The evaluation team independently assessed the outputs from Stage I and Stage II to measure the online accuracy of each stage. Due to the high cost of manual evaluation, assessments were conducted at fixed intervals only during the early rapid iteration phase of the project. The evaluation results from the 10 weeks' iteration period between August 2024 and October 2024 are shown in 7. As shown in the figure, LogSage achieves over **85%** RCA accuracy and over **80%** end-to-end accuracy in real-world deployment scenarios, clearly demonstrating its usability in production environments.

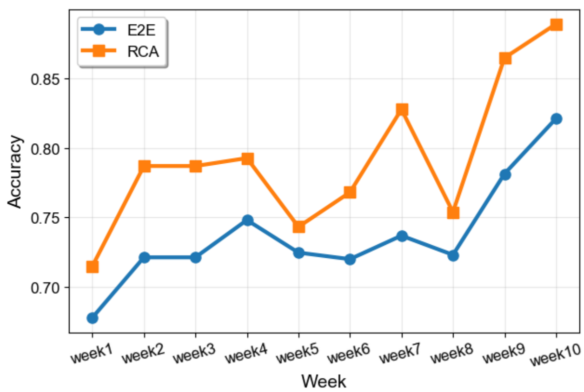

Fig. 7: Human-Annotated Online Accuracy

Auto-Repair Effectiveness. To assess the impact of auto-remediation, we collected engineering metrics on:

- Tool Coverage Rate: Ratio of cases where LogSage successfully recommended a repair tool over all cases that reached the solution generation stage.
- Re-run Success Rate: Proportion of re-runs that passed automatically after executing the suggested fix.

These metrics were gathered via system logging. The average tool coverage rate was 22.4%, with a median of 16.7%, reflecting that overall tool support across CI/CD failures remains partial and that there is significant room to expand coverage. Nevertheless, when tools were applicable, LogSage achieved a re-run success rate of 47.4% on average, with a maximum of 60.0%, median of 47.1%, and minimum of 36.8%. These results indicate that nearly half of the supported failures could be successfully resolved without human intervention—significantly reducing developer effort and turnaround time.

Taken together, these findings underscore both the practical feasibility and long-term potential of integrating automated repair into CI/CD workflows. As the first large-scale industrial deployment of its kind, LogSage demonstrates not only immediate operational value but also a promising direction for further improvement.

*4) **User Survey**:* Based on a recent survey of 191 front-line developers, LogSage received an average satisfaction score of 7.97 out of 10, reflecting strong acceptance and perceived utility in production workflows. Developers highlighted its ease of use, actionable suggestions, and seamless integration, while also providing feedback that has informed subsequent improvements.

VI. Conclusion & Future Work

In this paper, we present LogSage, the first end-to-end LLM-based framework for CI/CD failure diagnosis and automated remediation, validated through large-scale industrial deployment. LogSage operates in two complementary phases. In the offline preparation phase, it leverages log template extraction and enterprise knowledge integration to construct reusable references for diagnosis. In the online operational phase, it performs root cause analysis by filtering, expanding, and pruning failed logs, followed by dynamically assembled diagnostic prompts and multi-route knowledge retrieval. Finally, LogSage generates executable solutions through LLM-guided tool selection and automated reruns, enabling accurate, interpretable root cause analysis and effective remediation of complex CI/CD failures.

Empirical evaluations across multiple LLM backends and baselines show that LogSage achieves significant improvements in both performance and efficiency. It outperforms state-of-the-art LLM-based methods in RCA precision while reducing token consumption by over 85% compared to prior approaches. In industrial settings, LogSage demonstrates sustained adoption, processing over 1 million CI/CD failures and maintaining high user coverage with end-to-end precision exceeding 80%.

While encouraging, several aspects warrant discussion. The effectiveness of the *log diff* strategy relies on the repetitive nature of CI/CD pipelines: despite configuration heterogeneity, executions of the same pipeline remain largely stable across runs, allowing recurring outputs to be filtered regardless of system differences. Although our deployment used internal infrastructures such as Feishu documents and vector databases, the design is infrastructure-agnostic and can be instantiated with alternative knowledge bases in other organizations. Our baseline selection focused on LLM-based methods, as traditional CI/CD or AIOps approaches typically require training from scratch or depend on rigid log formats, making them less comparable to LogSage's training-free design. Finally, relying solely on LLMs for diagnosis introduces risks, as domain-specific semantics may not always be fully captured, leaving room for hallucinations or semantic gaps. These considerations highlight both the strengths and boundaries of LogSage, while pointing toward opportunities for broader integration.

In future work, we plan to upgrade LogSage into a more autonomous and adaptive LLM-Agent capable of orchestrating complex remediation workflows through iterative reasoning and proactive decision-making. We also aim to extend its scope beyond reactive failure handling to broader DevOps scenarios, including fault prediction, anomaly prevention, and automated incident response. These directions involve integrating with observability tools, modeling failure trends, and aligning with real-world operational workflows—pushing LogSage toward becoming an intelligent and proactive DevOps collaborator.

References

[1] B. Fitzgerald and K.-J. Stol, "Continuous software engineering: A roadmap and agenda," *Journal of Systems and Software*, vol. 123, pp. 176–189, 2017. doi: 10.1016/j.jss.2015.06.063

[2] Microsoft Inside Track, "DevOps is sending engineering practices up in smoke", Microsoft, Apr. 15, 2024. [Online]. Available: https://www.microsoft.com/insidetrack/blog/devops-is-sending-engineering-practices-up-in-smoke/ [Accessed: May 10, 2025].

[3] Engineering at Meta, "Rapid release at massive scale", Meta Engineering Blog, Aug. 31, 2017. [Online]. Available: https://engineering.fb.com/2017/08/31/web/rapid-release-at-massive-scale/ [Accessed: May 10, 2025].

[4] GitHub Engineering, "Building organization-wide governance and re-use for CI/CD and automation with GitHub Actions", GitHub Blog, Apr. 5, 2023. [Online]. Available: https://github.blog/enterprise-software/devops/building-organization-wide-governance-and-re-use-for-ci-cd-and-automation-with-github-actions/ [Accessed: May 10, 2025].

[5] M. Hilton, T. Tunnell, K. Huang, D. Marinov, and D. Dig, "Usage, costs, and benefits of continuous integration in open-source projects", in *Proc. 31st IEEE/ACM Int. Conf. Automated Software Engineering (ASE)*, Singapore, 2016, pp. 426–437. doi: 10.1145/2970276.2970358.

[6] X. Hou, Y. Zhao, Y. Liu, Z. Yang, K. Wang, L. Li, X. Luo, D. Lo, J. Grundy, and H. Wang, "Large language models for software engineering: A systematic literature review", *arXiv preprint arXiv:2308.10620*, 2024. [Online]. Available: https://arxiv.org/abs/2308.10620

[7] A. Hemmat, M. Sharbaf, S. Kolahdouz-Rahimi, K. Lano, and S. Y. Tehrani, "Research directions for using LLM in software requirement engineering: A systematic review", *Frontiers in Computer Science*, vol. 7, 2025. doi: 10.3389/fcomp.2025.1519437.

[8] H. Li, X. Zhou, and Z. Shen, "Rewriting the code: A simple method for large language model augmented code search", in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Bangkok, Thailand, Aug. 2024, pp. 1371–1389. doi: 10.18653/v1/2024.acl-long.75.

[9] T. Sun, J. Xu, Y. Li, Z. Yan, G. Zhang, L. Xie, L. Geng, Z. Wang, Y. Chen, Q. Lin, W. Duan, and K. Sui, "BitsAI-CR: Automated code review via LLM in practice", in *Proc. 33rd ACM Int. Conf. Found. Softw. Eng. (Ind. Track)*, 2025. *arXiv preprint arXiv:2501.15134*, 2024. [Online]. Available:https://arxiv.org/abs/2501.15134.

[10] N. Alshahwan, J. Chheda, A. Finogenova, B. Gokkaya, M. Harman, I. Harper, A. Marginean, S. Sengupta, and E. Wang, "Automated unit test improvement using large language models at Meta," in *Companion Proc. of the 32nd ACM Int. Conf. on the Foundations of Software Engineering (FSE 2024)*, Porto de Galinhas, Brazil, 2024, pp. 185–196. doi: 10.1145/3663529.3663839

[11] V.-H. Le and H. Zhang, "Log-based anomaly detection with deep learning: how far are we?," in *Proc. 44th Int. Conf. Software Engineering (ICSE)*, Pittsburgh, Pennsylvania, pp. 1356–1367, 2022. doi: 10.1145/3510003.3510155

[12] V.-H. Le and H. Zhang, "Log-based anomaly detection without log parsing," in *Proc. 36th IEEE/ACM Int. Conf. Automated Software Engineering (ASE)*, Melbourne, Australia, pp. 492–504, 2022. doi: 10.1109/ASE51524.2021.9678773

[13] M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: anomaly detection and diagnosis from system logs through deep learning," in *Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security (CCS)*, Dallas, Texas, USA, 2017, pp. 1285–1298. doi: 10.1145/3133956.3134015

[14] W. Meng, Y. Liu, Y. Zhu, S. Zhang, D. Pei, Y. Liu, Y. Chen, R. Zhang, S. Tao, P. Sun, and R. Zhou, "LogAnomaly: unsupervised detection of sequential and quantitative anomalies in unstructured logs," in *Proc. 28th Int. Joint Conf. Artificial Intelligence (IJCAI)*, 2019, pp. 4739–4745. Available: https://www.ijcai.org/proceedings/2019/0658.pdf

[15] L. Yang, J. Chen, Z. Wang, W. Wang, J. Jiang, X. Dong, and W. Zhang, "Semi-supervised log-based anomaly detection via probabilistic label estimation," in *Proc. 2021 IEEE/ACM 43rd Int. Conf. Software Engineering (ICSE)*, 2021, pp. 1448–1460. doi: 10.1109/ICSE43902.2021.00130

[16] X. Zhang, Y. Xu, Q. Lin, B. Qiao, H. Zhang, Y. Dang, C. Xie, X. Yang, Q. Cheng, Z. Li, J. Chen, X. He, R. Yao, J.-G. Lou, M. Chintalapati, F. Shen, and D. Zhang, "Robust log-based anomaly detection on unstable log data," in *Proc. 27th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, Tallinn, Estonia, 2019, pp. 807–817. doi: 10.1145/3338906.3338931

[17] F. Hassan, N. Meng, and X. Wang, "UniLoc: Unified fault localization of continuous integration failures," *ACM Transactions on Software Engineering and Methodology*, vol. 32, no. 6, pp. 1–31, Article 136, Sep. 2023. doi: 10.1145/3593799

[18] A. K. Arani, T. H. M. Le, M. Zahedi, and M. A. Babar, "Systematic literature review on application of learning-based approaches in continuous integration," *IEEE Access*, vol. 12, pp. 135419–135450, 2024. doi: ACCESS.2024.3424276

[19] A. S. Mohammed, V. R. Saddi, S. K. Gopal, S. Dhanasekaran, and M. S. Naruka, "AI-driven continuous integration and continuous deployment in software engineering," in *Proc. 2024 2nd Int. Conf. on Disruptive Technologies (ICDT)*, 2024, pp. 531–536. doi: 10.1109/ICDT61202.2024.10489475

[20] S. Lee, S. Hong, J. Yi, T. Kim, C.-J. Kim, and S. Yoo, "Classifying false positive static checker alarms in continuous integration using convolutional neural networks," in *Proc. 2019 12th IEEE Conf. on Software Testing, Validation and Verification (ICST)*, 2019, pp. 391–401. doi: 10.1109/ICST.2019.00048

[21] G. Grano, T. V. Titov, S. Panichella, and H. C. Gall, "How high will it be? Using machine learning models to predict branch coverage in automated testing," in *Proc. 2018 IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE)*, 2018, pp. 19–24. doi: 10.1109/MALTESQUE.2018.8368454

[22] G. Grano, T. V. Titov, S. Panichella, and H. C. Gall, "Branch coverage prediction in automated testing," *J. Softw. Evol. Process*, vol. 31, no. 9, pp. 1–18, Oct. 2019. doi: 10.1002/smr.2158

[23] K. Al-Sabbagh, M. Staron, M. Ochodek, and W. Meding, "Early prediction of test case verdict with bag-of-words vs. word embeddings," in *Proc. CEUR Workshop Proceedings*, Dec. 2019. Available: https://ceur-ws.org/Vol-2568/paper6.pdf

[24] K. W. Al-Sabbagh, M. Staron, R. Hebig, and W. Meding, "Predicting test case verdicts using textual analysis of committed code churns," in *Proc. Int. Workshop on Software Measurement and Int. Conf. on Software Process and Product Measurement (IWSM Mensura)*, Haarlem, The Netherlands, 2019, pp. 138–153. Available: https://ceur-ws.org/Vol-2476/paper7.pdf

[25] R. Abdalkareem, S. Mujahid, and E. Shihab, "A machine learning approach to improve the detection of CI skip commits," *IEEE Transactions on Software Engineering*, vol. 47, no. 12, pp. 2740–2754, 2021. doi: 10.1109/TSE.2020.2967380

[26] Y. Koroglu, A. Sen, D. Kutluay, A. Bayraktar, Y. Tosun, M. Cinar, and H. Kaya, "Defect prediction on a legacy industrial software: a case study on software with few defects," in *Proc. 4th Int. Workshop on Conducting Empirical Studies in Industry (CESI)*, Austin, TX, USA, 2016, pp. 14–20. doi: 10.1145/2896839.2896843

[27] R. Mamata, K. Smith, A. Azim, Y.-K. Chang, Q. Taiseef, R. Liscano, and G. Seferi, "Failure prediction using transfer learning in large-scale continuous integration environments," in *Proc. 32nd Annu. Int. Conf. on Computer Science and Software Engineering (CASCON)*, Toronto, Canada, 2022, pp. 193–198. Available: https://dl.acm.org/doi/10.5555/3566055.3566079

[28] A. Mishra and A. Sharma, "Deep learning based continuous integration and continuous delivery software defect prediction with effective optimization strategy," in *Knowledge-Based Systems*, vol. 296, p. 111835, 2024. [Online]. doi: 10.1016/j.knosys.2024.111835

[29] I. Saidani, A. Ouni, and M. W. Mkaouer, "Improving the prediction of continuous integration build failures using deep learning," in *Automated Software Engineering*, vol. 29, no. 1, pp. 1–61, May 2022. [Online]. doi: 10.1007/s10515-021-00319-5

[30] R. Mahindru, H. Kumar, and S. Bansal, "Log anomaly to resolution: AI based proactive incident remediation," in *Proc. 36th IEEE/ACM Int. Conf. on Automated Software Engineering (ASE)*, 2021, pp. 1353–1357. doi: 10.1109/ASE51524.2021.9678815

[31] Y. Xiao, V.-H. Le, and H. Zhang, "Demonstration-free: Towards more practical log parsing with large language models," in *Proc. 39th IEEE/ACM Int. Conf. on Automated Software Engineering (ASE)*, Sacramento, CA, USA, 2024, pp. 153–165. doi: 10.1145/3691620.3694994

[32] C. Egersdoerfer, D. Zhang, and D. Dai, "Early exploration of using ChatGPT for log-based anomaly detection on parallel file systems logs," in *Proc. 32nd Int. Symp. on High-Performance Parallel and Distributed Computing (HPDC '23)*, Orlando, FL, USA, 2023, pp. 315–316. [Online]. doi: 10.1145/3588195.3595943

[33] Y. Liu, S. Tao, W. Meng, J. Wang, W. Ma, Y. Chen, Y. Zhao, H. Yang, and Y. Jiang, "Interpretable online log analysis using large language

models with prompt strategies," in *Proc. 32nd IEEE/ACM Int. Conf. Program Comprehension (ICPC)*, Lisbon, Portugal, pp. 35–46, 2024. doi: 10.1145/3643916.3644408

[34] Y. Liu, S. Tao, W. Meng, F. Yao, X. Zhao, and H. Yang, "LogPrompt: Prompt engineering towards zero-shot and interpretable log analysis," in *Proc. 2024 IEEE/ACM 46th Int. Conf. on Software Engineering: Companion Proceedings (ICSE-Companion)*, 2024, pp. 364–365. doi: 10.1145/3639478.3643108

[35] J. Qi, S. Huang, Z. Luan, S. Yang, C. Fung, H. Yang, D. Qian, J. Shang, Z. Xiao, and Z. Wu, "LogGPT: exploring ChatGPT for log-based anomaly detection," in *Proc. 2023 IEEE Int. Conf. High Performance Computing & Communications, Data Science & Systems, Smart City & Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, pp. 273–280, 2023. doi: 10.1109/HPCC-DSS-SmartCity-DependSys60770.2023.00045

[36] S. Shan, Y. Huo, Y. Su, Y. Li, D. Li, and Z. Zheng, "Face it yourselves: an LLM-based two-stage strategy to localize configuration errors via logs," in *Proc. 33rd ACM SIGSOFT Int. Symp. Software Testing and Analysis (ISSTA)*, Vienna, Austria, pp. 13–25, 2024. doi: 10.1145/3650212.3652106

[37] C. Almodovar, F. Sabrina, S. Karimi, and S. Azad, "LogFiT: Log anomaly detection using fine-tuned language models," in *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 1715–1723, 2024. doi: 10.1109/TNSM.2024.3358730

[38] H. Ju, "Reliable online log parsing using large language models with retrieval-augmented generation," in *Proc. 2024 IEEE 35th Int. Symp. on Software Reliability Engineering Workshops (ISSREW)*, 2024, pp. 99–102. doi: 10.1109/ISSREW63542.2024.00055

[39] W. Guan, J. Cao, S. Qian, J. Gao, and C. Ouyang, "LogLLM: log-based anomaly detection using large language models," *arXiv preprint arXiv:2411.08561*, 2025. Available: https://arxiv.org/abs/2411.08561

[40] F. Hadadi, Q. Xu, D. Bianculli, and L. Briand, "LLM meets ML: Data-efficient anomaly detection on unseen unstable logs," *arXiv preprint arXiv:2406.07467*, 2025. Available: https://arxiv.org/abs/2406.07467

[41] A. Fariha, V. Gharavian, M. Makrehchi, S. Rahnamayan, S. Alwidian, and A. Azim, "Log anomaly detection by leveraging LLM-based parsing and embedding with attention mechanism," in *Proc. 2024 IEEE Canadian Conf. on Electrical and Computer Engineering (CCECE)*, 2024, pp. 859–863. doi: 10.1109/CCECE59415.2024.10667308

[42] M. He, T. Jia, C. Duan, H. Cai, Y. Li, and G. Huang, "LLMeLog: An approach for anomaly detection based on LLM-enriched log events," in *Proc. 2024 IEEE Int. Symp. on Software Reliability Engineering (ISSRE)*, pp. 132–143, 2024. doi: 10.1109/ISSRE62328.2024.00023

[43] L. Zhang, T. Jia, M. Jia, Y. Wu, A. Liu, Y. Yang, Z. Wu, X. Hu, P. S. Yu, and Y. Li, "A survey of AIOps for failure management in the era of large language models," *arXiv preprint arXiv:2406.11213*, 2024. Available: https://arxiv.org/abs/2406.11213

[44] D. Roy, X. Zhang, R. Bhave, C. Bansal, P. Las-Casas, R. Fonseca, and S. Rajmohan, "Exploring LLM-based agents for root cause analysis," in *Companion Proc. 32nd ACM Int. Conf. Found. Softw. Eng.*, Porto de Galinhas, Brazil, 2024, pp. 208–219. doi: 10.1145/3663529.3663841

[45] Z. Wang, Z. Liu, Y. Zhang, A. Zhong, J. Wang, F. Yin, L. Fan, L. Wu, and Q. Wen, "RCAgent: Cloud root cause analysis by autonomous agents with tool-augmented large language models," in *Proc. 33rd ACM Int. Conf. on Information and Knowledge Management (CIKM)*, Boise, ID, USA, 2024, pp. 4966–4974. doi: 10.1145/3627673.3680016

[46] X. Zhang, S. Ghosh, C. Bansal, R. Wang, M. Ma, Y. Kang, and S. Rajmohan, "Automated root causing of cloud incidents using in-context learning with GPT-4," in *Companion Proc. 32nd ACM Int. Conf. Found. Softw. Eng.*, Porto de Galinhas, Brazil, 2024, pp. 266–277. doi: 10.1145/3663529.3663846

[47] D. Goel, F. Husain, A. Singh, S. Ghosh, A. Parayil, C. Bansal, X. Zhang, and S. Rajmohan, "X-lifecycle learning for cloud incident management using LLMs," arXiv preprint, arXiv:2404.03662, 2024. Available: https://arxiv.org/abs/2404.03662

[48] K. Sarda, Z. Namrud, M. Litoiu, L. Shwartz, and I. Watts, "Leveraging large language models for the auto-remediation of microservice applications: An experimental study," in *Companion Proc. of the 32nd ACM Int. Conf. on the Foundations of Software Engineering (FSE)*, Porto de Galinhas, Brazil, 2024, pp. 358–369. doi: 10.1145/3663529.3663855

[49] K. Sarda, Z. Namrud, R. Rouf, H. Ahuja, M. Rasolroveicy, M. Litoiu, L. Shwartz, and I. Watts, "ADARMA: Auto-detection and auto-remediation of microservice anomalies by leveraging large language models," in *Proc. 33rd Annu. Int. Conf. on Computer Science and Software Engineering (CASCON)*, Las Vegas, NV, USA, 2023, pp. 200–205. doi: 10.5555/3615924.3615949

[50] T. Ahmed, S. Ghosh, C. Bansal, T. Zimmermann, X. Zhang, and S. Rajmohan, "Recommending root-cause and mitigation steps for cloud incidents using large language models," in *Proc. 45th Int. Conf. Software Engineering (ICSE)*, Melbourne, Victoria, Australia, pp. 1737–1749, 2023. doi: 10.1109/ICSE48619.2023.00149

[51] J. Wang, G. Chu, J. Wang, H. Sun, Q. Qi, Y. Wang, J. Qi, and J. Liao, "LogExpert: log-based recommended resolutions generation using large language model," in *Proc. 2024 ACM/IEEE 44th Int. Conf. Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, Lisbon, Portugal, pp. 42–46, 2024. doi: 10.1145/3639476.3639773

[52] P. Khlaisamniang, P. Khomduean, K. Saetan, and S. Wonglapsuwan, "Generative AI for self-healing systems," in *Proc. 2023 18th Int. Joint Symp. on Artificial Intelligence and Natural Language Processing (iSAI-NLP)*, 2023, pp. 1–6. doi: 10.1109/iSAI-NLP60301.2023.10354608

[53] D. Chaudhary, S. L. Vadlamani, D. Thomas, S. Nejati, and M. Sabetzadeh, "Developing a Llama-based chatbot for CI/CD question answering: A case study at Ericsson," in *Proc. 2024 IEEE Int. Conf. on Software Maintenance and Evolution (ICSME)*, Oct. 2024, pp. 707–718. doi: 10.1109/ICSME58944.2024.00075

[54] P. Sharma and M. S. Kulkarni, "A study on unlocking the potential of different AI in continuous integration and continuous delivery (CI/CD)," in *Proc. 2024 4th Int. Conf. on Innovative Practices in Technology and Management (ICIPTM)*, 2024, pp. 1–6. doi: 10.1109/ICIPTM59628.2024.10563618

[55] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *Proc. 2017 IEEE Int. Conf. on Web Services (ICWS)*, 2017, pp. 33–40. doi: 10.1109/ICWS.2017.13

[56] C. E. Brandt, A. Panichella, A. Zaidman, and M. Beller, "LogChunks: A data set for build log analysis," in *Proc. 17th Int. Conf. on Mining Software Repositories (MSR)*, Seoul, Republic of Korea, 2020, pp. 583–587. doi: 10.1145/3379597.3387485

APPENDIX A

Dataset Availability: The dataset is publicly available at https://github.com/ByteLuo1029/dataset.

Manual Evaluation Criteria for Solutions

2 Points (E2E Good Case):

- **Non-code issues:** The solution provides *direct and actionable instructions*, such as specific tools or recommended container images.
- **Code issues:** The solution clearly identifies the exact file and location of the issue based on logs, and proposes a concrete fix or a well-referenced example.

1 Point:

- **Non-code issues:** The solution is closely aligned with the root cause but only provides *indirect suggestions* (e.g., vague instructions without actionable configuration or tooling).
- **Code issues:** The solution references the relevant logs and proposes a plausible fix, but *fails to specify the exact file or code location*.

0 Point:

- The solution is irrelevant to the true root cause, lacks sufficient log grounding, or includes incorrect content possibly due to hallucinated knowledge (e.g., fabricated repository names).

Penalty:

- If any document link in the solution is invalid or broken, *deduct 1 point* from the overall score.

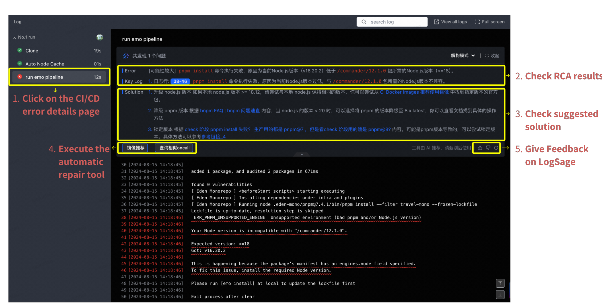

Fig. 8: Screenshot of the online user interface.