Repo-Data-Factory for S2AND: Dataset & System Design

1. Purpose and Evaluation Alignment

This project is a training-data factory that turns a local repository into **grounded instruction data** for two complementary tasks. The design is guided by four review criteria: scenario coverage and logical correctness; effective and automatable data processing; architectural completeness and extensibility; and clear, compliant examples—especially the provenance of the reasoning trace.

The system deliberately separates **what is deterministically knowable** from **what must be phrased and explained well**. Rules extract verifiable anchors; the LLM edits or composes responses under strict grounding constraints. This division is the core quality lever: correctness comes from deterministic extraction, while readability and pedagogy come from controlled generation.

---

2. Scenario Coverage

Scenario 1 targets "QA over repo rules/behaviors." It is designed for questions that can be answered directly from code facts: invariants, argument expectations, error behaviors, configuration flows, and similar concrete properties. The pipeline is **rule → draft sample → LLM rewrite**. Rules decide *what is worth asking*; the LLM is used as an editor, not an oracle.

Scenario 2 targets "architecture-anchored design tasks." It is designed for questions that require explaining intent, constraints, and evolution plans, but must remain grounded. The pipeline is **architecture anchors → evidence bundle → (optionally) LLM question synthesis → answer + trace**. Unlike Scenario 1, Scenario 2 can ask the model to *formulate* a precise design question from evidence—because architectural understanding is often better elicited by "why / tradeoff / evolution" prompts than by literal "what does this function do."

Together, the scenarios cover both ends of the spectrum: **behavioral truth** (Scenario 1) and **design reasoning under constraints** (Scenario 2). The shared schema keeps the dataset uniform while the metadata preserves scenario-specific semantics.

3. End-to-End System Architecture

The factory is composed of three layers:

(a) Extractors (rules): deterministic scanners that identify “question-worthy” facts or architectural anchors and assemble minimal drafts plus evidence spans.

(b) Editors (LLM enrichment): a strict rewriting stage that produces the final question/answer and a bounded reasoning trace, while obeying grounding rules.

(c) Validators and resumable writers: streaming JSONL output, idempotent resumption by id, and schema/grounding validation that rejects non-conforming samples.

The key extensibility property is that new rules can be added without changing the LLM contract. Scenario 2 anchors can also evolve (e.g., from “entrypoint” to “dataflow hub”) without schema changes; only rule extraction and prompts need adjustment.

4. Training Sample Schema (JSONL) and Field Semantics

Each line is a QAItem. The schema is intentionally compact yet audit-friendly: it can train a model and also support debugging, filtering, and ablation studies.

Field	Type	Meaning (Model-Facing vs System-Facing)
id	string	Stable identifier for dedup/resume (system)
scenario	string	scenario1 / scenario2 (system + analysis)
rule_id	string	Which extractor produced the draft (system)
title	string	Human-readable tag of what the sample is about (system + curation)
question	string	The instruction to the model (model-facing)
answer	string	The expected response; structured natural language (model-facing)
evidence	list	Grounding snippets with file + line span (model-facing constraints)

Field	Type	Meaning (Model-Facing vs System-Facing)
trace	list	Bounded rationale steps that cite evidence_refs (model-facing)
meta	object	Anchors, flags, warnings, attempts (system + quality control)

Evidence objects are immutable excerpts:

Evidence field	Meaning
span.file_path	Repository-relative path
span.start_line, span.end_line	Line range for auditability
snippet	Verbatim excerpt; the only admissible factual source

Trace objects encode an auditable reasoning path:

Trace field	Meaning
step	Monotonic index
kind	One of extract, reason, answer
content	Natural language rationale for the step
evidence_refs	Indices into evidence[] (non-empty)

This trace is **not free-form chain-of-thought**. It is an explicit citation graph over the evidence list, enforced by validation. When evidence is insufficient, the answer must say “Insufficient evidence” and list missing information; this is treated as a compliant outcome, not a failure.

---

5. Scenario-Specific Semantics

Scenario 1 (Rule + LLM, question fixed): rules generate a draft question and answer from a concrete fact pattern; the LLM rewrites only for clarity and completeness. The question is treated as non-editable (except whitespace normalization). This preserves logical correctness: the dataset teaches the model to answer *the actual repo-derived question*, not a softened paraphrase.

Scenario 2 (Evidence → question/answer/trace): rules provide architecture anchors and supporting evidence. Depending on metadata, the LLM either (i) keeps the question as-is (requirement-driven) or (ii) generates a precise engineer-style question from evidence (intent-driven). This allows architectural prompts to be natural (“Why is there a strategy switch here?”) while still grounded.

6. Data Quality Controls, Diversity, and Representativeness

Quality is enforced structurally rather than heuristically. Every enriched sample must pass: valid JSON; required fields present; trace steps cite valid evidence indices; and (when multiple evidence snippets exist) trace coverage must cite multiple snippets. Failures are written with diagnostics to an error log, enabling iterative tightening.

Diversity is engineered through **rule portfolios and anchor variety**, not through random sampling. Scenario 1 rules target different behavioral categories (configuration, errors, interfaces, serialization, concurrency). Scenario 2 anchors cover different architectural loci (entrypoints, pipelines, extension points, evaluation strategies, data assets). Representativeness is handled by balancing across files, modules, and anchor kinds, and by retaining "Insufficient evidence" cases as a legitimate class—because real repos contain partial context and ambiguous design signals.

7. Training Plan (and Why We Did Not Train)

The intended fine-tuning approach is **QLoRA on Qwen2.5-7B**, using the JSONL records as supervised instruction samples. The training objective would emphasize: (1) evidence-bounded answers; (2) stable instruction following; (3) producing structured responses; and (4) generating traces that cite evidence.

In this iteration, **we did not perform a full training run**. The primary reason is that the dataset quality is not yet stable enough: some anchors are weak (e.g., non-architectural files mislabeled as extension points), and a portion of LLM enrichments fail validation or produce shallow traces. Training on inconsistent supervision would risk reinforcing undesirable behaviors (overconfident inference, template answers, or brittle formatting). Instead, this phase is treated as a system and schema validation pass: ensuring the factory can reliably produce compliant samples before spending compute.

8. Strengths and Known Issues

The main advantage of the design is that it treats grounding as a first-class invariant. Evidence spans are immutable, trace is citation-based, and

“Insufficient evidence” is an explicit, acceptable output—together these properties produce data that is more auditable than typical instruction corpora.

The main limitation is that architectural intent is inherently underdetermined in many code snippets; if anchors are coarse or evidence extraction is shallow, Scenario 2 becomes either speculative or vacuous. This is not a prompting problem alone: it requires better anchor selection, richer evidence bundling (adjacent definitions, call sites, and docstrings), and stricter rule precision. The system architecture supports these improvements without schema changes, which is the intended path forward.

In summary, the factory is built to scale in correctness before scale in volume: deterministic extraction defines what may be claimed; the LLM is constrained to produce teachable, structured outputs; and the schema preserves provenance and traceability. The next milestone is not “more data,” but **more consistently meaningful evidence bundles**—so that both scenarios become reliably informative and trainable.