needed.



AI-assisted artifacts

Key outputs created or enhanced with AI.



Release notes

Clear, structured, user-friendly notes.



Workflow files

GitHub Actions YAML workflows.



Rollback plan

Steps and scripts to safely revert.



Deployment checklist

Pre-deploy, deploy, and post-deploy tasks.

8. Production, Monitoring & Incident Response

After release, use AI to detect, diagnose, and recover faster.

1 What happens in this phase

Observe usage, logs, metrics, and traces; respond to incidents; troubleshoot failures; and protect reliability.



Observe

Monitor usage, logs, metrics, and traces across your systems.



Respond

Detect and respond to incidents quickly and effectively.



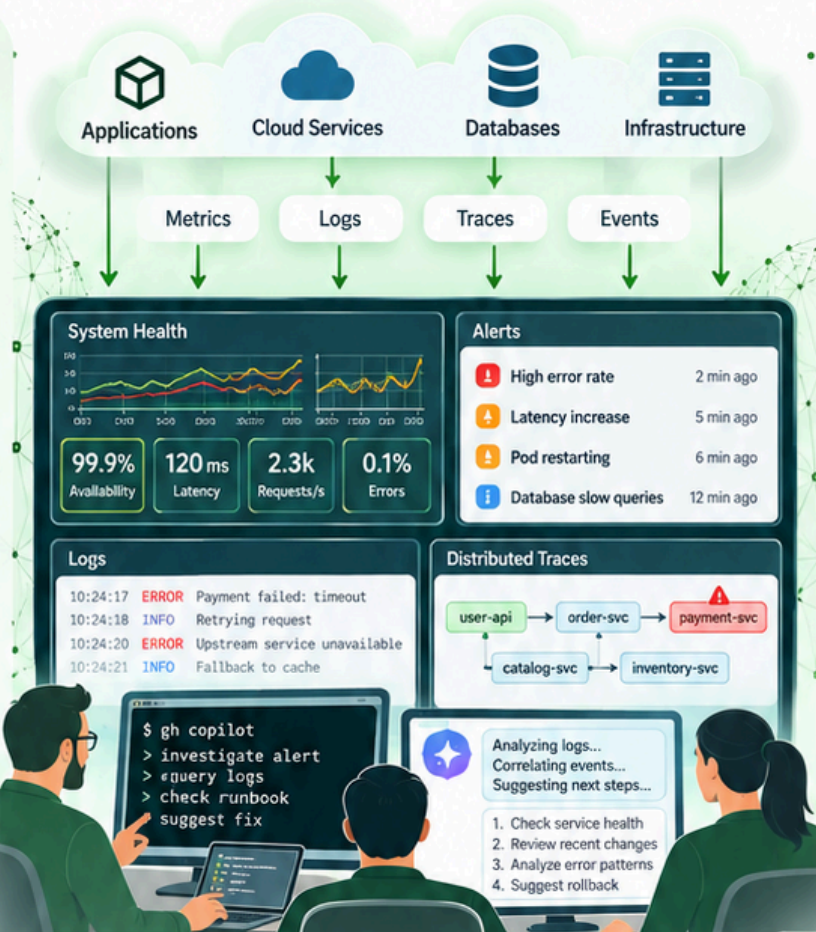
Troubleshoot

Investigate failures and find the root cause.



Protect reliability

Maintain performance, security, and a great user experience.



2 Where AI helps

Use AI to make sense of signals, speed up investigation, and take informed action.



Summarize alerts

Turn noisy alerts into clear summaries with impact.



Query logs

Ask natural language questions about logs.



Correlate symptoms

Find connections across metrics, logs, and traces.



Suggest hypotheses

Propose likely root causes and next steps.



Retrieve runbook steps

Find relevant runbook guidance instantly.



Guide rollback decisions

Assess risk and suggest safe rollback options.



Draft timelines & postmortems

Turn investigation notes into clear timelines and postmortems.

3 GitHub Copilot stack

Use the GitHub ecosystem, MCP, and your tools to investigate and respond, with AI assistance at every step.



GitHub Copilot CLI

Investigate incidents directly from your terminal.



GitHub CLI

Check issues, PRs, deployments, and service status.



MCP access

Connect to observability and ticketing systems (e.g., Datadog, Grafana, PagerDuty, ServiceNow).



RAG over runbooks

Search runbooks, knowledge bases, and prior incidents for relevant context.



Agents

Use agents for investigation workflows and multi-step analysis.



Skills

Use skills for incident templates, communication, and postmortem drafting.



Useful production tasks

Examples of everyday tasks you can do with GitHub Copilot.



Explain an alert

"Explain this Datadog alert and its likely cause."



Query logs

"Find all errors for the payment service in the last hour."



Build incident timeline

"Create a timeline of this incident from the logs and events."



Draft postmortem

"Draft a postmortem from these notes, including root cause and action items."

4 Outputs & benefits

A more resilient, observable, and self-improving system.



Faster MTTR

Detect, diagnose, and recover faster with AI assistance.



Clearer communication

Generate incident updates and summaries for stakeholders.



Better root-cause analysis

Correlate signals and find the real cause, not just the symptoms.



Improved runbooks

Turn learnings into better runbooks and preventive actions.

9. Feedback, Learning & Continuous Improvement

Close the loop and make the next cycle smarter.

Turn real-world usage, team insights, and operational data into actionable improvements with GitHub Copilot, so each cycle delivers more value, faster.



About This Phase

Capture feedback, learn from experience, and continuously improve your software, processes, and team capabilities with the help of GitHub Copilot and AI.

- ✓ Close the loop from usage to action
- ✓ Turn insights into measurable improvements
- ✓ Build a stronger, smarter team every iteration



1

What happens in this phase

- Gather customer feedback from real usage.
- Review product, delivery, and quality metrics.
- Analyze defects and recurring issues.
- Capture lessons learned from the team.
- Reprioritize improvements in the backlog.
- Update ways of working, processes, and standards.

2

Where AI helps

- Cluster feedback from multiple sources.
- Extract themes and insights.
- Prioritize improvements based on impact.
- Convert incidents into playbooks and runbooks.
- Update documentation and keep it current.
- Create reusable prompts and build team skills.
- Measure adoption and improvement over time.

3

GitHub Copilot stack

Use the right Copilot capabilities to turn learning into action.

- Copilot App**
Collect and synthesize feedback across tools and data sources.
- Copilot Chat**
Analyze insights, explore options, and draft improvements.
- RAG**
Search and ground answers in tickets, docs, and postmortems.
- Skills**
Create reusable workflows for common improvement tasks.
- Hooks**
Set up process reminders and checks (e.g., update docs).
- SDK**
Build custom learning assistants integrated with your tools.
- Agents**
Run recurring analysis tasks (e.g., feedback triage, metric reports).

4

Outputs & benefits

- Better backlog inputs**
Data-driven, user-focused improvements.
- Stronger knowledge reuse**
Runbooks, docs, and playbooks stay up to date.
- Improved team enablement**
Faster onboarding and upskilling with AI.
- Measurable productivity and quality gains**
Fewer repeat issues and faster delivery over time.



Metrics to watch

Track progress and the impact of continuous improvement.



Lead time
(idea to production)
↓ lower is better



PR cycle time
(open to merge)
↓ lower is better



Escaped defects
(in production)
↓ lower is better



MTTR
(mean time to recovery)
↓ lower is better



Knowledge reuse
(runbook/doc usage)
↑ higher is better

