TECHNICAL HANDBOOK

Build an AI Agent Code-Free

Technical Tips Handbook

Decisions, architecture, access controls,
and when to actually use AI — and when not to.

zenphi

WHAT'S INSIDE

This handbook covers

THE MARKET CONTEXT

Where the enterprise AI agent market stands right now

Source: Google Cloud — State of Infrastructure in the Agentic AI Era, 2026 (1,402 global IT leaders surveyed)

83%

of organizations say their infrastructure requires upgrades to support production-grade agentic AI.

4 in 5

IT leaders cite security, governance, and MLOps as the most significant challenges to scaling AI inference.

35%

specifically flag "insufficient security for multi-system access" as the primary gap preventing agentic deployment.

81%

say operational complexity and engineering overhead is the top hidden cost when scaling AI — above hardware costs.

02

No-code vs Traditional Development

How low-code and no-code AI agent builders compare to writing agents from scratch

NO-CODE VS TRADITIONAL DEVELOPMENT

Two ways to build an AI agent

	Traditional Development	No-Code (Zenphi)
Setup time	Weeks to months	✓ **Minutes to hours**
Skills required	Python, APIs, LLM frameworks	✓ **No coding — drag-and-drop logic**
Governance	Must be built from scratch	✓ **Built in — access rules, audit logs**
Flexibility	Unlimited (if you can code)	✓ **High within platform capabilities**
Maintenance	Dev team dependency	✓ **IT or ops team can own it**
Cost model	Engineering hours + infra	✓ **Flat SaaS pricing**
Time to first agent	Months	✓ **Under 30 minutes**

03

Common Challenges

Why most AI agent projects stall before reaching production — and what causes it

COMMON CHALLENGES

zenphi

Why AI agent projects fail in production

Governance gap

No clear rules on what the agent can see or do. Leads to data exposure or unpredictable behaviour.

Routing everything through AI

Using an LLM for simple IF conditions spikes cost and reduces predictability — deterministic logic is cheaper and safer.

No audit trail

Agents that can't be monitored create compliance risk. If you can't see what it did, you can't trust it's running correctly.

Too much autonomy, too soon

Giving agents broad permissions before building narrow, tested use cases. Start scoped; expand from there.

Trigger mismatch

The agent triggers on the wrong event, or not at all. Requires careful workflow design before any AI configuration.

Maintenance burden

No-code tools that require a dev to update. If the builder can't own changes, the agent becomes fragile.

zenphi

Why AI agent projects fail in production

4 in 5 IT leaders cite governance as their #1 challenge to scaling AI. 35% flag multi-system security access as the primary gap*

Governance gap

No clear rules on what the agent can see or do. Leads to data exposure or unpredictable behaviour.

Routing everything through AI

Using an LLM for simple IF conditions spikes cost and reduces predictability — deterministic logic is cheaper and safer.

No audit trail

Agents that can't be monitored create compliance risk. If you can't see what it did, you can't trust it's running correctly.

Too much autonomy, too soon

Giving agents broad permissions before building narrow, tested use cases. Start scoped; expand from there.

Trigger mismatch

The agent triggers on the wrong event, or not at all. Requires careful workflow design before any AI configuration.

Maintenance burden

No-code tools that require a dev to update. If the builder can't own changes, the agent becomes fragile.

** Google Cloud — State of Infrastructure in the Agentic AI Era, 2026*

04

Key Steps to Building an Effective Agent

The decisions that separate a working production agent from a demo that gets shelved

KEY STEPS

zenphi

Building an effective AI agent: in order

1. **Define the use case narrowly**
   Pick one process. Not 'HR automation' — 'submit a leave request.' The narrower the first use case, the faster and safer the deployment.

2. **Map the workflow before touching the agent**
   Every agent is only as good as the workflow it's connected to. Map trigger → logic → action → notification on paper first.

3. **Configure access controls before going live**
   Decide what the agent can see, say, and do — per role and context. This is not optional for production.

4. **Use AI only where it adds value**
   Classify intent, extract data from unstructured input, handle variation. Use deterministic logic for everything else.

5. **Connect to a governed workflow**
   The agent is the front door. The workflow is the engine. They must be connected before deployment.

6. **Test with real queries including edge cases**
   Ask it things it shouldn't answer. Try inputs that break the expected flow. Production agents see everything.

7. **Deploy to one channel, monitor for two weeks**
   Start with Google Chat or internal chat. Watch the dashboard: usage, errors, unexpected responses. Expand after.

05

AI Agent Architectures

These aren't four types to choose between — they're layers. The interface and the execution engine can be different things.

zenphi

Interface layer vs execution layer — they're separate choices

INTERFACE LAYER — how employees interact

Reactive

Single input → single action → stops. No memory between turns.

e.g. Employee types 'submit leave request' once and it's done.

Conversational

Multi-turn dialogue collects all inputs before acting. Maintains context across the conversation.

e.g. Agent asks clarifying questions (dates, leave type) before triggering the workflow.

EXECUTION LAYER — how the agent acts

Workflow-bound

Every action triggers a pre-defined, governed workflow. The agent can only initiate approved paths — no improvisation.

e.g. Leave request → checks entitlement → routes to manager → logs the outcome. All deterministic.

Autonomous / Agentic

Agent decides which tools to call, in what order, chaining multiple actions. Requires strong guardrails.

e.g. IT provisioning: detects new hire → checks OU → provisions apps → notifies teams.

Zenphi AI Studio agents are Conversational (interface) + Workflow-bound (execution) — the conversation happens in Google Chat; governed workflow logic runs underneath.

06

Architecture & Decision-Making

How the architecture you choose shapes what your agent can decide — and how safely

zenphi

The architecture IS the guardrail

Most governance failures in AI agents are architecture failures, not AI failures. How you structure the agent determines what it can decide autonomously — and what requires human approval.

Workflow-bound (Zenphi model)

- ✓ Agent can only trigger pre-approved workflows
- ✓ Decision space is explicitly defined by the builder
- ✓ Every path has known outcomes — no surprises
- ✓ Audit trail is automatic — logged at workflow level
- ✓ Easiest to govern, easiest to scale confidently

Autonomous / LLM-driven

- ✓ Agent decides which tools to call and in what order
- ✓ Flexible but unpredictable — harder to audit
- ✓ Requires careful prompt engineering + guardrails
- ✓ Higher hallucination risk on action steps
- ✓ Appropriate for complex tasks with strong oversight

07

The AI Agent Stack

What components make up a working agent — and how they connect

zenphi

Five layers, one working agent

1

Input / Trigger

How the agent receives a request. Google Chat message, Form submission, Gmail trigger, API call.

2

Intent Recognition

The AI layer. Classifies what the user wants and extracts structured data from natural language.

3

Decision Logic

Determines what happens next. Should be deterministic where possible — IF conditions, lookups, routing rules.

4

Workflow Execution

The actual process that runs. Leave request, provisioning, document generation, notification. This is Zenphi.

5

Governance & Logging

Access controls, audit trail, error handling, activity dashboard. Runs across every layer.

08

Building Without Coding Skills

How to create a governed AI agent when you're not a developer — practically, step by step

zenphi

How to build your first agent in Zenphi

1 Choose one process to start

Pick a single workflow employees do repeatedly: submitting a request, checking a status, filing something. One process only.

2 Build the workflow first in Zenphi

Use Zenphi's no-code builder to map the trigger, logic, and actions. Test it as a regular workflow before adding any agent layer.

3 Create an AI agent in AI Studio

Open AI Studio, name your agent, and connect it to the workflow you just built. Define the channel: Google Chat or Zenphi Chat.

4 Configure access rules

Set who can ask what. Which roles can see which data. What the agent should say if a request is out of scope.

5 Write the agent's opening prompt

Tell the agent what it is, what it can help with, and how to respond when it can't help. Keep it short and specific.

6 Test, watch the dashboard, adjust

Deploy to a small group first. Review the activity log. Adjust access rules or workflow logic based on real queries.

09

Governed Agents in Practice

Access configuration, when to use AI, and the mistakes that get agents shelved

The most important cost and reliability decision

Use AI when...

- ✓ Understanding natural language requests with variation
- ✓ Extracting structured data from unstructured input (emails, forms, messages)
- ✓ Classifying intent when the user hasn't followed a script
- ✓ Generating personalised text (welcome emails, policy summaries)
- ✓ Handling ambiguity that deterministic logic can't anticipate

Use deterministic logic when...

- → Checking a value against a known threshold (budget, limit, date)
- → Looking up a manager, a role, or a permission level
- → Routing a request based on a known rule (department, OU, region)
- → Sending a notification when a specific event occurs
- → Anything with one correct answer — don't use AI to guess

** Google Cloud — State of Infrastructure in the Agentic AI Era, 2026*

COMMON MISTAKES TO AVOID

zenphi

What gets AI agents shelved before they scale

Building the agent before the workflow.

An agent with no workflow behind it can't do anything. Map and test the workflow first, always.

Skipping access configuration.

An agent that can answer any question from any person is a liability. Configure scope before deploying.

Starting with too broad a use case.

'A general HR agent.' Start with one use case, prove it works, then expand.

Routing IF conditions through an AI model.

This makes your agent slower, more expensive, and less predictable. Use workflow logic for anything with a right answer.

Not watching the dashboard after launch.

The first two weeks of a deployed agent reveal everything. Gaps in access rules, unexpected queries, broken triggers.

Assuming the agent is self-maintaining.

Workflows change. Policies change. Someone needs to own the agent and review it when the underlying process changes.

Ready to build your first agent?

Talk to us or start you free Zenphi trial — no credit card required

Start free with Zenphi

zenphi