A Web-based Insider Threat Dashboard for Small and Medium Enterprises (SMEs)

151421

Kirimi, Christine Mutinda

152381

Kiley, Ivy Mutanu

CNS

Supervisor: Harrison Talo

An Informatics Project Document Submitted to the School of Computing and Engineering Sciences (SCES) in partial fulfilment of the requirements for the award of a Degree in Computer Networks and Cybersecurity

School of Computing and Engineering Science

Strathmore University

Nairobi, Kenya

August 2025

Declaration

We declare that this work has not been previously submitted and approved, in whole or partial, for the award of a degree by this or any other University. To the best of our knowledge and belief, the project document contains no material previously published or written by another person except where due reference is made in the dissertation itself.

Students' Signatures

Signature: ………[illegible]. ………………… Date: ……8/18/2025………………

Signature: ……[illegible]…………………………… Date: ……8/18/2025………………

Supervisor's Approval

Signature: [illegible]………………………………………… Date: 18/08/2025……………………

Acknowledgement

We would like to express our sincere gratitude to those who have made this project possible. We would like to give special thanks to our supervisor Mr. Harrison Talo, for his encouragement, assistance and advice during the project's milestones.

We would like to thank our parents for their unwavering support during this project. We would like to acknowledge Strathmore University for providing us with the resources, platform and the assistance needed to complete this project.

Table of contents

Declaration ii

Acknowledgement iii

List of Figures viii

List of Tables ix

List of Abbreviations x

Abstract xi

Chapter 1: Introduction 1

1.1 Background Information 1

1.2 Problem Statement 2

1.3 Objectives of the Study 3

1.3.1 General objectives 3

1.3.2 Specific objectives 3

1.4 Research Questions 3

1.5 Justification 4

1.6 Limitations 4

1.7 Delimitations 5

Chapter 2: Literature Review 6

2.1 Introduction 6

2.2 Small and medium enterprises (SMEs) and Insider Threats 6

2.3 Mechanisms used by SMEs to Track Employee Access 8

2.4 Challenges in the Current Mechanisms Used for Tracking User Activity 9

2.5 Related Applications 10

2.5.1 Splunk Enterprise Security (ES) 10

2.5.2 Microsoft Defender for Endpoint 11

2.5.3 Teramind 12
2.5.4 Observe IT 13
2.6 Gaps in Current Approaches and Related Applications 14
2.7 Conceptual Framework 14
Chapter 3: Methodology 17
3.1 Introduction 17
3.2 System Methodology 17
3.2.1 Product Backlog 19
3.2.2 Sprint Planning 19
3.2.3 Sprint Execution 19
3.2.4 Sprint Review 19
3.2.5 Sprint Retrospective 20
3.3 System Analysis 20
3.3.1 Use Case Diagram 20
3.3.2 Activity Diagram 20
3.3.3 Class Diagram 21
3.4 System Design 21
3.4.1 Sequence Diagram 21
3.4.2 Entity Relationship Diagram (ERD) 21
3.4.3 System Sequence Diagram 22
3.5 System Development Tools and Technologies 22
3.6 Deliverables 23
Chapter 4: System Analysis and Design 25
4.1 Introduction 25
4.2 Requirements Analysis 25

4.2.1 Functional Requirements 25

4.2.2 Non-Functional Requirements 27

4.3 System Analysis 28

4.3.1 Use Case Diagram 28

4.3.2 System Sequence Diagram 29

4.3.3 Activity Diagram 31

4.3.4 Sequence Diagram 32

4.3.5 Class Diagram 33

4.3.6 Entity Relationship Diagram (ERD) 35

4.4 System Design 36

4.4.1 System Architecture Diagram 36

4.5 System Wireframes 37

4.5.1 Login Wireframe 37

4.5.2 Dashboard Overview Wireframe 38

4.5.3 User Management Interface 39

4.5.4 Rules Engine Interface Wireframe 40

Chapter 5: System Implementation and Testing 42

5.1 Introduction 42

5.2 Description of the Implementation Environment 42

5.2.1 Hardware Requirements 42

5.2.2 Network Requirements 43

5.2.3 Software Requirements 43

5.3 System Implementation 45

5.3.1 User Login Page 45

5.3.2 Insider Activity Monitoring Dashboard 47

5.3.3 Threat Alerts and Incident Reporting 49
5.3.4 System Administration and Configuration 49
5.3.5 Employee Dashboard (User Interaction Module) 50
5.4 System Testing 51
5.4.1 Testing Paradigm 51
5.4.2 Testing Results 52
Chapter 6: Conclusion 54
6.1 Conclusion 54
6.2 Recommendations 54
6.3 Future Works 55
References 56
Appendices 59
Appendix A: Gantt Chart 59

List of Figures

Figure 2.1 Biggest Threat Actors (Nasir et al., 2021). 7
Figure 2.2 Types of Insider Threats (Gunuganti, 2024). 8
Figure 2.3 Splunk ES Application Architecture Design (Shelke & Frantti, 2025) 10
Figure 2.4 Teramind dashboard 12
Figure 2.5 Observe IT’s Dashboard (ObserverIT Explained - Search, n.d.) 13
Figure 2.6 Conceptual Framework 16
Figure 3.1 Agile-Scrum Framework (Lance, 2025) 18
Figure 4. 1 Use-Case Diagram 29
Figure 4. 2 System Sequence Diagram 30
Figure 4. 3 Activity Diagram 32
Figure 4. 4 Sequence Diagram 33
Figure 4. 5 Class Diagram 34
Figure 4. 6 Entity Relationship Diagram (ERD) 36
Figure 4. 7 System Architecture Diagram 37
Figure 4. 8 Login Wireframe 38
Figure 4. 9 Dashboard Overview Wireframe 39
Figure 4. 10 User Management Interface 40
Figure 4. 11 Rules Engine Interface Wireframe 41
Figure 5. 1 IT administrator Login Dashboard 46
Figure 5. 2 Employee Login Dashboard 47
Figure 5. 3 Overview of the Insider Threat Monitoring Dashboard 48
Figure 5. 4 Threat Logs 48
Figure 5. 5 Alerts Generation Module 49
Figure 5. 6 User Management panel 50
Figure 5. 7 Employee Dashboard 51

List of Tables

Table 5.1 Web Application Hardware Specifications 42
Table 5. 2 Client-Side Software Requirements 43
Table 5. 3 Server-side Requirements 44
Table 5. 4 Module Test Results 52

List of Abbreviations

API – Application Programming Interface

BYOD – Bring Your Own Device

DAD - Disciplined Agile Delivery

DSDM - Dynamic Software Development Method

ERD – Entity Relationship Diagram

ORM - Object Relational Mapper

OTP – One-time password

RBAC – Role-Based Access Control

SMEs – Small and Medium Enterprises

SSG – Static Site Generation

SSR – Server-Side Rendering

TDD – Test Drive Development

UI – User Interface

XP – Extreme Programming

Abstract

Insider threats pose a significant security risk to Small and Medium Enterprises (SMEs), where limited resources often hinder the implementation of advanced monitoring systems. These threats, originating from within an organization, can lead to unauthorized data access, data breaches, and compromised operational integrity. Current security tools are often expensive, complex, or lack real-time behavioral analysis, leaving SMEs vulnerable. This project proposes the design and development of a web-based Insider Threat Monitoring Dashboard tailored for SMEs. The system is built using the Django web framework, that simulates a company file sharing system that is used to collect and analyze user activity logs, such as login times, file accesses, and download behaviors. Through a combination of rule-based logic and optional machine learning integration, the dashboard detects and flags behavioral anomalies indicative of insider threats. Key features include a secure CRUD-based user management interface, a real-time log viewer, rule-based alert generation, and visual summaries of detected threats. The system also incorporates web security mechanisms to preserve the confidentiality and integrity of sensitive log data. By offering a cost-effective, user-friendly solution, the dashboard empowers SMEs to proactively detect and respond to suspicious internal activity. This research addresses the need for accessible and intelligent internal threat monitoring tools, filling a critical gap in SME cybersecurity.

Keywords: *Insider Threats, Behavioral Anomaly Detection, Rule-Based Logic, SMEs, Cybersecurity, Django, Activity Logs, Real-Time Monitoring, Web Security*

Chapter 1: Introduction

1.1 Background Information

Small and Medium enterprises (SMEs) are composed of non-subsidiary and independent firms which employ a given number of employees. In some areas, such businesses can employ up to 200 people. In Kenya, small enterprises are known to have around 10 – 49 employees and medium businesses averagely employ 50 – 99 people (Douglas et al., 2017). While having this limited number of employees, SMEs are the backbone of economic growth and development around the world. As they are 'engines of economic growth', they pave way for various advantages such as job creation, stability and innovation. In emerging countries, SMEs account for 40% of national income while creating job opportunities. This is as SMEs fall under various categories such as wholesale and retail trade, manufacturing, food services, education, entertainment and recreation, transportation and storage (Omondi Oketch & Okeyo, 2024).

In spite of these various advantages, SMEs often lack substantial resources to invest in robust cybersecurity measures or have designated cybersecurity teams. However, these wide-spread enterprises often handle sensitive data such as personal information and financial records generated from their customers. The employees within such businesses also lack comprehensive awareness which are acquired through training programs. This can further result to insecure access credentials such as weak passwords, careless and negligent behaviour from employees who may accidentally disclose sensitive information and even high likelihoods of employees falling prey to phishing attacks or any other kind of social engineering methodologies (Fahimah Mohd Nassir et al., 2024). This tends to make SMEs alluring and prime targets for cybercrimes such as malware and ransomware which are increasing ever so rapidly.

In SMEs, these cybercrimes are regularly capacitated through insiders who are defined as "any person who has legitimate and privileged access to internal digital resources, i.e., anyone who is allowed to see or change the organization's computer settings, data, or programs in a way that arbitrary members of the public may not (Hunker et al., n.d.)."

Insiders may include full-time employees, temporary employees, volunteers, contractors or even volunteers. In other words, an insider is a member of a workforce who has access and intentionally or unintentionally exceeds and/or misuses the access in a manner which impacts the firm's confidentiality, availability and integrity negatively (Alsowail & Al-Shehari, 2022). The negative mannerisms in which an insider can use their privilege include malicious activities, incompetence or unlawful conduct.

To counter this issue, the project aims to provide research on insider threats and mitigate the risk of insider threats in Small and Medium Enterprises (SMEs). This is accomplished through the implementation of a web-based insider threat dashboard tailored to meet the needs and limitations of SMEs. This dashboard is a centralized platform to monitor user activity, detect behavioral anomalies, and enforce access control policies. This is actuated by the utilization of real-time activity logging, behavioral anomaly detection through rule-based logic and basic machine learning techniques, and role-based access control. With this solution, SMEs shall be able to enhance internal security and data protection, thereby supporting business continuity, regulatory compliance, and trust in digital operations.

1.2 Problem Statement

Due to an increase in the digital business environment, insider threats have emerged as one of the most critical and challenging cybersecurity concerns, especially in Small and Medium Enterprises (SMEs). Insider threats occur when malicious or negligent actions by individuals within an organization misuse their authorized access to compromise sensitive data, disrupt operations or cause financial harm. Despite the growing impact of such threats, most SMEs lack the technical capacity, financial resources, or dedicated tools to monitor internal user behavior effectively. As a result, many SMEs operate without sufficient visibility into employee digital behavior. Existing security solutions are often designed for large enterprises, making them too complex or costly for SMEs to adopt. Furthermore, many do not offer real-time monitoring or contextual behavioral analysis, limiting their ability to detect subtle or emerging threats. As a result, suspicious activities such as unauthorized file access, abnormal login times, and excessive data downloads often go unnoticed until damage has already been done. There is a clear need for an affordable, easy-to-use solution that can simulate SME environments, monitor internal activity, detect anomalies, and flag potential insider threats in real time.

1.3 Objectives of the Study

1.3.1 General objectives

The main objective is to design, develop, test and evaluate a web-based Insider Threat Dashboard for Small and Medium Enterprises (SMEs) which logs real time activity and detects behavioral anomalies through rule-based logic.

1.3.2 Specific objectives

i. To analyze the current insider threat detection practices and limitations within Small and Medium Enterprises (SMEs), with a focus on behavioral monitoring and anomaly detection.
ii. To investigate the core security challenges that SMEs face regarding insider threats.
iii. To design the system architecture and interface of a web-based Insider Threat Dashboard that captures, analyzes, and visualizes employee activity in a simulates SME environment.
iv. To develop a secure Django-based web application that logs user activity and applies rule-based logic for anomaly detection.
v. To test the dashboard's accuracy, usability, and performance in detecting potential insider threats through scenario-based simulations and system evaluations.

1.4 Research Questions

i. What are the current practices and limitations in insider threat detection among Small and Medium Enterprises (SMEs)?
ii. What are the key challenges and risk factors that contribute to insider threats within SMEs?
iii. How can a web-based dashboard be effectively designed to simulate an SME environment and support real-time monitoring of employee activities?
iv. What functionalities and technologies are suitable for developing a secure insider threat monitoring system?

v. How effective is the developed dashboard in accurately detecting insider threats in real-world scenarios?

1.5 Justification

Small and Medium Businesses (SMEs), frequently lack the financial and technical resources to deploy enterprise-grade threat detection systems. Therefore, insider threats continue to present a rising cybersecurity concern. Although external dangers have received a lot of attention, internal hazards are frequently overlooked even though they can do serious harm. Insiders can abuse permitted access to steal confidential information, interfere with services, or reveal vulnerabilities, regardless of whether they are malevolent, careless, or compromised.

Most SMEs cannot afford the current threat detection systems because they are usually too expensive, too complicated, and require highly skilled workers to run. The urgent need for an affordable, user-friendly, and SME-friendly solution that permits real-time monitoring of internal operations serves as justification for this project. By providing a user-friendly interface constructed with open-source technologies (like Django), rule-based behavioral anomaly detection, and visual insights catered to non-technical users, the proposed Insider Threat Dashboard closes this gap.

The dashboard allows SMEs to proactively identify irregularities by mimicking a business file sharing system and recording employee activity records, including login timestamps, file access, and download patterns. Therefore, this project enhances the security posture of SMEs as well as advancing knowledge of internal risk reduction through workable and reasonably priced solutions.

1.6 Limitations

Insider Threat Monitoring Dashboard, as useful and operative in its on-scope application, takes place under a range of limitations that need to be identified. First, the system has never been tried in a real production environment, but it is simulated. This limits its capacity to capture the unpredictability and complexity of the real-world insider's behaviors, that is, its trustworthiness in actual implementation cannot be completely ensured. Also, the dashboard predominately uses a rule-based detector, which is effective in finding pre specified patterns, but ineffective in detecting subtle or mutating insider activity without regular update and refinement. The other weakness is that the system has a limited behavioral indicator it is programmed to detect; by hinging on a limited number of scenarios in detecting insider threats, the system might fail to capture the less evident or complex insider threat behavior as it can occur in the dynamic organizations. In addition, the use of synthetic data to perform tests and validate them necessitated by privacy and ethical concerns is not as realistic as in the assessment process and might not fully capture the reality of organizational data. Finally, the system is focused only on mitigating insider threats and does not necessarily cover any external cybersecurity attacks, which include phishing, malware, or brute-force attacks, and organizations may still face the threat caused by external activities.

1.7 Delimitations

First, live production networks are not integrated into the system, which was created in a simulated business context. This restricts its capacity to accurately mimic the complexity and unpredictability of organizational behavior in the actual world. Second, the system's primary method of identifying insider threats is rule-based logic. This method may not be able to detect complicated or changing threats unless rules are regularly updated, even though it works well for established conditions.

Furthermore, the system might not cover the entire range of possible insider behaviors because the project scope limits the implementation to a limited number of behavioral indicators and threat situations. The study is also constrained by ethical and privacy concerns; all testing and demonstration data is synthetic, which may compromise the breadth and realism of behavioral analysis. Lastly, it's critical to remember that the system only addresses internal concerns; it provides no defense against external cybersecurity threats like malware, phishing scams, or illegal external access. Despite their importance, these restrictions do not lessen the project's worth; rather, they draw attention to areas that could be expanded and improved upon in the future.

Chapter 2: Literature Review

2.1 Introduction

This section of this study provides a clear definition of Small and Medium Enterprises (SMEs), their benefits and their current mechanisms of tracking user access. It also examines the challenges SMEs face while using these methods of tracking user access. It highlights some of the related applications SMEs or any other organization may utilize to counter the problem of insider threats. Consequently, it explores the gaps that these current technologies possess and concludes with the conceptual framework of the web-based dashboard which illustrates the proposed solution of insider threats.

2.2 Small and medium enterprises (SMEs) and Insider Threats

Small and medium-sized businesses (SMEs) are autonomous, non-subsidiary companies with a specific workforce size. SMEs are the foundation of global economic growth and development despite having a small workforce. Being "engines of economic growth," they open the door for several benefits, including stability, innovation, and the creation of jobs. SMEs generate jobs and contribute 40% of the national income in developing nations. This is because SMEs can be found in a variety of industries, including manufacturing, food services, education, entertainment and recreation, wholesale and retail commerce, transportation, and storage (Omondi Oketch & Okeyo, 2024).

SMEs in Kenya, according to Douglas et al. (2017), can hold around 10 to 99 employees which can be considered as a measly number, in comparison with larger organizations. For these reasons they experience budget constraints which hinder such enterprises from having experienced cyber security teams and training for the employees (Tselios et al., 2022). This may lead to weak passwords, employees acting carelessly and negligently and unintentionally disclosing private information, and even a higher chance of employees becoming victims of phishing attacks or other forms of social engineering techniques (Fahimah Mohd Nassir et al., 2024).

Insiders Threats can be defined as threats originating from malicious or accidental insiders who use their legitimate access to networks, systems, and data of a company to compromise the privacy, availability, integrity, or physical condition of the company's data, information systems, or personnel (Le & Zincir-Heywood, 2021). Malicious workers that want to damage the business through theft and vandalism are referred to as "insider threats." However, in practice, staff who are reckless and inconsiderate may unintentionally result in high impact damage (66%) (Nasir et al., 2021). Regular workers (49%) and privileged IT users (59%) are the largest insider threat actors, followed by contractors (52%) as shown in Figure 2.1 (Nasir et al., 2021).

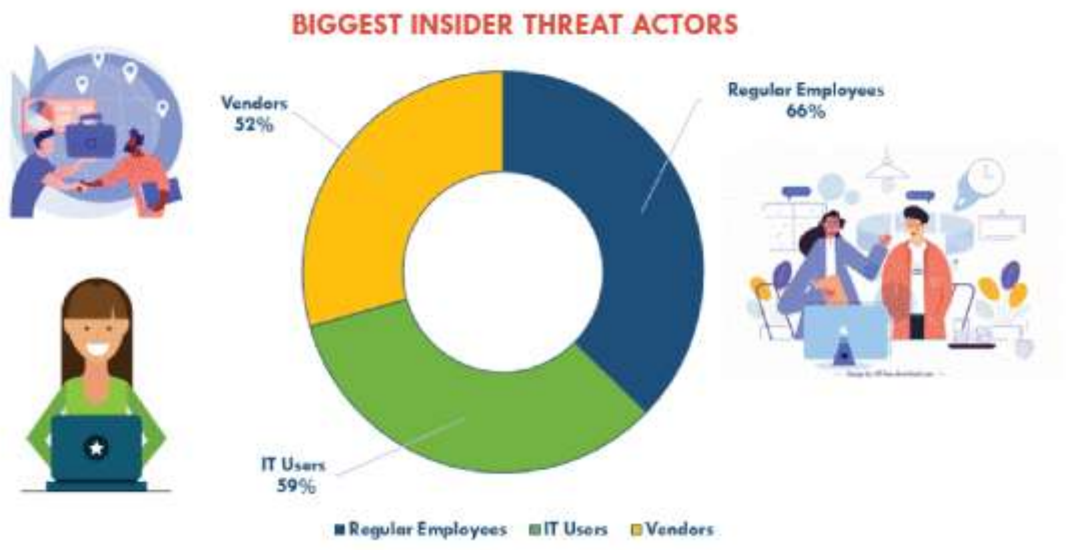


Figure 2.1 Biggest Threat Actors (Nasir et al., 2021).

Insiders may be categorized into three namely: malicious insiders, negligent insiders and compromised insiders. Malicious insiders act intentionally with the aim of using access and the privileges that come with it to impact the organization harmfully. Malicious insiders are usually motivated to use their access harmfully for reasons such as financial gain, an inclination to cause havoc or harm to the organization or to an employee or they can be driven by philosophical or political beliefs. Negligent insiders are personnel who ignore security measures and policies which often contribute to security breaches. Lastly, compromised insiders are employees whose credentials or privileges are taken advantage of by malicious individuals. The compromised insiders most likely had their credentials swindled out of them using methods such as phishing or any other social engineering attack (Gunuganti, 2024). These types of insiders can be demonstrated further using Figure 2.2.

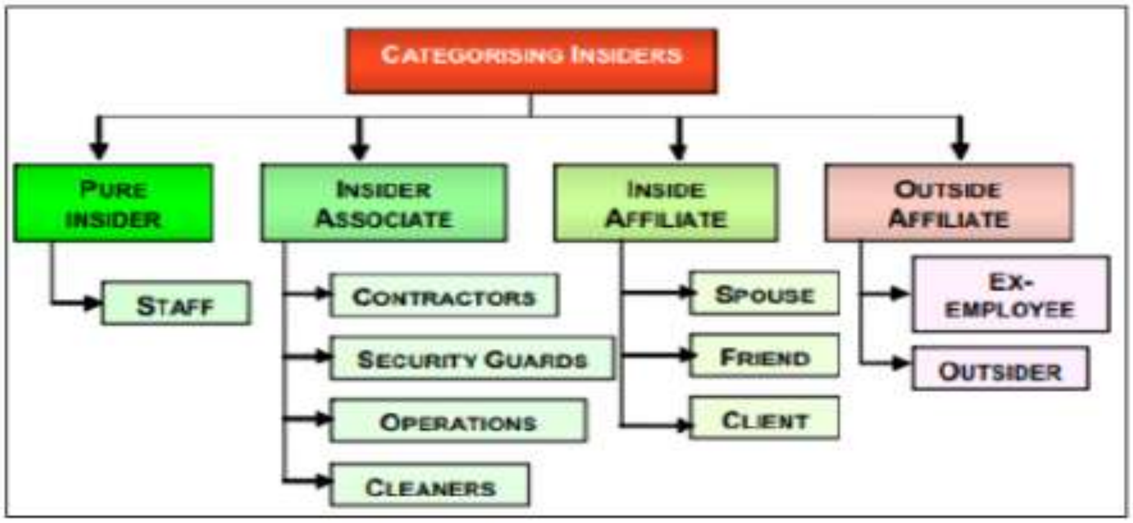


Figure 2.2 Types of Insider Threats (Gunuganti, 2024).

The insider threats mentioned above, although seen to be minor or unimportant, can have jarring effects for any organization. This is as it provides a means in which attackers can use to exploit further vulnerabilities within systems by deploying attacks such as Distributed Denial of Service (DDoS), Denial of Service (DoS), ransomware attacks, spoofing or planting any malicious software within a system. According to Al-Harrasi et al. (2023). Inside threats also pose as a big problem as they can carry out data theft which leads to further problems such as financial fraud, exfiltration attacks, theft of intellectual property and sabotage. This serves as a paramount issue especially since information technology is rapidly evolving due to creations such as artificial intelligence (AI), Internet of Things (IoT), cloud computing and big data which resulted in numerous advancements in various processes, phenomena and systems, not being limited to cybercrime (Tetteh, 2024).

2.3 Mechanisms used by SMEs to Track Employee Access

As explained through section 2.2, malicious and negligent employees need to have access to the working systems to pose an inside threat. There are various ways in which user access can be tracked. One of the most popular methods is through the use of logs. System performance metrics, login attempts, transaction histories, and other crucial information, such as error messages, are all preserved in logs, which record the history of operations in the IT environment. Logs are crucial auditing tools because they provide a trustworthy record of actions that firms may examine to verify compliance with industry standards and security procedures (Thallapally, 2025).

Another method in which SMEs can use to track access is through incorporating methods such as SaaS Monitoring. The process of centrally tracking and analyzing log data generated by SaaS applications is known as SaaS log monitoring. This involves collecting logs from a variety of SaaS infrastructure sources, including servers, apps, databases, and network devices. combining them into one platform for analysis and visualization after that. Organizations can gain insights into user behavior, security threats, application performance, and compliance issues by using SaaS log monitoring (Benson, 2024).

Finally, an additional way which SMEs use for accounting and user tracking is using Microsoft Office tools such as Microsoft Excel. Most commercial enterprises worldwide, and SMEs, now use Microsoft Excel worksheets. Although Microsoft Excel for accounting tools might be classified as manual accounting, SMEs utilize it since it is more affordable and easier to set up (Jusoh & Ahmad, 2019).

2.4 Challenges in the Current Mechanisms Used for Tracking User Activity

Tracking of user access using manual methods is a major concern as these records are easily susceptible to modification and deletion by insiders or external attackers. Also, having fragmented log sources such as spreadsheets, logs and SaaS monitoring, can hinder effective monitoring and threat detection (Thallapally, 2025). This is because there is no unified view of these logs and there is also a lack of automated anomaly detection within them. Hence suspicious activity such as users accessing the systems at unusual hours can be Mistakenly brushed off.

Another challenge that comes into light is that SMEs often lack the expertise, resources or cybersecurity-skilled workforce to implement and interpret advanced log management systems. This can lead to the underutilization of the basic tools that they already have and make SMEs prime targets for attacks that surround user access. Additionally, it is important to note that without structured and centralized logging and detention policies, SMEs may struggle to meet regulatory requirements for data protection and audit trails, which explains why SMEs are susceptible to the data breaches and attacks (Thallapally, 2025).

2.5 Related Applications

In recent years, the field of insider threat detection has seen a rise in the development of applications designed to monitor user behavior, detect anomalies, and alert administrators to potential internal security breaches.

2.5.1 Splunk Enterprise Security (ES)

Splunk ES is a popular Security Information and Event Management (SIEM) platform that gathers indexes and analyzes machine-generated data in real time. Its core functionalities include real-time monitoring of user behavior, advanced threat analytics, and customizable dashboards for visualization of anomalies and incidents. Splunk ES's modular and distributed architecture relies on components like indexers, search heads, forwarders, deployment servers, syslog server, Splunk app, Splunk web and a central license manager to collect and analyze logs from multiple data sources (Shelke & Frantti, 2025).

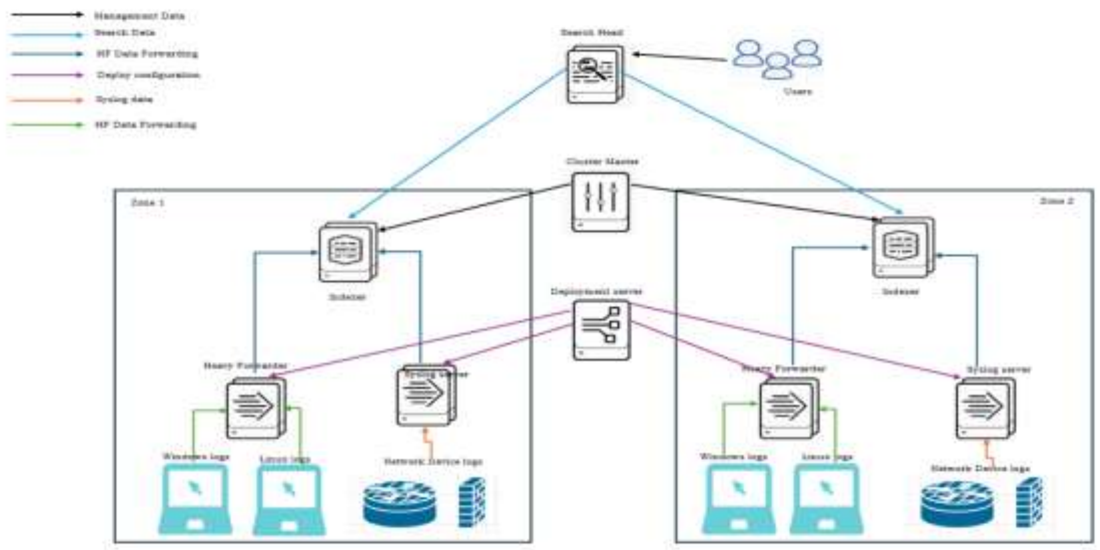


Figure 2.3 Splunk ES Application Architecture Design *(Shelke & Frantti, 2025)*

Splunk delivers seamless log collection, normalization, and correlation, offering users a centralized and cohesive view of their critical IT services. Splunk offers instant trial and seamless transition from proof of concept to production make it highly accessible and practical for testing and deployment. Additionally, the platform ensures reliability by offering a dedicated and secure environment for each customer. One of its standout features is the ability to configure custom alert triggers, which enable users to monitor data in real time and identify potential anomalies swiftly (Shelke & Frantti, 2025).

Despite the advantages of Splunk, it is costly due to the licensing fees, hardware requirements, and the technical expertise that is required for configuration and maintenance. For this reason, SMEs find it hard to use Splunk as an insider threat management tool. Additionally, the system is resource intensive.

2.5.2 Microsoft Defender for Endpoint

Microsoft Defender for Endpoint is an enterprise endpoint security platform that is designed to assist enterprise networks to prevent, detect, investigate, and respond to advanced threats (Microsoft Defender for Endpoint - Microsoft Defender for Endpoint | Microsoft Learn, n.d.). Examples of endpoints include laptops, phones, tablets, PCs, access points, routers and firewalls.

Microsoft Setinel is a SIEM tool that collects and analyzes data from users, devices, applications and infrastructure to detect and investigate cybersecurity threats and incidents. Based on log data and continuous profiling, Setinel's usage of UEBA technology allows it to compute risk scores and identify irregularities in employee behavior (Christl, 2024) .

Microsoft Purview provides a risk and a compliance functionality to monitor employee communication and search for information relating to the employees. It is embedded in Windows operating systems, and they collect and process behavioral signals from the operating system and send this sensor data to the private and isolated cloud instance of Microsoft Defender for Endpoint.

Threat intelligence reports are generated by Microsoft hunters and security teams and enhanced by threat intelligence provided by partners. Threat intelligence enables Defender for Endpoint to identify attacker tools, techniques, and procedures, and generate alerts when they're observed in collected sensor data.

2.5.3 Teramind

Teramind is an employee monitoring software with a list of advanced features that enables companies and organizations to stay compliant when faced with strict data regulations (Teramind Review: Features, Pros And Cons – Forbes Advisor, n.d.). Teramind works by collecting data on the employees by tracking employees' keystrokes, printing actions, recording employees' screens and taking screenshots. Once the data has been collected, it aggregates it into reports that help detect whether the company is compliant with the regulations and legal requirements.

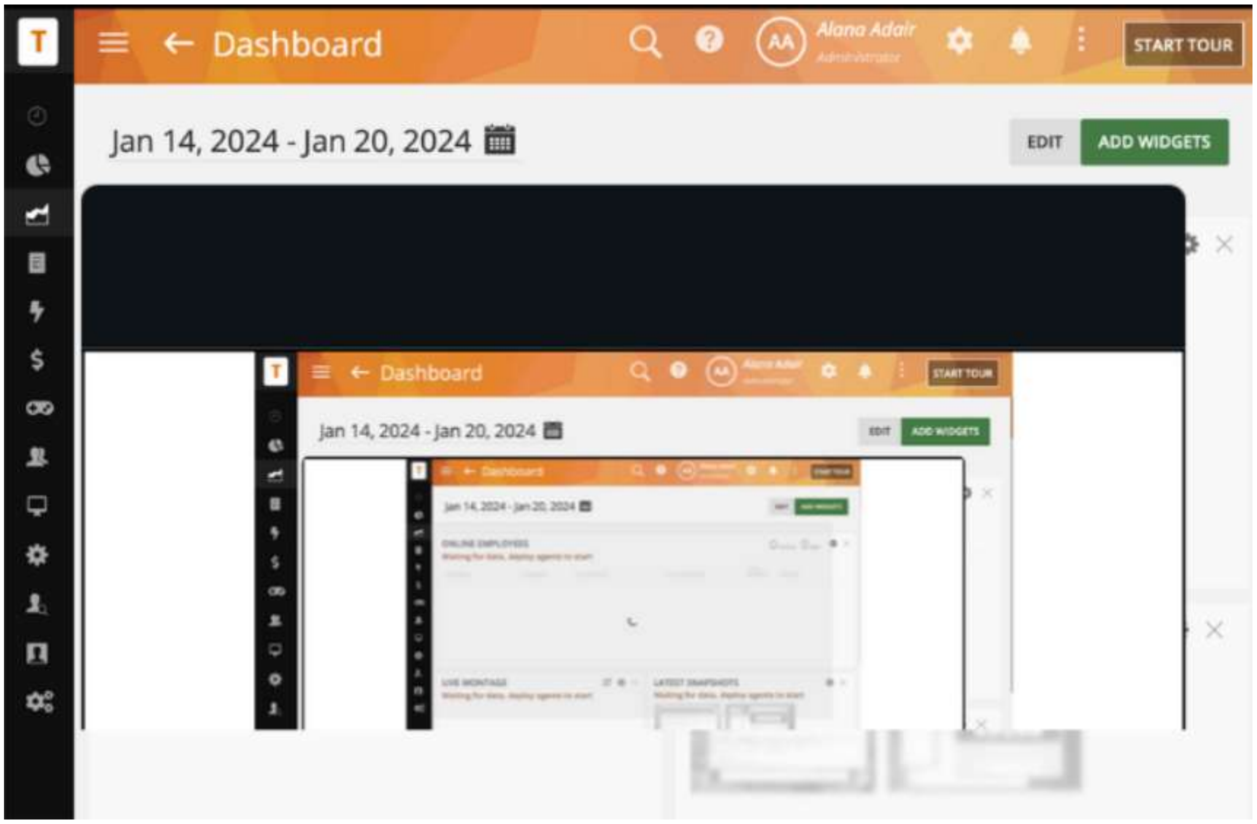


Figure 2.4 Teramind dashboard

Teramind comes with a built-in employee time tracker. It captures employees' clock-in activity and generates reports of working hours and employees' activities, task reports, productive time and cost reports (*Teramind Review: Features, Pros And Cons – Forbes Advisor*, n.d.). However, it has various shortcomings such as intrusive, high resource consumption, privacy concerns.

2.5.4 Observe IT

Observe IT enables security and risk analysts to track and monitor file activities to identify and alert on instances of data exfiltration (ObserverIT Explained - Search, n.d.) It is used to monitor both user activity and file activity, both of which are critical for detecting Insider Threats and data breaches. File activity monitoring enables organizations to track and alert when files are downloaded or exported using browsers or web-based applications, and when files are copied or moved to default local sync folders of cloud storage services (ObserverIT Explained - Search, n.d.)

ObserveIT's insider threat dashboard provides a platform for security analysts and investigators with an easy way to track users that have experienced any type of policy enforcement because of violating company policy or security rules.

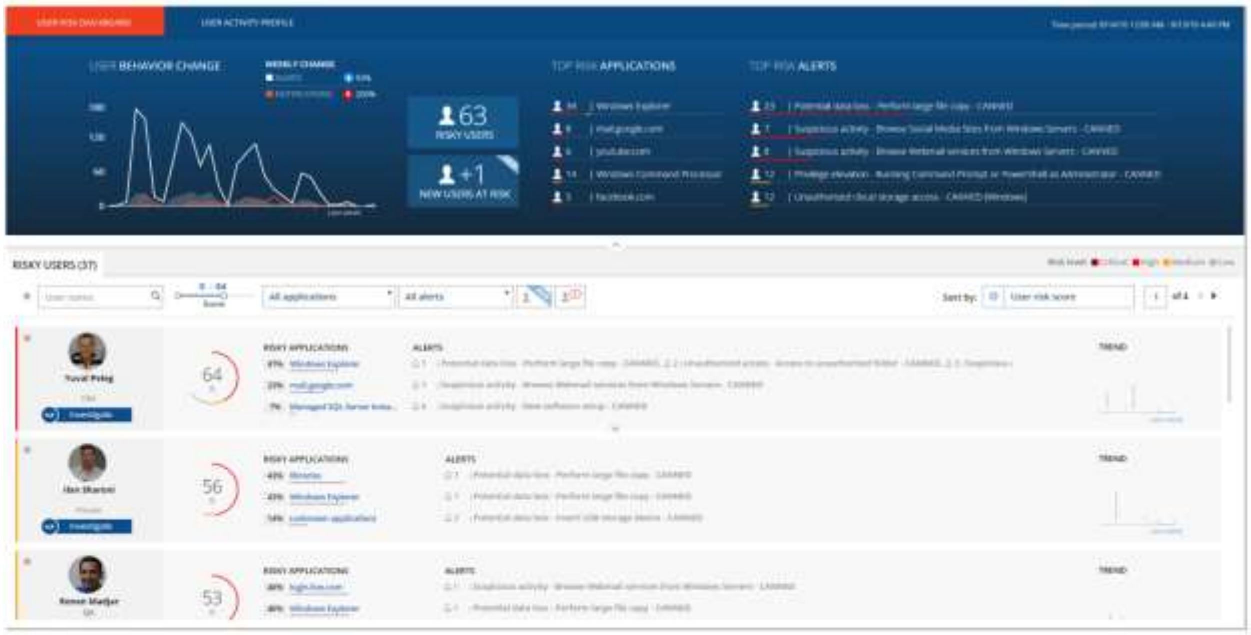


Figure 2.5 Observe IT's Dashboard (ObserverIT Explained - Search, *n.d.)*

2.6 Gaps in Current Approaches and Related Applications

Despite the presence of numerous security tools that are aimed at detecting insider threats, there is a significant gap that is left unaddressed. Most of the existing applications, such as Splunk, Microsoft Defender for Endpoint, and Teramind, are developed with large corporations in mind.

One major gap is the cost and complexity of deployment. Most insider threats systems like Splunk or Microsoft Defender are designed for large organizations. They require a large amount of funding to maintain their high IT capabilities, pay for their licensing and configuration. SMEs often lack the expertise and the budget to adopt such tools, leading to increased vulnerability in their intranets and poor internal real-time logins.

Additionally, some tools like Splunk require expertise and knowledge of scripting since they do not support flexible rule creation. This becomes a challenge for non-technical SME administrators who want to track user behavior.

Furthermore, most tools like Teramind and ObserverIT are resource intensive. This is because these applications require agents installed in every machine or device that is to be monitored. Therefore, this introduces performance issues, privacy concerns, and external administrative overhead. This becomes a challenge for organizations that use the Bring Your Own Device (BYOD) policy.

2.7 Conceptual Framework

Figure 2.6 below illustrates the conceptual framework of the Insider Threat Dashboard. The process begins at the organizational level in a simulated SME environment, where users such as employees, managers and interns interact with the system by performing actions such as logging into the company system, accessing and opening files, downloading documents, or browsing internal resources. These actions generate the real-time user activity logs, that are immediately captured and securely stored in the PostgreSQL database through Django's backend logging system. Each activity log captures metadata such as the username, activity type, file name and access location.

Once data is logged in the PostgreSQL database, the detection engine begins analyzing each entry by applying predefined behavior rules that are established by the administrator. These rules include checks for suspicious actions like logging in from multiple locations within a short span, accessing restricted files without authorization, or performing downloads that exceed the normal limits. The system continuously monitors and compares the live user behaviors against the rules to identify deviations that may indicate potential insider threats.

If any activity performed by a user is against the rules, an anomaly is detected, and the system immediately generates an alert. The alert contains information about the incident, the rules that were triggered, the employee involved, the action that caused the flag, and a severity rating. The alert is then automatically logged and made available for review within the administrative dashboard.

Through the web dashboard, the administrator views all the flagged anomalies and analyzes the trends. The administrator reviews each alert's full context by examining the employee's historical activities, the triggered value and the associated logs. Based on the review of the logs, the administrator takes actions such as revoking user access or exporting an incident report for auditing or compliance. The administrator can also use the insights from the past logs and alerts to create new rules, allowing the system to evolve with the changing threat patterns. Finally, the system logs, alerts, and user actions are securely stored to maintain a complete audit trail for future reference or legal review.

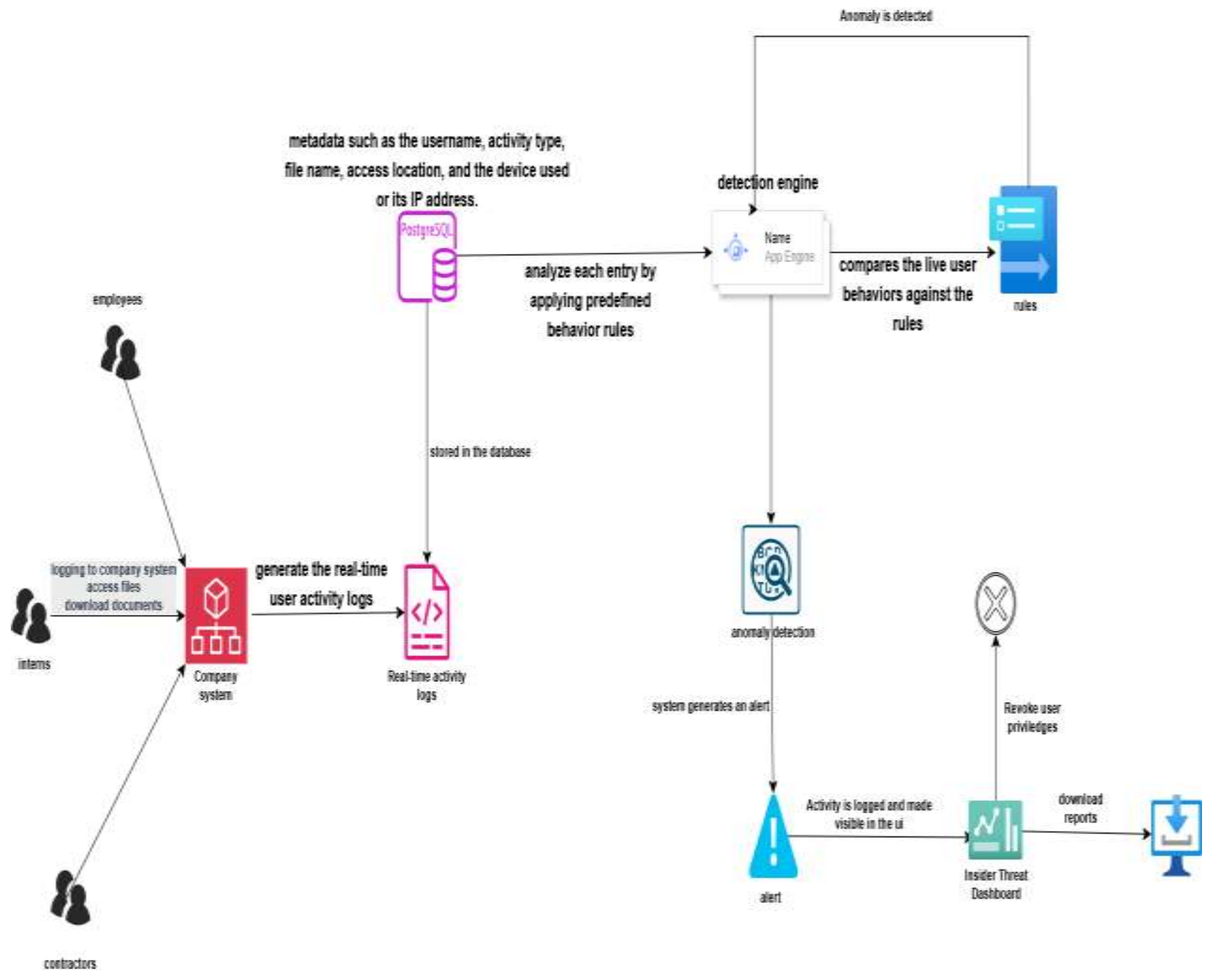

Anomaly is detected
metadata such as the username, activity type, file name, access location, and the device used or its IP address.
PostgreSQL
detection engine
Name
App Engine
analyze each entry by applying predefined behavior rules
compares the live user behaviors against the rules
rules
employees
stored in the database
logging to company system
access files
download documents
interns
generate the real-time user activity logs
Company system
Real-time activity logs
anomaly detection
system generates an alert
Revoke user priviledges
Activity is logged and made visible in the ui
download reports
alert
Insider Threat Dashboard
contractors

Figure 2.6 Conceptual Framework

Chapter 3: Methodology

3.1 Introduction

This chapter presents the methodology used in the development of the proposed Insider Threat Monitoring Dashboard for Small and Medium Enterprises (SMEs). The methodology that was proposed to be assimilated in this project is an Agile methodology known as Scrum. Therefore, this section of this study discusses the proposed methodology and the reasons why it is justified to be used within this project. It also highlights the tools and technologies that were used to actualize the web-based insider threat dashboard together with the deliverables this study produced at the end.

3.2 System Methodology

This system adapts Agile methodologies to ensure iterative development, continuous feedback, and adaptability to evolving requirements. Agile methods are a collection of progressive, self-organizing, emergent, iterative software development procedures. Scrum, Kanban, Extreme Programming (XP), Dynamic Software Development Method (DSDM), Test Drive Development (TDD), and Disciplined Agile Delivery (DAD) are some of the more well-known agile approaches (Tavares et al., 2021).

Among the many agile methodologies that are present, this project was designed, developed and tested using the Scrum framework. This methodology is well suited for the study as it allows flexibility and adaptability, which are key elements for the proposed solution (Sassa et al., 2023). This is as cyber security threats and SME requirements change rapidly. Therefore, this methodology can allow for the adaptation of new insights, emerging threats and user feedback through short development cycles called sprints. Scrum defines a sprint as a set period, usually one month or less, during which ideas are transformed into useful products (Lance, 2025). This ensures that the solution remains relevant and effective throughout the project life cycle.

The sprints within Agile methodologies enable the project to be broken down into smaller, and manageable increments. This enabled the project to deliver on some functional components such as user activity logging and rule-based detection early. It also enhances collaboration as agile methodologies were adopted within project-based teams. Consequently, this improves the usability and effectiveness of the tool proposed by this project (Sassa et al., 2023).

The agile process, Scrum usually follows iterative cycles and the following key phases as shown in Figure 3.1. The Agile-Scrum process comprises of key phases. These key phases include Product Backlog, Sprint Planning, Sprint Execution, Sprint Review and Sprint Retrospective. Facilitating the Scrum process is the responsibility of the Scrum master. They ensure that everyone is well-versed in Scrum principles and provide instruction and direction when needed. At the conclusion of each Sprint, a team of experts known as a Scrum development team is in charge of producing a releasable increment of "done." The business or clientele is represented by the product owner. They are there to make sure the other Scrum team members don't lose sight of the sprint's goal (Lance, 2025). In this study, the developers act as the Scrum master and the Scrum development team.

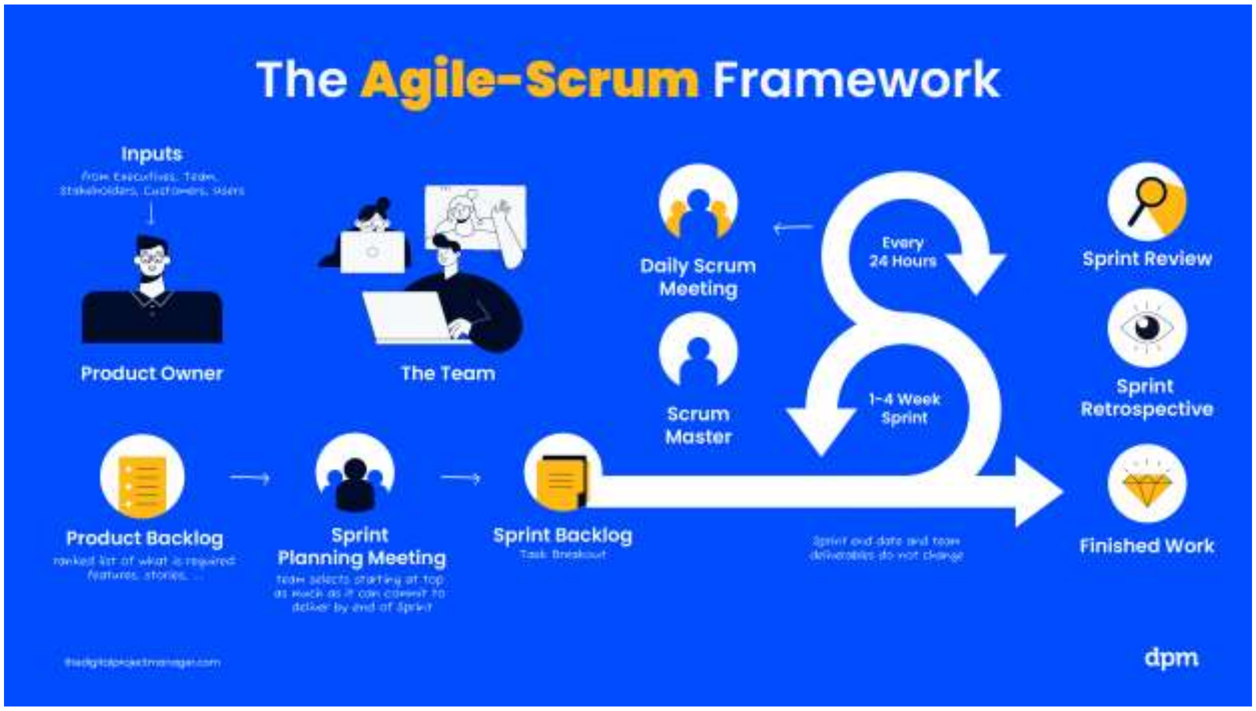

The Agile-Scrum Framework
Inputs
from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Daily Scrum Meeting
Scrum Master
Every 24 Hours
1-4 Week Sprint
Sprint Review
Sprint Retrospective
Finished Work
Product Backlog
ranked list of what is required: features, stories, ...
Sprint Planning Meeting
team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Backlog
Task Breakout
Sprint end date and team deliverables do not change
thedigitalprojectmanager.com
dpm

Figure 3.1 Agile-Scrum Framework (Lance, 2025)

3.2.1 Product Backlog

This is the phase in the developmental process which includes the listing if all features, requirements and functions that should be included within the end product. The list of features, requirements and functions are done in order of importance. The product backlog can change and be updated over time due to market changes such as, in this project's scope, cyber security threats and SME requirements (Lance, 2025).

3.2.2 Sprint Planning

In general, this stage includes the team deciding on the set of product backlog tasks they finish in the next sprint during a sprint planning meeting(Lance, 2025). In this project when this stage arises, the Scrum team establishes sprint objectives and rank features like dashboard visualizations, anomaly detection rules, logging systems, and user authentication. Team members were given tasks to complete during the sprint, which typically lasts two to three weeks.

3.2.3 Sprint Execution

In this stage, development takes place over a fixed period of time for a singles Sprint with regular daily check-ins for progress. These sessions, called daily Scrum meetings, are held to allow team members to talk about their accomplishments and challenges in order to celebrate achievements, coordinate ongoing work, and quickly remove obstacles to keep everyone moving forward (Lance, 2025).

3.2.4 Sprint Review

At the conclusion of a sprint, a sprint review meeting is held with the specific goal of reviewing the progress that has been made. The product owner evaluates if the outcomes fulfill the goals established at the sprint planning meeting during a sprint review. Here, the product owner confirms that the work increment satisfies the completed definition. The feedback can later be used to make adjustments the product backlog for upcoming sprints (Lance, 2025).

3.2.5 Sprint Retrospective

The team reflects on previous events and circumstances at a sprint retrospective meeting. An "opportunity for the Scrum team to assess itself and build a plan for changes to be performed during the following sprint," the sprint retrospective is described in the Scrum Guide (Lance, 2025). In order to promote continuous process improvement, it is utilized in this project to help the team reflect on the sprint process and determine what worked and what needs improvement. The cycle then repeats itself, gradually creating a dashboard that is more robust and feature-rich until the project is finished.

3.3 System Analysis

This section outlines the key diagrams that are used to analyze the functional and behavioral aspects of the proposed system.

3.3.1 Use Case Diagram

This diagram describes what activities are carried out by the systems from an outside observation point of view(Sutabri, 2023). It outlines how each actor interacts with the system through logging in, viewing user behavior logs, creating or editing detection rules, managing user access, and responding to system-generated alerts. For example, a system administrator may log in, view user behavior logs, define detection rules, and respond to alerts, while a security officer may only have access to view alerts and generate reports.

3.3.2 Activity Diagram

The activity diagram is a representation of the process flows in Insider Threat Monitoring Dashboard and depicts the paths of triggering various activities and connecting this activity to others. It starts with one of the users (employee or administrator) having logged in to the system and then his/her authentication takes place. Employees can view the available data, request access, or undertake mandated activities, and administrators keep a track of activities, assign authorization and manage users. Decisions are presented as nodes in the diagram, e.g. whether a set of login credentials are correct or whether a resource should be made accessible. This gives a distinctive visual image of the way the system approaches interaction with users, making decisions as well as work transition across tasks to make sure that the working process is secure and productive.

3.3.3 Class Diagram

The class diagram specifies the structural design of the system in the form of the principal entities and their attributes and relationships with others. The significant classes are described as the User having such attributes as the user_id, email, password, role and the Department where the users are organized in certain functional groups. The group classes are further divided into several categories of user roles including that of interns, regular members of the staff or managers whereas the Resource classes regulate files, access rights and ownership. The Access Log type captures specific events of the system, and the actions of the user are recorded to monitor and audit. This diagram gives a fixed model of data structure and relationship, which frames the backbone of the functional structure of the systems and how data relates in terms of security enforcement.

3.4 System Design

System design defines the architecture and components that make up the proposed solution.

3.4.1 Sequence Diagram

The sequence diagram describes how a specific process flows over time, such as when a user logs in, accesses a file, and triggers a behavior alert. The sequence diagram describes how objects interact in each situation. An important characteristic of a sequence diagram is that time passes from top to bottom and the interaction starts near the top of the diagram and ends at the bottom (Acharya, 2024). This diagram shows exactly how the dashboard responds in real-time to both normal and suspicious activities and help verify that all components communicate as intended under time-sensitive scenarios.

3.4.2 Entity Relationship Diagram (ERD)

The ERD shows a detailed representation of the attributes associated with each entity, encompassing crucial information such as admin ID, username, and password (Asrin et al., n.d.). It provides the foundation for database creation in PostgreSQL and ensures data consistency.

3.4.3 System Sequence Diagram

This diagram combines elements of use cases and logic to focus on how external actors interact with the system's backend logic during their specific operations. For instance, when an admin initiates an "Analyze User Behavior" request, the system fetches recent logs, applies behavior rules, and updates the alert panel in sequence.

3.5 System Development Tools and Technologies

The development of the web-based Insider Threat Monitoring Dashboard required a well-selected stack of tools, frameworks, and platforms that support secure, scalable, and maintainable application development. Below is a description of the core technologies and tools that were used across the frontend, backend, database, logging mechanism, development environment and the testing layers of the system.

The user interface of the system was developed using Next.js, which is a powerful React-based web development framework. Next.js supports server-side rendering (SSR), static site generation (SSG), and API routing, making it ideal for building high-performance web dashboards with real-time capabilities. Next.js supports both dynamic and static pages that ensure flexibility and fast loading times between pages, which is critical for administrators who need access to threat alerts and logs without delay. Material-UI was used in making the user interface appealing and clean for SMEs.

The backend logic of the system was developed using Django web framework. Django is efficient for its robust security features, built-in admin panel, scalability, and supports raid development. Django manages backend functions such as user authentication, log collection, alert rule management, and API endpoints for communication with the Next.js frontend. Django has an Object Relational Mapper (ORM) that simplifies interactions with the PostgreSQL database, thus allowing quick development and clear data handling logic for logs, employees and detection rules.

The system stores all the persistent data including user credentials, behavior logs, system rules and alerts, in a PostgreSQL database. PostgreSQL is an open-source relational database that is known for its robustness, support for complex queries, and strong data integrity enforcement. The schema of the database was optimized using relationships defined in Django's models and visualized in the ERD.

To simulate the insider activities and test the behavioral rules, custom Python Scripts were developed to mimic SME employee actions like login attempts, file access, and download behavior. These scripts generated synthetic logs that are fed into the system to test the dashboard's detection engine.

Git was used for Version control and GitHub for code hosting and collaboration. Git ensures that each interaction of the system is tracked, allowing easy rollbacks in case of issues. Postman was be used to test RESTful APIs between the Django backend logic and the Next.js frontend. Additionally, pytest was used to test Python modules, including detection rules and alert generation logic. Finally, the entire development is done on Visual Studio Code (VS Code).

3.6 Deliverables

First, the core deliverable is a fully functional web-based Insider Threat Monitoring Dashboard built with a Next.js frontend and a Django backend. This dashboard allows system administrators and security officers within SMEs to monitor real-time user activity logs, define detection rules, view alerts generated from behavioral anomalies, and manage user access and permissions. The system simulates insider threats based on real-world behavior patterns and provides an intuitive interface for threat monitoring.

Secondly, the project includes a custom-built rule engine that analyzes activity logs and flags insider threats based on predefined logic, such as unusual login times or large unauthorized file downloads. Anomaly detection logic is modular, allowing rules to be edited, activated, or deactivated via the admin dashboard.

In addition, a relational database schema implemented using PostgreSQL serves as the foundation for secure data storage. It holds data such as employee activity logs, alert records, and user roles. The schema design is optimized for fast querying and visualized through an Entity Relationship Diagram (ERD) that shows all core data relationships within the system.

The project delivers a complete set of system analysis and design diagrams, including a Use Case Diagram, a Sequence Diagram, a System Sequence Diagram, and an Entity Relationship Diagram. These diagrams are presented in Chapter four and serve as documentation for the logical and architectural structure of the system.

Another critical deliverable is a technical documentation report that includes system setup instructions, database configuration, system architecture explanation, and codebase walkthroughs. This report ensures that future developers or SME IT teams can install, maintain, and update the system independently.

Lastly, the project includes a working codebase hosted on GitHub, with clearly commented source code, a README.md for setup, and version history managed through Git.

Chapter 4: System Analysis and Design

4.1 Introduction

This chapter presents the system design and analysis of the Insider Threat Monitoring Dashboard. It outlines the use case diagram, the system sequence diagram, the sequence diagram, the activity diagram and the description of the components and interactions between the employees, IT administrators and the system. The goal is to offer a comprehensive blueprint for the implementation of the system by describing the user interactions, internal logic, and the structure of the solution.

4.2 Requirements Analysis

The Requirements Analysis is an important process in the systems development that involves identifying the needs, expectations, and the constraints of the system stakeholders. The requirements are categorized into functional and non-functional requirements.

4.2.1 Functional Requirements

Functional Requirements define the specific behavior or functions of the system by describing what the system must do to fulfill the needs of the user. The following are the functional requirements in the Insider Threat Monitoring Dashboard.

First authentication is an important aspect of the system. The system allows employees and IT administrators to log in using a combination of email, password and OTP for added security. Upon logging in, the system verifies the credentials and grants access to employees or IT administrators based on their roles. This ensures that only authenticated administrators and employees have access to the organization's resources.

Using Role-Based Access Control (RBAC), roles, like Admin, and Employee, are assigned specific privileges. Admins have full access to manage employees, define rules, respond to alerts, view logs, alerts, and generate reports. Employees have no direct access to the system interface; instead, their activities are logged and monitored in the background. Policies are implemented to different authorization levels to define the roles and responsibilities of the IT administrator, Security officer and the employees.

Through the system's activity logging function, the system monitors and logs on to user activities as they interact with the system in real time. The activities include login and logout events, file access, and configuration changes. Each user activity generates a log which has a timestamp, and it is associated with a user's unique identifier.

The system generates alerts that are automatically created in situations where the action of a particular user contravenes the set detection policies. These policies include uploading files past business hours, a failed log-in mismatch or an incorrect authorization to access some sensitive files. The alerts generated are documented and channeled based on their levels hence the admin can prioritize the responses.

In response to alerts, administrators operate in diverse manners. These include giving credence to the alert, or the immediate revocation of the user privileges to further devastate it. It is also recorded when the action is done along with reason and the time of action since it is audited.

Furthermore, behavioral rule management system offers a platform where administrators can define, edit or delete different rules of detection. They are necessary rules to detect suspicious user behavior due to known threats. The ability of administrators to turn off or turn on rules dynamically ensures flexibility and responsiveness to the changing threats.

The system enables admins to produce structured reports that they can use to facilitate the audit. Reporting capabilities include an activity summary of the systems, user activity trend, alerts detected, and actions implemented. The reports are exported in such a format as PDF to share and document.

The dashboard provides a view of the system status; example, number of employees accessing the system, what time they log in, summary of the alerts, and the real time alerts of threats. This enables Admins to make decent decisions in a short time and detect anomalies when they arise.

The system has a filtering system to enable the user to search for the activity logs using the username, time range, IP address and the type of event. This allows searching effectively and facilitates quick locating of critical incidents.

4.2.2 Non-Functional Requirements

The proposed Insider Threat Monitoring Dashboard must meet several non-functional requirements to ensure its effectiveness, reliability, and suitability for use within small and medium enterprises (SMEs).

First, in terms of performance, the system must be capable of logging user activity with minimal latency. Anomaly detection processes must be triggered in near real-time, with detection results returned within five seconds of a new log entry. This ensures that suspicious activity is identified quickly enough to take appropriate action.

Scalability is also crucial. Although the system is designed for SMEs, it must support up to 100 concurrent employees without significant degradation in responsiveness or throughput. Additionally, the architecture should be modular and extensible, allowing for new detection rules or monitoring components to be easily integrated in the future.

In terms of usability, the dashboard must be intuitive and user-friendly, allowing SME administrators to understand and operate the system with minimal training. Alerts are clearly visualized using color-coded severity levels (e.g., red for high-risk behaviors, yellow for moderate anomalies).

The system must also demonstrate high reliability, with an expected uptime of 99% under normal operating conditions. Even in the event of component failures, such as the temporary unavailability of the machine learning anomaly detector, the system must continue basic logging and provide manual alerting capabilities to ensure operational continuity.

Maintainability is another key factor. The backend codebase follows Python's PEP8 standards and Django framework conventions, ensuring clarity and consistency. System logs and alerts must be structured in a way that supports easy debugging and maintenance. The modular design facilitates future enhancements or replacement of specific components without the need for a complete overhaul.

4.3 System Analysis

This section illustrates various diagrams that are used to illustrate the interactions between IT administrators, employees (actors) and the system, the flow of activities, and the internal responses to user actions. These diagrams help in the identification of system functionalities, user roles, and the behavior of the system under different conditions.

4.3.1 Use Case Diagram

Figure 4.1 illustrates the use case diagram which expresses the top-level functional interactions between system, the employees and IT administrators (actors). It determines what operation employees can execute and the response by the system towards that operation. Regarding the Insider Threat Monitoring Dashboard, three actors were recognized:

The Administrator can access all the functionality of the system such as the management of the user, capacity to set rules, response to alerts, creation of reports, and monitoring the dashboard.

The employee does not interact actively with the system. Their actions are observed and recorded however they do not deal with the dashboard directly.

Using this diagram, one locates all functional parts of the system and responsibilities related to it, so that the development process meets the expectations of the employees and IT administrators and role-based access control would be properly applied.

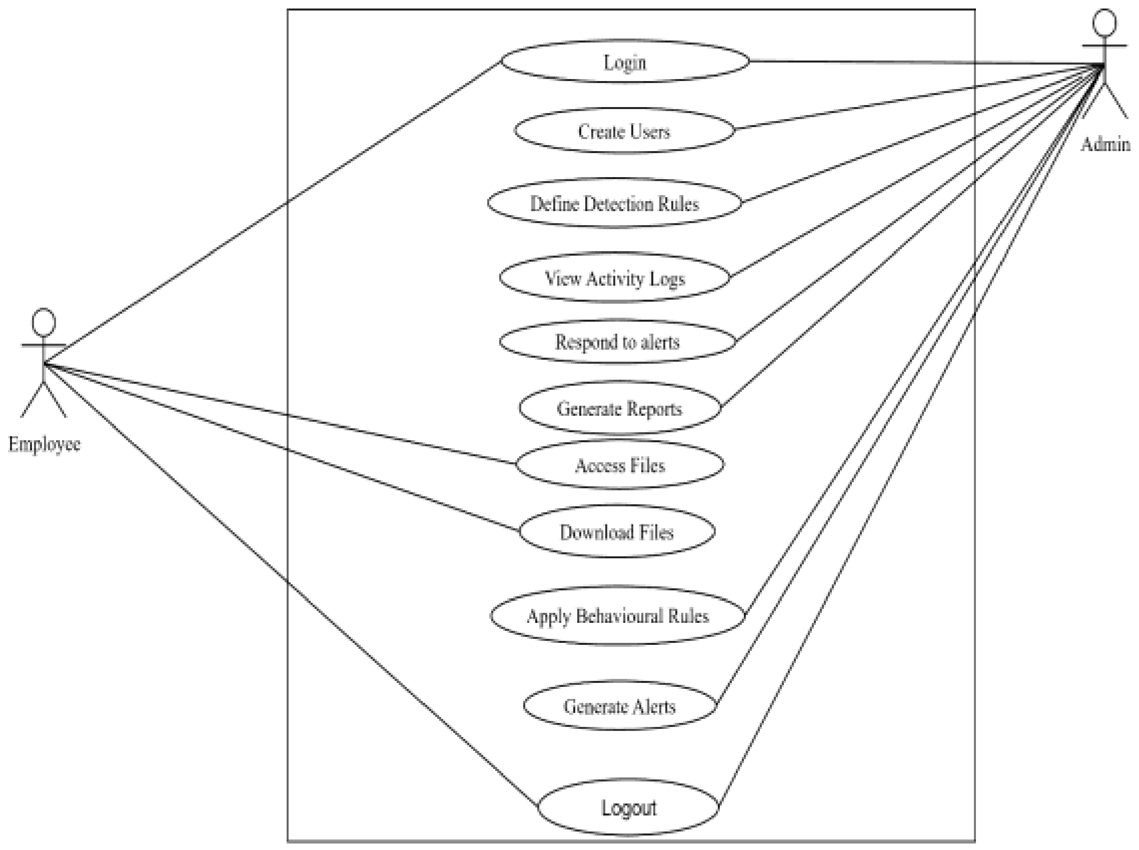


Figure 4. 1 Use-Case Diagram

4.3.2 System Sequence Diagram

Figure 4.2 illustrates the System Sequence Diagram (SSD) that demonstrates how actors and the system interact over time regarding a particular use case. In this project, SSDs were designed to simulate some important workflows of the system including Admin receiving receiver and Admin editing detection rules. The Admin Responds to Alert use case starts with logging the admin into the system and viewing of the alerts. The system sends back a list of alerts which have metadata associated with it like the type of alert, severity level and time stamp. When the Admin clicks a certain alert, they can also respond accordingly: acknowledge, escalate, delete, or withdraw access. This response is then logged together with a timestamp and the status of the alert is updated. This list makes sure that the critical incidents are dealt with in an efficient manner and records everything being done to be audited in the future. On the Manage Detection Rules sequence, the admin clicks on the detection rules module and posts a new rule or updates the existing one. The system checks input, stores the rules and it is instantly withdrawn to the activity monitoring engine. This helps guarantee that the system is always using the latest behavior policies to perform their threat detection operation, thus increasing its accuracy. The system sequence diagrams help break down the system in such a way that each process is easy to understand and work upon and this enables one to create a structured backend workflow, by merely modeling these sequences.

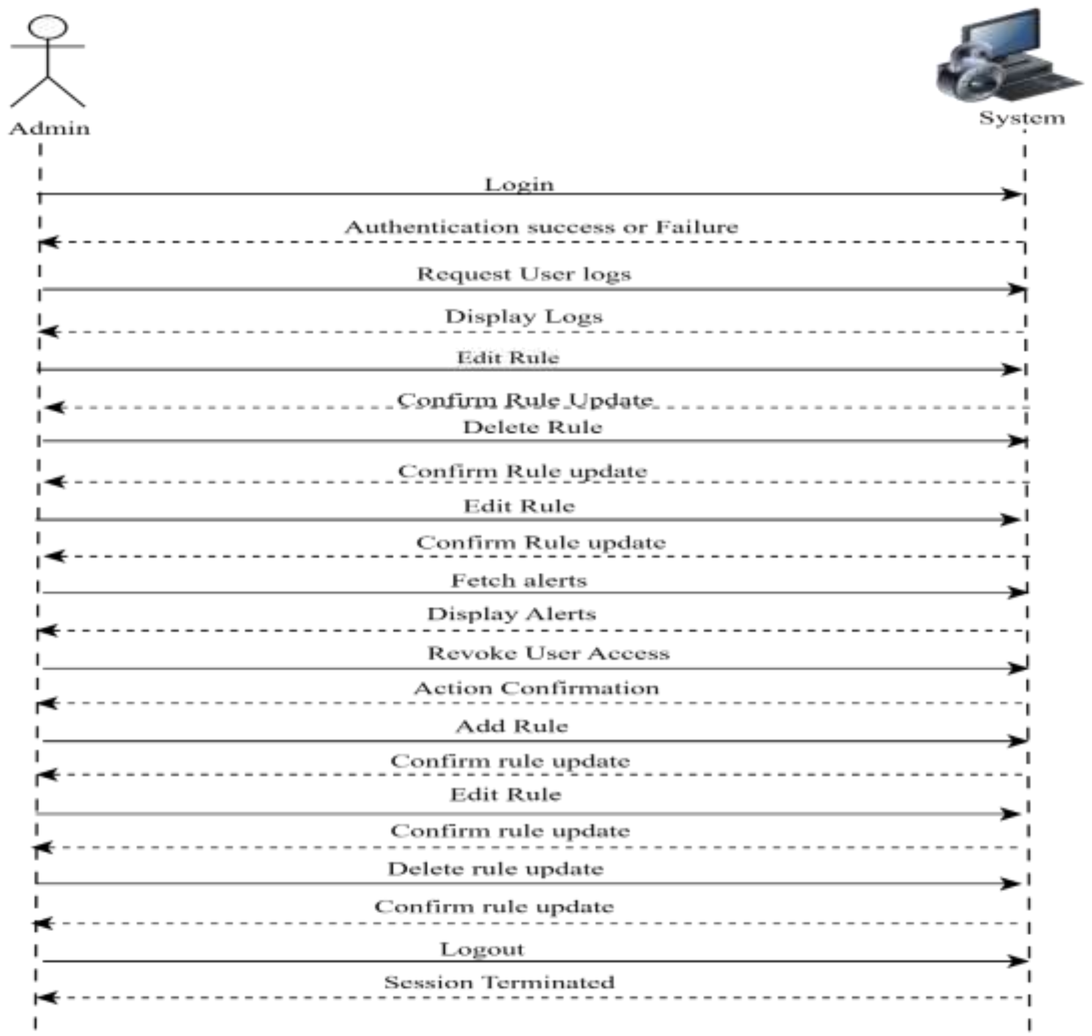


Figure 4. 2 System Sequence Diagram

4.3.3 Activity Diagram

Figure 4.3 illustrates the Activity Diagram which is Activity diagram is a graphically visible illustration of the workflow that an Admin would take when using the system. It represents the flow of control among one activity to another considering decision points, parallel activities and the possible outcomes. In the given project, the activity diagram is concerned with the relationships between the admin after the authorization.

After the authentication process, an Admin is given access to a dashboard that contains various options that include looking at logs, user management, responding to alerts, detection rules management, and even generating a report. The activity has decision-making nodes that represent the decisions made by the admin at any given stage.

Loops are also represented in the diagram and are an aspect of the nature of administrative actions which is an iterating one thus the admin can get back to the main menu every time an action is completed, and an infinite number of actions can be carried out one after the other. An option to log out is present anytime, and it ceases the session and halts the flow of the activity.

This activity diagram assists in visioning how the decisions making process is going to appear and it is also going to make sure that the system that would be designed would be intuitive, it would also be user-friendly and allow administrators to navigate easily.

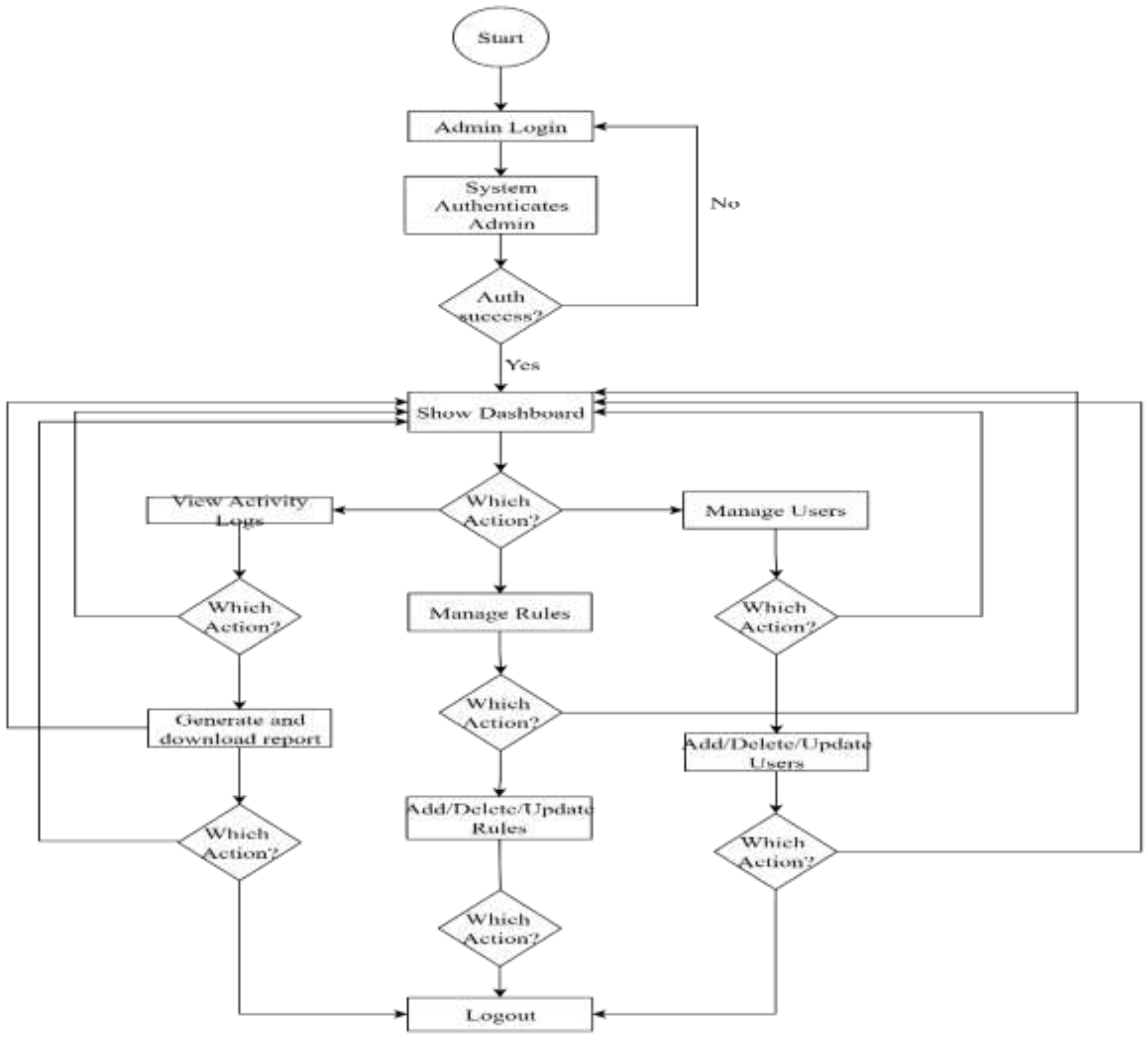


Figure 4. 3 Activity Diagram

4.3.4 Sequence Diagram

The sequence diagram provides a detailed representation of the interactions between various components of the Insider Threat Detection Dashboard during a typical user session. The flow begins when a user initiates a login request through the user interface. This request is forwarded to the authentication module, which in turn queries the database to validate the user's credentials. Upon successful authentication, a confirmation response is returned to the interface, granting the user access to the system.

Once logged in, the user begins interacting with the system. These actions are continuously monitored and captured by the database module. The logs are then forwarded to the detection module, which performs real-time analysis using a rule engine or a machine learning model to identify potential threats. If a suspicious activity is flagged, the policy enforcement module evaluates it against predefined rules and takes the necessary action, which include restricting access or elevating the event for administrative review. An alert is then generated and sent to the administrator, who can further investigate the issue. Periodically, the system also generates reports that summarize detected events and user behavior patterns. This sequence highlights the real-time nature of the threat detection system and its ability to respond dynamically to potential internal risks.

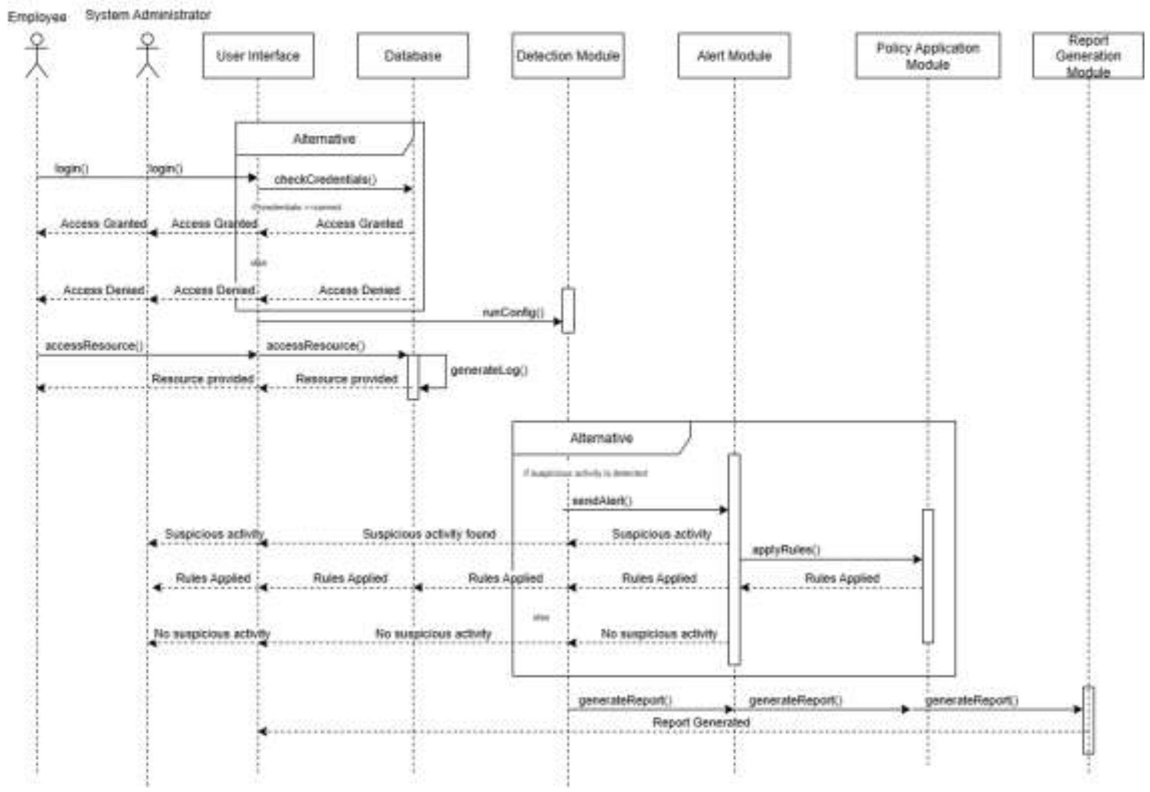


Figure 4. 4 Sequence Diagram

4.3.5 Class Diagram

The class diagram for the insider threat detection system is composed of several key entities that represent the structure and behavior of the system. At the core is the Employee class, which holds information about each user of the system, including employee_id (primary key), username, email, password_hash, role (such as Admin or Staff), and created_at timestamp. The ActivityLog class captures actions performed by employees, linking to the Employee class through a foreign key employee_id, and includes attributes such as activity_id, action_type (e.g., Read, Write, Download), resource, timestamp, and a success flag. An Anomaly class is connected to the ActivityLog and identifies unusual behavior through anomaly_id, detection_method (ML or rule-based), severity, description, and detected_at timestamp. The Alert class represents security notifications generated from detected anomalies and includes attributes like alert_id, anomaly_id, employee_id, status (Open, Investigating, Resolved), created_at, and resolved_at. Finally, an AccessPolicy class is defined to enforce role-based access control, containing attributes such as policy_id, role, resource, and allowed_actions to determine what resources each employee role can access and modify. The diagram illustrates the relationships between these classes, such as one-to-many between Employee and ActivityLog, and one-to-one or one-to-many between ActivityLog and Anomaly, enabling a clear visualization of how internal activity is monitored and managed within the system.

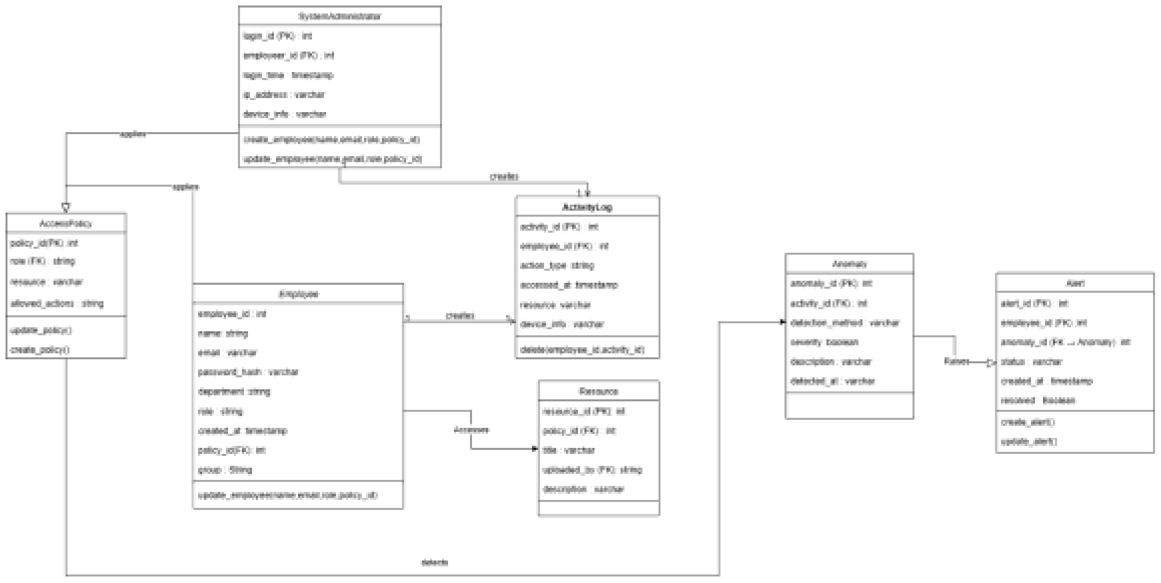


Figure 4. 5 Class Diagram

4.3.6 Entity Relationship Diagram (ERD)

The diagram below, Figure 4.6, can be considered as a representation of the logical structure of the database of Insider Threat Monitoring Dashboard, with the main entities (Employees, Departments, User Groups, Resources, Access Logs, and Threat Alerts) displayed with their connection between one another. Resources refer to digital assets that may be uploaded, allocated and accessed by designated users whereas employees are linked to Departments and differentiated according to the job category they find themselves in, i.e., Interns, Regular Staff, or Managers. Of particular interest are the relationships depicted by the ERD to explain how access activities are recorded and are linked to users and resources to make it practical in monitoring insider activity. It also shows how alerts are prompted out of suspicious access patterns so that administrators monitor and analyze and curb any potential threats in the system.

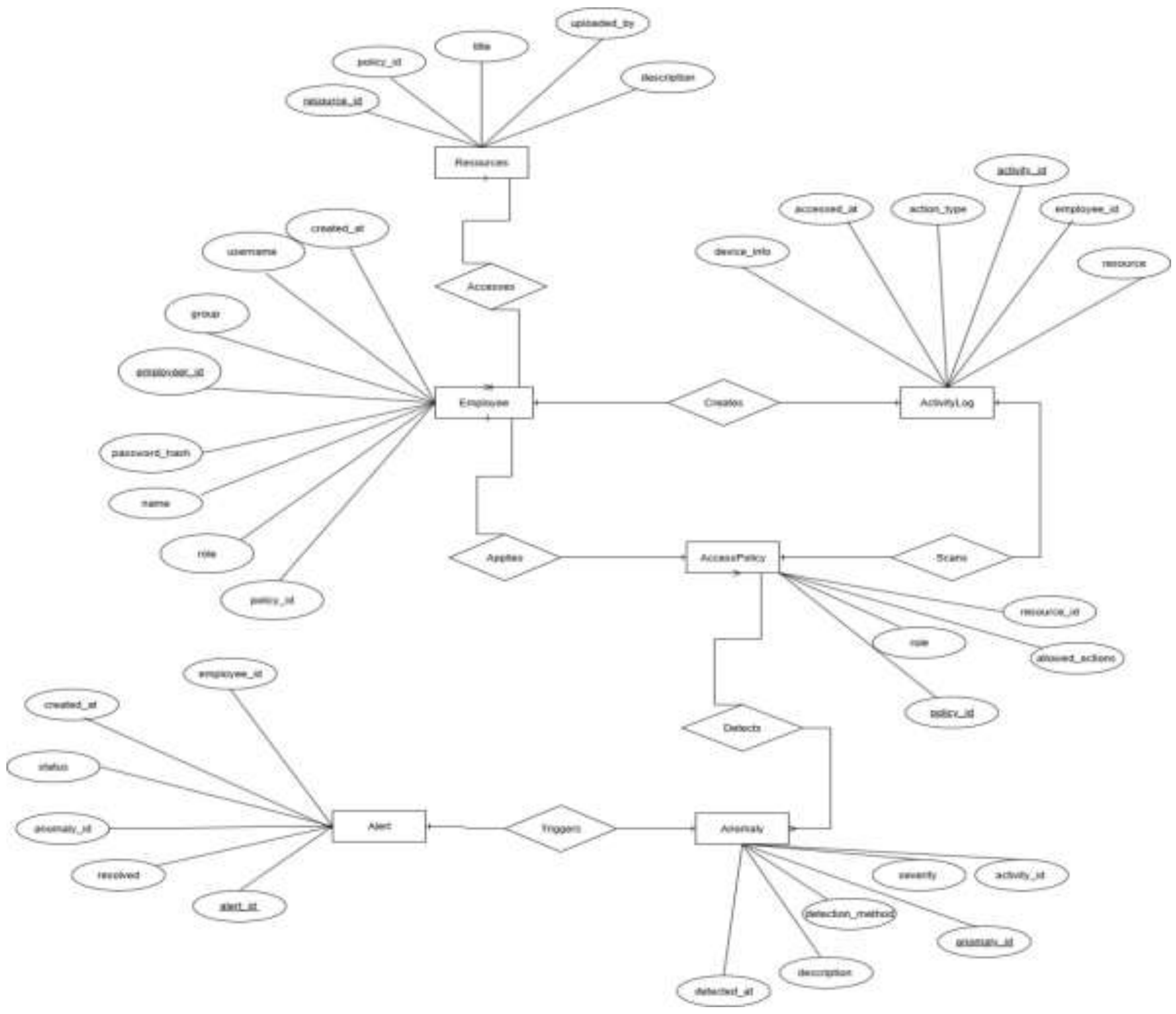


Figure 4. 6 Entity Relationship Diagram (ERD)

4.4 System Design

4.4.1 System Architecture Diagram

The architecture of the system layers is divided into three major layers, namely, the client side, the application layer and the data and security layer. At the client level, the employees maintain contact with the system on a secure web-based dashboard where they can log in, and access resources and work on the tasks assigned to them. A Django backend is used to run an application layer that authenticates, institutes access control policies, consumes business logic, and facilitates client / database communication. In its core, the data and security level rely on the PostgreSQL database that is used to save the data about the employees, resources, and system logs, which is encrypted and protected through the authentication services and constant monitoring. In order to enhance security, the architecture includes the features of firewalls, HTTPS as a secure communication point, and intrusion detection systems, and presents administrators with an admin panel to monitor and control the system.

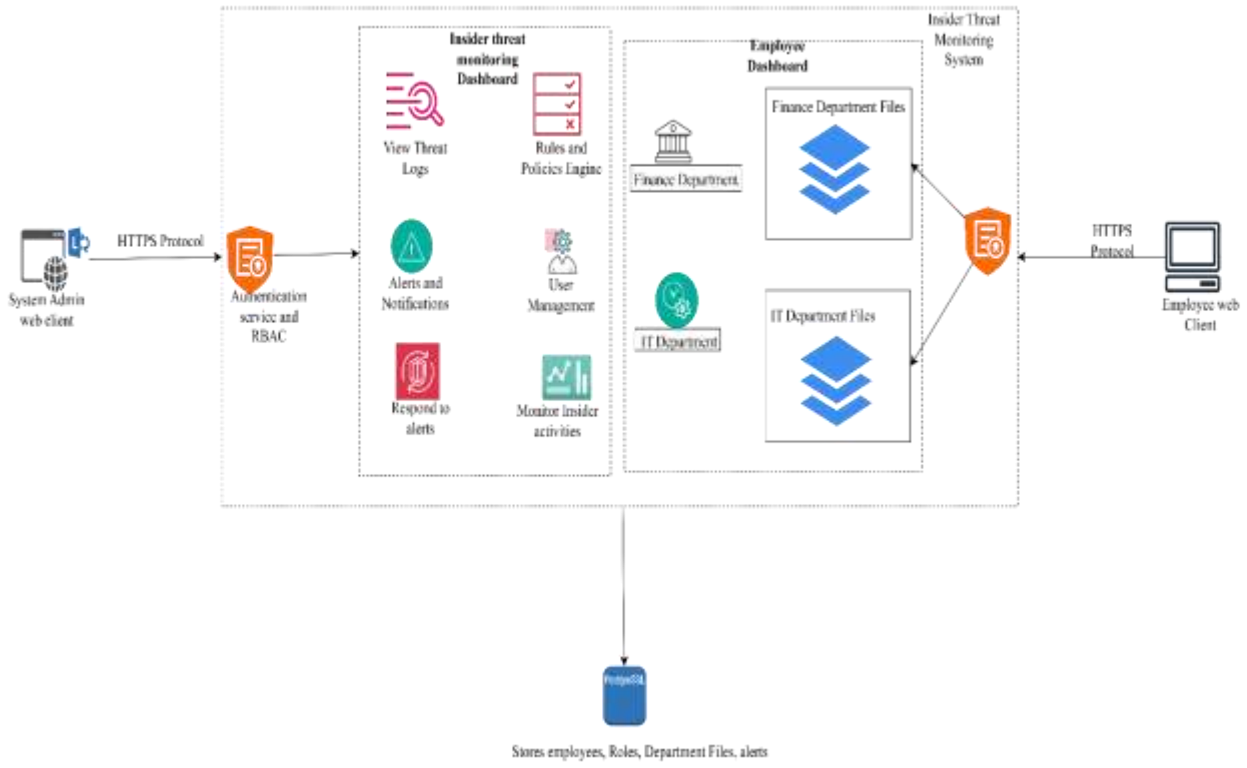


Figure 4. 7 System Architecture Diagram

4.5 System Wireframes

4.5.1 Login Wireframe

Figure 4.4 illustrates the Login wireframe in the form of a login page which is the main authentication field of IT administrators and employees. It also has the required fields of email addresses, password and one-time password (OTP) field, which support a two-factor authentication (2FA) strategy. This helps so that no unverified employees or IT administrators access the system and most especially those with higher privileges like the administrator. Under the OTP field is a Resend OTP link which is seen after a 60 second time counter. This ensures the employees can prompt at getting a new verification code in case one cannot be sent in a timely manner, and it is convenient and safe.

A link for “Forgot password” is also provided and it is supposed to help in case a user forgets his or her password. This results in a safe transfer of recovery of change in the password through the registered email.

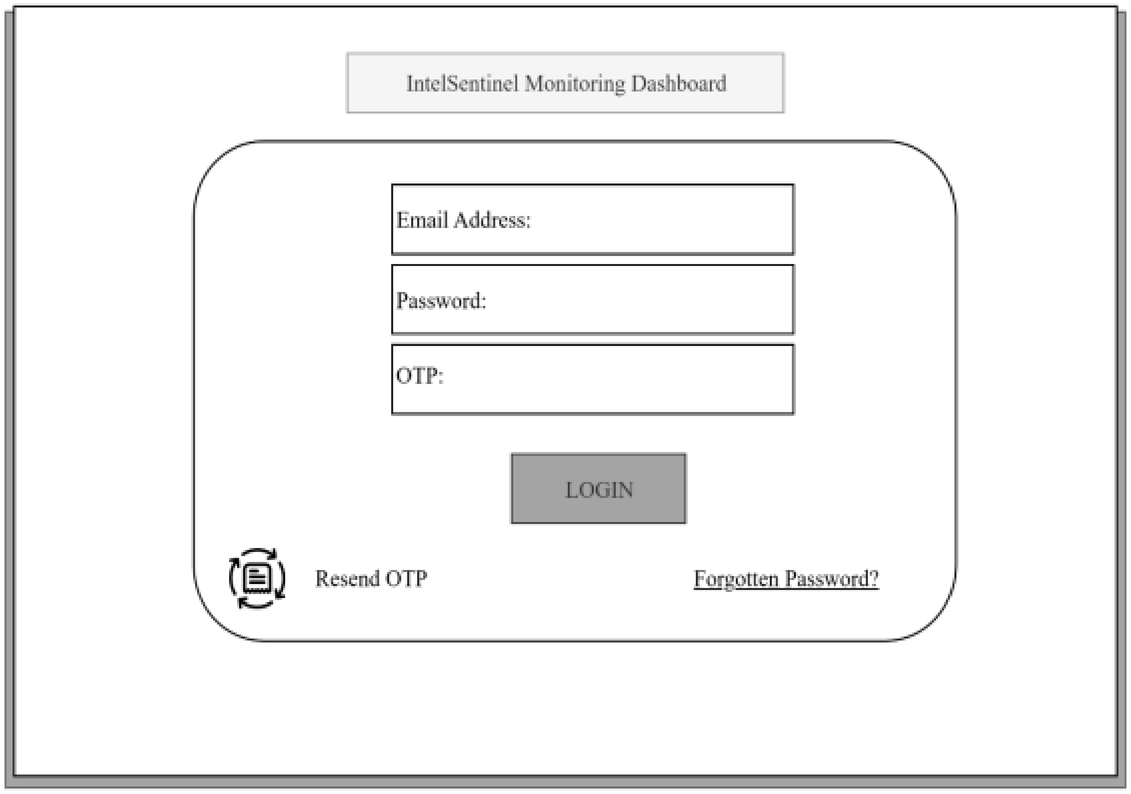


Figure 4. 8 Login Wireframe

4.5.2 Dashboard Overview Wireframe

Figure 4.5 illustrates the dashboard overview wireframe which is the main interface where the respective user parties access the main functionalities of the Insider Threat Monitoring System and are namely the Admin and the Security Officer. The wireframe is organized on a modular, structured layout with a top navigation bar, left-hand vertical side bar as well as a content area.

The sidebar menu has user-friendly links to important system modules that include Dashboard Overview, User Activity Logs, Service Alerts, Rule Engine User Management, Reports, and Settings.

The area of key contents is dynamic and shows module specific contents depending on the selection of the menu item. Examples of such dashboards include the Dashboard Overview screen which has real-time charts, and risk indicators. This allows the admin to get an idea of the security situation within the system at a single glance.

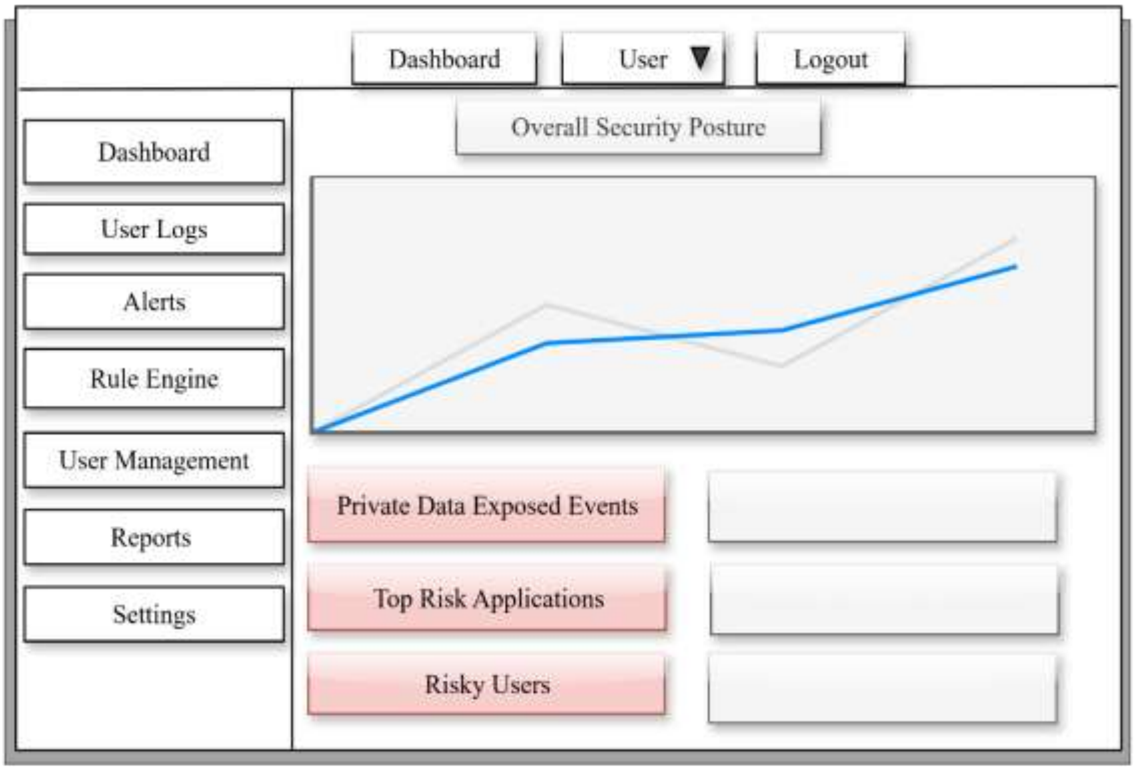


Figure 4. 9 Dashboard Overview Wireframe

4.5.3 User Management Interface

The first wireframe presents the layout and functionality of the User Management Interface, designed specifically for administrators. This interface allows for the management of user accounts, groups, and their associated access policies. The central feature is a structured table that lists all registered employees and groups along with their corresponding roles, permissions, and policy types.

The interface provides functionality to filter employees and interns by group or search for individual employees. Administrators can easily add new employees and interns or assign them to specific groups. Buttons such as “Assign Policy” and “Add New employee” enable streamlined user-role assignments. Furthermore, the system allows administrators to define whether a user’s access configuration should be manually or automatically enforced. This wireframe supports the enforcement of Role-Based Access Control (RBAC) and forms a foundational part of the system's access control mechanism, ensuring employees and interns only have access to resources necessary for their roles.

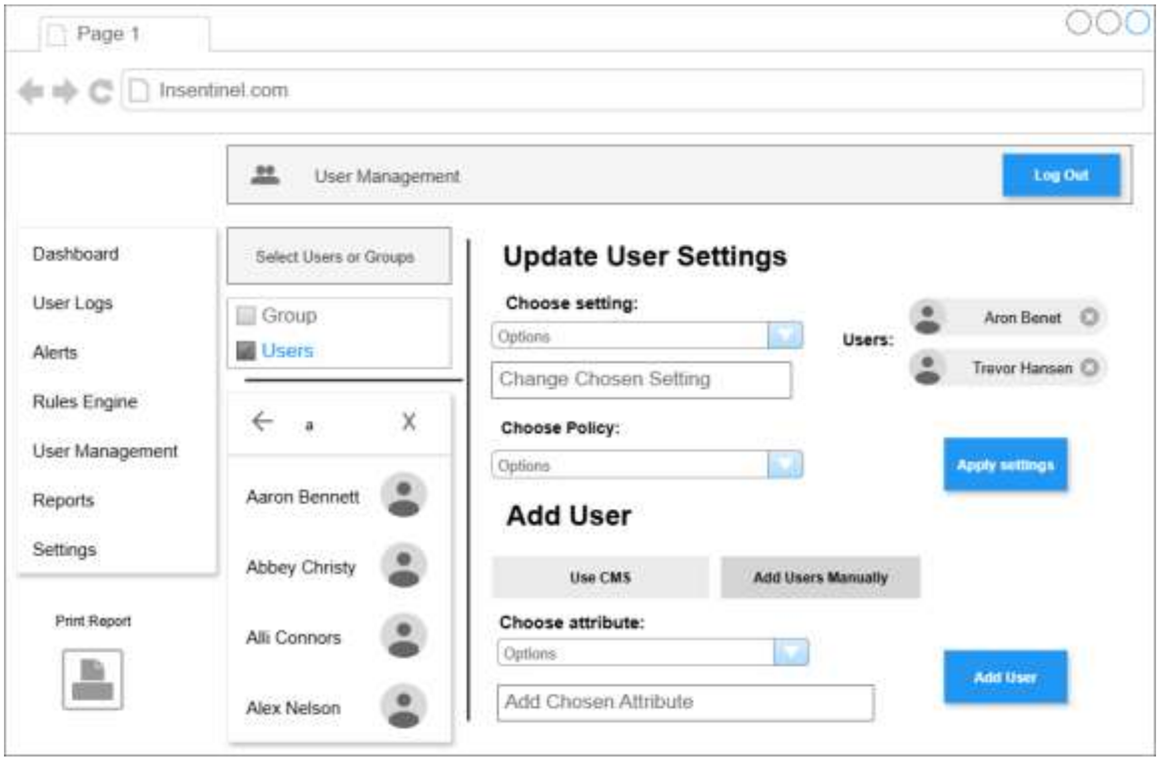


Figure 4. 10 User Management Interface

4.5.4 Rules Engine Interface Wireframe

The wireframe illustrates the Rules Engine Interface, which allows system administrators to define, modify, and manage access control policies and security rules. This section of the dashboard enables the creation of new rules by filling out a structured form. The form includes fields such as rule name, policy type (e.g., Role-Based, Discretionary, Mandatory Access Control), and the specific system resources that the rule applies to.

Administrators can select user or group targets from dropdown menus and configure whether the rule should permit or restrict access to a given resource. Once all parameters are specified, the rule can be saved and enforced immediately by the system. This interface is vital for customizing the security posture of the dashboard, ensuring that insider threats are proactively managed by dynamically adjusting access controls based on user behavior or administrative decisions. The design of this interface focuses on usability, enabling non-technical IT administrators to enforce complex rules with minimal configuration.

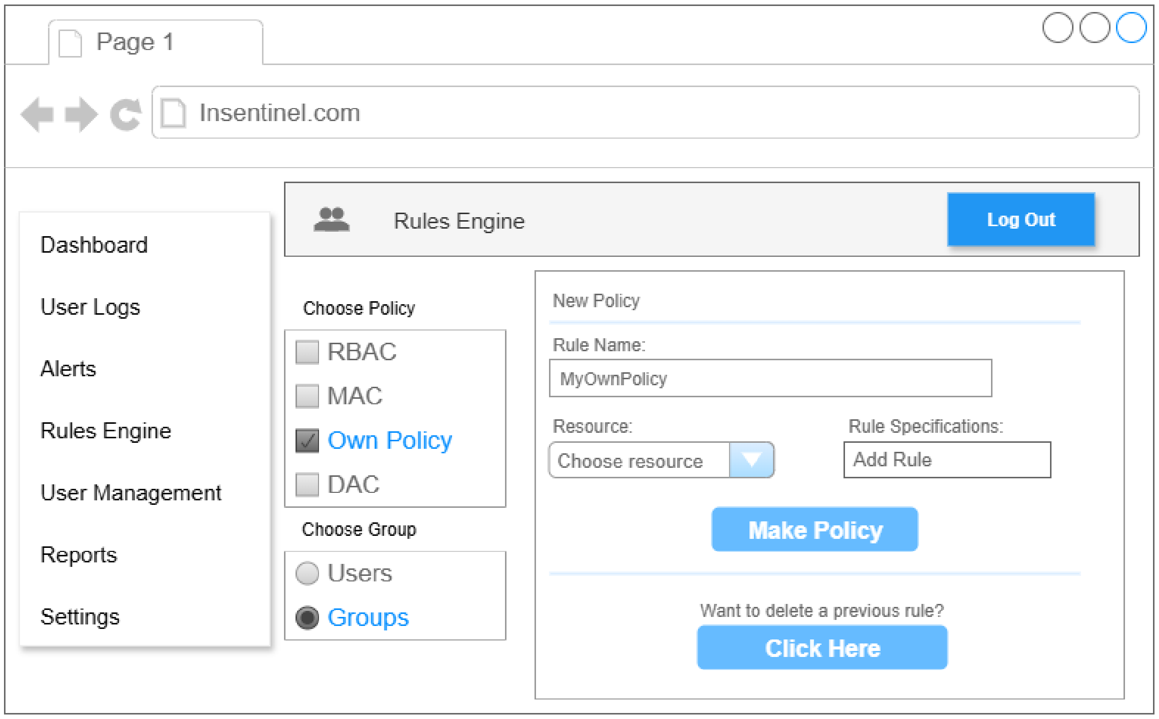


Figure 4. 11 Rules Engine Interface Wireframe

Chapter 5: System Implementation and Testing

5.1 Introduction

This chapter introduces the implementation and testing of the proposed Web based Insider Threat Monitoring Dashboard to Small and Medium sized enterprises (SMEs). It describes the context under which the system was created and the hardware and software requirements to be adopted in the implementation and test stage. In addition, the chapter outlines the process of the deployment, integration and evaluation of the system components to make sure that the expectations of the functional and non-functional requirements stated in the previous chapters have been satisfied.

5.2 Description of the Implementation Environment

In this section, the hardware and software specifications are defined to complete the system's implementation and set it up for us.

5.2.1 Hardware Requirements

The hardware requirements employed in implementation and testing were chosen to provide the best performance and compatibility with the development tools. The environment was made up of developer machines to be able to code and local servers where initial testing can be done.

Table 5.1 describes the hardware specifications for the development of web applications.

Table 5.1 Web Application Hardware Specifications

Component	Minimum specification	Purpose
Processor (CPU)	Quad-core processor, 2.4 GHz or higher	Supports multiple backend, frontend and database development and testing.
RAM	8 GB DDR4	Smooth execution of multiple development tools without lag.

Storage	256 GB SSD	Fast read/write operations, quick system boot, and rapid build times
Display	1366*768 resolution or higher	Allows enough workspace for editing codes and monitoring the system.

5.2.2 Network Requirements

The web application is characterized by real-time updates; therefore, internet connection is required. A minimum of a broadband internet connection with at least 10 Mbps speed is required to make package installations, APIs testing, and remote access to repositories reliable.

5.2.3 Software Requirements

Software environment has also been properly selected such that it matches the design needs of the project, safety objectives as well as the development criteria. It came with operating systems, application frameworks, database management systems, and building tools. This section describes the software requirements for both server-side and client-side.

i. Client-Side Requirements

Table 5.2 describes the requirements for the users accessing the web-application through their devices.

Table 5. 2 Client-Side Software Requirements

Component	Requirements	Purpose
Operating System	Windows 11	It is a stable and easy to use environment to run development tools and browsers.

Framework	Next.js	Framework to create dynamic, optimized frontend interface, that is React-based.
Styling	Material-UI	Framework for responsive and consistent styling.
HTTP Client	Axios	Supports the frontend to backend APIs communications.
Web Browser	Chrome, Firefox, Microsoft Edge	Applied in development to debug and UI testing.

ii. Server-side Requirements

Table 5.3 describes the Server-side software requirements.

Table 5. 3 Server-side Requirements

Component	Requirements	Purpose
Operating System	Windows 11 Pro (Local Dev)	It is used for running backend and database on local and staged test.
Programming Language	Python	Core language for backend development.
Backend Framework	Django	Server-side tool to handle APIs and business logic in addition to authentication.
API Toolkit	Django REST Framework (DRF)	Enables development of RESTful APIs to communicate with frontend.

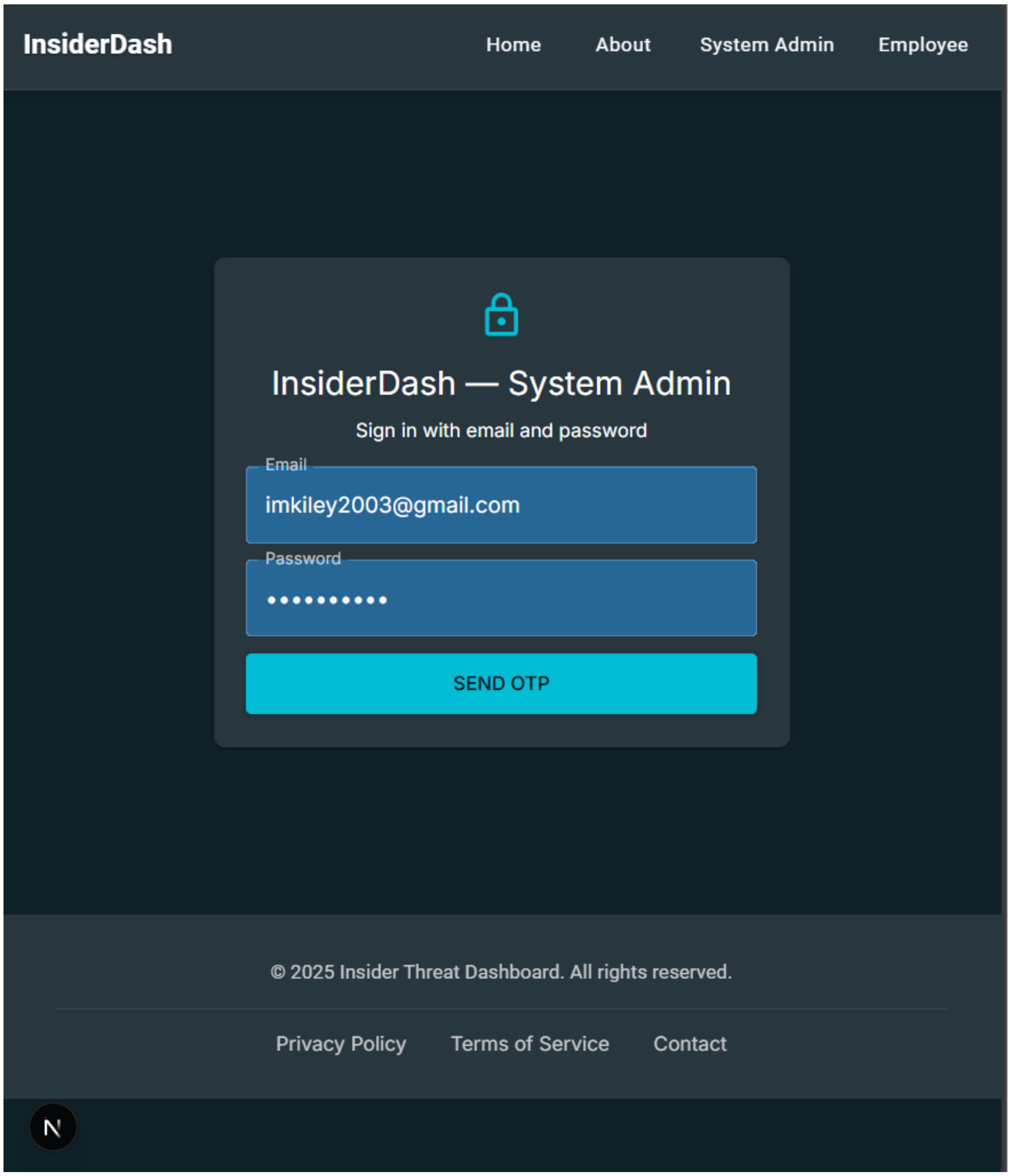
Database	PostgreSQL	Storage of system logs, user information, and settings in relational database.
Authentication	JSON Web Tokens (JWT)	This is stateless, secure authentication between the server and the client.

5.3 System Implementation

This section explains the key operative modules of the Insider Threat Monitoring Dashboard in terms of how they were designed during the development stage. Each of the modules had a specific task to complete on the functionality, security, and utilize the whole system. The screenshots of the developed modules are also offered to demonstrate the final work as a user interface and emphasize the main peculiarities.

5.3.1 User Login Page

The employees and the IT administrator's login to the system using their credentials and employ the use of one-time passwords (OTPs). It ensures that only authorized personnel can access the system and that each user's access is restricted according to their assigned role. Figure 5.1 below illustrates the Login page of the System Administrator, who uses their credentials to login into the system and view the insider threat dashboard. On the other hand, the employees must choose their departments as shown in Figure 5.2 below, before they are allowed to the login page as in Figure 5.1. Using their credentials they can login and access files according to their authorization levels.



InsiderDash — System Admin

Sign in with email and password

Email: imkiley2003@gmail.com

Password: ••••••••••

© 2025 Insider Threat Dashboard. All rights reserved.

Figure 5. 1 IT administrator Login Dashboard

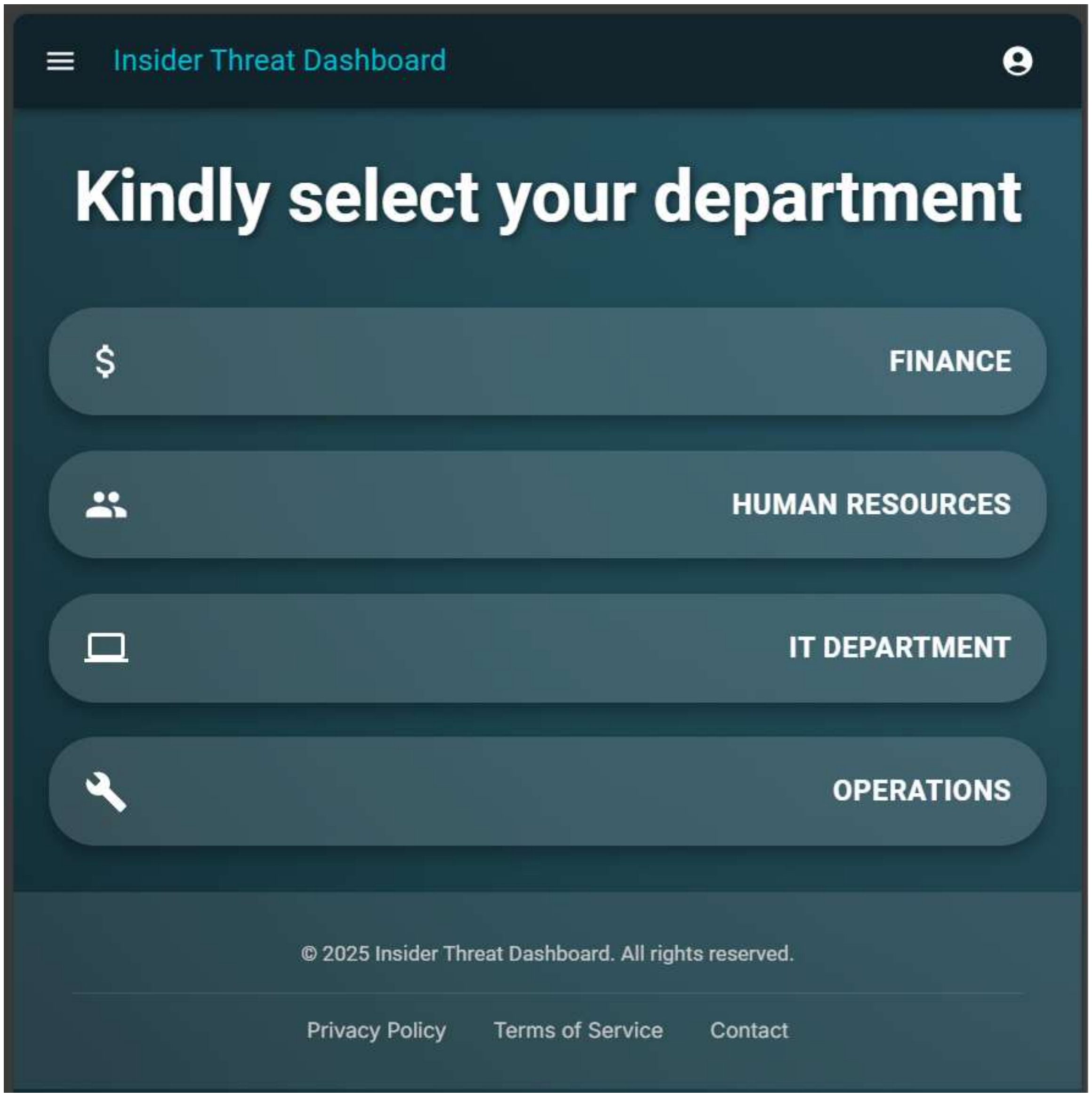


Figure 5. 2 Employee Login Dashboard

5.3.2 Insider Activity Monitoring Dashboard

The Insider Activity Monitoring Dashboard Module illustrated in Figure 5.3 below, offers up-to-date perception of who is doing what inside the system and network of the organization. It collects information across various logs and presents it in an interactive interface that must be analyzed by the security teams as illustrated in Figure 5.4 below.

This module forms the base intelligence level of the system, which allows administrators to identify strange trends that signal the presence of the threat of an insider. It provides interactive charts and tables in real-time views of data, options to filter and search by certain individual employees and administrators, time periods, activities on a normal, suspicious and high-risk basis and activity log export to offline review.

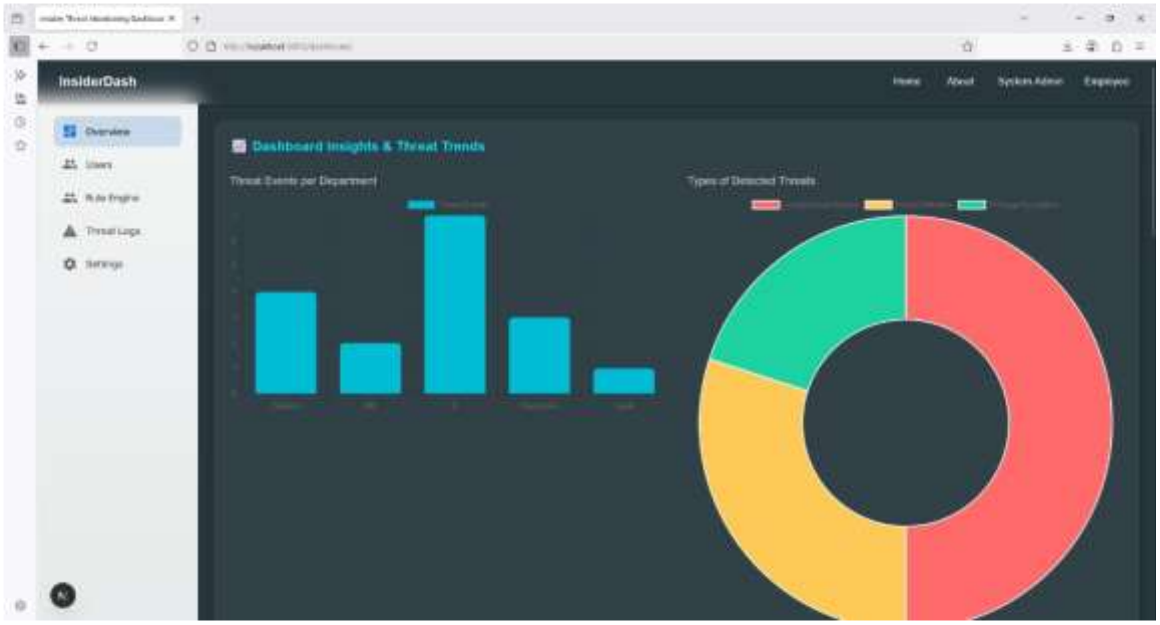

InsiderDash
Home
About
System Admin
Employee
Overview
Users
Rule Engine
Threat Logs
Settings
Dashboard Insights & Threat Trends
Threat Events per Department
Types of Detected Threats

Figure 5. 3 Overview of the Insider Threat Monitoring Dashboard

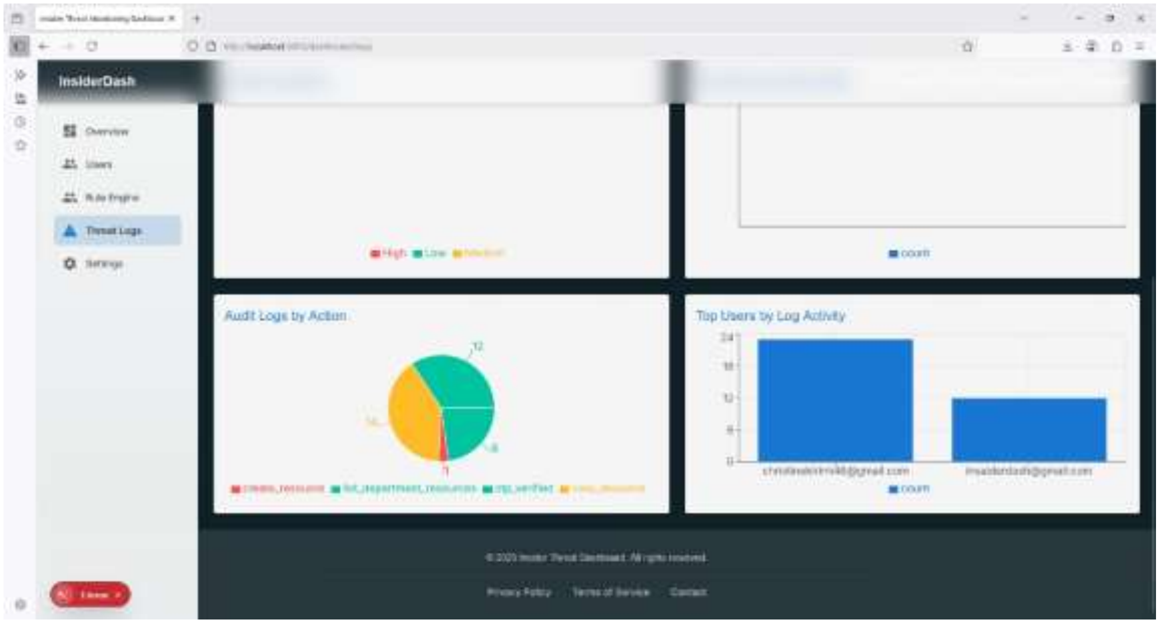
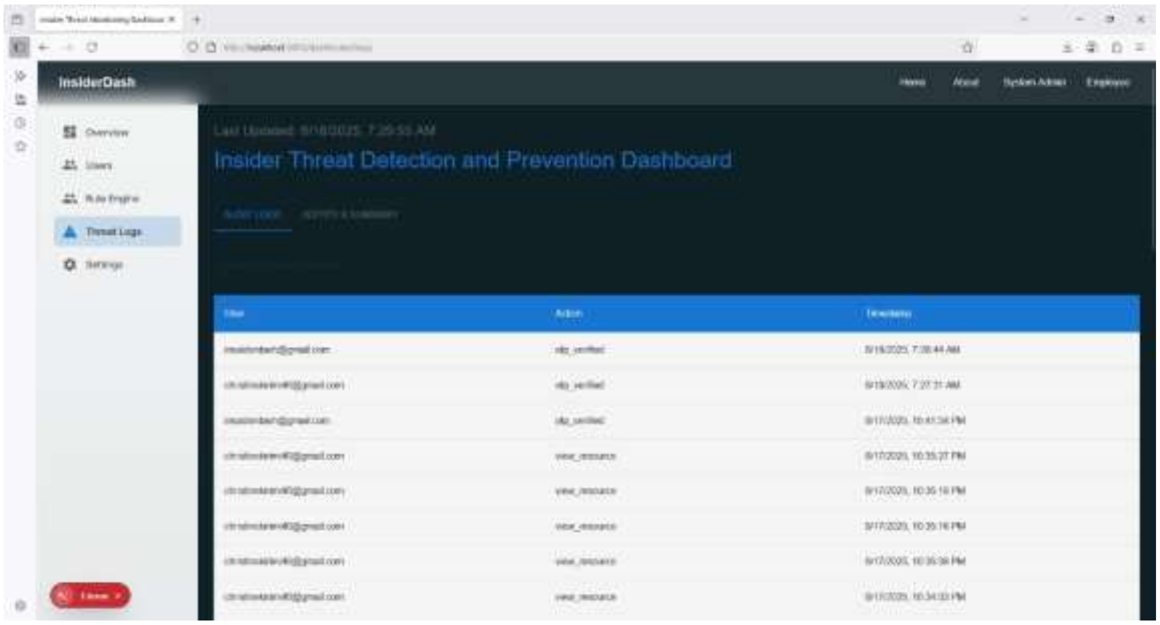


Figure 5. 4 Threat Logs

5.3.3 Threat Alerts and Incident Reporting

The Threat Alerts & Incident Reporting Module enables automated issues and sends alerts in situations in which specific threat conditions have been attained through many failed logins, data exfiltration attempts, or attempts to gain access to restricted files. The administrators are also allowed to conduct investigations through reporting an incident manually. The key functions include automatic warning in the form of dashboard messages, each alert has threat severity levels Low, Medium, High and Incident solving tracking.

Through this module, proactive measures in security can be achieved since unsuspicious activities are highlighted instantly and recorded, which can further be traced.



Last Updated: 9/18/2025, 7:29:55 AM

Insider Threat Detection and Prevention Dashboard

User	Action	Timestamp
[illegible]@gmail.com	otp_verified	9/18/2025, 7:38:44 AM
[illegible]@gmail.com	otp_verified	9/18/2025, 7:27:31 AM
[illegible]@gmail.com	otp_verified	9/17/2025, 10:41:34 PM
[illegible]@gmail.com	view_resource	9/17/2025, 10:35:27 PM
[illegible]@gmail.com	view_resource	9/17/2025, 10:35:19 PM
[illegible]@gmail.com	view_resource	9/17/2025, 10:35:16 PM
[illegible]@gmail.com	view_resource	9/17/2025, 10:35:38 PM
[illegible]@gmail.com	view_resource	9/17/2025, 10:34:33 PM

Figure 5. 5 Alerts Generation Module

5.3.4 System Administration and Configuration

System Administration and Configuration Module enables authorized administrators to configure or fix system parameters, user accounts and security policies. It is exclusive to the administrators and is essential to personalize the system to work directly with the organizational requirements.

The key functions include creation, editing, removal of user accounts, access permission assignment and group creation, Configuration of the system such as security thresholds and logging intervals. The module serves as the dashboard control center in the sense that the system is well configured, secured and maintained as time goes by.

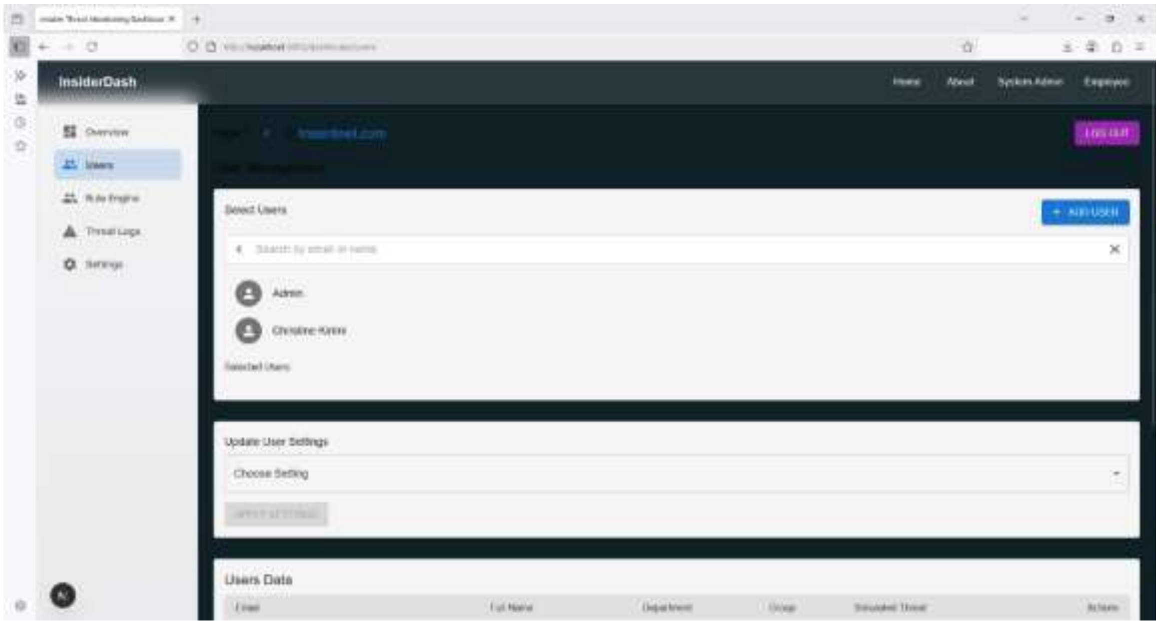


Figure 5. 6 User Management panel

5.3.5 Employee Dashboard (User Interaction Module)

The Employee Dashboard is the primary interface that is used by staff members who use the system every day. In contrast to the administrator or manager, employees can only see their own records of activity, they get appropriate alerts relating to their account, such as accessing unauthorized files, and suspicious login times, and they report suspicious activities on the system. The threats are visible from the admin dashboard as illustrated in Figure 5.3 and Figure 5.4. The Employee Dashboard provides employees with visibility to their authorized files and folders and thus interact with the system, hence generating activity logs.

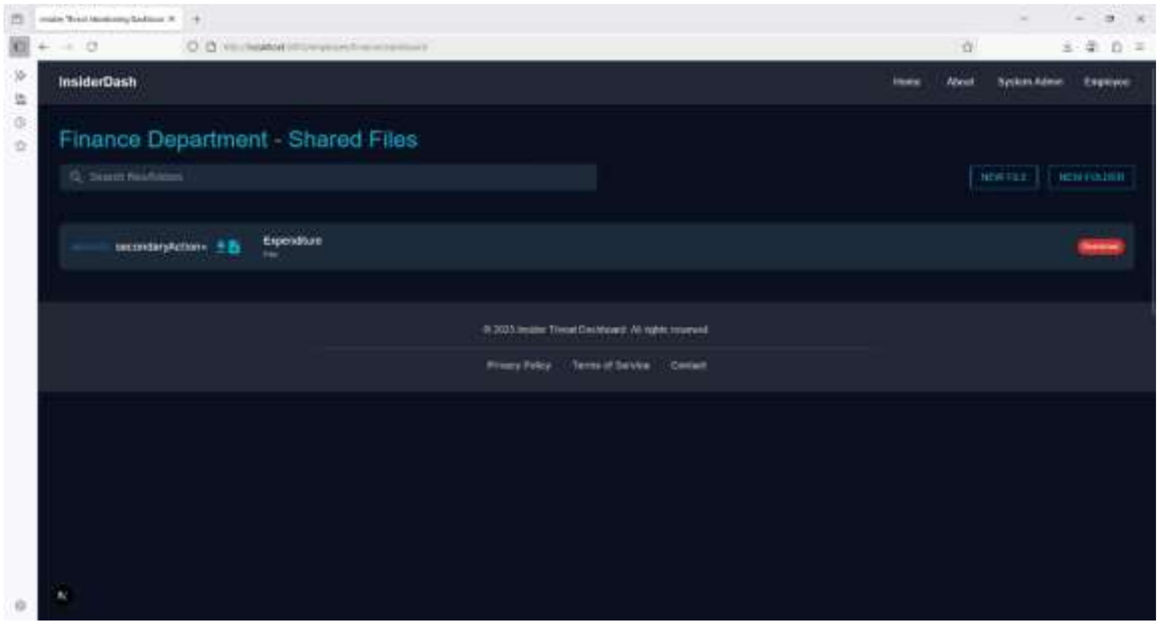


Figure 5. 7 Employee Dashboard

5.4 System Testing

System testing was implemented to ensure that the Insider Threat Monitoring Dashboard satisfies its intended functional and non-functional requirements. The testing procedure also guaranteed that all the modules introduced were functioning as desired both in regular and abnormal cases. Security reliability and functional correctness were also taken into consideration since the problem used was sensitive in terms of insider threat detection.

5.4.1 Testing Paradigm

This section outlines the testing process for the Insider Threat Monitoring Dashboard. Testing was carried out to ensure the proper operation of every module, profile data validation and verify that system interactions performed as desired. Both the White Box and Black Box Testing paradigms were employed.

Black Box Testing was adopted to check how the system works and not the code within it. It was illustrated with input and output behavior of behavior. As an illustration, successful log in of legitimate credentials to the system was supposed to give access and illegitimate credentials would be expected to cause error messages.

White Box Testing was used in the backend development process to ensure that logic flow of Django functions and API endpoints are valid. It included viewing the codebase and making sure that the authentication, database queries and checks of access were properly applied.

Unit Testing was also used where each part was tested separately, including endpoints of APIs, the queries in the database, and front-end forms. Programs such as the TestCase framework in the Django programming language.

5.4.2 Testing Results

The results of the system testing are summarized in the table below.

Table 5. 4 Module Test Results

Test Case	Description	Test Data	Experimental Outcome	Result	Test Verdict
Login Authentication	Verify that only authenticated users can login.	Valid credentials: insaiderdash@gmail.com / 1n5@1d3rd@5h Invalid credentials: insaiderdash@gmail.com / insaiderdash	System allowed valid login and rejected invalid login attempts.	Works as expected	Pass
Ensure role-based access control (RBAC) works	Ensure role-based access control (RBAC) works	Admin creates user and assigns "Manager" role	New manager logs in and had manager permissions only	Works as expected	Pass
Employee Dashboard	Check employee dashboard functionality	Employee logs in and views own activity logs	Dashboard displayed only that employee's	Works as expected	Pass

			department logs, not other logs from other departments		
Activity Monitoring Dashboard	Validate logs and charts for administrators	Test log entries: login success, file access, failed login	Admin dashboard displayed real-time logs and graphs	Works as expected	Pass
Threat Alerts	Verify that suspicious activities trigger alerts	5 consecutive failed login attempts	System flagged event and generated “High Risk” alert	Works as expected	Pass
User Management	Verify admin user management functions	Admin adds new user imkiley2003@gmail.com	User successfully added and visible in list	Works as expected	Pass
System Configuration	Test adjustment of system thresholds	Admin sets failed login threshold to 3	System generated alert after 3 failed attempts	Works as expected	Pass

Chapter 6: Conclusion

6.1 Conclusion

This Insider Threat Monitoring Dashboard was effectively designed and deployed to ensure that the growing insider threat requires SMEs to monitor internal activity and reduce the risks posed by insiders. The system accomplished the following significant goals: real-time monitoring that implemented an effective employee dashboard that enabled monitoring of activities within the system on an effective basis, role-based access control that effectively achieved the integration of authentication and authorization techniques and distinguished the employee, managers and administrators. Moreover, incident logging and analysis that provided implemented logging capabilities that generated logs on suspicious activity and offered logs that were analyzed to identify any possible threats.

However, despite all these accomplishments, the project faced some difficulties, that include: short development time that hindered the full development of the advanced features such as training of machine learning models with anomaly detection due to limited time and due to infrastructure limitations, testing was done in a small local testing environment and not in a scaled-out production environment.

In summary, the project has shown how SMEs can implement a lightweight and affordable version of insider threat monitoring tools that suit them in size of operations. Such a solution can provide the basis to improve organizational cybersecurity posture.

6.2 Recommendations

Based on the experience acquired during the creation and in the development of the Insider Threat Monitoring Dashboard, it is possible to make some recommendations. First, companies that are planning to buy the system must make sure it gets integrated with the existing Security Information and Event Management (SIEM) solutions, to enable centralized monitoring and enhance overall security stance. Moreover, there should be employee training and awareness programs alongside system deployment such that the staff members are well equipped with information about the security policies of the organization and best practices that they are expected to observe. Organizations are also advised to conduct periodic penetration tests on the system to target and eliminate vulnerabilities early enough and therefore afford resilience against the emerging threats. Finally, the system must be regularly benchmarked against cybersecurity frameworks and regulations e.g., GDPR, ISO 27001, or NIST to be compliant and provide cultural assurance to a stakeholder.

6.3 Future Works

Although the project fulfilled the set objectives there are several areas which have been identified to hold potential in the future research and system improvements. The possible enhancement would be the implementation of machine learning and artificial intelligence algorithms to allow predictive analytics and anomaly detection, thus reinforcing the capacity of the system at detecting insider threats in real time. It can also be done in future work to create a cloud-based dashboard where scalability, ease in updating the dashboard and the low costs of implementation on the SME could be accessed. The system may also be expanded to have cross platform accessibility using mobile applications that would have allowed administrators to detect certain threats remotely. The other potential area is to consider behavior biometrics or user behavior analytics to understand an additional detection delay. Such improvements implemented would alleviate the current gaps present and render the system more dynamic, intelligent and efficient at protecting SMEs against insider threats.

References

Acharya, K. (2024). *Web chatting application project report management system*. https://doi.org/10.22541/au.172254871.18588592/v1

Al-Harrasi, A., Shaikh, A. K., & Al-Badi, A. (2023). Towards protecting organisations' data by preventing data theft by malicious insiders. *International Journal of Organizational Analysis*, *31*(3), 875–888. https://doi.org/10.1108/IJOA-01-2021-2598/FULL/PDF

Alsowail, R. A., & Al-Shehari, T. (2022). Techniques and countermeasures for preventing insider threats. *PeerJ Computer Science*, *8*, e938. https://doi.org/10.7717/PEERJ-CS.938

Asrin, F., Informatics, G. U.-J. of I. S. and, & 2023, undefined. (n.d.). Implementing Website-Based School Information Systems in Public Elementary Schools Using Waterfall Model. *Journal-Isi.Org*. Retrieved June 26, 2025, from https://www.journal-isi.org/index.php/isi/article/view/495

Benson, D. (2024). *Leveraging Log Monitoring for Superior SaaS Performance*. https://logit.io/blog/post/log-monitoring-for-saas/

Christl, W. (2024). *EMPLOYEES AS RISKS*. https://crackedlabs.org/dl/CrackedLabs_Christl_SecurityRiskProfiling.pdf

Douglas, J., Douglas, A., Muturi, D., & Ochieng, J. (2017). *An Exploratory Study of Critical Success Factors for SMEs in Kenya*.

Fahimah Mohd Nassir, N., Fahri Abdul Rauf, U., Zainol, Z., & Abdul Ghani, K. (2024). REVEALING THE MULTI-PERSPECTIVE FACTORS BEHIND INSIDER THREATS IN CYBERSECURITY. *Journal of Media and Information Warfare*, *17*(2), 65–82.

Gunuganti, A. (2024). Insider Threat Detection and Mitigation. *Journal of Mathematical & Computer Applications*, 1–6. https://doi.org/10.47363/JMCA/2024(3)184

Hunker, J., Llc, A., & Probst, C. W. (n.d.). *Insiders and Insider Threats An Overview of Definitions and Mitigation Techniques*. *2*, 4–27.

Jusoh, S. N., & Ahmad, H. (2019). USAGE OF MICROSOFT EXCEL SPREADSHEET AS ACCOUNTING TOOLS IN SME COMPANY. *INWASCON Technology Magazine*, 23–25. https://doi.org/10.26480/ITECHMAG.01.2019.23.25

Lance, L. L. (2025). *Scrum Methodology: The Complete Guide & Best Practices*. https://thedigitalprojectmanager.com/projects/pm-methodology/scrum-methodology-complete-guide/

Le, D. C., & Zincir-Heywood, N. (2021). Anomaly Detection for Insider Threats Using Unsupervised Ensembles. *IEEE Transactions on Network and Service Management*, *18*(2), 1152–1164. https://doi.org/10.1109/TNSM.2021.3071928

Microsoft Defender for Endpoint - Microsoft Defender for Endpoint | Microsoft Learn. (n.d.). Retrieved June 25, 2025, from https://learn.microsoft.com/en-us/defender-endpoint/microsoft-defender-endpoint

Nasir, R., Afzal, M., Latif, R., & Iqbal, W. (2021). Behavioral Based Insider Threat Detection Using Deep Learning. *IEEE Access*, *9*, 143266–143274. https://doi.org/10.1109/ACCESS.2021.3118297

ObserverIT explained - Search. (n.d.). Retrieved June 26, 2025, from https://www.bing.com/search?q=ObserverIT%20explained&qs=n&form=QBRE&sp=-1&lq=0&pq=observerit%20expla&sc=9-16&sk=&cvid=3517CD4888CD4C31A2F50C821B8C1932

Omondi Oketch, J., & Okeyo, W. (2024). POLICY IMPLICATIONS FOR PERFORMANCE OF SMALL AND MEDIUM ENTERPRISES IN KENYA. In *African Journal of Emerging Issues (AJOEI). Online ISSN* (Issue 6).

Sassa, A. C., Alves De Almeida, I., Nakagomi, T., Pereira, F., & Silva De Oliveira, M. (2023). Scrum: A Systematic Literature Review. *IJACSA) International Journal of Advanced Computer Science and Applications*, *14*(4), 2023. www.ijacsa.thesai.org

Shelke, P., & Frantti, T. (2025). *Advanced Cyber Threat Detection*. *20*(1), 20. https://doi.org/10.34190/iccws.20.1.3326

Sutabri, T. (2023). Design of A Web-Based Social Network Information System. *International Journal Of Artificial Intelegence Research*, *6*(1), 2022. https://doi.org/10.29099/ijair.v6i1.454

Tavares, B. G., Keil, M., Sanches da Silva, C. E., & de Souza, A. D. (2021). A Risk Management Tool for Agile Software Development. *Journal of Computer Information Systems*, *61*(6), 561–570. https://doi.org/10.1080/08874417.2020.1839813;PAGE:STRING:ARTICLE/CHAPTER

Teramind Review: Features, Pros And Cons – Forbes Advisor. (n.d.). Retrieved June 26, 2025, from https://www.forbes.com/advisor/business/software/teramind-review/

Tetteh, A. K. (2024). *Issues in Information Systems Cybersecurity needs for SMEs*. *25*(1), 235–246. https://doi.org/10.48009/1_iis_2024_120

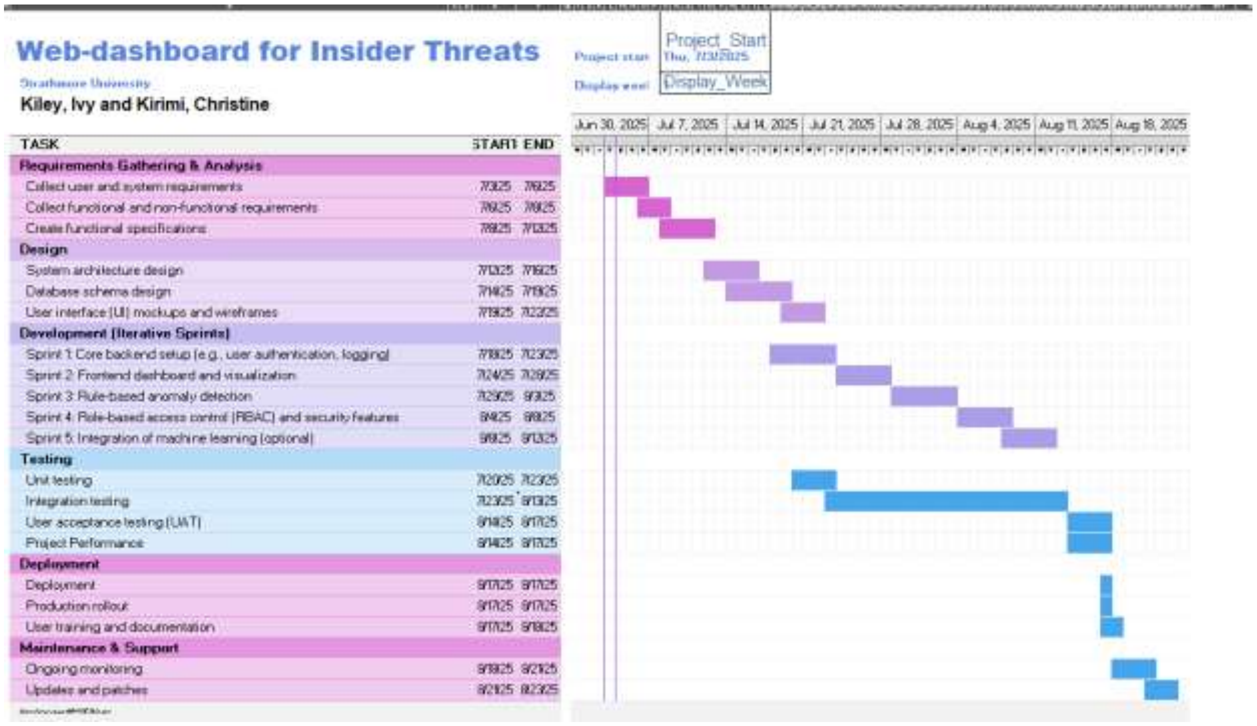
Thallapally, N. (2025). *View of How to Build and Maintain a Powerful Logging and Monitoring System*. https://journal.esrgroups.org/jes/article/view/8359/5624

Tselios, C., Politis, I., & Xenakis, C. (2022). Improving Network, Data and Application Security for SMEs. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3538969.3544426

van Dinter, R., Tekinerdogan, B., & Catal, C. (2023). Reference architecture for digital twin-based predictive maintenance systems. *Computers & Industrial Engineering*, *177*, 109099. https://doi.org/10.1016/J.CIE.2023.109099

Appendices

Appendix A: Gantt Chart



Web-dashboard for Insider Threats

Strathmore University

Kiley, Ivy and Kirimi, Christine

Project start: Project_Start Thu, 7/3/2025

Display week: Display_Week

TASK	START	END
Requirements Gathering & Analysis		
Collect user and system requirements	7/3/25	7/6/25
Collect functional and non-functional requirements	7/6/25	7/8/25
Create functional specifications	7/8/25	7/13/25
Design		
System architecture design	7/13/25	7/16/25
Database schema design	7/14/25	7/19/25
User interface (UI) mockups and wireframes	7/19/25	7/22/25
Development (Iterative Sprints)		
Sprint 1: Core backend setup (e.g., user authentication, logging)	7/18/25	7/23/25
Sprint 2: Frontend dashboard and visualization	7/24/25	7/28/25
Sprint 3: Rule-based anomaly detection	7/29/25	8/3/25
Sprint 4: Role-based access control (RBAC) and security features	8/4/25	8/8/25
Sprint 5: Integration of machine learning (optional)	8/8/25	8/13/25
Testing		
Unit testing	7/20/25	7/23/25
Integration testing	7/23/25	8/13/25
User acceptance testing (UAT)	8/14/25	8/17/25
Project Performance	8/14/25	8/17/25
Deployment		
Deployment	8/17/25	8/17/25
Production rollout	8/17/25	8/17/25
User training and documentation	8/17/25	8/18/25
Maintenance & Support		
Ongoing monitoring	8/18/25	8/21/25
Updates and patches	8/21/25	8/23/25

22% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

Bibliography

Match Groups

86 Not Cited or Quoted 17%

Matches with neither in-text citation nor quotation marks

20 Missing Quotations 4%

Matches that are still very similar to source material

0 Missing Citation 0%

Matches that have quotation marks, but no in-text citation

2 Cited and Quoted 1%

Matches with in-text citation present, but no quotation marks

Top Sources

15%	Internet sources
4%	Publications
16%	Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you