

ARTIFICIAL INTELLIGENCE ENABLED SOFTWARE FOR EARLY DISEASE DETECTION, RISK PREDICTION, AND PERSONALIZED TREATMENT PLANNING

Sofia Martinez
Spain

ABSTRACT

Artificial intelligence-enabled healthcare software offers an advanced approach to early disease detection, risk prediction, and personalized treatment planning. This study explores a software framework that combines electronic health records, medical imaging, laboratory results, wearable data, and patient history to identify abnormal patterns before serious complications develop. Machine learning models can estimate disease probability, classify patient risk levels, and support clinicians in selecting suitable diagnostic tests and treatment options. The system may also generate personalized care recommendations based on age, symptoms, genetics, lifestyle, medication response, and existing conditions. Continuous learning allows the software to improve predictions as new clinical data become available. Explainable outputs, data encryption, access control, and clinician review are included to support safety, transparency, and responsible decision making. Overall, the proposed software can improve diagnostic speed, reduce preventable risks, strengthen clinical decisions, and support more precise, timely, and patient-centered healthcare services across hospitals, clinics, and remote care environments.

keywords: Artificial Intelligence in Healthcare; Early Disease Detection; Risk Prediction; Personalized Treatment Planning; Clinical Decision Support.

1. Introduction

Enterprise applications depend on reliable diagnostics to detect runtime errors, failed workflows, API issues, slow transactions, and integration faults. Logs are the main evidence source for understanding what happened before, during, and after an incident. Effective application diagnostics require structured logging, traceability, contextual metadata, error classification, and operational visibility [1]. However, fixed logging strategies often create two problems. Low-detail logs may miss important causes, while high-detail logs may generate excessive storage cost and alert noise.

The main challenge is that diagnostic needs change during runtime. A normal transaction may require only basic logs, but a failing workflow, repeated exception, or security-sensitive event may need deeper traces. Adaptive logging improves this process by changing log depth according to system condition, transaction risk, and error behavior [2]. AI-assisted monitoring can identify when additional diagnostic detail is needed and when logging can return to normal levels. This article proposes an adaptive logging framework for enterprise application diagnostics.

2. Methodology

The proposed framework captures log metadata from application servers, API gateways, databases, background jobs, authentication modules, and integration services. It records event type, severity, timestamp, user role, transaction ID, request path, error frequency, response time, affected component, and workflow state. These features are converted into diagnostic context profiles. Adaptive logging requires context because the same event may be low-risk in one workflow but critical in another [3].

The logging controller classifies runtime states as normal, warning, error-prone, performance-risk, security-sensitive, workflow-critical, or incident-active. Based on this classification, it adjusts log level, trace depth, parameter capture, correlation ID tracking, and diagnostic enrichment. Lightweight scoring and policy rules are used so logging decisions remain fast and explainable [4]. Sensitive fields are masked before storage, and transaction-critical events are linked with workflow and database identifiers for easier diagnosis.

The framework is evaluated using simulated enterprise applications involving APIs, database workflows, authentication events, scheduled jobs, and service integrations. Static logging is compared with adaptive logging under repeated exceptions, slow requests, failed jobs, API timeout, suspicious login, and workflow interruption scenarios. Evaluation metrics include diagnosis time, log volume reduction, root-cause evidence quality, missed-event rate, storage overhead, alert usefulness, and administrator acceptance rate [5]. Feedback from resolved incidents, ignored logs, and administrator notes is used to refine logging policies over time.

3. Results and Discussion

The results show that adaptive logging improves diagnostics by capturing richer evidence only when runtime risk increases. In the baseline condition, static logs either lack enough detail or produce large volumes of low-value records. With the proposed framework, detailed traces are activated for high-risk events and reduced during normal execution. The strongest improvement appears in intermittent failures, where deeper logs are needed only around the failure window.

The discussion shows that adaptive logging must balance diagnostic depth with privacy and storage cost. Excessive logging can expose sensitive data or overload monitoring systems, while weak logging may hide failure causes. The main limitation is that log-level decisions depend on accurate runtime classification. Regular policy review and masking controls are necessary for safe long-term use.

4. Conclusion

This article presented an adaptive logging framework for enterprise application diagnostics. The framework adjusts log detail using runtime context, risk scoring, workflow state, and error behavior. It improves troubleshooting by increasing diagnostic evidence during abnormal conditions while reducing unnecessary log noise during normal operation. The study shows that adaptive logging can strengthen enterprise diagnostics when combined with structured metadata, privacy controls, and continuous incident feedback.

References

1. Keshireddy, S. R. (2019). Oracle APEX Integration with RESTful Services for Lightweight Enterprise Data Exchange. *Journal of Grid, Machines and Drives*, 6, 7-12.
2. Keshireddy, S. R. (2020). Package-Based PL/SQL Architecture for Business Logic Separation in Oracle APEX. *Emerging Digital Systems*, 7, 1-6.
3. Keshireddy, S. R. (2019). Audit Trails in Oracle APEX with Database Triggers and Session Metadata. *High-Performance Distributed Computing Review*, 6, 1-6.
4. Keshireddy, S. R. (2020). Declarative Web Applications in Oracle APEX for Department-Level Workflows. *Cross-Disciplinary Knowledge Systems*, 7, 1-6.
5. Keshireddy, S. R. (2020). Oracle APEX Management Dashboard Development Through SQL Aggregation Views. *Journal of Privacy-Aware Computing Systems*, 7, 1-6.