

RAGAPHENE

A RAG Annotation Platform with Human ENhancements and Edits

Kshitij Fadnis*, Sara Rosenthal*, Maeda Hanafi, Yannis Katsis, Marina Danilevsky

{kpfadnis, sjrosenthal}@us.ibm.com

IBM Research - AI

Abstract

Retrieval Augmented Generation (RAG) is an important aspect of conversing with Large Language Models (LLMs) when factually correct information is important. LLMs may provide answers that appear correct, but could contain hallucinated information. Thus, building benchmarks that can evaluate LLMs on multi-turn RAG conversations has become an increasingly important task. Simulating real-world conversations is vital for producing high quality evaluation benchmarks. We present RAGAPHENE, a chat-based annotation platform that enables annotators to simulate real-world conversations for benchmarking and evaluating LLMs. RAGAPHENE has been successfully used by approximately 40 annotators to build thousands of real-world conversations.

1 Introduction

Chat-Based web tools for conversing with Large Language Models (LLMs) such as ChatGPT (OpenAI, 2024) and Claude (Anthropic, 2024) have become very popular. One common use is the seeking of factual information where Retrieval Augmented Generation (RAG) is extremely important to ensure the model is answering faithfully to the relevant passages and not hallucinating. Thus, the ability to evaluate the performance of LLMs on multi-turn RAG-based conversations has become increasingly important and largely overlooked until recently (Rosenthal et al., 2026; Katsis et al., 2025; Dziri et al., 2022; Feng et al., 2021; Kuo et al., 2024; Es et al., 2024). Multi-Turn RAG is particularly important for enterprise use cases where domains are specific and there may be unique requirements such as specialized retrievers, generators and custom prompts (Sharma et al., 2024). Building a challenging high-quality Multi-Turn RAG benchmark requires in-depth fine-grained human annotation that employs these requirements. Existing annotation

*Authors Contributed Equally

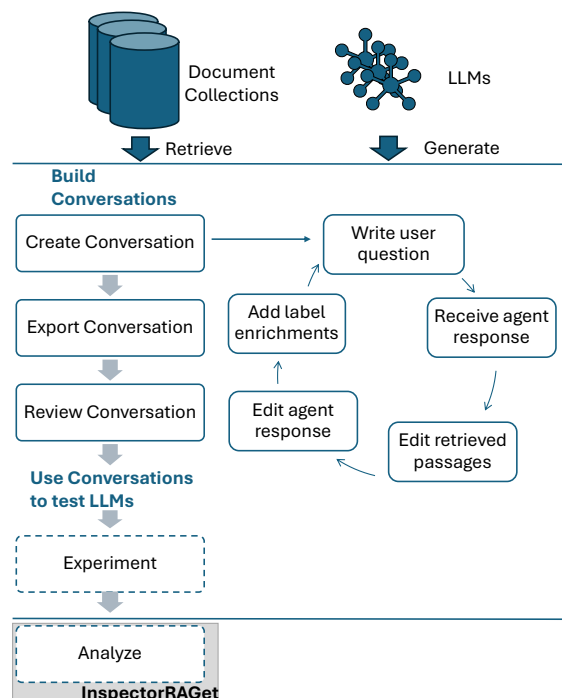


Figure 1: RAGAPHENE Pipeline: Using RAGAPHENE, the user creates a conversation against a chosen retriever and generator which can be exported in JSON format for further review, experimentation and analysis.

platforms (BasicAI, 2019; HumanSignal, 2020; LabelBox, 2024; Joko et al., 2021) include some basic features for annotating conversational data, such as thumbs up/down of questions/responses, adding metadata and tagging entities in the conversation. However, they do not support the creation of conversational multi-turn RAG datasets with real-time agent response generation. The First-Aid platform (Menini et al., 2025) is a conversational annotation interface which does include real-time agent response generation and editing. However, it doesn't have any other feedback mechanisms (e.g. thumbs up/down) and does not incorporate a retrieval component - the conversations are generated based on a small set of pre-loaded documents rather than

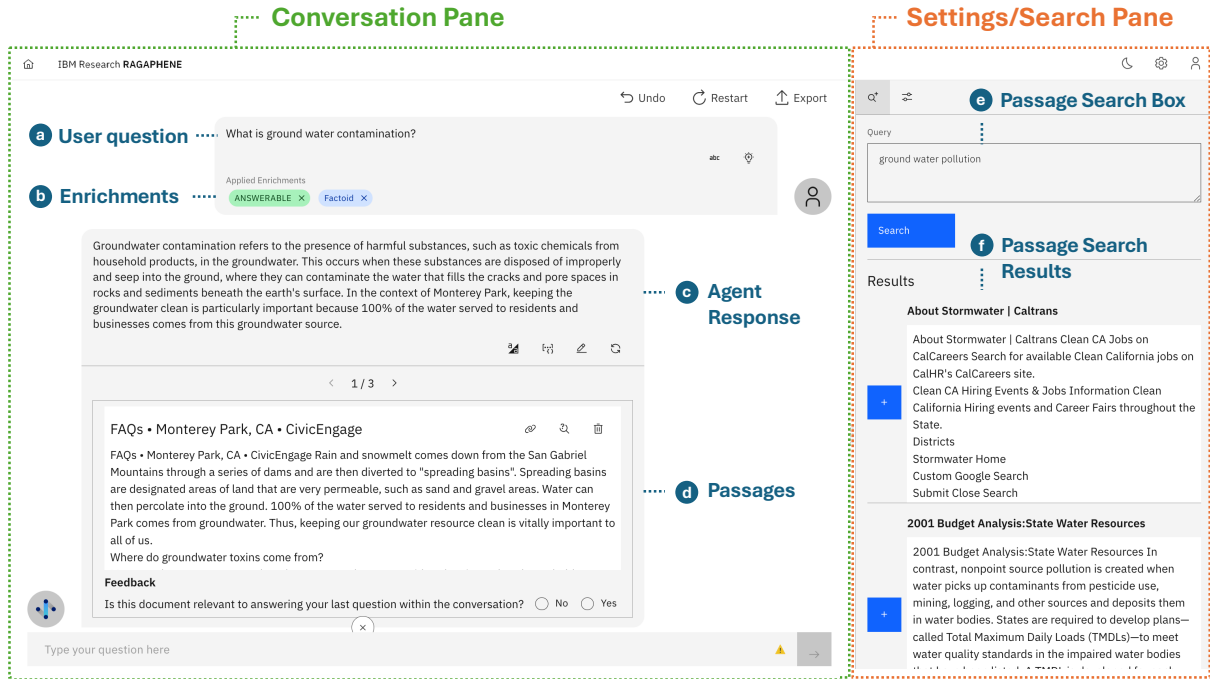


Figure 2: Screenshot of RAGAPHENE’s create mode annotated with its main components.

passages dynamically retrieved from a (potentially very large) underlying corpus. Furthermore, our annotation tool enables editing/correcting both the relevant passages and output which are both important pieces for simulating real-world conversations.

We present RAGAPHENE, a chat-based annotation platform for having conversations with an LLM that are grounded in an existing corpus to ensure faithfulness, primarily for building Multi-turn RAG benchmarks. The pipeline of RAGAPHENE is shown in Figure 1. Our platform adopts the RAG pipeline to enable a user to chat with an LLM agent with real-time retrieval and generation while providing the ability to improve the conversation when the retriever and/or generator fails. We allow the user to easily integrate with their desired retriever and generator and provide the ability to improve the conversation which enables users to create high-quality conversations that can be used to evaluate and improve RAG systems.

A platform for building and evaluating a challenging multi-turn should have the following:

RAG Chat with customizable retrieval and generation is an important scenario, particularly in industry where a domain-specific corpus is likely.

Enhanced Feedback is desirable, as typical chat feedback of thumbs up/down is limiting in its use. It does not give the user the opportunity to troubleshoot when the agent response is not satisfactory which is necessary for building high-quality

conversations.

Evaluation and Analysis is important for domain experts and engineers to rapidly evaluate how different models perform in their domain by building a small benchmark.

Our contributions are as follows:

- **RAGAPHENE:** A RAG annotation chat platform integrated with multiple interchangeable retrievers and generators for domain-specific use cases.¹
- **Enhanced user feedback** including adjusting passage retrieval and improving/repairing responses, validated through a user study with 31 professional annotators.
- **Real-time small-scale evaluation and analysis** of the full RAG pipeline on RAG conversations created in RAGAPHENE.

2 RAGAPHENE Platform

The RAGAPHENE platform pipeline is shown in Figure 1. It allows for integration using a desired corpus with any retriever (e.g. BM25, Elser²) and generator (e.g. Llama 3 (Grattafiori et al., 2024), GPT 4 (OpenAI et al., 2024)). The default settings used in our work are an ELSERV1 (ElasticSearch

¹RAGAPHENE is released under an Apache 2.0 license at <https://github.com/ibm/RAGaphene>.

²<https://www.elastic.co/guide/en/machine-learning/current/ml-nlp-elser.html>

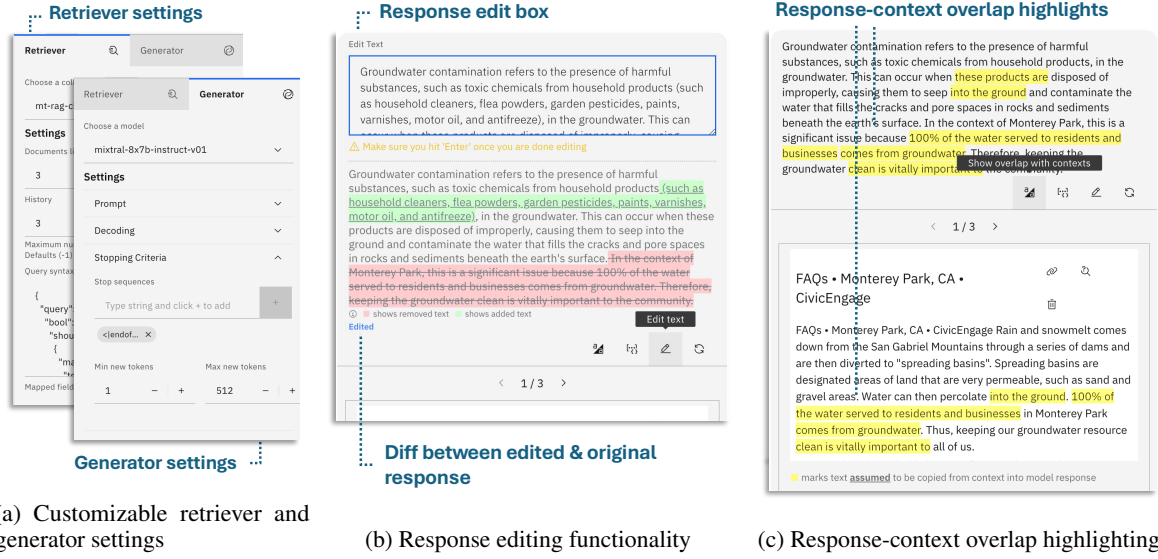


Figure 3: Screenshots of selected functionalities of RAGAPHENE.

8.10¹) retriever populated with the corpora from MTRAG (Katsis et al., 2025) and Mixtral 8X7b Instruct (Jiang et al., 2024) as the generator. RAGAPHENE consists of three main modes - Create, Review and Experiment - and integration with InspectorRAGet for analysis to provide a full benchmarking and evaluation life-cycle. We adopt a stateless approach to ensure privacy; all conversations can be preserved by exporting in JSON format. Further details are available in Appendix A.

2.1 Create Mode

In create mode, a user or annotator can use RAGAPHENE as a chat interface (Figure 2) to interact with an LLM. The user is first provided with configuration settings on the right hand side to choose the desired retriever and generator (Figure 3a). In the retriever settings, they can pick a specific collection of documents and adjust settings such as how many passages to return, how to formulate the query, and other more sophisticated features such as how to render retrieved results. In the generator settings, they can choose a model to chat with, adjust the prompt and other settings such as decoding and number of tokens. On the left-hand side, the user can chat with the generator chosen in the settings. During conversation creation, tips to assist the user appear on the upper portion of the interface to encourage useful feedback. The completed conversation can be exported as a structured JSON for future use. We describe the process of creating conversation turns, as well as features such as adding question enrichments, answer repair (Figure 3b),

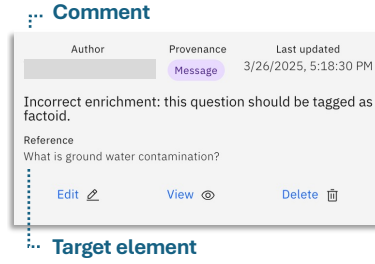


Figure 4: Screenshot of review feedback functionality.

and passage search (Figure 2) in Section 3.1.

2.2 Review Mode

The review mode supports using RAGAPHENE as an annotation platform to review previously created conversations and decide whether the conversation should be kept in the benchmark (accept) or not (reject). In this mode, an existing conversation (or batch of conversations) is loaded and the retriever and generator cannot be run. At each turn, the annotator can modify the enrichments, edit the answers, and change the relevance score of the passages. On the left hand side there is a comment section (Figure 4) where feedback can be provided for all changes made. Specific comments can be left by highlighting text in the conversation and general comments can be left for overall feedback. Once one conversation review is complete, the annotator accepts or rejects the conversation and then continues to the next one. Review mode keeps a browser-local snapshot and offers to resume after a refresh or unexpected tab closure. Reviewed conversations can be exported as a structured JSON

file. We describe the review flow in Section 3.2.

2.3 Experiment Mode

The experiment mode provides a way for users to estimate the complexity of their newly created conversational data by running a small scale experiment against various LLMs. The key advantage of this mode is that it operates directly on conversations created in RAGAPHENE, eliminating the need to set up a separate evaluation framework or reconcile data formats. The prediction by the LLM is compared to the target response that was composed during create mode. The user uploads their conversation data and chooses which part of the conversation(s) to evaluate: the midpoint turn of each conversation (default), the first turn, the last turn, or all turns. Each split becomes a task to be evaluated. Additionally, a user can choose to experiment either with the model’s generation capabilities by keeping the retrieved documents constant, or to execute the full RAG pipeline using different retriever and generator combinations. The user can easily adjust the retriever and generator configurations and select evaluation metrics before launching the experiment, which can be monitored live (Figure 5).

The platform has built-in metrics such as response length, ROUGE, Recall and LLM-as-a-Judge which can be easily extended to run complex compute intensive metrics. Our LLM-judge metrics target RAG-specific failure modes: grounded (claims are supported by the retrieved context), pertinent (the response addresses the question), coherent (the response is consistent with the rest of the conversation), concise (the response has no unnecessary information). See Katsis et al. 2025 for further details. Each judge has a prompt template and a scoring rubric which can be viewed during configuration. The judge model reasons over the claims or context before giving a verdict, Yes, No, or Unsure, inside a tagged span; an unparseable response is scored as an Error rather than a pass or a fail. New judges can be added without changing the evaluation pipeline.

Once the experiment is complete, the results can be exported in a structured JSON format. We describe the flow of the experiment process in detail in Section 3.3. We purposefully restrict the size of the dataset in this mode to a maximum of 100 tasks. RAGAPHENE is not intended to be a full blown evaluation platform, but rather a quick way of identifying and monitoring data on a small scale.

Larger experiments should be run offline.

2.4 InspectorRAGet

InspectorRAGet (Fadnis et al., 2025) is an introspective platform for debugging and analyzing model output and evaluation metrics. Once a conversation has been created and experiments run, it is important to be able to analyze the performance in a general and detailed manner across multiple models and metrics to enable quick decision making regarding model and metric choice for deployed systems. The output from the experiment mode can be loaded into InspectorRAGet to achieve this.

3 RAGAPHENE Workflow

In this section we describe the typical workflow of a user in RAGAPHENE for the different modes: Create, Review, and Experiment.

3.1 Conversation Creation

Our custom chat enables users to chat with a live RAG agent consisting of a retriever and generator and correct the retriever and generator outputs as desired. In many scenarios, we expect the conversation is domain specific and has a pre-decided corpus such as for a specific company or topic. In particular, after customizing the retriever and generator settings (Figure 3a), users can use the application to perform the following actions at every turn in the conversation:

- (i) **Write user question:** The conversation begins by writing an initial question for the associated domain (Figure 2a). We expect the user to have some basic domain knowledge, but they ask a question without seeing a document.
- (ii) **Receive agent response:** Once the user writes a question, the agent calls the retriever to retrieve potentially relevant passages and then the passages along with an appropriate prompt are sent to the generator to produce a response which is presented to the user (Figure 2c). The passages may not always be relevant and the response may not be appropriate. The next steps provide the user with a means of improving the passages and response to help the user receive a helpful and correct answer.
- (iii) **Edit retrieved passages:** The passages returned by the retriever can be reviewed and edited to generate a set of passages that are indeed relevant to the question (Figure 2d). This includes (a) discarding (or marking as irrelevant) passages returned by the retriever that are deemed non-relevant

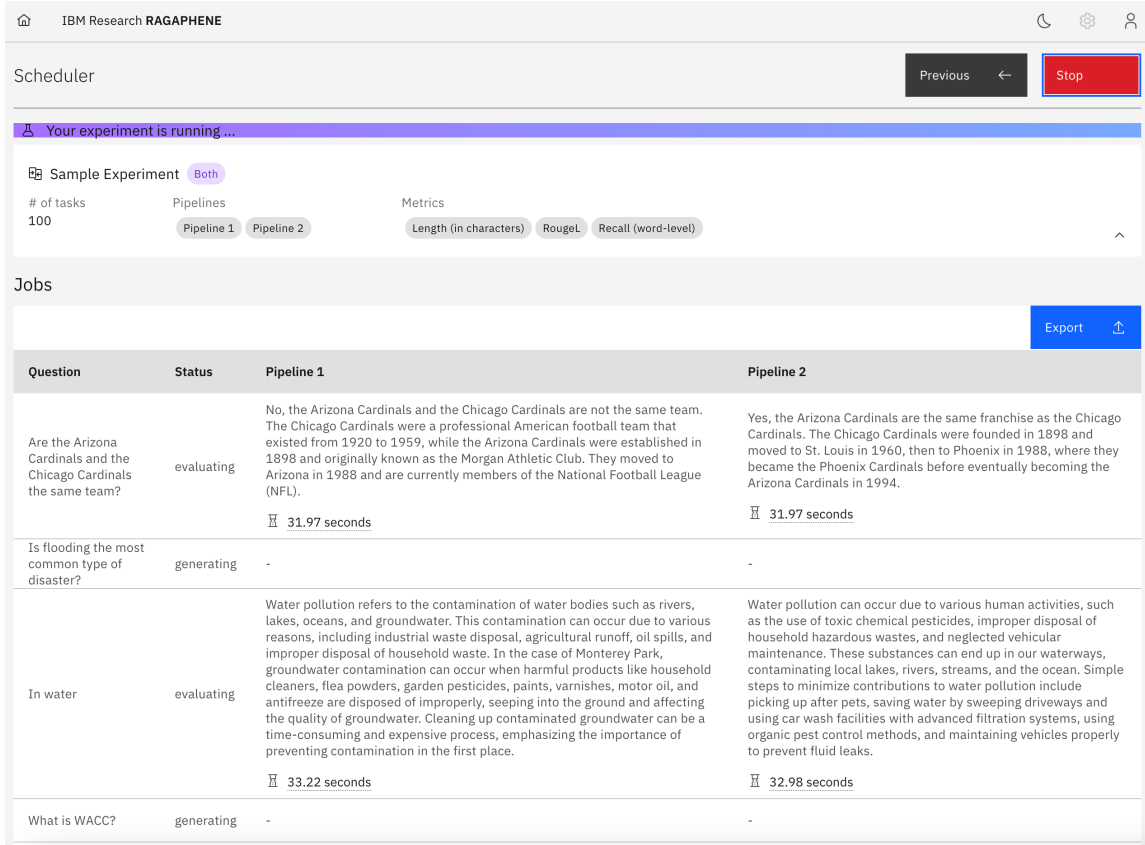


Figure 5: Screenshot of RAGAPHENE’s experiment mode showing an experiment in progress.

to the question, as well as (b) adding other relevant passages present in the corpus that were missed by the retriever. To facilitate the latter, we provide a separate search interface on the side that allows users to try alternate formulations of their question to bring in additional relevant passages (Figure 2e and 2f). The user can then regenerate the response so that it is based on the latest relevant passages. The set of relevant passages can be used to evaluate and improve the retrieval component of RAG systems. We provide reminder tips if they forget to mark relevant passages.

(iv) **Edit agent response:** Once the relevant passages are identified, if the response is still not appropriate, users can edit the generated agent response to repair and improve it using the relevant passages. This helps ensure that the next turn in the conversation will be based on the correct information and can be used for future LLM improvement. As the users edit the response, RAGAPHENE shows the diff between the original and edited response (Figure 3b). Finally, to help users check if the response is faithful to the passages, the platform highlights the lexical overlap between the response and the passages (Figure 3c).

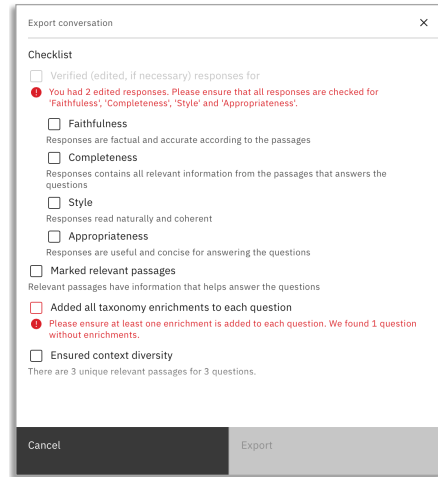


Figure 6: Screenshot of checklist during export.

(v) **Add Label Enrichments:** Each turn can also be enhanced with tags, such as question type (e.g. factoid, opinion), answerability (e.g. answerable, unanswerable), and multi-turn (e.g. clarification, follow up) (Figure 2b). We provide reminder tips to encourage this feedback if they forget to add enrichments. These enrichments provide a valuable way of exploring the data during evaluation.

(vi) **Export Conversation:** When the person is

finished chatting they can save their conversation by exporting it as a JSON. This can later be used to continue the conversation, or for evaluation of RAG systems. During export they are provided with optional checkboxes to encourage high quality including providing statistics about the conversation and enrichments that were added (Fig 6).

3.2 Conversation Review

Creating conversations is a comprehensive task with many parts. In order to use these conversations for evaluation they need to be of high quality with all repairs made. The review mode specifically targets users that are annotators tasked with providing high quality conversations for evaluation. In this mode a reviewer will receive a batch of conversations. The reviewer reads through each conversation and can accept the conversation as is, accept it but edit the conversation, or reject it:

(i) **Accept Conversation:** If the conversation is a good conversation that flows well the reviewer can accept the conversation as is. However, in many cases there will be some adjustment needed.

(ii) **Accept with Edits:** Even if the conversation is good, it may still need small tweaks or repairs to improve responses according to desired properties. For example, there may be a sentence in the response that is not faithful to the passages or is misspelled. In addition to changing the response, the reviewer can change passage relevance and adjust the enrichments. A reviewer cannot change the questions or query for more relevant passages as that could alter the conversation significantly causing the initial intent to no longer be valid.

(iii) **Reject:** In some cases the reviewer may attempt to repair the conversation but the changes can be too significant. There can be a repeated or unnecessary question that disrupts the conversation flow. It can also be clear that the passages provided are not sufficient. In this case the reviewer can reject the conversation instead of repairing it.

(iv) **Feedback:** In addition, the reviewer can leave specific and general comments for the creator of the conversation. They can highlight specific parts/elements of the conversation and leave textual comments (Figure 4). Feedback is particularly useful when a conversation is rejected.

3.3 Running Experiments

We provide the experiment mode to give users the ability to run quick experiments on a small scale which can then be exported to InspectorRAGet for

quick decision making. In this mode, the user can set up an experiment on a set of conversations and choose the tasks, models and metrics to evaluate.

(i) **Choose Tasks:** The user first uploads the conversation and selects the slice of the data they would like to evaluate. This can be all turns of the conversation or different subsets.

(ii) **Choose Models:** The user can choose to evaluate the retriever, generator, or full RAG pipeline. The user then sets up the retriever and generator model settings for each model they want to evaluate. This includes adjusting parameters and prompts.

(iii) **Choose Metrics:** Next, the user chooses the metrics to evaluate on each task. We provide built-in metrics such as Rouge, Recall and LLM judges.

(iv) **Run Experiment:** Once satisfied with the experimental setup, the user launches an experiment. The time to run the experiments depends on the number of models, tasks, and metrics being evaluated. Once completed, the evaluation can be exported and loaded in InspectorRAGet for analysis.

4 Use Cases

We describe three specific uses and applications of RAGAPHENE: Multi-Turn RAG annotation, RAG chat assistant and Real-Time Evaluation.

4.1 Multi-Turn RAG Annotation

The primary use of RAGAPHENE is to create a challenging benchmark that simulates real world conversations to evaluate the performance of RAG systems. As an annotation task, the various feedback and repair mechanisms provide a valuable resource for creating a strong benchmark that is diverse and challenging for LLMs. While interacting with the RAG-based agent, the annotators add feedback that enriches, repairs, and improves the responses. They then export their conversation so it can be used for evaluation and insights. Creating such conversations is a sophisticated process and prone to errors. We specifically employ the review workflow to ensure the conversations for improvement are of high quality. We have successfully created 110 high quality Multi-Turn RAG conversations entirely using RAGAPHENE, released as a public benchmark (Katsis et al., 2025). In addition, the platform is being used for ongoing work in this area, with over 5,000 conversations created and over 1,000 conversations reviewed by over 30 annotators.

Platform Feature	μ	σ
Editing the agent responses	4.26	0.82
Highlights for overlapping text in the context and response	4.13	0.88
Regenerating the agent response	4.10	0.98
Requery Tool	3.97	1.37
Checklist before export	3.74	1.34
Marking contexts relevant/irrelevant	3.26	1.43
Hints	3.19	1.42
Enriching the questions	2.74	1.24

Table 1: RAGAPHENE features ranked based on their impact to the quality of created conversational data. A higher average value (μ) indicates a bigger impact (1 is “No decrease [in data quality] at all” and 5 is “Extreme decrease in quality”).

4.2 RAG Chat Assistant

Although primarily built as an annotation platform, RAGAPHENE can also be used as a RAG-based chat platform by a standard user seeking information. Unlike RAGFlow (Infiniflow, 2024), which provides RAG chat capabilities, RAGAPHENE also allows users to repair and redirect the agent by searching for better passages and improving answers.

4.3 Real-Time Evaluation

A typical client needs to perform due diligence prior to selecting a vendor to power their chat-based assistant using proprietary documents. This typically involves exploring multiple LLMs and retriever engines to identify the best setup. Using RAGAPHENE, they can create domain specific conversations on their content that can power evaluations, performance analysis (via InspectorRAGet (Fadnis et al., 2025)), and go / no-go decisions for stakeholders. We have several clients that use RAGAPHENE to perform such due diligence.

5 User Study

We performed a user study to understand the importance of our tool for helping annotators create high-quality conversations for benchmarking RAG. We surveyed 31 professional annotators (21 females, 10 males), with 13 of them having more than 3 years experience of annotating. At the time of this writing, the tool has been in production for more than 9 months, with most of the annotators creating and reviewing 75+ conversations. Creating and

annotating a conversation for RAG applications is a hard and time-consuming task for human annotators (Hanafi et al., 2025); most annotators reported that it took them on average more than 30 minutes to create a single high-quality conversation.

The survey focused on questions for 8 platform features: “On a scale of 1 to 5 if we were to remove <FEATURE> from the platform, how would this decrease the quality of the conversations you create?”, where 1 is “No decrease at all” and 5 is “Extreme decrease in quality”. Table 1 shows the average Likert scores of each platform feature.

Although our professional annotators reported an advanced beginner-level of RAG understanding (on a 1 to 5 Likert scale; $\mu = 2.61$, $\sigma = 1.33$), our survey results show that they recognized the necessity for platform features that involve analyzing and fixing the generator and retriever model outputs for high-quality RAG data creation (such as improving the set of retrieved contexts or editing the agent response). Certain features, such as marking contexts as relevant or irrelevant and enriching the questions are seen as annotation artifacts rather than part of the process of *creating* a conversation. One annotator says, “[Marking contexts as relevant/irrelevant] is perhaps useful to the end-user of the data, but is not necessary for creating the data in the tasks...”.

6 Conclusion

We present RAGAPHENE, a platform for creating and evaluating high quality conversations for RAG. We provide advanced features to improve passage search, repair responses, and enrich the questions to aid in conversation creation. We also include the option of performing small-scale evaluation with integration to InspectorRAGet (Fadnis et al., 2025) to help perform due-diligence in model selection for domain specific use. Our user study shows that the provided features, such as highlighting and editing responses, improve the quality of the conversations created in the platform. We release the RAGAPHENE code with Apache 2.0 license on GitHub at <https://github.com/ibm/RAGaphene>.

7 Ethical Considerations

We take care to ensure the use of our platform is private by providing a stateless approach. Our platform only requires an email login, but does not store any personal information or retain created conversations. The interface is built primarily with

Carbon Design System components, which provide baseline keyboard and screen-reader support for standard controls such as buttons, modals, and form inputs. We have not run a dedicated accessibility audit, and some custom elements fall outside this baseline. We plan to address these gaps in future work.

During our user study we asked for some personal details to understand our annotators' background. All such details were only looked at in aggregate and not attributed to the annotator individually.

References

- Anthropic. 2024. [Introducing the next generation of Claude](#).
- BasicAI. 2019. LLM & GenAI annotation. <https://www.basic.ai/basicai-cloud-data-annotation-platform/large-language-model-and-generative-ai-data-annotation-toolset>.
- Nouha Dziri, Ehsan Kamalloo, Sivan Milton, Omar Zaiane, Mo Yu, Edoardo M. Ponti, and Siva Reddy. 2022. [FaithDial: A faithful benchmark for information-seeking dialogue](#). *Transactions of the Association for Computational Linguistics*, 10:1473–1490.
- Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. [RAGAs: Automated evaluation of retrieval augmented generation](#). In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, St. Julians, Malta. Association for Computational Linguistics.
- Kshitij P Fadnis, Siva Sankalp Patel, Odellia Boni, Yanis Katsis, Sara Rosenthal, Benjamin Sznajder, and Marina Danilevsky. 2025. [InspectorRAGet: An introspection platform for RAG evaluation](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 125–134, Albuquerque, New Mexico. Association for Computational Linguistics.
- Song Feng, Siva Sankalp Patel, Hui Wan, and Sachindra Joshi. 2021. [MultiDoc2Dial: Modeling dialogues grounded in multiple documents](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6162–6176, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yearly, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kam-badur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vitor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenxin Fu, Whit-

- ney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit San-gani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchan-dani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yaz-dan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Han-cock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Fe-ichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkan Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Este-ban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippas Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanaz-eri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jen-nifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan Mc-Phie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khan-delwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Ki-ran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrst-edt, Madian Khabza, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Pa-tel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pe-dro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lind-say, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wen-wen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Maeda F Hanafi, Kshitij Fadnis, Marina Danilevsky, Sara Rosenthal, and Yann Katsis. 2025. Creating conversational datasets for retrieval-augmented generation applications is hard: Challenges and research opportunities. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*.
- HumanSignal. 2020. Label Studio – Dialog analysis. https://labelstud.io/templates/dialogue_analysis.
- Infiniflow. 2024. Ragflow. <https://github.com/infiniflow/ragflow>.
- Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gi-anna Lengyel, Guillaume Bour, Guillaume Lam-ple, L lio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao,

- Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.
- Hideaki Joko, Faegheh Hasibi, Krisztian Balog, and Arjen P. de Vries. 2021. [Conversational entity linking: Problem definition and datasets](#). In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '21*, page 2390–2397, New York, NY, USA. Association for Computing Machinery.
- Yannis Katsis, Sara Rosenthal, Kshitij Fadnis, Chulaka Gunasekara, Young-Suk Lee, Lucian Popa, Vraj Shah, Huaiyu Zhu, Danish Contractor, and Marina Danilevsky. 2025. [MTRAG: A multi-turn conversational benchmark for evaluating retrieval-augmented generation systems](#). *Preprint*, arXiv:2501.03468.
- Tzu-Lin Kuo, Feng-Ting Liao, Mu-Wei Hsieh, Fu-Chieh Chang, Po-Chun Hsu, and Da-Shan Shiu. 2024. [RAD-Bench: Evaluating large language models capabilities in retrieval augmented dialogues](#). *Preprint*, arXiv:2409.12558.
- LabelBox. 2024. Chat applications. <https://labelbox.com/solutions/conversational-ai/>.
- Stefano Menini, Daniel Russo, Alessio Palmero Aprosio, and Marco Guerini. 2025. [First-AID: the first annotation interface for grounded dialogues](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 563–571, Vienna, Austria. Association for Computational Linguistics.
- OpenAI. 2024. ChatGPT: Conversational AI model. <https://openai.com/chatgpt>. Accessed: 2025-03-07.
- OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O’Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. [Gpt-4 technical report](#). *Preprint*, arXiv:2303.08774.

Sara Rosenthal, Yannis Katsis, Vraj Shah, Lihong He, Lucian Popa, and Marina Danilevsky. 2026. [MTRAG-UN: A benchmark for open challenges in multi-turn RAG conversations](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 10363–10369, San Diego, California, United States. Association for Computational Linguistics.

Sanat Sharma, David Seunghyun Yoon, Franck Dernoncourt, Dewang Sultania, Karishma Bagga, Mengjiao Zhang, Trung Bui, and Varun Kotte. 2024. [Retrieval augmented generation for domain-specific question answering](#). *Preprint*, arXiv:2404.14760.

A Tooling

RAGAPHENE is a React web application built with NextJS 16 framework³ and requires Python ≥ 3.10 with minimal dependencies to power the experiment flow. We use the Carbon Design System⁴ for the user interface. RAGAPHENE has built-in connectivity to Elasticsearch, MongoDB Atlas, IBM Cloudant retrieval engines and WatsonX.AI and OpenAI generator engines via corresponding Node SDKs, Claude from Anthropic, Gemini from Google and Ollama to follow soon. Furthermore, RAGAPHENE can be extended to support any RESTful retrieval and generation engines with limited data transformations. Our platform is lightweight; it can easily be run on virtual machines or even personal laptops/desktops with 2 CPUs and 8GB RAM. The bulk of the resource consumption is observed only during an experiment run.

To enable privacy, RAGAPHENE is a stateless application and does not retain any created, reviewed conversations or uploaded datasets. A stateless architecture also lowers the barrier to adopting RAGAPHENE: there is no database, user-management layer, or retention policy to stand up before a team can start using the platform, on top of the privacy benefit of not retaining proprietary or client documents on the server. The trade-off is that RAGAPHENE keeps no shared state for coordinating a batch across annotators; tracking review status and routing work is left to tooling the deploying team already uses, such as shared drives or version control over the exported JSON files. This is by design: RAGAPHENE is not a crowd-annotation platform, and coordinating many annotators at scale is a problem already well served by tools such as Amazon Mechanical Turk or Appen.

³<https://react.dev>, <https://nextjs.org>

⁴<https://carbondesignsystem.com>