# Agentic Commerce — Guardrails & Workflow

> **Status:** design specification. ALPHA does not execute payments in the current build. This document defines the
> rules any future commerce capability must follow, and the machine-checkable policy enforced in CI.

## 1. Principles

1. **Mandates, not blank cheques** — an agent may spend only under an explicit, scoped, time-boxed spend mandate
   granted by a human.
2. **Human-in-the-loop above thresholds** — any transaction above the mandate's auto-approve limit, to a new
   merchant, or in a restricted category requires human approval.
3. **Never hold raw payment credentials** — agents use tokenized, provider-issued payment methods; card numbers never
   enter prompts, memory, or logs.
4. **Everything is auditable** — intent, cart, approval, and receipt are linked by one trace ID in an append-only log.
5. **Sandbox first** — every integration ships in test mode before live mode can be enabled.

## 2. Transaction Workflow

```
 1. Intent        Agent proposes purchase (item, merchant, amount, reason, traceId)
 2. Policy check  Mandate lookup → limits, categories, merchant allow-list, period spend
 3. Risk check    Anomaly score (velocity, new merchant, price deviation, prompt-injection signals)
 4. Decision      auto-approve  | require human approval | deny
 5. Approval      HIL queue in ALPHA (Self-Healing / Assistant Hub) with full context
 6. Execute       Provider API with idempotency key; tokenized payment method only
 7. Verify        Receipt / order confirmation matched to intent
 8. Record        Signed audit entry; memory stores summary only (no payment data)
 9. Reconcile     Daily job compares ledger vs. provider settlement; mismatches raise P1
```

## 3. Spend Mandate Schema

Policies live in `docs/policies/commerce-policy.json` and are validated by `scripts/validate-commerce-policy.mjs` in the `agentic-commerce-policy` workflow.

| Field | Meaning |
|---|---|
| `id`, `agentId` | Mandate identity and the agent it is bound to |
| `currency` | ISO 4217 code |
| `perTransactionLimit` | Hard cap for a single transaction |
| `autoApproveLimit` | Transactions at or below this skip HIL (must be ≤ `perTransactionLimit`) |
| `periodLimit`, `period` | Cumulative cap per `day` / `week` / `month` |
| `allowedCategories` | Merchant categories permitted |
| `merchantAllowList` | Optional explicit merchants; empty means category rules only |
| `requireHumanApprovalFor` | Always-HIL conditions: `new_merchant`, `subscription`, `refund`, `international` |
| `expiresAt` | Mandate expiry (ISO 8601); max 90 days from issue |
| `mode` | `sandbox` or `live` |

## CI guardrail rules

- `autoApproveLimit ≤ perTransactionLimit ≤ periodLimit`
- Global ceiling: `autoApproveLimit` may not exceed `globalGuardrails.maxAutoApprove`
- `requireHumanApprovalFor` must include `new_merchant`
- Categories in `globalGuardrails.prohibitedCategories` may never be allowed
- `live` mode requires `approvedBy` (a human) to be set
- `expiresAt` must be a valid date

## 4. Threats specific to commerce

| Threat | Mitigation |
| --- | --- |
| Prompt injection instructing an agent to buy | Intent must originate from a mandate-bound task; web content is untrusted data |
| Price / merchant spoofing | Merchant allow-list, price deviation checks, receipt verification |
| Runaway loops | Period limits, velocity checks, loop budgets (Controlled Loop Engineering) |
| Credential leakage | Tokenized payment methods; secret scanning in CI |
| Replay / double charge | Idempotency keys, one-time approval tokens |

## 5. Protocol landscape

Emerging open protocols for agent-initiated payments (for example Google’s Agent Payments Protocol and the Agentic
Commerce Protocol from OpenAI and Stripe) share the mandate-and-approval pattern above. Evaluate them against these
guardrails before integrating; no protocol is integrated today.