Quality-Cost Matrix

The Quality-Cost Matrix answers the central routing question before any call is made: which tasks actually need an expensive model and which tasks are wasted spend at the premium tier? Each workflow step is classified along two axes — required quality/risk level and the minimum model tier sufficient to meet that quality bar — so that e6-routing-policy has a step-level look-up table rather than a per-run judgment call. The hard protection rule: any step mapped to a row in s1-failure-mode-register with severity critical or high may not be tier-demoted without a written waiver co-signed by the project owner and an evaluator from s2-evaluator-roster.

Model Tier Definitions (from glossary)

Tier	Description	Typical use
premium	Highest reasoning, context, and generation quality; highest cost and often highest latency	Complex multi-step reasoning, legal/compliance language, sensitive escalation drafts
standard	Strong general quality; cost roughly 5–20x lower than premium	Structured extraction, document summarization, moderate-complexity generation
cheap	Fast, low-cost; suitable for classification and simple retrieval-augmented generation	Intent classification, slot-filling, FAQ-style lookup, simple format conversion
deterministic	Rule engine, regex, lookup table, or scripted logic; zero model cost	Schema validation, exact-match routing, static rule application, threshold checks

Step Classification Schema

```
- step_id: <kebab-case; matches the process map or workflow step id>
  step_description: <one sentence>
  quality_requirement: <high | medium | low>
  failure_mode_ids: [<list of failure_id from s1-failure-mode-register>]
  max_severity_present: <critical | high | medium | low | info>
  minimum_tier: <premium | standard | cheap | deterministic>
  demotion_allowed: <true | false>
  demotion_waiver_ref: <canonical id of the waiver record, or "none">
  rationale: <one sentence explaining the tier assignment>
  notes: <any special caching, batching, or routing modifier>
```

Rule: demotion_allowed: false whenever max_severity_present is critical or high. Attempting to override this field without a demotion_waiver_ref that resolves to a real, co-signed record is a gate-failing violation. The routing policy enforces this at call time; the matrix is the declarative record of intent.

Step Quality Classification Table

The following table summarizes the logic for assigning a step to a tier. It is not a substitute for the per-step YAML above — use both.

Quality requirement	Typical signal	Minimum tier
High — critical/high failure mode	Step appears in s1-failure-mode-register at critical or high; output is irreversible, regulatory, or user-facing without review	premium
High — judgment-heavy, recoverable	Multi-hop reasoning, comparative analysis, ambiguous input; recoverable if wrong	standard
Medium — structured extraction	Field extraction from a defined schema; output reviewed before action	standard or cheap depending on schema complexity
Low — classification	Binary or small-enum classification with observable ground truth	cheap
None — lookup or rule	Exact-match, threshold comparison, static transform	deterministic

Worked Example: Point Preserve STR Compliance Assistant (illustrative)

Point Preserve (725 J D Miller Road, 30A) operates as a short-term rental campus under South Walton County STR regulations. The compliance assistant handles intake, classification, permit-status lookup, and guest communication drafts. All cost figures are (illustrative).

```
# Step: classify incoming guest inquiry by topic
- step_id: guest-inquiry-classify
  step_description: Classify a free-text guest inquiry into {booking, permit-question,
maintenance, noise, other}
  quality_requirement: low
  failure_mode_ids: [ambiguous-intent]
  max_severity_present: low
  minimum_tier: cheap
  demotion_allowed: true
  demotion_waiver_ref: none
  rationale: Misclassification is observable; correct tier is cheap classifier + fallback
to standard on low confidence.
  notes: If intent_score delta < 0.15, escalate to standard tier before routing; see e6-
routing-policy threshold table.

# Step: look up current STR permit status via county API
- step_id: permit-status-lookup
  step_description: Query the county permit database for the active STR permit status of a
given parcel ID.
  quality_requirement: low
  failure_mode_ids: [stale-data, connector-auth-failure]
  max_severity_present: high
  minimum_tier: deterministic
  demotion_allowed: false
  demotion_waiver_ref: none
  rationale: Pure API lookup — no model required. Stale data and auth-failure failure modes
```

```
are high-severity but the mitigation is connector-level (retry/escalate), not model-tier-
dependent.
  notes: connector auth-failure triggers C8 HITL event independently; tier does not change
that path.

# Step: draft applicant-facing response citing STR ordinance
- step_id: str-ordinance-response-draft
  step_description: Draft a guest or applicant reply citing the applicable South Walton
County STR ordinance section.
  quality_requirement: high
  failure_mode_ids: [hallucinated-citation, unsafe-action-attempted, false-success-report]
  max_severity_present: critical
  minimum_tier: premium
  demotion_allowed: false
  demotion_waiver_ref: none
  rationale: Hallucinated citation is critical-severity; output is user-facing and cites
legal text. Premium tier is the floor.
  notes: Output is always routed through clerk HITL gate before delivery; even premium tier
does not waive the HITL.

# Step: extract structured fields from a submitted permit application
- step_id: permit-application-field-extract
  step_description: Extract {parcel_id, applicant_name, proposed_use, square_footage} from
an uploaded PDF application.
  quality_requirement: medium
  failure_mode_ids: [schema-drift-input, false-success-report]
  max_severity_present: high
  minimum_tier: standard
  demotion_allowed: false
  demotion_waiver_ref: none
  rationale: Schema-drift-input and false-success-report are high-severity; cheap models
miss edge cases in complex PDF layouts.
  notes: Post-extract schema validation is deterministic; only the extraction step itself
is standard-tier.

# Step: check extracted fields against static validation rules
- step_id: permit-field-validation
  step_description: Validate extracted permit fields against the declared schema (required
fields present, types correct, parcel ID format matches county format).
  quality_requirement: low
  failure_mode_ids: [schema-drift-input]
  max_severity_present: high
  minimum_tier: deterministic
  demotion_allowed: false
  demotion_waiver_ref: none
  rationale: Validation is pure rule application — no model cost. High severity is handled
by a hard stop and escalation, not by tier.
  notes: Validation failure triggers C8 HITL event.

# Step: summarize permit history for a parcel into a 3-sentence brief
- step_id: parcel-history-brief
  step_description: Summarize the last three permit actions on a parcel into a clerk-facing
brief.
  quality_requirement: medium
  failure_mode_ids: [stale-data, hallucinated-citation]
  max_severity_present: high
  minimum_tier: standard
  demotion_allowed: false
  demotion_waiver_ref: none
  rationale: Hallucinated citation is high-severity; cheap models produce fluent but
ungrounded summaries on sparse historical data.
  notes: Source confidence_band on permit-history source should be verified as 'high'
before calling; if medium, cap applies per s2-scoring-system.
```

Tier distribution for this workflow (illustrative)

Tier	Steps assigned	Estimated % of model API spend
premium	1 (ordinance draft)	~38%
standard	2 (field extract, history brief)	~45%
cheap	1 (inquiry classify)	~6%
deterministic	2 (permit lookup, field validation)	~0%

A well-tuned workflow routes the majority of volume through cheap and deterministic tiers, reserving premium for the few steps where the failure-mode register demands it. In this example the ordinance-draft step is low-frequency but high-cost per call, which is the correct trade-off: a single hallucinated citation is more damaging than the per-call premium cost.

Demotion Waiver Process

When a project owner believes a critical/high step can be safely served at a lower tier — for example, because a downstream HITL gate provides a safety net that makes the quality requirement effectively medium — they must:

1. File a waiver record as a named artifact (e.g., e6-demotion-waiver-<step-id>) with: the step id, the original tier, the proposed tier, the failure modes present, the compensating control (e.g., mandatory HITL gate before output reaches user), the evaluator who reviewed it, and the review date.
2. Update demotion_waiver_ref in this matrix to point to that artifact id.
3. Notify the e6-routing-policy owner to reload the matrix before next deploy.

The waiver does not remove the HITL requirement — it only permits a lower-cost model to do the generation, with the HITL gate as the quality backstop. Waivers expire on the same cadence as the s1-failure-mode-register review cycle (or project milestone, whichever is sooner).