salesforce research

Enterprise Deep Research: Steerable Multi-Agent Deep Research for Enterprise Analytics

Akshara Prabhakar, Roshan Ram, Zixiang Chen, Silvio Savarese, Frank Wang*, Caiming Xiong, Huan Wang, Weiran Yao

Salesforce AI Research

Abstract. As information grows exponentially, enterprises face increasing pressure to transform unstructured data into coherent, actionable insights. While autonomous agents show promise, they often struggle with domain-specific nuances, intent alignment, and enterprise integration. We present **Enterprise Deep Research (EDR)**, a multi-agent system that integrates (1) a Master Planning Agent for adaptive query decomposition, (2) four specialized search agents (General, Academic, GitHub, LinkedIn), (3) an extensible MCP-based tool ecosystem supporting NL2SQL, file analysis, and enterprise workflows, (4) a Visualization Agent for data-driven insights, and (5) a reflection mechanism that detects knowledge gaps and updates research direction with optional human-in-the-loop steering guidance. These components enable automated report generation, real-time streaming, and seamless enterprise deployment, as validated on internal datasets. On open-ended benchmarks including DeepResearch Bench and DeepConsult, EDR outperforms state-of-the-art agentic systems without any human steering. We release the EDR framework and benchmark trajectories to advance research on multi-agent reasoning applications.

Code github.com/SalesforceAIResearch/enterprise-deep-research
Dataset huggingface.co/datasets/Salesforce/EDR-200

1 Introduction

Recent advances in LLMs have driven the emergence of general-purpose language agents capable of performing a wide range of reasoning-intensive tasks—including research assistance [36, 12], retrieval-augmented generation (RAG) [38], coding [47, 46], and knowledge synthesis. At the enterprise level, such capabilities have spurred growing demand for autonomous systems that can conduct an extensive search of available resources relevant to a given prompt and produce structured, markdown-formatted reports with evidence. This has motivated a new class of *deep research agents* [10, 26, 25]—autonomous LLM-based systems that emulate human analysts by iteratively retrieving evidence, decomposing problems, and constructing higher-order insights.

In open-ended deep research, agents must reason across unbounded corpora and evolving goals,

*Work done while at Salesforce AI Research.

where a “complete” answer may require synthesizing hundreds of heterogeneous documents rather than locating a single fact [20]. The objective is to take a user-provided research prompt, perform extensive web-scale or domain-specific searches, and generate a structured report aligned with the prompt’s requirements—far beyond the scope of typical RAG systems or conversation-oriented tool-calling agents [27, 52]. Enterprise settings amplify these challenges [6]: information is dispersed across emails, databases, and reports; domain knowledge shifts rapidly; and goals are strategic rather than factual. Tasks like feature adoption forecasting or productivity diagnostics demand contextual reasoning, long-horizon planning, and transparent evidence attribution—areas where conventional LLM pipelines yield shallow, unverifiable outputs.

Despite rapid progress, most deep research systems remain opaque and inflexible. They operate as black boxes where users cannot inspect intermediate reasoning states, source provenance, or decision trajectories once execution begins. Although some systems permit limited plan editing before launch [32], they lack real-time steering—the capability that enables users to dynamically guide the agent’s reasoning trajectory–if an agent misinterprets intent, confuses entities, or diverges from task goals, it seldom recovers without costly manual restarts. Such rigidity leads to redundant API calls, inefficient exploration, and results that deviate from user intent [53, 29, 30]. Effective research systems should instead make reasoning transparent and steerable, enabling users to intervene mid-process to correct direction, refine scope, or prioritize evidence. This interactivity narrows the search space, conserves computation, and aligns outcomes more tightly with domain constraints and user expectations.

To address these limitations, we present **E**nterprise **D**eep **R**esearch **(EDR)**—a transparent, steerable multi-agent framework for adaptive, interpretable, and user-aligned research. Drawing on Anthropic’s notion of *thinking in context* [4], EDR formalizes *steerable context engineering*, enabling humans to modify agent context dynamically during execution. Steering occurs at the context curation layer—directly influencing what information enters the agent’s attention at each decision point—rather than through pre-specified constraints or post-hoc corrections [23]. By exposing the agent’s internal planning state (via `todo.md`) and translating natural language steering inputs into context modifications—such as task additions, cancellations, or reprioritizations—EDR empowers users to act as *context curators* instead of passive observers. The framework ensures that reasoning remains *visible* (through explicit task and provenance tracking), *modifiable* (via real-time steering), and *traceable* (with full evidence transparency). This design establishes a dynamic human–AI collaboration loop that maintains contextual grounding over long horizons, mitigates lost-in-the-middle issues [21], and ultimately delivers faithful and scalable research for enterprise applications.

Our key contributions are threefold:

- we introduce EDR (Figure 1), a modular, extensible, multi-agent architecture that is fully configurable for optimal research execution for enterprise;
- we provide users fine-grain control over the research process via a todo-driven steering framework that enables human-in-the-loop guidance during research execution, not just at the planning stage;
- we open-source EDR-200, a dataset comprising EDR’s complete research trajectories from

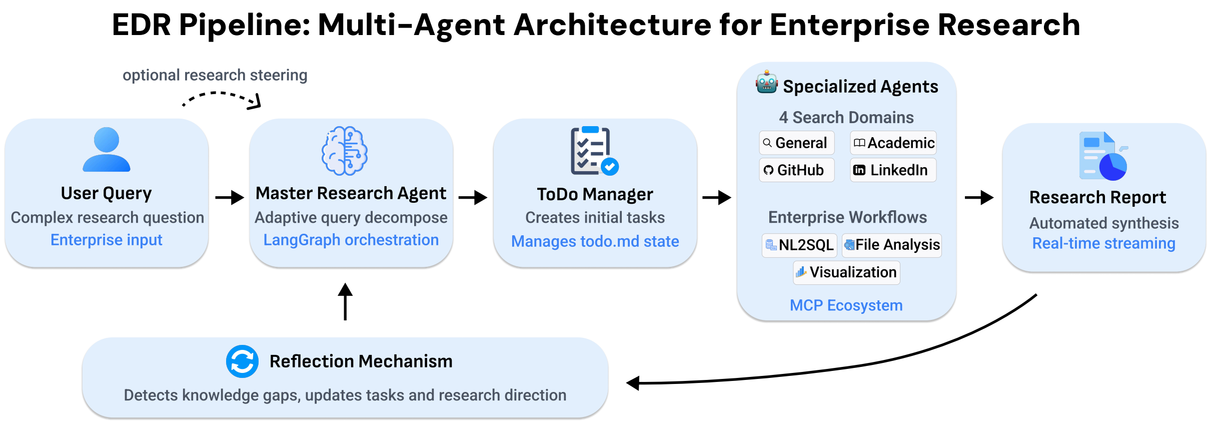


Figure 1: Overview of **E**nterprise **D**eep **R**esearch framework—orchestrated system combining planning, specialized search, extensible enterprise tools, visualization, and reflection with optional human steering.

benchmark evaluations to promote future research.

2 System Architecture

Enterprise Deep Research (EDR) adopts a modular, multi-agent architecture engineered for scalability, robustness, and adaptive orchestration of complex research workflows (Figure 1). The framework integrates a central planning agent, structured task management, domain-specialized retrieval tools, and extensible enterprise connectors, enabling iterative, human-in-the-loop research at scale. Prompts used across all stages are detailed in Appendix A.

2.1 Master Research Agent

The Master Research Agent functions as the central orchestrator, decomposing high-level research objectives into discrete tasks and coordinating multi-agent execution. Leveraging function calling and context-aware prompt engineering, the agent performs adaptive query decomposition, intent classification, and entity extraction. Each task is annotated with priority, recommended tools, metadata, and execution dependencies.

Decomposition integrates three contextual signals: (1) the user query, (2) knowledge gaps identified from prior iterations, and (3) steering directives issued by the Research Todo Manager. Dynamic replanning permits adjustments based on intermediate results, agent efficacy, or updated directives. Quality control is embedded within the Master Agent. Mechanisms include semantic consistency validation, cross-agent result comparison, automated fact-checking, and confidence scoring. Multi-agent outputs are aggregated into coherent interim summaries, resolving conflicts, normalizing citations, and compressing context while preserving provenance.

Adaptive Decomposition. The agent employs LLM function calling to classify user queries as simple or complex. Simple queries (e.g., "What is GPT-4?") yield single targeted searches. Complex queries (e.g., "AI's impact on healthcare, education, and employment") decompose into parallel tasks with independent search strategies. In subsequent loops, the LLM receives all pending tasks, preventing redundant work through explicit duplicate-awareness.

Temporal and Knowledge Grounding. Prompts embed temporal markers (`CURRENT_DATE`, `CURRENT_YEAR`) to ground recency-sensitive queries. User-uploaded documents receive authoritative status, guiding the agent to generate complementary—not redundant—searches that validate or extend internal knowledge with external evidence.

2.2 Research Todo Manager

The Research Todo Manager provides a structured, human-interpretable task management layer. Tasks are stateful objects with unique IDs, natural language descriptions, priority scores (5–10), lifecycle status (*pending*, *in-progress*, *completed*, *canceled*), and provenance tags (`initial_query`, `knowledge_gap`, `steering_message`). The `todo.md` file functions as a persistent, shared representation between agents and human users, facilitating transparency and collaborative context management; functioning as both agent execution plan and user progress tracker. Where Manus [22] employs todos solely for agent planning, EDR extends this approach by exposing todos to human oversight, combining structured task management with persistent note-taking. This hybrid design allows both agent and human to collaboratively maintain context and steer research direction.

Priority-based scheduling ensures high-impact research items are addressed first, mitigating context dilution during long-horizon LLM planning. Users may issue natural language directives (e.g., "focus on peer-reviewed sources," "prioritize recent publications"), which are translated into task adjustments, exclusions, or priority modifications. Versioning enables efficient frontend synchronization, maintaining a consistent view of system progress without continuous state streaming.

2.3 Search API and Tools

General Web Search. Retrieves broad web content, including news, reports, and general knowledge. Implements top-k result retrieval with full content extraction, semantic deduplication, and relevance scoring. Integrates the Tavily API to provide high-throughput, structured access to web content.

Academic Search. Targets scholarly publications and peer-reviewed content. Supports higher result limits, fuzzy deduplication for title variations, and optional temporal weighting to emphasize recent research. Integrates multiple academic repositories, including arXiv, for comprehensive coverage and ensures structured metadata extraction for downstream processing.

GitHub Search. Focuses on code repositories, technical implementations, and software documentation. Employs repository-level deduplication to prevent redundancy and prioritizes file-level URLs for maximum utility. Direct API integration with GitHub ensures reliability and high throughput, while results include structured metadata for seamless integration with domain agents.

LinkedIn Search. Extracts professional profiles, company information, and domain expertise. Optimized for top-K retrieval, selective raw content retention, and strict domain restriction to linkedin.com, enabling precise enterprise-focused research queries. Metadata extraction supports downstream evaluation of expertise and relevance.

2.4 Domain-Specific and MCP Tools

File Analysis. Processes structured and unstructured files (db, sqlite, pdf, docx, txt, csv, xlsx, images) using format-specific parsers and LLM-powered content summarization. Includes metadata extraction, layout preservation, and semantic content analysis to integrate uploaded knowledge into the research context.

NL2SQL Agent. Translates natural language queries into SQL statements for structured database interrogation. Incorporates schema awareness, query decomposition, multi-layered validation (syntax, semantics, performance, security), and result interpretation. Optimized for enterprise data warehouses, it supports context-sensitive filtering and automated reasoning over relational schemas.

Visualization Agent. Generates visual representations from quantitative findings. Chart types are selected adaptively based on data characteristics, including bar, line, scatter, heatmap, and pie charts. Visualizations are rendered in sandboxed execution environments and support interactive exploration, export to multiple formats (pdf, docx, html, png, svg), and structured report integration.

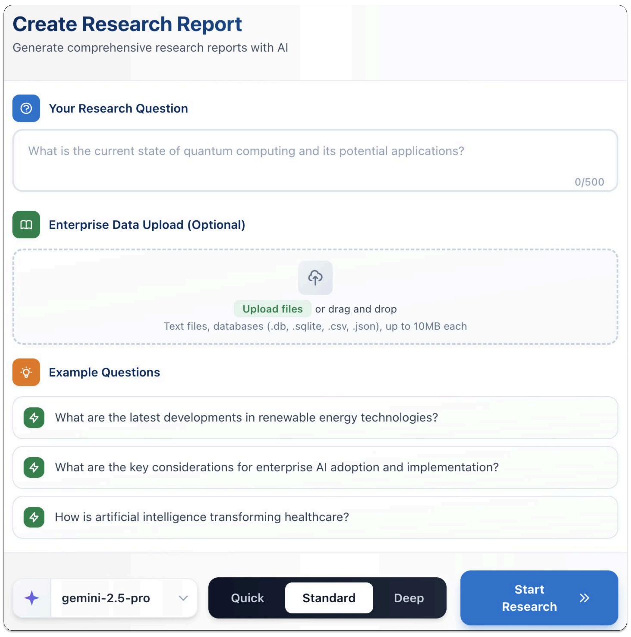
Enterprise Connectors via MCP. EDR supports extensible integration through the Model Context Protocol (MCP), enabling connection to custom enterprise systems, remote computation services, and additional domain tools. MCP supports HTTP and stdio transport for universal tool interoperability, reducing integration overhead for new agents. Supported examples include code search, database querying, and organization-specific computational tools.

2.5 Integration and Workflow Coordination

EDR tightly couples task decomposition, search execution, and domain tool utilization within iterative research loops. Outputs from all agents are consolidated into a unified running summary and task reflection mechanism, continuously informing the Research Todo Manager and Master Agent. This design preserves context coherence, supports cumulative knowledge integration, and provides a seamless transition into the Research Flow Mechanism, enabling systematic report generation from complex, multi-domain queries.

3 Research Flow Mechanism

EDR translates high-level user queries into comprehensive analytical reports through a structured, iterative workflow that integrates human-in-the-loop steering with multi-agent execution. The API specifications and frontend implementation is detailed in Appendix B.


Create Research Report

Generate comprehensive research reports with AI

Your Research Question

What is the current state of quantum computing and its potential applications?

0/500

Enterprise Data Upload (Optional)

Upload files or drag and drop

Text files, databases (.db, .sqlite, .csv, .json), up to 10MB each

Example Questions

What are the latest developments in renewable energy technologies?

What are the key considerations for enterprise AI adoption and implementation?

How is artificial intelligence transforming healthcare?

gemini-2.5-pro

Quick [✓ Standard] Deep

Start Research

Figure 2: Screenshot of EDR's home screen UI. This is the initial screen where users can submit their research query and add optional data files (e.g., images, sheets, databases). The research offers 3 run modes: *quick* (for a high-level investigation), *standard* (the standard deep research mode), and *deep* (for a max-effort report) and allows the user to choose their underlying base LLM.

3.1 User Query Ingestion and Todo Initialization

Upon submission, the user query is received via the frontend interface (Figure 2) and a new session is spawned. The `ResearchTodoManager` constructs an initial lightweight 3–5 task plan for the user query (Appendix Figure 4), providing immediate UI feedback. Each task is annotated with a unique identifier, a priority score (ranging 5–10, either LLM-suggested or calculated as $5 + (N - i)$ where N is the total subtask count and i is the task index), and provenance metadata tracking its source. Tasks originating from the initial user query are labeled `initial_query`, while those generated in subsequent loops to address knowledge gaps are tagged `knowledge_gap`. Tasks created in response to user steering messages are marked `steering`, with steering-derived tasks receiving the highest priority (10), followed by original query tasks (9), and knowledge-gap tasks (7). The `ResearchTodoManager` maintains these tasks in a persistent `todo.md` representation as a named variable in the code execution state. As the research progresses, EDR updates it and checks off completed items.

3.2 Task-Query Transformation

At the beginning of each research iteration, the `MasterResearchAgent` constructs an integrated prompt (Appendix Figure 5) embedding the original research objective, the progressively refined summary of prior findings, the set of unresolved information needs, and all active steering constraints derived from user interactions. Guided by this comprehensive context, the LLM performs adaptive query decomposition, generating 3–7 new tasks and corresponding search queries, with task source transitioning from `initial_query` (Loop 0) to `knowledge_gap` (Loop 1+). The agent's decomposition prompt includes existing pending tasks, instructing the LLM to avoid redundant work—though fuzzy matching provides a safety net for any remaining overlaps. Each proposed search query includes metadata specifying the recommended retrieval domain (e.g., academic_search for scholarly content, github_search for code, general_search for broad topics, nl2sql for database queries) and a descriptive aspect indicating the query's focus area.

Before dispatching these search queries, three layers of quality control are enforced. First, semantic deduplication prevents redundant searches by fuzzy string matching with prefix normalization, merging duplicates and updating their priority if higher. Second, constraint enforcement ensures compliance with steering instructions: for instance, excluding terms marked by the user or elevating those aligned with focus directives. Finally, priority adjustment dynamically reorders the execution sequence so that high-impact and user-aligned queries receive preferential processing. Tasks associated with selected queries are transitioned to the *in-progress* state, ensuring consistent synchronization between planning intent and execution progress across research loops. Tasks transition through a formal lifecycle: *pending* (awaiting execution) → *in-progress* (actively querying) → *completed* (successfully resolved) or *canceled* (rendered obsolete). Queries are dispatched in parallel to specialized agents, including general search, academic literature, code repositories, and domain tools (NL2SQL, MCP tools) as appropriate, ensuring domain-specific execution efficiency. Agents perform filtering, deduplication, and relevance scoring before returning results for aggregation.

3.3 Result Aggregation and Incremental Synthesis

Returned results from domain-specific agents undergo three-stage processing:

Stage 1: Inter-Agent Deduplication. Results are consolidated through semantic similarity comparison, identifying overlapping content across multiple search tools. Citation normalization ensures consistent URL and title formatting, with priority given to the highest-quality representation of each unique source.

Stage 2: LLM-Driven Synthesis. In this stage, the LLM merges newly gathered research content into the existing running summary using a dedicated synthesis process. It takes four inputs: (1) the previous iteration's running summary, (2) newly fetched web research results, (3) knowledge gaps from reflection, and (4) user-uploaded knowledge (if present). The LLM performs context compression, extracting key insights while preserving citation links and metadata. This prevents exponential context growth—rather than accumulating all raw search outputs, the system maintains a progressively refined knowledge representation across iterations.

Stage 3: Source Citation Management. Extracted sources are tracked in a deduplicated dictionary, maintaining URL-to-metadata mappings for downstream report generation. Unused sources are logged for transparency but excluded from final citations.

The synthesis step ensures that each iteration builds upon prior findings without losing coherence or exceeding context limits, enabling sessions with 10+ iterations and hundreds of sources.

3.4 Steering Integration

EDR implements a queue-based, race-condition-safe steering mechanism that enables real-time user guidance without interrupting ongoing execution. User messages are queued during research execution and, if multiple messages accumulate, they are summarized to extract core directives (e.g., "focus on peer-reviewed sources" and "prioritize recent papers" → "emphasize recent peer-reviewed literature"). Messages are processed atomically between iterations during the reflection phase (Subsection 3.5), preventing interference with active queries. To prevent data loss, the system employs snapshot-based merging: steering messages that arrive *during* reflection are automatically preserved and appended to the post-reflection queue. This ensures user input is never lost while maintaining deterministic LLM reasoning over a stable message set. The extracted directives update the `ResearchTodoManager`; incorporate steering constraints as priority boosts, exclusion filters, or focus directives, ensuring alignment with user intent across subsequent query generation and execution cycles.

3.5 Reflection and Todo Update

The reflection mechanism serves as the core process in EDR. After each iteration, the system evaluates the aggregated results against the current todo plan and accumulated knowledge, identifying:

- **Knowledge Gaps:** Missing concepts, unexplored domains, or insufficient evidence relative to the original user query.
- **Task Misalignment:** Tasks that are no longer relevant due to new findings or user steering directives.
- **Quality Inconsistencies:** Contradictory or low-confidence information returned by agents.

Based on this analysis, the `ResearchTodoManager` updates the todo plan (Appendix Figure 6):

- **Generate New Tasks:** New tasks are generated to address knowledge gaps, with priority scores reflecting their importance.
- **Update Task Status:** Misaligned or resolved tasks are *canceled* or marked as *completed*.
- **Clear Queue:** Specifying indices of user steering messages fully addressed through task creation/cancellation. Unaddressed messages remain queued for subsequent loops.

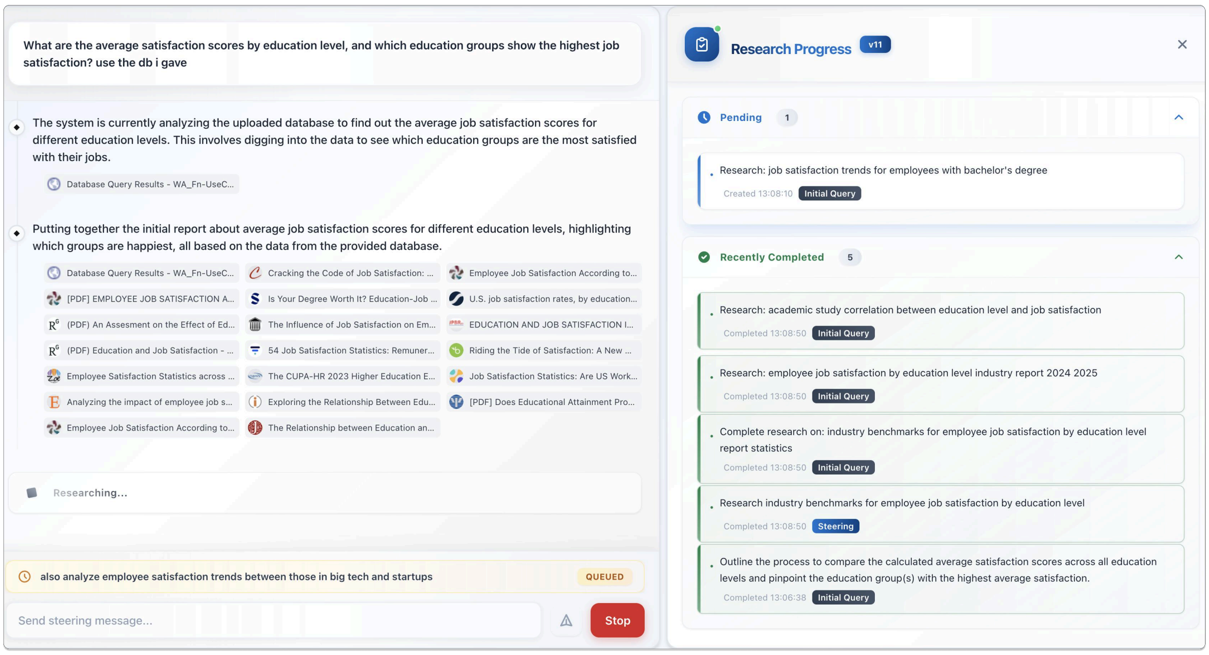
The `ResearchTodoManager` increments a version counter on every state modification, triggering frontend updates only when changes occur. This version-based polling provides real-time visibility into task status and provenance without continuous state streaming (Figure 3). Reflection is iterative and cumulative, ensuring that subsequent research loops increasingly focus on unaddressed knowledge gaps while maintaining continuity with prior findings.

3.6 Iterative Refinement and Loop Continuation

The cycle of query planning, agent execution, steering integration, result aggregation, and reflection repeats iteratively. Each loop incorporates feedback from previous iterations and user steering messages, gradually converging toward comprehensive coverage of the research question. Termination occurs when knowledge gaps are resolved, maximum loop limits are reached, or the system determines sufficient report completeness.

3.7 Final Report Generation and Validation

Upon completion, the system synthesizes the running summary, aggregated sources, code snippets, and steering history into a structured report. Quality assurance checks validate citation


What are the average satisfaction scores by education level, and which education groups show the highest job satisfaction? use the db i gave

The system is currently analyzing the uploaded database to find out the average job satisfaction scores for different education levels. This involves digging into the data to see which education groups are the most satisfied with their jobs.

Database Query Results - WA_Fn-UseC...

Putting together the initial report about average job satisfaction scores for different education levels, highlighting which groups are happiest, all based on the data from the provided database.

Database Query Results - WA_Fn-UseC...
Cracking the Code of Job Satisfaction: ...
Employee Job Satisfaction According to...
[PDF] EMPLOYEE JOB SATISFACTION A...
Is Your Degree Worth It? Education-Job ...
U.S. job satisfaction rates, by education...
(PDF) An Assesment on the Effect of Ed...
The Influence of Job Satisfaction on Em...
EDUCATION AND JOB SATISFACTION I...
(PDF) Education and Job Satisfaction - ...
54 Job Satisfaction Statistics: Remuner...
Riding the Tide of Satisfaction: A New ...
Employee Satisfaction Statistics across ...
The CUPA-HR 2023 Higher Education E...
Job Satisfaction Statistics: Are US Work...
Analyzing the impact of employee job s...
Exploring the Relationship Between Edu...
[PDF] Does Educational Attainment Pro...
Employee Job Satisfaction According to...
The Relationship between Education an...

Researching...

also analyze employee satisfaction trends between those in big tech and startups — QUEUED

Send steering message...

Stop

Research Progress v11

Pending 1

Research: job satisfaction trends for employees with bachelor's degree
Created 13:08:10 Initial Query

Recently Completed 5

Research: academic study correlation between education level and job satisfaction
Completed 13:08:50 Initial Query

Research: employee job satisfaction by education level industry report 2024 2025
Completed 13:08:50 Initial Query

Complete research on: industry benchmarks for employee job satisfaction by education level report statistics
Completed 13:08:50 Initial Query

Research industry benchmarks for employee job satisfaction by education level
Completed 13:08:50 Steering

Outline the process to compare the calculated average satisfaction scores across all education levels and pinpoint the education group(s) with the highest average satisfaction.
Completed 13:06:38 Initial Query

Figure 3: Screenshot of EDR's research execution screen. Left pane shows the progress with a summary of the current step, while the progress pane shows the todo status. Each task is annotated with its status and provenance with timestamps for full traceability. The received steering message is queued until the reflection phase.

completeness, structural coherence, query coverage, and adherence to user directives. The final document, delivered via the frontend interface, provides an interactive and detailed account of the research process, preserving transparency, reproducibility, and user-aligned insight generation.

4 Evaluations

4.1 Experimental Setup

For all evaluations, we use `gemini-2.5-pro` as the underlying base model, as it showed the strongest performance in our preliminary tests. We set the max research loops to 5 for DeepResearch Bench, 10 for DeepConsult, and 2 for ResearchQA. Real-time steering is kept disabled as the research query is fixed for these evaluation settings, with no human intervention.

Baselines. We test against popular proprietary deep research assistants OpenAI DeepResearch `openai-deepresearch` [25], Gemini-2.5-pro-deep-research [32], Claude-research [3], Perplexity Deep Research [26], doubao-research [7], kimi-research [16] and open source systems like langchain-open-deep-research [17], WebShaper [42], and the more recent WebWeaver [20].

Datasets. We evaluate on 3 popular open-ended deep research benchmarks:

(1) DeepResearch Bench [9], comprising 100 PhD-level complex research tasks meticulously formulated by domain experts across 22 distinct fields, such as Science & Technology, Finance & Business, Software Engineering, and Art & Design.

(2) DeepConsult [50], a specialized collection of business and consulting-focused prompts designed for in-depth research, encompassing a broad range of topics such as marketing strategy, financial analysis, emerging technology trends, and business planning.

(3) ResearchQA [49], a large-scale, multi-field benchmark comprising 3750 scientific test questions curated by PhD scholars mined from academic survey papers spanning 7 broad domains. Each query is annotated with one or more rubrics which are scored to indicate the extent to which a response satisfies the criterion, enabling detailed, fine-grained analysis across domains and question types.

4.2 Results

DeepResearch Bench Results. It employs two evaluation metrics [9]: RACE (Report Quality) measures the quality of generated reports against references across four dimensions—Comprehensiveness, Insight, Instruction-Following, and Readability—aggregated into an overall weighted score represented by Over. Citation Accuracy (CitAcc.) evaluates retrieval accuracy and reliability. Following the benchmark setup, the judge model for the comparison is Gemini-2.5-pro. EDR obtains exceptional performance on the DeepResearch Bench leaderboard. As shown in Table 1, EDR outperforms all proprietary and most open-source agentic systems with an overall score of 49.86. Specifically, on the Instruction Following and Readability criteria, EDR obtains particularly high scores. In terms of cost, we see that EDR consumes 4x lesser tokens than `langchain-open-deep-research`.

Agentic System	**Over.**	**Comp.**	**Ins.**	**Inst.**	**Read.**	**CitAcc.**	**Tokens**	**Cost ($)**
gemini-2.5-pro-preview-05-06	31.90	31.75	24.61	40.24	32.76	-	-	-
WebShaper (32B)	34.93	31.58	26.17	44.81	40.38	-	-	-
langchain-open-deep-research	43.44	42.97	39.17	48.09	45.22	-	207,005,549	87.83
doubao-research	44.34	44.84	40.56	47.95	44.69	52.86	-	-
kimi-research	44.64	44.96	41.97	47.14	45.59	-	-	-
Claude-research	45.00	45.34	42.79	47.58	44.66	-	-	-
openai-deepresearch	46.45	46.46	43.73	49.39	47.22	75.01	-	-
WebWeaver (qwen3-30b-a3b-instruct-2507)	46.77	45.15	45.78	49.21	47.34	25.00	-	-
WebWeaver (gpt-oss-120b)	48.11	48.03	47.20	48.94	48.11	66.14	-	-
Gemini-2.5-pro-deepresearch	49.71	49.51	49.45	50.12	**50.00**	78.30	-	-
WebWeaver (Claude-sonnet-4-20250514)	50.58	**51.45**	50.02	50.81	49.79	**93.37**	71,922,021	-
WebWeaver (qwen3-235b-a22b-instruct-2507)	**50.62**	51.29	**51.00**	49.98	48.89	78.25	-	-
Enterprise Deep Research	49.86	49.01	50.28	50.03	49.98	72.50	**53,926,192**	117.48

Table 1: Performance on DeepResearch Bench. Best number shown in **bold**, second best is underlined. ⬜ indicates the system is not accessible to test.

DeepConsult Results. Table 2 presents the performance of agentic systems on DeepConsult. Evaluation is conducted via pairwise comparison against the OpenAI DeepSearch baseline, using win, tie, and loss rates as primary metrics, along with an average quality score. The judge model used is `gpt-4.1-2025-04-14`. We observe that EDR achieves the highest win rate of 71.57% and a superior average score of 6.82, which is higher than other open-source systems. Additionally, the lose rate is much lower at 9% indicating that most reports are significantly superior or at par with `openai-deepresearch`. Overall, one run with EDR takes 48,850,862 tokens amounting to a total cost of $105.51.

ResearchQA Results. This benchmark evaluates research responses across six distinct rubric categories – Citation, Impact, Comparison, Example, Limitation, or Other. Queries can be tagged with multiple types, capturing intersections like Comparison + Impact or Example +

Agentic System	**Win Rate (%)**	**Tie Rate (%)**	**Lose Rate (%)**	**Avg. Score**
WebShaper (32B)	3.25	3.75	93.00	1.63
WebWeaver (qwen3-30b-a3b-instruct-2507)	28.65	34.90	36.46	4.57
Claude-research	25.00	38.89	36.11	4.60
doubao-research	29.95	40.35	29.70	5.42
Perplexity Deep Research	32.00	-	-	-
openai-deepresearch	0.00	100.00	0.00	5.00
WebWeaver (qwen3-235b-a22b-instruct-2507)	54.74	28.61	16.67	6.47
Gemini-2.5-pro-deepresearch	61.27	31.13	**7.60**	6.70
WebWeaver (gpt-oss-120b)	65.31	11.22	23.47	6.64
WebWeaver (Claude-sonnet-4-20250514)	66.86	10.47	22.67	**6.96**
Enterprise Deep Research	**71.57**	19.12	<u>9.31</u>	<u>6.82</u>

Table 2: Performance on DeepConsult. Best number shown in **bold**, second best is <u>underlined</u>. ▢ indicates systems that are not accessible to test.

Impact, which allows assessment of both individual capabilities and integrated reasoning across multiple research competencies. From Table 3, we see that Perplexity Deep Research (Sonar) achieves the highest overall coverage at 75.3%, while EDR attains a competitive 68.5%. Analysis by rubric type reveals EDR's strong performance on General, Impact, and Comparison items, but severe weaknesses in citations (85% failures), example generation, and multi-criteria rubrics like Comparison+Example+Impact. Domain-specific patterns emerge, with Life & Earth Sciences and Business & Economics performing best, whereas Humanities & Arts struggles with examples. Score distributions are strongly bimodal, indicating that responses either fully satisfy a rubric item or fail completely. These findings highlight critical areas for improvement, specifically citation handling and example generation to enhance EDR's reliability for scientific research tasks.

Agentic System	**Avg Length**	**All**	**Health.**	**Life.**	**Engg.**	**PhysSci.**	**SocSci.**	**Human.**	**Econ.**
sonar	242.2	58.61	56.61	60.55	59.43	61.62	59.97	57.48	62.80
openscholar-8b+feedback	788.8	58.72	57.77	59.96	58.62	61.48	57.27	57.29	62.48
sonar-reasoning	280.5	64.33	62.73	66.00	65.19	68.11	62.68	61.76	67.49
gpt-4o-search-preview	255.0	65.98	65.52	68.21	65.01	66.60	66.07	62.62	65.63
gemini-2.5-pro+grounding	278.5	68.51	67.38	70.02	68.76	70.99	68.09	65.98	71.21
claude-4-sonnet+ws	327.8	69.18	69.54	70.49	67.59	70.28	68.14	64.70	67.13
o4-mini-deep-research	271.6	72.69	74.02	73.58	70.57	74.04	73.25	68.99	74.54
sonar-deep-research	267.3	**75.29**	**75.01**	**76.31**	**74.48**	**76.77**	**75.34**	**72.47**	**78.01**
Enterprise Deep Research*	**220.1**	68.52	67.98	69.70	68.61	69.80	68.41	65.11	70.35

Table 3: Coverage (%) of various agentic LLM systems across seven research domains in ResearchQA. *We run EDR for a max of 2 research loops, to limit costs for running across 3K test queries.

Enterprise Usecase. We also conduct evaluations on internal enterprise use cases encompassing both open-ended research and research over complex internal proprietary databases. EDR achieves over 95% accuracy in SQL generation and execution, 99.9% uptime, while maintaining reliability and scalability across diverse workloads. User studies report a 98% task completion rate, a 4.8/5 satisfaction score, and a 50% reduction in time-to-insight for complex analytical tasks.

4.3 Trajectory Collection

We collect 201 complete agentic trajectories generated by EDR—99 on DeepResearch Bench and 102 on DeepConsult. Unlike prior benchmarks that capture only final outputs, these trajectories expose the full reasoning process—search, reflection, and synthesis—enabling fine-grained analysis of planning and decision dynamics. It aids in studying long-horizon agentic behavior and developing training and evaluation methods for more efficient research agents. Table 4 summarizes key statistics. Report synthesis peaks at iterations 4–5 with a +1,785-word gain (3× average), marking the most productive growth phase following sufficient information accumulation. Reflection analysis across 1,422 instances reveals recurring knowledge gaps in market (27.1%), comparative (18.4%), and cost (14.2%) analyses. Source usage remains stable at ≈ 14 per iteration, reflecting EDR's sustained diversity and coherence throughout extended research workflows.

Metric	Value
Avg. Iterations per Trajectory	7.19
Avg. Tool Calls per Trajectory	49.88
Avg. Tool Calls per Iteration	6.93
Avg. Searches per Trajectory	28.30
Avg. Report Length (words)	6,523
Avg. Report Growth per Iteration (words)	600

Table 4: Statistics of EDR-200 trajectory dataset.

5 Related Works

AI-driven Research Systems. AI-driven research has advanced from short-answer retrieval pipelines to long-horizon systems that emulate human analysts through iterative reasoning and evidence synthesis. Proprietary systems such as OpenAI Deep Research, Gemini Deep Research [10], and Claude Research [3] achieve expert-level performance on open-domain tasks like fact-checking and report writing but remain closed-source and non-reproducible. Open efforts initially targeted benchmark-oriented QA and retrieval-augmented generation [19, 41, 39, 31], later extending to long-form report generation via OpenDeepResearch [34] and GPT Researcher [33]. However, these systems typically follow rigid draft-then-retrieve pipelines that fix outlines before evidence collection, leading to incoherence and hallucination [11]. Recent frameworks like WebWeaver [20] and NVIDIA-AIQ [23] move toward adaptive, strategy-aware research but still rely on static control and public web data. A key gap remains: current systems cannot conduct enterprise-scale deep research over proprietary, heterogeneous sources while remaining interpretable and steerable. Enterprise Deep Research (EDR) addresses this gap with a multi-agent framework combining adaptive planning, specialized retrieval, and human-guided reflection to enable transparent, auditable, and domain-aware research automation.

Multi-Agent Systems. The paradigm of Multi-Agent Systems (MAS)—where autonomous agents collaborate to solve problems beyond a single agent's capability—has been revitalized by LLMs [43]. Frameworks like AutoGen [44] enable flexible conversational workflows, while MetaGPT [13] and ChatDev [28] simulate structured team dynamics. Individual agents leverage reasoning and feedback [48], self-reflection [37], and external tool use [35] to enhance collaboration. Yet most research focuses on open-domain or benchmark tasks. The challenge of orchestrating specialized agents for deep research across private, heterogeneous enterprise data—spanning databases, internal reports, and document repositories—remains largely unsolved.

Transparency and Enterprise Adoption. Enterprise deployment amplifies challenges of data heterogeneity, evolving knowledge, and governance, which opaque, one-shot pipelines cannot handle [6, 18]. High-stakes settings require auditable provenance, cost efficiency, and stakeholder alignment [51, 40], making black-box agents unsuitable [5] and driving the need for interpretable systems [2, 24]. Existing enterprise benchmarks—CRMArena-Pro [15, 14], OSWorld [45], and WorkArena [8]—evaluate multi-application task execution but overlook long-horizon synthesis over proprietary data. Concurrently released, DRBench [1] evaluates agents on an enterprise scale, multi-source reasoning that integrates public and private knowledge for open-ended research objectives. EDR fills this specific gap by externalizing reasoning into a transparent, modifiable plan, enabling human-in-the-loop control and auditability essential for reliable enterprise deep research.

6 Conclusion

We introduced Enterprise Deep Research (EDR), a multi-agent autonomous research framework that advances AI-driven enterprise analytics through steerable context engineering—a paradigm enabling dynamic and interpretable human-AI collaboration. EDR combines intelligent tool selection, adaptive planning, and cross-system retrieval to facilitate large-scale, transparent, and goal-aligned research workflows. The system achieves state-of-the-art performance on deep research benchmarks and internal enterprise evaluations, underscoring its effectiveness in complex analytical environments. This work demonstrates the potential of AI-powered automation for enterprise research and decision support, integrating advanced reasoning capabilities with enterprise-grade system design. Future work will focus on enhancing output factuality through improved citation and evidence grounding, developing predictive steering mechanisms, and expanding integration across broader enterprise data ecosystems.

References

[1] Amirhossein Abaskohi et al. *DRBench: A Realistic Benchmark for Enterprise Deep Research*. 2025. arXiv: 2510.00172 [cs.CL]. URL: https://arxiv.org/abs/2510.00172.

[2] Anthropic. "Effective Context Engineering for AI Agents". In: *Anthropic Engineering Blog* (2024).

[3] anthropic. *Meet Claude*. 2025. URL: https://www.anthropic.com/claude.

[4] Anthropic Applied AI Team. *Effective Context Engineering for AI Agents*. Published September 29, 2025. Accessed: October 2025. URL: https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents.

[5] AryaXAI. "LLM Observability: A Guide to AI Transparency for Agents". In: *AryaXAI Blog* (2025).

[6] Prafulla Kumar Choubey et al. *Benchmarking Deep Search over Heterogeneous Enterprise Data*. 2025. arXiv: 2506.23139 [cs.CL]. URL: https://arxiv.org/abs/2506.23139.

[7] Doubao. *Doubao Deep Research*. https://www.doubao.com/chat/. 2025.

[8] Alexandre Drouin et al. *WorkArena: How Capable Are Web Agents at Solving Common Knowledge Work Tasks?* 2024. arXiv: 2403.07718 [cs.LG].

[9] Mingxuan Du et al. “DeepResearch Bench: A Comprehensive Benchmark for Deep Research Agents”. In: *arXiv preprint arXiv:2506.11763* (2025).

[10] google. *Try Deep Research and our new experimental model in Gemini, your AI assistant.* 2025. URL: https://blog.google/products/gemini/google-gemini-deep-research/.

[11] Rujun Han et al. “Deep Researcher with Test-Time Diffusion”. In: *CoRR* abs/2507.16075 (2025). DOI: 10.48550/ARXIV.2507.16075. arXiv: 2507.16075. URL: https://doi.org/10.48550/arXiv.2507.16075.

[12] Yichen He et al. “PaSa: An LLM Agent for Comprehensive Academic Paper Search”. In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Wanxiang Che et al. Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 11663–11679. ISBN: 979-8-89176-251-0. DOI: 10.18653/v1/2025.acl-long.572. URL: https://aclanthology.org/2025.acl-long.572/.

[13] Sirui Hong et al. “Metagpt: Meta programming for multi-agent collaborative framework”. In: *CoRR* abs/2308.00352 (2023).

[14] Kung-Hsiang Huang et al. *CRMArena-Pro: Holistic Assessment of LLM Agents Across Diverse Business Scenarios and Interactions*. 2025. arXiv: 2505.18878 [cs.CL]. URL: https://arxiv.org/abs/2505.18878.

[15] Kung-Hsiang Huang et al. “CRMArena: Understanding the Capacity of LLM Agents to Perform Professional CRM Tasks in Realistic Environments”. In: *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by Luis Chiruzzo, Alan Ritter, and Lu Wang. Albuquerque, New Mexico: Association for Computational Linguistics, Apr. 2025, pp. 3830–3850. ISBN: 979-8-89176-189-6. DOI: 10.18653/v1/2025.naacl-long.194. URL: https://aclanthology.org/2025.naacl-long.194/.

[16] Kimi. *Kimi Deep Research*. https://www.kimi.com/. 2025.

[17] Langchain. “Open Deep Research”. In: (2025). URL: https://github.com/langchain-ai/open_deep_research?tab=readme-ov-file#results.

[18] Fangyu Lei et al. *Spider 2.0: Evaluating Language Models on Real-World Enterprise Text-to-SQL Workflows*. 2025. arXiv: 2411.07763 [cs.CL]. URL: https://arxiv.org/abs/2411.07763.

[19] Kuan Li et al. “WebSailor: Navigating Super-human Reasoning for Web Agent”. In: *CoRR* abs/2507.02592 (2025). DOI: 10.48550/ARXIV.2507.02592. arXiv: 2507.02592. URL: https://doi.org/10.48550/arXiv.2507.02592.

[20] Zijian Li et al. “WebWeaver: Structuring Web-Scale Evidence with Dynamic Outlines for Open-Ended Deep Research”. In: *arXiv preprint arXiv:2509.13312* (2025).

[21] Nelson F Liu et al. “Lost in the middle: How language models use long contexts”. In: *arXiv preprint arXiv:2307.03172* (2023).

[22] Manus AI. “Context Engineering for AI Agents: Lessons from Building Manus”. In: *Manus Blog* (2025). URL: https://manus.im/blog/Context-Engineering-for-AI-Agents-Lessons-from-Building-Manus.

[23] NVIDIA. "AI-Q NVIDIA Research Assistant Blueprint". In: (2025). URL: https://github.com/NVIDIA-AI-Blueprints/aiq-research-assistant.

[24] Chris Olah et al. "The building blocks of interpretability". In: *Distill* 3.3 (2018), e10.

[25] OpenAI. *Introducing Deep Research*. OpenAI Blog. Accessed: 2025-02-10. Feb. 2025. URL: https://openai.com/blog/deep-research.

[26] Perplexity Team. *Introducing Perplexity Deep Research*. Perplexity Blog. Accessed: 2025-02-10. Feb. 2025. URL: https://blog.perplexity.ai/deep-research.

[27] Akshara Prabhakar et al. *APIGen-MT: Agentic Pipeline for Multi-Turn Data Generation via Simulated Agent-Human Interplay*. 2025. arXiv: 2504.03601 [cs.CL]. URL: https://arxiv.org/abs/2504.03601.

[28] Chen Qian et al. "Chatdev: Communicative agents for software development". In: *arXiv preprint arXiv:2307.07924* (2023).

[29] Cheng Qian et al. *UserBench: An Interactive Gym Environment for User-Centric Agents*. 2025. arXiv: 2507.22034 [cs.AI]. URL: https://arxiv.org/abs/2507.22034.

[30] Cheng Qian et al. *UserRL: Training Interactive User-Centric Agent via Reinforcement Learning*. 2025. arXiv: 2509.19736 [cs.AI]. URL: https://arxiv.org/abs/2509.19736.

[31] Zile Qiao et al. *WebResearcher: Unleashing unbounded reasoning capability in Long-Horizon Agents*. 2025.

[32] Gemini Research. "Gemini Research". In: (2025). URL: https://gemini.google/overview/deep-research/.

[33] GPT Research. "GPT Research". In: (2025). URL: https://github.com/assafelovic/gpt-researcher.

[34] Open Deep Research. "Open Deep Research". In: (2025). URL: https://github.com/langchain-ai/open_deep_research.

[35] Timo Schick et al. "Toolformer: Language models can teach themselves to use tools". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 68539–68551.

[36] Samuel Schmidgall et al. *Agent Laboratory: Using LLM Agents as Research Assistants*. 2025. arXiv: 2501.04227 [cs.HC]. URL: https://arxiv.org/abs/2501.04227.

[37] Noah Shinn et al. "Reflexion: Language agents with verbal reinforcement learning". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 8634–8652.

[38] Aditi Singh et al. "Agentic Retrieval-Augmented Generation: A Survey on Agentic RAG". In: *arXiv preprint arXiv:2501.09136* (2025).

[39] Liangcai Su et al. *Scaling Agents via Continual Pre-training*. 2025.

[40] SambaNova Systems. "Open-Source Deep Research Agents for the Enterprise". In: *SambaNova AI Blog* (2025).

[41] Zhengwei Tao et al. "WebShaper: Agentically Data Synthesizing via Information-Seeking Formalization". In: *CoRR* abs/2507.15061 (2025). DOI: 10.48550/ARXIV.2507.15061. arXiv: 2507.15061. URL: https://doi.org/10.48550/arXiv.2507.15061.

[42] Zhengwei Tao et al. "Webshaper: Agentically data synthesizing via information-seeking formalization". In: *arXiv preprint arXiv:2507.15061* (2025).

[43] Michael Wooldridge. *An introduction to multiagent systems*. John wiley & sons, 2009.

[44] Qingyun Wu et al. “Autogen: Enabling next-gen LLM applications via multi-agent conversations”. In: *First Conference on Language Modeling*. 2024.

[45] Tianbao Xie et al. *OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments*. 2024. arXiv: 2404.07972 [cs.AI]. URL: https://arxiv.org/abs/2404.07972.

[46] John Yang et al. *InterCode: Standardizing and Benchmarking Interactive Coding with Execution Feedback*. 2023. arXiv: 2306.14898 [cs.CL]. URL: https://arxiv.org/abs/2306.14898.

[47] John Yang et al. *SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering*. 2024. arXiv: 2405.15793 [cs.SE]. URL: https://arxiv.org/abs/2405.15793.

[48] Shunyu Yao et al. “ReAct: Synergizing Reasoning and Acting in Language Models”. In: *International Conference on Learning Representations (ICLR)*. 2023.

[49] Li S Yifei et al. “ResearchQA: Evaluating Scholarly Question Answering at Scale Across 75 Fields with Survey-Mined Questions and Rubrics”. In: *arXiv preprint arXiv:2509.00496* (2025).

[50] You.com. *DeepConsult*. https://github.com/Su-Sea/ydc-deep-research-evals. Accessed: 2025-09-07. 2025.

[51] Jifan Yu et al. “EKRAG: Benchmark RAG for Enterprise Knowledge Question Answering”. In: *Proceedings of the 4th Workshop on Knowledge-Augmented Methods for NLP* (2025), pp. 152–159.

[52] Jianguo Zhang et al. *xLAM: A Family of Large Action Models to Empower AI Agent Systems*. 2024. arXiv: 2409.03215 [cs.CL]. URL: https://arxiv.org/abs/2409.03215.

[53] Wenlin Zhang et al. “Deep Research: A Survey of Autonomous Research Agents”. In: *arXiv preprint arXiv:2508.12752* (2025).

A Prompts

Here we provide all the prompts used in EDR in the order in which they are invoked during execution.

1. Initial task decomposition prompt. (Figure 4). Generates the initial `todo.md` plan by decomposing the research query into 3–5 structured, high-priority research tasks that establish the foundation for subsequent retrieval and reasoning.

Initial Task Decomposition Prompt

```
You are creating an initial research plan for the topic: "{research_topic}"

Initial Query: "{initial_query}"
Research Context: {research_context if research_context else "Starting fresh research"}

Decompose this query into 3–5 actionable research tasks. Return a JSON array with each task having:
• "description": Clear, actionable task (string)
• "priority": 1–10 (integer, higher = more important, default=5)
• "type": "research" (always "research")

Focus on: understanding the topic, gathering information, identifying key aspects, and building foundational knowledge.

Example for "Impacts of Generative AI on Scientific Research":

<answer>
[
  {"description": "Survey major applications of generative AI in scientific discovery", "priority": 8, "type": "research"},
  {"description": "Identify key papers and institutions leading AI-assisted science research", "priority": 7, "type": "research"},
  {"description": "Examine methodological advances enabled by generative models in ...", "priority": 6, "type": "research"},
  {"description": "Assess challenges and ethical considerations of AI-generated scientific results", "priority": 5, "type": "research"}
]
</answer>
CRITICAL: Wrap JSON in <answer>tags.
Output ONLY valid JSON.
```

Figure 4: Prompt for generating the initial research task plan with 3–5 prioritized tasks guiding the first research loop.

2. Task-to-Query breakdown and execution prompt. (Figure 5). Defines the core query decomposition and execution mechanism. It guides the model in generating precise, context-aware search queries from research tasks, ensuring alignment with user-provided knowledge and steering plans.

3. Reflection prompt. (Figure 6). Governs the post-synthesis evaluation stage of the research pipeline. It systematically audits coverage, identifies missing knowledge, and updates the task queue—marking completed, cancelled, or newly added search tasks. The prompt ensures iterative refinement until all critical knowledge gaps are closed.

B System Implementation Details

B.1 API Overview

The system exposes a comprehensive REST API built on FastAPI, providing programmatic access to all capabilities through well-designed endpoints and advanced features.

Research Endpoints.

- `POST /deep-research` – performs comprehensive research on topics with optional steering capabilities.

Primary Search Query Generation Prompt

```
<TIME_CONTEXT>
Current date: {current_date}
Current year: {current_year}
One year ago: {one_year_ago}
</TIME_CONTEXT>

<AUGMENT_KNOWLEDGE_CONTEXT>
{AUGMENT_KNOWLEDGE_CONTEXT}
</AUGMENT_KNOWLEDGE_CONTEXT>

You are an expert research assistant tasked with generating a targeted web search query. The query will gather in-depth information
related to a specific topic through comprehensive web search.

<AUGMENT_KNOWLEDGE_INTEGRATION>
CRITICAL: When user-provided external knowledge is available, it should be treated as highly trustworthy and authoritative.
1. Prioritize uploaded knowledge.
2. Complement, don't duplicate.
3. Validate and expand.
4. Focus on gaps.
Use uploaded knowledge to guide targeted query generation.
</AUGMENT_KNOWLEDGE_INTEGRATION>

<ANTI_ASSUMPTION_DIRECTIVE>
CRITICAL: Do not assume current roles, statistics, or entities. Use generic, time-aware phrasing such as "current leadership" or "latest
data."
</ANTI_ASSUMPTION_DIRECTIVE>

<RECENCY_SENSITIVITY_FRAMEWORK>
For time-sensitive topics, append temporal markers such as "current," "latest," or {current_year}.
Example: "Company X current CEO {current_year}."
</RECENCY_SENSITIVITY_FRAMEWORK>

<TOPIC_ANALYSIS_AND_STRATEGY>
Decompose each task into 2–5 subtopics covering major facets—historical, technical, practical, and recent developments.
</TOPIC_ANALYSIS_AND_STRATEGY>

<RESEARCH_STAGE_GUIDANCE>
If first query → broad overview.
If follow-up → target specific gaps.
</RESEARCH_STAGE_GUIDANCE>

<TOPIC> {research_topic} </TOPIC>
<RESEARCH_CONTEXT> {research_context} </RESEARCH_CONTEXT>

<STEERING_INSTRUCTIONS>
Read the todo.md plan and follow active steering tasks, priorities, and constraints.
{task_list}
</STEERING_INSTRUCTIONS>

<QUERY_REQUIREMENTS>
Queries must: align with todo.md tasks, avoid boolean operators, remain under 400 characters, and incorporate domain-specific terms
when useful.
</QUERY_REQUIREMENTS>

<FORMAT_GUIDELINES>
Return JSON format:

{
  "query_complexity": "complex",
  "main_query": "agentic RAG systems architecture benefits",
  "tasks": [
    {"name": "Core Architecture", "query": "...", "aspect": "..."},
    {"name": "Strategic Retrieval", "query": "...", "aspect": "..."}
  ]
}
</FORMAT_GUIDELINES>

<AVOIDING_ASSUMPTION_EXAMPLES>
Incorrect: "John Smith Company X CEO background"
Correct: "current Company X CEO name current_year"
</AVOIDING_ASSUMPTION_EXAMPLES>

<EXAMPLES>
Example: "streaming video platform market share 2021–2025"
Example: "current Country X president {current_year}"
Example: "agentic RAG systems implementation case studies"
</EXAMPLES>

Provide your response in JSON format only.
```

Figure 5: Prompt defining the query decomposition and execution process for targeted, time-sensitive, and context-aware research queries.

- `GET /research-status` – checks API operational status.
- `GET /stream/{stream_id}` – provides real-time streaming updates via Server-Sent Events (SSE).

File Analysis Endpoints.

- `POST /api/files/upload` – uploads and analyzes individual files.
- `POST /api/files/batch-upload` – uploads multiple files simultaneously.
- `GET /api/files/{file_id}/analysis` – retrieves analysis results.
- `POST /api/files/{file_id}/analyze` – manually triggers analysis.
- `GET /api/files/{file_id}/content` – downloads original files.
- `GET /api/files/{file_id}/status` – checks processing status.
- `GET /api/files/` – lists all uploaded files with filtering options.
- `DELETE /api/files/{file_id}` – removes files from the system.

Database Integration Endpoints.

- `POST /api/database/upload` – uploads SQLite, CSV, and JSON database files.
- `GET /api/database/list` – retrieves all uploaded databases.
- `GET /api/database/{database_id}/schema` – accesses database schema information.
- `POST /api/database/query` – executes natural language queries against databases.
- `DELETE /api/database/{database_id}` – removes uploaded databases.

Interactive Steering Endpoints.

- `POST /steering/message` – sends natural language steering messages during research.
- `GET /steering/plan/{session_id}` – retrieves current research plans in Markdown format.
- `GET /steering/status/{session_id}` – monitors real-time plan status.
- `GET /steering/interactive/session/{session_id}` – supports frontend polling compatibility.
- `GET /steering/sessions` – lists all steerable research sessions.
- `GET /steering/examples` – accesses natural language steering examples.

Advanced API features include streaming responses using Server-Sent Events (SSE) with heartbeat mechanisms for real-time updates, background task processing with asynchronous execution and status tracking, comprehensive error handling with detailed error responses and recovery suggestions, CORS support for cross-origin resource sharing in web applications, automatic API documentation generation using OpenAPI/Swagger standards accessible at /docs, cache control middleware for optimal performance, React frontend integration with static file serving and catch-all routing, multiple LLM provider authentication methods including OpenAI, Anthropic, Groq, and SambaNova API keys configured through environment variables, and rate limiting with configurable delays and exponential backoff for API protection.

B.2 Frontend Architecture

The frontend architecture is implemented using React 18 and TypeScript, leveraging concurrent rendering, automatic batching, and improved error boundaries for high performance and robust developer experience. TypeScript ensures comprehensive type safety, strict type checking, and seamless refactoring. The interface design employs Material-UI for accessible, responsive, and themable components that adhere to WCAG guidelines, complemented by advanced UI elements such as sortable data tables, validated forms, and interactive visualizations. Tailwind CSS provides a utility-first styling framework with custom design tokens, responsive breakpoints, and optimized CSS generation for minimal bundle sizes, along with dark mode and accessibility utilities. Data fetching and state synchronization are managed by React Query, which supports intelligent caching, background refetching, optimistic updates, and automatic retry mechanisms to ensure a smooth and responsive user experience.

B System Implementation Details

Research Loop Reflection Prompt

```
You are an expert research evaluator analyzing a summary about {research_topic}.

<GOAL>
Conduct a structured evaluation to determine:
1. Whether the summary is sufficiently comprehensive and accurate overall
2. Which SECTIONS of the summary need more detail or data
3. What specific knowledge gaps exist
4. How to formulate a targeted follow-up query to address those gaps
</GOAL>

<TODO_DRIVEN_REFLECTION>
The research loop maintains a dynamic task queue.
- PENDING TASKS: {pending_tasks}
- ALREADY COMPLETED: {completed_tasks}
- USER STEERING MESSAGES (if any): {steering_messages}

YOUR TASK - Update the todo list:
1. MARK COMPLETED: Which PENDING tasks were addressed in this research loop?
- Review ONLY the pending tasks listed above
- Check if the current summary now covers those areas
- Return their task_ids in "mark_completed" list
- IMPORTANT: ONLY evaluate the PENDING tasks - do NOT mark tasks from "ALREADY COMPLETED"

2. CANCEL TASKS: Which pending tasks are no longer relevant?
- Based on current findings OR user steering messages
- Cancel tasks that don't align with the research direction
- Return their task_ids in "cancel_tasks" list

3. ADD NEW TASKS: What critical areas still need research?
- Identify knowledge gaps in the current summary
- If user sent steering messages, add tasks for their requests
- IMPORTANT: Check "ALREADY COMPLETED" section to avoid creating duplicate tasks
- Each new task = one specific search topic
- Keep tasks simple and searchable (e.g. "Research X's work at Y")
- Return as objects with "description" and "rationale" in "add_tasks" list

4. CLEAR MESSAGES: Which steering messages can be cleared?
- Return list of indices of messages that have corresponding tasks created or are no longer relevant

RULES:
- ONLY mark tasks as completed if they're in the "PENDING TASKS" section above
- DO NOT create tasks similar to those in "ALREADY COMPLETED" section
- Each task = one Tavily search query
- Focus on WHAT to search, not HOW to organize
</TODO_DRIVEN_REFLECTION>

<KNOWLEDGE_GAP_PRIORITIZATION>
Use this structured framework to identify and prioritize knowledge gaps:

1. Gap categorization by impact type:
- Critical gaps: Missing information that fundamentally undermines main conclusions
- Contextual gaps: Missing background or explanatory information that would enhance understanding
- Detail gaps: Missing specifics that would provide greater precision or confidence
- Extension gaps: Related areas that would broaden perspective but aren't central

2. Prioritization matrix:
- Priority 1 (Highest): Critical gaps in central topic areas directly related to the original research topic
- Priority 2: Critical gaps in peripheral areas OR contextual gaps in central areas
- Priority 3: Detail gaps in central areas OR contextual gaps in peripheral areas
- Priority 4 (Lowest): Extension gaps OR detail gaps in peripheral areas
- NEVER prioritize gaps that would lead research away from the original research topic
</KNOWLEDGE_GAP_PRIORITIZATION>

<QUANTITATIVE_SUFFICIENCY_METRICS>
Use these measurable criteria to objectively assess research completeness:

1. Coverage Completeness Score:
- Assign a completion percentage to each identified subtopic:
* 90-100%: Comprehensive coverage with specific details and examples
* 70-89%: Substantial coverage but missing some specific details
* 40-69%: Basic coverage that outlines key points but lacks depth
* 0-39%: Minimal coverage or completely missing
- Calculate overall coverage by averaging across all required subtopics
- Research is considered "complete" ONLY when average coverage exceeds 95% AND no critical subtopic falls below 85%

2. Source Quality Assessment:
- Measure proportion of information from high-authority sources
- Research is considered "complete" only when at least 80% of critical assertions are supported by Tier 1-2 sources

3. Evidence Density Evaluation:
- For technical/scientific topics: ≥ 5 specific metrics or measurements per key assertion
- For market/business topics: ≥ 4 quantifiable data points per major segment
- For biographical topics: ≥ 5 specific career achievements, publications, projects
</QUANTITATIVE_SUFFICIENCY_METRICS>

<FORMAT>
Return a JSON with:
• "research_complete": bool (true if ≥ 95% coverage, no critical gaps)
• "section_gaps": map of section → missing details
• "priority_section": section with most pressing gap
• "knowledge_gap": concise explanation of missing info
• "follow_up_query": ≤ 400 char search query (none if complete)
• "evaluation_notes": brief overall remarks
• "research_topic": unchanged original topic
• "todo_updates": { "mark_completed": [...], "cancel_tasks": [...], "add_tasks": [{"description": "...", "rationale": "..." }] }
Task IDs must match pending list. Add new tasks only for uncovered, unique items.
</FORMAT>
```

Figure 6: Structured reflection prompt for evaluating research completeness and dynamically updating `todo.md` in the Research Loop.