OCTOCO.AI

PRACTICAL AI WORKSHOPS · FOR SOFTWARE TEAMS

# Cut through AI noise. Ship engineering, not party tricks.

Honest, hands-on workshops that turn coding agents from a curiosity into durable engineering practice. Three combos, designed to be combined.

THE BRIEF

# Practice that survives the Monday after the workshop.

Your team is using coding agents every day — Copilot, Claude Code, Cursor, Codex. Individual engineers are getting better, unevenly. But the practice *around* the practice is missing: specs don't hold up under non-determinism, there are no evals to catch regressions, and no shared model of what to delegate and what to own. Lots of code, unclear quality signals, and no honest answer for leadership on whether AI is actually helping.

These three workshops close that gap. They're grounded in the XP and Agile fundamentals that ship reliable software for a living, re-applied to a technology where the thing you're building is itself unpredictable. No 10x promises. Durable practice that earns trust.

## Who these are for

Working software teams. Mid-level, senior, and staff engineers; tech leads; engineering managers; product and design partners who pair with engineering. Mixed-seniority cohorts are welcome; 6–16 attendees is the sweet spot.

**Not the right fit:** absolute beginners, developers who haven't yet shipped production code, and non-technical executives — a separate briefing format fits them better.

**Prerequisites:** comfortable reading Python (or another mainstream language); have used an LLM coding assistant for a few months; know HTTP, JSON, git, and a terminal. Combo 3 additionally assumes Python asyncio and basic TypeScript + React.

## Delivery basics

**DELIVERY**

On-site preferred. Remote with video + unmuted discussion is fine.

**COHORT SIZE**

6–16 attendees. Larger needs a second facilitator.

**CUSTOMISATION**

Pre-engagement call to tune examples to your stack.

THE OUTCOME

# What every attendee walks out with.

Four durable takeaways — a mental model, a set of practices, real artefacts built in the room, and a point of view honest enough to brief leadership with. Workshop deliverables you can defend in a code review.

01 · Mental Model

### A clear picture of what an agent actually is

Loop, context, tools, LLM — not a slide, not a metaphor. The four pieces and how they interact, named the same way across the team.

02 · Durable Practices

### Spec patterns, evals, delegation, review loops

The engineering discipline that turns agent output into something a team can trust — and that survives staff churn and model upgrades.

03 · Concrete Artefacts

### Working agent, rewritten spec, team playbook

You leave with built things — a small Python agent, a real backlog item rewritten with teeth, or a production-readiness checklist for your stack.

04 · A Point of View

### Honest framing for leadership

Grounded in engineering fundamentals, with the research numbers — enough to brief executives without hype or fatalism.

COMBO
01

THE LEAD-WITH COMBO

# Agentic Engineering for Teams (Intro)

One-day · Lightweight, practice-first.

ONE-DAY WORKSHOP
**R 31,500**
6–10 attendees · +R 2,300 / extra

Lightweight, practice-first. Builds a shared mental model of what an agent actually is — loop, context, tools, LLM — and applies it to the work the team is already doing.

**For:** any team using coding agents daily but lacking a shared model for how.

## You'll leave able to

- Explain the agent loop — LLM, context, tools, observation — without a slide.
- Build a tiny Python agent and run it locally.
- Sort tickets into **delegate** / **delegate-review** / **own yourself** with confidence.
- Rewrite a user story to include performance thresholds, graceful degradation, and failure modes.
- Name the perception-vs-reality productivity gap, with the numbers to back it up.

## Topics covered

- Opening framing — vibe coding to agentic engineering, with the honest research numbers.
- Live build of a ~150-line Python agent, mirroring Thorsten Ball's canonical reference.
- The delegation framework, applied to your actual sprint tickets.
- Specs with teeth — rewriting a real backlog item together.
- The 5-step hypothesis cadence, and context discipline.

**RIGHT CHOICE IF** *your team uses agents but hasn't built shared practice around them.*

COMBO
02

ENGINEERING DISCIPLINE

# Advanced Agentic Engineering

Two-day or three-day · Three-day recommended — the material needs the time.

RECOMMENDED · THREE-DAY
**R 91,000**
6–10 attendees · +R 6,600 / extra

**OR TWO-DAY**
R 63,000 · +R 4,550 / extra

Specs, evals, review loops, team patterns, governance — the engineering discipline that turns agents into something you can trust in production.

**For:** engineering teams with at least one senior engineer in the room, past the "we use Copilot" phase, running into real questions — how to review AI-generated code at scale, keep agents honest in production, and not get hacked.

## You'll leave able to

- Write full AI user stories with measurable acceptance criteria — and turn those criteria into running evals.
- Stand up a basic CI/CD/CE pipeline where a failing eval blocks a merge.
- Apply the plan/execute split and the writer/reviewer loop in team exercises.
- Use Claude Code sub-agents (context isolation) and agent teams (cross-session coordination) appropriately — they're not the same thing.
- Produce a first-draft agent inventory and least-privilege access matrix.
- Brief leadership honestly on AI productivity.

## Topics covered

- Lightweight spec-driven development via a Claude Code skill + CLAUDE.md; SpecKit shown as the cross-vendor alternative.
- Hands-on evals with DeepEval, wired into a CI pipeline that blocks a regressing commit.
- Writer/reviewer loop exercise; plan/execute split across Opus and Haiku.
- Multi-agent patterns (sub-agents, agent teams, MCP) with the CooperBench caution — multi-agent is not automatically better.
- Team-shape workshop: conductor model, centaur pod, evolving role definitions.
- Security, governance, shadow-agent inventory; MCP as the cross-vendor standard.

> **RIGHT CHOICE IF** *your team has adopted agents and needs the discipline to take them seriously in production.*

COMBO
03

SHIP AGENTIC PRODUCT

# Building Agentic Backends & Chat UIs

Two-day · For teams building agentic features into their *own* products.

TWO-DAY WORKSHOP
**R 63,000**
6–10 attendees · +R 4,550 / extra

From "I've called an LLM API" to a deployed agentic backend with a streaming chat UI, evals, tracing, and guardrails.

**For:** backend and full-stack engineers embedding AI features into a product. Python + TypeScript (React + Vite) comfort assumed.

## You'll leave able to

- Implement an agent loop from scratch in Python — no framework.
- Design tool schemas an LLM uses reliably; handle parallel tool calls and strict mode.
- Stream tool-call activity and text to a React chat UI (SSE).
- Integrate MCP — consuming an existing server, and exposing your own tools.
- Use prompt and context caching with awareness of TTL and invalidation.
- Build evals covering tool-use correctness, task success, and catastrophic failures.
- Instrument tracing and debug a broken agent run from a trace.
- Mitigate prompt injection, runaway loops, tool misuse, and data exfil.
- Deploy a containerised agent and reason about session state, concurrency, and cost.

## Topics covered

- Agent loop from scratch — "an LLM, a loop, and enough tokens."
- Streaming backend in FastAPI; chat UI in Vite + React rendering tool-call activity.
- MCP (spec 2025-11-25): primitives plus Elicitation as a first-class client capability.
- Anthropic prompt caching vs Gemini context caching — the cost-shape comparison.
- DeepEval in CI; Opik for production tracing; Inspect AI for trajectory evals at scale.
- Two running-artefact tracks — codebase-Q&A + stubbed PR drafting, or helpdesk with escalation — chosen to fit your domain.

**RIGHT CHOICE IF** *your team is shipping AI features into a product and needs the engineering muscle to go past "works in the happy path."*

COMMERCIALS

# Investment & terms

Pricing is in South African Rand, exclusive of VAT. The base rate covers a cohort of 6–10 attendees, with each additional attendee billed at the rate shown. Online cohorts cap at 16; on-site cohorts at 20. Recommended duration is highlighted on each combo.

### DEPOSIT

40% on booking. Balance due within 14 days after delivery.

### EARLY-PAY DISCOUNT

10% off when the full fee is paid in advance.

### ON-SITE TRAVEL

Travel and accommodation billed at cost, agreed up front.

*Quotes valid for 60 days from the date of issue.*

## How to engage

### DELIVERY

In-person preferred. Remote with working video and unmuted discussion is fine.

### CUSTOMISATION

Every combo includes a pre-engagement call to tune examples to your stack and pick the right running-artefact track for Combo 3.

### COHORT SIZE

6–16 attendees. Larger cohorts need a second facilitator.

### NEXT STEP

A 30-minute call is usually enough to scope the right combo for your team.

LET'S TALK

## Let's build practice that lasts.

Tell us how your team uses coding agents today and where it's breaking. We'll come back with a tailored combo and a date.

**workshops@octoco.ai** · **octoco.ai**

*Ground truth over hype. Fundamentals over fashion. Working software over wishful thinking.*