Evidence-carrying Los Angeles city and county permit intake in Lean

Matha Labs Research | Version 0.1 | November 15, 2026

Abstract

Municipal permitting combines applicant descriptions, jurisdictional boundaries, changing filing processes, professional judgment, and public-authority decisions. A language model can help interpret a project description, but its answer alone is a weak foundation for an automated operational decision. This paper presents a bounded Lean 4 implementation that separates proposed facts from deterministic intake routing. The prototype covers new-building intake in the City of Los Angeles and unincorporated Los Angeles County, with a first-class distinction between their authorities. It validates a registered policy revision, calendar date, model applicability window, jurisdiction attestation, scope attestation, evidence span, and four model-specific intake references. Its only outcomes are a department intake route and a request for review. Fourteen theorems establish the conditions required for successful routing, and 30 synthetic fixtures exercise accepted requests and rejected mutations. The implementation demonstrates a reproducible checking boundary; it does not establish legal completeness, automate plan examination, or issue permits.

1. Research question

The useful question is narrower than whether a model can understand every city code: can a proposed routing decision be accompanied by enough structured evidence for a small, explicit checker to reproduce it and reject unsupported inputs? Such a checker can be reviewed independently of the system that reads applicant prose.

This distinction matters because deterministic execution and factual correctness are different properties. A program may reproducibly follow an incorrectly transcribed rule. A quoted sentence may accurately occur in a document while being irrelevant, hypothetical, or contradicted elsewhere. A jurisdiction label may be syntactically correct while identifying the wrong municipality. The prototype therefore exposes these dependencies instead of claiming that formal verification resolves them.

The design is motivated by Lexifina's evidence-carrying approach to legal formalization: an interpretation supplies a certificate that a separate checker evaluates. Our implementation is independently authored and much smaller. We make no comparison of empirical accuracy, no claim to cover a municipal legal corpus, and no assertion of research priority.

2. Los Angeles jurisdiction comes first

The City of Los Angeles and unincorporated Los Angeles County have distinct permitting authorities. The prototype starts by requiring a reviewed jurisdiction determination for the same parcel referenced in the request. A postal city name is insufficient. LA County Planning expressly directs people to its jurisdiction lookup because mailing addresses can name an adjacent city. County jurisdiction guidance.

For a supported City parcel, the route is LADBS intake. City parcel information is reviewed through the official ZIMAS context. City zoning and ZIMAS. The City offers several permit and plan-review processes, and the prototype does not choose among express, counter, expanded-counter, or regular review. LADBS plan review and permitting.

For a supported unincorporated County parcel, the route is LA County Public Works Building and Safety / EPIC-LA intake. Public Works publishes a service locator and an electronic application process. This route does not replace separate planning, environmental, fire, or other required reviews. County Building and Safety, EPIC-LA.

The geographical County also contains incorporated cities. Some receive contracted County services, but this release does not encode their individual arrangements. Every other incorporated city and every unknown jurisdiction returns review. It would be incorrect to route all addresses containing "Los Angeles" to LADBS, or all County parcels to the unincorporated County workflow.

The implemented work category is new building only. Ordinary repairs, additions, demolition, and other scopes return review without an exemption decision. Retained existing elements and flagged special conditions also trigger review. These are conservative model exclusions, not assertions that the City and County classify every project identically.

3. Data and certificate model

`Jurisdiction` distinguishes City of Los Angeles, unincorporated LA County, other incorporated cities, and unknown jurisdiction. `Scope` includes new building and several explicitly unsupported categories. `IntakeRoute` has two values: LADBS and County Public Works / EPIC-LA. `routeFor` returns no route for unsupported jurisdictions. There is no constructor for an approved permit, a fee paid, code compliance, or permission to begin work.

Each `Policy` contains its jurisdiction, source identifier, revision, and applicability interval. `policyFor` provides the compiled registry. Before routing, the checker requires equality of the entire supplied policy with the registered policy for the request's jurisdiction. This prevents a caller from reusing a County record for a City parcel or extending an applicability window while retaining a familiar source identifier.

The policy revision is `research-la-snapshot-2026-11-15`. The model window starts November 15, 2026, inclusive, and ends December 1, 2026, exclusive. Those dates deliberately bound this research snapshot's use. They are not legal effective dates, enactment dates, guarantees of source freshness, or a model of transitional rights. `ValidDate` checks calendar structure, including month lengths and leap-year handling, before the interval comparison.

A `Request` carries jurisdiction, date, structured scope, source note, evidence span, reviewer flags, and four reference strings: address, parcel, drawings, and applicant. Each reference must contain a non-whitespace character. This verifies presence only. It does not authenticate a parcel, inspect a drawing, or prove professional licensure.

A separate `JurisdictionEvidence` record identifies its evidence category, lookup source identifier, lookup reference, parcel reference, reviewed jurisdiction, and review flag. `JurisdictionVerified` requires a supported jurisdiction, the reviewed-official-parcel-lookup category, a positive review attestation, an identical request parcel and jurisdiction, a nonblank lookup reference, and the registered lookup-source identifier for that jurisdiction. Mailing-address-only evidence cannot satisfy this predicate. The model checks these relationships;

it does not contact a government service or authenticate the supplied result. An actual integration must capture and review authoritative parcel and service-lookup evidence before constructing this input.

The evidence span records a character offset, length, and quote. `Grounded` requires a positive length, a range within the note, and exact equality with the extracted character sequence. The rule excludes fabricated and out-of-bounds quotations. It says nothing about the quote's meaning. `scopeReviewed` is a separate, explicit attestation that an upstream reviewer has checked the structured interpretation.

4. Decision procedure

`Eligible` is a conjunction of nine checks: registered policy, applicable model date, verified jurisdiction evidence, supported scope, no retained existing elements, no flagged special conditions, present intake references, grounded evidence, and reviewed scope.

`evaluate` is a pure function. If eligibility holds, it returns the intake route obtained from the request's jurisdiction. Otherwise it returns `needsReview` with the first failed category. The priority is fixed and visible in source: policy, date, supported jurisdiction, jurisdiction evidence, scope, retained elements, special conditions, references, grounding, and scope review. A request with several problems receives one reason at a time; this release does not aggregate all defects.

For identical compiled source, policy, and request values, evaluation is reproducible. There is no model sampling, network fetch, hidden retrieval state, or floating-point threshold in the checker. Any stochastic extraction takes place before its input boundary. Persisting the complete input alongside the revision is sufficient to replay this model's routing result.

5. Theorem inventory and meaning

The authored source contains fourteen named theorems:

1. `routed_requires_eligible`: successful routing implies the full eligibility conjunction.
2. `routed_uses_registered_policy`: successful routing uses the compiled policy for that jurisdiction.
3. `routed_within_model_window`: the date is valid and inside the model interval.
4. `routed_requires_jurisdiction`: reviewed official parcel-lookup evidence is tied to the same parcel and supported jurisdiction.
5. `routed_requires_supported_scope`: the scope is new building.
6. `routed_excludes_retained_elements`: retained elements are not flagged.
7. `routed_excludes_special_conditions`: special conditions are not flagged.
8. `routed_requires_documents`: all four model references contain non-whitespace text.
9. `routed_requires_grounded_evidence`: the evidence span satisfies the exact-match conditions.
10. `routed_requires_reviewed_scope`: the scope review attestation is present.
11. `routed_to_correct_jurisdiction`: the output route equals the route for the request's jurisdiction.
12. `routed_requires_supported_jurisdiction`: a routed request is in City LA or unincorporated LA County.

13. `mailing_address_alone_cannot_route`: mailing-address-only evidence cannot produce a routed decision.
14. `ineligible_needs_review`: an ineligible request produces the corresponding review result.

These are universal properties of the implemented checker over its typed inputs. Most are consequences of the explicit eligibility guard; they make that acceptance contract inspectable and protect it against accidental weakening. They are not independent proofs that municipal guidance has been correctly or completely formalized. The Lean kernel checks the proof terms, with the Lean environment and its standard foundations in the trusted computing base.

6. Synthetic evaluation

`Main.lean` supplies 30 executable fixtures: three successful routes and twenty-seven review cases. The successful examples include City and unincorporated County requests and the final included date in the model window. Every example uses conspicuously synthetic references and a short authored note. No private applicant records or real addresses are used.

Review cases include City/County policy confusion, other incorporated cities, unknown jurisdiction, an invented revision, a caller-extended interval, invalid or out-of-window dates, unreviewed jurisdiction evidence, mailing-only evidence, lookup evidence for a different jurisdiction or parcel, wrong or missing lookup source/reference, unsupported scopes, retained elements, special conditions, missing and whitespace-only references, fabricated or invalid spans, and an unreviewed scope.

The executable compares each actual decision with a fixed expected decision and exits with failure on disagreement. These cases test boundaries and rejection behavior; their pass rate is not a measure of real-world legal accuracy. No city staff labels, external held-out dataset, filing success metric, or precision/recall estimate is available in this release.

Reproduction requires Lean 4.35.0 and Lake. Run `lake build`, `lake exe fixtures`, and `sh scripts/check-proof-holes.sh`. The project uses `Std` and has no external Lean dependencies. The source scan rejects unchecked declaration markers in authored Lean files; it complements kernel checking rather than replacing it.

7. Trust boundary and limitations

The checker trusts the authored registry and the human interpretation behind it. The source manifest preserves official City and County URLs, the review date, and a short scope statement. Its revision string is not a cryptographic digest of a city website, and the checker does not retrieve, archive, or authenticate live publications. A changed website does not update compiled behavior automatically.

Jurisdiction lookup evidence and scope attestations are externally supplied reviewed inputs. A malicious or mistaken upstream actor can assert them incorrectly. A lookup reference proves only that a string was supplied, not that its contents are authentic or current. Exact quotation checking does not prevent a contradictory note or identify negation. Likewise, `specialConditions = false` means no special condition was flagged by the supplier; it does not prove that a zoning overlay, landmark designation, environmental restriction, or other requirement is absent.

The implementation contains no zoning calculations, occupancy classification, accessibility determination, structural engineering, fire review, trade-permit coordination, fee schedule, license verification, inspections, agency API integration, or issuance logic. Routing must remain visibly distinct from approval in any product built on this code. Other agencies can impose additional requirements even when department intake is appropriate. County contract-city services require their own reviewed jurisdiction-specific rules before they can be supported.

8. Research extension

The next useful extension is not a larger unsupported claim. It is a reviewed source-to-rule workflow: immutable source captures, legally meaningful effective and applicability dates, a documented interpretation process, and independent expert labels. Additional work categories should be admitted one at a time with explicit exceptions and regression fixtures.

A language model could then propose typed scope facts with evidence spans and uncertainty estimates. An external reviewer or authenticated data service could establish jurisdiction and reference authenticity. The Lean boundary would continue to check the resulting certificate and execute the selected rule version. Calibration of the upstream extractor and correctness of the formal acceptance contract should be evaluated separately.

References

- Lean project, installation and toolchain guidance.
- Lexifina, Formalizing all US case law in Lean, November 13, 2026. Inspiration for the evidence-carrying architecture; no code or evaluation data is reused.
- Official City and County materials linked in Section 2 and recorded in `docs/sources.json`, reviewed November 15, 2026.

The MIT license covers this repository's authored material. It does not relicense third-party agency publications.