



University of
Zurich^{UZH}

Threat Identification and Mitigation using Process Models

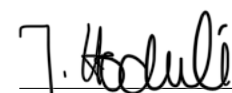
Lukas Sager, Jasmin Hochuli
Zurich, Switzerland
Student ID: 19-487-859, 20-705-711

Supervisor: Jan von der Assen, Thomas Grübl
Date of Submission: August 11, 2025

Declaration of Independence


I hereby declare that I have composed this work independently and without the use of any aids other than those declared (including generative AI such as ChatGPT). I am aware that I take full responsibility for the scientific character of the submitted text myself, even if AI aids were used and declared (after written confirmation by the supervising professor). All passages taken verbatim or in sense from published or unpublished writings are identified as such. The work has not yet been submitted in the same or similar form or in excerpts as part of another examination.

Zürich, 11. 8. 2025



Signature of student

Zürich, 11. 8. 2025



Signature of student

Abstract

Diese Arbeit stellt den Prototyp *“Insider Threat Modeler 2.0“* vor, der die Identifikation von Insider-Threats sowie Vorschläge geeigneter Gegenmassnahmen aus einer prozessorientierten Perspektive unterstützt. Das System nutzt ein Large Language Model (LLM), um BPMN-basierte Geschäftsprozesse und deren Kontext dynamisch zu analysieren und adressiert damit eine in der Literatur identifizierte Forschungslücke. Im Rahmen der Studie werden verschiedene LLMs hinsichtlich ihrer Fähigkeit verglichen, Bedrohungen zu erkennen und sinnvolle Massnahmen zur Risikominderung zu generieren. Die Ergebnisse zeigen, dass kleinere Modelle nicht über das notwendige Verständnis verfügen, um relevante Erkenntnisse zu liefern, während grössere, cloud-basierte LLMs strukturierte und verwertbare Einschätzungen generieren. Der finale Prototyp wurde in Experteninterviews evaluiert, wobei die Vollständigkeit und Korrektheit der Ausgabe sowie die Qualität der Methodik und Benutzerfreundlichkeit der Implementation bewertet wurden. Obwohl gewisse Einschränkungen in der Granularität der Ausgabe bestehen bleiben, erkannten die Experten das Potenzial des Systems zur Unterstützung und Beschleunigung der Identifikation von Insider-Threats an. Die Arbeit leistet damit einen neuartigen Beitrag zur prozess-basierten Erkennung und Minderung von Insider-Threats mithilfe von LLMs und zeigt sowohl deren Potenziale als auch aktuelle Grenzen auf.

This paper introduces the *“Insider Threat Modeler 2.0“*, a prototype designed to identify insider threats and propose countermeasures from a process-based perspective. Leveraging a large language model (LLM), the tool dynamically analyzes BPMN-based business processes and their context, addressing a gap identified in the literature. The study compares various LLMs in terms of their ability to detect threats and generate meaningful mitigation strategies. Findings show that smaller models often lack sufficient understanding of the business process to produce relevant insights, while large-scale cloud-based LLMs deliver more structured and actionable assessments. The final prototype was evaluated through expert interviews, assessing the completeness, correctness of the output as well as the methodological soundness and usability of the prototype. Although some limitations in report granularity remain, the security experts recognized the system’s potential to support and accelerate insider threat identification. Overall, this work contributes a novel approach to process-based insider threat modeling using LLMs, highlighting both their promise and current limitations.

Acknowledgments

We would like to express our sincere gratitude to our supervisors for their valuable guidance and support throughout the course of this master's project. In particular, we extend our special thanks to Jan von der Assen for his continuous support, insightful feedback, and dedicated mentorship. In addition, we would like to thank the two security experts for helping us evaluate the prototype.

Contents

Declaration of Independence	i
Abstract	iii
Acknowledgments	v
1 Introduction	1
1.1 Motivation	2
1.2 Description of Work	2
1.3 Thesis Outline	3
2 Background	5
2.1 Threat Modeling	5
2.2 Insider Threats	6
2.3 Countermeasures	7
2.4 LLMs and RAG	8
2.5 Business Process Modeling	9
2.5.1 Flow Objects	10
2.5.2 Connecting Objects	11
2.5.3 Swimlanes	11
2.5.4 Artifacts	11

3	Related Work	13
3.1	Threat Modeling Using Business Process Models	15
3.2	Threat Modeling Using LLMs	15
3.3	Business Process Modeling Using LLMs	16
3.4	Limitations	17
4	Architecture	19
4.1	Feasibility of LLM Integration	19
4.2	Proposed Methodology	20
4.3	RAG Architecture	22
5	Implementation	23
5.1	Technology Stack	23
5.2	Implementation of the Methodology	23
5.3	BPMN Integration	28
5.3.1	XML File Reduction	28
5.4	Retrieval-Augmented Generation	28
5.4.1	Chunking	29
5.4.2	Embedding	30
5.4.3	Vector Database	30
5.5	Prompt Construction	31
5.5.1	Iterative Improvement	31
5.5.2	Input	31
5.5.3	Output	33
5.6	LLM	35
5.7	Processing of the LLM Response	36
5.8	Frontend Improvements	36
5.8.1	Routing	37
5.8.2	Descriptions	37

6	Evaluation	39
6.1	LLM Comparison	41
6.1.1	Input Data	41
6.1.2	Method	41
6.2	Expert Evaluation	44
6.2.1	Output Evaluation	45
6.2.2	Methodological Discussion	48
6.2.3	Usability Evaluation	50
6.3	Discussion	51
6.3.1	Correctness and Completeness of the Output	51
6.3.2	Methodological Applicability	51
6.3.3	Design and Usability Considerations	52
6.3.4	Limitations of the Evaluation	52
6.4	Refinement	52
6.4.1	Rephrasing of the First Context Question	53
6.4.2	Extending the Output Format by a Threat Description	53
6.5	Comparison with Previous Work	54
7	Conclusion	57
7.1	Future Work	58
	Bibliography	59
	Abbreviations	63
	List of Figures	63
	List of Tables	65
	List of Listings	67

A	Supplementary Content	71
A.1	Context Information	71
A.2	Threat Overview	72
A.3	Supplementary Files	74
B	Other	75
B.1	Use of AI	75
B.2	Installation Guidelines	75

Chapter 1

Introduction

Securing modern information systems is increasingly complex, as threats not only arise from external actors but also from legitimate users who may act maliciously or negligently [1], [2]. Moreover, such threats are challenging as they exploit legitimate access embedded in complex organizational processes [3].

According to a recent report by IBM [4], attacks involving legitimate accounts increased by 71% in 2023 and now account for nearly 30% of all cyberattacks. The same study suggests that 84% of such attacks could have been prevented by applying established best practices [4]. These numbers highlight that insider threats are not only frequent but also often avoidable [4].

The problem is that traditional threat modeling approaches usually focus on technical vulnerabilities and often overlook weaknesses introduced by the design of the business process [5]. This shows that organizations need to detect and mitigate insider threats early, ideally already when designing a process flow [4].

At the same time, recent advances in artificial intelligence, especially large language models (LLMs), create new opportunities for improving threat detection and analysis [6]. Their ability to understand natural language and reason about complex structures makes them well-suited to assist in security tasks that previously required expert interpretation [7]. Moreover, they enable threat modeling tools to keep up with the constantly evolving threat landscape [7]. However, no existing tool explores the capabilities of LLMs to reason over process models for insider threat modeling.

Therefore, this thesis investigates the integration of LLMs within business process models to support automated insider threat modeling. It builds on previous studies that introduced a rule-based, expert-driven prototype and enhances it by embedding contextual reasoning, mitigation suggestions, and Retrieval-Augmented Generation (RAG)-based retrieval from an established insider threat knowledge base [8]. The goal of this Master's project is to reduce manual effort, enhance the quality of insider threat analysis, and explore the extent to which AI can assist in process-based threat modeling.

1.1 Motivation

This work builds on the thesis of [8] and the paper of [5], which proposed a prototype that implements threat modeling of insider threats within business processes. The previous thesis identified a notable research gap: traditional threat modeling approaches often overlook insider threats, which can be addressed by taking a process-based perspective. For this, [8] introduced a methodology using Business Process Modeling Notation (BPMN) models to link insider threats to specific process elements. While that work is an important foundation, it did not make use of recent advancements in artificial intelligence, such as large language models (LLMs), to enhance automation or contextual reasoning in threat identification [8].

According to the research in [8], the papers [9], [10] confirmed that the process-based perspective can be used to detect and model insider threats. Furthermore, other works have shown that BPMN is a valid framework for insider threat identification [11]–[15]. Moreover, recent studies about the use of LLMs combined with a RAG approach have shown great potential in generating threat vectors and attack graphs based on structured vulnerability data [6], [16], [17]. Additionally, exploring LLMs in the context of business process modeling itself have demonstrated their ability of generating BPMN models from natural language [18], [19].

In summary, the literature confirms three key points: BPMN is a valid framework for threat modeling, LLMs can enhance threat analysis and process modeling tasks, and insider threats are critical but not well addressed by many threat modeling tools. Although various works explore LLM integration, BPMN-based process analysis, and insider threat modeling, there is no work that combines all three aspects. This thesis aims to fill that gap.

1.2 Description of Work

This work proposes a five-step methodology and a prototype called *Insider Threat Modeler 2.0* that enables the identification of insider threats in business processes using the support of a large language model. The main contributions of this thesis are: (1) a methodology that combines LLMs and business process models for insider threat modeling, (2) a prototype that implements the use of LLMs combined with RAG for contextual retrieval, (3) a comparison of different LLMs, and (4) an expert evaluation.

The approach is designed to assist users in conducting structured threat modeling using business process models. It implements a RAG architecture that builds on the insider threat knowledge base introduced by [8]. By integrating context information provided by the user, the system analyzes each element of the business process to detect relevant insider threats. The output includes a tailored overview of possible threats, their impacts, and corresponding mitigation strategies. These additional features reduce the time needed from security experts and enable a more efficient, scalable, and context-aware threat modeling process.

1.3 Thesis Outline

The next chapters of this work are structured as follows. Chapter 2 elaborates on the theoretical concepts that need to be understood, while Chapter 3 analyzes academia for relevant sources that concern the topic of this project. Next, Chapter 4 explains the methodology and architecture of the prototype on a high level. The actual implementation of this methodology is then further described in Chapter 5. The evaluation of the prototype can be found in Chapter 6, whereas the summary and conclusions follow in Chapter 7.

Chapter 2

Background

After introducing and motivating the topic of threat modeling of insider threats, the following sections discuss the most important theoretical foundations for this project. The aim is to give the reader an overview and explanations to understand the techniques and methodologies used in this work.

2.1 Threat Modeling

Threat modeling is a procedure to establish a high level of security in an information system by identifying critical elements that may be vulnerable to attacks [20], [21]. It is typically conducted early in the development process when implementing new software applications [20], [21]. While threat modeling has been used for many years, it has received increasing attention in the literature lately, as mentioned by [20].

Various approaches to threat modeling exist, including manual textual descriptions, formal models based on mathematics and stochastics, graphical notations, and automated tools [21]–[23]. Graph-based modeling approaches are most common when modeling a system [20]. These approaches may differ in focus such as concentrating on a system’s assets, potential threats, or vulnerabilities in the services provided by an organization [24]. [22] explores various dimensions of threat modeling approaches, including asset-centric, threat-centric, data-centric, and system-centric methodologies. Furthermore, as shown in Figure 2.1, [22] categorizes commonly used threat modeling techniques within a structured matrix, including a graphical, automatic, formal, and manual dimension.

Most threat modeling approaches rely on specific threat classification frameworks [20]. According to [20], STRIDE is the most commonly applied methodology. Others, such as LINDDUN and CAPEC, are less widespread. [21] also discusses STRIDE, along with other methodologies, including Abuser Stories, the STRIDE Average Model, Attack Trees, Fuzzy Logic, the SDL Threat Modeling Tool, T-MAP, and CORAS. They all support the threat modeling process in a structured way [20]. Most of them can be found in the matrix of Figure 2.1 [22].

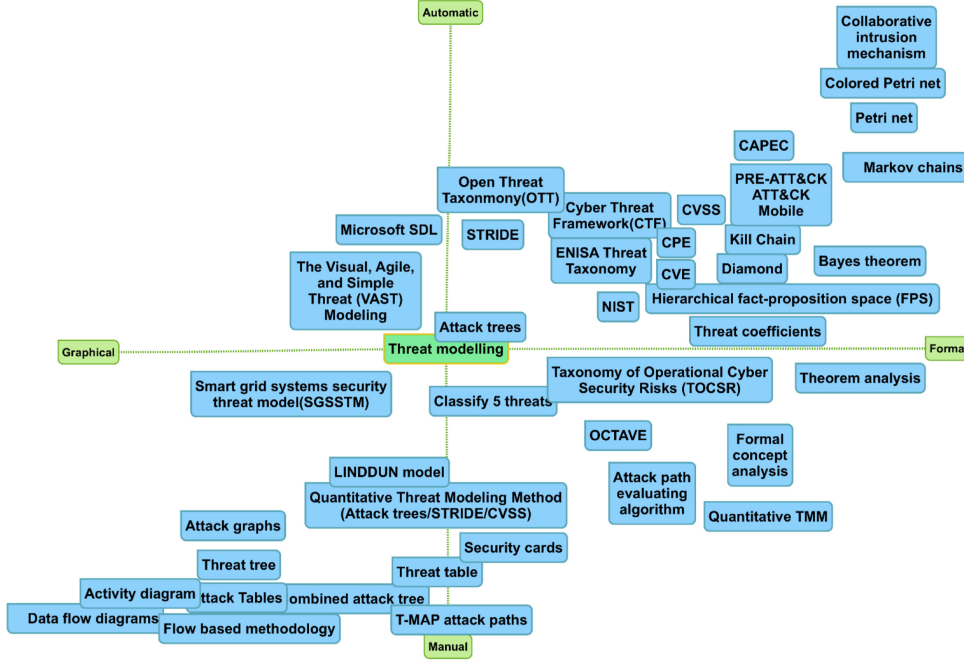


Figure 2.1: Threat Modeling Approaches [22]

2.2 Insider Threats

According to [2] and [1], many cybersecurity attacks originate from malicious insiders within an organization. An *insider* is defined as a person who either is a trusted user within the organization or has legitimate access to its network, systems, and resources [1]–[3]. The unauthorized misuse of this access is known as an *insider threat* [2], [3].

However, insider threats are not always driven by malicious intent [1], [3]. Unintentional attacks, *i.e.*, an employee mistakenly clicking on a malicious email link, also fall under this category [1]. The most commonly used classification in the literature distinguishes between intentional and unintentional insider threats [3]. However, [3] argues that this unidimensional classification is inadequate as it does not account for evolving technologies considering the rise of AI, Machine Learning, and IoT.

To address this limitation, [3] introduces a more comprehensive framework with four dimensions: knowledge, compliance, volition, and goal, as shown in Figure 2.2. *Knowledge* reflects the insider’s awareness of their unauthorized actions, while *compliance* assesses the adherence to the organization’s security policies. *Volition* considers the willingness of the attacker to harm an organization, and *goal* examines the motivation behind the attack, namely if it was for the attacker, the society, or for ideological or malicious reasons [3].

As insiders have legitimate access to a company’s information systems, it is harder to detect their unjustified actions [1]–[3]. Furthermore, because of their familiarity with the system’s vulnerabilities and interconnections, insiders can potentially harm the organization to a bigger extent [3]. Such harm can take various forms, violating core security principles: Confidentiality, Integrity, and Availability (CIA). Examples include viewing,

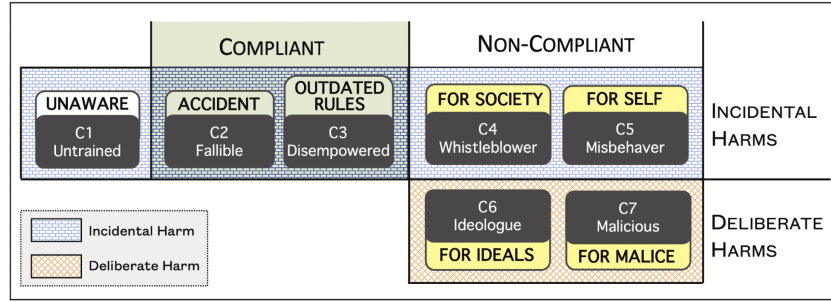


Figure 2.2: VISTA Framework [3]

exfiltrating, or leaking sensitive data for personal or financial reasons, while others might sabotage information or commit fraud [1], [2].

2.3 Countermeasures

Insider attacks remain a recurring topic for companies due to their likelihood. According to [4], there was a 71% increase in attacks on systems through legitimate accounts in 2023. Those attacks account for 30% of all cyber attacks. Further, the study implies that 84% of the attacks via an access vector could have been prevented by applying best practices. While the occurrence of insider threats is an issue, their detection is as well [2]. For either, it is crucial to establish a good understanding of the potential insiders as well as their intentions. As noted in Section 2.2, most frameworks distinguish between intentional and unintentional insider attacks. While unintentional attacks can be addressed effectively through training and education of insiders, intentional attacks create further challenges for security experts [1].

A standardized and recognized framework for countermeasures, addressing insider threats, has not been established. In literature reviews, unifying frameworks are proposed. For instance, [25] proposes a prevention hierarchy on three levels with prevention methods within them: *Security Awareness*, *Identity and Access Management*, and *Implementation of Security Controls*, as shown in Figure 2.3. In [1], prevention methods are divided into biometrics and asset-metrics and then further split into categorized approaches.

The National Institute of Science and Technology (NIST) proposes a multitude of frameworks within the *NIST Cybersecurity Framework (CSF) 2.0* that companies can use to proactively address cybersecurity threats in an organization. The frameworks within CSF 2.0 that are most relevant to insider threats are Identify, Protect, and Detect. Especially in the *Protect* category exist numerous measures addressing insider threats through *identity management*, *authentication*, and *access control* frameworks. This is extended by *awareness and training*, *data security*, and *platform security* frameworks. While the CSF provides a comprehensive coverage of cyber defense, it does not focus specifically on insider threats [26].

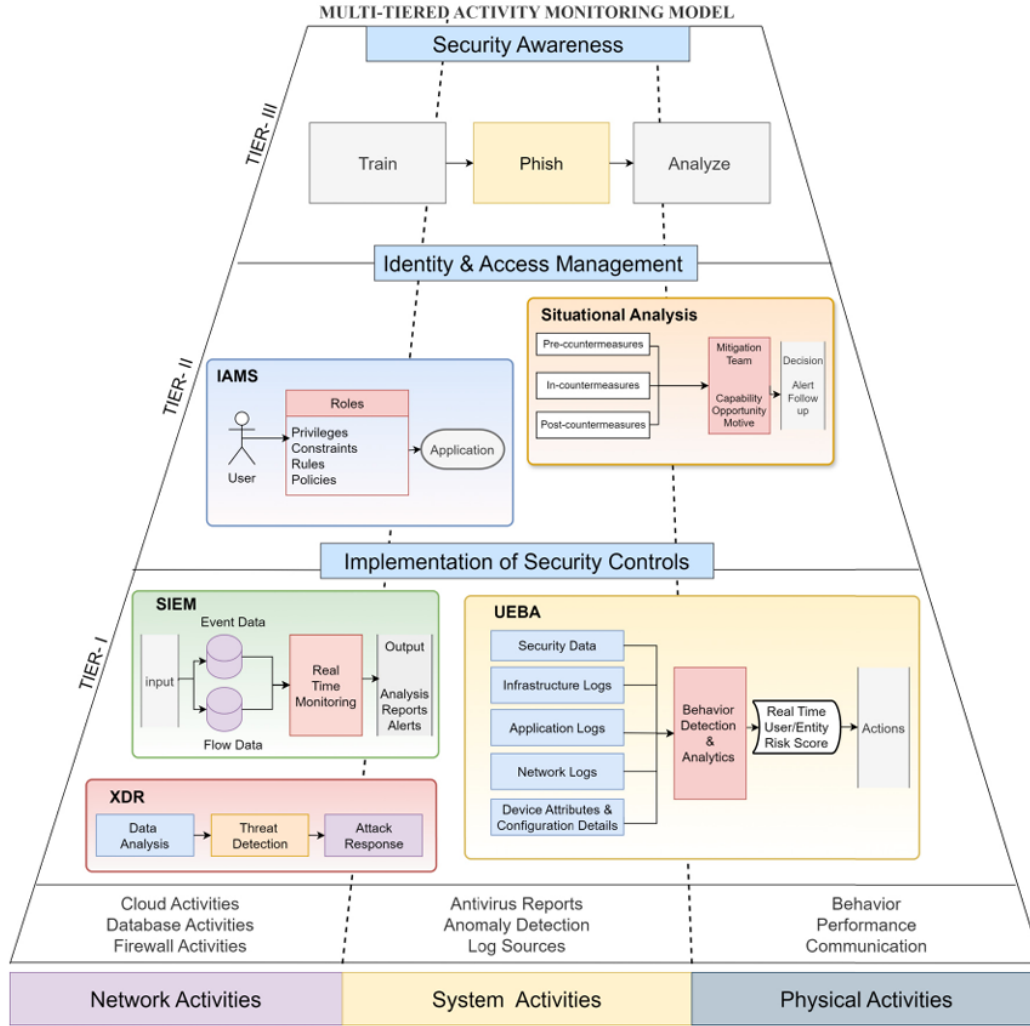


Fig. 8. Multi-tiered activity monitoring model.

Figure 2.3: Multi-Tiered Activity Monitoring Model [27]

2.4 LLMs and RAG

Large Language Models are sophisticated neural networks trained on vast amounts of textual data, allowing them to execute a wide range of language-related tasks [28]. They have transformed artificial intelligence by demonstrating remarkable proficiency in understanding, generating, and reasoning with natural language [7]. As a result, LLMs have significantly impacted industries that rely on content creation in various applications [28].

In the field of cybersecurity, LLMs have the potential to drive transformative advancements [7]. As traditional systems have struggled to keep up with the rapidly evolving cybersecurity threat landscape, language models offer a promising solution to address this challenge [7]. Their applications span various areas, including *security threat detection*, *engineering*, and *verification*, *incident response*, and *cyber defense*, as well as *security automation* and *awareness* [7], [27], [28]. These fields can be inspected in Figure 2.4.

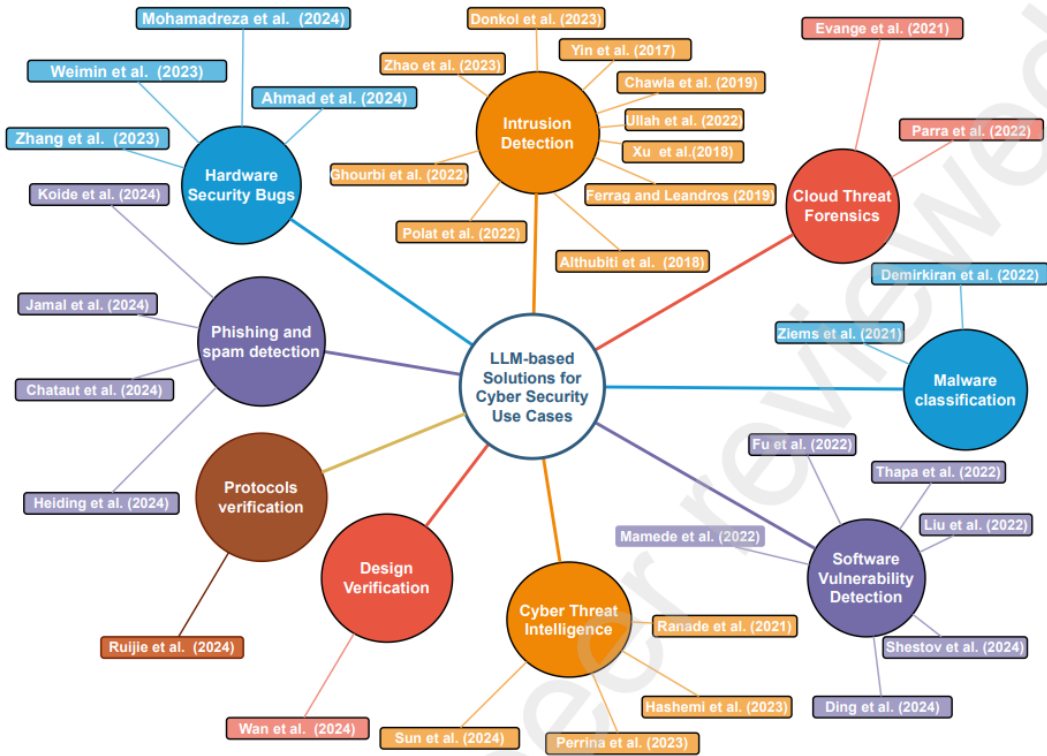


Fig. 4: LLM-based Solutions for Cyber Security Use Cases.

Figure 2.4: Use Cases of LLMs in Cybersecurity [27]

Various companies have developed numerous LLMs, with some being open source and others proprietary [7], [28]. Examples for LLMs are: *T5*, *GPT-3*, *PaLM-2*, *Llama*, *Xuan Yuan 2.0*, *AlexaTM*, *GLM-130B*, *GPT-4*, *Goat*, etc. [28]. Figure 2.5 provides an overview created by [7] of LLMs that have been utilized in cybersecurity.

A more advanced approach after LLMs is Retrieval-Augmented Generation (RAG). By integrating external databases, RAG enhances language models with additional context and real-time information, improving their performance. This capability is particularly beneficial in the field of cybersecurity, where dynamic retrieval of current information provides a significant advantage compared to traditional approaches [27].

2.5 Business Process Modeling

Business Process Modeling Notation (BPMN) is a standardized framework widely used in academia and business to visualize and protocol processes. The business process is defined as “A set of one or more linked procedures or activities which collectively realize a business objective or policy goal, normally within the context of an organizational structure defining functional roles and relationships” [29]. The purpose of the notation includes simulations and quantitative analysis to improve efficiency, enterprise modeling, quality assurance, and software process improvements [30].

Table 2 A Summary of LLMs used in cybersecurity (this paper)

Organization	LLMs	Size	Open-Source	Count	Link
OpenAI	GPT-3.5	175B	×	86	https://chat.openai.com/
	GPT-4	–	×	56	https://chat.openai.com/
	Codex	–	×	13	https://openai.com/blog/openai-codex
	davinci(-002,-003)	175B	×	9	https://openai.com/blog/openai-api
Google	Bard&Gemini	–	×	12	https://gemini.google.com/
	PaLM(-1,-2)	540B	×	7	https://ai.google.dev/models/palm
Anthropic	Claude(-1,-2)	–	×	2	https://claude.ai/
Github	Copilot	–	×	2	https://github.com/features/copilot
Microsoft	BingChat	–	×	2	https://www.bing.com/chat
EleutherAI	GPT-J	6B	✓	2	https://huggingface.co/EleutherAI/gpt-j-6b
	GPT-Neo	2.7B	✓	3	https://huggingface.co/EleutherAI/gpt-neo-2.7B
Meta	Llama(-1,-2)	7B/13B/70B	✓	38	https://huggingface.co/meta-llama
	LlamaGuard	7B	✓	1	https://huggingface.co/meta-llama/LlamaGuard-7b
	InCoder	1B/6B	✓	4	https://huggingface.co/facebook/incoder-1B
LMSYS	Vicuna	7B/13B	✓	12	https://huggingface.co/lmsys/vicuna-7b-v1.5
LianjiaTech	BELLE	7B/13B	✓	1	https://github.com/LianjiaTech/BELLE/
Databricks	Dolly	6B	✓	3	https://huggingface.co/databricks/dolly-v1-6b
–	Guanaco	7B	✓	2	https://huggingface.co/JosephusCheung/Guanaco
Salesforce	CodeGen(-1,-2)	3B/7B/16B	✓	9	https://github.com/salesforce/CodeGen/
	CodeT5	6B	✓	3	https://huggingface.co/Salesforce/codet5p-6b
BigCode	StarCoder(-1,-2)	3B/7B/15B	✓	3	https://huggingface.co/bigcode/
THUDM	ChatGLM	6B	✓	8	https://github.com/THUDM/ChatGLM-6B
KaistAI	Prometheus	7B/13B	✓	1	https://github.com/kaistAI/Prometheus
MistralAI	Mistral	7B	✓	6	https://huggingface.co/mistralai/Mistral-7B-v0.1
	Mixtral	8*7B	✓	5	https://huggingface.co/mistralai/Mixtral-8x7B-v0.1

Figure 2.5: LLMs in Cybersecurity [7]

The notation contains four categories related to the process modeling: *Flow Objects*, *Connecting Objects*, *Swimlanes*, and *Artifacts* [30]. To further understand them, a brief overview is provided to each category in the following subsections.

2.5.1 Flow Objects

Flow Objects are the main element for creating a process. They contain Activities, Events, and Gateways that are further split into multiple representation versions [8].

An **Activity** includes a single action or a subprocess. The connection of subsequent activities makes up the process, starting with a start and end event. Activities are further used to represent communication between pools and tasks done through systems or a script [8].

Events set the boundaries for the process. They include the start event, *i.e.*, what is triggering a process, as well as what event will end a process. In addition, intermediate events often represent messages, errors, time, and delays [8].

The **Gateway** is the last Flow Object and is used to represent decisions. It supports processes which include *branching, forking, merging or joining of paths* [8], [30].

2.5.2 Connecting Objects

In BPMN, there are two distinguished Connecting Objects to represent the direction of a process. **Sequence Flows** represent the sequence of connected Flow Objects through an arrow and can only be used within a pool. **Message Flows** are used to show the direction of information or message from one Flow Object to another Flow Object in a different pool [8].

2.5.3 Swimlanes

A **Pool** represents an organization and consists of one or more **Swimlanes**. Each swim lane represents an actor participating in a process. An actor can be a role, system, or user capable of executing Flow Objects and therefore participating in the process [8], [30].

2.5.4 Artifacts

The information used within a process is represented via **Artifacts**. They do not have a direct impact on the process, but they help visualize that there is a connection between information and a Flow Object (*e.g.*, guidelines are in place for a certain Activity). This information can be modeled as "*data objects, data stores, groups, and annotations*" [8], [30].

Chapter 3

Related Work

After explaining the theory behind the topic of insider threats, LLMs, and business process modeling, this chapter reviews related studies and technologies. Their synthesis reveals a research gap that motivates the architecture and implementation of the prototype.

The basis for the topics and research questions discussed in this work is [8] with its contribution of an insider threat knowledge base, a threat modeling prototype, and an evaluation in a practical setting. The literature review of [8] discovered a research gap, namely that the process view does not exist for threat modeling with a focus on insider threats. Therefore, the thesis proposed BPMN as a framework for a process-based perspective on insider threat modeling [8].

As a starting point, the relevant works found in the literature review of [8] were picked and extended with more recent publications in the domain of threat modeling of insider threats. Furthermore, the use of AI in these topics was investigated in more depth. The following Table 3.1 provides a comprehensive overview of the related work investigated with a comparison of their view and dimensions, the methodology used, their proposed prototype, and if there is a focus on insider threats and the use of LLM. The findings are further discussed in the sections of this chapter to identify research gaps and limitations.

Table 3.1: Overview of the Related Work

Source	View	Dimension	Methodology	Prototype	Insider only	LLM
[9] 2014	Threat Modeling	Process	Fault Tree Analysis, Finite State Verification	Automated fault tree analysis tool	yes	no
[10] 2014	Threat Monitoring	Process	analyze business process logs, social media behavior	social media & process monitoring tools	yes	no
[11] 2020	Threat Modeling	Process	SQUARE	no	no	no
[12] 2017	Threat Modeling	Process	SecBPMN-ml, SecBPMN-Q	query engine	no	no
[13] 2023	Threat Modeling	Process	ENISA, OWASP	BPMN modeler with annotations	no	no
[14] 2021	Threat Modeling	Process	NIST known vulnerabilities list	coreLang	no	no
[15] 2024	Threat Modeling	Process	MARISMA framework	BPMN-based threat modeler	no	no
[6] 2024	Threat modeling	Information System	Threat Manifesto	KeyBert, Llama 2	no	yes
[16] 2024	Threat Modeling	Threat Reports	Attack Graphs, CVEs	CrystalBall	no	yes
[17] 2024	Threat Modeling	DFD	Microsoft CoT, LoRA	TMT, OPRO, ThreatModeling-LLM	no	yes
[18] 2024	Business Process Modeling	Process	-	BPMN-Chatbot	no	yes
[19] 2024	Business Process Modeling	Process	-	LLM-Based Process Modeling Framework	no	yes
[5], [8] 2024	Threat Modeling	Process	insider threat mapping to business process elements	BPMN.io modeler	yes	no
This work 2025	Threat Modeling	Process	insider threat modeling on business process elements with RAG	BPMN.io modeler, Claude Opus 4	yes	yes

3.1 Threat Modeling Using Business Process Models

In the domain of threat modeling, various approaches were found in the literature review of [8]. They are either organizational, information system-based, or process-based. Table 3.1 aggregates the more closely related work. Within the table, all papers focus on threat modeling with the exception of [10]. [9], [11]–[14] validate the process-based view as an approach to analyze and detect threats.

A closely related prototype based on BPMN is proposed by [13]. The approach uses a BPMN model designed by a business process expert. This model is then analyzed in the context of an ENISA methodology, which focuses on the entirety of cyber threats. Finally, the identified threats are prioritized with the OWASP Risk Rating methodology. Countermeasures or the integration of an LLM are not part of the prototype. Other papers, such as [9] or [10], do focus on insider threats, with other models than BPMN to analyze threats within their prototype [13].

In general, the analysis of [8] showed that BPMN is a valid approach to support threat modeling methodologies. Additionally, this process view can help to find insider threats in the early process of implementing a system [11]–[14].

In [15], similar to the prototype in [8], a threat modeling approach focusing on BPMN with an automated threat identification is introduced. The underlying MARISMA framework consists of the MARISMA methodology, the eMARISMA infrastructure, and the threat patterns. The MARISMA methodology is a metadata structure with the considered threats, mainly focusing on threats within the process modeling and organizational structure, without an emphasis on insider threats. The eMARISMA supports the risk assessment by defining and reusing risk components. Its proposed prototype is tested in a healthcare scenario. However, the support of AI in eMARISMA is not discussed and neither is the focus on insider threats [15].

3.2 Threat Modeling Using LLMs

In the literature review of *threat modeling* and *LLMs*, various works were found. Some of them, for example, [31] or [32], discuss threat modeling approaches that investigate the potentially dangerous use of LLMs. Others explore methodologies where LLMs assist in threat modeling, streamlining the process for security experts [6], [16], [17]. Since the latter align more closely with the objectives of this work, these studies are discussed in greater detail in the following sections. [31] and [32] are not as relevant to this project and therefore not considered in the overview or below.

As mentioned, [6] leveraged LLMs to support the threat modeling process. KeyBERT was employed for NLP tasks to ensure accurate input parsing, while LLaMA 2 and RAG were applied to extract threat vectors out of the system design using recent and accurate information. The input for RAG stemmed from the National Vulnerability Database (NVD), a repository of commonly known vulnerabilities of the U.S. government. The

study found that 75% of the generated threat model aligned with human expectations, with RAG outperforming the LLM [6].

Similar to [6], [16] utilized LLMs with a RAG approach to generate attack graphs. Instead of taking information of the NVD, they based the RAG inputs on threat reports. Specifically, they leveraged GPT-4 in conjunction with RAG to associate attack data with MITRE Corporation’s Common Vulnerabilities and Exposures (CVEs), enabling the automated creation of detailed attack graphs. The approach called CrystalBall showed that leveraging LLMs with RAG is a valid approach for threat modeling [16].

While the tools of the previous works can be generally used, [17] introduced a novel approach called *ThreatModeling-LLM*, a threat modeling framework specifically developed for the banking sector. Their contributions include 1) the creation of a dataset for banking threat vectors with the Microsoft Threat Modeling Tool (TMT), 2) prompt engineering using the Chain of Thought (CoT) and Optimization by PROmpting (OPRO) technique, and 3) model fine-tuning by applying Low-Rank Adapation (LoRA). The evaluation of *ThreatModeling-LLM* resulted in significantly more accurate threat identification and mitigation generation. These findings contributed to the evidence of an effective threat modeling process that is automated and of great use in the banking sector [17].

These works in academia demonstrate the effective use of LLMs for facilitating threat modeling [6], [16], [17]. In addition, RAG improves the process even more, as mentioned in [6], [16]. However, none of the authors used business process models as input for their methodology, and they include many different attack vectors, not only insider threats.

3.3 Business Process Modeling Using LLMs

Besides threat modeling, a literature review was conducted to examine LLMs supporting business process modeling. [18] and [19] are examples that demonstrate how LLMs can help process engineers for modeling their business processes.

[18] developed a tool that creates business process models from natural language. The parser creates chunks of the words, which are then sent to an LLM that is prompted to generate a JSON object describing the business process. Next, the JSON is read by the tool and translated to XML format so that the BPMN model can be created and, lastly, visualized with the bpmn.js JavaScript library. Acceptance tests proved the effectiveness and usefulness of the prototype. Moreover, it showed a correctness rate of 95% on average [18].

A similar approach is applied by [19]. However, instead of JSON as an intermediate process representation between natural language and the BPMN model, they use Partially Ordered Workflow Language (POWL). The reasons for choosing POWL are its soundness, simplicity, and expressive power. Next to generating a business process model, the tool allows for error handling and feedback loops with the LLM. The evaluation undermines the robustness of the methodology and concludes that theoretically correct business process models are created [19].

These business process modeling tools have explored the use of LLMs and to what extent this support can be effective. Nevertheless, none of the approaches analyzes a process model in order to evaluate insider threats using LLMs.

3.4 Limitations

There are different approaches in academia that either focus on modeling insider threats, conducting threat modeling with business processes, or focusing on the use of LLMs in combination with threat or business process modeling. However, there is no methodology or prototype combining all of these dimensions by analyzing business processes for threat modeling using LLMs while focusing on insider threats.

Chapter 4

Architecture

The literature gap discovered in Chapter 3 is the basis for the design of the prototype. Defining its architecture first required evaluating the effectiveness of local LLMs. Based on this testing, a suitable workflow and system design were developed. The following sections describe the process until the methodology and the basic design of the prototype were defined.

4.1 Feasibility of LLM Integration

Currently, many LLMs are available that can run on local machines using tools such as Ollama. With them, the smaller models (*e.g.*, 7B or 8B) can be downloaded to a commonly used laptop with a RAM size of approximately 8-16 GB. Such tools make AI features more accessible and open the door for many applications to use LLMs as a support to their services [33].

In order to gain a deeper understanding of how LLMs work, the models were tested. The underlying goal was to make sure that the LLMs were capable of fulfilling the following requirements:

1. Understand how a business process works from parsing a BPMN file.
2. Extract relevant elements from the business process.
3. Find possible security threats that could occur in the process.

Different LLMs were selected to compare their performance across different architectures and sizes, with a focus on their usability in local environments. For this evaluation, a simple Python file, which is displayed in Listing 4.1, was created that started the Ollama client with a defined model, took as input the .bpmn file and a prompt, and then generated the response. At a later stage, the integration of the knowledge base was also tested.

Listing 4.1: Ollama Testing Code

```

import ollama

client = ollama.Client()

model = "OLLAMA MODEL"

with open("resources/testProcess.bpmn", "r", encoding="utf-8")
    as file: bpmn_file = file.read()

prompt = """PROMPT, {bpmn_file}"""

response = client.generate(model=model, prompt=prompt)

```

With this setup, various models (*i.e.*, *Mistral:7b*, *Llama 3.1:8b*, *Llama 3.2:3b*, *Deepseek-r1:14b*, *GPT 4.0*, and *Phi4:14b*) and different variations of the prompt were tested, which resulted in the following insights. The LLMs mostly reflected the tree structure of the XML document, rather than understanding the semantics of the underlying business process. To mitigate this, two strategies were applied: simplifying the XML and adding context beside the XML file. The XML file was simplified by reducing it to tags that described the business process, whereas the `<bpmndi:...>` tags were filtered out, because they only describe the coordinates of the elements in the canvas. For additional context, a sentence was added that briefly described the purpose of the business process.

After these adaptations, the locally-run LLMs improved significantly regarding the acceptance criteria. The models were generally able to understand the business process and its context, extract elements from the process, and identify suitable security threats, though still not with the desired level of accuracy. The authors therefore concluded that the LLM integration is feasible for implementation, but only with additional necessary steps to improve the output quality. These aspects are further discussed and evaluated in Chapter 5.

4.2 Proposed Methodology

After the LLM tests were successful, the methodology for the prototype could be developed. To enable users, regardless of their security expertise, to analyze business processes for insider threats, a five-step methodology was developed. Its goal is to guide users from defining their security objectives to receiving clear, context-specific threat insights and recommendations.

The methodology combines structured input from the user with the reasoning capabilities of LLMs. This enables automated and tailored threat modeling. Each step builds upon the previous one to ensure a targeted and meaningful analysis of the business process at hand. The methodology is structured into five sequential steps that combine user input with LLM capabilities to identify insider threats and is visualized in Figure 4.1.

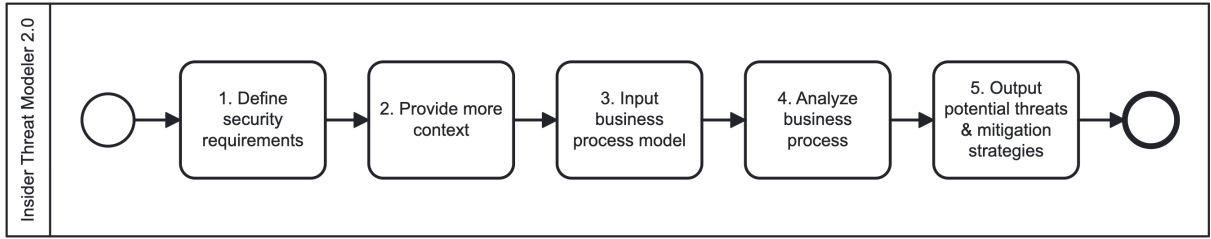


Figure 4.1: Five-Step Methodology

Step 1: Define security requirements

In the first step, the user selects the security objectives relevant to their business process. These objectives are based on the five security principles: *Confidentiality*, *Integrity*, *Availability*, *Accountability*, and *Authenticity*. This selection serves as a filter to ensure that only the most relevant insider threats are considered in the analysis.

Step 2: Provide more context about the business process

To help the system better understand the setting of the business process, the user answers a set of guided questions. These questions address the process itself, the roles involved, supporting systems, and its broader environment. This contextual information ensures that the analysis is tailored to the specific use case.

Step 3: Input business process model

In this step, the user uploads the model of the business process as a bpmn-file. Once uploaded, the system prepares the process model for analysis by extracting the relevant structural information and connecting it with the previously gathered user inputs.

Step 4: Analyze business process

The system then analyzes the business process with the support of a large language model. Each element of the process is reviewed considering the selected security requirements and the provided context. The system identifies insider threats that may occur at specific process elements and matches them to the user-defined security objectives.

Step 5: Output potential threats and their mitigation strategies

Finally, the system presents the user with a comprehensive overview of the potential insider threats identified in the process. Each threat is described in plain language besides the associated process element, the expected impact, and suggested mitigation strategies. This output provides practical, actionable insights to help users strengthen the security of their business processes.

Compared to [8], the methodology of this Master's project aims to reduce the dependency on a security expert and therefore simplifies it. Moreover, it adds the proposal of counter-measures and gives more detailed and tailored feedback on the business process at hand. These improvements aim to reduce the time required for security analysis and help users better understand the contextual relevance of the identified threats.

4.3 RAG Architecture

In Section 3.2, various sources showed that the use of RAG can enhance threat modeling by combining threat databases with LLMs. To be specific, the benefits are that the responses are more concise and contain more meaningful information [6], [16].

As stated before, this project builds on the foundation of [8], which contributed not only a threat modeling tool, but also a knowledge base consisting of a comprehensive list of insider threats. This database can be used for a RAG-based approach to feed the LLM more precise insider threats that can be retrieved when the answer is generated.

The architecture of the RAG approach is illustrated in Figure 4.2. First, chunks of the insider threat knowledge base are embedded in a vector database. Then, if the user starts the tool and enters security principles, a BPMN process, and information about the context is uploaded. With this information, the application conducts a similarity search in the vector database to check for possible insider threats from the knowledge base. Then, it processes the prompt and generates a response that is given back to the user.

The choice of a vector database over a relational database was made because it allows for a similarity search, which works effectively with natural language input, such as business process descriptions and context. A relational database would have required predefined queries to retrieve specific information, which is impractical when the business process structure is not known in advance. Vector databases, in contrast, support flexible similarity search using natural language input [34].

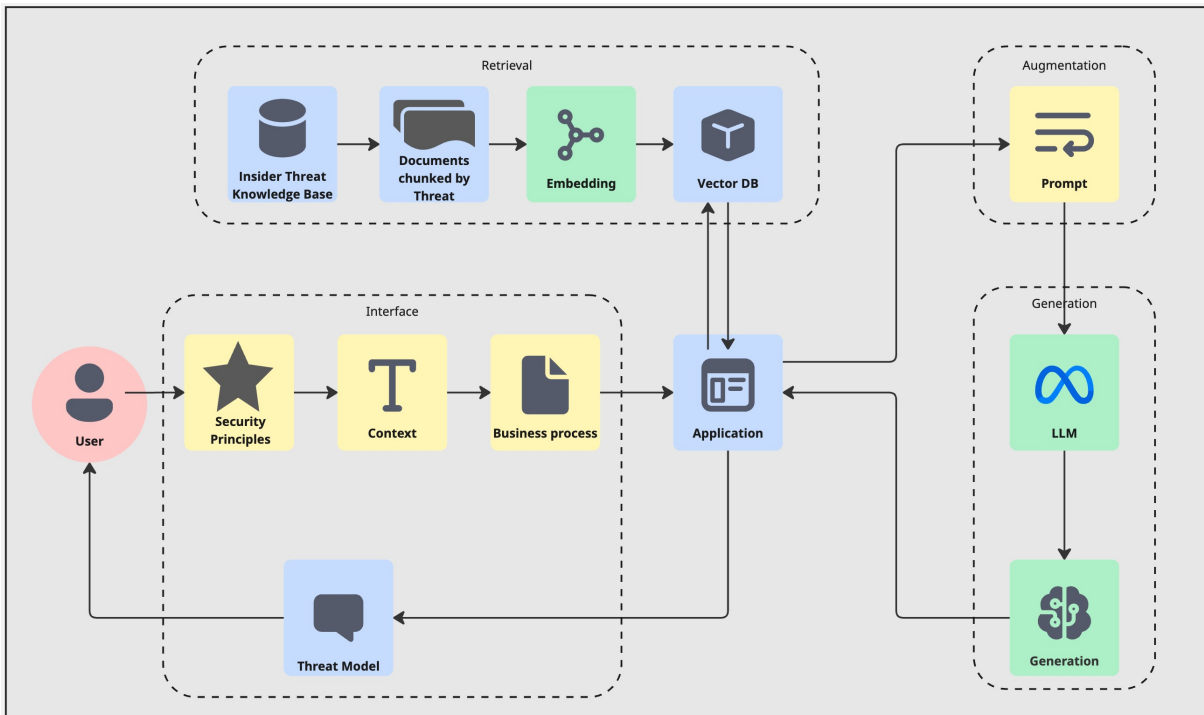


Figure 4.2: RAG Architecture

Chapter 5

Implementation

While the architecture chapter outlined the prototype at a high level, this chapter describes in detail how it was realized. It begins with an overview of the technology stack, and is followed by a step-by-step description of the user-facing flow. Then, the implementation details of the core backend modules, such as BPMN processing, RAG pipeline, and LLM implementation, are presented. Finally, prompt engineering, response handling, and frontend enhancements are discussed.

5.1 Technology Stack

The architecture described in Chapter 4 was implemented using a combination of frontend and backend components. The frontend was adapted from the tool developed in [8] with its existing technology stack. This includes JavaScript as the programming language, BPMN.io for parsing and displaying BPMN diagrams, and a user interface built with React. Extensions include a questionnaire about the context of the business process, a start page, and various UI changes.

In contrast, the backend was newly developed in the scope of this project. It establishes the connection to an LLM and implements the necessary components for the RAG pipeline. The backend handles BPMN file processing, prompt construction, LLM interconnection, and threat output generation. An overview of the most important technologies used in the system is provided in Table 5.1.

5.2 Implementation of the Methodology

The following paragraphs explain and visualize how each step of the methodology was realized in the prototype. On the start page, the user is instructed on how to use the prototype and what steps must be followed, as shown in Figure 5.1.

Table 5.1: Technology Stack Used in the Prototype

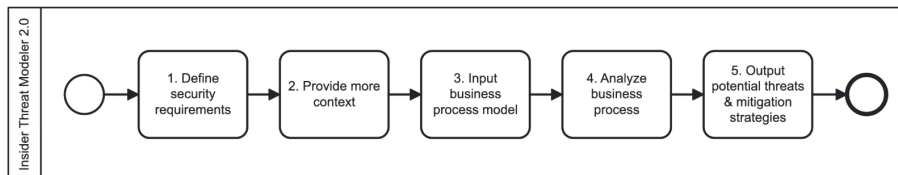
Component	Technology / Tool
Programming Language	Python 3.9 (backend), JavaScript (frontend)
LLM Runtime	Claude API
LLM Model	Claude Opus 4
LLM Orchestration	LangChain
Vector Store (for RAG)	LanceDB
Embedding Model	HuggingFace Sentence Transformers
Prompt Handling	LangChain PromptTemplate
Backend Framework	FastAPI
Frontend Interface	React 18
HTTP Client	Axios
BPMN Viewer	BPMN.io / bpmn-js

Insider Threat Modeler in BPMN 2.0

Welcome to the Insider Threat Modeler 2.0

This tool helps you to analyze business processes for potential insider threats. It is supported by a retrieval augmented generation architecture to provide tailored threats to the security goals and context of your business process.

Please follow the steps described below to analyze your business process:



Start

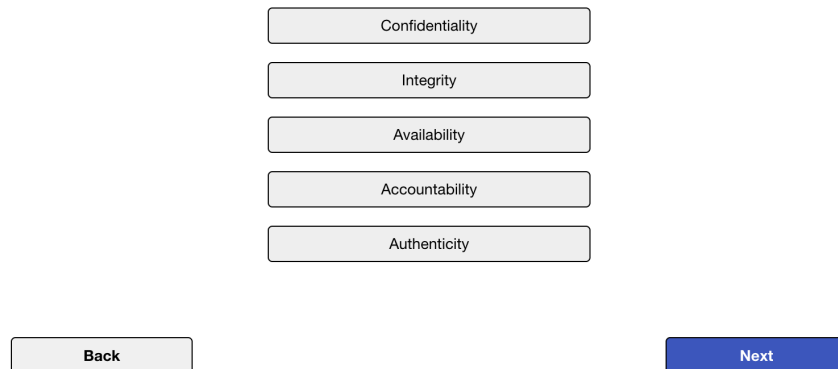
Figure 5.1: Start Page

Step 1: Define Security Requirements

As soon as the process flow is started, the user can select one or more of the five security principles: *Confidentiality*, *Integrity*, *Availability*, *Accountability*, and *Authenticity*. These principles, visualized in Figure 5.2, define the security objectives relevant to the business process. When hovering over a selection option, a tooltip is presented, so that the user understands the meaning of the security principle. Once the selection is made, the user proceeds by clicking the “Next” button. This step acts as a filter, narrowing down the set of insider threats to those most relevant, thus avoiding information overload.

Security Principle Selection

Please select the security principles that are the most important for your process.



The interface displays five security principles in a vertical stack, each within a light gray rectangular button: Confidentiality, Integrity, Availability, Accountability, and Authenticity. Below this stack are two navigation buttons: a light gray 'Back' button on the left and a blue 'Next' button on the right.

Figure 5.2: Step 1: Define Security Requirements

Step 2: Provide More Context about the Business Process

Next, the user is asked to answer a set of open text questions. These aim to capture additional context, including the purpose of the business process, involved roles, systems, and special considerations and seen in Figure 5.3. This contextual input is later incorporated into the prompt, which is sent to the LLM to ensure a more accurate and tailored threat analysis.

Process Context

Please answer the following questions to provide the prototype more context for the process you are analyzing.

1. What is the main goal of this process? Describe the steps.

Describe the purpose or intent and important steps.

2. What systems or technologies are involved?

List IT systems, applications, platforms, authentication etc.

3. Who are the primary roles or actors?

E.g., users, admins, external partners.

4. Other notes, risks, or special considerations?

Describe any other relevant context. E.g. high security assets, sensitive data



The interface shows two navigation buttons at the bottom: a light gray 'Back' button on the left and a blue 'Next' button on the right.

Figure 5.3: Step 2: Provide More Context about the Business Process

Step 3: Input Business Process Model

The user then uploads the business process model in `.bpmn` (BPMN 2.0) format visualized in Figure 5.4. Once uploaded, the process is rendered visually using BPMN.io in an interactive canvas, where the user can review or modify the model if needed. By clicking the “Show Threat”-button, the BPMN file, selected security principles, and contextual inputs are sent to the backend for analysis.

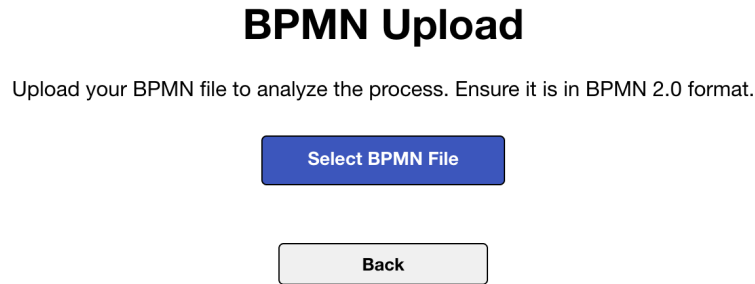


Figure 5.4: Step 3: Input Business Process Model

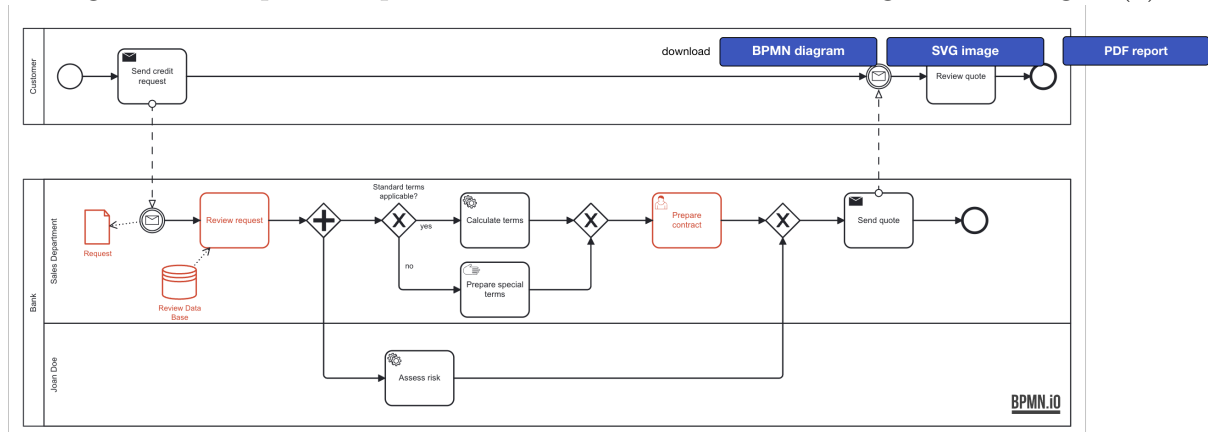
Step 4: Analyze Business Process

Upon receiving the input, the backend initiates processing through a **Query Builder** module. This module first cleans the BPMN file by removing irrelevant tags (e.g., `<bpm-ndi:...>`) to improve readability for the LLM. The contextual data and user-selected principles are formatted and embedded into a structured prompt. This prompt is then passed to the LLM, which analyzes the business process for potential insider threats. The response includes security principles, identified threats and their description, affected process elements, their potential impact, and mitigation strategies. Additionally, the LLM returns a list of element IDs corresponding to the vulnerable BPMN elements. These IDs are extracted from the markdown response and used for visualization purposes in the frontend.

Step 5: Output potential threats and their mitigation strategies

Finally, the markdown response generated by the LLM is displayed in the frontend as visualized in Figure 5.5 and Figure 5.6, providing the user with a structured overview of possible insider threats and recommended mitigation strategies. Figure 5.5 shows the top of the page with the highlighted BPMN diagram as well as the top of the generated output which continues further down on the page and is shown in Figure 5.6. The extracted element IDs are used to highlight the corresponding BPMN elements in the visual model, offering an intuitive connection between threats and specific parts of the process. In the upper right corner are three buttons, where the user can download the modified business process model in BPMN or SVG format or create a PDF report including the LLM answer.

Figure 5.5: Step 5: Output Potential Threats and their Mitigation Strategies (1)



Identified Threats

In the part below each threat found by the Insider Treat Modeler with its potential process elements, impact, and mitigation strategies are listed.

Review request

- **Potential Threat:** Confidential Data Acquisition
- **Principle:** Confidentiality
- **Potential Impact:** An insider reviewing requests may inappropriately acquire or use confidential customer or business data beyond the scope of their duties. This can result in unauthorized data exposure, information theft, or misuse, leading to reputational damage, regulatory violations, and loss of client trust.
- **Mitigation Strategies:**
 - Enforce strict access controls and role-based permissions to ensure employees can only access data necessary for their tasks.
 - Monitor and log all data access and retrieval activities for audit purposes.
 - Provide regular training on data privacy and confidentiality obligations.
- **Potential Threat:** Confidential Data View
- **Principle:** Confidentiality
- **Potential Impact:** Employees may view sensitive information in the review database or request details without a legitimate business need, risking exposure of confidential data and potential data leaks.
- **Mitigation Strategies:**
 - Implement data masking or redaction for sensitive fields not required for the review process.
 - Conduct periodic reviews of access logs to detect and investigate inappropriate data viewing.
 - Establish clear policies and disciplinary measures for unauthorized data inspection.

Prepare contract

- **Potential Threat:** Confidential Data Acquisition
- **Principle:** Confidentiality
- **Potential Impact:** During contract preparation, insiders may access, copy, or misuse confidential terms, customer data, or proprietary information, leading to unauthorized disclosure or competitive disadvantage.
- **Mitigation Strategies:**
 - Restrict access to contract documents to only those directly involved in preparation.
 - Use document management systems with detailed access tracking and version control.
 - Apply watermarking and digital rights management to sensitive contract files.

Figure 5.6: Step 5: Output Potential Threats and their Mitigation Strategies (2)

5.3 BPMN Integration

The initial tests with LLMs in Section 4.1 showed that when providing some context information, the LLM is able to reason about BPMN files [30]. For this reason and because BPMN is a standard for business process modeling, this format was kept as a constraint for using the threat modeling tool. Moreover, the literature review in Chapter 3 has shown that BPMN is a valid medium for threat modeling.

5.3.1 XML File Reduction

As there is a lot of information about the graphical location of the elements in the canvas, the XML file can become large and complicated, which increases the processing time and decreases the effectiveness of the LLM. The initial tests, previously described in Chapter 4, made evident that the BPMN file needs to be reduced to make it more readable for the LLM. The following method in Listing 5.1 cleaned up the tags that only describe the location of the different elements in space. It can be found in the file `query_builder.py`.

Listing 5.1: XML File Reduction Code

```
cleaned_xml = re.sub(
    r"<bpmndi:BPMNDiagram[^>]*>.*?</bpmndi:BPMNDiagram>",
    "",
    xml_string,
    flags=re.DOTALL
)
```

5.4 Retrieval-Augmented Generation

In Section 4.3, the architecture of the Retrieval-Augmented Generation (RAG) process was already explained. Here, a more detailed view is provided, including the names of the technologies that were implemented. Figure 5.7 illustrates the end-to-end workflow of the RAG pipeline implemented in the prototype.

The user interacts with the system via a frontend interface by selecting security principles, uploading a BPMN file, and entering context information in natural language. These inputs are sent to the backend application, where they get processed and included in a predefined prompt. The insider threat knowledge base is embedded in a vector database (LanceDB), which can be accessed during the retrieval phase. Hence, the most relevant threat documents are retrieved using similarity search and then passed to the LLM, which generates a structured threat model. The result, including the insider threats, vulnerable BPMN elements, and mitigation strategies, is returned to the frontend for visualization and review.

The backend implementation of the RAG pipeline is built using *LangChain*, a widely adopted framework for constructing applications around large language models. LangChain

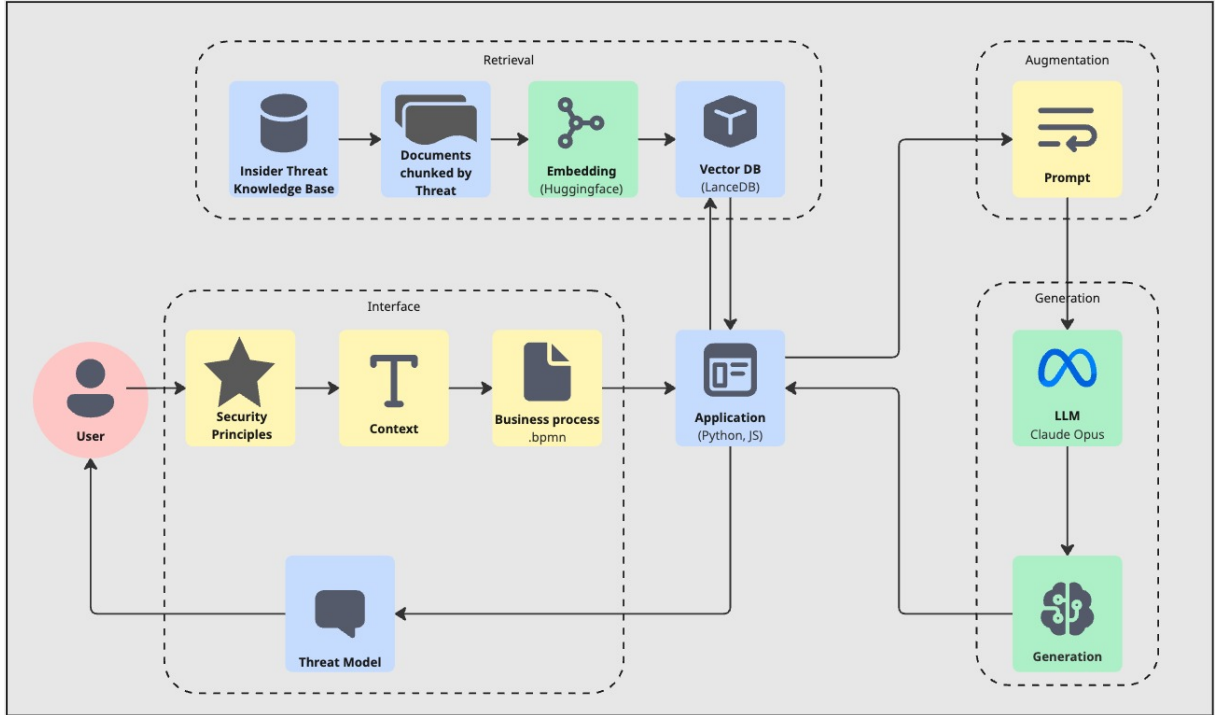


Figure 5.7: RAG Implementation

provides modular abstractions for combining prompts, memory, chains, and agents, which simplifies the orchestration of retrieval and generation components [35]. In this project, it is used to handle the embedding, storage, retrieval, and prompt composition processes in a structured and maintainable way. For instance, LangChain’s prompt template is used to handle input variables of the prompt.

5.4.1 Chunking

The insider threat knowledge base developed by [8] already followed a hierarchical structure, with each security principle linked to multiple insider threat types in a *1 to n* relationship. To preserve this structure during retrieval, a structured chunking strategy was designed to operate at the level of the individual threat groups. Each chunk includes the security principle, the name of the insider threat group, a detailed description, and a list of example threats gathered during the literature review in [8]. Therefore, the chunking follows the logic of the existing hierarchical structure of the knowledge base, where each vector points to an insider threat group with its security principle, description, and examples.

This approach ensures that the LLM has access to several specific example threats and context-specific information during the retrieval process, enhancing the quality and precision of the generated outputs. An example of a typical chunk used in the RAG pipeline is shown in the Listing 5.2 below:

Listing 5.2: RAG Chunk Example

Security Principle: Confidentiality

Insider Threat: Confidential Data Acquisition

Description: Data that is being stolen or used inappropriately is in this category. The attack can target not only a computer system but also a web service when for instance a session is being hijacked.

Example Threats:

- inappropriate acquisition of data
- inappropriate use of confidential data in computer system
- information theft
- session hijacking: no method for users to log out of an application / logging out does not clear session states / residual cookies
- exploitation of web services where identities are being hijacked & data is being stolen
- attacks on session-dependent information
- tampering with URL query strings (modify URL in browser's address bar to expose information to attacker)

5.4.2 Embedding

For generating the vector embeddings (*e.g.*, numerical representation) from the text, the *Hugging Face Sentence Transformers* library was integrated via LangChain. This framework provides access to a variety of pre-trained transformer models optimized for semantic similarity and retrieval tasks [36]. The specific model selected for this project is: `sentence-transformers/all-mpnet-base-v2`.

This model is widely regarded as one of the most effective general-purpose embedding models. It offers a strong balance between semantic accuracy and computational efficiency. Although it may not be the fastest among all sentence embedding models, it delivers high-quality results for similarity search [37].

5.4.3 Vector Database

For storing and retrieving the embedded chunks of the insider threat knowledge base, the vector database *LanceDB* was selected. LanceDB is an efficient and lightweight vector store optimized for local and embedded use cases [38]. It supports fast similarity search and integrates seamlessly with LangChain, which allows an efficient RAG implementation [39].

In this project, LanceDB stores the vector representations of the structured threat chunks. When a user submits a BPMN process model along with context information and selected security requirements, a query is constructed and embedded. This embedding is then compared against the stored vectors using similarity search to retrieve the most relevant insider threat chunks from the vector database. These threats are then used to enrich the prompt that is passed to the large language model.

The following code snippet in Listing 5.3 represents the initialization of the vectorstore, including the location (*i.e.*, path to the database), embedding, and the table (*i.e.*, insider threat database):

Listing 5.3: Vectorstore Initialization

```
vectorstore = LanceDB(
    connection=connection ,
    embedding=embeddings_model ,
    table=table)
```

5.5 Prompt Construction

The prompt is a crucial element of the RAG architecture, as it gives the LLM guidelines on how to process the information it is given. In this section, different elements of the prompt are described to show why they were introduced or what they should accomplish.

5.5.1 Iterative Improvement

The iterative prompt improvement process followed a PDCA (Plan-Do-Check-Act) cycle [40]. Initially, the existing implementation was tested using a predefined input, and the resulting output was analyzed. **Plan:** The weaknesses in the output, whether in structure, content, or other noticeable features, were identified and clear goals were defined for improvement. **Do:** The prompt was then adjusted to steer the LLM toward the desired outcome. **Check:** The prototype was rerun, and the output was evaluated against the criteria established in the planning phase. **Act:** Based on the analysis, the next steps were determined. If the output met the predefined goals, the focus was shifted to refining other aspects. However, if the results were unsatisfactory, either the planned adjustments were revised, or alternative approaches were explored. In some cases, previously tested strategies were abandoned in favor of more radical changes. Regardless of the outcome, the cycle concluded with further planning for additional improvements.

5.5.2 Input

The prompt takes as input the context information, the security principles, and the BPMN file that is needed for the analysis of the business process. The role of the LLM, as well as clear rules, are defined to tell the LLM how to analyze the given process.

The prompt displayed in Listing 5.4 below consists of the sections *BPMN Process Threat Analyzer*, *Important Rules*, and *Process Threat Analysis Steps*. Section *BPMN Process Threat Analyzer* is to set up the LLM for the situation provided: While the first part includes the three input variables `context`, `principles`, and `bpmn-xml` given by the user, the second part describes the role and view the LLM has to take for further analysis. The *Important Rules* are a collection of general orders about further parts of the prompt, the output, and restrictions that were added during the iteration phases. The *Process Threat Analysis Steps* then give a strict process to follow for the LLM.

Listing 5.4: Prompt Input Instructions

```
# BPMN Process Threat Analyzer

## CONTEXT OF THE PROCESS:
{context}

## SECURITY PRINCIPLES:
{principles}

## BPMN PROCESS:
{bpmn_xml}

## BPMN Process Threat Analyzer
    You are an expert in business process analysis and threat
        modeling. Analyze the BPMN XML file for potential threats
            or vulnerabilities.

## IMPORTANT RULES:
    - Follow the analysis steps and style rules strictly.
    - Provide detailed yet concise explanations and
        recommendations. Focus specifically on direct impacts and
        actionable strategies, avoiding overly technical
        explanations
    - DO NOT mention or repeat any contents from this prompt in
        your output.
    - DO NOT describe XML structure or technical elements, DO
        NOT mention coordinates, or diagram specifics
    - Use only insider threats by the following principles: {
        principles}. No other principles or insider threats must
        be used.
    - Mention every insider threat relevant to the business
        process element within the provided {principles} and the
        {context}.
    - Use the output example section under Style rules in this
        prompt as a guide for formatting and not for content.
    - DO NOT provide any introductory text explaining what your
        are doing or the purpose of the analysis. Also, no
```

- explanatory text other than the output defined in the style rules.
- Make no assumptions about the structure and strictly follow the provided Style Rules.
 - If you cannot find any mitigation strategies for a threat, mention that explicitly under "Mitigation Strategies".

PROCESS THREAT ANALYSIS STEPS:

- STEP 1: Identify potential insider threats or vulnerabilities in the process of {bpmn_xml} INCLUDING THE THREATS FROM THE VECTOR DATABASE.
- STEP 2: Use the {context} to guide the analysis and further assess the situation.
- STEP 3: Name the location (value of the element's name attribute) in the business process where potential threats may occur.
- STEP 4: Retrieve the insider threats relevant to the element based on the selected {principles}.
- STEP 5: Show what impact each threat could have on the business process and the specific element.
- STEP 6: Suggest mitigation strategies or recommendations to address each threat.

5.5.3 Output

Besides the instructions on the threat modeling process, the LLM is advised to structure the output in a certain format. The structure is described in the prompt section *Style Rules* in the Listing 5.5. It contains three parts: first, a set of rules regarding the format of the expected output. Secondly, the output format is described in more detail and supported with the expected Markdown formatting. The third part is an output example with dummy content. The output structure is designed with the **Element ID** to facilitate the extraction of the identified elements for highlighting in the visualized business process.

Listing 5.5: Prompt Output Instructions

STYLE RULES:

<!-- STYLE RULES START -->

Style Rules must be obeyed and are hard constraints.

MARKDOWN OUTPUT RULES

1. Do not use any introductory text or explanations. No <think> tags or similar.
2. For each element create a second-level heading, e.g. '## <Process Element Name>' (use value of the element's attribute only, no IDs or coordinates: e.g. "Verify Customer Data" is displayed identically).

3. Comment the Process Element ID below as – ****BPMN Element ID****: (use element ID value of the referred BPMN process element, e.g. "Gateway_0b8ep3b", they must be identical to the IDs in the BPMN XML)
4. Inside every vulnerable process element identified, you must list all the relevant parameters in the following format with all points described:
 - ****Potential Threat****: (name of the insider threat, e.g. "Data Corruption")
 - ****Principle****: (the principle name where the potential threat belongs to, e.g. "Integrity")
 - ****Potential Impact****: one paragraph, not within the same line
 - ****Mitigation Strategies****: bullet list
5. No tables, code fences, or raw XML.
6. Leave a space between each element described to improve readability.
7. Order the elements by the order they appear in the BPMN XML.
8. Use Markdown for formatting, no HTML tags or other formats.
9. The final output must be a valid Markdown document, ready to be rendered. There is NO introductory text and no conclusion.
10. If no threats are found for an element, do not include that element in the output.

Output Format:

- Element-based Output Organization: Organize your threat analysis clearly by the as vulnerable identified process elements. Begin each section with the element's name.
- Comment the choosen Element ID as following – ****BPMN Element ID****: [ID of the element, e.g., "Gateway_0b8ep3b"]
- Explicit Element Identification Format: For each element seperately, clearly list each the following parameters in the identical format and repeat for each threat identified within an element:
- ****Potential Threat****: [List of insider threats of the element affected, layout according to Style Rules]
 - ****Principle****: [Name of the principle from {principles}, e.g., "Integrity"]
 - ****Potential Impact****: Concisely describe the business process disruption or risks posed by this threat.
 - ****Mitigation Strategies****: Provide at least one practical recommendation per threat. Clearly link each mitigation to the specific threat identified.

Example Output:

- Verify Customer Data
- ****BPMN Element ID****: Activity_1ehwt8q
- ****Potential Threat****: Data Corruption
- ****Principle****: Integrity
- ****Potential Impact****:
 - Data corruption while verifying customer data can lead directly to processing errors such as incorrect billing, delays in customer transactions, compromised data accuracy, loss of customer trust, and potential regulatory non-compliance issues. Such disruptions significantly impact operational reliability and may result in financial penalties.
- ****Mitigation Strategies****:
 - Implement real-time data validation and error-checking protocols at each identified vulnerable step.
 - Conduct regular backups and schedule periodic data integrity audits to ensure data accuracy.
 - Introduce mandatory multi-factor authentication (MFA) for all critical data-access points to prevent unauthorized manipulation.

<!-- STYLE RULES END -->

5.6 LLM

The LLM integration was initially done with Ollama, an open-source tool used to run LLMs on local machines [41]. The following code snippet in Listing 5.6 shows how the LLM is initialized. The model is set in the `config.py` file of the backend.

Listing 5.6: Ollama Initialization

```
def build_qa_chain(vectorstore):
    retriever = vectorstore.as_retriever(search_type="similarity", k=5)
    llm = ChatOllama(model=LLM_MODEL)
    return RetrievalQA.from_chain_type(llm=llm, retriever=retriever,
                                       chain_type="stuff", return_source_documents=True)
```

At a later stage, the setup was changed to an cloud-based LLM, namely Anthropic’s model Claude Opus 4. In Section 6.1, an explanation for this is provided, as several LLMs were qualitatively tested and compared. Its integration requires, in addition to changing the definition of the LLM model in Listing 5.6, an API key, which is set in the `config.py` file, shown in Listing 5.7.

Listing 5.7: Claude Initialization

```
LLM_MODEL = "claude-opus-4-20250514"

ANTHROPIC_API_KEY = <APIKey>
```

5.7 Processing of the LLM Response

The response of the LLM is structured as a markdown as described in the prompt in Section 5.5. Along with the information that is given back to the user, the element IDs are part of the LLM output. Such that the highlighting of the BPMN elements in the frontend works, it is important that the structure of the output is strictly followed by the LLM.

In the file called `response_processor.py` a RegEx filters out all the element IDs and saves them in an array. Because the LLM sometimes hallucinated and fabricated IDs that did not exist in the BPMN file, it was decided to only take the suffixes of the element IDs (e.g., `123abc` instead of `Task_123abc`). The following code snippet in Listing 5.8 shows how this is implemented.

Listing 5.8: Element ID Filter Code

```
# Locate '**BPMN Element IDs**' blocks
pattern = r"- \*\*BPMN Element ID[s]?*\*\*:s*(.+) "
matches = re.findall(pattern, llm_response, flags=re.
    IGNORECASE)

suffixes = []
for match in matches:
    # Match the ids that after an underscore
    found = re.findall(r"[A-Za-z]+-(\w+)", match)
    suffixes.extend(found)
```

The response of the LLM, as well as the element IDs, is sent to the frontend. There, the answer is shown to the user in the interface by applying `ReactMarkdown`. The reason for visualizing the markdown structure is that the LLMs did not always return the exact structure that was specified in the prompt. For instance, sometimes the security principle was not a heading, or the BPMN elements were not shown as a list. This increased the difficulty of creating regular expressions for extracting and styling headings. With `ReactMarkdown`, the answer of the LLM can simply be visualized in the frontend.

5.8 Frontend Improvements

Compared to the prototype introduced in [8], the Insider Threat Modeler 2.0 has undergone several changes that should improve the value for the user. Part of those changes

include additional features outlined in the subsections below, as well as minor layout adjustments. At the same time, certain features like the threat selection and collection of relevant elements, or the visualized ranking of vulnerable elements were removed. The intent was to reduce complexity in the tool, decrease the time of a security expert needed, and create an understandable process flow. The orientation for the following UI features is based on the common understanding of usability in applications.

5.8.1 Routing

The ability to go back in the process emerged as an important feature, especially as additional steps were added to the prototype to reduce complexity and enhance the user flow. Hence, after uploading the BPMN file, the user can, by clicking on the designated “Back“-buttons, still return to the previous page and *e.g.*, change the answers in the context questionnaire.

In addition, a start page was added as an introduction to the prototype. It shows the process steps, helps the user understand what to do or which information is needed, and creates a sense of position while using the tool.

5.8.2 Descriptions

On each page, the user is now informed about how to use the tool in order to guide them through the entire process. Additionally, on the context page, each answer box has a placeholder text to support the questions further and to ensure the context is provided in more detail, which ultimately influences the final output quality and value for the user.

To further improve the usability of the prototype and guide the user, tool tips were added to the security principles page. Each principle has a hover effect that will display a short description of each listed principle, helping the user to decide what to choose. Also, after clicking on the “Show Threat“-button on the last page, a hint is displayed to indicate the user that the LLM is currently processing the request. Due to the integration of a cloud-based LLM, the waiting time has been drastically reduced in this step.

Chapter 6

Evaluation

This chapter describes the evaluation of the developed prototype in terms of functionality, threat coverage, accuracy, and user-centered output. The evaluation focuses on how effectively the system identifies insider threats in business process models. First, the performance of different LLMs is analyzed to assess their impact on output quality and quantitative measures. Then, an expert evaluation is conducted by interviewing two security consultants. Finally, the results are discussed and compared to a previously developed version of the *Insider Threat Modeler*.

To provide a meaningful baseline, the process scenario from the case study introduced in [8] is reused. This scenario was developed in collaboration with security and domain experts and served as the foundation for evaluating the first version of the Insider Threat Modeler [8]. The business process used is visualized in Figure 6.1 and shortly described in the following paragraph:

The business process displayed in Figure 6.1 deals with the issuing of insurance cards and numbers. An external member will trigger the process with a request. This will then be distributed either to an internal clerk or, if another office is responsible for the response's processing, to the responsible office. The external ZAS, a system for generating new insurance numbers, will provide new insurance numbers if necessary. Meanwhile the responsible clerk will request further information from the customer if needed. Finally, the clerk will send the insurance card to the issuing member and file the customer's case [8].

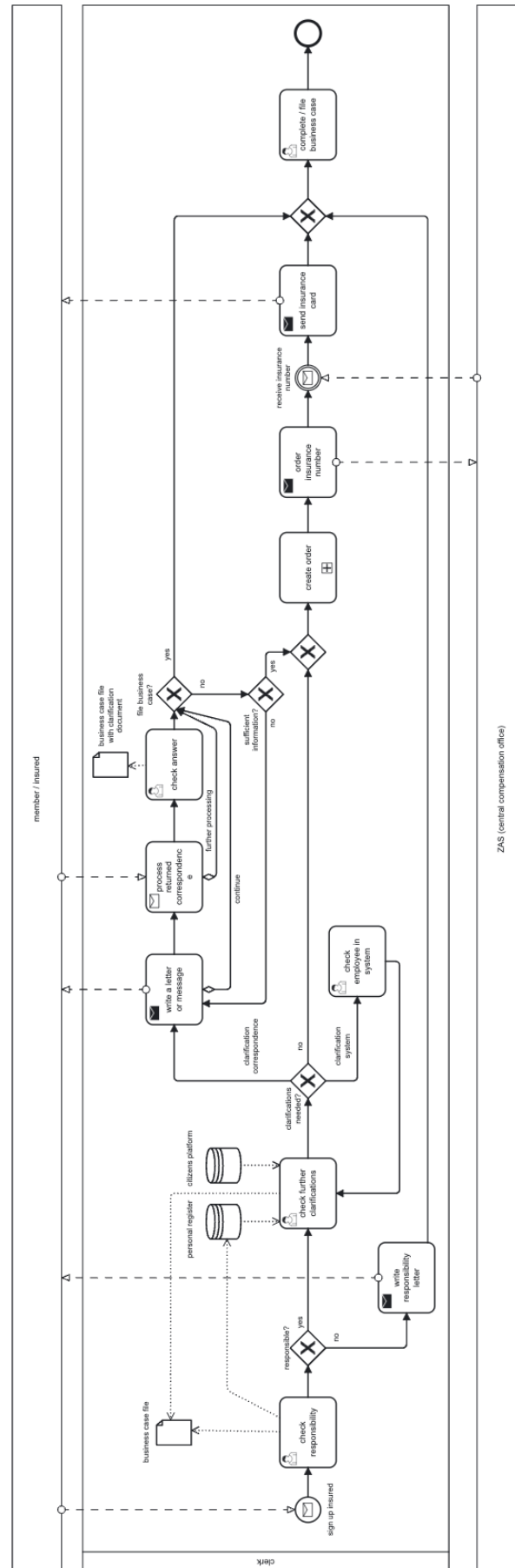


Figure 6.1: Business Process Used for Evaluation

6.1 LLM Comparison

To assess the impact of different language models on system output quality, several LLMs that can be run on Ollama were tested using identical prompts and input data. This comparison highlights the differences in threat relevance, clarity, and mitigation coverage across models.

6.1.1 Input Data

The input data for the evaluation was taken from [8], where a business process of an insurance company was analyzed. The process deals with the issuance of insurance cards and is further elaborated in the context questions below. It was chosen as it is a strictly controlled process including sensitive information. The three security requirements, Confidentiality, Integrity, and Availability, were originally set by the domain experts and served here as the selected security principles as well. Furthermore, the context questions were answered with the help of the interview taken in [8]. All the tests were conducted with the exact same answer set and can be inspected in further detail in the appendix A.1 [8].

6.1.2 Method

The methods chosen for the evaluation are a quantitative comparison as an overview for the reader and a qualitative assessment of different dimensions to give comprehensive feedback. The qualitative assessment takes precedence for the choice of LLM and is based on the relevance, clarity, and mitigation quality of each result. The quantitative comparison merely provides a first overview and uses consistent rubrics across all models.

Comparative Overview

The purpose of the comparative overview is to give the reader a comprehensive overview of the tested LLMs with their attributes. Six different LLMs were assessed by their parameters, integration method, performance, and number and quality of threats.

Table 6.1: Comparison of LLMs for Insider Threat Detection

Model	Parameters	Hosted	Time (min)	Identified Threats	Viable Threats
gemma3	4b	Device	9	11	3
deepseek-7b	7b	Device	11	27	4
deepseek-14b	14b	Device	4.1	30	9
llama3.2	3b	Device	0.4	15	4
GPT-4.1	n/d	Cloud	0.5	32	17
Claude	n/d	Cloud	0.9	23	22

Table 6.1 shows the different parameter sizes of the LLMs, with Llama 3.2 being the smallest model with 3 billion parameters and GPT-4.1 and Claude Opus 4 being the

largest with an undisclosed amount with estimates going from 300-500 billion for Claude Opus 4 to 1.8 trillion for GPT-4.1 [42]–[44]. The size also allows the smaller models to be hosted on personal devices and therefore, provides a further benefit to the security needs of the prototype, although cloud-based integration still provides a high level of security for business clients when implemented as the example in [45]. The time metric describes the average time in minutes that the respective LLM needed to fulfill a request. *Identified threats* is the number of threats the LLMs have generated over the final test request in all principles. They have then been evaluated and reduced to the *Viable threats*.

To qualify as a *Viable threat*, the proposed threat, in combination with its element, principle, and countermeasures, must fulfill all of the following rules: (1) The threat must belong to the requested principles. (2) The LLM must generate a known threat, principle, and element. Deviations or groupings are allowed as long as the attributes are still assignable and coherent. (3) A generated threat must not have more than one mistake in one element-threat-principle-countermeasure combination (most common mistake and acceptable: correct threat, wrong principle). The content must still be usable and coherent. (4) The generated threat must not be a duplicate within the same element.

Qualitative Comparison

With an understanding of the tested LLMs and their performance quantified, the analysis moves on to its qualitative assessment of their respective output. The output for all of the LLMs mentioned above is based on the same input, prompt, and technical setup described in Chapter 5 and the previous subsection. Each report is assessed if: (1) The output follows a given structure that is designed to improve readability and comprehension. This includes understanding the process through the context and identifying one or multiple elements in the process to address. (2) The generated threats are based on the given database structure of principles and insider threats. They do not have to copy the insider threats but understand the principles and provide a fitting insider threat within. (3) The output provides recommendations that are within themselves complete and correct, meaning the combination of element, threat, principle, and countermeasures is sensible. (4) The report proposes countermeasures that address the identified insider threat. (5) The output provides the intended value to the user. The user is a security expert and the solution should support them in their work.

The authors assessed the reports generated by the defined rules above in a qualitative discussion. The findings of the qualitative evaluation are the following:

In the assessment of **gemma3:4b** the positive aspects of the generated report are that it was successful in providing a comprehensive and generally concise structure. Some of the elements and threats were suitable and are correct and complete, meaning the LLM identified the element correctly, found a fitting insider threat with its principle from the database, and named adequate countermeasures to mitigate the mentioned insider threat. The majority of the identified threats lack the precision needed to provide value: They falsely name non-existent elements, the combinations of principles and threats from the database do not match, and are sometimes invented or heavily generalized, and lastly, they repeat already mentioned combinations of elements, threats, and countermeasures.

The smaller LLM from `deepseek-r1:7b` provides a comprehensive structure after providing an introduction representing its thinking process. The removal of any introduction is, however, explicitly mentioned in the prompt. It also underperformed due to having the same issues as `gemma3:4b` by inventing elements, failing to provide suitable principle-threat combinations, and repeating itself. Therefore, the value provided by its report is not adequate.

The larger version of deepseek, `deepseek-r1:14b`, also provides an introduction, which should have been omitted according to the output instructions. Otherwise, the output structure is strongly in line with the defined structure. It also identifies external systems and organizations as part of the process, which can be helpful and shows a greater understanding of the situation. Similarly, but slightly less often to the previous models, the LLM struggles to identify threats within the demanded principles and does not mention a single confidentiality threat, although there are several. Therefore, the larger deepseek model can also not provide the sought value to the security expert.

`llama3.2:3b`, the smallest LLM tested with 3 billion parameters creates reports with a concise structure. Otherwise, the content lacks the quality necessary to compete with the other LLMs. It mentions fictional elements from the prompt example, ignores the principles that are demanded, and also struggles with the combination of principles and matching threats, making its output unusable.

Switch to Cloud-Based Solutions

The testing of the locally-hosted LLMs was not sufficient to produce a report that could have been effectively used in a security expert environment after the qualitative evaluation. Therefore, the authors decided to test larger LLMs, namely OpenAI's GPT-4.1 and Anthropic's Claude Opus 4. This was used to evaluate whether adjustments of the implementation were needed to create reports that are adequate for a sensible expert evaluation.

The cloud-based LLM, `GPT-4.1`, is able to build a fitting structure according to the prompt and also generates multiple threats of the same principle for a single element. The content of the output largely avoids mistakes made by the smaller, device-hosted LLMs, like faulty principle-threat connections or incorrect element identification. It does repeat certain threats heavily and also uses generic threats like 'sabotage' without a clear description of what exactly the addressed threat should be.

`Claude Opus 4` has successfully generated a report based on the structure and rules given by the prompt. All identified threats are in themselves complete and correct with appropriate principles and countermeasures. It also avoids the mistakes previously made by the smaller LLMs. The principles 'Availability' and 'Accountability' have not been addressed as strongly as other principles. The output still provides sensible and usable content.

LLM Decision

After a careful comparison, Anthropic’s Claude Opus 4 is chosen as the most fitting LLM for the prototype. The quality of the output is significantly better than device-hosted models and also generates more sensible threats and countermeasures than its cloud-based counterpart, GPT-4.1. The use of a cloud-based LLM further benefits from a lower runtime while not using large amounts of the personal device’s system resources, which also provides additional value to the user. Since the prototype is a security-related tool, the European-based Anthropic solution is aligned with EU regulations for AI and data protection, providing additional value in this context [46].

6.2 Expert Evaluation

The expert evaluation aimed to assess (1) the correctness and completeness of the identified insider threats as well as their mitigation strategies, (2) the effectiveness of the methodology, and (3) the usability and perceived value of the prototype from a practitioner’s perspective. For this, two experienced cybersecurity experts, who have several years of professional consulting background, were interviewed.

In the first part, to assess the correctness and completeness of the prototype’s output, the experts were introduced to the chosen business process and then presented with the results generated by the prototype, including a high-level overview of the identified insider threats and the corresponding vulnerable process elements. In addition, they were given a detailed example of a specific process element, showcasing the full LLM-generated output, which included the detected threat, its security principle, potential impact, and proposed mitigation strategies.

In the second part, the overall architecture and methodology of the prototype were explained to experts to discuss their applicability in a security management context. Based on this, they were asked to analyze the RAG approach, the choice and configuration of the LLM, and the integration of contextual information.

The third part of the interview focused on usability and design. The interviewees were asked to provide critical feedback on the system’s interface, workflow, and the clarity of its contextual reasoning components.

Each interview was conducted individually and scheduled for approximately one hour. As time constraints limited the depth of analysis for every single threat and element, only a small selection of threats were shown and assessed in detail.

The interviews followed a semi-structured format with predefined questions, while allowing flexibility in how they were answered. The responses were documented in real time and audio-recorded for later analysis.

6.2.1 Output Evaluation

To generate a viable output from the prototype, the same business process, context information, and security principles were selected for the input as described in Section 6.1. The output of the prototype was then downloaded and prepared for the expert evaluation.

The preparation included two steps: first creating an overview of all process elements identified as vulnerable, together with the insider threats linked to them. Secondly, selecting a single process element and all threats associated with it for a more detailed discussion of the LLM-generated analysis.

Table 6.2 represents the overview of all threats per process element returned by the LLM. The summarized table view was only created for the evaluation and is not part of the output generated by the LLM.

In contrast, Table 6.3 shows the actual output received in the frontend and visualized for the user. For better readability, it is presented in a table instead of the markdown structure. For the interview, all four threats (*Unauthorized data access*, *Data exfiltration*, *Malicious code modification*, and *Resource exhaustion attack*) related to the process element *Check responsibility* were analyzed in detail. This element was chosen because all the security principles were represented. In this process step, a case worker checks whether their office is responsible for the case at hand. For better readability, only one threat is shown as an example here. More can be found in the appendix A.1 and A.2.

The output above was presented in the evaluation, such that the interviewees could respond to content and usability questions. For each interview question, the answers of both experts are summarized in the following paragraphs. For simplicity and anonymity reasons, the two security experts were given the names *Expert 1* and *Expert 2*.

1. For the vulnerable elements identified, are there any elements missing?

Both of the experts noted that there could be possible insider threats in the ZAS system. Also, they suggested that the event where the message is received from the ZAS called *Receive insurance number*, could be vulnerable to *e.g.*, injection attacks, as specified by Expert 1. Furthermore, Expert 2 pointed out that the *Personal registry* and the *Citizen platform* could be affected by data-related risks and that *Create order* could be exploited as well.

2. Are there any elements that are incorrect or irrelevant?

Neither Expert 1 nor Expert 2 identified any incorrect or irrelevant elements. Both experts agreed that the listed elements were reasonable within the scope of the threat model.

3. What insider threats are missing that you would have identified?

Expert 1 recommended including the risk of mass generation of social security numbers, which was not reflected in the prototype's output. Furthermore, the expert emphasized

Table 6.2: Assignment of Potential Insider Threats to Process Elements

Process Element	Potential Threats
Check responsibility	Unauthorized data access Data exfiltration Malicious code modification Resource exhaustion attack
Check further clarifications	Unauthorized data access Data exfiltration Malicious code modification Resource exhaustion attack
Write a letter / message	Information disclosure
Check employee in system	Unauthorized data access Malicious code modification Denial of service (DoS)
Check answer	Unauthorized data access Malicious code modification
Process returned correspondence	Unauthorized data access Buffer overflow attack
Create order	Malicious code modification
Order insurance number	Information disclosure Distributed denial of service (DDoS)
Send insurance card	Information disclosure
Complete / file business case	Data exfiltration Malicious code modification

that injection attacks at the task *Receive insurance number*, such as media disruption or issues related to interim data storage, should be taken into account.

Expert 2 raised concerns about scenarios in which responsible personnel might attempt to avoid assigned tasks or improperly forward requests. The expert also mentioned other risks, such as the introduction of false information in the process step *Write a letter*, and the possibility of identity theft within the task *Process returned correspondence*.

4. What insider threats are listed that you would not mention for the given element?

Both experts considered resource exhaustion attacks to be unlikely or to have a low impact in this context. Expert 2 stated that code manipulation was implausible, as caseworkers usually do not have access to the application's source code.

5. Are the scenarios describing the impact of insider threats realistic?

Expert 1 found most scenarios realistic but pointed out that some of the impacts, *i.e.*, legal consequences for the organization, are unlikely in practice. Private organizations often exclude such liabilities in their terms of service, and public institutions are typically protected by law.

Table 6.3: Example Threat for Process Element: *Check Responsibility*

Potential Threat	Unauthorized Data Access
Principle	Confidentiality
Potential Impact	A clerk with malicious intent could access sensitive personal information including birth certificates, IDs, and other confidential documents during the responsibility check process. This unauthorized access could lead to identity theft, privacy violations, data breaches affecting multiple individuals, and potential legal consequences for the organization. The compromised data could be sold or used for fraudulent insurance claims.
Mitigation Strategies	<ul style="list-style-type: none"> • Implement role-based access controls limiting data visibility to only necessary information for the task • Enable audit logging to track all data access and modifications by clerks • Apply data masking techniques to hide sensitive portions of documents unless specifically required • Require supervisor approval for accessing highly sensitive documents

Also here, both experts noted that DoS scenarios are unlikely in this particular business process, as internal bottlenecks can usually be addressed quickly given the manual setting of the process steps.

6. Which proposed countermeasures are adequate and meaningful, and which are not?

Both experts emphasized that audit logs are only effective if they are actively monitored. Moreover, they expressed concerns about the requirement of supervisor approvals too often, as it could lead to blind confirmation.

Expert 1 mentioned that request throttling is less effective than anomaly detection via monitoring. Regarding data exfiltration, he suggested that restrictions on copying data or taking screenshots must be supplemented by physical workspace controls, *e.g.*, phone-free zones or restricted viewing angles. For technical threats, he highlighted the importance of code reviews, access restrictions, and rollback capabilities, while questioning the necessity of version control if other safeguards are in place.

Expert 2's suggestions included disabling USB ports or encrypting removable media, limiting system access during off-hours, and prohibiting the use of personal cloud services. Additionally, he stressed the importance of thorough testing, enforcing the four-eyes principle during code releases, and ensuring that no code changes occur after testing.

7. Are the identified insider threats useful to your work as a security expert?

Expert 1 said the output serves as a valuable starting point. Rather than starting from scratch, a structured overview allows experts to apply their knowledge more strategically and focus on advanced issues.

Expert 2 emphasized that the output helps security professionals better understand the business process and associated risks. This understanding not only supports risk assessment but also supports the process of defining concrete countermeasures. From a consulting perspective, the ability to present clients with a documented and risk-annotated process was considered particularly helpful. He added that it aligns with common practices where security experts focus their evaluations on key control points.

8. What is your overall assessment of the usefulness of the results?

Expert 1 considered the results a strong foundation but noted that the output was not fully exhaustive. He advised that additional scenarios should be manually reviewed to ensure completeness. However, even though the threats were not always applicable, the countermeasures mostly fit the scenarios. Expert 2 found the output very helpful, especially because it allows experts to build upon an existing analysis rather than generating it on their own.

6.2.2 Methodological Discussion

For the second part, the interviewees were provided with information on the architecture and methodology of the prototype. The goal of these questions was to find out whether the prototype setup is suitable for a security management context.

1. Is the process in the prototype understandable and reasonable?

Expert 1 noted that the prototype's approach is generally understandable for security experts, but not immediately for end users or customers with no technical background. He claimed that the usability depends on who is responsible for filling in the input. Ideally, a combination of roles, such as the process owner, IT, management, and security experts, should be involved. He also suggested that the prototype should provide clearer explanations for certain decisions, such as the use of a specific LLM. Making these choices transparent would increase trust and improve user understanding.

Expert 2 confirmed the general validity of the prototype's approach, as it makes sense from a methodological point of view. However, he pointed out that currently only gross threats are visible, and it is not possible to account for controls that are already implemented. Showing net threats, after their mitigation, would add practical value and likely increase the acceptance of the tool.

2. From a process-oriented analytical perspective, what additional information beyond the business process is relevant and should be considered?

Expert 1 recommended adjusting the phrasing of questions to better fit the user's role and context. Moreover, instead of asking about the goal of the process, the business process itself should be thoroughly described. Expert 2 added that additional parameters such as industry, number of employees, and geographic distribution should be included. These can have a significant influence on risk exposure. The expert also suggested changing the question about the objective of the business process to what the process does.

3. Which input information is unnecessary or could be omitted without losing analytical value?

While Expert 1 did not state specific questions, Expert 2 believed that asking about potential risks directly is not necessary at this stage, as the tool is expected to identify such risks independently.

4. The LLM retrieves relevant insider threats from a vector database based on a static insider threat model. What risks and advantages does this setup present?

Expert 1 stated that this architecture is a fundamental component of such a modern security tool. However, Expert 2 pointed out the risk of incomplete threat coverage if the database is proprietary and based on limited training data. This can, on the other hand, also be an advantage because of reduced hallucination and improved output precision. Furthermore, both experts emphasized the need to regularly update the vector database to ensure comprehensive threat coverage.

5. The LLM suggests countermeasures based on identified insider threats and contextual information. What are the risks and advantages of this approach?

Expert 1 did not provide a detailed answer to this question. Expert 2 noted that standard countermeasures could be retrieved directly from the vector database. However, for newly emerging threats, the LLM's generative capabilities would be required to complement the missing data. For this, Expert 2 suggested a hybrid approach in which the LLM is used in conjunction with expert review and database queries of mitigation strategies.

6. The LLM is cloud-based for better performance and quality, but locally hosted LLMs would be more secure since the data is not used for training. Which solution is preferable in a business context and why?

Expert 1 noted that there is an ongoing debate among experts about this. He recommended giving users the option to choose between local and cloud-based models, which would increase trust and flexibility. Expert 2 argued that a cloud-based RAG approach allows for more current threats, as new threats can be integrated dynamically. However, he warned that data sent to the API must be anonymized to preserve confidentiality.

6.2.3 Usability Evaluation

Lastly, the experts were questioned about the appearance of the prototype and asked to assess its usability.

1. Are the process and actions needed in the prototype clearly structured, intuitive to follow, and easy to understand without prior training?

Expert 1 noted that the clarity of the workflow largely depends on the user's background. While the process is comprehensible for security professionals, customers or non-expert users would likely require additional explanations. On the other hand, Expert 2 argued that the processes are presented clearly and intuitively. However, the expert suggested enhancing the "Define Security Requirement" section by including a broader set of security objectives, which could better support users in aligning threat analysis with their specific needs.

2. Which design features could improve the usability of the prototype?

Expert 1 proposed adding more guiding questions and integrating drop-down menus for specific input fields. These could serve as mental support to help users formulate more complete and accurate answers, particularly for users who are less familiar with threat modeling. In contrast, Expert 2 stated that he perceived the design as good overall and warned of making it too cluttered.

3. Are there additional features that could improve the tool?

Expert 1 emphasized the importance of complementing the existing outputs with a description for each insider threat. Simply naming a threat can be insufficient or misleading for users to understand them. Therefore, a concise explanation or an illustrative scenario could provide context and improve comprehension.

Expert 2 noted that currently, only gross risks are provided. He suggested that it would be valuable if the tool could incorporate controls that allow users to determine net threats. For example, examining whether key controls are met could be integrated into the threat model. The expert also proposed that the tool could offer suggestions for optimizing controls in the business process. Furthermore, if a client does not have a BPMN model for its business processes, it would be helpful if the tool could generate BPMN diagrams directly from flowcharts or plain text, a feature that currently does not exist. The expert also mentioned the potential for process optimization within the tool.

4. Are there any other suggestions you would like to express?

Expert 1 suggested enhancing the look and feel of the tool, for example, by adding a side-bar for navigating between steps more easily. Expert 2 expressed a positive overall impression, noting that the tool is already well developed. However, the expert acknowledged that further iterations and refinements are both feasible and encouraged to elevate the practical value of the tool.

6.3 Discussion

The feedback of the security experts provided valuable insights into the strengths and limitations of the proposed methodology and prototype. This section synthesizes the findings of the evaluation, organized along three central dimensions: (1) the correctness and completeness of the identified threats and countermeasures, (2) the effectiveness of the underlying methodology, and (3) the usability and perceived value of the tool from a cybersecurity practitioner’s perspective.

6.3.1 Correctness and Completeness of the Output

Both experts confirmed the general correctness and contextual relevance of the threats and mitigation strategies produced by the prototype. The structured format, consisting of process element, threat, principle, impact, and countermeasure, was perceived as helpful for expert-level review.

However, limitations were also observed. Both experts questioned the realism of certain threats, particularly resource exhaustion attacks, which were perceived as less applicable to the evaluated process scenario. Furthermore, both experts identified gaps in the threat model itself. Examples such as task avoidance or mass generation of social security numbers were missing. Moreover, some elements in the BPMN were not mentioned in the report, suggesting that the LLM only considered tasks without including events or being aware of the specific type of the task (*e.g.*, user task or service task). Hence, while the model provides a strong baseline, it is not yet exhaustive.

Despite these shortcomings, the output of the prototype was seen as a useful starting point that can reduce manual effort and support higher-level analysis compared to the previous version of the *Insider Threat Modeler*. As the manual step required domain knowledge, context information, and security expertise to filter out irrelevant insider threats in the prototype of [8], in the current version, the LLM takes over this part. Furthermore, by identifying plausible insider threats with the help of more context and directly proposing mitigation strategies, the tool enables security experts to focus their expertise more strategically.

6.3.2 Methodological Applicability

The experts supported the value of using an LLM in combination with a RAG architecture. This hybrid setup was assessed as a strength, allowing the system to retrieve structured knowledge from a predefined insider threat database while still using the reasoning capabilities of the LLM to generate context-aware suggestions.

Regarding the cloud-approach used by choosing Claude Opus 4 as LLM, Expert 1 suggested a more flexible approach to let the user decide which LLM to use. Expert 2 brought up that the information sent to a cloud-based LLM would need to be anonymized, which would, on the other hand, decrease the output quality.

In addition, the prototype currently evaluates threats without taking into account security controls already in place in the organization. As a result, the output generated can overestimate the severity of the threat by presenting “*gross threats*” in the report. A more refined analysis that accounts for existing countermeasures and identifies “*net threats*” was recommended by Expert 2.

In summary, the methodology was perceived as clear and logical. When some more important information can be collected for the context and a process is introduced that allows for the regular incorporation of dynamic updates or expert reviews, the methodology already builds a strong foundation.

6.3.3 Design and Usability Considerations

The overall usability of the prototype was positively evaluated by both experts. The interface, structure, and clarity of the output were found to be suitable for security professionals. The tool was considered beneficial in their daily work as consultants, especially for documenting processes, identifying risks, and communicating security concerns to clients or stakeholders.

At the same time, the security experts highlighted that the prototype may be less intuitive for users without technical expertise. Additional explanations, guided input prompts, and design improvements could make the tool more accessible. Suggestions included adding drop-down menus for key questions, improving the layout with sidebars or navigational aids, and offering transparency about architectural decisions such as LLM selection.

6.3.4 Limitations of the Evaluation

The findings in this evaluation are subject to a number of limitations. In particular, only two cybersecurity experts were interviewed. While their feedback was rich and informative, larger participation would likely reveal additional domain-specific requirements and usability issues.

Furthermore, the evaluation was based on a single business process. Applying the methodology to a more diverse set of processes and organizational contexts would be necessary to generalize the findings and assess the scalability of the approach.

6.4 Refinement

Based on the expert evaluation in Section 6.2, the authors decided to implement two of the mentioned changes into the final prototype: (1) rephrasing of the first context question in the UI and (2) extending the output format by a threat description. The reason for those specific changes is their ability to improve the prototype small changes to the system’s architecture, and enhance the final solution while maintaining the integrity of the evaluated prototype.

6.4.1 Rephrasing of the First Context Question

The initial question in the UI was “1. What is the main goal of this process? Describe the steps.” The experts both pointed out that the intention here should be to receive detailed information about the process instead of the possible economic goals of the process. The current phrasing does not clearly steer a user into the intended description of the process steps. A rephrasing of the question would help improve the understanding of the LLM and ultimately influence the output provided. Therefore, the question was adjusted to “1. How would you explain the BPMN process, including the main steps and their purpose?”.

6.4.2 Extending the Output Format by a Threat Description

The second change is the extension of the output format by adding a threat description. As pointed out by Expert 1 during the questionnaire about the usability of the prototype, a short description of the identified insider threat helps provide a better understanding for the user. Since the exact meaning might not be clear to some users and the insider threats generated by the LLM may also differ from the insider threat knowledge base, the following additions in the prompt file `prompt_threat_analysis.py` were made and are shown in the Listings 6.1, 6.2, 6.3, and 6.4:

Listing 6.1: ‘Process Threat Analysis Steps’, line 39

```
STEP 5: Shortly describe the identified insider threat
within the context of the process under “**Threat
Description**.”
```

Listing 6.2: ‘Style Rules, Markdown Output Rules’, line 53

```
– **Threat Description**: (short description of the threat
identified under in the “**Potential Threat**”: above. Do
not include the impact since it is mentioned later on.)
```

Listing 6.3: ‘Style Rules, Output Format’, line 69

```
– **Threat Description**: [Short description of identified
threat]
```

Listing 6.4: ‘Style Rules, Example Output’, line 78

```
– **Threat Description**: A malicious insider may
intentionally alter or falsify customer data during the
verification step.
```

The changes in the prompt resulted in the new output format that is visualized in Figure 6.2, where the threat description now appears.

Review request

- **Potential Threat:** Confidential Data Acquisition
- **Threat Description:** An insider with access to the Review Data Base could inappropriately acquire sensitive customer credit information during the request review process.
- **Principle:** Confidentiality
- **Potential Impact:** Unauthorized acquisition of customer credit data could lead to identity theft, financial fraud, or sale of sensitive information to competitors. This breach could result in severe reputational damage to the bank, loss of customer trust, regulatory penalties, and potential lawsuits from affected customers.
- **Mitigation Strategies:**
 - Implement role-based access controls limiting database access to only necessary data for each review
 - Enable comprehensive audit logging for all database queries and data access attempts
 - Deploy data loss prevention (DLP) tools to monitor and alert on unusual data extraction patterns
 - Enforce the principle of least privilege for all employees accessing the Review Data Base

Figure 6.2: Refined Output Example

6.5 Comparison with Previous Work

The previous sections have assessed various dimensions of the prototype developed in this thesis. To conclude the evaluation, this section compares the current prototype and findings to [8], highlighting the enhancements achieved in this Master’s project.

Table 6.4 summarizes the key differences between the two works. This thesis introduces a broader and more advanced technology stack, integrates contextual information, and automates the generation of mitigation strategies using a RAG-enhanced LLM approach. As a result, it reduces manual effort and lowers the level of expert knowledge required from users. However, the prototype by [8] included a manual filtering step, allowing experts to adjust threat relevance based on already implemented controls, which is not supported in the current implementation.

The evaluation in Section 6.2 has shown that with the context information, the insider threats were more tailored to the situational context and the experts perceived the threats as plausible. However, in [8], no threats were claimed as false positives compared to this work, where the experts judged the threat *Resource Exhaustion Attack* as unlikely. This indicates the potential hallucination by the LLM.

Table 6.4: Comparison between Bachelor Thesis and Master Project Prototype

	Bachelor Thesis [8]	This Work
Methodology	6 steps	5 steps
Threat detection approach	Static mapping from BPMN to threats	LLM-driven threat identification with contextual input
Mitigation strategies	Not included	Automatically generated based on retrieved threat knowledge
Use of context	None	User-provided process context integrated into prompt
Output format	Basic threat listing	Structured markdown with linked BPMN elements
User guidance	Requires expert knowledge	Requires less expert knowledge
Technology stack	JavaScript, BPMN.io	JavaScript, BPMN.io, FastAPI, LangChain, LLM, LanceDB
AI model support	None	Claude Opus 4
Knowledge base use	Insider threat knowledge base	RAG-enhanced retrieval from Insider Threat KB
Control consideration	Expert can manually filter out elements with controls	LLM only considers information given in questionnaire
Priorization of elements	Number of threats per element	None
Output Detail Level	List of threats with description and principle	Full threat description with principle, impact, and mitigation strategies
Evaluation approach	Case study with expert interview	Qualitative evaluation of LLM and case study with expert interview

Chapter 7

Conclusion

The previous chapters described the design, implementation, and evaluation of the *Insider Threat Modeler 2.0*. A tool that combines insider threat modeling using business process models with the support of an LLM, a research gap identified in Chapter 3.

The main contributions include (1) a five-step methodology, that describes the process of the prototype, as well as the system architecture using a RAG-enhanced approach by embedding the knowledge base previously created by [8] into a vector database. Furthermore, (2) an enhanced prototype with improved usability, contextual reasoning, and suggested mitigation strategies is contributed. Additionally, (3) a comparison of different LLMs, and (4) an expert evaluation to assess the first two contributions mentioned above.

The qualitative comparison of various LLMs indicated that smaller, locally-hosted LLMs often struggled to fully comprehend the process from the provided input, resulting in generated reports that lacked the necessary accuracy. In contrast, larger, cloud-based solutions demonstrated significantly improved performance, producing well-structured and valuable outputs. For this reason, Anthropic’s Claude Opus 4 was chosen as it additionally offered a time-saving value when assessing new process scenarios.

The expert evaluation revealed that the threats and mitigation strategies generated by the prototype were generally accurate, well-structured, and valid, although some were unrealistic or incomplete. The RAG approach was perceived to effectively combine structured retrieval with contextual reasoning. Some enhancements were recommended by the experts such as incorporating control-aware modeling, enabling dynamic updates to the knowledge base, and allowing users to choose between different LLMs. Overall, the methodology and prototype design were considered well-suited for security professionals and helped reduce manual effort, while noting that additional guidance would make it more accessible to non-experts.

To conclude, this work demonstrates the potential of large language models to support the identification and mitigation of insider threats from a business process perspective. It further elaborates on potential future versions that could be used in a business setting and is already considered as a potential method of supporting cybersecurity experts in their work.

7.1 Future Work

Based on the expert feedback, several enhancements were proposed to improve the prototype in future iterations:

- **Control-aware threat modeling:** Introduce a mechanism for assessing already implemented security controls to refine gross threats into net threats and reduce over-reporting.
- **BPMN generation from text:** Enable the automatic generation of BPMN models from unstructured input such as text descriptions or flowcharts, supporting organizations without formal modeling tools.
- **Threat model expansion:** Provide descriptive scenarios or illustrative explanations for each threat to make them more understandable for non-experts.
- **Process optimization features:** Extend the tool to identify process inefficiencies or bottlenecks along with security issues, offering a dual perspective on risk and performance.
- **Dynamic updating or hybrid workflows:** Allow for regular updates to the threat knowledge base and incorporate hybrid human-AI workflows to combine expert validation with generative suggestions.
- **LLM Selection:** Give the user the ability to choose from different models, such that cloud-based and open-source LLMs can be selected.

The authors believe that future developments of the proposed enhancements could yield novel and valuable insights for the *Insider Threat Modeler*. However, it is important to acknowledge the technical limitations identified in this study. In particular, LLMs are prone to hallucinations and face challenges in accurately interpreting and understanding BPMN-based files, which currently constrains the system's overall capability. With fast improvement in the field of AI, those limitations might be reduced over time.

Bibliography

- [1] U. Inayat, M. Farzan, S. Mahmood, M. F. Zia, S. Hussain, and F. Pallonetto, “Insider threat mitigation: Systematic literature review”, *Ain Shams Engineering Journal*, vol. 15, no. 12, p. 103 068, 2024, ISSN: 2090-4479. DOI: <https://doi.org/10.1016/j.asej.2024.103068>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S209044792400443X>.
- [2] I. Homoliak, F. Toffalini, J. Guarnizo, Y. Elovici, and M. Ochoa, “Insight into insiders and it: A survey of insider threat taxonomies, analysis, modeling, and countermeasures”, *ACM Comput. Surv.*, vol. 52, no. 2, Apr. 2019, ISSN: 0360-0300. DOI: 10.1145/3303771. [Online]. Available: <https://doi.org/10.1145/3303771>.
- [3] K. Renaud, M. Warkentin, G. Pogrebna, and K. van der Schyff, “Vista: An inclusive insider threat taxonomy, with mitigation strategies”, *Information & Management*, vol. 61, no. 1, p. 103 877, 2024.
- [4] C. Caridi, J. Dwyer, G. Prassinis, *et al.*, *IBM X-Force Threat Intelligence Index 2024*, <https://www.ibm.com/reports/threat-intelligence>, Accessed: 2025-03-06, Feb. 2024.
- [5] J. Von der Assen, J. Hochuli, T. Grübl, and B. Stiller, “The danger within: Insider threat modeling using business process models”, in *2024 IEEE International Conference on Cyber Security and Resilience (CSR)*, IEEE, 2024, pp. 186–192.
- [6] I. Elsharef, Z. Zeng, and Z. Gu, “Facilitating threat modeling by leveraging large language models”, in *Workshop on AI Systems with Confidential Computing*, 2024.
- [7] J. Zhang, H. Bu, H. Wen, *et al.*, “When llms meet cybersecurity: A systematic literature review”, *Cybersecurity*, vol. 8, no. 1, pp. 1–41, 2025.
- [8] J. Hochuli, “Design and implementation of a support system for organizational controls in information security”, University of Zurich, 2024.
- [9] M. Bishop, H. M. Conboy, H. Phan, *et al.*, “Insider threat identification by process analysis”, in *2014 IEEE Security and Privacy Workshops*, IEEE, 2014, pp. 251–264.
- [10] V. Stavrou, M. Kandias, G. Karoulas, and D. Gritzalis, “Business process modeling for insider threat monitoring and handling”, in *Trust, Privacy, and Security in Digital Business*, 2014, pp. 119–131.
- [11] S. Zareen, A. Akram, and S. Ahmad Khan, “Security requirements engineering framework with bpmn 2.0.2 extension model for development of information systems”, *Applied Sciences*, vol. 10, no. 14, 2020.

- [12] M. Salnitri, F. Dalpiaz, and P. Giorgini, “Designing secure business processes with secbpmn”, *Software and systems modeling*, vol. 16, no. 3, pp. 737–757, 2017.
- [13] D. Granata, M. Rak, G. Salzillo, G. Di Guida, and S. Petrillo, “Automated threat modelling and risk analysis in e-government using bpmn”, *Connection Science*, vol. 35, no. 1, p. 2284645, 2023.
- [14] S. Hacks, R. Lagerström, and D. Ritter, “Towards automated attack simulations of bpmn-based processes”, in *2021 IEEE 25th International Enterprise Distributed Object Computing Conference (EDOC)*, IEEE, 2021, pp. 182–191.
- [15] D. G. Rosado, L. E. Sánchez, Á. J. Varela-Vaca, *et al.*, “Enabling security risk assessment and management for business process models”, *Journal of Information Security and Applications*, vol. 84, p. 103829, 2024, ISSN: 2214-2126. DOI: <https://doi.org/10.1016/j.jisa.2024.103829>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2214212624001315>.
- [16] R. T. Prapty, A. Kundu, and A. Iyengar, “Poster: Crystalball-attack graphs using large language models and rags”, in *2024 IEEE 44th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2024, pp. 1450–1451.
- [17] S. Yang, T. Wu, S. Liu, D. Nguyen, S. Jang, and A. Abuadbba, “Threatmodeling-llm: Automating threat modeling using large language models for banking system”, *arXiv preprint arXiv:2411.17058*, 2024.
- [18] J. Köpke and A. Safan, “Introducing the bpmn-chatbot for efficient llm-based process modeling”, in *International Conference on Business Process Management*, 2024.
- [19] H. Kourani, A. Berti, D. Schuster, and W. M. van der Aalst, “Process modeling with large language models”, in *International Conference on Business Process Modeling, Development and Support*, Springer, 2024, pp. 229–244.
- [20] D. Granata and M. Rak, “Systematic analysis of automated threat modelling techniques: Comparison of open-source tools”, *Software quality journal*, vol. 32, no. 1, pp. 125–161, 2024.
- [21] S. Hussain, A. Kamal, S. Ahmad, G. Rasool, and S. Iqbal, “Threat modelling methodologies: A survey”, *Sci. Int.(Lahore)*, vol. 26, no. 4, pp. 1607–1609, 2014.
- [22] M. Tatam, B. Shanmugam, S. Azam, and K. Kannoorpatti, “A review of threat modelling approaches for apt-style attacks”, *Heliyon*, vol. 7, no. 1, 2021.
- [23] W. Xiong and R. Lagerström, “Threat modeling – a systematic literature review”, *Computers & Security*, vol. 84, pp. 53–69, 2019, ISSN: 0167-4048. DOI: <https://doi.org/10.1016/j.cose.2019.03.010>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167404818307478>.
- [24] M. S. Jawad and M. Hlayel, “Intelligent cybersecurity threat management in modern information technologies systems”, in *Lightweight Cryptographic Techniques and Cybersecurity Approaches*, IntechOpen, 2022.
- [25] R. A. Alsowail and T. Al-Shehari, “Techniques and countermeasures for preventing insider threats”, eng, *PeerJ. Computer science*, vol. 8, e938–, 2022, ISSN: 2376-5992.

- [26] National Institute of Standards and Technology, “The NIST Cybersecurity Framework (CSF) 2.0”, National Institute of Standards and Technology, Gaithersburg, MD, NIST Cybersecurity White Paper NIST CSWP 29, Feb. 2024, Accessed: 2025-08-08. DOI: 10.6028/NIST.CSWP.29. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf>.
- [27] M. A. Ferrag, F. Alwahedi, A. Battah, B. Cherif, A. Mechri, and N. Tihanyi, “Generative ai and large language models for cyber security: All insights you need”, *Available at SSRN 4853709*, 2024.
- [28] F. N. Motlagh, M. Hajizadeh, M. Majd, P. Najafi, F. Cheng, and C. Meinel, “Large language models in cybersecurity: State-of-the-art”, *arXiv preprint arXiv:2402.00891*, 2024.
- [29] J. Siegel, “What is the definition of business process?”, Object Management Group, OMG-Certified Expert in BPM (OCEB) Program White Paper, 2008. [Online]. Available: https://www.omg.org/certification/bpm/documents/OCEB_Definition_Of_Business_Process.pdf.
- [30] G. Aagesen and J. Krogstie, “Bpmn 2.0 for modeling business processes”, *Handbook on Business Process Management 1: Introduction, Methods, and Information Systems*, pp. 219–250, Apr. 2015. DOI: 10.1007/978-3-642-45100-3_10.
- [31] Z. Bokolo and O. Daramola, “Chatbots security: Stride-based elicitation of security threats and vulnerabilities in insurance chatbots”, 2024.
- [32] J. Von der Assen, A. Huertas, J. Sharif, C. Feng, G. Bovet, and B. Stiller, “ThreatFinderAI: Automated Threat Modeling Applied to LLM System Integration”, in *2024 20th International Conference on Network and Service Management (CNSM)*, 2024, pp. 1–3.
- [33] L. Tuggenier, P. Sager, Y. Taoudi-Benchekroun, B. F. Grewe, and T. Stadelmann, “So you want your private llm at home? a survey and benchmark of methods for efficient gpts”, in *2024 11th IEEE Swiss Conference on Data Science (SDS)*, IEEE, 2024, pp. 205–212.
- [34] X. Xie, H. Liu, W. Hou, and H. Huang, “A brief survey of vector databases”, in *2023 9th International Conference on Big Data and Information Analytics (BigDIA)*, 2023, pp. 364–371. DOI: 10.1109/BigDIA60676.2023.10429609.
- [35] O. Topsakal and T. C. Akinici, “Creating large language model applications utilizing langchain: A primer on developing llm apps fast”, in *International Conference on Applied Engineering and Natural Sciences*, vol. 1, 2023, pp. 1050–1056.
- [36] L. Inc., *Sentence transformers on hugging face*, Accessed: 2025-06-16, 2024. [Online]. Available: https://python.langchain.com/docs/integrations/text_embedding/sentence_transformers/.
- [37] N. Reimers and I. Gurevych, *Sentence-bert: Pretrained models*, Accessed: 2025-06-16, 2020. [Online]. Available: https://www.sbert.net/docs/sentence_transformer/pretrained_models.html#all-mpnet-base-v2.
- [38] LanceDB, *Lancedb*, Accessed: 2025-06-16, 2024. [Online]. Available: <https://lancedb.github.io/lancedb/>.

- [39] L. Inc., *Lancedb*, Accessed: 2025-06-16, 2024. [Online]. Available: <https://python.langchain.com/docs/integrations/vectorstores/lancedb/>.
- [40] Lean Enterprise Institute, *Plan, do, check, act (pdca) - a resource guide*, Accessed: 2025-08-07, 2025. [Online]. Available: <https://www.lean.org/lexicon-terms/pdca/>.
- [41] A. M., *What is ollama? understanding how it works, main features and models*, Accessed: 2025-08-02, 2025. [Online]. Available: <https://www.hostinger.com/tutorials/what-is-ollama>.
- [42] E. Topics, *Gpt parameters: How many parameters are in gpt-3, gpt-4, and other ai models?*, Accessed: 2025-08-07, 2023. [Online]. Available: https://explodingtopics.com/blog/gpt-parameters?utm_source=chatgpt.com.
- [43] B. Mashouf, *Claude 4 vs llama 4: Benchmarking technical capabilities*, Accessed: 2025-08-07, 2025. [Online]. Available: <https://medium.com/@bob.mashouf/claude-4-vs-llama-4-benchmarking-55b99c17d3f7>.
- [44] Ollama, *Ollama search*, Accessed: 2025-08-07, 2025. [Online]. Available: <https://ollama.com/search>.
- [45] sourabhkv, *Make your own private chatgpt*, Microsoft Tech Community, Accessed: 2025-08-08, 2024. [Online]. Available: <https://techcommunity.microsoft.com/blog/educatordeveloperblog/make-your-own-private-chatgpt/4357607>.
- [46] Anthropic, *Privacy policy*, Accessed: 2025-08-08, 2025. [Online]. Available: <https://www.anthropic.com/legal/privacy>.

Abbreviations

AI	Artificial Intelligence
BPM	Business Process Modeling
BPMN	Business Process Modeling Notation
CIA	Confidentiality, Integrity, Availability
CoT	Chain of Thought
CSF	Cybersecurity Framework
CVEs	Common Vulnerabilities and Exposures
DFD	Data Flow Diagram
IoT	Internet of Things
LLM	Large Language Model
LoRA	Low-Rank Adaptation
ML	Machine Learning
NIST	National Institute of Standards and Technology
NLP	Natural Language Processing
NVD	National Vulnerability Database
OPRO	Optimization by PROMpting
PDCA	Plan-Do-Check-Act
POWL	Partially Ordered Workflow Language
RAG	Retrieval Augmented Generation
RAM	Random Access Memory
STRIDE	Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege
TMT	Threat Modeling Tool
UI	User Interface

List of Figures

2.1	Threat Modeling Approaches [22]	6
2.2	VISTA Framework [3]	7
2.3	Multi-Tiered Activity Monitoring Model [27]	8
2.4	Use Cases of LLMs in Cybersecurity [27]	9
2.5	LLMs in Cybersecurity [7]	10
4.1	Five-Step Methodology	21
4.2	RAG Architecture	22
5.1	Start Page	24
5.2	Step 1: Define Security Requirements	25
5.3	Step 2: Provide More Context about the Business Process	25
5.4	Step 3: Input Business Process Model	26
5.5	Step 5: Output Potential Threats and their Mitigation Strategies (1)	27
5.6	Step 5: Output Potential Threats and their Mitigation Strategies (2)	27
5.7	RAG Implementation	29
6.1	Business Process Used for Evaluation	40
6.2	Refined Output Example	54

List of Tables

3.1	Overview of the Related Work	14
5.1	Technology Stack Used in the Prototype	24
6.1	Comparison of LLMs for Insider Threat Detection	41
6.2	Assignment of Potential Insider Threats to Process Elements	46
6.3	Example Threat for Process Element: <i>Check Responsibility</i>	47
6.4	Comparison between Bachelor Thesis and Master Project Prototype	55
A.1	Threat Overview for Process Element: <i>Check Responsibility</i>	72

Listings

4.1	Ollama Testing Code	20
5.1	XML File Reduction Code	28
5.2	RAG Chunk Example	30
5.3	Vectorstore Initialization	31
5.4	Prompt Input Instructions	32
5.5	Prompt Output Instructions	33
5.6	Ollama Initialization	35
5.7	Claude Initialization	36
5.8	Element ID Filter Code	36
6.1	'Process Threat Analysis Steps', line 39	53
6.2	'Style Rules, Markdown Output Rules', line 53	53
6.3	'Style Rules, Output Format', line 69	53
6.4	'Style Rules, Example Output', line 78	53

Appendix A

Supplementary Content

A.1 Context Information

1. What is the main goal of this process? Describe the steps.

The goal of this process is to create an insurance card and create an individual account. Either an insured person or their employer notifies the responsible office of the Canton to create a new insurance card. Several information needs to be clarified and a check is done, whether the person already exists in the system. As soon as all clarifications are verified, a new insurance number can be ordered, which is then used to create a new insurance card.

2. What systems or technologies are involved?

ZAS is a central system that knows if a person already has a insurance card.

3. Who are the primary roles or actors?

There are three major actors: the member or insured person, the case worker for individual accounts that works for the canton and the central compensation office.

4. Other notes, risks, or special considerations?

The process includes confidential information such as the birth certificate, IDs, etc.

A.2 Threat Overview

Table A.1: Threat Overview for Process Element: *Check Responsibility*

Potential Threat	Unauthorized Data Access
Principle	Confidentiality
Potential Impact	A clerk with malicious intent could access sensitive personal information including birth certificates, IDs, and other confidential documents during the responsibility check process. This unauthorized access could lead to identity theft, privacy violations, data breaches affecting multiple individuals, and potential legal consequences for the organization. The compromised data could be sold or used for fraudulent insurance claims.
Mitigation Strategies	
	<ul style="list-style-type: none"> • Implement role-based access controls limiting data visibility to only necessary information for the task • Enable audit logging to track all data access and modifications by clerks • Apply data masking techniques to hide sensitive portions of documents unless specifically required • Require supervisor approval for accessing highly sensitive documents
Potential Threat	Data Exfiltration
Principle	Confidentiality
Potential Impact	During the responsibility verification process, an insider could copy or photograph sensitive personal documents and business case files. This could result in mass data theft of citizen information, exposure of personal identification details, and compromise of the entire insurance registration system's integrity. The stolen data could be used for identity fraud or sold to external parties.
Mitigation Strategies	
	<ul style="list-style-type: none"> • Disable copy/paste and screenshot capabilities on workstations handling sensitive data • Implement data loss prevention (DLP) tools to monitor and block unauthorized data transfers • Use watermarked documents that trace back to the accessing employee • Conduct regular security awareness training on data handling procedures

Potential Threat	Malicious Code Modification
Principle	Integrity
Potential Impact	A malicious insider could inject harmful code or logic bombs into the responsibility checking system, causing it to incorrectly validate or reject legitimate insurance applications. This could result in unauthorized individuals receiving insurance cards, legitimate applicants being denied coverage, or the corruption of the personal register database, leading to cascading errors throughout the insurance system.
Mitigation Strategies	
<ul style="list-style-type: none"> • Implement code review processes requiring multiple authorized personnel to approve any changes to the responsibility checking logic • Deploy integrity monitoring tools to detect unauthorized modifications to the system • Maintain version control with rollback capabilities for all system components • Restrict code modification privileges to essential personnel only with regular access reviews 	
Potential Threat	Resource Exhaustion Attack
Principle	Availability
Potential Impact	A malicious insider could deliberately slow down or block the responsibility checking process by creating excessive workload, submitting numerous false requests, or manipulating the system to generate repetitive checks. This would create a bottleneck at the very beginning of the insurance card creation process, preventing legitimate applications from being processed and causing significant delays in service delivery to insured persons.
Mitigation Strategies	
<ul style="list-style-type: none"> • Implement request throttling and rate limiting per user/employee • Set up automated monitoring for unusual patterns in responsibility check requests • Establish workload distribution mechanisms to prevent single-point bottlenecks • Create alerts for abnormal processing times or request volumes 	

A.3 Supplementary Files

The following files are part of the folder shared with the project supervisors:

- Source Code (frontend and backend)
- Powerpoint Presentations (final and midterm)
- Evaluation LLMs (Excel)
- Expert Evaluation (presentation, notes, consent forms, LLM output)

Appendix B

Other

B.1 Use of AI

AI-based tools were used to support parts of the programming and to assist in drafting and refining sections of this report. All AI-generated content was reviewed and adapted by the author, who retains full responsibility for the final work. The use of AI in the source code is indicated with comments in the corresponding files.

B.2 Installation Guidelines

To start the prototype on the local machine, please refer to the README.md in the frontend and backend project.