Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

Spec-Driven Development with GitHub Spec Kit

Class Duration

7 hours of live training delivered over 1-2 days to accommodate your scheduling needs.

Student Prerequisites

- Professional software development experience
- Working familiarity with at least one AI coding assistant (Copilot, Claude Code, Cursor, or Windsurf)
- Comfort with the command line and Git

Target Audience

Software engineers, tech leads, and platform engineers who want a disciplined, agent-friendly workflow that produces specifications first and code second. Equally relevant for teams shipping AI-generated code at scale who need a structured process for grounding agents, and for organizations rolling out spec-driven development as a team standard.

Description

Most AI coding workflows start with a prompt and hope for the best. Spec-driven development inverts that: the specification becomes the primary artifact and the source of truth, and code is what the agent produces from it. GitHub Spec Kit is the open-source toolkit that operationalizes this discipline—a CLI (`specify`) and a four-phase workflow (specify → plan → tasks → implement) with broad agent compatibility including GitHub Copilot, Claude Code, Cursor, Windsurf, Gemini CLI, and OpenAI Codex. This course teaches the methodology and the tool: writing specifications that agents can actually act on, decomposing plans into reviewable tasks, driving each implementation phase with the agent of your choice, and integrating the workflow into a real engineering process with reviews and CI.

Learning Outcomes

- Explain spec-driven development and how it changes the relationship between human, specification, and AI agent.

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

- Install and configure Spec Kit (`specify` CLI) for a project and select the right agent template.
- Write specifications that capture intent, acceptance criteria, and constraints precisely enough for an agent to execute against.
- Run the four-phase workflow—specify, plan, tasks, implement—end-to-end on a realistic feature.
- Use Spec Kit's project constitution to encode persistent rules and architectural constraints.
- Integrate Spec Kit with GitHub Copilot, Claude Code, Cursor, and Windsurf and choose the right agent per phase.
- Apply review discipline at the spec, plan, and task levels—not just on generated code.
- Wire Spec Kit phases into CI and PR workflows for spec-as-source-of-truth pipelines.

Training Materials

Comprehensive courseware is distributed online at the start of class. All students receive a downloadable MP4 recording of the training.

Software Requirements

The Spec Kit `specify` CLI installed, Git, an active subscription or API key for at least one supported agent (GitHub Copilot, Claude Code, Cursor, Windsurf, or Gemini CLI), and a sample repository for labs.

Training Topics

Spec-Driven Development Principles

- Why prompt-first workflows produce inconsistent code at scale
- Specifications as durable contracts vs. one-shot prompts
- The four-phase mental model: specify → plan → tasks → implement
- Where spec-driven sits relative to TDD, BDD, and ADRs

Spec Kit Setup and Architecture

- Installing the `specify` CLI
- Project initialization and template selection
- The `.specify/` directory and what each file does
- Choosing an agent template: Copilot, Claude Code, Cursor, Windsurf, Gemini CLI

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

- Anatomy of a Spec Kit project on disk

Writing Effective Specifications

- The `specify` command and prompt patterns that produce useful specs
- Acceptance criteria, constraints, and out-of-scope sections
- Domain-specific vocabulary in specs
- Reviewing and iterating on a spec before moving to plan
- Anti-patterns: vague specs, implementation-leaking specs

The Project Constitution

- What goes in `constitution.md` vs. an individual spec
- Architectural rules, style conventions, library choices
- How the constitution constrains every downstream phase
- Versioning and maintaining the constitution alongside code

Plan, Tasks, and Implement Phases

- The `plan` phase: turning a spec into a technical approach
- The `tasks` phase: decomposing the plan into reviewable units
- The `implement` phase: agent-driven code generation against tasks
- Iteration patterns when an agent goes off-track
- When to revise the spec vs. the plan vs. the implementation

Agent Integration in Practice

- Copilot in Spec Kit: chat modes, prompt files, and Spaces
- Claude Code in Spec Kit: CLAUDE.md alignment with the constitution
- Cursor and Windsurf in Spec Kit: rules files vs. spec source-of-truth
- Switching agents mid-project and what’s portable
- Side-by-side comparative lab on a single spec

Review Discipline at Every Phase

- Reviewing specs as carefully as code
- Plan review: catching scope creep before implementation
- Task-level review and acceptance gates
- Code review with the spec open for reference

CI and Workflow Integration

- Spec Kit phases in pull request workflows
- Spec drift detection between code and `spec.md`
- Branch and PR conventions for spec-driven work

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

- Metrics: spec-to-implementation accuracy, rework rate

Workshop

- End-to-end Spec Kit lab: build a feature using all four phases
- Cross-agent comparison: same spec, two agents, compare outputs
- Q&A session