Enhancing Observability: Real-time Application Health Checks

Tim Eichhorn[1], João Luiz Rebelo Moreira[1 [0000−0002−4547−7000]], Luís Ferreira Pires[1 [0000−0001−7432−7653]], and Lucas Meertens[1,2]

[1] University of Twente, Drienerlolaan 5, 7522 NB Enschede, The Netherlands
[2] CAPE Groep, Transportcentrum 16, 7547 RW Enschede, The Netherlands
t.s.eichhorn@student.utwente.nl
https://www.utwente.nl/

Abstract. Log management and application health monitoring practices are cumbersome and often still require significant human intervention to prevent inaccurate data and information. Existing technologies like Elasticsearch and Grafana offer opportunities to automate and improve these practices. This paper reports on a design solution aimed at enhancing log categorization, anomaly detection, and real-time application health reporting for CAPE Groep's service application. The proposed solution leverages Elasticsearch's Machine Learning capabilities and Grafana's dynamic visualization tools, alongside a newly developed dashboard named Horus, to centralize log data and automate monitoring processes. Preliminary results indicate that the proposed solution significantly improves the accuracy and timeliness of health reports, reduces manual intervention, and provides comprehensive real-time insights into application performance. This paper outlines the requirements, architectural design, and phased implementation plan, demonstrating the potential to streamline operations, enhance service delivery, and support future more stringent scalability requirements.

Keywords: Observability · Application monitoring · Log Management · Log Analytics · Elasticsearch · Dashboard

1 The Situation

This paper explores the enhancement of log management and real-time application health monitoring for CAPE Groep's service application, leveraging Elasticsearch's machine learning capabilities, Grafana's visualization tools and the development of a custom dashboard, that we called Horus. This section introduces application health monitoring through log management and provides an overview of CAPE Groep and the CAPE Service Point (CSP) application, detailing the current challenges and proposed solutions to improve log management and application health reporting.

1.1 Application Health Monitoring

Application health monitoring is critical in modern software development and operations, providing essential insights into the behaviour of systems in production [5]. Effective monitoring allows for the detection and diagnosis of undesired behaviours, contributing significantly to system reliability and operational efficiency. Central to this monitoring is log management and analysis, which involves collecting, processing, and analyzing log data generated by applications and their runtime environments [5].

Log data records events about the internal state of a system, and plays a pivotal role in understanding system behaviour, diagnosing issues, and improving overall reliability. As highlighted in [5], monitoring complex systems and deriving actionable insights from log data is a challenging task, requiring sophisticated tools and techniques. Despite the availability of advanced log management solutions, such as the Elastic stack, challenges remain in effectively leveraging these technologies to extract meaningful insights. Current log analysis processes are often time-consuming, error-prone, and lack real-time data insights, underscoring the need for fully automated solutions. Furthermore, the importance of data visualisation for complex data, such as application health metrics, has been repeatedly highlighted in literature [26]. However, the adoption of Machine Learning and advanced log analysis methods, and real-time visualisation offer promising avenues to enhance the efficiency and effectiveness of log management practices. By automating the categorization, analysis, and visualisation of log data, organizations can improve their ability to monitor application health, detect anomalies, and respond proactively to potential issues.

1.2 CAPE Service Point

CAPE Groep[3] is an IT consultancy firm specialized in developing applications and integrations for organizations in the transport and logistics, supply chain, smart construction, and agrifood sectors. They leverage low-code technology, specifically Mendix[4] and eMagiz[5], to quickly create effective solutions with flexibility for future needs. Next to development and consultant services, CAPE Groep also offers application monitoring and support services. These services include, but are not limited to, monitoring the status and health of applications, providing insights into causes of issues and problems, and managing customer contact through tickets.

To provide these support services, CAPE Groep has developed the Cape Service Point (CSP) application that is used by customers in a customer's specific environment and by various CAPE employees. Originally the system was created as a ticketing system for the support department of CAPE Groep, however, it has evolved into a system that provides multiple functionalities and services. For the support department to provide the highest quality of services to both

[3] https://capegroep.nl/
[4] https://www.mendix.com/
[5] https://emagiz.com/

clients and employees, automation and optimization of the many processes within CSP is essential. Furthermore, automation and optimization provide scalability, allowing CAPE to work towards taking a more proactive role in predicting and solving issues before they affect the customers. However, although the system has many automated components there is room for improvement.

1.3 Log Management & Analytics

In the current setup, CSP collects log files daily using direct API integrations provided by Mendix. At the same time applications send log entries in real-time to a database leveraging Elastic's Elasticsearch, which is an open-source, highly scalable, and distributed real-time RESTful search and analytics engine [11]. These logs are transmitted using a custom LogTransporter module publicly available [13]. Initially, logs were to be centralized in Elastic, however, due to the perceived inability for categorization, logs are also collected by CSP for this purpose. Log categorization refers to the process of grouping similar log entries to prevent data from being cluttered by a single repeating error message. This results in large volumes of redundant data that not only consume storage space but also complicate data management processes. Apart from redundancy, currently Elasticsearch effectively aggregates logs from various applications but is not extended with functionality to categorize or analyze these logs meaningfully. The current setup only allows for alerting on the total count of log messages, which is not a very informative performance indicator [5, 7]. This limitation results in a significant underutilization of Elasticsearch's capabilities, particularly its powerful data processing and analysis tools which could provide insightful analytics and real-time reporting.

Health Check Reports One of the services provided by CAPE performs regular health checks of applications and generates reports for customer review. These health check reports are crucial for maintaining transparency for clients about the performance and status of their applications. However, the current implementation for generating these health checks is notably inefficient and often produces undesirable results, requiring manual intervention.

In the current setup, a daily scheduled event within CSP collects all log files of each managed Mendix application individually using the Deploy API [20]. CSP then compares log entries line by line to combine individual log lines into categories if they are similar. This is computationally intensive as log files contain between 4 million and 20 million entries. Therefore, the process only runs during the night and even rejects log files with a size larger than 15MB to prevent CSP from crashing due to CPU and memory issues. Given that large log files are often an indicator of problems [5], potentially excluding this valuable information is not only undesirable but also partially defeats the purpose of log collection and categorization.

Furthermore, the reports generated based on these data often provide incomplete information on issues that occurred. In this scenario, manual intervention is

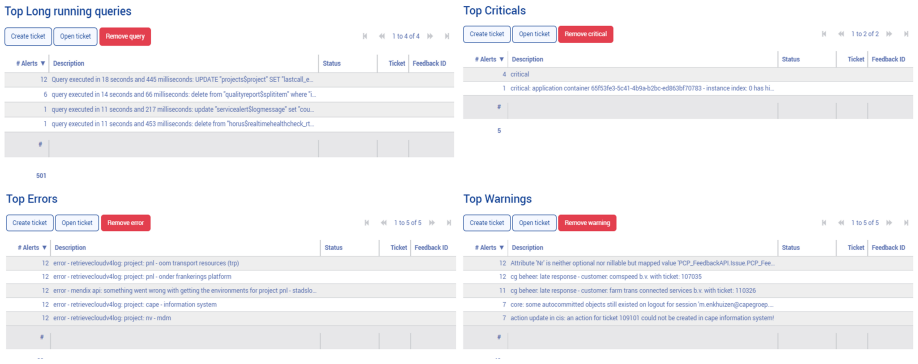

Fig. 1. The log analysis data in a health check report

required by the support staff, who manually download the log files from Mendix, convert them to categories using a Python script that runs locally on a personal computer and then generate the health check report based on these data instead of the CSP database. This process is both time-consuming and error-prone because a support member has to spot missing data in the automatically generated health check report, before being aware of the need to import log files manually. Figure 1 shows example of how log data is presented in the health check report.

Dashboard. Besides the health check reports, the current dashboard available in CSP provides functionality for monitoring alerts and managing tickets. This system enables support staff to receive and respond to notifications regarding various application issues, and to track the status and resolution of tickets. However, to gain comprehensive insights into application health additional steps are necessary, as the existing CSP interface lacks an advanced analytical overview that combines this information with application health metrics, such as log data. Consequently, staff often need to access multiple systems such as Grafana and Mendix separately to gain a comprehensive view. In this scenario, Grafana is used for real-time data visualization and general monitoring dashboards, while Mendix displays application-specific metrics. Consolidating this information into a single overview should provide users with a single source of information related to application health displayed on a dashboard.

1.4 Challenges and Opportunities

The internal consultants at CAPE Groep face challenges in monitoring their IT landscape effectively because they need to simultaneously view data from multiple sources, which is time-consuming and error-prone. This fragmented approach hinders efficient workflow and comprehensive system health assessment, requiring a more integrated and streamlined solution within CSP. Furthermore, the current system lacks accurate real-time data, diminishing the value of the health

reports, highlighting the need for improved log management and analytics as well as a comprehensive, real-time dashboard to improve operational efficiency and decision-making. Additionally, shifting the log collection and analysis to a separate dedicated system allows CAPE to extend their observability services beyond Mendix applications, but also potentially support other projects.

The current state of log management and reporting in CSP highlights a significant gap between the capabilities of the utilized technologies and their application within the observability infrastructure. The reliance on manual processes for critical functions such as application health reporting and the underutilization of Elasticsearch for data analysis represent key opportunities for improvement. By leveraging the advanced features of Elasticsearch for log categorization and employing automated tools for data analysis and reporting, CSP can significantly improve its operational efficiency, reduce costs, and enhance service delivery to clients. This project aimed to exploit these areas of improvement.

2 The Task

2.1 Goals

The primary task of the project was to enhance the CSP system and CAPE's observability infrastructure by designing and implementing improvements to its log collection and analytics capabilities, focusing specifically on overcoming the challenges related to partial matching and grouping of log messages. Furthermore, we also enhanced the presentation and visualisation of the improved data and insights for both internal consultants and in the future customers through the development of a new customized dashboard, which we called Horus.

The project was conducted within the constraints of a Master's student internship period, totalling 14 weeks. Although budget details were not specifically outlined at the start of the project, the inherent nature of an internship project implies limited availability of both time and funding. Furthermore, one of the goals was to reduce the needed computational resources for log analysis, therefore the added costs from the solution should not exceed the saved costs.

Furthermore, the initial requirements of the project were:

- The proposed solution had to seamlessly integrate with CSP and the broader observability infrastructure of CAPE Groep.
- The solution had to be scalable to handle increasing volumes of log data efficiently. Performance efficiency was essential to ensure that the log analysis process did not negatively impact the overall system performance.
- CAPE Groep expressed a preference for the solution to be built with technologies already applied in their IT landscape, therefore reducing the need for new major investments and training employees.
- The new dashboard should provide a comprehensive overview of the metrics and information related to the application's health status while also supporting a landscape overview.

2.2 Stakeholders

The key stakeholders related to this project include the Support team lead, the CSP product owner, and various consultants who utilize CSP. The Support team lead was the company-assigned supervisor for the project. This role involved participation in regularly scheduled progress meetings and feedback sessions, and overall guidance. The CSP product owner was responsible for overseeing the development and enhancements of the CSP modules. The product owner was consistently available to guide the project, approve key decisions, provide suggestions, and ensure alignment with the CSP goals and purpose. Finally, various consultants were contacted during the project. To provide valuable input for understanding the broader use cases and integration requirements of the system. Furthermore, they provided direct feedback on the layout and functions of the Horus dashboard. The consultants were intermittently available but participated in focused interviews and feedback sessions.

3 The Approach

The approach applied in this project consisted of a structured process aimed at addressing the identified challenges and meeting the set objectives. The methodology was primarily iterative and incremental, leveraging principles from Agile development to ensure flexibility and responsiveness to evolving requirements and potential solutions. Additionally, a user-centred design philosophy was adopted, focusing on enhancing the user experience and addressing the specific pain points identified during stakeholder interviews. This approach facilitated continuous improvement through iterative cycles of development, feedback, and refinement ensuring that the solution remained aligned with user needs and organizational goals.

3.1 Project Steps

The first step of the project was an analysis and review of the current observability infrastructure at CAPE as well as the internal workings and architecture of the CSP system. In this step, we also performed stakeholder interviews to gather detailed requirements and gain insights into the pain points of the current implementations within CSP. We focused on the support team and internal consultants as they are the main actors who interact with both the CSP system and the observability infrastructure.

After that, we performed a literature review on both Elasticsearch and other existing state-of-the-art Machine-Learning techniques for log analysis. Based on the results of this review, further investigation and research were conducted to identify the benefits of implementing the Elastic Common Schema (ECS), which defines a common set of fields to be used when storing event data, such as logs [9], and capturing the requirements for this.

The next step was the development of a proof-of-concept to explore the feasibility of using Elasticsearch for log categorization and analysis given the determined requirements. This involved setting up a test environment, configuring Elasticsearch, and setting up and validating log parsing and categorization functionalities.

After validating the quality and feasibility of the proposed solution, the new log categorization setup was integrated with CSP, ensuring compatibility and minimal disruption to ongoing operations. Furthermore, various dashboards were developed leveraging Grafana's visualisation capabilities on the newly available data and insights. Next, feedback was collected from the stakeholders on the proof-of-concept and initial implementation, identifying areas for improvement.

Due to the large scope and limited time, a decision was made to focus on showing the feasibility and quality of the solution design rather than fully implementing it and deploying it to the production environment. We also developed a migration and implementation plan that describes various migration strategies and their associated benefits and downsides.

Finally, the Horus dashboard was also interactively developed. Design ideas and decisions were continuously discussed with both the CSP product owner and consultants to ensure requirements satisfaction and a user layout that provides a streamlined user experience. After the validation, the first version of this dashboard has been deployed to the production environment and is currently being used by the staff, confirming the success of the project.

3.2 Solution Design Requirements

Requirements were gathered from a comprehensive analysis of CSP's current challenges, the capabilities of Elasticsearch, and the business processes supported by the current log collection and aggregation infrastructure. Following the CSP analysis, and a discussion with the product owner, the main requirements and objectives of this solution design have been:

- *Improved log categorization (RQ1):* the proposed solution should improve the ability to analyse individual log messages and combine them into coherent categories, ultimately improving the accuracy and relevance of the generated health check reports. This should support CAPE's analysis of common issues and trends across their IT landscape.
- *Enhanced anomaly detection (RQ2):* building upon the enhanced log analysis and categorization, the solution should provide improved automated anomaly detection capabilities, preferably with customised alerting options.
- *Insightful dashboarding and reporting (RQ3):* by providing dynamic insights into top recurring log entries and operational anomalies, the dashboard should enable proactive management and optimization of CAPE's IT operations. Furthermore, it should replace the current implementation for health check report generation.

3.3 Technical Requirements

Apart from the solution design requirements, the technical foundation of the proposed solution should also adhere to the following constraints, amongst others:

- *Scalability and performance:* the solution should scale efficiently with CAPE's growth and be able to handle the increasing volume and velocity of log data.
- *Integration compatibility:* is seamless integration with the existing CSP and CAPE observability infrastructure, requiring minimal changes to the current operational workflows.
- *Data ingestion and processing:* the solution should support log data ingestion from at least Mendix applications. Furthermore, the log ingestion and processing is preferred to be a single solution to ensure the overall observability architecture does not become too complex.
- *API integrations:* the solution has to offer API endpoints to facilitate integration with both CSP and the other applications in CAPE's IT landscape, allowing for automated retrieval and transmission of logs, alerts, and reports.
- *Security:* all log-related data should be handled in a both fast and secure manner, ensuring that all data and information within the solution adhere to relevant security standards and privacy regulations while minimizing the performance overhead.

This solution design should set the foundation for a robust observability framework that leverages advanced data processing and analysis techniques to enhance operational intelligence and efficiency.

4 The Results

This section presents the findings from our study on enhancing log management and real-time application health monitoring. The results encompass three key areas: the insights gathered from our literature review on Machine Learning techniques for log analysis, the capabilities of Elasticsearch in our context, and the detailed design of the proposed solution. Together, these findings offer a robust framework for improving log categorization, anomaly detection, and real-time health reporting.

4.1 Machine Learning in Log Analysis

In the ever-evolving landscape of computing, the volume and complexity of logs generated by systems and applications have escalated dramatically due to the scale of distributed systems [2]. This section delves into various state-of-the-art methodologies currently found in the literature on log analysis, leveraging machine learning and advanced computational techniques to extract meaningful insights from vast datasets of log entries. Furthermore, at the end of the chapter,

a discussion on whether these methodologies are suitable for the current problem is presented.

The analysis of logs, especially in large-scale IT infrastructures, has transitioned from simple pattern matching to more complex machine learning models that predict, classify, and help in the proactive maintenance of systems [2]. The problem with the Log-based Anomaly Detection (LAD) problem consists of detecting anomalies from execution logs that record both abnormal and normal system behaviour [2]. In the current literature, there are many different streams of ideas towards optimizing log anomaly detection such as using Natural Language Processing [3, 17, 31], Word2Vec [29, 28, 18] and Deep Learning [31][21][34]

Natural Language Processing (NLP), techniques are increasingly utilized in the analysis of log data to automate error handling, pattern recognition, and predictive maintenance within complex systems [17, 3]. The integration of NLP with log analysis leverages the textual nature of log data, applying various linguistic models to interpret, categorize, and analyze data in a way that mimics human reading and comprehension [3]. This enables the extraction of valuable information from log files as if they were regular text documents. Leveraging modern NLP techniques to analyze the grammatical structure and context of log events, features and patterns can be extracted and then processed by standard machine learning algorithms for anomaly detection [3].

Furthermore, using NLP techniques makes log mining for anomaly detection more efficient, automated, and scalable overall reducing the need for manual intervention in log analysis processes. One of these modern NLP techniques is word embedding, specifically Google's Word2Vec algorithm [22], this algorithm can be applied to map words in log files to high-dimensional vector representations. These representations can then be used as a feature space for training classifies to detect anomalies in log data [29] [28].

Word2Vec, as stated previously Word2Vec techniques provide powerful tools for feature extraction from text data, which is instrumental in analyzing and interpreting large volumes of logs. Developed by Tomas Mikolov and his team at Google, Word2Vec models capture semantic relationships between words by learning to predict a word from its neighbours in a sentence, or vice versa [22]. It employs a shallow neural network architecture with one of two model frameworks: Continuous Bag of Words (CBOW) or Skip-Gram [22]. Both of these models use a similar approach, however they differ in the direction of the prediction objective. CBOW predicts a word based on its context words and Skip-Gram predicts context words from a target word [21].

The paper by Wang et al. [29] introduce LogUAD, a Word2Vec-based log unsupervised anomaly detection method designed to address challenges in analyzing system logs in large-scale distributed systems. The authors discuss the challenges in analyzing logs in large-scale distributed systems due to log instability, the increasing volume of logs, computational costs and lack of labelled data for supervised methods [29]. LogUAD is proposed as a solution to these challenges, showing promising results when compared to existing methods [29].

Another example, is that of the Log2Vec framework presented in [21]. This framework extends the traditional Word2Vec model by incorporating domain-specific semantics that significantly enhances the effectiveness of log analysis [29]. This is done by embedding the out-of-vocabulary (OOV) [6] words at runtime and extracting semantic information from the logs, which is crucial for tasks such as anomaly detection and system monitoring [5].

Although all of these start-of-art methods and machine learning models look very promising, there is a major downside, most of these are not fully developed and tested in practice. Most have been tested against commonly used Log datasets for research and comparisons, however, most of the models have not been implemented in a real-life scenario. Furthermore, most of the papers discussed do not provide the model or steps for recreation, therefore the time needed to fully develop and test a setup leveraging these advanced machine learning concepts will not be feasible in the limited time available for the assignment.

4.2 Elasticsearch capabilities for the project

Elasticsearch is a widely used open-source, highly scalable, and distributed real-time RESTful search and analytics engine designed for horizontal scalability, reliability, and easy management. One of the most common use cases for Elasticsearch is for logging [30], which is also used for monitoring applications in real-time and analyzing large datasets on the fly [32]. This is because Elasticsearch leverages the robust, full-text search capabilities of Apache Lucene [4], a popular search engine Java library, making it an adequate choice for applications requiring complex search features across large volumes of data. Lucene indexes a document through inverted index [4], which is a data structure that tracks which documents contain certain values, allowing efficient document search [11].

Elasticsearch Text Categorisation is a built-in component that uses machine learning (ML) for categorizing text documents or sentences into predefined categories. It examines the content and meaning of the text and then assigns the most suitable label through text labelling. In particular, the Elasticsearch categorization anomaly detection ML job aims at automatically categorizing similar text values together [1]. This feature enables the analysis of large volumes of machine-written text like log messages, being a suitable candidate for our scenario. The model is trained on the incoming log data and learns the normal values and patterns of a category over time. This allows the detection of anomalous behaviour based on the number of occurrences or based on the rarity of a message. The categorization job uses an unsupervised ML model, so it does not require predetermined categories or prelabeled training data. Instead, it automatically identifies patterns and similarities in the provided log data. The main

[6] Out-of-vocabulary (OOV) words refer to words that appear in a text but were not included in the training set vocabulary when a language model or a word embedding system, like Word2Vec, was initially trained.

downside of the ML job is that it only returns anomalies. Although it categorizes log messages in a desired manner, it will not return all categories, only the anomalous ones. This is useful for anomaly detection, however, it is insufficient given the fact that we also want to provide insight to the end users into, for example, the top 10 most occurring errors of a specific period.

Categorize Text Search Function is another built-in feature, introduced in Elasticsearch version 7.16, for text aggregation categorization [6]. This is a multi-bucket aggregation that groups semi-structured text into distinct categories based on textual similarity [8]. Similar to the anomaly detection job this process involves analyzing the text with a tokenizer, which breaks down the text into tokens. Once the text is analyzed, the tokens are clustered together with a modified version of the DRAIN algorithm [14]. The DRAIN algorithm is an online log parsing method that can parse logs in a streaming and timely manner. To accelerate the parsing process, DRAIN builds a token tree and considers earlier tokens as more important. Elastic has modified the algorithm slightly to allow for earlier merging of tokens in the provided text when building categories [27]. The difference between this function and the ML job is that this is done once when the search request is sent to Elasticsearch. Therefore it is not well suited to use for constant anomaly detection, however, this feature is particularly useful for creating overviews of top log messages occurring in applications, allowing system administrators to quickly identify and address frequent or critical issues.

Elastic Common Standard (ECS) is an open-source specification published and maintained by Elastic [9], which defines a set of common fields to be used when storing event data in Elasticsearch, such as logs and metrics. Transitioning current Elastic indices to adhere to ECS can bring significant benefits to the current log monitoring and analysis setup [5, 33]. However, like any significant system update, this transition also comes with potential downsides and costs [33]. Among the main benefits, we highlight that log data and its formats are one of the most important parts of a system observability or analysis input setup [5]. However, they often offer many possibilities of different formats, often custom-defined, which makes interpreting the fields a big challenge. This is a common challenge associated with using data from multiple heterogeneous data sources[16], and a common language for all log events can address this problem [5].

Implications. Although Elasticsearch does not offer one built-in solution that satisfies all of our requirements, it does offer solutions that can be combined with each other to realise the desired situation. In order to leverage the full functionality and benefits of Elasticsearch we further developed the current logging infrastructure to a more mature state by not only addressing the analytical processes but also the overall data format and structure. In a landscape with many different applications and systems, the struggle and need for interoperability and scalability is ever-increasing [10]. A standardized logging schema like

ECS makes it easier to manage and scale the logging infrastructure, enabling seamless communication among different systems [33]. Establishing and documenting a single vocabulary to describe the various field names of data reduces the chance for ambiguity and confusion. This also directly improves data analysis by making analysis and querying data more straightforward [16]. The same principle applies to alerting, standardizing streamlines the creation of alerting rules because the structure and meaning of logs are predictable and consistent. Combined these result in a reduced learning curve, minimized chances of errors, more accurate alerts, and faster incident response times.

Apart from general standardisation benefits, ECS has a few specific benefits in comparison to other standards. First of all, ECS is not only a standard for log formats. Implementing ECS simplifies the analysis of disparate data sources, supporting a wide range of use cases, including application performance, security, and other metrics from all types of sources [9]. Defining a common set of fields and objects to ingest data into Elasticsearch enables cross-source analysis of diverse data. As a result, cross-source correlations become implicit with every search, but if necessary you can still filter down to specific data sources.

Secondly, implementing ECS unlocks and unifies all modes of analysis currently available in the Elastic Stack. This includes search, drill-down and pivoting, data visualization, machine learning-based anomaly detection, and alerting [23]. Alongside this, it provides the ability to easily adopt analytics content directly from other parties that use ECS, whether Elastic, a partner, or an open-source project within the environment without modifications. Furthermore, fully adopting ECS allows users to search with the power of both structured and unstructured query parameters [9]. Overall, by implementing ECS we leverage the full power of both the ever-growing Elastic stack and all other projects built upon this community-driven standard.

Nevertheless, we also have to be aware that adopting ECS involves certain downsides, potential pitfalls and costs that are both general to any system's migration and ECS. First of all, transitioning to ECS from the current custom schema can be a resource-intensive process, requiring development effort and adjustments to existing data pipelines and log transmitting modules. Next to this, changing the current naming conventions necessitates investment in both training and documentation updates. Additionally, during the migration process, CAPE and its customers might face temporary reductions in logging and monitoring efficiency, which could (in)directly affect operational capabilities.

Furthermore, as always with the implementation of standards there is the risk of over-standardization, where the schema might not support all custom use cases without significant customization, potentially leading to data being fitted into unsuitable fields or losing granularity in data [10]. ECS allows for customisation, and while this does offer flexibility it could undermine some of the benefits of adopting a standardized schema by introducing inconsistencies and complicating data analysis. Furthermore, relying heavily on ECS's new releases means that regular updates to logging practices are required to align with the new versions of the schema, resulting in ongoing maintenance costs and efforts.

Finally, there is also a risk for the potential of increased storage costs, as ECS encourages the inclusion of extensive contextual information alongside each event, which could lead to larger indices in Elasticsearch. Without proper monitoring and management of these indices to optimize performance and size it could increase costs. However, given that the number of stored fields for the current format is 9 fields and compliance with ECS requires 10 fields the risk should be minimal.

Overall, the choice to fully adopt ECS in the logging setup is not merely a technical upgrade but also a strategic move towards a more integrated, scalable, and efficient observability architecture. Apart from optimizing the current process, it paves the way for future innovations in the overall monitoring and analysis capabilities.

Grafana is an open-source platform renowned for its powerful capabilities in querying, visualizing, alerting, and exploring metrics, logs, and traces, regardless of where they are stored [12]. It serves as a valuable tool for creating insightful graphs and visualizations from time-series data, enhancing the observability and operational intelligence of IT environments. Grafana's flexible plugin framework supports integration with various data sources, making it a versatile choice for data analytics [15].

Next to this, in a log observability infrastructure, Grafana can be seamlessly integrated with Elasticsearch to enhance monitoring and analysis capabilities. Elasticsearch efficiently handles the storage and querying of large volumes of log data, while Grafana provides the user interface to visualize and explore this data. Together, they offer a comprehensive solution for monitoring applications. Furthermore, in comparison with Elastic's visualisation component, Kibana, it offers more suitable pricing models for CAPE's needs. Due to these reasons, CAPE has opted to use Grafana for all its time-based data visualization needs.

4.3 Proposed Architecture of Observability Infrastructure

Figure 2 shows an excerpt of the high-level architectural overview of the observability infrastructure, showing only the relevant components to ensure readability and understanding. As we agreed before, the direct connection between Mendix applications and CSP made little sense given the structure of the rest of the setup. This connection represented the daily collection of log files for aggregation, categorization and finally reporting. Furthermore, the current setup already focused on leveraging Elastic as a central processing and storage environment. Therefore it was logical to focus on further centralizing this flow of data. The changes in the architecture are shown using a red cross for the removal of the integration and a green plus for the addition of categorised log anomaly detection.

Reduced Complexity and Resource Requirements. Figure 2 shows that CSP relied on a direct integration, with nightly API calls to Mendix for log collection, which are inefficient and should be removed as we discussed in section

1. The new architectural scenario removes the connection between the Mendix applications and CSP, as indicated in 2, simplifying the overall monitoring infrastructure. By centralizing log data, CSP makes multiple direct integrations unnecessary reducing the complexity of the application. Moreover, with centralization computational tasks previously handled within CSP can be removed, thereby freeing up resources that can be redirected to improve application performance, and user experience, and enable future innovations.

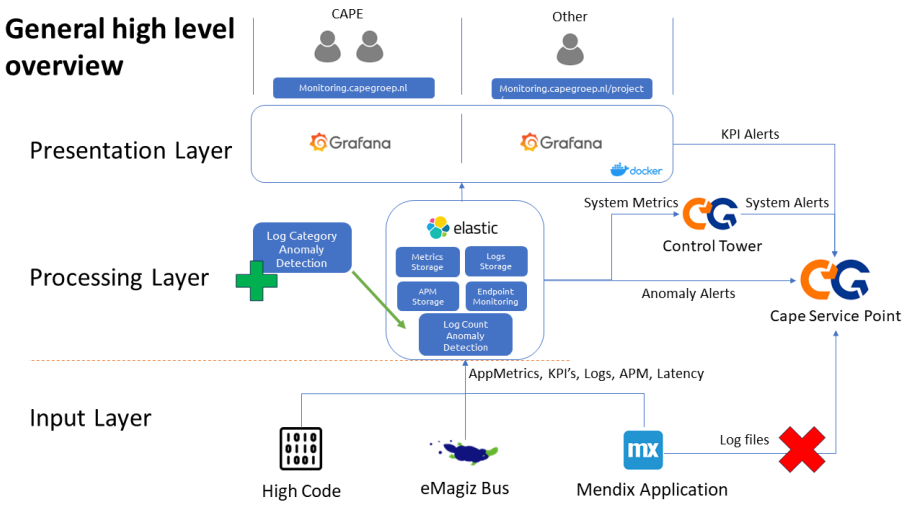

Fig. 2. Changes to the observability infrastructure architectural high-level overview

Centralisation of Log and Metrics Data. The proposed solution also centralises all log and metrics data in Elasticsearch. This approach offers several advantages such as simplified data management and improved correlation capabilities. By consolidating all logs and metrics into a single system, CAPE can manage its observability data more effectively, reducing the complexity and overhead associated with handling multiple data sources providing performance and scalability. Furthermore, storing data related to different components of applications allows for cross-functional and domain analysis by allowing different types of data from various sources to be correlated and analysed together. This is particularly useful for gaining comprehensive insights across various operational domains within CAPE, and has the potential to even be further extended in the future, for example with security data.

Standardisation to ECS. To fully leverage the benefits of centralizing logs, metrics, and other data, the data format should be standardised inside the com-

pany. Adopting ECS will facilitate this by providing a consistent framework for data formatting across all domains. Leveraging this standardisation, administrators and users can perform searches, analyses, and correlations across diverse data sets. Furthermore, standardizing ensures that as new types of data are incorporated into the system, which in turn can be easily integrated and analysed without significant modifications to the existing infrastructure.

Grafana. In addition to changes to Elasticsearch, we integrated advanced visualization capabilities in Grafana in order to improve even more CAPE's observability infrastructure. Grafana has been employed to effectively visualize the results from the newly implemented log categorization and analysis processes, facilitating an intuitive and actionable display of data for operational monitoring and decision-making.

Grafana has been used to display the categorized logs using dynamic visual components such as bar charts, pie charts, and tables. These visualisations show the distribution of log categories over time, highlighting the frequency and trends of various log types, including errors and warnings. This allows system administrators and support personnel to quickly identify and focus on the most critical issues that require attention. Figure 3 shows an example of a dashboard that we created to demonstrate these capabilities.

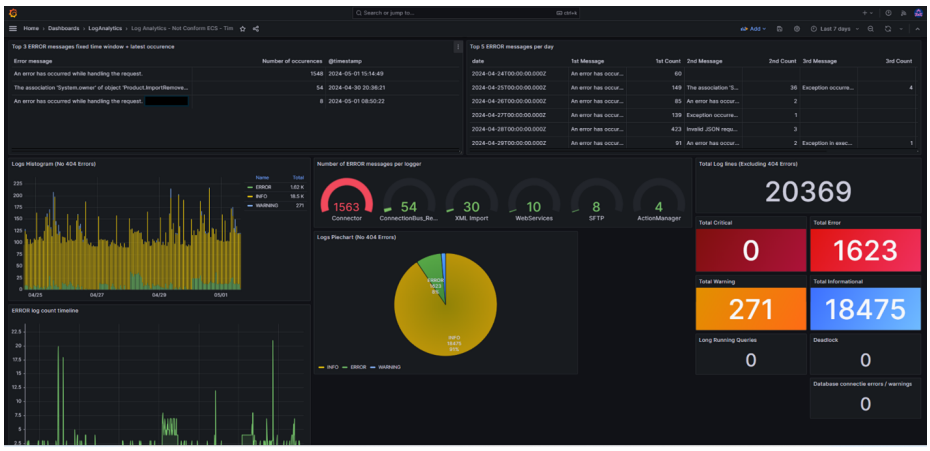

Fig. 3. A Dashboard leveraging the capabilities offered by the new setup

In addition, Grafana's real-time monitoring capabilities can now leveraged to offer up-to-the-minute insights into the application's health. The dashboards can be configured to receive live data feeds from Elasticsearch, ensuring that the displayed information is always up-to-date, which is critical for enabling real-time monitoring and rapid response to emerging issues. Furthermore, Grafana can now be set up to send alerts based on user-specified triggers identified through the

log analysis. For example, if the frequency of a particular error category exceeds a predefined threshold, Grafana can trigger an alert to notify the relevant teams. These alerts can be customised to match the severity and nature of the issues, ensuring appropriate prioritisation and response.

Cape Service Point. In this project, we also modified the CSP Mendix model. In the new scenario, the workflow is simplified by reducing the dependency on multiple processes and storage within CSP. Leverage the categorization capabilities of Elasticsearch instead. These changes resulted in the development of a new microflow that handles the interaction with Elasticsearch and subsequently retrieves the data for the generation of health checks. This single streamlined microflow replaces the entire log collection, aggregation, and log storage logic within CSP. The microflow directly queries Elasticsearch instead of calling the Mendix API for logs. The response from Elasticsearch includes logs that are already categorized and ranked based on frequency and severity. These categorised and ranked logs are used to automatically compile improved health check reports. These reports are more accurate, timely, and comprehensive than the current implementation.

4.4 Horus Dashboard

In addition to the improvements to the observability infrastructure, we also created an overview page to monitor application health as a dashboard, which we called Horus. This dashboard is made up of two layers of information, namely basic panels and advanced panels. The general overview displays the basic panels and provides information related to four aspects of application health. These panels provide the most high-level information on the status of the application selected in the dashboard. As shown in Figure 4.

This overview shows the status of all the tickets, alerts and log metrics related to the two selected applications. For each of these panels, we present the most high-level information, such as the number of open Priority 1 tickets or critical alerts. Additionally, for each piece of information, we include a trend that shows the difference between the currently selected time window (seven days in the example) and the same window before that. Furthermore, based on these trends different traffic lights indicate the severity of the trend accordingly with the colours green, orange, or red. The severity thresholds can be personalised for each end user, providing users with an easy and clear overview of the current IT landscape health status. If the user wants to know more details about one of the panels, they can click on them, and the advanced panels are opened.

Figure 5 shows that these panels include more detailed information on each category of information. For tickets and alerts, this consists of graphs showing the distribution of the count presented in the basic overview. Furthermore, a time series chart shows how tickets and alerts progress over time. Both of these charts were implemented using the Plotly JavaScript Open Source Graphing Library [25], which also means that they are interactive, so clicking them opens overview pages for the selected data point.

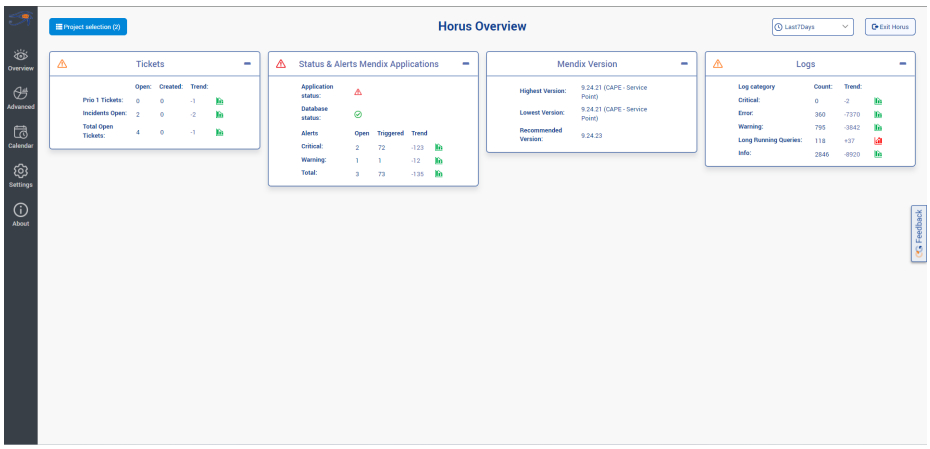

Fig. 4. The basic overview panels of Horus

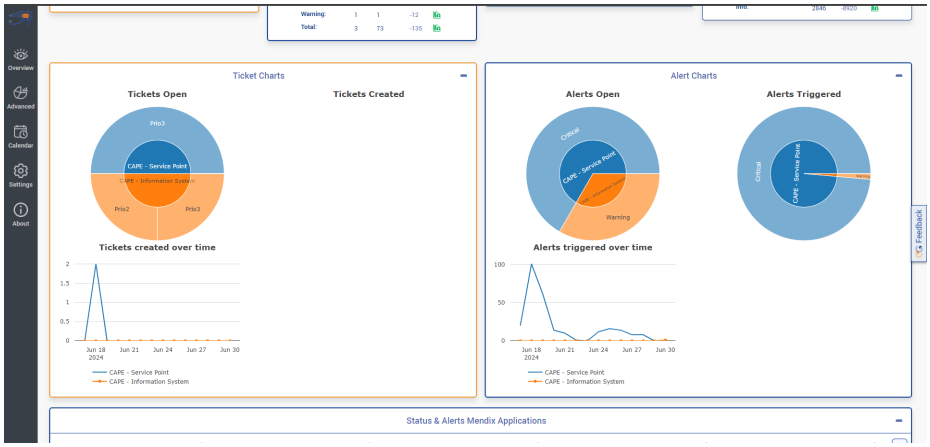

Fig. 5. The advanced panels for tickets and alerts of Horus

For example, if a user clicks on a certain alert level of an application, it opens an overview of those alerts for that specific application in the time window, so that users do not have to manually look for the tickets or alerts when needed.

Next to this, the user can click on the status & alerts and the logs basic overview panel to open the two advanced panels shown in Figure 6. This provides an overview of the status of each component of the selected applications. Furthermore, the logs advanced panel shows a Grafana embedded panel, implemented with IFrame, which is a log histogram that displays the distribution of different error severities over time. The screenshots in Figure 6 should give some insights into the functionality offered by Horus.

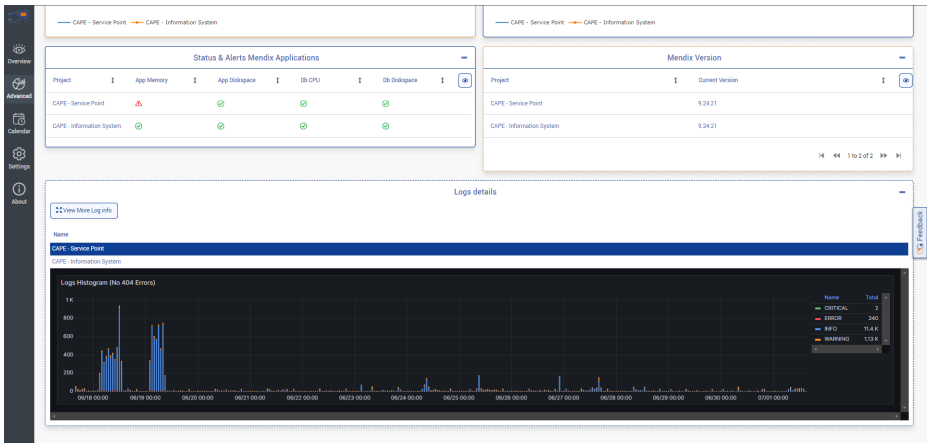

Fig. 6. The advanced panels for health status and logs of Horus

4.5 Solution Implementation Plan

The implementation plan for any IT system, particularly business-critical solutions like the one we proposed, involves careful consideration of various deployment strategies to ensure effectiveness and minimize disruption to ongoing operations. Before deciding on an implementation strategy, we evaluated each strategy based on specific organizational needs, risk assessments, and the critical nature of the systems involved. The following are common strategies were considered for implementing the proposed solution:

- *Big Bang Rollout [24, 19]:* This strategy consists of a complete and simultaneous transition from the old system to the new system across the entire organisation at a specific point in time. While this can be the fastest method to implement, it is also the riskiest, as it leaves little room for error and adjustment.

- *Phased Rollout [24, 19]:* This strategy consists of implementing the new system in stages over a specific period. In each phase, the solution is rolled out to a different segment of the end users. This can be done based on application, team, customer or even department. This strategy reduces the risk because it allows lessons learned in earlier phases to be applied to later ones.
- *Parallel Adoption [19]:* In this strategy, both the old and new systems run simultaneously for a significant period of time. This strategy is less risky because it allows users to transition gradually and ensures that the organisation can revert to the old system if significant issues arise in the new system. In this strategy, it is critical to ensure backwards compatibility.
- *Pilot Implementation [24]:* This consists of rolling out the new system to a small controlled group within the organisation before a full-scale implementation. This strategy can help uncover potential issues with the new system while limiting the impact on the broader organization.

For the deployment of our solution, we recommend a combination of the aforementioned strategies. This consists of creating a pilot implementation after which a phased rollout will begin, the exact segmentation of the phases can be determined at a later point, however, for this plan the choice is made to do this per application. Following the phased rollout, a transition period in which a parallel adoption takes place. During this period, both the old and new ways should run in parallel, ensuring that business-critical processes are not hindered and end users experience a smooth transition. Figure 7 shows that the implementation process iterates between rolling out the solution for new phases and running parallel. After successfully upgrading all the applications the process transitions to the last step namely the Full Rollout. In this step, all the transitions are double-checked to ensure they are set up properly before the old indices and field names are removed from Elastic.

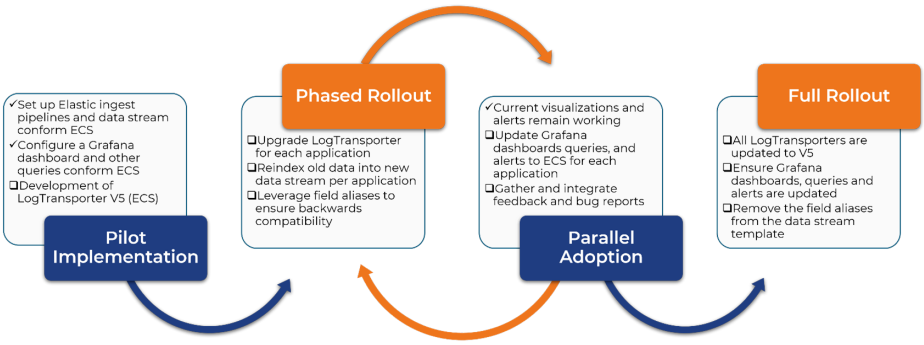

Fig. 7. A visualisation of the solution implementation plan

4.6 Achievements

Overall, this project at CAPE has culminated in an enhancement of CSP and CAPE's observability infrastructure, through the integration of advanced Elasticsearch functionalities and the use of Grafana and the custom Horus dashboard for visualisation. The primary objective of the project was to automate or enhance one of the processes within CSP. After initial investigation, interviews, and research the decision was made to focus the assignment on addressing the inefficiencies in CSP's existing log management and health check processes. By leveraging Elasticsearch's machine learning features for log categorization and anomaly detection, along with its powerful text categorisation and aggregation functions, we significantly automated and streamlined CSPs approach to log analysis. Hereby satisfying RQ1 and RQ2. Furthermore, automated processes have significantly reduced the time and support required to manage logs and generate reports. This has not only reduced errors-prone, and the manual labour previously required, but also increased the accuracy and timeliness of the health check reports provided to customers, satisfying RQ3.

Improved Health Check Reports. By leveraging Elasticsearch's categorization capabilities, CSP can improve the way health checks are conducted. Currently, health checks are generated through a resource-intensive process that combines log entries, which often requires manual intervention. This method is not only inefficient but also prone to errors, affecting the reliability and validity of the reports provided to customers. With the proposed solution, logs are automatically categorized and analyzed using Elasticsearch's machine learning capabilities or search categorization functionality, which can detect anomalies and categorize text in real time. This process not only speeds up data processing but also increases the accuracy of health checks by ensuring that all data is considered and appropriately categorized. We improved the old and new implementation and as is shown in Figure 8, multiple types of errors were not identified and could not be shown in the report. This can be verified by comparing Figure 8 to Figure 1, which displays the results of the old implementation for the same application. A satisfied consultant pointed out that "In the past, almost all reports were missing data or contained inaccurate information, I can already tell this is greatly improved". Overall, by eliminating the need for nightly log collection and aggregation within CSP, we significantly reduce the computational load on its servers. Furthermore, as the logs are processed and categorized at real-time by Elasticsearch, the health check reports generated are more current and reflect better the actual system status. The streamlined CSP microflow allows for a more efficient process flow, reducing the steps involved in generating reports and thereby decreasing the potential for errors and reducing the overall CSP complexity. All of this combined results in a more efficient, accurate, scalable, and reliable observability infrastructure.

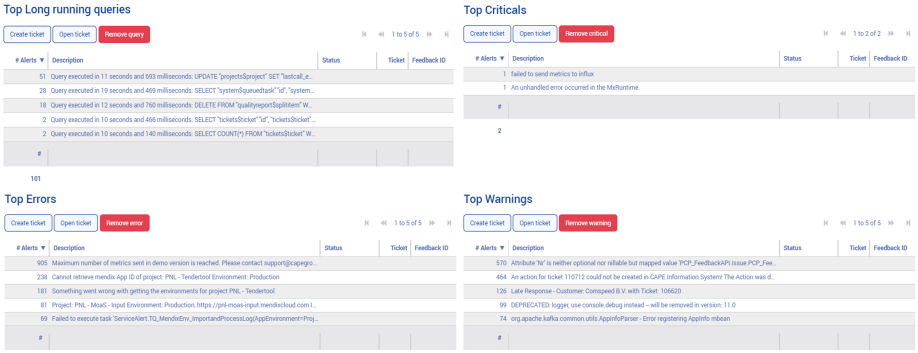

Fig. 8. The log data reported by leveraging the proposed solution

Improved Service Quality. The shift to an automated, Elasticsearch-powered setup for log analysis and health checks directly translated to improved service quality for CSP's customers. Automated categorisation and anomaly detection provide CSP with the ability to quickly identify and address issues before they impact service delivery. Furthermore, the enhanced data analysis capabilities ensure that health reports are both accurate and timely, fostering trust and reliability among clients. Next to this, the further development of the Horus dashboard has improved the visualization landscape at CSP, further satisfying RQ3. Customized dashboards now provide more and improved dynamic and real-time insights into the application's health, allowing for rapid response to emerging issues. This enhancement has not only improved internal operational efficiency but has also boosted the quality of service CSP provides to its clients, enhancing customer satisfaction and trust.

More generally, the architectural redesign using Elasticsearch as the backbone for CAPE observability infrastructure presents a robust solution that reduces architectural complexity, decreases resource consumption, and significantly improves the reliability and accuracy of operational health checks. This strategic overhaul not only streamlines internal processes but also enhances the quality of service delivered to customers. By effectively leveraging modern technologies like Elasticsearch and Grafana, CSP has not only addressed some of its immediate operational challenges but also laid a foundation for continuous improvement and innovation. As CSP continues to grow and evolve, the flexibility and scalability provided by these enhancements are expected to play a crucial role in its ability to meet future challenges and capitalize on new opportunities.

5 The Reflection

5.1 Limitations

Given that the project was mainly conducted by a student during an internship of only 14 weeks, there is a potential for sub-optimal solutions and missed opportunities. Many of the technologies and concepts explored during this project were new for the student, and given the fairly limited time frame concessions had to be made in terms of time spent researching possible solutions. This could have resulted in a sub-optimal solution design, however, given the produced benefits and usage of industry-wide adopted technologies this should be fairly limited. Furthermore, the initial implementation prioritised functionality over performance optimization. As a result, some processes can possibly be made more efficient. Finally, given that the research focused on the practical implications of the solution, a systematic or methodological evaluation of the benefits is not preferred. However, the results have been validated with qualitative feedback from management and consultants involved with the process.

5.2 Future Directions

While the project has achieved substantial improvements, there are several areas where future work could further enhance CAPE's and CSP's capabilities. First of all, something can be done to integrate non-Mendix applications and other additional data sources. The current setup is tested for Mendix applications, but there are many other types of applications within CAPE's IT portfolio. Expanding the types and sources of data integrated into the Elasticsearch framework could provide more comprehensive insights across other operational areas. Furthermore, the current setup lays the foundation for enhanced security features by integrating access logs into Elasticsearch. Integrating advanced security analytics into the observability infrastructure could help in better detecting and mitigating potential security threats. Next to this, performance benchmarking and optimization efforts can be made to ensure that the system remains responsive and scalable as data volumes grow. Finally, there is the opportunity for advanced predictive analytics. More complex machine learning models could predict potential system failures before they occur, further enhancing proactive monitoring.

6 Evidence

The case report was authored through a combination of primary practitioner involvement, stakeholder interviews, literature review, and practical implementation. The main author was a practitioner involved in the project as an intern. One of the co-authors is directly associated with CAPE Groep and provided firsthand insights and guidance on report writing. Interviews and regular feedback sessions with key stakeholders, such as the Support team lead and CSP

product owner, were conducted to gather requirements and validate findings. A review of relevant academic literature and technical documentation from Elastic and Grafana informed the project's methodologies and solutions. Practical implementation and testing within CAPE Groep's infrastructure were employed to ensure the feasibility and effectiveness of the proposed solutions. This comprehensive approach, combining direct practitioner input, research, and iterative development, ensured the accuracy and relevance of the project and its report.

7 Conclusion

To conclude, the project to enhance CSP's log analysis capabilities was a significant step forward to improve the system's functionality and user experience. While several challenges were faced and some limitations were identified, the integration of Elasticsearch's Machine Learning capabilities and Grafana's visualization tools, along with the Horus dashboard, streamlined operations, reduced manual intervention and provided comprehensive real-time insights into application health. The improved log categorization supported CAPE's analysis of common issues, enhancing the accuracy of health check reports. Enhanced automated anomaly detection and customisable alerts ensured proactive issue identification, improving system reliability. The Horus dashboard offered dynamic insights into recurring log entries and operational anomalies, optimizing IT operations. This strategic overhaul reduced architectural complexity, decreased resource consumption, and significantly enhanced operational health checks, ultimately boosting service quality and customer satisfaction. Seamlessly integrating with CSP and CAPE's infrastructure, the proposal provides a solid solution with a foundation for future improvements. By addressing the identified omissions and continuing to refine the solution, CAPE Groep can further enhance CSP, delivering even greater value to its users and customers.

References

1. Anomaly detection job types, https://www.elastic.co/guide/en/machine-learning/current/ml-anomaly-detection-job-types.html
2. Ali, S., Boufaied, C., Bianculli, D., Branco, P., Briand, L.C., Aschbacher, N.: An Empirical Study on Log-based Anomaly Detection Using Machine Learning. ArXiv **abs/2307.16714** (2023), https://api.semanticscholar.org/CorpusID:260334470
3. Bertero, C., Roy, M., Sauvanaud, C., Tredan, G.: Experience Report: Log Mining Using Natural Language Processing and Application to Anomaly Detection. In: 2017 IEEE 28th International Symposium on Software Reliability Engineering (ISSRE). pp. 351–360 (2017). https://doi.org/10.1109/ISSRE.2017.43
4. Białecki, A., Muir, R., Ingersoll, G.: Apache Lucene 4 (8 2012)
5. Cândido, J., Aniche, M., van Deursen, A.: Log-based software monitoring: a systematic mapping study. PeerJ Computer Science **7**, e489 (5 2021). https://doi.org/10.7717/peerj-cs.489
6. Courcy, D.: Elastic 7.16: Streamlined data integrations drive results that matter (12 2021), Elastic 7.16: Streamlined data integrations drive results that matter

7. Du, S., Cao, J.: Behavioral anomaly detection approach based on log monitoring. In: 2015 International Conference on Behavioral, Economic and Socio-cultural Computing (BESC). pp. 188–194 (2015). https://doi.org/10.1109/BESC.2015.7365981
8. Elastic: Categorize text aggregation, https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-bucket-categorize-text-aggregation.html
9. Elastic: Elastic Common Schema, https://www.elastic.co/elasticsearch/common-schema
10. Folmer, E., Verhoosel, J.: State of the Art on Semantic IS Standardization, Interoperability & Quality. Journal of Biomechanics - J BIOMECH (1 2011)
11. Gormley, C., Tong, Z.: Elasticsearch the definitive guide : a distributed real-time search and analytics engine. O'Reilly Media, 1 edn. (3 2015)
12. Grafana Labs: About Grafana, https://grafana.com/docs/grafana/latest/introduction/
13. Groep, C.: Mendix Marketplace - LogTransporter (2024), https://marketplace.mendix.com/link/component/218262
14. He, P., Zhu, J., Zheng, Z., Lyu, M.R.: Drain: An Online Log Parsing Approach with Fixed Depth Tree. In: 2017 IEEE International Conference on Web Services (ICWS). pp. 33–40 (2017). https://doi.org/10.1109/ICWS.2017.13
15. Kozhukh, D.: An easy look at Grafana architecture (3 2024), https://www.kozhuhds.com/blog/an-easy-look-at-grafana-architecture/
16. Kumar, G., Basri, S., Imam, A.A., Khowaja, S.A., Capretz, L.F., Balogun, A.O.: Data Harmonization for Heterogeneous Datasets: A Systematic Literature Review. Applied Sciences **11**(17), 8275 (9 2021). https://doi.org/10.3390/app11178275
17. Layer, L., Abercrombie, D.R., Bakhshiansohi, H., Adelman-McCarthy, J., Agarwal, S., Hernandez, A.V., Si, W., Vlimant, J.R.: Automatic log analysis with NLP for the CMS workflow handling. EPJ Web of Conferences **245**, 03006 (2020). https://doi.org/10.1051/epjconf/202024503006
18. Le, V., Zhang, H.: Log-based Anomaly Detection Without Log Parsing. In: 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE). pp. 492–504 (2021). https://doi.org/10.1109/ASE51524.2021.9678773, http://doi.ieeecomputersociety.org/10.1109/ASE51524.2021.9678773
19. Madkan, P.: Empirical Study of ERP Implementation Strategies-Filling Gaps between the Success and Failure of ERP Implementation Process. International Journal of Information & Computation Technology **4**(6), 633–642 (2014), http://ripublication.com/irph/ijict_spl/ijictv4n6sp_13.pdf
20. Mendix: Deploy API (4 2024), https://docs.mendix.com/apidocs-mxsdk/apidocs/deploy-api/
21. Meng, W., Liu, Y., Huang, Y., Zhang, S., Zaiter, F., Chen, B., Pei, D.: A semantic-aware representation framework for online log analysis. pp. 1–7 (2020). https://doi.org/10.1109/ICCCN49398.2020.9209707
22. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems **26** (2013)
23. Mitra, M., Sy, D.: The Rise of Elastic Stack (11 2016). https://doi.org/10.13140/RG.2.2.17596.03203
24. Münch, J., Armbrust, O., Kowalczyk, M., Soto, M.: Prescriptive Process Models. In: Münch, J., Armbrust, O., Kowalczyk, M., Soto, M. (eds.) Software Process Definition and Management, pp. 19–77. Springer Berlin Heidelberg, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-24291-5{_}2, https://doi.org/10.1007/978-3-642-24291-5_2

25. Plotly: Plotly JavaScript Open Source Graphing Library, https://plotly.com/javascript/
26. Srivastava, D.: An introduction to data visualization tools and techniques in various domains. International Journal of Computer Trends and Technology **71**, 125–130 (04 2023). https://doi.org/10.14445/22312803/IJCTT-V71I4P116
27. Trent, B.: Categorize your logs with Elasticsearch categorize_text aggregation (3 2022), https://www.elastic.co/blog/categorize-your-logs-with-the-new-elasticsearch-categorize-text-search-aggregation
28. Wang, J., Tang, Y., He, S., Zhao, C., Sharma, P.K., Alfarraj, O., Tolba, A.: LogEvent2vec: LogEvent-to-vector based anomaly detection for large-scale logs in internet of things. Sensors (Switzerland) **20**(9) (5 2020). https://doi.org/10.3390/s20092451
29. Wang, J., Zhao, C., He, S., Gu, Y., Alfarraj, O., Abugabah, A.: LogUAD: Log unsupervised anomaly detection based on word2Vec. Computer Systems Science and Engineering **41**(3), 1207–1222 (2022). https://doi.org/10.32604/csse.2022.022365
30. Wei, Y., Li, M., Xu, B.: Research on Establish an Efficient Log Analysis System with Kafka and Elastic Search. Journal of Software Engineering and Applications **10**(11), 843–853 (2017). https://doi.org/10.4236/jsea.2017.1011047
31. Yu, B., Yao, J., Fu, Q., Zhong, Z., Xie, H., Wu, Y., Ma, Y., He, P.: Deep Learning or Classical Machine Learning? An Empirical Study on Log-Based Anomaly Detection. In: Proceedings of the IEEE/ACM 46th International Conference on Software Engineering. ICSE '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3597503.3623308, https://doi.org/10.1145/3597503.3623308
32. Zamfir, V.A., Carabas, M., Carabas, C., Tapus, N.: Systems monitoring and big data analysis using the elasticsearch system. In: Proceedings - 2019 22nd International Conference on Control Systems and Computer Science, CSCS 2019. pp. 188–193. Institute of Electrical and Electronics Engineers Inc. (5 2019). https://doi.org/10.1109/CSCS.2019.00039
33. ZHAO, K., XIA, M.U.: Forming Interoperability Through Interorganizational Systems Standards. Journal of Management Information Systems **30**(4), 269–298 (2014), http://www.jstor.org/stable/43590191
34. Zhou, J., Qian, Y., Zou, Q., Liu, P., Xiang, J.: Deepsyslog: Deep anomaly detection on syslog using sentence embedding and metadata. IEEE Transactions on Information Forensics and Security (2022). https://doi.org/10.1109/TIFS.2022.3201379