Specializzare e *scalare* l'IA Generativa nel settore legale

AI Conf 2026
24/06/2026

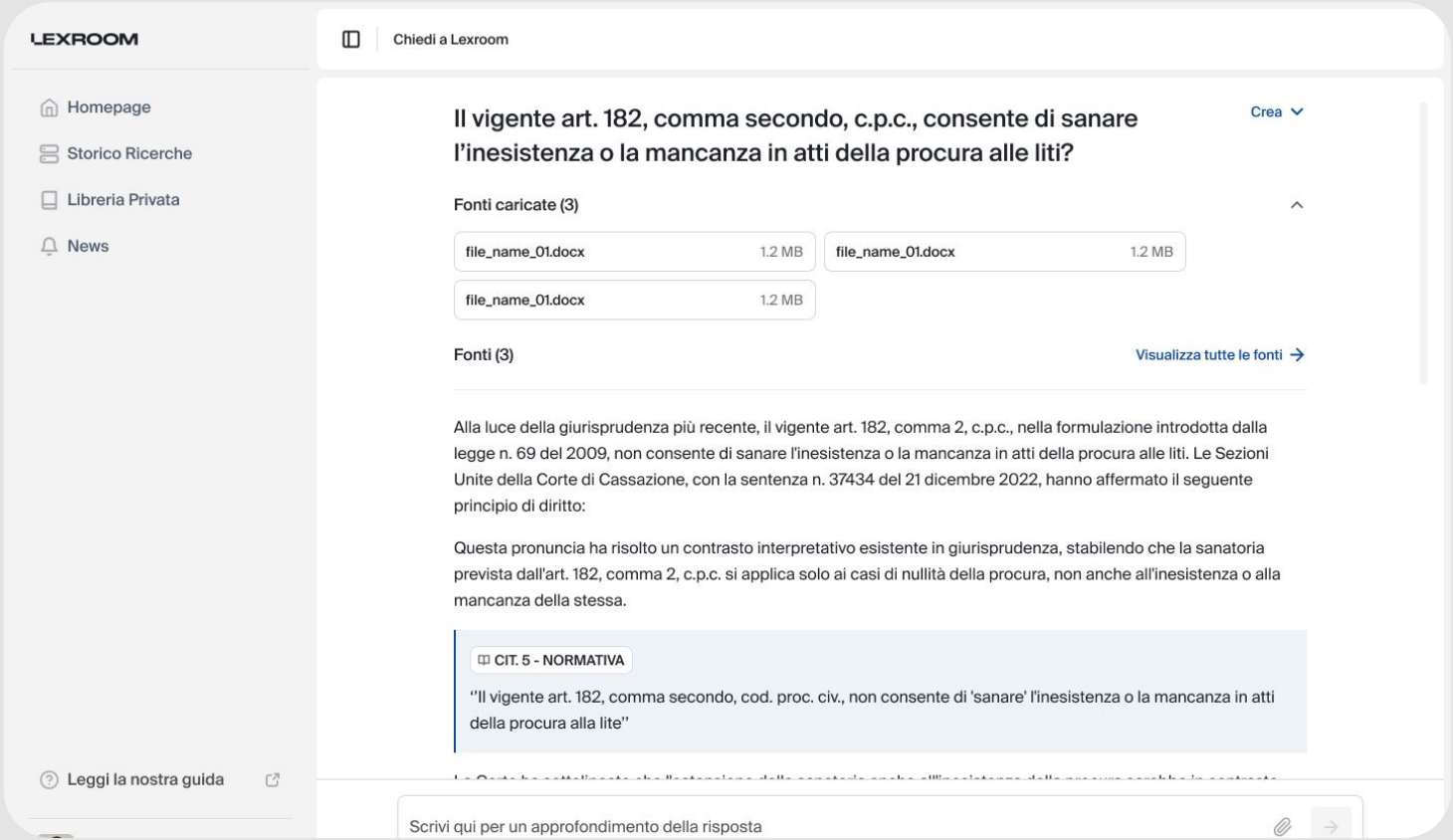
LEXROOM

LEXROOM

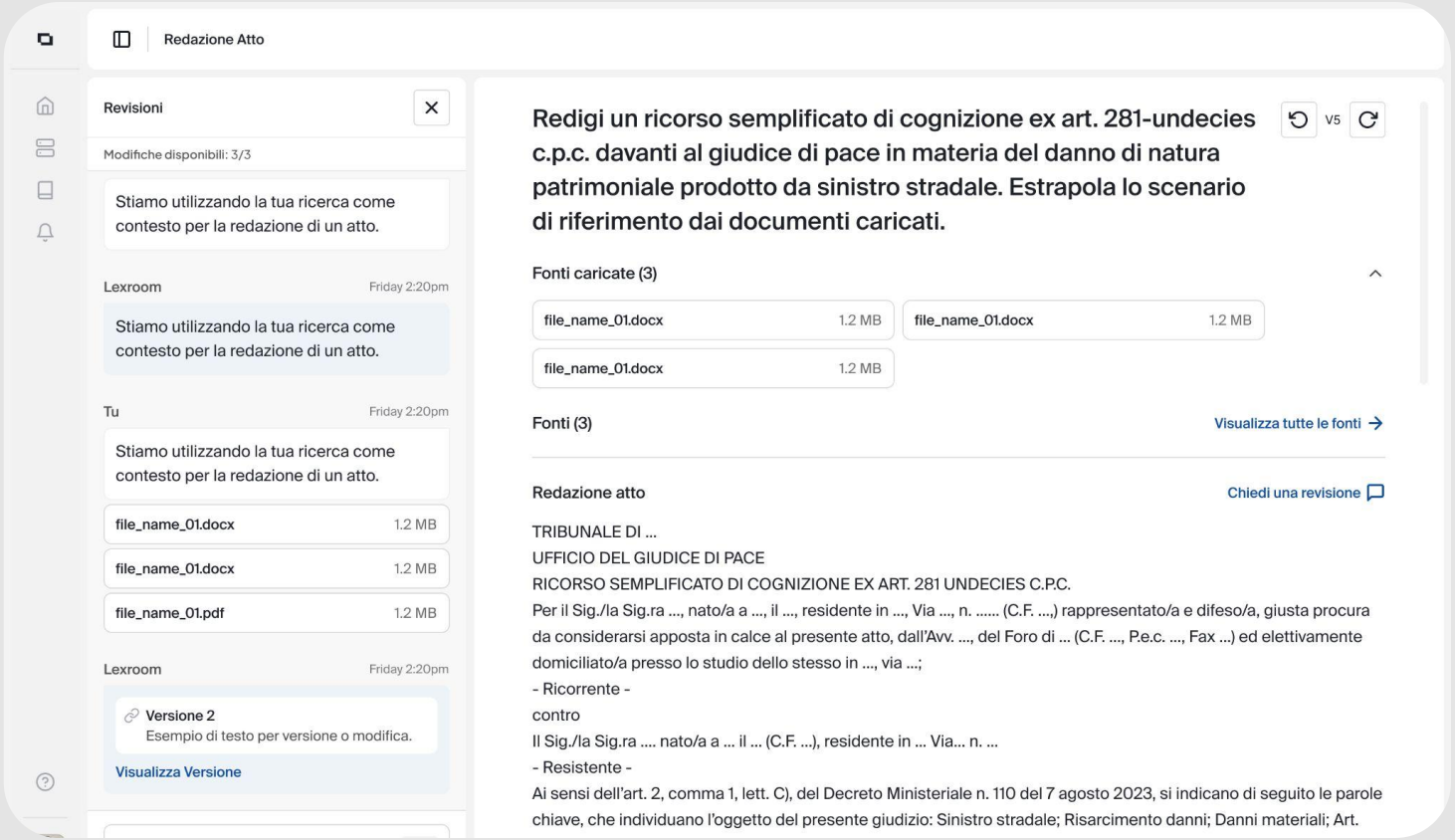
The New *legal* Standard

LEXROOM

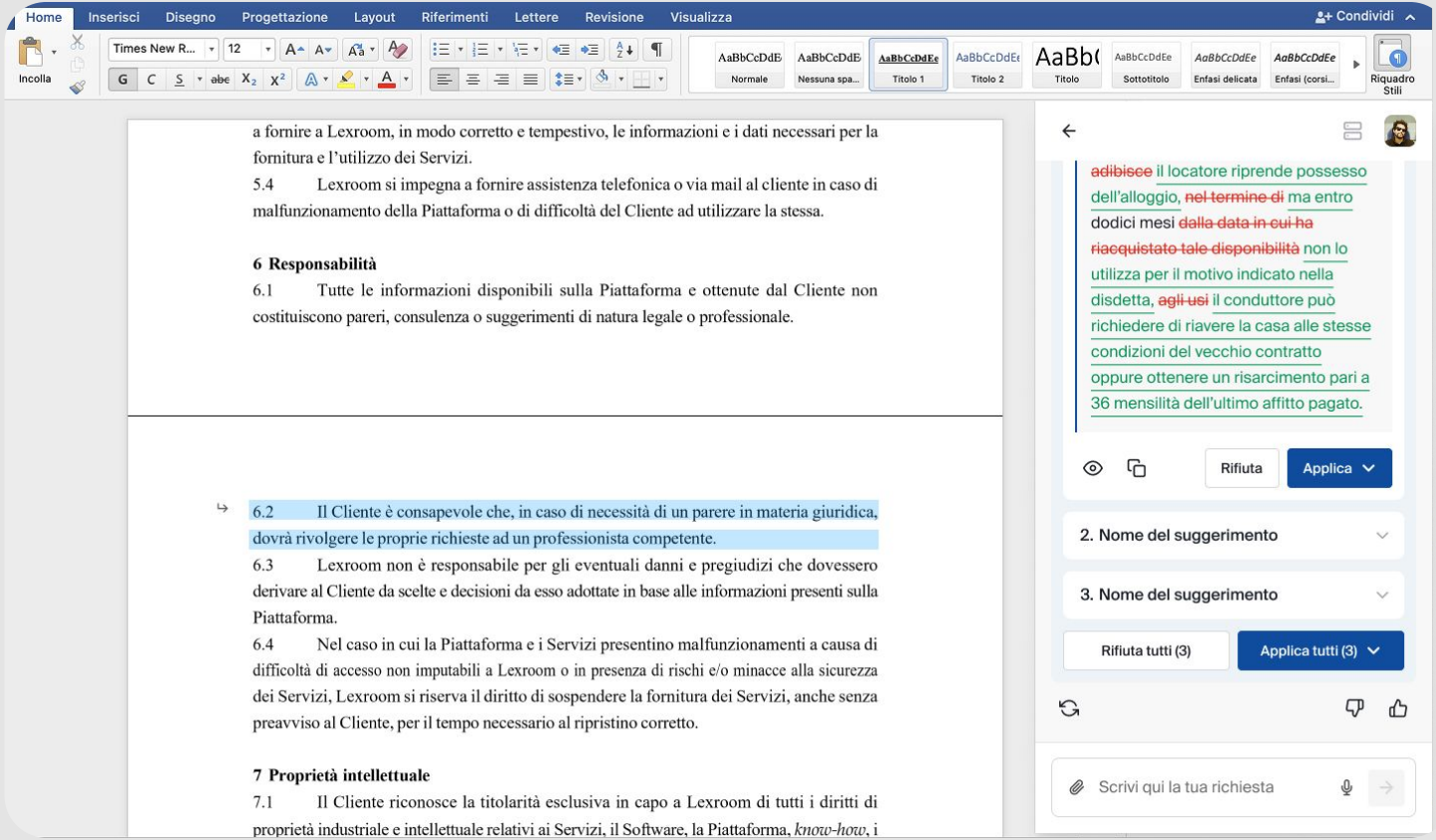
Ask complex questions. Get reliable answers.



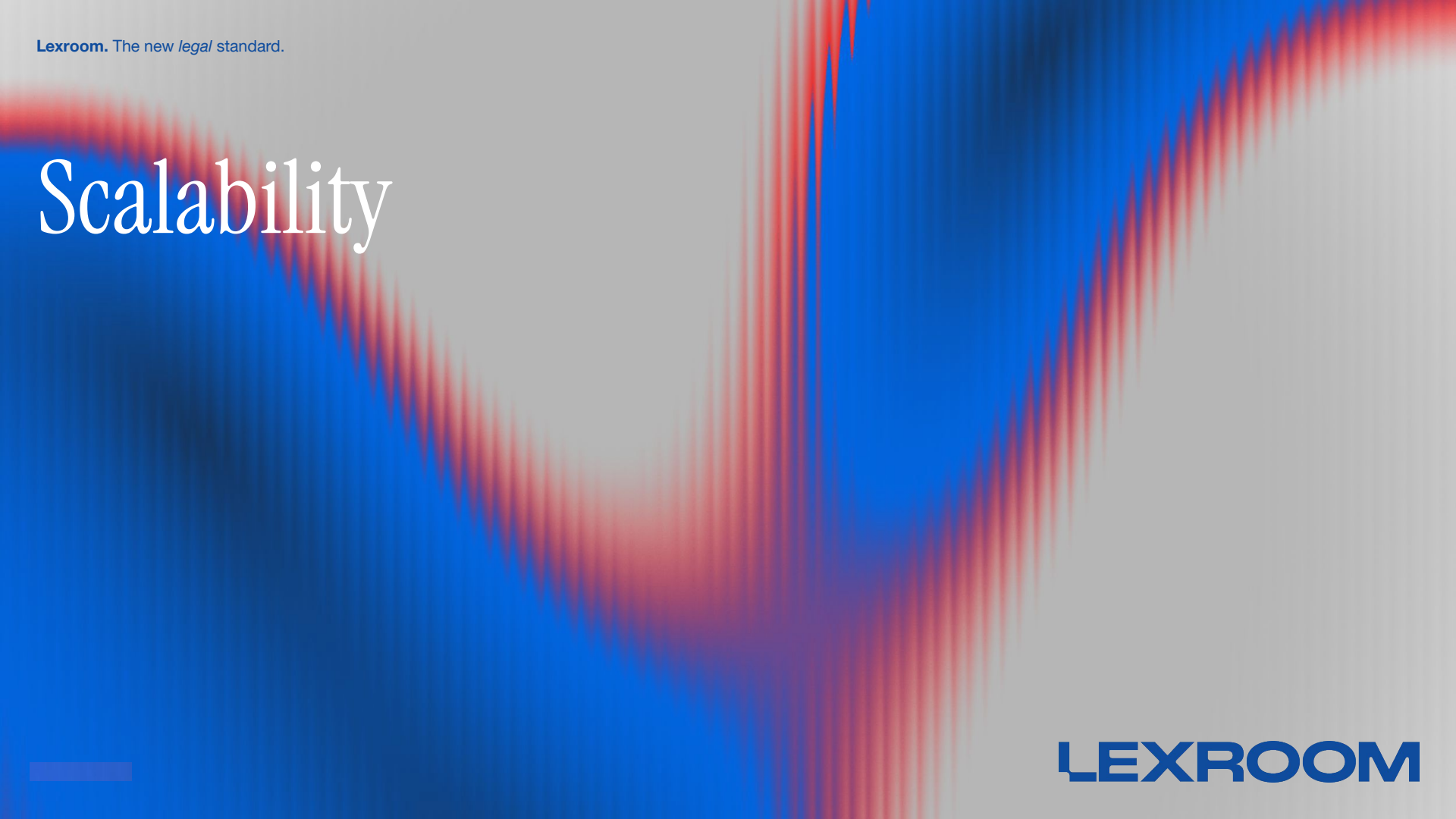
From your ideas to the first draft. In minutes.



Lexroom's intelligence, directly in Word.



Scalability



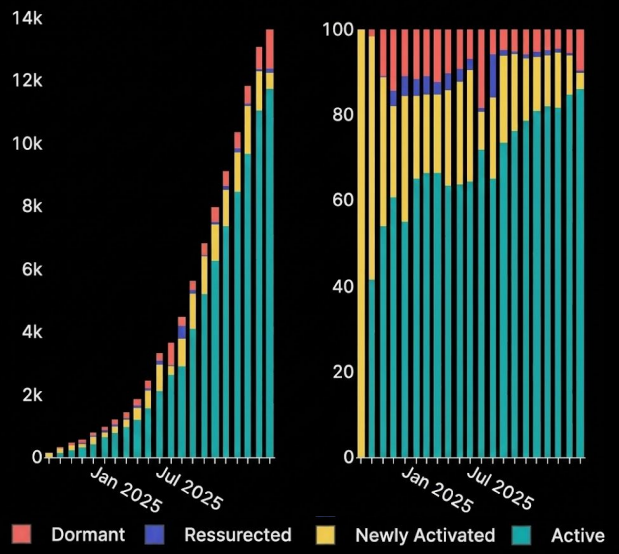
LEXROOM

From a Tool To a habit

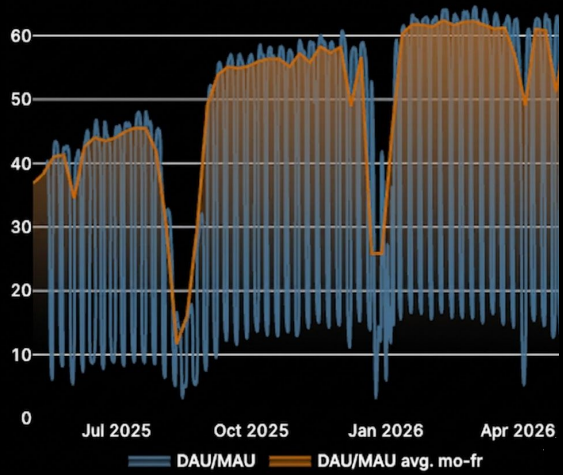
From 35% to +60% DAU/MAU in 12 months.

Across >10k customer.

Monthly Active Users (MAU)



Daily Active Users (DAU)/MAU



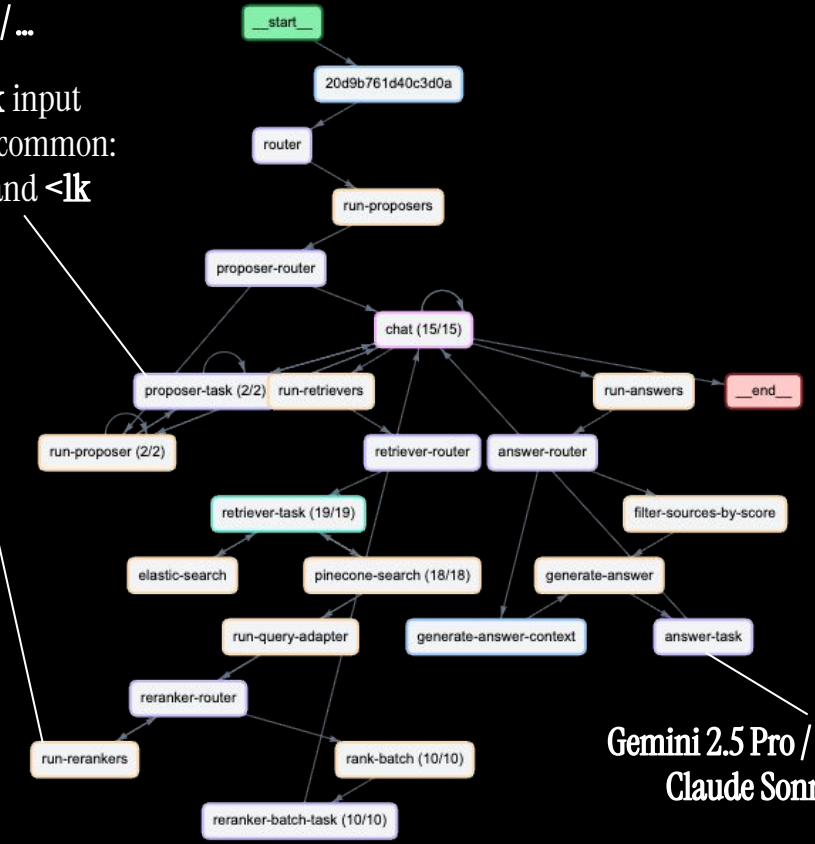
Under the Hood

One user query translates to ~60-70 tasks and subtasks, most of them relying on LLM calls.

Calls can be **very** heterogeneous in terms of prompt and output sizes (e.g. structured output vs. final response).

Gemini 3.1 Flash Lite / GPT-5.4 mini / ...

On average **8k** input tokens (most common: 1-2k at 35%) and **<1k** output tokens



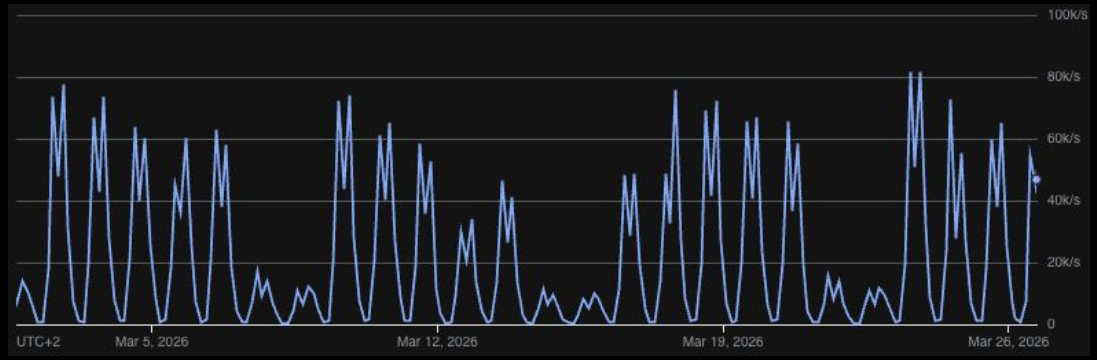
Gemini 2.5 Pro / GPT 5.4 / Claude Sonnet 4.6 / ...

On average **33k** input tokens and **4k** output tokens

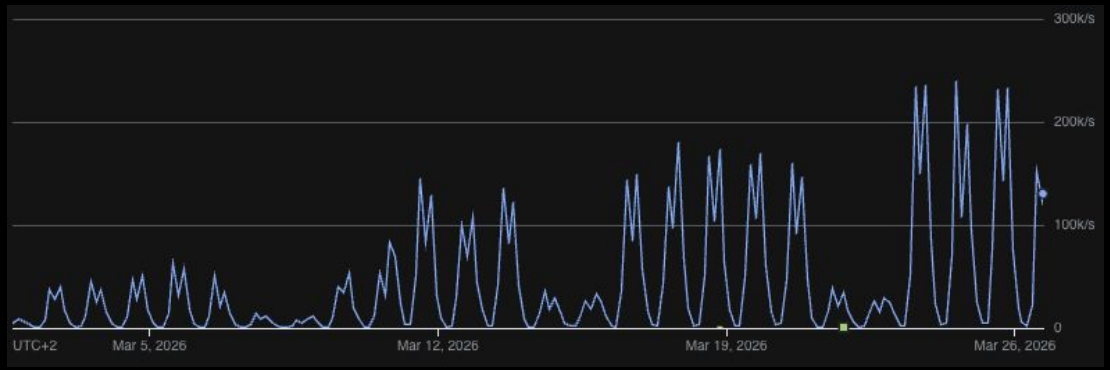
How does this Translate to Token Throughput?

Very "spiky" traffic, roughly matching peak business hours (**70k token/s** for final answer generation, **>200k token/s** for intermediate tasks).

Gemini 2.5 Pro (0.32 avg. QPS, 3.7x peak QPS)



Gemini 2.5 Flash (1.13 avg. QPS, 23.3x peak QPS)



How do we Handle High Traffic Loads?

First of all: a **robust** and **region-aware** handoff between Lexroom LLM orchestration and LLM providers.

A mix of different strategies (all working together):

1. **Routing to multiple regions and providers.** We (a) rely on different LLM providers (Google/GEAP, AWS Bedrock, OpenAI) and (b) distribute requests across all supported EU regions (**7** in Google/GEAP, **4-6** regions in AWS)

2. **Smarter retries** . Many LLM provider SDK includes native retry behavior, but generalized solutions (like Tenacity in Python) work better with 429 Resource Exhausted errors thanks to techniques like exponential backoff and jitter

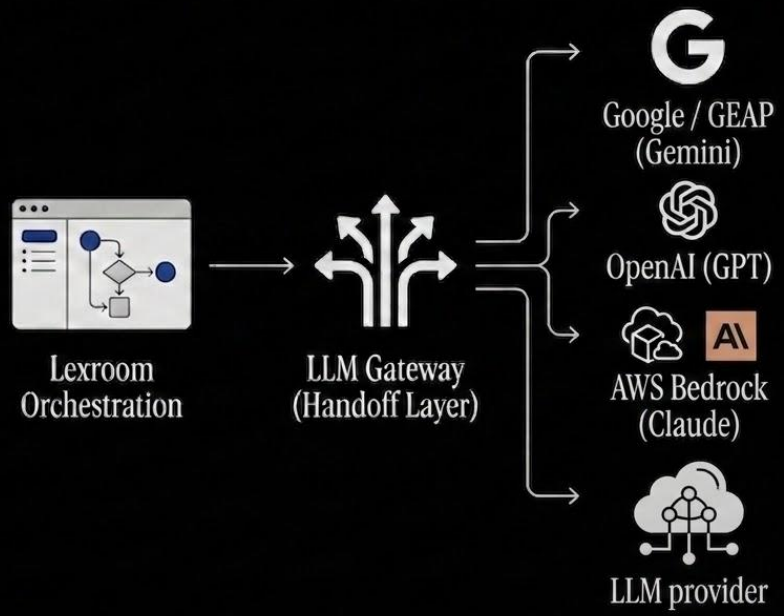
3. **Fallback models.** All LLM requests have a primary model *and* a chain of fallback models. Fallbacks get triggered in sequence when (a) a request fails or (b) a request reaches a predefined timeout

4. **Scale down latency-impacting parameters (e.g reasoning effort)** . Latency and stability are highly affected by size of the input and thinking budget/reasoning effort. With high-traffic scenarios and/or fallback requests we reduce both.

How do we Handle High Traffic Loads?

First of all: a **robust** and **region-aware** handoff between Lexroom LLM orchestration and LLM providers.

We use our own **LLM Gateway** to (1) implement the handoff layer as *decoupled* from our agentic workflows, (2) have full control on *how* we interact with LLM providers, and (3) take care of all the bookkeeping (e.g. tracking of costs, token throughput, latencies per model)



How do we Handle High Traffic Loads?

Last resort: **Provisioned Throughput** to reserve a certain fraction of token throughput.

Most LLM providers (Google GEAP, AWS Bedrock) have cost models that revolve around two different paradigms:

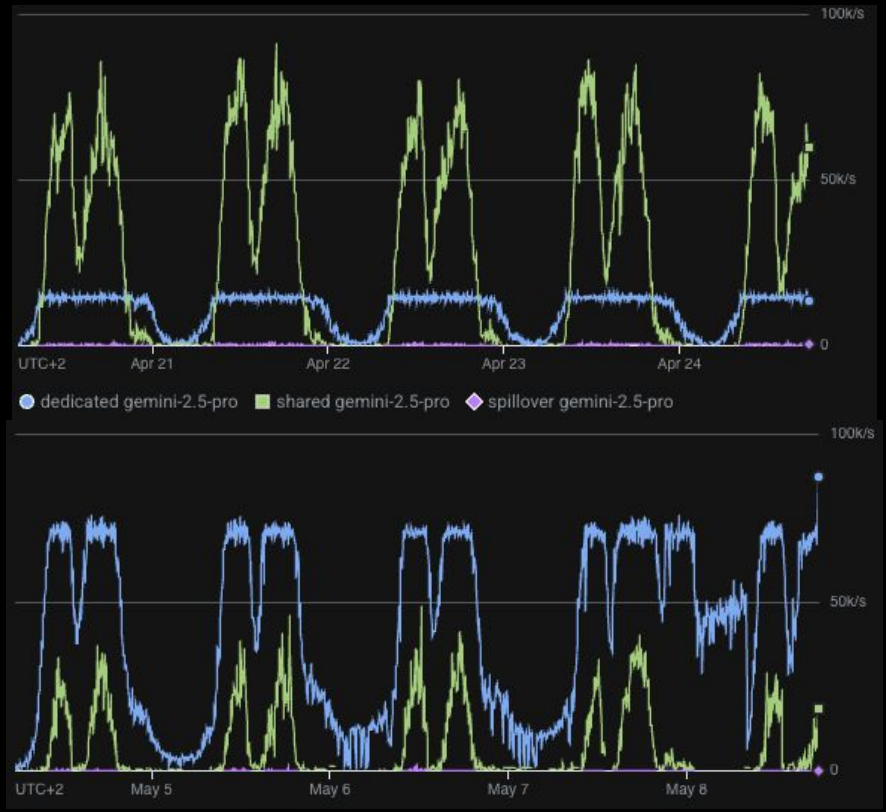
- **Pay-as-you-go** (token-based, usually billed per **1M** input/output tokens)
  - ✓ Costs scale flexibly with the token throughput
  - ✗ Treated with lower priority by the provider, uses **shared** endpoints so it is subject to resource availability

- **Provisioned throughput** (resource-based, usually billed based on a measure of *guaranteed* token throughput)
  - ✓ Provider reserves **dedicated** hardware, so treated with highest priority until the provisioned throughput is reached
  - ✗ Fixed monthly cost, depends on the minimum throughput to guarantee

How do we Handle High Traffic Loads?

Last resort: **Provisioned Throughput** to reserve a certain fraction of token throughput.

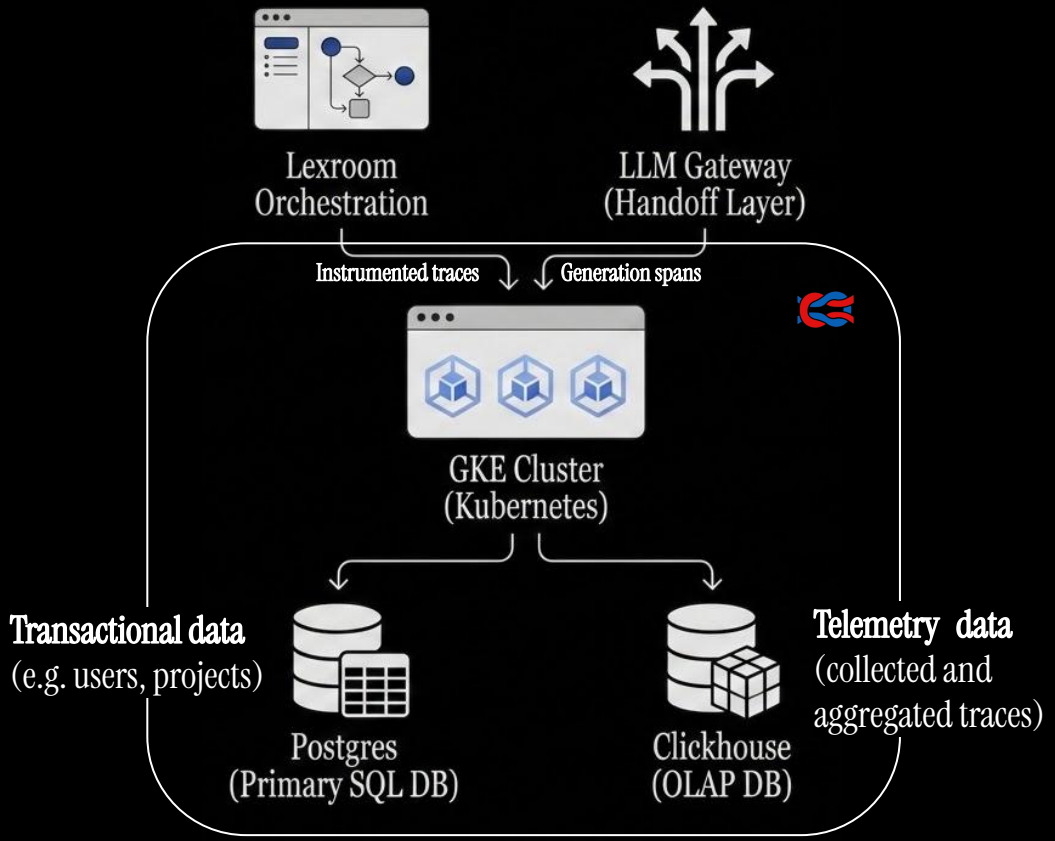
22 GSU (0.2 QPS, 14k token/s) + ~80% PayGo



109 GSU (1 QPS, 70k token/s) + ~30% PayGo

LLM Observability at Scale

A self-hosted **Open Telemetry** (**OTel**) framework (**Langfuse**) to monitor logs, track metrics, in-depth debugging whenever needed.

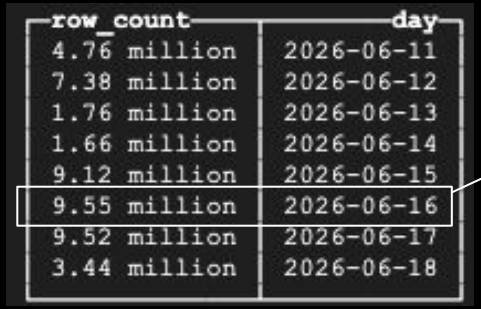


LLM Observability at Scale

An **Open Telemetry** (**OTel**) framework to monitor logs, track metrics, in-depth debugging when needed

Why **self-hosted** ?

1. **Better scalability** . Cost increases with infrastructure sizing and maintenance effort (vs. trace-based or span-based pricing)



row_count	day
4.76 million	2026-06-11
7.38 million	2026-06-12
1.76 million	2026-06-13
1.66 million	2026-06-14
9.12 million	2026-06-15
9.55 million	2026-06-16
9.52 million	2026-06-17
3.44 million	2026-06-18

60% of the **monthly** amount of spans generated (in November 2025)

2. **Full control over telemetry data** . Important for compliance: we decide on data retention and per-user deletion (GDPR), we can implement audit logs, we can handle span anonymization and PII removal (1) asynchronously, and (2) with our own anonymization models

Privacy and Compliance at Scale

We do **not** compromise on privacy and compliance, *even when* it gets in the way of scalability.

We ensure GDPR compliance through three core pillars:

EU Data Sovereignty: We route production LLM requests exclusively through *regionalized*, EU-only endpoints with *Zero Data Retention* (ZDR).

PII and Anonymization: We strip user document content and apply anonymization on generation spans asynchronously and across languages.

No Training on User Data: We never use user data for training, fine-tuning, or model evaluation.

Specialization

LEXROOM

Why not just a *generalist* LLM?

(with clever prompt engineering and/or a few Markdown files)

Specializing LLMs to the Legal Domain

A handful of instructions is not enough; we need a **legal-specific harness** .

From seminal evaluation studies on **Legal AI systems** (not even generalist LLMs) we learn that basically *every single task* in a legal workflow (search, rank, etc.) require *in-depth legal reasoning*, and can easily fail when offloaded to generic tools.

TABLE 6 | This table shows the prevalence of different contributing causes among all hallucinated responses for each model. Because the types are not mutually exclusive, the proportions do not sum to 1.

Contributing cause	Lexis	Westlaw	Pract. Law
Naive Retrieval	0.47	0.20	0.34
Inapplicable Authority	0.38	0.23	0.34
Reasoning Error	0.28	0.61	0.49
Sycophancy	0.06	0.00	0.03

On top of that: different countries can rely on *completely different* legal system and paradigms, e.g. **Civil Law** vs. **Common Law** .

Hallucination-Free? Assessing the Reliability of Leading AI Legal Research Tools. Magesh et al. (Stanford), Journal of Empirical Legal Studies, 2025

Specializing LLMs to the Legal Domain

A handful of instructions is not enough; we need a **legal-specific harness** .

Two prominent **Italian-specific** examples that are just about *search* in the legal domain:

The **Cartabia Reform** (D.Lgs. 149/2022, D.Lgs. 150/2022), as a sweeping overhaul of the Italian justice system, requires:

- Careful handling of normative vs. case law search
- Explicit temporal reasoning to integrate sources *before* and *after* the reform (with the former being a much larger data pool!)

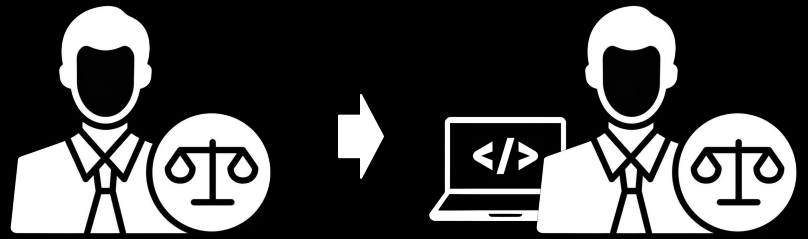
The **Favor Rei** principle in Criminal Law, which dictates that the effects of a law should be those of the revision of that law that's *most favorable* to the offender (also retroactively). In other words, we need:

- Full versioning of Criminal Law data
- Explicit temporal reasoning to integrate search results with available contextual information

Specializing LLMs to the Legal Domain

A handful of instructions is not enough; we need a **legal-specific harness** .

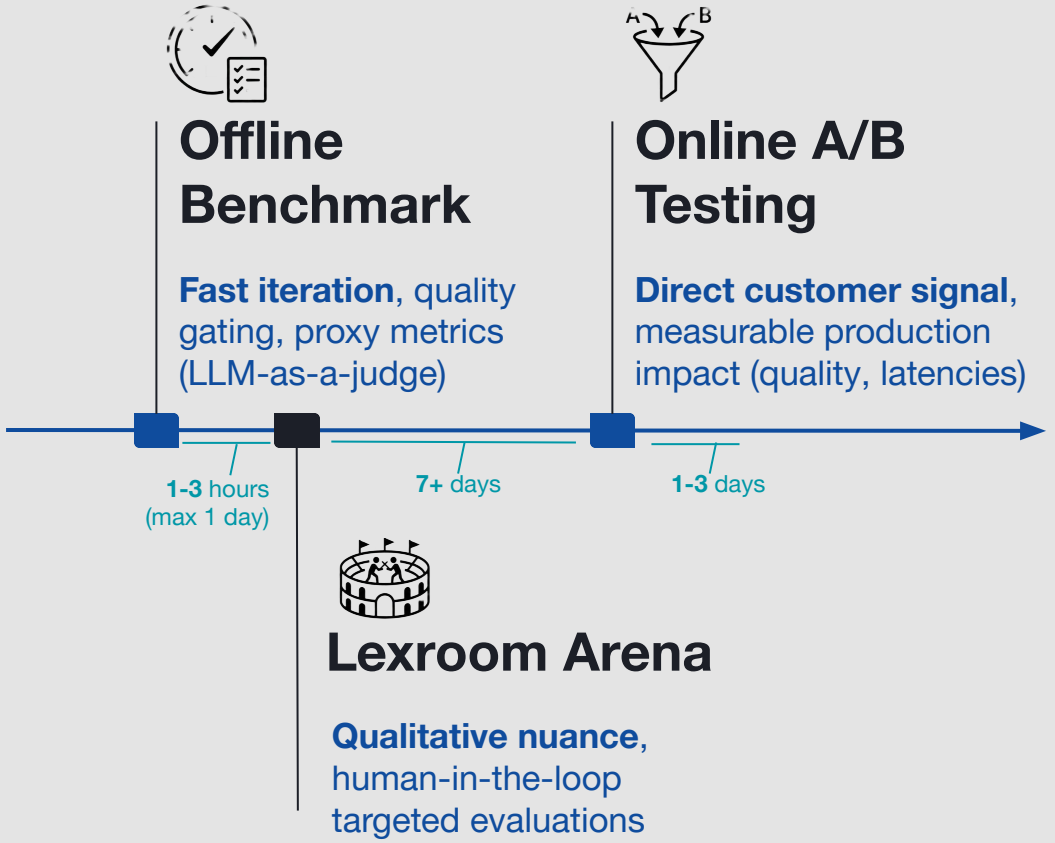
In Lexroom we don't have **Legal Experts** , we have **Legal Engineers** .



Activity	Partner tech team(s)
Own *feature shaping* for Lexroom products	Product
Own and develop *versioned legal knowledge* (e.g. prompts, skills, sources metadata)	Data, AI & Search
Translate back and forth between *legal requirements and technical requirements* (e.g. Favor Rei support → Criminal Law versioning)	Product, AI & Search

How do We Evaluate which LLMs to Use?

A tiered evaluation strategy as a tradeoff among **speed** , **nuance** , and **customer-first** principles.



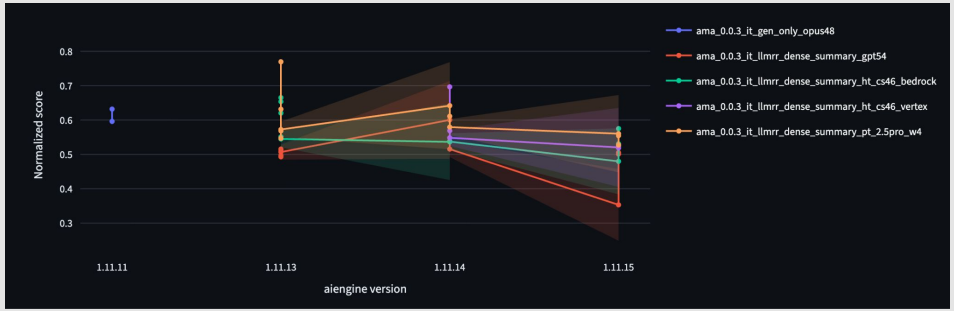
How do We Evaluate which LLMs to Use?

A tiered evaluation strategy as a tradeoff among **speed** , **nuance** , and **customer-first** principles.

Offline Benchmark

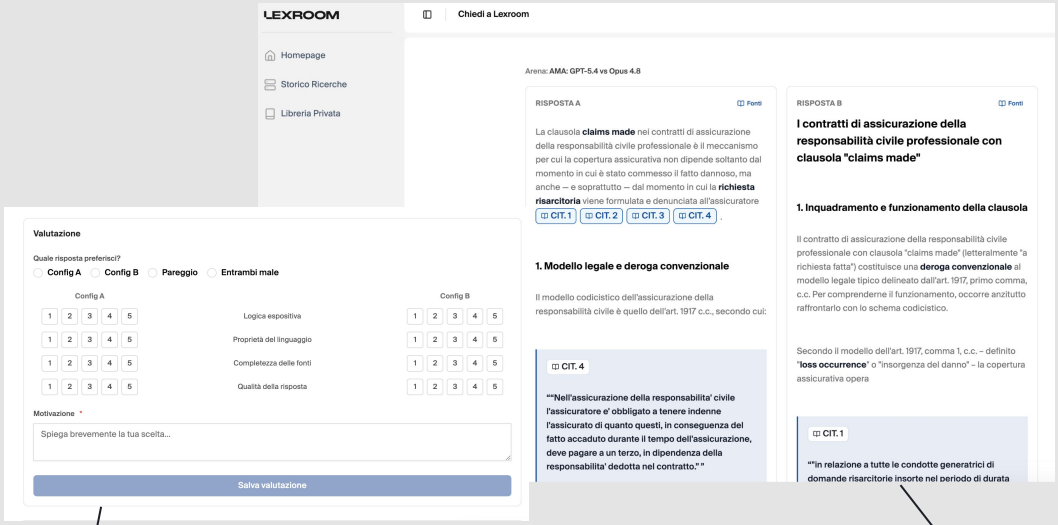
Fast iteration, quality gating, proxy metrics (LLM-as-a-judge)

- **100+** (and counting) end-to-end *question-answer pairs with grounded sources*, spanning different modules and production use cases
- Detailed human-crafted evaluation rubrics (**>30**) per question-answer pair, runnable with an *LLM-as-a-judge harness*
- Internal leaderboard to compare different versions of the Lexroom AI Engine and/or different generalist LLMs



How do We Evaluate which LLMs to Use?

A tiered evaluation strategy as a tradeoff among **speed** , **nuance** , and **customer-first** principles.



ELO scoring **and** qualitative evaluation

Head-to-head comparison between two Lexroom versions

Lexroom Arena

Qualitative nuance, human-in-the-loop targeted evaluations

AI & Search @ Lexroom

Claudio
Delli Bovi
TECH LEAD

Andrea
Ponti
AI ENGINEER

Daniele
Brambilla
AI ENGINEER

Emanuele
Costantini
AI ENGINEER

Giuseppe
Mantineo
AI ENGINEER

Simone
Pierro
AI ENGINEER

We're
hiring!
AI ENGINEER

Q&A