How to Use LangSmith Online Evals as an Agent Quality Monitoring Retainer

LangSmith online evaluation can be packaged as a paid monitoring layer: sampling rules, evaluators, alert thresholds, and monthly regression reports that justify ongoing retainers.

Source https://yetyield.com/blog/langsmith-online-evals-as-agent-quality-monitoring-retainer/

Most AI “consulting” fails because it ends when the prototype works.

Agent businesses fail later, when the prototype stops working quietly.

That gap is exactly what a monitoring retainer is for.

LangSmith is useful here not as a dashboard, but as a way to attach evaluators to production runs so you can measure quality over time and detect regressions.

The monetization angle

Treat “online evals” as the measurement engine for a paid retainer:
• you define what “good” means (evaluators)
• you decide how much to evaluate (sampling + filters)
• you report trends and incidents (monthly scorecard)
• you sell fixes as scoped sprints (implementation revenue)

This is how you turn evaluation into yield.

If you want the productized foundation first:
• How to Turn Agent Evaluation Checklists Into a Paid Product (/blog/how-to-turn-agent-evaluation-checklists-into-a-paid-product/)
• Agent Regression Tests Can Be a Retainer Business (/blog/agent-regression-tests-as-a-retainer-business/)

What LangSmith gives you (in business terms)

LangSmith describes offline evaluation (“test before you ship”) and online evaluation (“monitor in production”).
Official entry point: https://docs.langchain.com/langsmith/evaluation

Translate that into an offer:
• Offline evals are what you sell to get the first check (audit + gate).
• Online evals are what you sell to keep getting checks (monitoring retainer).

A retainer blueprint that does not feel like “tooling”

Step 1: define a small set of evaluators

Don’t start with 20 metrics.

Start with the few that map directly to money-risk:
• Task completion / correctness (did the workflow actually succeed?)
• Tool use quality (tool selection + argument validity)
• Groundedness / hallucination risk (when context exists)
• Format compliance (schema, JSON, required fields)
• Cost ceiling compliance (stop-loss for runaway tokens)

Use a mix:
• deterministic evaluators (format, schema, exact checks)
• LLM-as-judge evaluators (quality rubric scoring)

Step 2: set sampling rules that match the economics

If you score every run, you will blow your budget.

Sampling is part of the product:
• sample a fixed percentage of runs
• oversample high-risk workflows
• always evaluate error paths and tool-call failures

Your retainer can literally include:

“We evaluate 10% of runs + 100% of failures.”

That’s a crisp scope.

Step 3: define what triggers a “paid fix”

Most monitoring services fail because they don’t tie signals to actions.

Define intervention thresholds:
• quality score drops below X for 3 days
• tool-call success drops below Y
• hallucination/groundedness failures exceed Z per week

Then package fixes:
• prompt boundary repairs
• tool schema fixes
• evaluation dataset expansion
• guardrail additions (block/mask/deny topics)

The monthly deliverables that make it billable

Deliverables turn “monitoring” into a product.

Include these:
• scorecard (trend lines for top evaluators)
• top 10 failure modes (ranked by business impact)
• “new eval cases added” log (your compounding moat)
• recommended fix list + estimate

This avoids the trap of being perceived as “just an observability subscription.”

Positioning: sell the outcome, not the platform

Good positioning sounds like:

We keep your agent’s quality stable as you change prompts, models, and tools.

Not:

We set up LangSmith.

LangSmith is the implementation detail.

The paid product is “quality continuity.”

Where this fits in the YetYield map

This retainer model is the operational version of what credential funnels do for individuals: proof !’ paid translation !’ ongoing support.

If you want a parallel example from the credential world:
• Microsoft Applied Skills: Turn Lab-Based Agent Credentials Into Paid Enablement Sprints (/blog/microsoft-applied-skills-lab-assessment-as-enablement-sprint/)

What to avoid

• dashboards without evaluator definitions
• evaluation scores without thresholds and escalation rules
• “we monitor everything” promises without sampling economics

Next up in this cluster:
• how to implement a quality gate using Vertex GenAI evaluation service when the buyer is on Google Cloud