Revista Eletrônica de Iniciação Científica em Computação, 2026, 24:1
ISSN: 3085-8461 • doi: 10.5753/reic.2026.6811

Research Paper

Exploring the Feasibility of Retrieval-Augmented Generation for Software Defect Analysis: An Empirical Study

Pedro Lucas de Lavor Farias [Universidade Estadual do Ceará | *pedrinho.lavor@aluno.uece.br*]
Levi Almeida da Silva [Universidade Estadual do Ceará|*levi.silva@aluno.uece.br*]
Pedro Almir M. Oliveira [Instituto Federal do Maranhão |*pedro.oliveira@ifma.edu.br*]
Ismayle de Sousa Santos [Universidade Estadual do Ceará|*ismayle.santos@uece.br*]

Bacharelado em Ciência da Computação, Universidade Estadual do Ceará, Silas Munguba St., 1700, Itaperi, Fortaleza, CE, 60714-903, Brazil.

Abstract. A defect database is a structured collection of information about software bugs, errors, and their resolutions throughout the development lifecycle. These databases have valuable information for improving software quality, but their growing volume and complexity make manual analysis inefficient and error-prone. This paper presents an empirical study using real-world data investigating the feasibility of Retrieval-Augmented Generation (RAG) to support interactive exploration of software defect repositories. The study is structured with reference to selected phases of the CRISP-DM framework, which was used as an organizational guideline. Using a real-world software defect database as the empirical basis, the approach automates insight extraction from defect repositories, aiming to improve decision-making in quality assurance and software development by enabling scalable and intelligent exploration of software failures.

Keywords: Data Mining, Software Defect Repositories, Large Language Models, Software Testing.

Received: 04 November 2025 • **Accepted:** 03 June 2026 • **Published:** 07 August 2026

1 Introduction

Software testing consists of a systematic process to verify that a program operates according to specifications and doesn't exhibit unintended behaviors, as defined by Badgett *et al*. [2012]. The practice also encompasses the identification of bugs, errors, or missing requirements in the developed system or software, according to Jamil *et al*. [2016].

The testing phase crucially involves generating and logging bug reports. As defined by Al-Batlaa *et al*. [2019], a bug report is "a software document that describes software bugs, which is submitted by a developer, a tester, or an end-user". These reports document the failures encountered during testing and provide information about identified issues, allowing development teams to understand software deficiencies and apply fixes. Moreover, the history of recorded defects represents a valuable source of knowledge for system evolution, enabling retrospective analyses of the most problematic areas and supporting strategic decisions for future software versions.

The application of data mining techniques in software testing has gained significant attention due to its potential to automate the induction of functional requirements from execution data, recover incomplete specifications, and design efficient regression tests. Last *et al*. [2003] highlight that by leveraging algorithms, data mining enables the creation of models capable of distinguishing between correct and faulty software versions, thus reducing reliance on manual testing and improving overall testing efficiency.

To efficiently manage software development and maintenance, project management systems such as Jira and similar tools are widely used, as highlighted by Milojević *et al*. [2023]. These systems facilitate structured storage of information, including detailed descriptions of functionalities, a change history, and defect reports. Based on these records, it is possible to build databases composed exclusively of reported bugs, which can be explored to identify patterns and trends in the development process. However, manually analyzing defect databases can be time-consuming and prone to inconsistencies, as it relies on human effort to categorize, filter, and interpret large volumes of defect reports.

In this context, the application of knowledge extraction techniques is of great importance. The relevance of using these techniques lies in their ability to enable non-trivial discoveries and the creation of hypotheses that, before the statistical approach, could not be made. Knowledge reduction therefore becomes a tool to support decision-making, as noted by Sousa *et al*. [2022]. It enables the identification of patterns, trends, and hidden relationships within large datasets, supporting informed decision-making across various domains. In this context, structured knowledge extraction processes play an important role in supporting systematic analysis. Among them, the Cross Industry Standard Process for Data Mining (CRISP-DM) is widely recognized as a cyclical framework that organizes data mining activities into six phases, guiding projects from business understanding to deployment PÁDUA and SOUSA [2018]. In this study, CRISP-DM is used only as a high-level organizational reference to structure data-related activities, rather than as a fully implemented methodological contribution.

Additionally, the recent popularization of Large Language Models (LLMs) and Artificial Intelligence (AI) paradigms such as Retrieval-Augmented Generation (RAG) has transformed the landscape of data analysis and knowledge extraction. LLMs, with their ability to understand and generate natural language, enable a more intuitive and interactive exploration of complex datasets. In this context, Retrieval-Augmented Generation (RAG) has emerged as one of the most

popular paradigms enabling LLMs to access external data. It serves as a grounding mechanism to mitigate the phenomenon of hallucinations, as pointed out by Lewis *et al*. [2020].

In this context, this article presents an empirical study investigating the viability of Retrieval-Augmented Generation (RAG) to support interactive exploration of software defect repositories. Rather than proposing a new data mining methodology, the study evaluates how a RAG-based solution can be implemented and assessed in a real-world industrial setting.

To guide this investigation, and aiming to assess the approach, we formulated the following research question (**RQ**): Is Retrieval-Augmented Generation a viable approach to support interactive exploration of software defect data in a real-world industrial dataset? Exploring this question enables a deeper understanding of how AI techniques, such as LLMs, can enhance defect mining processes by providing context-aware insights. It also highlights the potential of integrating LLMs with well-structured knowledge extraction frameworks.

The remainder of this paper is organized as follows: Section 3 discusses the related work; Section 4 presents the RAG-based solution structured according to selected CRISP-DM phases; Section 5 describes the results from the empirical study; Section 6 outlines the limitations and threats to validity; and finally, Section 7 concludes the paper and discusses future work.

2 Background

The Cross-Industry Standard Process for Data Mining (CRISP-DM) is a widely used methodology that provides a systematic and iterative framework for structuring data mining projects. In this study, selected CRISP-DM phases were used as an organizational reference to guide the data-related activities supporting the RAG-based solution. CRISP-DM was considered suitable due to its iterative structure, which aligns with the exploratory nature of configuring retrieval strategies and refining prompt interactions. However, the framework was not fully implemented in its entirety, as the primary objective of this work is to assess the viability of RAG for defect data exploration rather than to execute a complete data mining lifecycle. While other knowledge extraction techniques, such as KDD (Knowledge Discovery in Databases), a structured process for discovering meaningful patterns and insights from large datasets, focus primarily on sequential extraction of patterns from data, CRISP-DM emphasizes a cyclical approach. This iterative structure is particularly advantageous in our context, where insights derived from RAG often require frequent refinement of both the retrieval process and query strategies.

The CRISP-DM process consists of six key phases: **(1) Business Understanding**, which aligns project objectives with strategic goals; **(2) Data Understanding**, involving data collection, exploration, and quality assurance; **(3) Data Preparation**, where data is cleaned, transformed, and structured for usability; **(4) Modeling**, which applies techniques to extract meaningful patterns (In this study, the Modeling phase served as a conceptual reference for implementing the Retrieval-Augmented Generation pipeline, rather than applying traditional statistical or machine learning techniques); **(5) Evaluation**, assessing model performance and reliability; and **(6) Deployment**, integrating the solution into real-world applications for decision-making. CRISP-DM is highly adaptable and iterative, allowing continuous refinement as new data becomes available or project requirements evolve.

Retrieval-Augmented Generation (RAG) enhances the capabilities of Large Language Models (LLMs) by incorporating an external knowledge retrieval mechanism. Unlike traditional models that rely solely on pre-trained knowledge, RAG dynamically retrieves relevant documents or data chunks before generating responses. This two-stage process involves **Retrieval**, where the system searches a structured database or repository for contextually accurate information, and **Generation**, where the LLM combines its pre-trained knowledge with the retrieved data to produce precise and context-aware responses. This approach improves accuracy, reduces hallucinations, and ensures that responses are grounded in up-to-date information.

3 Related Works

In the literature, there are other studies on defect analysis or related information aimed at supporting software development. The recent related work discussed in this section was selected based on whether they utilize tools and methodologies analogous to those presented in this article.

Sousa *et al*. [2022] used the Knowledge Discovery in Databases (KDD) methodology to mine defects, in order to identify problematic areas of the system based on the insights obtained. The mining process was conducted using values such as Epics and system versions that exhibited a higher number of bugs. In contrast, our work focuses on evaluating a RAG-based approach for interactive exploration of defect repositories. While CRISP-DM was used as a partial organizational reference to structure data-related activities, the main contribution of this study lies in assessing the feasibility of integrating LLM-based retrieval mechanisms into real-world defect analysis workflows.

Getachew *et al*. [2024] used CRISP-DM to develop a defect detection model for steel rebar. The study specifically focused on overcoming the challenges associated with the time and cost of manual physical inspection in the steel industry. While their work applies the full lifecycle of CRISP-DM to a predictive modeling problem, our study uses selected CRISP-DM phases only as structural guidance. Our focus is not on predictive modeling, but on evaluating how Retrieval-Augmented Generation can support insight retrieval and interactive analysis of software defect repositories.

Edström [2024] aimed to use RAG for a data extraction pipeline specified for second-life batteries. By predefining the prompts, the user may only provide the documents that are wished to be analyzed; this is to ensure that the answers are in the correct format for further data processing. In comparison, our work applies RAG within a process structured according to selected CRISP-DM phases, specifically tailored to support interactive defect analysis in software systems.

Compared to these related works, this study contributes by empirically evaluating the viability of Retrieval-Augmented Generation for interactive exploration of industrial software defect repositories. While CRISP-DM was used as a partial structuring reference, the primary contribution lies

in demonstrating how LLM-based retrieval can automate and accelerate the extraction of meaningful insights from defect data.

4 METHODOLOGY

For this research, we used a defect database from the X software[1], obtained through JIRA. The reports that generated the defect database were created by the developers and QAs who worked on the project. The data were collected using JIRA's search tool with the query: "project = BR AND issuetype in (Bug, 'Dev Bug') ORDER BY created DESC".

The query used on JIRA's search tool retrieved all incidents associated with the project, ensuring comprehensive coverage of reported defects. The extracted data was then exported in CSV (Comma-Separated Values) format, resulting in a dataset containing 1,047 defect reports recorded between 2018 and 2021.

To preserve confidentiality and prevent disclosure of sensitive industrial information, a fictitious defect dataset was constructed based on the structural and semantic characteristics of the original industrial dataset. The fictitious version preserves the statistical distribution and structure necessary for evaluation, while anonymizing identifiers and modifying potentially sensitive content.

The analyzed system, referred to in this study as HospitalCare, represents a hospital infrastructure management and clinical monitoring platform. The domain of the system involves critical infrastructure management and medical telemetry.

The dataset contains 1,047 defect records stored in CSV format, organized into eleven primary attributes: Issue Type, Key, Summary, Status, Created, Linked Issues, Development metadata, Epic Link, Reporter, Epic Name, and Sprint. The Summary field varies between approximately 30 and 200 characters and typically follows a structured technical pattern combining system identifier and concise error description.

Representative defect examples include failures in dropdown rendering, unknown error messages during group creation, and status problems with WardTerminal. The dataset provides sufficient semantic variability and structured metadata to support evaluation of retrieval performance.

To extract valuable insights from the raw dataset, we structured our activities according to selected phases of the CRISP-DM framework. Its iterative perspective supported incremental refinements in data preparation and retrieval configuration.

The cleaned dataset was then integrated into the RAG pipeline. This approach was chosen because RAG can combine the data retrieved with the inference power of the LLM. In that sense, the system not only recovers defect records, but also infers patterns and potential risk areas in the software.

4.1 Process Overview Based on Selected CRISP-DM Phases

This section presents the integrated process in which CRISP-DM is adopted as a structured methodological reference to organize and guide the development of the RAG pipeline. Figure 1 illustrates the proposed RAG pipeline structured according to the CRISP-DM methodology. The diagram makes explicit which CRISP-DM phases were partially implemented in this study (highlighted in yellow) and which components are specifically related to the RAG pipeline (highlighted in red). The yellow elements represent the adapted and partially executed CRISP-DM phases that provide the structural foundation of the research, while the red elements denote the stages exclusively associated with the design and operation of the RAG-based modeling approach.

1. **Business Understanding:** As shown in Figure 1, the first phase of our process involved defining the target audience as project developers and testers. The objective of the project is to ensure an easy and quick way to associate software bugs with possible problem areas.
2. **Data Understanding:** Understanding the data types, structures, and meanings is a crucial step in the CRISP-DM methodology (Figure 1: Phase 2). In this case, we are dealing with a CSV file containing a software defect database from System X. The CSV columns represent categories of data associated with each report, such as Issue Type, Linked Issues, Reporter, etc.
3. **Data Preparation:** As depicted in Figure 1: Phase 3, the data preparation phase involved cleaning the dataset by removing duplicates, filling missing values, and assigning specific IDs to each CSV row. These steps were necessary because tokenization and ID assignment are essential for populating the vector database. The vector database stores document embeddings in which each vector is mapped to the textual bug description from its corresponding CSV row. All other columns are converted into structured metadata associated with their respective embeddings.
4. **Modeling:** In this phase, illustrated in Figure 1: Phase 4, the query is first processed by the embedding model, which converts it into a vector representation. This vector is then used to perform a semantic search in the vector database, retrieving the most relevant stored embeddings. The retrieved context is then provided to the LLM, which generates the final response based on both the query and the retrieved information. The LLM selected was Mistral 7B, as it represented the best open-source option in terms of cost-benefit available through Ollama. Models requiring more than 4 GB of RAM were difficult to test due to the computational limitations of the available hardware. Similarly, the embedding model BAAI-large HF, obtained cost-free via Hugging Face, was chosen for comparable reasons, offering a practical balance between performance and resource efficiency within these constraints.
5. **Evaluation:** To assess the quality and accuracy of responses, we followed the evaluation process as outlined in Figure 1: Phase 5, which is part of the CRISP-DM methodology. We used evaluation metrics from `sklearn.metrics`:
   (a) **Mean Reciprocal Rank (MRR):** Measures how early the first relevant document appears in the ranking. It is computed as the reciprocal of the rank position of the first relevant document. Higher val-

[1]The actual name and characteristics of the software were omitted due to confidentiality agreements.

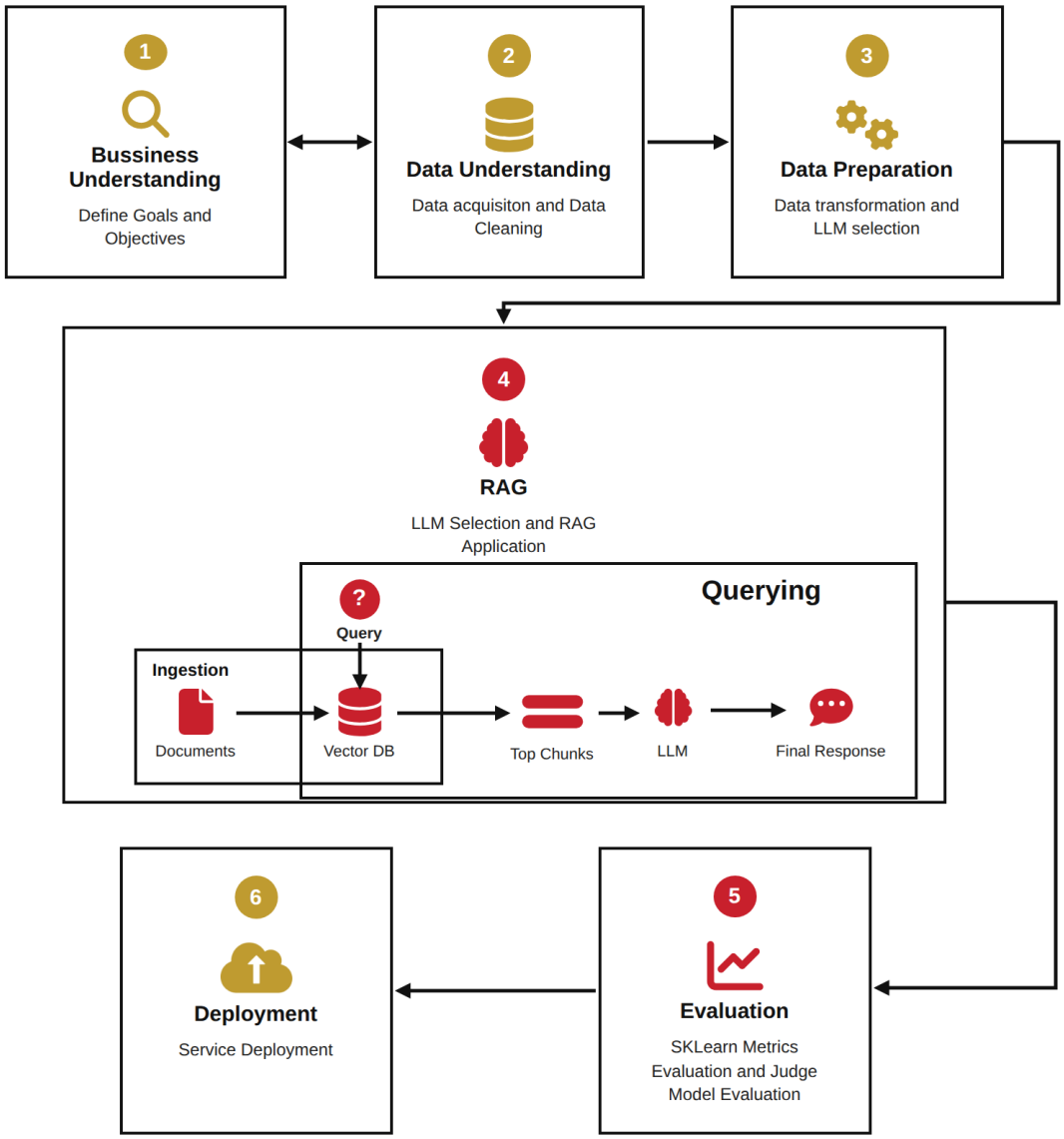

Figure 1. Adaptation of the CRISP-DM Process Model for RAG-Based Data Mining Workflow

ues indicate that relevant information is retrieved at top positions.

(b) **Normalized Discounted Cumulative Gain (NDCG):** Evaluates the overall ranking quality by considering both the position and graded relevance of retrieved documents. It assigns higher importance to relevant documents appearing earlier in the ranking and normalizes the score to the interval [0,1].

(c) **Hit@1:** Verifies whether the top-ranked document is relevant. This metric directly reflects the system's ability to place the correct document in the first position.

(d) **Average Precision (AP):** Summarizes the precision values across all ranking positions where relevant documents are retrieved. It provides a single score that reflects both ranking order and retrieval consistency.

(e) **Precision@K:** Measures the proportion of relevant documents among the top-K retrieved results. It focuses on the immediate usefulness of the highest-ranked documents presented to the user.

6. **Deployment:** At this stage, the model is prepared for deployment, enabling its adaptation to different knowledge bases. The system is designed to be flexible, allowing organizations to customize the retrieval and inference process for any domain, extracting valuable insights from their data sources.

5 RESULTS AND DISCUSSION

This section presents the evaluation results of the proposed RAG-based solution applied to a fictitious defect database. The assessment combined objective retrieval metrics with structured human validation to provide a comprehensive analysis of system performance.

5.1 Evaluation Protocol

Four analysts were granted full access to the fictitious defect dataset and instructed to explore its contents. Based exclusively on the available data, each analyst formulated three domain-related questions reflecting realistic defect investiga-

tion scenarios, including factual retrieval, metadata inspection, linked-issue analysis, and inferential reasoning.

All questions were submitted to the RAG pipeline. For each query, the system retrieved relevant documents and generated a response using the LLM. To reduce bias, a cross-evaluation protocol was adopted: the pair (question + LLM-generated answer) from Analyst 1 was evaluated by Analyst 2; Analyst 2's output was evaluated by Analyst 3; and so forth, forming a sequential validation cycle among the four analysts.

Each evaluator revisited the dataset and assessed the LLM response using a 1–5 Likert scale considering: (i) factual correctness, (ii) grounding in the dataset, (iii) coherence, and (iv) completeness.

In parallel, objective ranking-based metrics were computed to assess the retrieval component:

- **Mean Reciprocal Rank (MRR)** – evaluates how early the first relevant document appears in the ranking.
- **Normalized Discounted Cumulative Gain (NDCG)** – measures ranking quality considering position and graded relevance.
- **Hit@1** – verifies whether the top-ranked document is relevant.
- **Average Precision (AP)** – summarizes ranking precision across relevant positions.
- **Precision@K** – proportion of relevant documents within the top-K results.

This evaluation corresponds to Phase 5 (Evaluation) of the CRISP-DM process adopted in this study. The resulting multi-layer evaluation framework enables assessment of retrieval performance, semantic adequacy, and practical applicability.

5.2 Quantitative Results

Table 1 presents the retrieval performance per query and analyst.

The analysis of the results demonstrates that the effectiveness of the RAG pipeline is directly linked to the nature of the query. The system achieved superior performance in structured queries involving explicit metadata (e.g., HMS-E20Q3R and HMS-EMEVCK), reaching maximum scores in MRR and Hit@1. These cases were validated with high coherence (Likert 4 and 5) by the analysts. This confirms that specific terms in the Summary field facilitate vectorization and precise retrieval at the first position.

Even when the initial ranking was not perfect, as in the case of HMS-NTW1Z7 (MRR: 0.5), retrieval within the Top 5 provided sufficient context for the LLM to generate satisfactory responses, receiving a coherence score of 4. However, the study revealed limitations in inferential queries or complex relations (e.g., "linked issues"), where retrieval metrics were null and coherence was low (Likert 1). In these scenarios, embedding models tend to prioritize generic keywords over deep semantic connections.

Finally, the "synthesis power" of the LLM (Mistral 7B) was noted in mitigating retrieval failures. In complex queries such as HMS-AJEG4X, despite the reduced MRR (0.0588), the model's inference capability ensured acceptable coherence (Likert 3). The data supports that RAG is a viable approach for exploring defect repositories, especially via metadata, although it requires future improvements such as Re-ranking to strengthen robustness in relational queries.

The results that presented null values (0.0) in metrics such as MRR, NDCG, and Hit@1 reveal the critical boundaries of the semantic retrieval layer in the context of industrial data. Cases such as issue HMS-ZTDJ6G, which sought to identify "linked issues," and HMS-MW4SDM, focused on sprint associations, completely failed in the initial ranking process. This lack of retrieval occurs because the embedding model (BAAI-large HF) prioritizes the lexical similarity of generic terms present in the Summary over the structural relationships contained in complex metadata. Consequently, the analyst assigned a score of 1 on the Likert scale for these interactions, highlighting that, without a strong retrieval signal, the LLM lacks an external knowledge base to ground a coherent response.

5.3 Qualitative Cross-Analyst Assessment

The Likert-based cross-evaluation revealed high perceived correctness in responses to structured queries, especially those involving metadata filtering or direct attribute retrieval. For inferential or causality-driven questions, evaluations remained satisfactory but displayed greater dispersion, indicating that semantic inference introduces additional uncertainty. Importantly, the cross-review protocol reduced self-assessment bias and ensured that each response was independently validated by a different analyst familiar with the dataset.

Taken together, the quantitative and qualitative findings suggest that the proposed RAG architecture effectively integrates structured retrieval and generative synthesis, supporting interactive exploration of defect repositories. While the previous analysis focused on controlled experimental validation, an additional practitioner-oriented evaluation was conducted to assess real-world applicability.

In addition to the direct coherence evaluations, the research form collected subjective perceptions regarding the model's performance and system efficiency. Participants highlighted that, although the LLM faced specific difficulties in identifying data that merged date and time within the CSV, performance in other tasks was considered satisfactory and fulfilled the objectives of interactive exploration. On a scale of 1 to 5, the analysts' willingness to adopt this technology to facilitate future analyses was significant, with 75% of responses concentrated at levels 4 and 5, reinforcing the perception of the tool's practical utility in the quality assurance workflow.

Regarding response time, perceptions varied significantly, with reports ranging from 3 seconds to 100 seconds per question.

This temporal variation not only indicate the processing latency of the RAG system but but also stems from multiple factors, such as the professional's level of technical expertise, the inherent complexity of inferential questions, and eventual structural inconsistencies in the original CSV, which may have hindered the immediate location of data points during the human validation phase.

Query	RAG Metrics	Analyst	Coherence
What is the summary of issue HMS-NTW1Z7?	MRR: 0.5 NDCG: 0.6309 Hit@1: 0.0 AP: 0.5 Precision@K: 0.05	1	4
Which issue is associated (linked issues) with issue HMS-ZTDJ6G?	MRR: 0.0 NDCG: 0.0 Hit@1: 0.0 AP: 0.0 Precision@K: 0.0	1	1
Which issue is epically linked to issue HMS-SM2XAR?	MRR: 0.0 NDCG: 0.0 Hit@1: 0.0 AP: 0.0 Precision@K: 0.0	1	1
Question 1 - What is the Summary of issue HMS-I3BFBD?	MRR: 0.125 NDCG: 0.3155 Hit@1: 0.0 AP: 0.125 Precision@K: 0.05	2	5
Question 2 - Which sprint is associated with key HMS-MW4SDM?	MRR: 0.0 NDCG: 0.0 Hit@1: 0.0 AP: 0.0 Precision@K: 0.0	2	4
Question 3 - Who is the Reporter of Key HMS-E20Q3R?	MRR: 1.0 NDCG: 1.0 Hit@1: 1.0 AP: 1.0 Precision@K: 0.05	2	4
What is the summary of issue HMS-45GA1R?	MRR: 0.1667 NDCG: 0.3562 Hit@1: 0.0 AP: 0.1667 Precision@K: 0.05	3	5
Who is the reporter of issue HMS-EMEVCK?	MRR: 1.0 NDCG: 1.0 Hit@1: 1.0 AP: 1.0 Precision@K: 0.05	3	5
What is the creation time of issue HMS-GUSWO8?	MRR: 0.3333 NDCG: 0.5 Hit@1: 0.0 AP: 0.3333 Precision@K: 0.05	3	1
Why does issue HMS-AJEG4X signal two incidents and why is it invalid?	MRR: 0.0588 NDCG: 0.2398 Hit@1: 0.0 AP: 0.0588 Precision@K: 0.05	4	3
Why is issue HMS-L6VPKY blocked?	MRR: 0.0 NDCG: 0.0 Hit@1: 0.0 AP: 0.0 Precision@K: 0.0	4	4
What should be rediscussed in issue HMS-ZS5BNF?	MRR: 1.0 NDCG: 1.0 Hit@1: 1.0 AP: 1.0 Precision@K: 0.05	4	5

Table 1. RAG Retrieval Performance and Analyst Likert Scale Evaluation

6 LIMITATIONS AND THREATS TO VALIDITY

Although the proposed approach demonstrated promising results, several limitations must be acknowledged.

First, the study relied on open-source models for both embeddings and response generation. While this choice promotes transparency, reproducibility, and experimental control, such models may present constraints when compared to larger proprietary state-of-the-art alternatives, particularly in terms of semantic depth and contextual representation. Consequently, the observed performance should be interpreted within the capacity bounds of the selected model architecture.

Second, the experimental environment was subject to computational constraints that limited the use of larger-scale models. The available hardware restricted the execution of models requiring substantial memory resources, which influenced the selection of efficient open-source alternatives. Although the available hardware was adequate for this study, different configurations could potentially yield improved performance outcomes.

Third, the empirical validation was conducted using a single defect dataset (System X), covering records from 2018 to 2021. While the dataset reflects a real industrial context, the results may not generalize to repositories with substantially different structures, volumes, or documentation standards. Further replication across heterogeneous defect databases would be necessary to strengthen external validity.

Overall, these limitations do not invalidate the findings but delineate the scope within which the results should be interpreted.

7 FINAL REMARKS

This study aimed to address the research question: **"Is Retrieval-Augmented Generation a viable approach to support interactive exploration of software defect data in a real-world industrial dataset?"**

The findings indicate that Retrieval-Augmented Generation (RAG) constitutes a viable approach for this purpose. By combining structured retrieval mechanisms with generative reasoning, the proposed architecture enables interactive and context-aware exploration of defect records. The evaluation demonstrated that the retrieval layer performs consistently well for structured and metadata-driven queries, while the generative component supports the interpretation of semantically complex or relational questions.

The integration of objective ranking metrics with cross-analyst qualitative assessment provided complementary evidence of both technical effectiveness and perceived practical usefulness. Although performance varied depending on query complexity, the overall results suggest that RAG-based systems can assist practitioners in navigating and interpreting defect repositories more efficiently than manual search processes alone.

Rather than replacing expert judgment, the approach functions as a decision-support mechanism that enhances exploratory analysis and facilitates knowledge extraction from historical defect data. The empirical validation on an industrial dataset reinforces the practical relevance of the proposed method, while the identified limitations define the scope within which the results should be interpreted.

For future research, the methodology may be extended to heterogeneous defect repositories and larger-scale datasets to assess scalability and generalizability. Additional user-centered evaluations involving broader practitioner samples could further validate real-world applicability. Moreover, experimenting with more advanced language models and adaptive retrieval strategies may improve contextual reasoning and robustness, particularly for complex inferential queries. In particular, graph-based approaches such as Graph Embeddings and GraphRAG could be investigated to better capture the structural relationships between defect records, addressing the limited retrieval performance observed for linked-issue queries.

Overall, this work contributes empirical evidence supporting the feasibility of RAG-based architectures as interactive tools for software defect exploration in industrial contexts.

Declarations

Authors' Contributions

The undergraduate student contributed to the conceptualization, methodology, software development, data curation, formal analysis, and was the main contributor and writer of this manuscript. The master's student contributed to the literature review, investigation of available tools, and review and editing of the manuscript. The two advisors contributed to the study design, supervision, textual revision, and provided overall guidance during the development of the paper. All authors read and approved the final manuscript.

Acknowledgements

This work was developed with support from the Ceará Foundation for Support of Scientific and Technological Development - FUNCAP and the National Council for Scientific and Technological Development - CNPq (Process 312173/2025-3).

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

The datasets used in the current study are available on github.

Further relevant information

Generative AI tools were used exclusively to assist in the English language revision of each section of the manuscript. The scientific content, analysis, and conclusions are the sole responsibility of the authors.

References

Al-Batlaa, A., Abdullah-Al-Wadud, M., and Anwar, M. (2019). A method to suggest solutions for software bugs. In *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pages 169–172. IEEE. DOI: 10.1109/ICIEV.2019.8858514.

Badgett, T., Myers, G. J., *et al*. (2012). *The Art of Software Testing*. Wiley-Blackwell.

Edström, J. (2024). Rag-based data extraction: Mining information from second-life battery documents. Available at: `https://uu.diva-portal.org/smash/record.jsf?pid=diva2:1877456`.

Getachew, M., Beshah, B., Mulugeta, A., and Kitaw, D. (2024). Application of artificial intelligence to enhance manufacturing quality and zero-defect using crisp-dm framework. *International Journal of Production Research*, pages 1–25. DOI: 10.1080/00207543.2024.2407919.

Jamil, M. A., Arif, M., Abubakar, N. S. A., and Ahmad, A. (2016). Software testing techniques: A literature review. In *2016 6th international conference on information and communication technology for the Muslim world (ICT4M)*, pages 177–182. IEEE. DOI: 10.1109/ICT4M.2016.045.

Last, M., Friedman, M., and Kandel, A. (2003). The data mining approach to automated software testing. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 388–396. DOI: 10.1145/956750.956795.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., *et al*. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474. Available at: `https://papers.nips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html`.

Milojević, D., Macuzic, I., Djordjevic, A., Savković, M., and Djapan, M. (2023). Comparative analysis of software tools for agile project management. Available at: `https://scidar.kg.ac.rs/handle/123456789/18405`.

PÁDUA, A. and SOUSA, F. (2018). Metodologia crisp-dm: potencialidades na descoberta do conhecimento em dados educacionais. In *XVI Congresso Internacional de Tecnologia na Educação. Educação e Tecnologia em Tempos de Mudança*. Available at: `http://www.pe.senac.br/congresso/anais/2018/pdf/poster/METODOLOGIA%20CRISP-DM%20POTENCIALIDADES%20NA%20DESCOBERTA%20DO%20CONHECIMENTO%20EM%20DADOS%20EDUCACIONAIS.pdf`.

Sousa, É. M., Rodrigues, A., Teixeira, N., Santos, I. S., Francisco, M. S., Andrade, R. M. d. C., and Vasconcelos, D. R. (2022). From past to future: An experience using data mining to guide tests. In *Proceedings of the XXI Brazilian Symposium on Software Quality*, pages 1–10. DOI: 10.1145/3571473.3571493.