# PAGER: Proactive Monitoring Agent for Enterprise AI Assistant

**Sujan Dutta[1*†], Junior Francisco Garcia Ayala[2*†], Pranav Pujar[3*†], Sai Sree Harsha[4], Dan Luo[4], Nikhil Vasudeva[4], Bikas Saha[4], Pritom Baruah[4], Yunyao Li[4]**

[1]Rochester Institute of Technology
[2]New York University
[3]The University of Texas at Arlington
[4]Adobe

## Abstract

We present a Proactive Monitoring Agent designed for large-scale customer data platforms, such as Adobe Experience Platform (AEP), to predict and prevent workflow disruptions before they impact business operations. Unlike existing reactive solutions that assist engineers only after failures occur, our agent anticipates potential failures across multiple workflow stages, explains its predictions in natural language, and interacts with customer support engineers through a conversational interface. The system integrates a machine learning-based Prediction Module, Knowledge Graph APIs for contextual data access, and a Query Processor that powers an interactive Q&A experience, enabling timely and actionable insights to minimize operational risks and maximize business continuity.

## Introduction

Businesses rely on customer data platforms such as Adobe Experience Platform (AEP) to manage large-scale customer journeys. These workflows span interconnected stages – data ingestion, audience segmentation, and delivering journeys to target audiences (AEP Documentation 2025). Failures at any stage can cascade downstream, causing disrupted customer journeys and revenue loss. Such failures are typically discovered only after adverse effects occur, with businesses raising support tickets that engineers resolve reactively. Existing enterprise AI assistants embedded in these platforms provide limited help as they mainly answer customer-facing queries (e.g., product documentation via RAG or NL2SQL over customer data) and lack the ability to anticipate failures or support proactive remediation.

Recent research has explored LLM-based assistants for system troubleshooting. RCACopilot (Chen et al. 2024) and ReAct (Roy et al. 2024) leverage LLMs to predict root causes of incidents and generate natural language explanations. ReAct further improves adaptivity by dynamically gathering additional runtime information. DTTool (Hanafi et al. 2025) surfaces annotations of anomalous events in speech pipelines through an interactive dashboard. While useful, all three approaches remain reactive and help engineers only after an issue has already disrupted operations.

In this work, we present PAGER, (**P**roactive monitoring **AG**ent for **E**nterp**R**ise AI Assistant), a unified interactive agent for preemptive failure management. PAGER (1) predicts failures across workflow stages, (2) explains predictions through clear natural language narratives, and (3) supports engineers through conversational Q&A, enabling them to remediate issues before customers are impacted.

## System Overview

PAGER's focus lies on identifying possible errors in three interdependent stages in AEP: *batch ingestion* of customer data, *segmentation* of customer data into audiences, and creation of *journeys* for each audience. This is achieved through a plug-and-play agent designed to seamlessly integrate with the existing AI Assistant at AEP (Maharaj et al. 2024). It is made up of three parts: (1) Prediction Module, (2) Runtime Query Processor, and (3) Knowledge Graph API.

**Prediction Module** It is responsible for predicting potential overlapping job failures in the workflow and generating a natural language description explaining the likely cause. This module consists of two parts.

*A. Predictive Modeling:* We train two separate random forest classifiers on historical error logs: one for predicting overlap between ingestion and segmentation, and another for predicting overlap between segmentation and journey. Each model leverages a feature set engineered from temporal and performance characteristics of prior runs, such as job start time, time until the next job, average number of records ingested in the past week, number of failed jobs in the past seven days, and average time duration of jobs in the past week. This design allows the models to capture both short-term operational anomalies and long-term historical patterns that are predictive of downstream disruption.

*B. Explanation Generation:* To ensure that predictions are interpretable and actionable, we couple the predictive models with an explanation generation pipeline. First, we compute feature contribution scores using Shapley values (Lundberg and Lee 2017). The Shapley scores, along with feature descriptions and the model's prediction, are then passed to an LLM, which is prompted to generate a natural language

*These authors contributed equally.
†Work done while interning at Adobe.

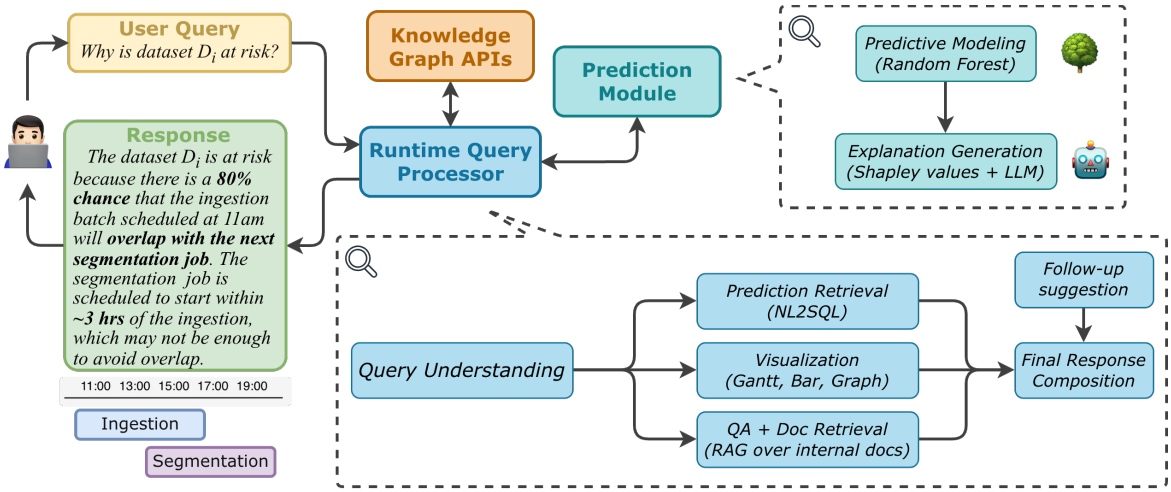


Figure 1: Architecture diagram of PAGER.

narrative describing the likely cause of the predicted issue.

**Runtime Query Processor** It provides a multi-turn conversational user interface for support engineers to investigate issues identified by the prediction module. The pipeline includes several tasks powered by LLMs:

*A. Query Understanding:* Rewrites ambiguous queries using chat history and determines whether the query should be routed to one or more of the following tasks: prediction retrieval, QA and document retrieval, or visualization.

*B. Prediction Retrieval:* Runs NL2SQL (Liu et al. 2025) over the prediction data and returns the predicted classes, their respective probabilities, and the LLM-generated explanation from the Prediction Module.

*C. QA and Document Retrieval:* Uses Retrieval-Augmented Generation (Gao et al. 2024) to generate possible error remediation solutions using AEP documentation.

*D. Visualization:* Transforms prediction and knowledge graph data into the following visualizations: a Gantt chart that visualizes the timing overlaps between stages, a bar or donut chart showing predicted class probabilities, and an interactive graph visualization that visualizes lineage between three AEP entities: datasets, segments, and journeys.

*E. Final Response Composition:* Synthesizes the outputs of tasks 1–4 into a digestible natural language response.

*F. Follow-up Question Suggestion:* Generates suggested follow-up questions based on chat history and the current response, seeded by the capabilities of our prototype.

**Knowledge Graph API** The knowledge graph API captures lineage across three AEP entities: datasets, segments, and journeys. These entities are pairwise interconnected in a knowledge base and uniquely identified by IDs. When a request is made for an entity's lineage (e.g., a dataset), the API returns all related entities (e.g., related segments and journeys).

## Results

**Model Evaluation** For predicting overlaps between ingestion and segmentation, the random forest model achieves an F1 score of 67.8 $\pm$ 1.1, significantly better than a random predictor (8.8 $\pm$ 1.9) and logistic regression (44.4 $\pm$ 4.8). For predicting overlaps between segmentation and journey, the random forest model achieves an F1 score of 57.5 $\pm$ 4.4, significantly better than a random predictor (3.3 $\pm$ 6.7) and logistic regression (43.6 $\pm$ 21.8).

**User Study** In a within-subject user study design, 10 AEP employees were asked to identify at-risk AEP entities with PAGER and with a baseline representing existing enterprise error-finding tools in the organization. Through a 5-point Likert-scale questionnaire, users reported that they found the task easier overall with PAGER (4.70 $\pm$0.483) compared to the baseline (2.10 $\pm$0.876) and reported greater confidence in their results when using PAGER (4.10 $\pm$ 0.738) versus the baseline 2.40 $\pm$ 1.265). Both differences were statistically significant ($p < 0.01$).

## Case Study

We demonstrate PAGER 's capabilities through a scenario where a customer support engineer (CSE) utilizes PAGER to monitor potential issues affecting a fictional company. PAGER begins the session with a dashboard highlighting two journeys at risk. When the CSE inquires about the first journey, PAGER traces the problem to a segmentation job likely to overlap with the dependent journey. A Gantt chart complements the natural language explanation, and when asked for a fix, PAGER synthesizes remediation steps from AEP documentation. For the second journey, PAGER identifies a batch ingestion job feeding multiple datasets as the source of risk. When the CSE probes further, the agent explains that the risk comes from an overlap between the batch ingestion and a segmentation job, also visualized by a Gantt chart. The CSE then prompts to see the downstream impact, and PAGER provides a lineage graph showing affected journeys and segments, along with possible remediation steps.

## References


AEP Documentation. 2025. Adobe Experience Platform Documentation. https://experienceleague.adobe.com/en/docs/experience-platform/ingestion/batch/overview; https://experienceleague.adobe.com/en/docs/experience-platform/segmentation/home; https://experienceleague.adobe.com/en/docs/journey-optimizer/using/orchestrate-journeys/journey. Accessed: 2025-11-11.

Chen, Y.; Xie, H.; Ma, M.; Kang, Y.; Gao, X.; Shi, L.; Cao, Y.; Gao, X.; Fan, H.; Wen, M.; et al. 2024. Automatic root cause analysis via large language models for cloud incidents. In *Proceedings of the Nineteenth European Conference on Computer Systems*, 674–688.

Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, M.; and Wang, H. 2024. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv:2312.10997.

Hanafi, M. F.; Reiss, F.; Katsis, Y.; Moore, R.; Falakmasir, M. H.; Wang, P.; Wood, D.; and Liu, C. 2025. Diagnosing and Prioritizing Issues in Automated Order-Taking Systems: A Machine-Assisted Error Discovery Approach. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, 1–18.

Liu, X.; Shen, S.; Li, B.; Ma, P.; Jiang, R.; Zhang, Y.; Fan, J.; Li, G.; Tang, N.; and Luo, Y. 2025. A Survey of Text-to-SQL in the Era of LLMs: Where are we, and where are we going? arXiv:2408.05109.

Lundberg, S. M.; and Lee, S.-I. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Maharaj, A. V.; Qian, K.; Bhattacharya, U.; Fang, S.; Galatanu, H.; Garg, M.; Hanessian, R.; Kapoor, N.; Russell, K.; Vaithyanathan, S.; and Li, Y. 2024. Evaluation and Continual Improvement for an Enterprise AI Assistant. In *Proceedings of the Fifth Workshop on Data Science with Human-in-the-Loop (Language Advances)*, 17–24. Mexico City, Mexico: Association for Computational Linguistics.

Roy, D.; Zhang, X.; Bhave, R.; Bansal, C.; Las-Casas, P.; Fonseca, R.; and Rajmohan, S. 2024. Exploring llm-based agents for root cause analysis. In *Companion proceedings of the 32nd ACM international conference on the foundations of software engineering*, 208–219.