



CHALMERS
UNIVERSITY OF TECHNOLOGY



UNIVERSITY OF GOTHENBURG

Designing AI-Assisted Writing Interfaces for Requirements Engineering

Exploring Factors Influencing Usability and Design Guidelines

Master's thesis in Computer science and engineering

MATILDA DAVIDSSON
WILMA ÖLJEMARK

MASTER'S THESIS 2026

Designing AI-Assisted Writing Interfaces for Requirements Engineering

Exploring Factors Influencing Usability and Design Guidelines

MATILDA DAVIDSSON
WILMA ÖLJEMARK



UNIVERSITY OF
GOTHENBURG



CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2026

Designing AI-Assisted Writing Interfaces for Requirements Engineering
Exploring Factors Influencing Usability and Design Guidelines
MATILDA DAVIDSSON
WILMA ÖLJEMARK

© MATILDA DAVIDSSON, 2026.

© WILMA ÖLJEMARK, 2026.

Supervisor: Thommy Eriksson, Department of Computer Science and Engineering
Advisor: Dan-Erik Bergström, Geely Technology Europe
Examiner: Jasmina Maric, Department of Computer Science and Engineering

Master's Thesis 2026
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in L^AT_EX
Gothenburg, Sweden 2026

Designing AI-Assisted Writing Interfaces for Requirements Engineering
Exploring Factors Influencing Usability and Design Guidelines
MATILDA DAVIDSSON

WILMA ÖLJEMARK

Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

The integration of Large Language Models (LLMs) into professional workflows has created new opportunities for supporting information-intensive tasks such as Requirements Engineering (RE). However, despite the growing interest in AI-assisted writing tools, limited research has explored how such tools should be designed to support usability in industrial RE contexts. This thesis investigates how interaction design can support the integration of AI-assisted writing tools into requirements management systems to assist engineers in formulating clear and consistent requirements.

The study was conducted in collaboration with Geely Technology Europe and focused on a custom-built extension for the requirements management platform SystemWeaver. Using a Research through Design approach combined with the Double Diamond framework, the project included a literature review, semi-structured interviews, thematic analysis, ideation, prototyping, and usability testing.

The results identified four key factors important for usability when integrating AI-assisted writing tools in requirement management systems: control, trust, flexibility, and efficiency. Based on these findings, fourteen design guidelines across interaction, functionality, and design were developed.

The study highlights the importance of balancing trust and over-reliance, efficiency and control, as well as flexibility and consistency when integrating AI into professional requirements engineering environments. The findings contribute to the fields of Human-Computer Interaction, Human-AI Interaction, and Requirements Engineering by providing practical design guidance for future AI-assisted writing tools in industrial contexts.

Keywords: Requirements Engineering, Large Language Models, AI-assisted writing, Human-AI Interaction, Interaction Design, User Experience, Requirements management Systems, Human-Centered AI.

Acknowledgements

We would like to express our sincere gratitude to everyone who contributed to and supported us throughout this thesis process.

First, we would like to thank Geely Technology Europe for giving us the opportunity to conduct this thesis project and for welcoming us to the company during the project process. We truly appreciate the trust, support, and insight into an industrial context that made this project possible. We would especially like to thank our industry supervisor, Dan-Erik Bergström, for his continuous support, valuable feedback, and expert knowledge throughout the project. His deep understanding for the SystemWeaver platform helped us gain valuable insights into both the system's functionality and the practical requirements working within the platform. His guidance and encouragement throughout the entire process have been greatly appreciated.

We would also like to thank our academic supervisor, Thommy Eriksson, for his valuable guidance and patience throughout the thesis process. His academic expertise has been highly valuable in guiding the research process and helping us reflect on suitable approaches, processes, and delimitations throughout the project.

We would further like to thank all participants involved in the interviews and usability testing for contributing with valuable insights, concrete examples of challenges, and important user needs and preferences. Their participation and feedback were essential in shaping the direction of the project and making the final outcome possible.

Matilda Davidsson & Wilma Öljemark, Gothenburg, 2026-06-09

Contents

List of Figures	xi
List of Tables	xiii
1 Introduction	1
1.1 Aim and Goal	2
1.2 Research Questions	2
1.3 Deliverables	2
1.4 Stakeholders	2
1.4.1 Chalmers University of Technology	2
1.4.2 Geely Technology Europe	3
1.4.3 SystemWeaver AB	3
1.5 Scope and Limitations	4
2 Background	5
2.1 Project Context: An LLM-Framework Architecture for a SystemWeaver Integrated GUI	5
2.2 Artificial Intelligence	6
2.2.1 Natural Language Processing	6
2.2.2 Large Language Models	6
2.2.3 Generative AI and co-creative systems	7
2.3 AI Interactions and Interfaces	7
2.3.1 Co-creative interaction in generative AI systems	7
2.3.2 Common GUI patterns in generative AI systems	7
2.4 Requirements Engineering	9
2.4.1 Challenges in Requirements Engineering	10
2.4.2 Requirements Management Tools	10
2.4.3 SystemWeaver	11
2.5 AI in Requirements Engineering	11
2.6 Characteristics of Well-Formed Requirements	12
2.7 Related Work	13
3 Theory	14
3.1 Interaction Design	14
3.2 Human-Computer Interaction	15
3.2.1 Human-AI Interaction	15

3.2.2	Human-Centered AI	15
3.3	Design Theory	16
3.3.1	Human-Centered Design	16
3.3.2	User Experience	16
3.3.3	Usability	17
3.3.4	Accessibility	18
3.4	Design Guidelines	18
3.5	Ethical and Social Considerations	19
3.5.1	Framework for Responsible AI	19
3.5.2	Climate Concerns	20
3.5.3	Job Displacement	20
4	Methodology	21
4.1	Research Through Design	21
4.2	The Double Diamond Design Framework	22
4.3	Discover	23
4.3.1	Literature Review	23
4.3.2	Pilot Study	23
4.3.3	Interviews	23
4.4	Define	24
4.4.1	Thematic Analysis	24
4.5	Develop	24
4.5.1	Brainwriting	24
4.5.2	Crazy 8's	24
4.5.3	Wireframing	25
4.5.4	Prototyping	25
4.5.5	Design Critique	25
4.6	Deliver	26
4.6.1	Usability Testing	26
4.6.1.1	Cooperative Evaluation	26
4.6.1.2	Think-Aloud Protocol	26
4.7	Ethical Considerations of Data Collection	26
5	Execution and Process	28
5.1	Discover	28
5.1.1	Internal Data Review	28
5.1.2	Literature Review	28
5.1.3	Interviews	29
5.1.3.1	Interview Procedure	29
5.2	Define	30
5.2.1	Thematic Analysis - Interviews	30
5.2.2	Thematic Analysis Findings	34
5.2.2.1	Challenges in Current Workflow	34
5.2.2.2	Attitudes Towards and Use of AI	35
5.2.2.3	Functionality	36
5.2.2.4	Trust in AI	37
5.2.2.5	Control	37

5.2.3	Defining The Use Case	37
5.2.4	Design guidelines - First iteration	38
5.3	Develop	39
5.3.1	Ideation	39
5.3.1.1	Brainwriting	39
5.3.1.2	Crazy 8's	40
5.3.2	Wireframing	40
5.3.3	Design Critique	42
5.3.4	Prototype - First iteration	43
5.3.5	Functional Prototype	45
5.4	Deliver	47
5.4.1	End-user Testing	47
5.4.2	Thematic Analysis - User tests	48
5.4.3	User Test Findings	50
5.4.3.1	Unclear Terminology	50
5.4.3.2	Unclear UI Elements	51
5.4.3.3	System Feedback and Control	51
5.4.3.4	Trust and Verification	51
5.4.4	Prototype - Second Iteration	51
6	Results	54
6.1	Factors	54
6.2	Design Guidelines	55
6.3	Design Proposal	56
6.3.1	Requirement Analysis	57
6.3.2	AI Suggestion	59
6.3.3	Knowledge Source & Chat	61
6.3.4	History and Versioning	62
6.3.5	Review and Apply	63
7	Discussion	65
7.1	Reflecting on the Results	65
7.1.1	Balancing Trust and Over-reliance	65
7.1.2	Efficiency vs Human Control	65
7.1.3	Flexibility vs Consistency	66
7.1.4	Dependency on AI model Performance	66
7.2	Reflecting on Methodologies and Process	66
7.2.1	Thematic analysis	66
7.2.2	User testing	67
7.2.3	Prototyping	67
7.2.4	AI in the Design Process	68
7.3	Ethical Concerns	68
7.3.1	Job Displacement and Professional Pride	69
7.3.2	Accountability	69
7.4	Future Work	70
7.5	Rethinking AI as the Primary Solution	70
7.6	Use of AI	71

8 Conclusion	72
Bibliography	73
A Semi-structured Interview Guide	I
B Cooperative Evaluation Protocol	II

List of Figures

4.1	The Double Diamond framework illustration, adapted from Design Council [45].	22
5.1	Preliminary themes.	31
5.2	Preliminary themes.	32
5.3	Candidate themes.	33
5.4	Final Themes.	34
5.5	Example of Brainwriting (left) and Crazy 8s (right) ideation for G7: Offer linguistic support to ensure requirements are clearly and well-formulated.	40
5.6	Wireframes of the main interface and editing state.	41
5.7	Wireframes of the review dialog and chat history panel.	42
5.8	Overview of the requirement analysis components, including the categories Language, Testability, and Focus with status tags.	43
5.9	Components for editing and comparison, showing edit mode and the comparison view with highlighted changes.	44
5.10	Components for review and commit, showing the comparison dialog (left) and buttons for handling multiple suggestions (right).	44
5.11	History view showing chat overview with filtering options (left) and additional management actions such as pinning, renaming, and deleting (right).	45
5.12	Functional high-fidelity prototype demonstrating the AI tool's placement within the surrounding system context.	47
5.13	Physical setup of the evaluation sessions, showing the test environment and equipment arrangement.	48
5.14	Quotes organized under their respective codes.	49
5.15	Preliminary themes from first iteration of grouping codes.	50
5.16	Final themes.	50
5.17	Analysis refinement.	52
5.18	Multiple AI-suggestions refinement.	53
6.1	Overview of the side panel integrated into a SystemWeaver base UI prototype.	57
6.2	Requirement Analysis in active state with results (left) and disabled state (right). The analysis is disabled until a requirement description is initiated in the description field.	58

6.3	Detailed view of the Requirement Analysis, showcasing the expanded Clarity section (left) and the informational tooltip defining the analysis categories (right).	59
6.4	AI-generated requirement suggestions based on analysis (left) and multiple suggestions when a split is recommended (right).	60
6.5	Comparison (left) and editing (right) features for refining requirement suggestions.	61
6.6	Menu for selecting knowledge source (left), and the resulting AI response using the "Internet Search" capability (right).	62
6.7	Sidebar navigation for chat history, featuring search and sorting filters (left) and the management menu for specific chat sessions (right). . .	63
6.8	The "Review and apply" process: a side-by-side comparison dialog (left), and the confirmation snackbar displayed after successfully saving the changes (right).	64

List of Tables

5.1	Example of the coding process.	30
5.2	Overview of potential AI-supported use cases and scope selection . .	38

1

Introduction

The integration of artificial intelligence (AI) and large language Models (LLMs) has undergone remarkable development over the past decade, rapidly transforming work practices in both academia and industry through the automation of complex workflows [1]. By processing large amounts of textual data and identifying complex language patterns, LLMs are ideal for a wide range of Natural Language Processing (NLP) tasks, including text analysis, machine translation, and summarization. This technological development opens up new possibilities for managing information-intensive processes.

Requirements Engineering (RE) is a domain where information management is particularly critical. RE is a central activity within complex software and product development processes that aims to document, analyze, trace, prioritize, and control changes in requirements throughout the product life cycle, ensuring that stakeholder needs and organizational and technical goals are clearly understood and met [2], [3]. However, RE processes often face challenges such as fragmented information and interpretation variations between different actors [4]. Research shows that the quality of requirements is directly linked to project outcomes, as ambiguous or inconsistent requirements are a major cause of errors, delays, and cost overruns in system development [5]. When requirements lack clarity or consistency, engineers must invest additional effort to interpret, reconcile, and reason about large volumes of interconnected requirements. This increased cognitive demand raises the likelihood of misinterpretations and human error [6].

Here, LLMs emerge as a valuable tool to support the user and improve the writing quality of requirements. By leveraging their ability to analyze complex linguistic patterns, these models can help identify ambiguous or inconsistent requirement formulations and provide actionable feedback [7]. Integrating such AI support into existing workflows has the potential to minimize the risk of costly and time consuming errors caused by misunderstandings.

However, for LLMs to be truly effective as collaborative tools, the focus should extend beyond purely technical capabilities to include a user-centric perspective. Understanding user intent, along with prioritizing satisfaction, reliability, and cognitive ease, is essential for effective and successful human-AI collaboration in professional settings [8]. Therefore, exploring how interaction design can support the integration of AI-assisted writing tools into requirements engineering is of interest.

1.1 Aim and Goal

This study explores the intersection of AI-assisted writing and usability in the context of requirements engineering. The aim is to identify the practical needs, challenges, and conditions that are important to achieve strong usability when integrating AI-assisted writing tools in requirements management systems. Based on these insights, the goal is to propose a set of guidelines for designing AI-assisted writing interfaces within such systems.

1.2 Research Questions

The study is guided by the following research questions:

RQ1: *What factors are important to achieve strong usability when integrating AI-assisted writing tools into requirements management systems?*

RQ2: *What guidelines should be followed when designing interfaces for AI-assisted writing tools to support the creation of well-formed requirements?*

1.3 Deliverables

The expected outcomes are as follows:

- **Design Guidelines:** A set of guidelines derived from identified factors on how AI-assisted writing tools should be designed to support the creation of clear and consistent requirements.
- **Design Proposal:** A high-fidelity prototype demonstrating the application of these guidelines in an interface for AI-assisted requirements writing.

1.4 Stakeholders

The following section presents the key stakeholders of the thesis and their interests in the project.

1.4.1 Chalmers University of Technology

Chalmers University of Technology is the academic institution where this master's thesis is conducted and published. As a key stakeholder, the university has an interest in the project as it contributes to the academic fields of Human-Computer Interaction (HCI) and Human-AI Interaction. The study is particularly central to the Department of Computer Science and Engineering (CSE), where Interaction Design and Software Engineering are core research areas.

Chalmers benefits from studies that deepen the understanding of User Experience (UX) and usability, specifically within the context of complex digital environments and industrial workflows. By exploring how AI-assisted writing can be integrated

into professional settings, this research provides a foundation for future academic work. Furthermore, this thesis supports Chalmers' ambition to bridge research and real-world challenges, making the outcomes useful for both academic and industrial contexts.

1.4.2 Geely Technology Europe

Geely Technology Europe (GTEU) is a unified research and development (R&D) organization within the Geely Auto Group, formally established in March 2026 by integrating premier engineering centers in Sweden and Germany [9]. The organization is dedicated to advancing next-generation vehicle development for global markets, building on more than a decade of European expertise. This foundation includes the development of industry-leading vehicle platforms that currently support over four million vehicles of different brands worldwide. The organization has evolved from the 2013 establishment of China Euro Vehicle Technology (CEVT) and, more recently, Zeekr Technology Europe. This study was conducted in collaboration with the IT Services and Governance department and focuses on a Geely-specific, custom-built extension of the external requirements management platform SystemWeaver. The study is aimed at engineers managing complex industrial requirements within the automotive development process.

As the organization expands its mandate to serve multiple brands, consistent and efficient requirement formulation is critical. Therefore, this study is of interest to GTEU to gain an understanding of how AI-assisted writing interfaces can enhance user experience and streamline the creation of high-quality requirements in a large-scale industrial setting.

1.4.3 SystemWeaver AB

SystemWeaver AB is the Swedish company behind the SystemWeaver software, a collaborative product information management platform extensively used within the automotive industry for systems engineering and requirements management [10]. In the context of this project, SystemWeaver AB acts as the technical implementer responsible for the development and delivery of the custom-built extension for GTEU. As the provider of the infrastructure, SystemWeaver AB is a key stakeholder with a strong interest in how new features and interfaces are integrated into their environment.

While GTEU provides the functional requirements and business needs, this thesis contributes by identifying user needs and designing the interface for the AI-assisted extension. Consequently, this study is of high value to SystemWeaver AB, as it provides a researched and user-tested design foundation that informs the actual implementation of the tool. The findings help SystemWeaver AB understand the practical requirements for incorporating an LLM model into the SystemWeaver ecosystem, ensuring that the final product is both technically robust and optimized for the end-users.

1.5 Scope and Limitations

This thesis does not aim to develop or implement any Large Language Model (LLM) architectures or training algorithms. The focus is on the application and presentation of existing LLM capabilities. Furthermore, the scope of this thesis is limited to the requirements writing activity within Requirements Engineering, i.e., the documentation and specification, as well as the maintenance of requirements. Activities such as requirement validation and verification, as well as requirements traceability and change management across the development life cycle, are outside the scope of this study.

2

Background

This chapter presents the theoretical and technical background of the study. It begins by introducing the broader project context in which the thesis is situated and outlines the AI-supported requirements engineering initiative that motivates the work. The chapter then presents core concepts within Artificial Intelligence and Requirements Engineering. Furthermore, it examines interaction patterns and interface design in generative AI systems, as well as characteristics of well-formed requirements. The chapter concludes with related work on human-AI collaboration and AI-assisted writing interfaces.

2.1 Project Context: An LLM-Framework Architecture for a SystemWeaver Integrated GUI

This thesis is part of a larger ongoing internal project at GTEU to develop an LLM-framework architecture for a SystemWeaver-integrated graphical user interface. The project aims to bridge the gap between structured requirements engineering in the database environment of SystemWeaver and the cognitive capabilities of LLMs.

The core objective of this larger initiative is to establish a seamless AI-driven validation ecosystem that allows users to verify requirements against a controlled set of document standards, corporate benchmarks, or global industry trends. A primary focus is on maintaining data integrity while providing high-quality, context-aware feedback.

To achieve this objective, while accounting for the specific limitations of an on-premises AI-stack, the project is built upon several critical technical pillars. Among these, the most relevant and applicable pillars within the context of interaction and interface design, which constitutes the primary focus of this thesis, is presented below:

- *Integrated Feedback Pane:* The framework should introduce a dedicated, non-intrusive side-pane within the SystemWeaver GUI.
- *Direct Workflow Integration:* The user should be able to edit, copy, or insert LLM-generated suggestions directly into the database.
- *Multi-Tiered Knowledge Scoping:* The framework should offer granular control over the "Source of Truth", allowing users to toggle between corporate verified

repositories, targeted project subsets, and global web intelligence, while also supporting temporary file uploads for session-specific tasks.

- *Contextual History*: The framework should maintain a persistent session memory, enabling iterative refinements through continuous dialogue with the AI, allowing users to progressively refine their requirements toward an optimal state.

While the broader project encompasses a larger AI-supported ecosystem for requirements engineering, this thesis focuses specifically on the user-facing dimension of that initiative. More specifically, as mentioned in section 1.5, the study is limited to the interaction design and evaluation of an system-integrated interface intended to support requirements writing activities. Technical aspects of the underlying AI architecture, as well as other requirements engineering activities beyond requirements writing, are outside the scope of this thesis.

2.2 Artificial Intelligence

Artificial Intelligence (AI) refers to computer systems capable of performing tasks that typically require human intelligence, such as reasoning, learning, perception, and decision-making. Early AI systems were based on symbolic approaches and were able to perform tasks such as theorem proving and game playing. However, progress was limited by computational power and data availability. Since the 2010s, advances in deep learning, combined with large datasets and increased computational resources, have led to significant breakthroughs in AI capabilities, leading to the intelligent systems we see today [11].

2.2.1 Natural Language Processing

Natural Language Processing (NLP) is a field of AI that enables computers to understand and generate human language. It comprises two main components: Natural Language Understanding (NLU), which focuses on extracting meaning from text, and Natural Language Generation (NLG), which focuses on producing text. NLP is applied in a wide range of tasks, including machine translation, information extraction, and summarization [12].

2.2.2 Large Language Models

Large Language Models (LLMs) are a class of NLP models trained on large-scale text datasets to perform a wide range of language-related tasks. Compared to earlier NLP systems, LLMs are capable of generating more coherent and context-aware outputs. They are widely used in applications such as chatbots, virtual assistants, and content generation tools [13].

2.2.3 Generative AI and co-creative systems

Generative AI refers to computational techniques capable of producing new and meaningful content, such as text, images, or audio, based on patterns learned from training data [14]. These contents are increasingly difficult to distinguish from human-made work. Beyond generating content autonomously, generative AI can also function as a creative collaborator, working alongside human in a shared creative process [15]. Such human-AI co-creative systems raise important questions about interaction design, as the quality of collaboration depends not only on the AI's capabilities but also on how effectively humans and AI can communicate with each other.

2.3 AI Interactions and Interfaces

Generative AI systems are shifting interaction design from command-based interfaces toward more collaborative and conversational forms of engagement. In these systems, the interface supports an ongoing dialogue where input and output continuously shape each other. This chapter describes how these interaction patterns are reflected in modern AI interfaces.

2.3.1 Co-creative interaction in generative AI systems

Generative AI systems have shifted interaction design from traditional command-based interfaces toward more collaborative and conversational forms of interaction [15]. Unlike conventional software, where users typically execute predefined actions through menus and buttons, co-creative AI systems rely on continuous communication and iterative exchange between the user and the AI system. In these systems, creativity emerges through the interaction itself rather than solely from either the human or the AI independently [15].

Within co-creative systems, the user interface functions as the communication layer between the human and the AI collaborator. Modern AI interfaces increasingly support multi-modal forms of communication such as text prompts, sketches, gestures, voice interaction, and embodied interaction. This supports interaction patterns where users can iteratively guide and negotiate with the AI through continuous input, rather than issuing isolated commands [15].

2.3.2 Common GUI patterns in generative AI systems

As generative AI systems have become increasingly integrated into creative workflows, recurring GUI patterns have emerged across modern AI platforms. Systems such as ChatGPT, Claude, and Gemini build on conversational interaction, multi-modal communication, and iterative refinement, where users continuously shape outputs through ongoing dialog. This interaction is typically structured around prompt-based input, editable outputs, and regeneration functions that support exploration and revision. These tools share recurring GUI patterns that can be structured in three interrelated layers: input controls, output controls, and feedback loops. The

following analysis of GUI patterns is based on official product documentation and observed interface behavior of ChatGPT, Claude, and Gemini [16], [17], [18], [19], [20].

The primary interaction pattern in all three above mentioned systems is the chat-based prompt interface. This input field acts as a flexible entry point for natural language instructions, allowing users to shift between conversational questions, task-oriented commands, and creative exploration. Beyond basic chat input, several additional input components are commonly used:

- **File upload interfaces:** Users can upload documents, images, or databases to provide contextual grounding for the AIs responses.
- **Multi-modal input options:** Gemini and ChatGPT support multi-modal input, enabling users to provide image-based prompts or combine images with textual instructions within the same prompt.
- **Prompt suggestions/starter prompts:** All three systems offer quick-start suggestions to help users initiate conversations.
- **Context injection mechanisms:** Some tools allow users to set system-level instructions, e.g., "custom instructions" in ChatGPT, shaping tone, output format, or behavior across interactions.

The output layer in generative AI interfaces is no longer static. Instead, responses are treated as editable and negotiable artifacts. Common output components include:

- **Regenerate/retry functions:** Allowing users to request alternative versions of the same input.
- **Editable responses:** ChatGPT and Gemini present canvas-like features or document modes, allowing users to directly modify generated content.
- **Structured output formats:** AI responses often include headlines, bullet points, or code blocks to improve readability and usability.
- **Follow-up suggestion chips:** After an answer, the system may propose related questions or actions, encouraging continued interaction.

Furthermore, a defining feature of these systems is the continuous feedback loop between input and output. Rather than producing a single final result, the interaction evolves through refinement cycles. Key mechanisms enabling this include:

- **Conversational memory within sessions:** The AI maintains context across turns, enabling progressive refinement of ideas.
- **Inline editing and revision cycles:** Users can ask for targeted improvements, e.g. "make this more formal", "shorten this section", or "change tone").
- **Version generation and comparison:** Some system allow multiple variants of an output to be generated and compared.

Together, these three layers illustrate how generative AI interfaces move beyond traditional input-output models towards an iterative co-creative workflow. This

structure positions the user not only as an operator of the system, but as an active participant in shaping and iterating the generated content.

2.4 Requirements Engineering

Requirements are defined as the essential attributes, conditions, or capabilities that a system or product must meet or possess to satisfy, e.g., a contract, a standard, or a specific customer need. Within the development of complex systems, Requirements Engineering (RE) is recognized as the most critical process because it has a dominant impact on the final product’s capabilities [21]. It serves as a systematic approach to collecting demands from a diverse set of stakeholders and implementing them throughout the entire system life cycle. In the context of large-scale and complex system development, such as in the automotive industry, it is practically impossible for requirements to remain stable over years of development. Consequently, RE is, in these contexts, an iterative process that must be performed in parallel with other development activities.

The RE process is broadly divided into two main groups of activities: Requirements Development and Requirements Management. Development focuses on the discovery and definition of the system, while management ensures the integrity of these requirements as the project evolves. These two groups consist of several phases that move the project from initial concept to a valid system.

Requirements development includes the following three phases: (1) *Elicitation*, which focuses on identifying system stakeholders to gather their diverse objectives and viewpoints, such as those of customers, users, and regulatory standards. The goal of elicitation is to capture a wide range of perspectives, including business, security, and information requirements, as well as operational constraints; (2) *Documentation*, in which a formal specification document is produced of the system’s external behavior, intended purpose, and environmental constraints. Each requirement is assigned a unique identifier to ensure precise tracking and management across various sub-systems. The purpose of the specification is to serve as a comprehensive blueprint that minimizes development effort and costs; (3) *Validation and verification*, which includes performing reviews and prototyping to confirm that the documented requirements meet both actual user needs (validation) and technical specifications (verification) [21].

While the requirements development activities define the system, requirements management acts as the continuous control mechanism that tracks changes and dependencies. A core component is *traceability*, which allows engineers to follow a requirement’s life in both forward and backward directions across all development artifacts. The management phase is often seen as the most complex part of the engineering process, since it relies on a rigorous *change management* process to assess the broader costs and impacts of modifications before they are reintroduced into the development cycle. Since even a single change can be difficult to estimate and can have a large impact on the developing system, a structured process is essential to maintain stability and ensure product quality. [21]

2.4.1 Challenges in Requirements Engineering

As previously mentioned, requirements engineering is often described as one of the most critical phases in software and product development processes, but also one of the most difficult to master and one of the most problematic. According to the comprehensive global study *Naming the Pain in Requirements Engineering* by Fernández et al. [22], the discipline is affected by a range of recurring challenges that have a direct negative impact on project success.

The single most common issue in practice is the presence of incomplete or hidden requirements, which often results in critical functionality being discovered late in the development process [22]. This lack of completeness is frequently linked to requirements being underspecified or too abstract, creating room for subjective interpretations and misunderstandings. In addition to these content-related issues, communication flow represents a central barrier. Breakdowns in communication between the project team and the client are identified as the challenge most often leading to complete project failure, even though it is not always the most frequently occurring issue. Beyond the precision of requirements, organizations also commonly struggle with so-called "moving targets", where both specific requirements and overarching business goals change throughout the project life cycle. This places high demands on flexibility and change management. [22]

These challenges can rarely be traced to isolated mistakes. Rather, they emerge from a complex interplay of human and organizational factors. The study shows that the primary root causes are often a lack of experience or insufficient qualifications within the RE team, combined with stakeholders who lack a clear business vision or struggle to articulate their actual needs. The situation is further exacerbated by general time constraints, where the RE process is often forced into overly tight schedules, reducing the opportunity for thorough analysis. [22]

How these challenges manifest also depends on the development methodology. Agile projects tend to struggle more with customer availability and time constraints, while plan-driven projects experience greater difficulties with "moving targets". If these challenges are left unmanaged, they trigger a chain reaction of negative consequences, ranging from time and budget overruns due to extensive rework, to a lower-quality final product that fails to meet standards of usability and user acceptance. [22]

2.4.2 Requirements Management Tools

To facilitate the critical maintenance and navigate the large amount of data in the requirements management process, specialized Requirements Management (RM) tools are used. These tools are designed to handle the entire life cycle and serve as a digital archive or database for end-to-end management and administration of requirements, including their creation, modification, and associated relationships [23]. In addition, these tools provide the necessary infrastructure for multidisciplinary teams to collaborate securely while managing the complexities of a system's life cycle [21].

Beyond simple storage, RM tools automate traceability by enabling engineers to follow a requirement's evolution from its origin to its final deployment through a

comprehensive audit trail [21]. The tools further support the change management process by providing formal mechanisms to authenticate and approve change requests, ensuring that only validated modifications are implemented into the system.

2.4.3 SystemWeaver

SystemWeaver is the specific life cycle management platform utilized by GTEU to manage their requirements processes. SystemWeaver is a collaborative platform for Model-Based Systems Engineering (MBSE) that provides a unified environment for product overviews, end-to-end traceability, and the administration of system functionality throughout the development process [24]. The SystemWeaver platform supports company-specific extensions, which allow its core functionality to be adapted to organizational needs. This flexibility makes it possible to integrate additional features directly into the requirements workflow. Within this environment, AI-based tools have been introduced to support tasks such as translation and interaction with individual requirements through a chat-based interface. These tools generate responses based on selected requirements and related data stored within the company's SystemWeaver database.

2.5 AI in Requirements Engineering

Large language models (LLMs) have shown potential to support and automate labor-intensive requirements engineering tasks, for example, by generating, analyzing, and validating requirements in a more efficient and accurate manner. However, the use of AI in requirements management also introduces several risks, including limited explainability and lack of transparency in AI-generated outputs. Many AI models, particularly those based on deep learning, operate with limited transparency, which makes it difficult for users to understand why requirements are proposed or formulated in a certain way. [25]

Another critical challenge is bias in AI models. If training data contains biases, this can result in certain requirements being prioritized incorrectly or in the reinforcement of existing inequalities. This, in turn, can negatively affect fairness, sustainability, and the extent to which requirements reflect the needs of different stakeholders. In addition, there are ethical concerns regarding liability, transparency, and oversight, particularly in terms of responsibility for incorrect or problematic AI-generated requirements and ensuring compliance with regulations such as GDPR. Therefore, well-defined guidelines and accountability mechanisms are required to establish and maintain trust. [25]

Abbasi et al. [25] developed a framework to promote responsible and effective collaboration between humans and AI in the requirements engineering process. The framework is based on four principles: (1) human-in-loop validation, where final decisions are made by experts to avoid automation bias; (2) explainability and transparency, so that AI-generated requirements can be understood and reviewed by users; (3) bias reduction strategies, which continuously monitor and address bias

in AI output; and (4) trust calibration, which involves adapting models to the needs and priorities of different stakeholders to ensure trust.

2.6 Characteristics of Well-Formed Requirements

In this study, as well as in the larger project this thesis is part of, the guidelines developed by the International Council on Systems Engineering (INCOSE) serve as a primary reference for how requirements should be formulated. INCOSE is a global professional organization dedicated to advancing systems engineering best practices and has developed the *Guide to Writing Requirements* (GtWR) [26]. The fundamental purpose of the INCOSE GtWR is to provide a structured framework for ensuring that requirements are clear, testable, and fit for purpose throughout the entire system life cycle. Rather than relying on subjective notions of quality, the guide outlines specific characteristics and rules that needs and requirements must satisfy to be considered well-formed.

The guide defines a set of 15 quality characteristics that must be satisfied. An individual requirement statement must be *necessary* (C1), meaning it defines a capability or constraint required to satisfy a higher-level source, as well as *appropriate* (C2) to its architectural level. It must also be *unambiguous* (C3), ensuring its intent can be interpreted in only one way by all audiences, and *complete* (C4), sufficiently describing the necessary conditions or quality factors. Furthermore, individual requirements must be *singular* (C5), focusing strictly on a single capability, *feasible* (C6) within project constraints, and *verifiable* (C7). Lastly, they must be *correct* (C8), accurately representing the need, source, or higher-level requirement, and they must be *confirming* (C9) to approved patterns and style standards [26].

At a collective level, a set of requirements must stand alone as *complete* (C1) by sufficiently describing all functionalities and constraints without requiring other external sets. The set must also be *consistent* (C11), meaning its individual components are unique, do not conflict or overlap, and utilize homogeneous units of measurement. Finally, a requirements set must be *feasible* (C12) as a whole, *comprehensible* (C13) in its macro-system relation, *able to be validated* (C14) against overarching product goals, and *correct* (C15) in its representation of higher-level needs.

To support the practical application of these abstract quality characteristics, the INCOSE GtWR provides 42 concrete rules that promote grammatical and structural consistency during the drafting process [26]. To ensure that requirements are clearly formulated, each requirement must follow an agreed structure (R1), use active voice with a clearly identified responsible entity as the subject (R2), and express quantities using common units of measurement (R6). Requirements must also avoid ambiguity by eliminating unnecessary phrases such as "to be able to" (R10), vague terms like "some" or "many" (R7), and expressions that weaken or leave requirements open-ended, such as "where possible" (R8) or "etc." (R9). To maintain conciseness and focus, each requirement should express a single idea in one sentence (R18) and avoid using words such as "and", "or", or "as well as" to combine multiple clauses (R19). Finally, the rules require specific and measurable performance targets (R34) and

prohibit vague time-related terms such as "eventually" or "before", requiring explicit and measurable timeframes instead (R35).

2.7 Related Work

Previous research on generative AI has emphasized the role of interface design in shaping how users interact with and make sense of AI-supported writing tools. Rather than focusing solely on model capabilities, these studies examine how different interface structures influence user control, revision practices, and perceptions of agency in human-AI collaboration.

Lou et al. [27] developed AnchoredAI, a system that anchors AI-generated comments directly to specific text segments. In a comparison between AnchoredAI and a chat-based LLM interface, they found that AnchoredAI led to a stronger sense of ownership among writers. The authors highlight how interface design shapes AI-assisted writing and argue that anchoring can reduce overreliance and enable more precise, user-driven revision methods.

In another study, Reza et al. [28] presents ABScribe, an interface that enables the organization and comparison of multiple alternative text variations in human-AI co-writing tasks. A user study with 12 writers indicates that ABScribe reduced perceived task workload and improved users experience of the revision process compared to a standard writing workflow.

Another study by Lehmann and Buschek [29] explored how users envision delegating writing tasks to AI and developed an interface grounded in these insights. An initial user study revealed that participants tended to revert to short, command-like prompts when interacting with the system. In a subsequent iteration, these commands were offered as explicit toolbar shortcuts alongside open-ended prompting, which led users to fluidly switch between specific and more flexible AI functions. The authors emphasize the importance of moving beyond purely conversational interfaces, arguing that interface design plays a critical role in shaping users access to and understanding of the functional space of generative AI models.

While prior work has explored AI-assisted writing interfaces in general contexts, less attention has been given to industrial settings requiring high precision and consistency. This study addresses this gap by focusing on AI support for requirement writing in a professional Requirements Engineering environment.

3

Theory

This chapter provides the theoretical foundation for the research, grounded in interaction design and theories of human-computer interaction, human-AI interaction, and design theory. It also presents the concept of design guidelines and ethical frameworks related to the study.

3.1 Interaction Design

To academically ground the development and evaluation of AI-assisted writing interface, this project is situated within the field of Interaction Design (IXD). According to Sharp et al. [30], interaction design is defined as: *"designing interactive products to support the way people communicate and interact in their everyday and working lives"*. Interaction design prioritizes the human experience, usability, and how users perceive and operate the system image, rather than the internal structure of realizing a technical system, as in software engineering. In demanding professional environments, such as requirements engineering (RE), an advanced tool may be effective from a technical perspective, but its usability is limited by how well the interface is designed with the users' needs, abilities, and cognitive limitations in mind.

A fundamental principle within interaction design is the user-centered approach, where the needs, goals, and capabilities of users guide the development process rather than technological possibilities alone. Originally established by Gould and Lewis in 1985, a user-centered design process is based on three fundamental principles: an early focus on users and their tasks, empirical measurement, and iterative design [30]. An early focus on users involves understanding their characteristics, work practices, and needs through direct observation and participation in the design process. Empirical measurement refers to continuously evaluating designs with intended users through scenarios, prototypes, and usability testing to collect data on user performance and experiences. Iterative design emphasizes that identified usability problems should be addressed through repeated cycles of design, testing, evaluation, and redesign. Together, these principles ensure that interactive systems are developed in close alignment with users real-world activities and support the creation of solutions that are both useful and easy to use.

To apply these principles within a structured development lifecycle, Sharp et al. [30] describe interaction design as a process consisting of four interconnected activities: (1) identifying needs and establishing requirements for the user experience; (2) de-

veloping alternative designs that meet those requirements; (3) building interactive versions of those designs; and (4) evaluating what is being built throughout the process and the user experience it offers. These activities are intended to continuously inform one another and are repeated throughout development, enabling the refinement of a product based on direct user feedback.

3.2 Human-Computer Interaction

Human-Computer Interaction (HCI) is an interdisciplinary field that studies the interaction between humans and computer systems [31]. The purpose is to understand how interactive technologies can be best designed and implemented to be both effective and user-friendly. According to Sinha et al. [31], HCI acts as a bridge between computer science, cognitive psychology, and design.

HCI is defined by the relationship between three central elements: the human, the computer, and the process through which they communicate - the interaction. The focus is on understanding the user's abilities and limitations in order to create systems that minimize cognitive load and maximize productivity. [31]

3.2.1 Human-AI Interaction

The interaction between humans and artificial intelligence differs from traditional human-computer interaction primarily due to the probabilistic nature of AI systems [32]. Unlike conventional software, which exhibits predictable and linear behavior, AI systems can act inconsistently and produce unexpected results or errors. This creates uncertainty among users, as they often have difficulty understanding why an AI system acts as it does, which can lead to either overtrust (blindly trusting incorrect suggestions) or undertrust (dismissing a functioning system).

This uncertainty places higher demands on interface design, as designers must help users interpret and appropriately rely on AI systems. To support designers in meeting these demands, specific guidelines have been developed. Amershi et al. [32] have established one such set, consisting of 18 guidelines aiming to bridge the technical capacity of the AI and the user's expectations. These guidelines treat the interaction as a cycle of four phases: (1) the initial setting of expectations, (2) how the system acts during use, (3) how it handles misjudges, and (4) how it learns over time. The main goal is to make the system transparent and predictable despite its complexity. This means that the interface must actively communicate system uncertainty and offer contextual explanations, so that the user can make informed decisions based on the AI's output.

3.2.2 Human-Centered AI

Within the research field of Human-centered Artificial Intelligence (HCAI), the focus is on designing systems that augment human abilities rather than replacing them through complete autonomy. According to Shneiderman [33], the aim of AI development should be to create systems that are reliable, secure, and trustworthy,

which is achieved by placing the human user at the center of the design process. The HCAI framework advocates a combination of human control and machine automation, meaning that the system should be able to perform advance operations while the user maintains a high degree of overview and control over the process. Striking this balance requires interfaces that counteract the perception of AI as a non-transparent "black box" [33]. Central to such a design are transparency and explainability, which means that the system can explain in a comprehensible way the factors that underlie its results. By using visualizations and clear explanatory models, complex information can be made manageable, which strengthens the user's trust in the system's suggestions.

Furthermore, HCAI emphasizes the importance of user agency, where the design actively supports human decision-making by offering opportunities for review and correction, creating a human-centered system that functions as a powerful instrument rather than an independent actor [33]. By combining high automation with intuitive control mechanisms, interfaces can be created that streamline complex tasks without the user losing ultimate responsibility or critical perspective.

3.3 Design Theory

This section outlines the design theories and principles used to bridge the gap between technical functionality and human interaction. It establishes a framework for creating intuitive systems by focusing on usability, user experience, and human-centered methodologies.

3.3.1 Human-Centered Design

Human-Centered Design (HCD) is a design framework that prioritizes human experience over technical capability [34]. Rather than letting technology dictate the solution, HCD begins with a deep understanding of human needs, values, and constraints to ensure that every system we build serves a real purpose. According to Auernhammer [34], HCD is about seeing design as a social process where the goal is to improve human well-being and create meaningful interactions between humans and technology.

3.3.2 User Experience

A central concept within Interaction Design is user experience (UX), which refers to how interactive products are perceived, experienced, and emotionally evaluated by users in real-world contexts [30]. Rather than focusing solely on functional performance, user experience encompasses the overall quality of the interaction, including how satisfying, engaging, and meaningful a product feels to use. A user experience cannot be explicitly designed; rather, designers can only design *for* a user experience by creating specific interface features that can evoke intended emotional responses.

User experience is not a static state but a process that changes over time. Hassenzahl [35] emphasizes the importance of understanding interaction in three stages:

anticipation before use, real-time interaction, and post-interaction reflection. It is often the memory of the interaction rather than the interaction itself that defines the user's attitude towards the product in the long term.

3.3.3 Usability

Usability and user experience are closely related concepts. While user experience addresses the subjective and emotional qualities of an interaction, usability focuses on optimizing human-computer interaction to ensure that products are easy to learn, effective to use, and support productivity [30]. Usability contributes to the quality of the user experience, while experiential qualities can influence how usable a system is perceived to be. In interaction design, both perspectives are considered during the design process in order to support the development of interactive systems that are functional and experience-oriented.

Usability is defined by the ISO-standard as:

The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. [36]

According to this definition, usability is not an isolated characteristic of a product, but rather a result of the interaction between the product, the user, the specific task, and the environment where the interaction occurs [37]. To quantify and measure usability, the concept is divided into three measurable factors: effectiveness, efficiency, and satisfaction. Effectiveness measures to which the user actually succeeds in achieving their goal. Efficiency refers to the amount of effort or resources required to achieve the goal. And satisfaction is a subjective measurement of how comfortable and acceptable the product is perceived by the user.

When designing for usability, Jordan [37] advocates ten design principles. These serve as guidelines to ensure that the product is consistent with the user's cognitive and physical capabilities. The most central and relevant principles for this study include:

- **Consistency:** Similar tasks should be performed in similar ways. Through consistent design, the user can generalize their knowledge from one part to another, reducing the risk of errors and confusion.
- **Compatibility:** The design should be consistent with the user's reality, language, and expectations. To create an easily understandable system, the interface should use terms, symbols, and workflows that feel natural and familiar to the user.
- **Feedback:** The product should provide clear confirmation that an action has been performed and inform about the result of the action. Lack of feedback often results in uncertainty and duplication of work.
- **Visual Clarity:** Information must be presented in a way that can be read quickly and easily without confusion. This includes everything from text size to visual noise or cluttered interfaces.

- **Error Prevention and Recovery:** The design should minimize the risk of user errors and offer easy ways to recover the system if errors do occur.
- **User Control:** The user should be in control of the pace and sequence of the interaction. The system should avoid forcing actions that take control away from the user.

In this study, the term *strong usability* is utilized within the first research question to emphasize a holistic and context-driven approach to evaluation. Rather than viewing usability as a superficial quality focused on interface aesthetics or isolated task execution speeds, this thesis uses the term to describe a state where the system integrates naturally into the users' actual workflows, practices, and environment. In this sense, achieving strong usability means that the developed interface acts as a tool that addresses the real-world operational challenges and supports the user needs identified in the fieldwork, without introducing an additional layer of complexity or disruption to established professional practices.

3.3.4 Accessibility

Accessibility is closely related to usability and can be approached through established usability principles. Moreno et al. [38] demonstrate how traditional usability heuristics can serve as a bridge to web accessibility, as several principles, such as consistency, feedback, and error prevention, also address accessibility concerns when applied to diverse user needs. This highlights how improving general usability can contribute to making systems more inclusive.

3.4 Design Guidelines

Design guidelines are used to structure and communicate design knowledge in the form of recommendations that guide design decisions in specific contexts. Unlike more general design principles, guidelines are often context-dependent and more practically oriented. They are typically based on prior experience, empirical studies, or expert knowledge, and aim to increase the likelihood of a successful design solution. According to Fu et al. [39], guidelines are defined as "a context-dependent directive, based on extensive experience and/or empirical evidence, which provides design process direction to increase the chance of reaching a successful solution".

Design guidelines also serve as a way to store and reuse design knowledge, especially in early design phases where broad guidance is valuable. They may include recommendations for how to address design problems, as well as rationales and examples that help designers make informed decisions. At the same time, guidelines are flexible and evolving, meaning they can be revised over time as technology, user behavior, and design practices change [39].

3.5 Ethical and Social Considerations

When developing systems that involve Artificial Intelligence (AI), it is important to consider and analyze ethical implications and potential societal impacts to be aware of risks and consequences.

3.5.1 Framework for Responsible AI

To ensure that AI-systems are developed responsibly, Leslie [40] presents a framework called FAST. These principles act as an ethical compass throughout the design and implementation process.

- **Fairness:** Fairness in AI is about preventing the system from producing results that are biased or discriminatory, a phenomenon often referred to as algorithmic bias [40]. AI systems, therefore, need to be designed in a way that does not reproduce or reinforce existing prejudices in society. This includes both distributional justice, that is, how resources, opportunities, and consequences are distributed between different groups, and actively identifying and preventing bias in the data used to train the algorithms.
- **Accountability:** There must be clear accountability structures for the AI-system's decisions and actions throughout the whole life cycle [40]. This means that it must be possible to answer who is legally and morally responsible if the systems causes harm or makes incorrect decisions.
- **Sustainability:** Sustainability in this context refers to the technical robustness of the system combined with its long-term social impact [40]. The system needs to be designed to function reliably under pressure and be secure against manipulation, while evaluating its impact on human well-being. The goal is a sustainable implementation that promotes social cohesion and ensures that technology supports rather than undermines human agency over time.
- **Transparency:** Transparency is a prerequisite for an AI to be auditable and trusted, which means that it must be possible to understand how the system achieves its results [40]. This includes requirements for explainability, where the technical logic behind a specific decision is made understandable to people affected by it.

Taken together, responsible AI is about transforming ethical principles into concrete design choices throughout the entire life cycle, from the initial data collection to the final deployment. By combining technical robustness with understanding for the social contexts in which the system will operate, technology can act as a supporting tool that enhances rather than replaces human judgment. A key aspect of this is the integration of a "Human-in-the-loop" approach to ensure continuous human oversight, so that people can stay involved and step in when key decisions are made. This holistic approach ensures that ethical considerations do not become a separate checklist, but an integral part of the system's actual utility and social acceptance. [40]

3.5.2 Climate Concerns

As artificial intelligence has grown more advanced, the environmental impact of developing these systems has increased. Training large language models requires extensive computing resources, resulting in high energy use, carbon emissions, and water consumption across power plants, data centers, and hardware production [41]. In addition, inference, the process of generating outputs whenever a user interacts with the model, has become a major environmental factor due to its continuous and large scale nature [42].

3.5.3 Job Displacement

A central ethical debate in the implementation of AI concerns the tension between supporting human and replacing them [43]. As AI develops abilities resembling human intelligence, concerns about job loss and reduced human dignity are increasing. Researchers indicate that a significant portion of today's jobs are at risk of being automated, which highlights a social challenge, as work is often closely linked to a sense of meaning in life.

To address these concerns, a Human-Centered AI (HCAI) approach is required, which redefines AI not as an autonomous agent but as a tool designed to enhance human control and mastery [33]. Following this idea, the goal of AI should not be to remove humans from the process, but to combine high levels of automation with human control. Therefore, designers should prioritize human judgment and responsibility, which require ongoing human oversight to ensure that the systems actions remain ethical and reliable.

4

Methodology

This chapter describes the research perspective and the specific methods used to execute the project. The study is grounded in a Research through Design (RtD) approach, where knowledge is primarily generated through the design process itself. This approach is implemented through the Double Diamond framework, which offers a systematic way to navigate both the problem space and the development of the final solution. To put this into practice, the study combines qualitative methods, such as interviews and thematic analysis, with creative design practices like wireframing and prototyping, ensuring a rigorous transition from initial user insights to an evaluated design artifact.

4.1 Research Through Design

Design, as an academic field, is based on the idea that design practice itself is a method for generating new knowledge. Traditionally, research has been associated with a strict "critical-rational" model where the researcher observes an object from the outside, like a scientist in a laboratory. Christopher Frayling [44] believes, however, that this perspective is limited and that design, through its focus on making and creating, offers unique ways of understanding the world. To define how knowledge is generated within the design field, Frayling introduced three categorizations that distinguish different research approaches: research into art and design, research for art and design, and research through art and design. Research into art and design is the most established form of design research, where design is studied as a phenomenon from a historical, social, or economic point of view. Research for art and design often describes the collecting of reference materials and preparatory studies before the creation of a work. And in research through art and design, knowledge occurs through the act of doing.

The most central approach for projects that involve prototyping is Research through Design (RtD). In this perspective, the design process, in terms of sketching, building prototypes, and iterating, acts as the engine of the research. Instead of just studying existing phenomena, the researcher is creating artifacts to investigate how they affect a situation or a user. RtD assumes that knowledge not only consists of text and numbers (propositional knowledge) but also of tacit knowledge. This is the kind of knowledge that is embedded in the craft and in the decisions made during a design process. By systematically documenting these decisions, for instance,

through research diaries or reports, this tacit knowledge can be made explicit and communicated to others [44].

4.2 The Double Diamond Design Framework

To practically execute the four core activities of the interaction design process outlined in the theoretical framework - namely identifying needs, developing design alternatives, prototyping, and evaluating in iterative cycles - this project utilizes the Double Diamond design framework. Developed by the Design Council [45], the Double Diamond provides a structured lifecycle model that translates the iterative and user-centered philosophy of interaction design into practice. While the universal activities of interaction design define *what* actions must be taken to support the user experience, the Double Diamond framework guides *how* the team navigates these actions. It achieves this by structuring the workflow into two distinct diamonds representing alternating periods of divergent thinking, where the problem and solution spaces are explored broadly, and convergent thinking, where the focus is narrowed and decisions are made, as illustrated in Figure 4.1. Consequently, this model helps to ensure a rigorous transition from the initial qualitative fieldwork to the final evaluated high-fidelity prototype, thereby ensuring that the project addresses both the right problem and develops the right solution.

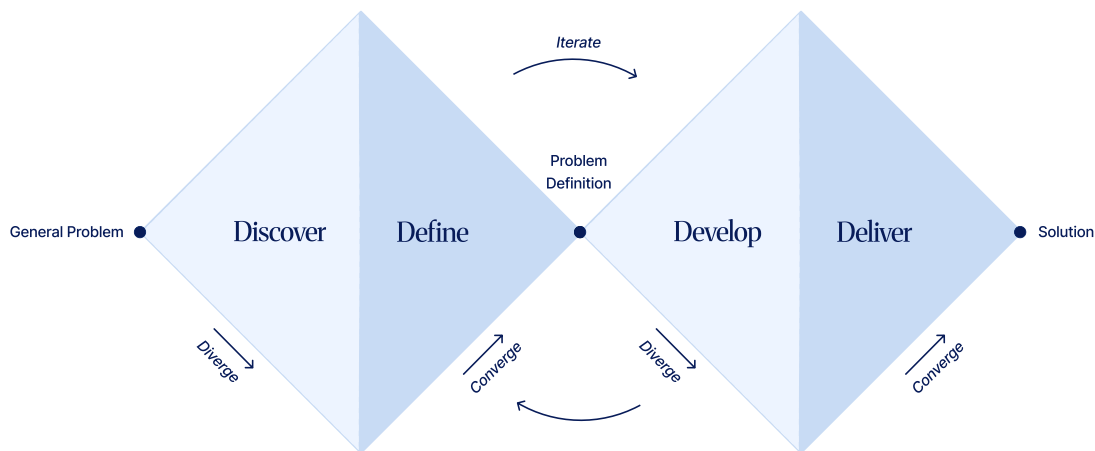


Figure 4.1: The Double Diamond framework illustration, adapted from Design Council [45].

The first diamond focuses on the problem area and begins with the Discover phase. Here, the emphasis is on understanding the actual needs and challenges through empathy and user-centered methods rather than relying on assumptions. By engaging with the people affected, deep insights are gathered. These insights are then processed in the Define phase, where the process is narrowed down to distill the actual core challenge. The goal is to formulate a clear design brief that guides further work.

The second diamond is focused on the solution space and begins with the Develop

phase. During this phase, the process is opened up once again through divergent thinking, allowing for the exploration and generation of multiple creative responses to the defined challenge. Finally, the process reaches the Deliver phase, where the solutions are tested and evaluated on a small scale. By discarding what does not work and iteratively refining the most promising concepts, the final delivery is ensured to be both functional and valuable to the user.

Design Council [45] emphasizes that the Double Diamond should not be seen as a strictly linear instruction manual, but as an iterative process, where you often move back and forth between phases.

4.3 Discover

In the *Discover* phase, methods such as literature review and interviews were employed to explore and understand users needs and challenges.

4.3.1 Literature Review

A literature study is a systematic method to review and evaluate existing research in a specified field [46]. The purpose is to create an overview of the state of knowledge, identify relevant theories, and ensure that the study's research questions rest on a scientific basis.

4.3.2 Pilot Study

A pilot study is a test of the main study that is done to ensure that the planned method works in practice. For example, equipment and instructions can be tested, or interview questions and questionnaires can be tested for clarity. The aim is to detect any problems in advance so that they can be addressed before the main study is carried out [47].

4.3.3 Interviews

Interviews are a research method for collecting first-hand information on people's experiences, opinions, and perceptions through direct contact. The method is often used to reach a depth that surveys often lack and is particularly valuable for capturing nuances in personal expressions and body language, which is best achieved in physical meetings [48]. There are different types of interviews, including structured, unstructured, semi-structured, and group interviews. Structured interviews are based on predetermined questions that are standardized across participants. Unstructured interviews are more exploratory and may vary between participants depending on the direction of the conversation. Semi-structured interviews represent a hybrid between structured and unstructured approaches, where the interviewer follows an interview guide with predetermined questions while also asking follow-up questions to explore responses in greater depth. The choice of interview approach

and level of structure depends on the purpose of the interview and the research questions [30].

4.4 Define

The following section describes the methods used in the *Define* phase to process and interpret collected data, shaping it into defined needs and guiding principles for the solution.

4.4.1 Thematic Analysis

Thematic analysis is a well-used method for identifying, analyzing, and interpreting patterns in qualitative data. Braun and Clarke’s framework [49] for thematic analysis includes the following six phases: 1) familiarizing with the data, 2) generating initial codes, 3) searching for themes, 4) exploring themes, 5) defining and naming themes, and 6) writing the report. The framework emphasizes the importance of transparency, iteration, and reflection to generate reliable insights.

4.5 Develop

This section describes the methods used in the *Develop* phase, including ideation and prototyping methods that helped transform insights into concrete design solutions.

4.5.1 Brainwriting

Brainwriting is an ideation method similar to brainstorming. However, instead of expressing ideas verbally, participants document their ideas individually on paper [50]. This approach reduces the risk of ideas converging too quickly or being influenced by dominant voices, allowing a broader range of independent and diverse concepts to emerge. After a set period, typically between 5 and 15 minutes, the participants can exchange their papers and build upon each others ideas, and to inspire new ideas. In this way, brainwriting combines individual idea generation with collaborative development, supporting both originality and iteration in the ideation process.

4.5.2 Crazy 8’s

Crazy 8’s is a method for rapid idea generation that aims to visualize a large number of solution concepts in a short amount of time. The core of the method is a structured sketching exercise in which participants are challenged to generate eight distinct ideas in eight minutes, corresponding to one minute per sketch [51]. By working under strict time constraints, participants are encouraged to set aside their inner critic and instead prioritize quantity over visual quality or technical detail.

4.5.3 Wireframing

As part of the design process, wireframes are used to create a visual drawing of the system's interface [52]. A wireframe is a simplified representation that focuses on layout and structure rather than aesthetic details like color and shape. The purpose is to establish a clear hierarchy between different elements and to define how navigation should work for the user.

By working with wireframes early in a project, design choices can be quickly evaluated and iterated [52]. This facilitates communication between project stakeholders and ensures that technical solutions rest on a user-centric foundation before more time-consuming design and development work begins.

4.5.4 Prototyping

Prototyping is used as an experimental method to visualize, test, and refine design ideas before implementation [53]. A prototype is a simplified model of a product that simulates its functionality and design. Prototypes can be developed in both physical and digital forms, depending on the specific requirements of the design inquiry. They also vary significantly in level of detail, ranging from low-fidelity paper sketches used for early conceptualization to high-fidelity interactive models that resemble the final product. By creating prototypes with varying levels of detail, usability problems can be identified and addressed early. This reduces the risk of costly mistakes and ensures that the final solution is both technically robust and user-centric.

4.5.5 Design Critique

Design critique is a structured and collaborative method used to critically review and evaluate a design proposal in relation to defined goals. The method is based on group discussions, where the purpose is to improve the quality and effectiveness of the design through constructive and iterative feedback, rather than simply judging it [54]. By identifying strengths, uncovering issues, and exploring alternative solutions, the critique helps refine the design while it is still open to change, unlike more formal evaluations conducted at the end of a process.

For the critique to be effective, it requires clear framing and shared objectives. Participants need to agree on the problem being addressed, and the user needs that guide the design. The process is often facilitated to ensure that feedback remains focused on how well the design meets these goals, rather than reflecting personal opinions or dictating specific solutions. Design critiques are typically iterative and play an important role in supporting reflection, learning, and continuous improvement throughout the design process.

4.6 Deliver

In the *Deliver* phase, methods such as cooperative evaluation and usability testing, supported by a think-aloud protocol, were employed to assess the design and identify usability issues.

4.6.1 Usability Testing

Usability testing is an evaluation method in which a user is observed while performing one or more predefined tasks in an interface. The purpose is to identify challenges in order to improve usability [48].

4.6.1.1 Cooperative Evaluation

One variation of usability testing is cooperative evaluation, which is particularly suitable in early stages of development. In cooperative evaluation, the user performs predefined tasks while thinking aloud, and the designer acts as a moderator during the test. Unlike traditional usability testing, the moderator actively participates by asking questions, and the user is also allowed to ask questions, creating a two-way dialog during the session. This method enables designers to gain immediate feedback and quickly identify usability issues, making it a cost-effective approach when the goal is to improve the design during development rather than evaluate a finished product [55].

4.6.1.2 Think-Aloud Protocol

The think-aloud protocol is a usability evaluation method in which participants verbalize their thoughts, actions, and feelings while completing tasks. This method allows researchers to observe task completion in real time, identify usability issues, and understand what users find confusing or engaging. It is applicable to both prototypes and existing products [48].

4.7 Ethical Considerations of Data Collection

The data collection through interviews and user testing will be conducted in accordance with the ethical consideration guidelines presented by Nii Laryeafio and Ogbewe [56]. Interviews will be audio recorded, and usability testing will be recorded using both audio and screen recording. Participants will be informed about the purpose of the study, what data will be collected, and how the data will be handled and stored, and informed consent will be obtained before participation.

All participants will be anonymized, and no personal identifiers will be included in the final report. Quotes from interviews and usability testing may be used, but all identifying information will be removed or generalized so that individuals cannot be traced.

All data will be stored on OneDrive, which is protected by two-factor authentication, requiring both a password and verification through an authenticator application, and will be deleted once the project is completed. During the storage period, the data will not be shared with any third parties. Only the authors of the study will have access to the data.

5

Execution and Process

The following chapter outlines the process of the study and the execution of the methodology described in Chapter 4. The chapter is structured according to the four phases of the Double Diamond model: Discover, Define, Develop, and Deliver.

5.1 Discover

This section presents the internal data review, literature review, and interviews carried out as the initial divergent phase of the study, with the aim of exploring the problem space and building an understanding of the contextual environment of the study.

5.1.1 Internal Data Review

The project began with a review of company-provided documentation relevant to the study. As part of this, a basic course in SystemWeaver was completed, providing an understanding of the interface, functionality, and ways of working. This initial study also involved studying documentation from the larger, ongoing project of developing an LLM-Framework Architecture for a SystemWeaver integrated GUI presented in section 2.1. Reviewing this helped us to understand the context, goals, and scope of the work this study is part of.

5.1.2 Literature Review

To gain a deeper understanding of the research area, a literature review was conducted focusing on interaction design, artificial intelligence, requirements engineering, and related studies. Search engines such as Google Scholar and Chalmers Library were used to find relevant articles and books. Keywords such as "Artificial intelligence", "Requirements engineering", "Human-computer interaction", "Cognitive load theory", and "Human-AI co-writing" were used. To broaden the theoretical foundation in design theory, keywords such as "Usability" and "User experience guidelines" were also included in the search.

5.1.3 Interviews

Interviews were conducted to explore participants' experiences and opinions related to their use of SystemWeaver, requirements writing, and attitudes toward AI. The interviews followed a semi-structured approach, covering predetermined topics while allowing for follow-up questions.

The interview guide was developed based on insights from the literature review and discussions with supervisors at GTEU and Chalmers University of Technology. The literature review highlighted trust as a central challenge in Human-AI Interaction and AI in Requirements Engineering. Therefore, questions were included to explore participants' current use of AI, their attitudes towards AI support in requirements writing, and factors influencing trust in AI-generated output. Discussions with the supervisors helped ensure that the interview guide also captured the practical context of requirements engineering at GTEU, including the use of SystemWeaver, current workflows, and existing challenges. The complete interview guide is presented in Appendix A.

Participants were selected through purposive sampling. Potential participants were identified by the supervisor at GTEU based on their experience of writing, reviewing, or managing requirements in SystemWeaver. These activities were central to the research focus and were therefore used as the primary selection criterion. From an initial list of 15 potential participants, eight were selected. During the selection process, an effort was made to include participants from different departments and with somewhat different responsibilities in order to capture a broader range of perspectives. Participants were first contacted by email to ask for their participation, after which the interviews were scheduled.

5.1.3.1 Interview Procedure

A total of eight interviews were conducted. The first interview functioned as a pilot study to test the audio recording, interview structure, questions, and planned duration. As these elements functioned as intended and no modifications were required, the pilot was included in the subsequent data analysis. Six interviews were conducted on-site at Geely's office, and two were conducted online via Microsoft Teams, as those participants were unable to attend in person for the scheduled time. All interviews lasted approximately 30 to 40 minutes and were conducted in Swedish, the participants native language.

Each interview began with a brief introduction to the projects background and purpose. Participants were informed about what data would be collected, how it would be used, stored, and later deleted upon completion of the project. They were assured that all responses would be anonymized and not traceable to any individual. Written informed consent, including consent to audio recording, was obtained prior to the interviews.

The interviews consisted of an introductory part with general questions, followed by successively more specific and in-depth questions on the subject of the project. At the end, participants were given the opportunity to add further thoughts or

reflections. During the interviews, one interviewer led the conversation and asked questions from the interview guide, while the other handled the recording and asked follow-up questions. Before the interview ended, participants were asked whether they would be willing to participate in subsequent evaluation sessions for the project.

5.2 Define

Following the interviews, the gathered data underwent a qualitative analysis to establish a deeper understanding of the specific context and problem areas. The insights gained from this process served as a bridge between research and creation, providing a grounded foundation that guided the subsequent development of the design solution.

5.2.1 Thematic Analysis - Interviews

The qualitative data gathered from the semi-structured interviews underwent a rigorous thematic analysis following the six-phase framework established by Braun and Clarke [49]. To capture the nuances of the participants' experiences and preserve the authenticity of the material, the analysis was conducted using an inductive approach where codes and candidate themes emerged originally from the dataset rather than being fitted into pre-defined matrices or codes.

The analysis began by transcribing the audio recordings from the interviews and checking them against the original recordings. This ensured that the transcriptions matched the material and allowed us to become familiar with the data. Following the initial transcription and data familiarization phase, relevant statements and quotes were extracted from the material, and then coded inductively. Some quotes were assigned multiple codes, as they related to several different aspects of the material. Table 5.1 illustrates a few examples of how interview quotes were assigned codes during the thematic analysis.

Table 5.1: Example of the coding process.

Interview quote	Coded for
"There are so many departments here and everyone works with it in different ways, whether its functions, physical items, or physical attributes."	Differences between departments
"You can tell that it gets a certain personal touch. Different areas have been written by different people, especially linguistically."	Individual variations in requirement formulation

The initial coding resulted in a large number of detailed codes describing participants interactions with SystemWeaver, technical workflows, and attitudes towards AI. To create an early overview of the data, these codes were visually grouped in FigJam based on conceptual similarities, resulting in a total of 21 initial preliminary themes

shown in figures 5.1 and 5.2. These figures illustrate the extensive breadth of the initial material and the complexity of the early clustering phase. At this stage, a high number of specific clusters were maintained to ensure that no thematic nuances were lost, as well as providing a clear initial understanding of the various emerging areas.



Figure 5.1: Preliminary themes.

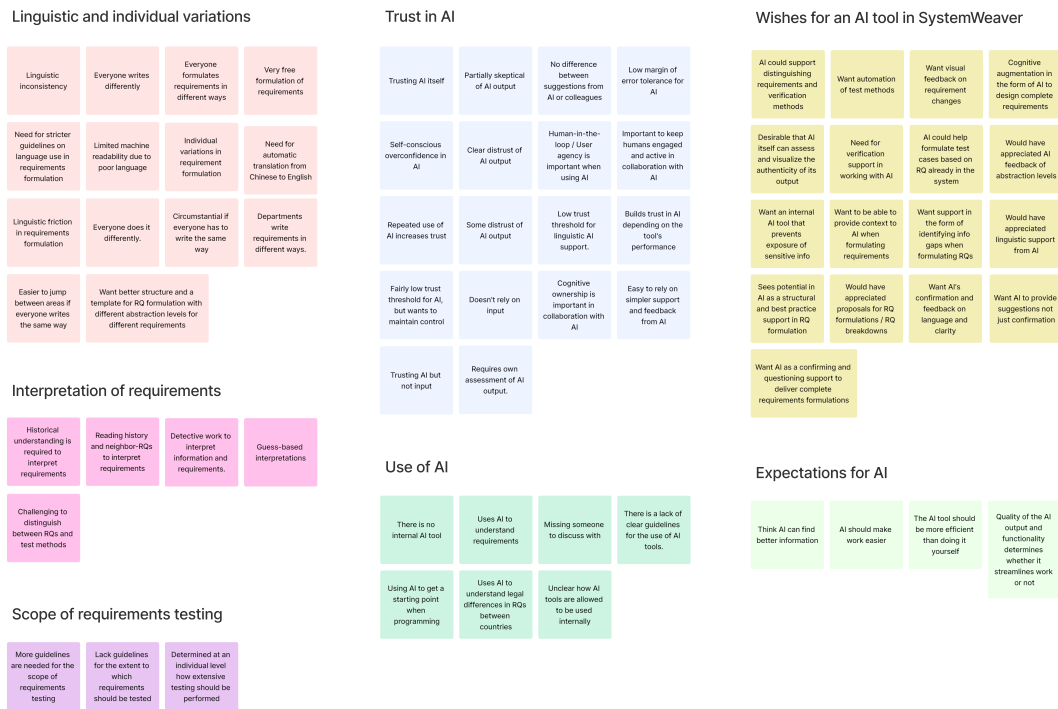


Figure 5.2: Preliminary themes.

While the initial clustering provided a detailed representation of the raw data, the large number of clusters made it difficult to create a clear overview. Therefore, the preliminary themes from the first grouping of codes were reviewed and refined. Some codes were moved to other themes, while others were split to better reflect the data. For example, Trust in AI was divided into Trust and User Agency, as these captured different aspects of the user experience. Several themes were excluded as they were considered outside the focus of the study, for example, broader aspects of SystemWeaver such as system efficiency and poor learnability. This iterative refinement led to 12 candidate themes, as seen in figure 5.3.



Figure 5.3: Candidate themes.

While the second iteration of grouping codes provided a clearer view of the problem space, several themes were still perceived to have overlapping characteristics and interdependencies. Therefore, a final iteration was conducted in which the 12 candidate themes were combined, contextualized, and refined. Rather than discarding the 12 candidate themes, this final step focused on restructuring them into broader themes. For instance, the different types of problems or challenges encountered in the requirements work were grouped together into one theme. This resulted in five final themes presented in figure 5.4.



Figure 5.4: Final Themes.

5.2.2 Thematic Analysis Findings

The thematic analysis resulted in five themes, presented in Figure 5.4. Each theme builds on several aspects identified during the analysis. In the following sections, the themes are described in more detail and supported by direct quotes from the interviews.

5.2.2.1 Challenges in Current Workflow

Participants described several challenges in the current workflow, particularly related to consistency, interpretation, and determining the appropriate level of detail in requirements.

The most recurring problem was lack of structure for writing requirements, such as standardized methods and guiding principles. Several participants described that this creates uncertainty and that they sometimes have to rely on guesswork. As one participant explained:

It feels very vague (writing requirements and test cases). It would need to be much more structured in order to ensure higher quality, and you end up guessing a bit since there are no clear processes or frameworks to follow. - P7

Participants further described that the absence of standardized practices contributes

to variation in how requirements are written. Requirements were said to acquire a certain "personal touch", influenced by individual experiences, professional background, and language preferences. As P4 expressed:

[...] you can tell that it gets a certain personal touch. Different areas have been written by different people, especially linguistically.

These differences were described as extending beyond individuals and also reflecting variations between roles and departments. Participants explained that requirements are formulated differently depending on the functions and artifacts involved.

There are so many departments here and everyone works with it in different ways, whether it's functions, physical items, or physical attributes.
- P2

Several participants described that these variations make requirements more difficult to interpret. Understanding a requirement often required extensive "detective work" (P5) to identify the original intent.

Related to this, participants described that information gaps frequently occur throughout the workflow. Information was said to sometimes "fall between the cracks" when transferred between departments, as different units have different focuses and access to background information. As a result, requirements may not always be fully understandable to other stakeholders or downstream engineers. P5 explained:

It's important that any change you make does not have unexpected consequences and is understandable, which is surprisingly difficult. [...] When someone else receives it they don't have the same mental picture that I have built up from the analysis.

Furthermore, participants described that organizational differences also affect how requirements are divided and delimited. A central part of the work involves adapting incoming requirements to the needs of their own department and determining the appropriate level of detail. Requirements that are too detailed were described as time-consuming to manage, while requirements that are too broad were described as difficult to follow-up. P4 explained the difficulty of this balance:

[...] it is mostly about making sure the requirements are written in a reasonable way, trying not to include multiple requirements in the same one, but also not breaking them down too much. You have to find a balance in how extensive a requirement can be before you need to split it.

5.2.2.2 Attitudes Towards and Use of AI

This theme captures participants' attitudes towards AI, including their current use of AI tools, and their views on future AI support in SystemWeaver.

Some participants described that they already use AI tools in their work today, particularly for code generation and as a source of information. A few also reported using AI in their requirements management work. However, at the time of the interviews, no dedicated internal AI-tool was available within the organization, leading

participants to reformulate and de-identify the context to avoid leaking sensitive information to external platforms.

Despite these limitations, participants expressed a positive attitude towards the integration of an internal AI tool specifically designed for requirements management in SystemWeaver. They see strong potential for streamlining workflows and optimizing several important aspects of their work, some of which are further described in the theme Functionality.

Furthermore, efficiency emerged as a key factor in their willingness to adopt such a tool. Participants emphasized that it must be more efficient than performing the work manually to be considered valuable. As P5 stated:

[...] the important thing is, of course, that it should be faster to do this than to do it yourself [...] there are many examples in SystemWeaver where there are intended methods for performing certain tasks that are slow [...] it simply has to be time-efficient.

5.2.2.3 Functionality

Participants expressed several expectations regarding how AI could support their work and help improve the quality of requirements.

Linguistic support was one of the most frequently requested functionalities. Participants described how Swedish sentence structures often appear in requirements, as well as requirements written in Chinese, which can create ambiguity for international teams or external suppliers. They also noted that such linguistic inconsistencies can make it difficult for translation tools to interpret the content correctly, increasing the risk of misinterpretation. P4 expressed it like this:

[...] linguistically, I think that AI would be very helpful. The problem is often that not many people here are native English speakers, but all requirements are written in English. Sometimes you can clearly see Swedish sentence structure in the text. It is usually understandable for us, but it may not be as clear for someone with another language background, and it can also make it difficult for translation or other tools to handle the requirement properly.

Another recurring request was AI support to ensure complete and testable requirements. Participants expressed a desire for an AI that could act as a reviewer, suggesting missing information or flagging requirements that are too vague to be tested and verified.

It would be very helpful if an AI could suggest or give feedback, for example, you wrote this here, you might want to add this as well, or what you have written does not seem very testable [...] - P7

5.2.2.4 Trust in AI

This theme captures participants' perspectives on trust in AI and the factors that influence it. Almost all participants emphasized that trust is built through consistent performance and the ability to verify the output. If the AI frequently makes mistakes, trust is quickly lost. A hallucinating AI tool that requires constant correction can result in more effort than benefits. As P8 explained:

If you start using an AI tool and it gives good suggestions, then you would have more trust to let it create more [...] But if it messes up, so to speak hallucinates [...] you would lose some trust. You might start to feel that it takes more time to sit and correct everything it produces than to just do it yourself.

Participants also highlighted transparency as an important factor in building trust. They expressed a need to understand what data the AI uses and the reasoning behind its output.

5.2.2.5 Control

This theme captures participants' views on the role of human control in AI-supported work. In addition to trust, control was also described as important in the given context. Participants emphasized that AI suggestions should never be accepted blindly. Instead, they should be actively reviewed and manually approved, with users always making their own judgment.

Some participants pointed out that there is a risk that users become too passive and stated that the work still requires a person who can think creatively and understand the bigger picture. In addition, one participant expressed that the process of writing and reviewing requirements is how they build their knowledge. P1 explained it like this:

I think it can be good to start writing yourself. If you havent been part of the process, you dont remember things afterwards. There is value in knowing the material without having to look it up, for example in a meeting when something comes up. Its not the same if you havent been involved in producing it.

5.2.3 Defining The Use Case

While the study investigates AI-assisted writing in requirements engineering, the interviews revealed a variety of potential use cases for AI support, both related and unrelated to writing. Writing-related use cases included writing requirements, generating test cases from requirements, and naming signals and parameters. Non-writing-related use cases included identifying verification methods, mapping between requirements, and extracting status information from SystemWeaver.

This breadth made it necessary to define the use case more clearly. Based on discussion with our supervisor at GTEU, we decided to focus on requirements writing, as it can be seen as the foundation of the requirements engineering process. Poorly

written requirements can affect all downstream work, such as testing and validation. By focusing on this activity, the study can address an important core problem without becoming too broad or scattered.

Category	Use case	Included in scope	Reason
Writing	Write requirements	Yes	Core focus of the study
Writing	Generate test cases from requirements	No	Downstream activity in the testing phase
Writing	Name signals and parameters	No	Implementation-level detail, not requirement authoring
Non-writing	Identify verification methods	No	Validation and analysis activity outside writing scope
Non-writing	Map between requirements	No	Traceability/modelling task rather than writing
Non-writing	Extract status from SystemWeaver	No	Tool-specific information retrieval, not writing activity

Table 5.2: Overview of potential AI-supported use cases and scope selection

5.2.4 Design guidelines - First iteration

Based on the findings from the thematic analysis of interviews, an initial set of design guidelines was developed to support the subsequent Develop phase, particularly the ideation process. The guidelines represent a preliminary proposal rather than a final set, as they were created early in the process and intended to evolve and be refined throughout the project. The first iteration consisted of ten guidelines, formulated to be clear, concise, and easy to understand, in order to ensure their usability during ideation. The guidelines were as follows:

Trust

- **G1.** Be transparent about what data and sources the AI uses.
- **G2.** Be transparent about how AI arrives at its conclusions.

Control

- **G3.** Enable users to review and verify AI-generated output before use.
- **G4.** Make it easy to edit or ignore AI-generated output.
- **G5.** AI should never make decisions on its own; the user should always make the final decision.

- **G6.** Provide feedback on changes made by the AI.

Functionality

- **G7.** Offer linguistic support to ensure requirements are clearly and well-formulated.
- **G8.** Help identify missing information to ensure complete requirements.
- **G9.** Assist the user in breaking down requirements to ensure correct scope.

Design

- **G10.** Allow for flexibility across different contexts and departments.

5.3 Develop

The development of the design solution focused on turning the identified needs into a concrete concept through an iterative process. By exploring different ideas and visualizing them as wireframes and prototypes, the design was continuously reviewed and refined.

5.3.1 Ideation

Based on the findings of the data collection and analysis, an ideation phase was conducted to generate potential solutions. The ideation methods Brainwriting and Crazy 8's were used to produce multiple ideas. The goal was to develop concepts that follow the defined design guidelines.

5.3.1.1 Brainwriting

The initial ideation method used was Brainwriting, where one session was conducted for each guideline. During each session, ideas were individually written for two minutes. The ideas were then passed to the other member, who had two minutes to add, modify or expand on the ideas. After this, the ideas were explained and discussed together, and if the discussion led to additional ideas these were also written down on paper. The session concluded with a selection of the most promising and feasible ideas for each guideline, which were then carried forward to the next iteration of ideation. See figure 5.5 (left) for an example of the brainwriting results for G7.

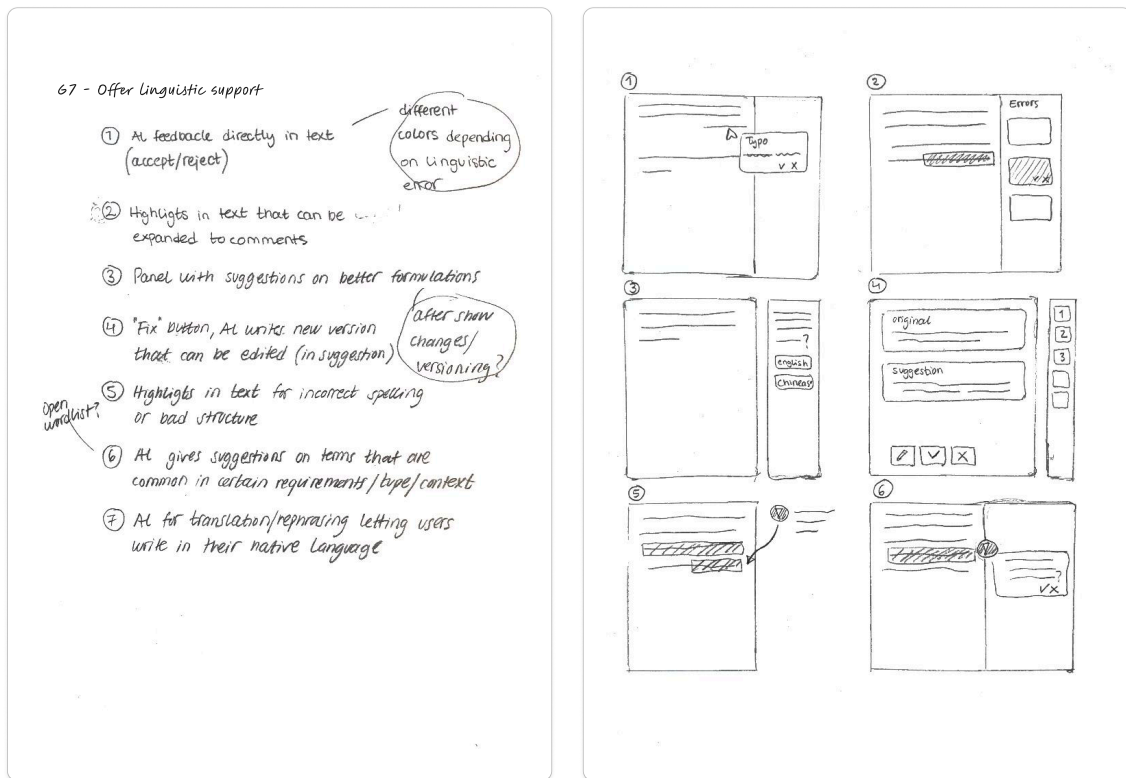


Figure 5.5: Example of Brainwriting (left) and Crazy 8s (right) ideation for G7: Offer linguistic support to ensure requirements are clearly and well-formulated.

5.3.1.2 Crazy 8's

In the second iteration, Crazy 8's was used to generate sketches based on the outcomes of the brainwriting session. As mentioned, this phase focused on a selection of ideas from the brainwriting activity that were considered to have the greatest potential to fulfill the established guidelines effectively. One Crazy 8's session was conducted for each selected idea, where up to eight sketches were generated individually within eight minutes. In some cases, fewer than eight sketches were produced due to time constraints or a lack of further ideas.

Following this, all sketches were presented individually for discussion and evaluation. The most relevant and effective concepts were selected based on how well they addressed the guidelines, similarly to the approach used in the brainwriting session. In some cases, elements from multiple sketches were combined to form more refined solutions. These selected and combined ideas served as the foundation for the development of wireframes and an initial prototype. Figure 5.5 (right) illustrates an example of the Crazy 8's results for G7.

5.3.2 Wireframing

Wireframes were created in Figma based on ideas generated during the ideation phase. The purpose was to visualize the main functionality and interaction flow before moving on to high-fidelity prototyping. The resulting concept introduced a

side panel integrated within SystemWeaver. The intention is for the panel to be expandable into a full-screen view or repositioned within the interface depending on the users working context and preference. The main interface (left side in Figure 5.6) presents an analysis of the requirement description within SystemWeaver, followed by an AI-generated suggestion for refinement. The suggestion is accompanied by sources to support transparency and enable verification. To further support control over the input context, a dedicated icon button allows users to define the AIs knowledge scope by selecting specific project documents or enabling external web sources. Users can also chat with the AI to refine the suggestion or edit it manually before committing the changes back to the system (right side in Figure 5.6).

To support review before changes are committed, a dialog enables comparison between the original requirement and the AI-generated suggestion, where color-coded highlights are used to indicate changes between the two versions (left side in Figure 5.7). A secondary side panel was also included, providing access to the history of previous chats (right side in Figure 5.7).

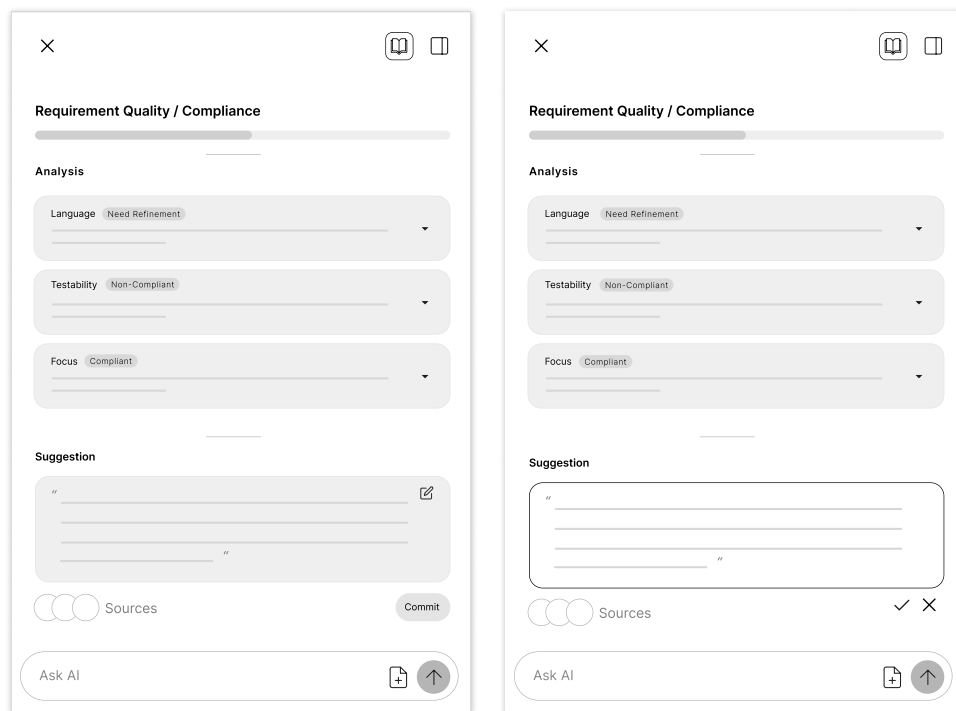


Figure 5.6: Wireframes of the main interface and editing state.

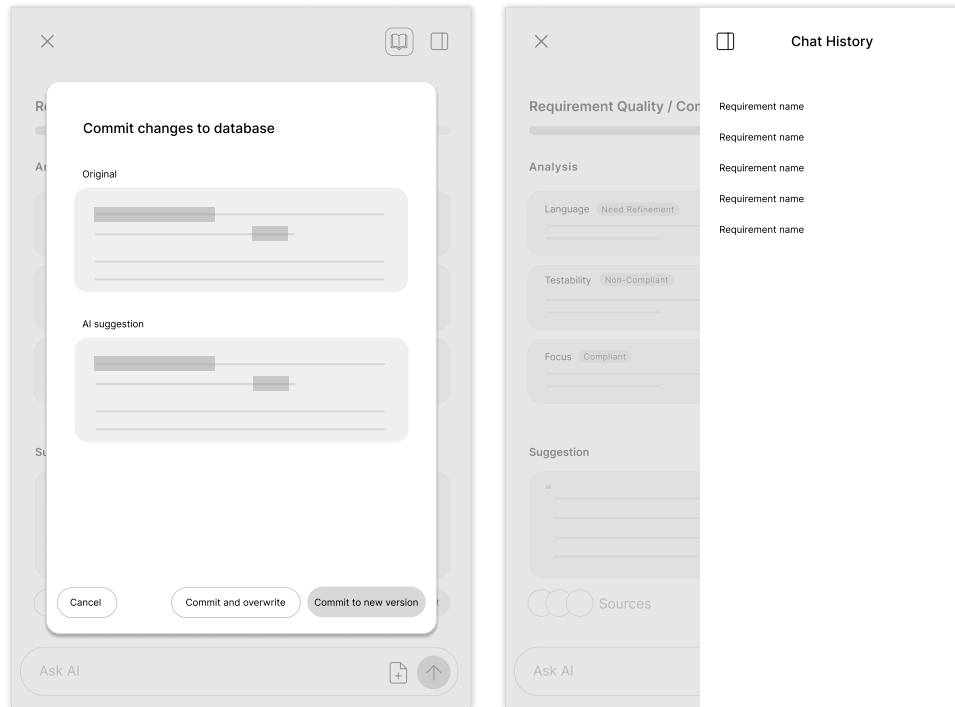


Figure 5.7: Wireframes of the review dialog and chat history panel.

5.3.3 Design Critique

A design critique session was conducted with two experts from GTEU, one a SystemWeaver application specialist and the other a software developer with experience in AI development. The session began with a brief recap of the identified themes and resulting design guidelines. This was followed by a demonstration of the wireframes, including the reasoning behind design decisions. Experts were encouraged to provide feedback on features, general functionality, and technical feasibility. Their feedback was written down in a shared document, ensuring a detailed record of the discussions.

One of the most important takeaways from the session was that users might want to start a new chat within a requirement if the previous conversation has taken an unhelpful direction. Supporting this, the history view should allow multiple chats for the same requirement to be displayed and include filtering options for easier navigation. Another takeaway was making the analysis section expandable and collapsible because of the limited space in the interface. Additionally, the labeling of the commit button should be revised, as its current wording does not clearly communicate what will happen when it is clicked. Experts also mentioned that users might want to compare the original requirement with the AI-generated suggestion earlier in the workflow, rather than only in a commit-triggered dialog, so that differences are visible while working on the proposal. The session concluded with a structured set of action items that guided further refinement during the prototyping work.

5.3.4 Prototype - First iteration

Based on the initial wireframes and feedback from the design critique session, a high fidelity prototype was developed. The prototype was designed in Figma, where all interface elements, including components, color styles, and typography, were created from scratch. This approach was taken due to the absence of an established design system for this context. Furthermore, the AI tool is intended as a Geely specific extension within SystemWeaver, therefore strict adherence to SystemWeavers existing visual style was not considered necessary. The functionality and design decisions are described in the following sections.

Requirement Analysis - The AI provides an analysis of the requirement description in SystemWeaver, displayed through a status bar and three categories: Language, Testability, and Focus. The entire analysis section can be expanded or collapsed to support flexible use of screen space, as highlighted in the design critique session. Within this section, each category can also be expanded or collapsed individually. These categories reflect both issues identified in the interviews and established principles for well-written requirements. Language evaluates the clarity of the text, Testability checks whether the requirement is written in a verifiable way, and Focus assesses whether it addresses a single specific aspect. Each category is assigned a status tag: Compliant indicates no issues, Needs refinement indicates areas for improvement, and Non-compliant indicates significant issues (Figure 5.8). Based on this analysis, the AI generates a suggestion that can be further refined and edited.

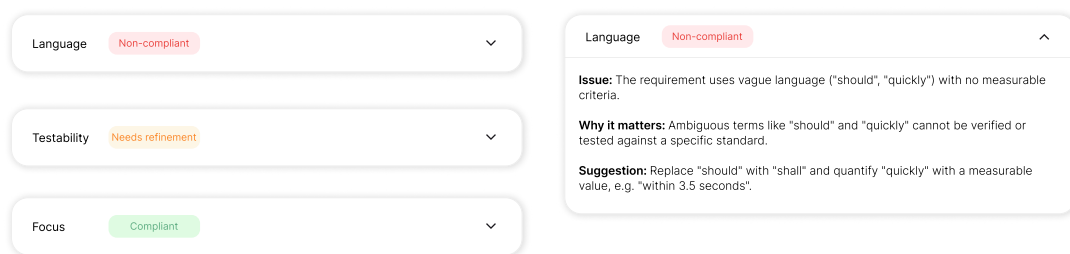


Figure 5.8: Overview of the requirement analysis components, including the categories Language, Testability, and Focus with status tags.

Editing and Comparison - The edit and comparison features can be accessed through icon buttons in the top right corner of the AI-generated suggestion. The comparison feature presents the original requirement alongside the AI-generated suggestion, with changes highlighted using color to support easy comparison. This design builds on existing SystemWeaver functionality, where color coding is used to represent changes between different versions of requirements and support understanding of what has been updated. The edit feature allows users to manually adjust the suggestion before committing it back to the system, giving users control over shaping the final outcome (Figure 5.9).

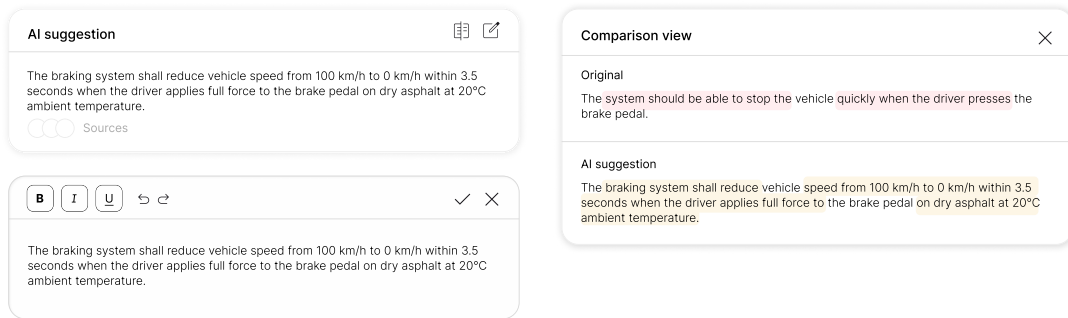


Figure 5.9: Components for editing and comparison, showing edit mode and the comparison view with highlighted changes.

Review and Commit - The Review and Commit button opens a dialog showing the original requirement and the AI suggestion side by side. The user can either commit the suggestion as a new version or overwrite the existing requirement. Committing as a new version allows the user to keep the original requirement description while saving the updated one as a separate version. If the AI suggests splitting a requirement and generates multiple suggestions, the user can choose to open each suggestion as a new requirement or select one of them to either create a new version or overwrite the existing requirement (Figure 5.10).

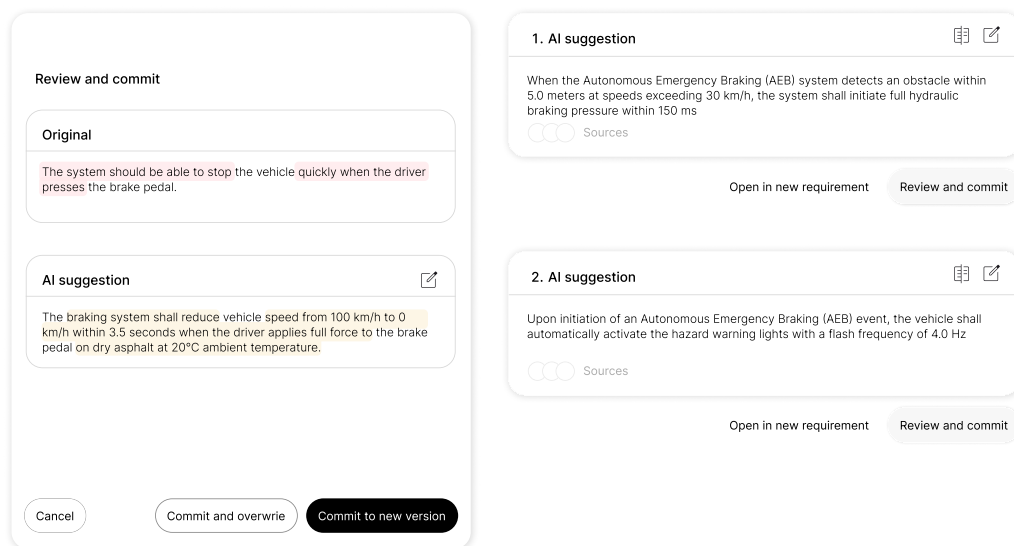


Figure 5.10: Components for review and commit, showing the comparison dialog (left) and buttons for handling multiple suggestions (right).

History View - The history view is a secondary side panel that can be accessed through the icon in the top right corner of the interface. It provides an overview

of previous chats across different requirements. If multiple chats exist for the same requirement, they are grouped in a dropdown under that requirement. This was an important consideration from the design critique session, where experts highlighted that users may want to start a new chat within a requirement if a previous conversation goes in an unhelpful direction. To support navigation, filtering options and a search bar was also included as well as basic management actions such as pinning, renaming, and deleting chats (Figure 5.11).

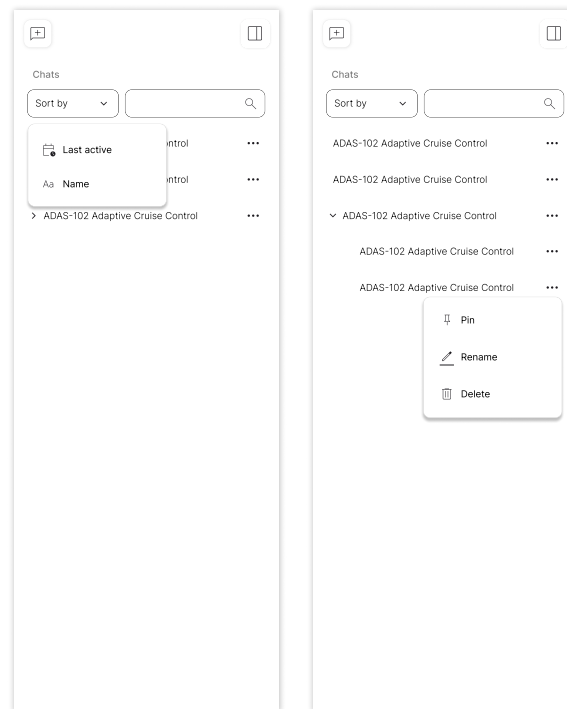


Figure 5.11: History view showing chat overview with filtering options (left) and additional management actions such as pinning, renaming, and deleting (right).

5.3.5 Functional Prototype

While the user interface design was conducted in Figma, Figma’s interactive prototyping capabilities did not provide the level of interaction wanted for testing the prototype. For instance, the user needed to be able to type real input, which was not possible with Figma prototyping. To achieve the desired interactivity, the development of the prototype continued in Lovable, an AI-powered development platform. This further allowed us to integrate a functional Gemini API, enabling real chat functionality. By taking this approach, participants in the user tests were able to interact and chat with an actual AI, resulting in an authentic experience with fewer constraints on what could be tested and how to interact with the prototype than in a simulated flow.

The work in Lovable focused on translating the design from Figma frames into a

functional environment. We began by implementing the Figma frames to build the sidebar, using a combination of detailed prompts and screenshots to ensure that elements like color codes, border radius, and shadows precisely matched the original Figma design. The functionality was then refined through an iterative process of continuous prompting and testing of the behavior. When navigation and functions worked as expected, the Gemini API was integrated to power the chat. Additionally, to place the AI tool in a realistic context for the user testing of the prototype, a basic SystemWeaver-like UI was created in Figma and then implemented in Lovable. This served as a simplified base interface where the AI extension could be integrated as a side-pane and tested in an environment similar to the expected. This system and AI integration ensured that the tool felt like a native part of the surrounding environment rather than a standalone application, allowing AI to reference the requirement analysis section and provide dynamic suggestions.

In addition, to ensuring visual consistency with the Figma designs and integrating the necessary interactivity, another central part of the process was defining the AI's behavior and personality. We especially prompted the AI to avoid "conversational fluff" and instead provide direct, professional feedback with clear explanations for its suggestions. The desire for this behavior lies in both UX and environmental factors, since the sidebar format provides a limited display area, unnecessary conversational filler would force excessive scrolling and obscure the actual value of the output. Also, as a critical aspect for the users was for the tool to be less time-consuming than doing the work manually, it was essential that the AI provided concise and efficient responses to meet this user need.

Finally, the prototype was prepared for the user evaluation to ensure expected functionality and consistency across all user sessions. By loading the system with specific content, such as mock requirements and creating functional shortcuts for the different test cases, we ensured that the starting point and conditions for each test were equivalent. This allowed the participants to focus entirely on the interaction with the AI and the quality of its feedback while also making the results from different users easier to compare and analyze. The finalized functional prototype is seen in Figure 5.12.

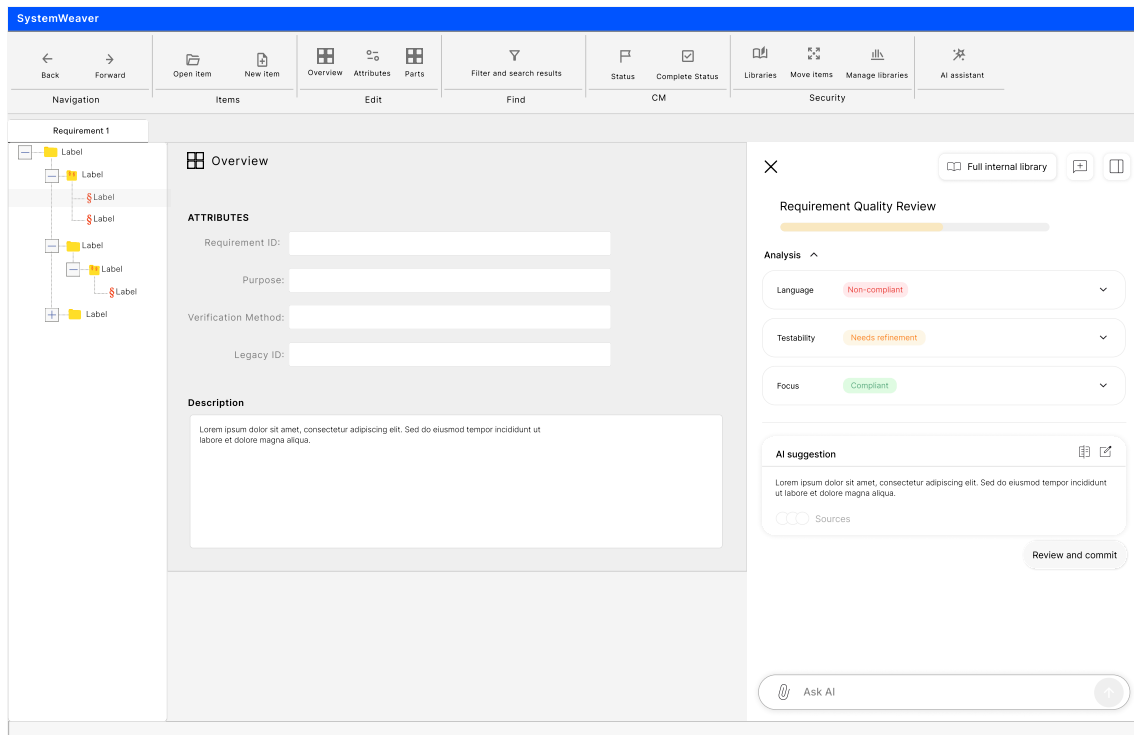


Figure 5.12: Functional high-fidelity prototype demonstrating the AI tool’s placement within the surrounding system context.

5.4 Deliver

The final phase focused on evaluating the design through end-user testing and refining the solution based on the feedback. Engaging with users ensured that features and terminology felt intuitive, validating that the solution effectively addresses the core problem and provides a meaningful response to the user needs identified earlier in the process while feeling like a natural fit in their work environment.

5.4.1 End-user Testing

The end-user testing was conducted as a cooperative evaluation with four participants from the previous interview round. The sessions were carried out in person at GTEU’s office and lasted approximately 30 to 40 minutes each (See Figure 5.13 for the physical setup used during the evaluation sessions).

Each session began with a brief introduction outlining the background, scope, and purpose of the evaluation. The structure of the session was explained, and participants were encouraged to think aloud while completing the tasks. Informed consent was obtained prior to each session, including permission to record both audio and screen activity.

The testing was structured around three scenarios, each with associated tasks and follow-up questions. The scenarios covered writing and refining requirements with the support of AI, including using the AI to search for relevant information when

needed. The tasks involved practical actions within these scenarios, such as editing AI-generated suggestions or publishing them to the SystemWeaver database. Each task was followed by questions exploring aspects such as clarity of information, perceived usefulness, and trust in AI outputs. The full test guide is provided in Appendix B. During the sessions, one researcher facilitated the tasks and asked follow-up questions, while the other handled recording and took notes.

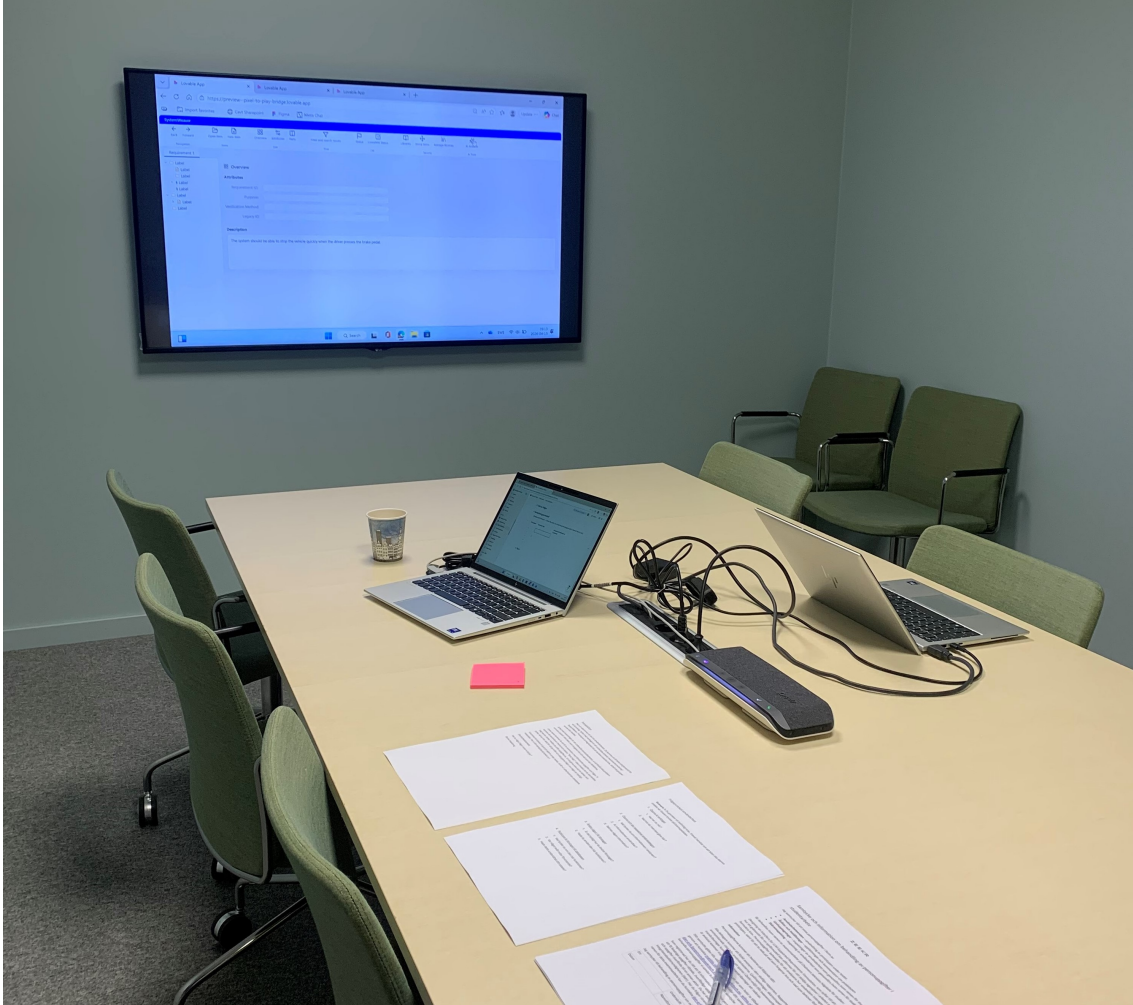


Figure 5.13: Physical setup of the evaluation sessions, showing the test environment and equipment arrangement.

5.4.2 Thematic Analysis - User tests

The data from the cooperative evaluation, including audio recordings and observational notes, were analyzed using Braun and Clarkes [49] framework for thematic analysis, as described in Section 5.2.1. The initial steps of the analysis followed the same procedure as previously described, beginning with transcription of the audio recordings and reviewing them alongside the original material to ensure accuracy and familiarization with the data.

The material was then coded using a combination of deductive and inductive ap-

proaches. The deductive codes were based on the evaluation guide and its predefined questions, while the inductive codes were developed to capture patterns and insights emerging directly from the data. Once the coding was completed, all quotes were compiled under their respective codes (see Figure 5.14).

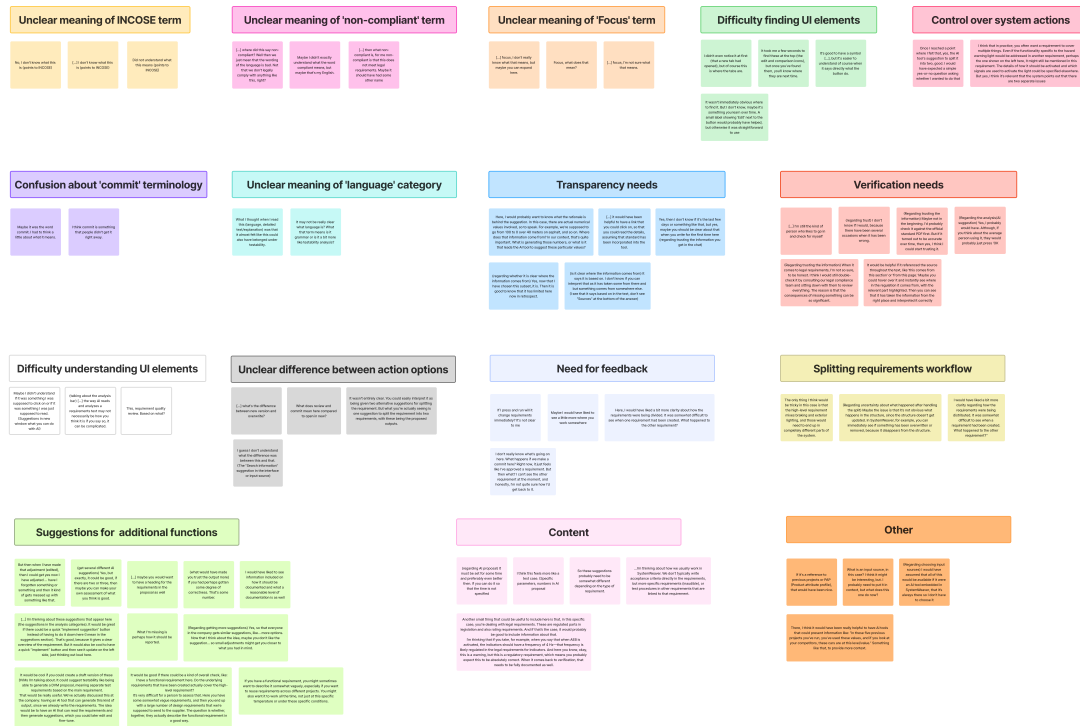


Figure 5.14: Quotes organized under their respective codes.

The codes were then grouped into five preliminary themes (see figure 5.15), which were subsequently reviewed against the original quotes and transcripts to ensure consistency and accuracy. Based on this review, the themes were further refined in a second iteration. During this stage, the theme Suggestions/Other was removed, as it primarily consisted of comments on example texts in the interface and suggestions for additional functionality beyond the scope of the prototypes intended use case. Other themes were renamed to better reflect their content and meaning.

Through this process, four final themes were identified: Unclear Terminology, Unclear UI Elements, System Feedback and Control, and Trust and Verification (see figure 5.16).

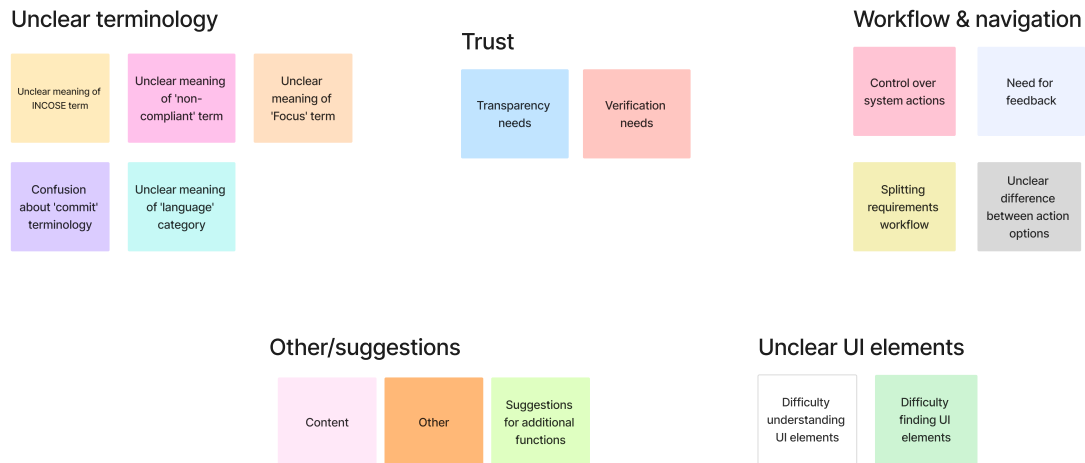


Figure 5.15: Preliminary themes from first iteration of grouping codes.

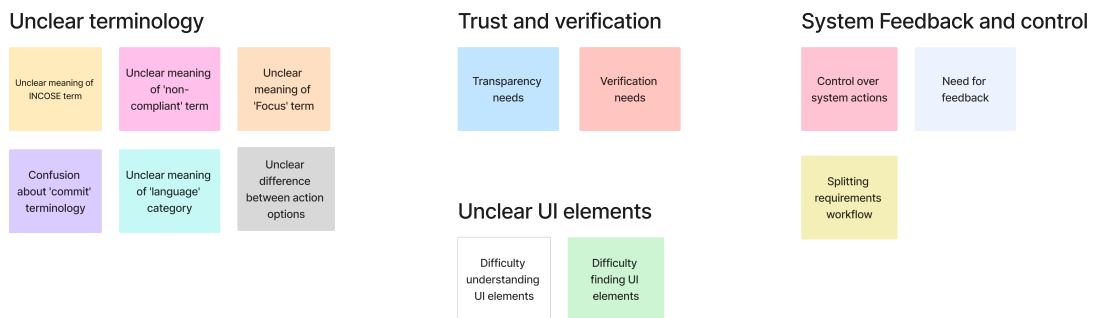


Figure 5.16: Final themes.

5.4.3 User Test Findings

The cooperative evaluation resulted in four themes reflecting participants experiences with the prototype. Overall, participants were positive about the functionality and design, and saw potential in using AI to support writing requirements. The tool was perceived as helpful for identifying unclear or underspecified requirements and as a valuable complement to existing workflows. Participants also provided suggestions for improvements and further development, indicating engagement with the concept and interest in its future potential. The following sections describe each theme in more detail, supported by quotes from the users tests.

5.4.3.1 Unclear Terminology

Participants frequently struggled with understanding specific terms used in the interface, which created uncertainty about meaning and intent. Words such as compliant, focus, and commit were perceived as ambiguous or lacking context. This confusion made it harder for participants to understand what actions meant or what outcomes to expect. As one participant noted:

"I think commit is something that people would not understand right away" - P1.

5.4.3.2 Unclear UI Elements

Interface elements such as the progression bar, icons, and buttons also caused uncertainty. Some functions were difficult to discover, while others lacked clarity about whether they were interactive or informational. One participant expressed this confusion:

I did not quite understand whether it was something I was supposed to click on or something I was just supposed to read. - P3

5.4.3.3 System Feedback and Control

Difficulties were observed in how users understood the impact of their actions, particularly when splitting requirements or committing changes. It was often unclear what happened after a commit, how to return to previous steps, or how multiple requirements were managed within the system. One participant described this uncertainty:

Im not sure whats happening here? What happens if I commit here [...] Now I just feel like I have approved a requirement. - P2

Some participants also expressed a need for greater control over AI-generated outputs and system actions. They preferred to decide when improvement suggestions should be generated and if a requirement should be split or not, rather than having these initiated automatically by the system. P1 expressed it like this:

"When I get to the point where I feel like yes, sure, the AI-tool suggests splitting the requirement in two. Then I would have expected a yes or no question, do you want to do that?"

5.4.3.4 Trust and Verification

Participants continued to emphasize the importance of transparency and verification in building trust in AI. They appreciated the ability to limit the AIs knowledge base to specific project documents. However, they requested sources to be shown directly in the AIs responses, both in the text and in a more specific form, such as page or paragraph references in documents, along with clear explanations and links to the standards used in the analysis. As P3 expressed

"It's good if it references a lot in the text [...] so that you can maybe hover over it and it will show you directly where in the legal requirement it comes from [...]"

5.4.4 Prototype - Second Iteration

The second iteration of the prototype, based on feedback from user testing, introduced a series of changes, mostly minor ones such as clarifying terminology and UI

elements. The analysis category names were revised: language was changed to clarity and testability was changed to verifiability, while the category focus remained unchanged. The decision to change the category names as well as keeping the name focus was made to better correspond with the terms used by the requirements standard INCOSE. To help users better understand what the analysis is based on and what the categories mean, an information icon with an accompanying tooltip was added. The terminology used for status tags and buttons in the commit workflow was also revised to better align with user language and familiarity. Furthermore, to improve the visibility of interactive elements, the icon buttons for edit and compare were given a subtle background.

Another change concerned the visibility of the analysis section. Previously, the analysis section was only displayed when there was text in the description field in SystemWeaver. This was changed so that the section is always visible but in a disabled state when there is no text. This change was made to reinforce user understanding of when analysis functionality is available, addressing issues related to unclear UI elements.

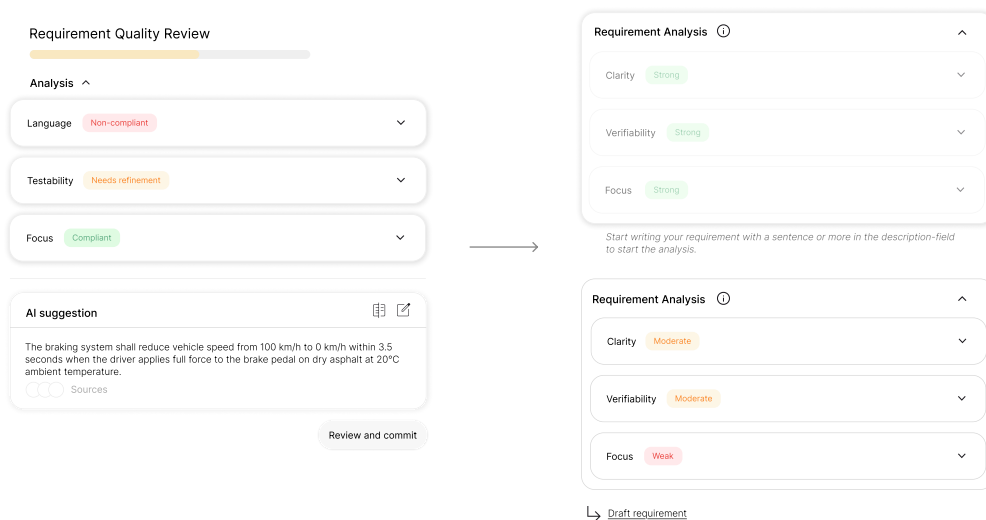


Figure 5.17: Analysis refinement.

Furthermore, it was decided to introduce a button to generate improvement proposals based on the analysis, instead of these being created automatically. This decision was directly informed by the theme System feedback and control, where participants expressed a need for greater agency over when AI-generated suggestions should appear. The purpose of this change was to give the user more control over system-initiated actions. The same applies to whether a requirement should be split or not. During the evaluation, participants stressed that the scope of a requirement often depends on the level of abstraction at which it is written. In some cases, it is both reasonable and necessary to include multiple aspects in the same

requirement.

In addition, the analysis bar linked to the analysis categories was removed. Most participants did not pay attention to this feature, nor was it perceived as particularly relevant in the context. Instead, it risked contributing to a more complex and difficult-to-understand interface without adding clear value.

Another change concerned the way multiple proposals were presented when the analysis suggested splitting requirements. The presentation was made more space-efficient, as the interface is relatively limited in space and needs to be utilized in a thoughtful way. Instead of showing each proposal in a more scattered format, the layout was compressed to allow for a quick overview without taking up unnecessary space. The goal was to make it easier for the user to compare different options side by side, while keeping the interface easy to navigate and not perceived as overloaded.

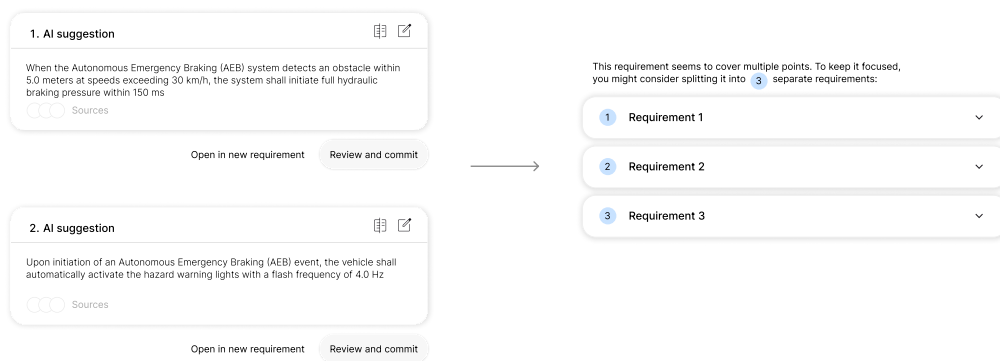


Figure 5.18: Multiple AI-suggestions refinement.

Finally, to help users verify AI-generated output, clearer and more specific links to documentation were included directly in the AI-generated responses in the form of clickable tags. This allows for quick tracing of the origin of the information and comparison with the source material. By providing easier access to sources, the design aims to support a more critical and controlled use of AI-generated information.

Some feedback related to system behavior was left unchanged as it lies outside the scope of the AI interface being designed. This includes how commits are handled in SystemWeaver, where a new item is created as confirmation of the action. Since this is part of the SystemWeaver platform rather than the AI component, it was not addressed in this iteration. The same applies to structural placement of requirements after splitting, which is handled within SystemWeaver's core functionality rather than the AI interface.

6

Results

The results of this study consist of two main deliverables. First, a set of key factors identified as important when integrating AI-assisted writing tools into requirements management systems. Second, a set of design guidelines when designing interfaces for AI-assisted writing tools in the context of requirements engineering. In addition, an interactive prototype was developed based on these guidelines, demonstrating how they can be applied in practice.

6.1 Factors

Throughout the design process, it became clear that certain factors are more important than others when integrating AI-assisted writing tools into requirements management systems. These factors were identified through a combination of literature review, interviews, and user testing. Together, they address the following research question:

RQ1: *What factors are important to achieve strong usability when integrating AI-assisted writing tools into requirements management systems?*

The identified factors are as follows:

Trust - Trust was considered important by all participants. Recurring factors for building and maintaining trust included consistent performance, transparency, and verifiability. Trust was described as something that develops over time but is quickly reduced by incorrect or inconsistent output. At the same time, participants emphasized the importance of maintaining control to avoid over-reliance on AI.

Control - Maintaining control, both in terms of system actions and over the final requirements specification was described as essential. Participants emphasized that AI-generated proposals should always be reviewed and manually approved to ensure correctness. Maintaining control was also considered important for staying engaged in the process and preserving understanding of the material.

Flexibility - Flexibility was described as important due to variations between roles, departments, and the abstraction levels at which requirements are written. This makes rigid standardization of requirements writing difficult to apply in practice. At the same time, participants expressed a desire for more structure and consistency in how requirements are written across roles and departments.

Efficiency - Efficiency was identified as the main motivation for using AI tools in requirements writing. Participants emphasized that the tool needs to be more efficient than writing requirements manually in order to be considered useful. At the same time, efficiency is dependent on the effectiveness of the AI output; if the generated suggestions require extensive correction, the perceived efficiency is reduced.

6.2 Design Guidelines

Through the design process, a final set of design guidelines was developed. These guidelines were created iteratively, with refinements made as new insights and data were collected. The guidelines are based on multiple sources of knowledge. Literature research provided a theoretical foundation for best practices in human-centered design and human-AI interaction, particularly related to transparency, trust, and human control. Insights from user interviews helped identify user needs, challenges, and expectations within requirements engineering workflows, ensuring that the guidelines were grounded in real user experiences. Furthermore, feedback from the cooperative evaluation was used to refine existing guidelines and generated new insights that informed additional ones. The resulting guidelines were developed to address the following research question:

RQ2: *What guidelines should be followed when designing interfaces for AI-assisted writing tools to support the creation of well-formed requirements?*

The guidelines were formulated in line with existing literature (see Section 3.4), they are descriptive in nature and specifically tailored to the field of AI-assisted writing tools within requirements management systems. The final set of guidelines is as follows:

Interaction

- **G1.** Be transparent about what information the AI uses and its sources. *E.g. showing references to inputs or data sources used in generating outputs.*
- **G2.** Be transparent about how AI arrives at its conclusions. *E.g. showing the reasoning or factors that led to the output.*
- **G3.** Users should control when AI generates outputs or takes actions. *E.g. deciding whether and how to use AI functionality.*
- **G4.** Make it easy to edit or ignore AI-generated output. *E.g. adjusting or dismissing output as needed.*
- **G5.** Ensure users retain final decision-making over AI-generated outputs. *E.g. saving or applying changes to the system.*
- **G6.** Provide feedback on changes made by the AI. *E.g. highlighting what has been added, removed, or modified.*
- **G7.** Make it easy to review and verify AI-generated output before use. *E.g. by checking sources or underlying information.*

Functionality

- **G8.** Offer linguistic support to ensure requirements are clearly and well-formulated. *E.g. by checking grammar, spelling, and clarity.*
- **G9.** Help ensure that requirements are formulated in a way that makes them testable. *E.g. Highlight missing constraints, dependencies, or unclear terms in a requirement description.*
- **G10.** Assist users in breaking down requirements to ensure correct scope and level of detail. *E.g. suggesting how and when to divide requirements.*
- **G11.** Support flexibility across different types of requirements and contexts. *E.g. different levels of detail or different domain needs.*

Design

- **G12.** Support active user review and confirmation of AI-generated changes before they are applied. *E.g. Use confirmation steps or deliberate friction that encourages users to reflect before accepting.*
- **G13.** Ensure the AI extension integrates seamlessly with the core systems structure and workflows. *E.g. Align AI actions with existing interaction patterns, data structures, and user expectations.*
- **G14.** Use terminology that feels natural and familiar to the users domain and context. *E.g. avoiding ambiguous or conflicting terms.*

6.3 Design Proposal

Based on the final guidelines, a design proposal of the user interface was created. The design features a side panel integrated within SystemWeaver, as illustrated in Figure 6.1. The main features of the prototype, together with explanations of how they adhere to the design guidelines, are listed below.

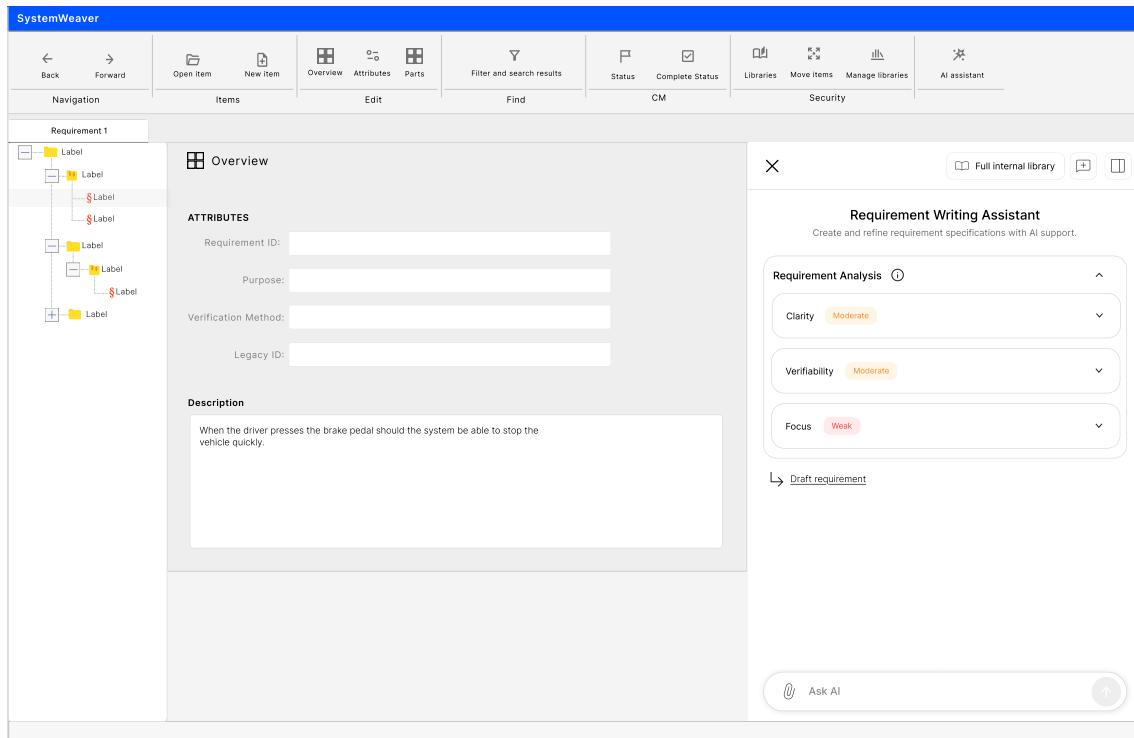


Figure 6.1: Overview of the side panel integrated into a SystemWeaver base UI prototype.

6.3.1 Requirement Analysis

Opening the AI-assisted writing tool reveals an analysis of the requirement description in SystemWeaver. The analysis section is persistently visible and can be expanded or collapsed, supporting flexible use of screen space while maintaining awareness of the feature. The analysis is structured into three categories: Clarity, Verifiability, and Focus. Clarity evaluates spelling, grammar, and overall wording quality, aligning with **G8**. Verifiability assesses whether the requirement is formulated in a way that allows it to be tested, supporting **G9**. Focus examines whether the requirement addresses a single, well-defined need, following **G10**. Each category is assigned a status tag: Strong, Moderate, or Weak, using accessible terminology that avoids unnecessary complexity and supports quick interpretation, in line with **G14**. To support understanding and transparency, an information icon and accompanying tooltip are included. This provides contextual information about what the analysis is based on and explains the meaning of the three categories, consistent with **G2**.

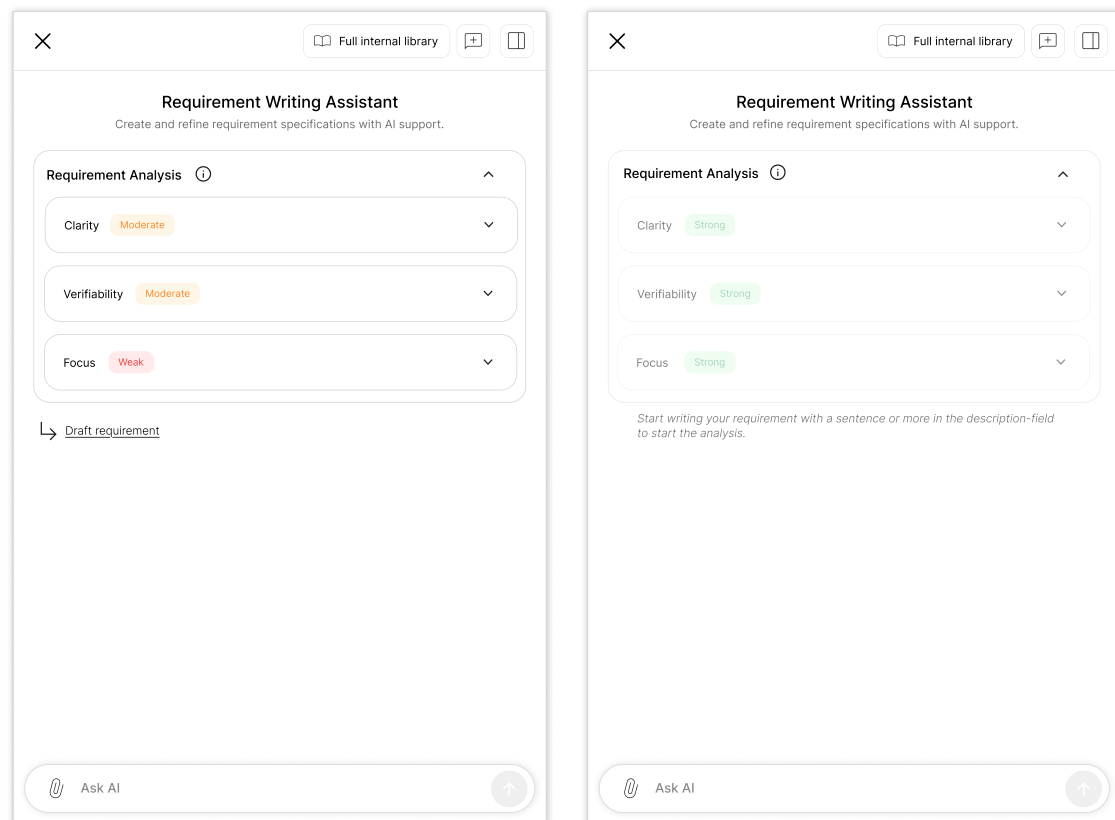


Figure 6.2: Requirement Analysis in active state with results (left) and disabled state (right). The analysis is disabled until a requirement description is initiated in the description field.

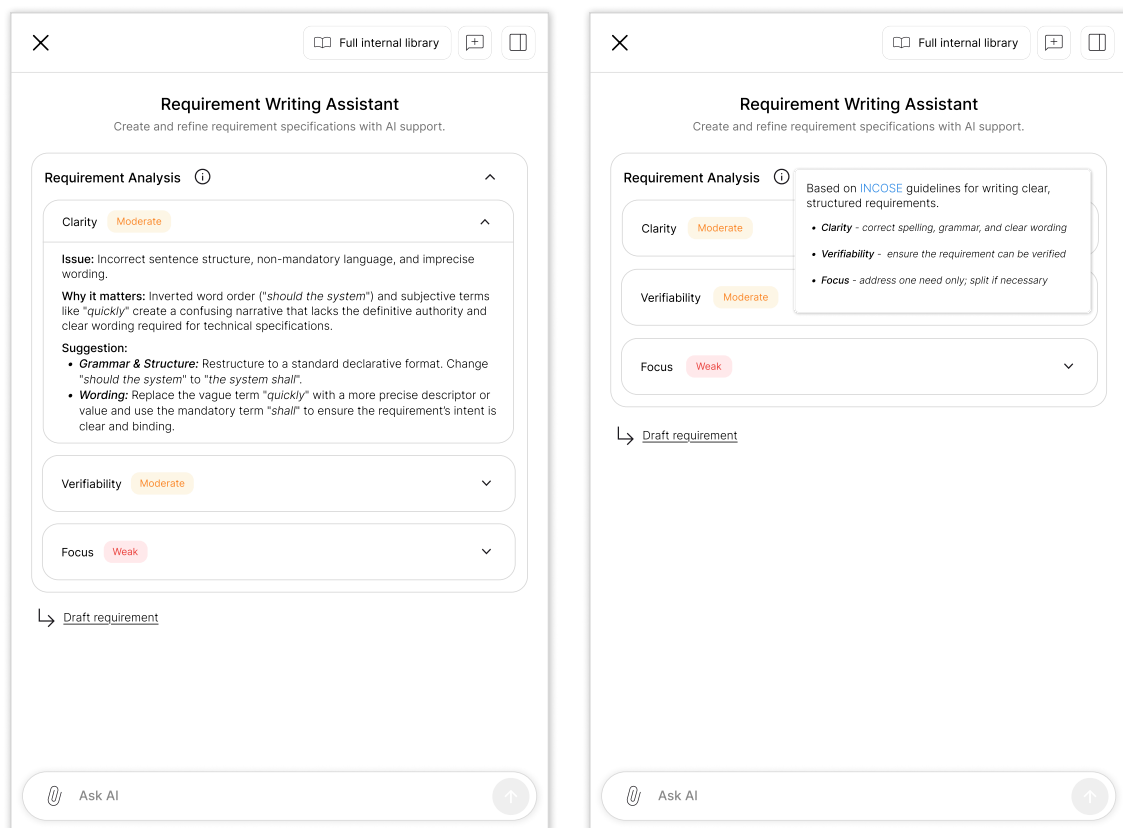


Figure 6.3: Detailed view of the Requirement Analysis, showcasing the expanded Clarity section (left) and the informational tooltip defining the analysis categories (right).

6.3.2 AI Suggestion

A central interaction element in the design is the Draft requirement shortcut, which activates the AI's improvement proposal based on the previously presented analysis. In line with **G3**, the generation is user-initiated rather than automatic, giving the user control over when a suggestion should be produced. Once a suggestion has been generated, the user can interact with it through two main functions: Edit and Compare. The edit function allows direct modification of the AI-generated suggestion, enabling the user to refine the wording or adjust the structure before accepting it back into SystemWeaver, aligning with **G4**. The comparison feature presents the original requirement alongside the AI-generated version, with changes visually highlighted, adhering to **G6**. In cases where the AI identifies that a requirement may benefit from being split, the user is prompted with a simple yes/no decision, allowing them to evaluate whether splitting it is meaningful in the given context, aligning with **G3** and **G11**.

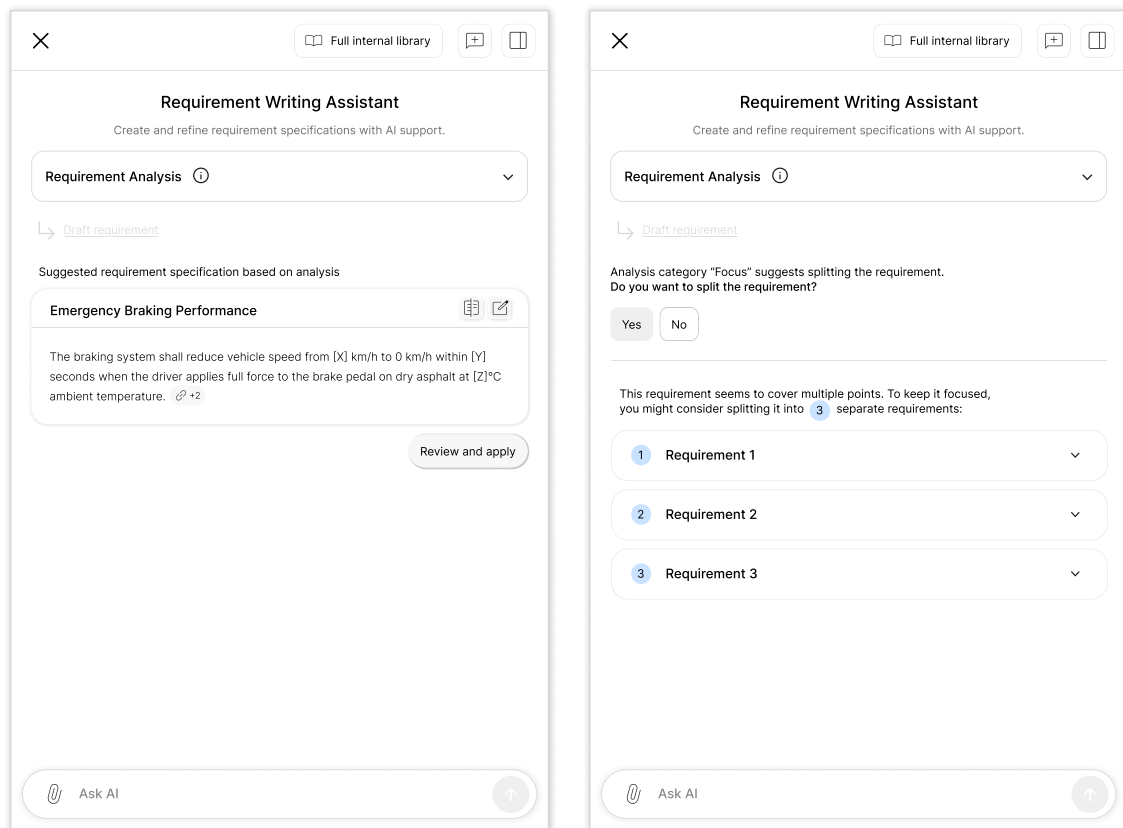


Figure 6.4: AI-generated requirement suggestions based on analysis (left) and multiple suggestions when a split is recommended (right).

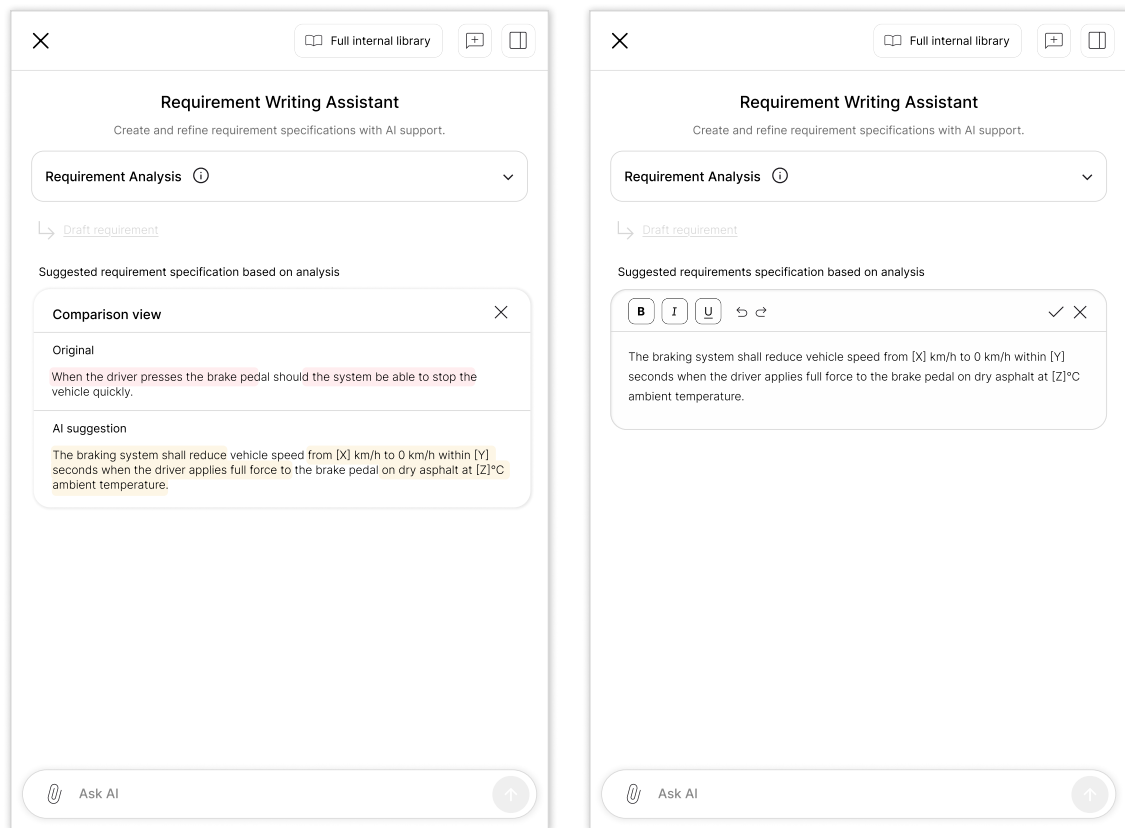


Figure 6.5: Comparison (left) and editing (right) features for refining requirement suggestions.

6.3.3 Knowledge Source & Chat

In the top bar, a Knowledge scope control allows users to define where the AI retrieves information from. The user can switch between the full internal documentation library, targeted subsets of the system, or external web search. In addition, users can upload temporary files directly through an icon in the message input bar. These files become part of the temporary knowledge context and can be queried during the current session, but are not permanently stored in the system. The top bar also includes an option to start a new conversation via a dedicated icon button, enabling users to separate different analytical tasks or requirement discussions. To support transparency (**G1** and **G7**), the AI provides direct, clickable references to the source material used in its responses. These links are embedded as tags within the output, allowing users to quickly verify where information originates and cross-check it against the original documentation.

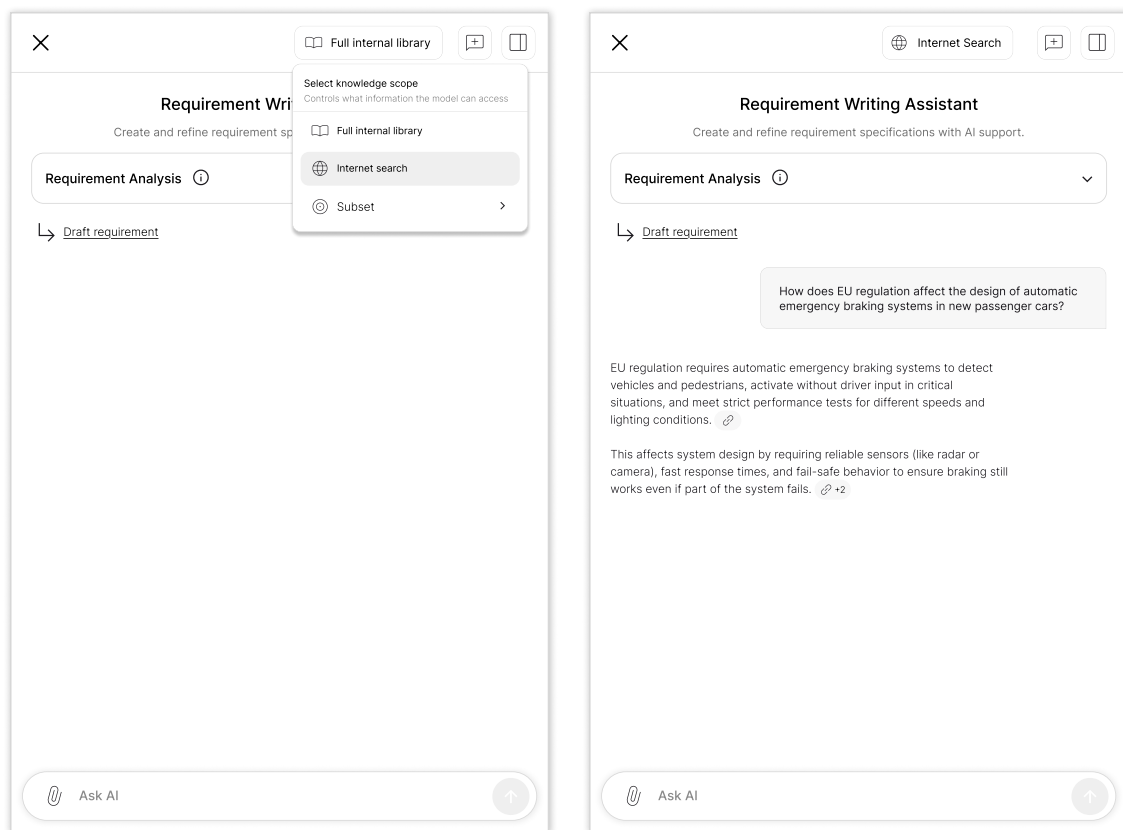


Figure 6.6: Menu for selecting knowledge source (left), and the resulting AI response using the "Internet Search" capability (right).

6.3.4 History and Versioning

The history view is accessed through an icon button to the right in the top bar and opens as a secondary side panel. It provides an overview of previous chats linked to different requirements. Chats belonging to the same requirement are grouped together and displayed in a collapsible structure. To improve navigation, the panel includes filtering options and a search bar, allowing users to quickly locate specific conversations. Basic management actions such as pinning, renaming, and deleting chats are also available.

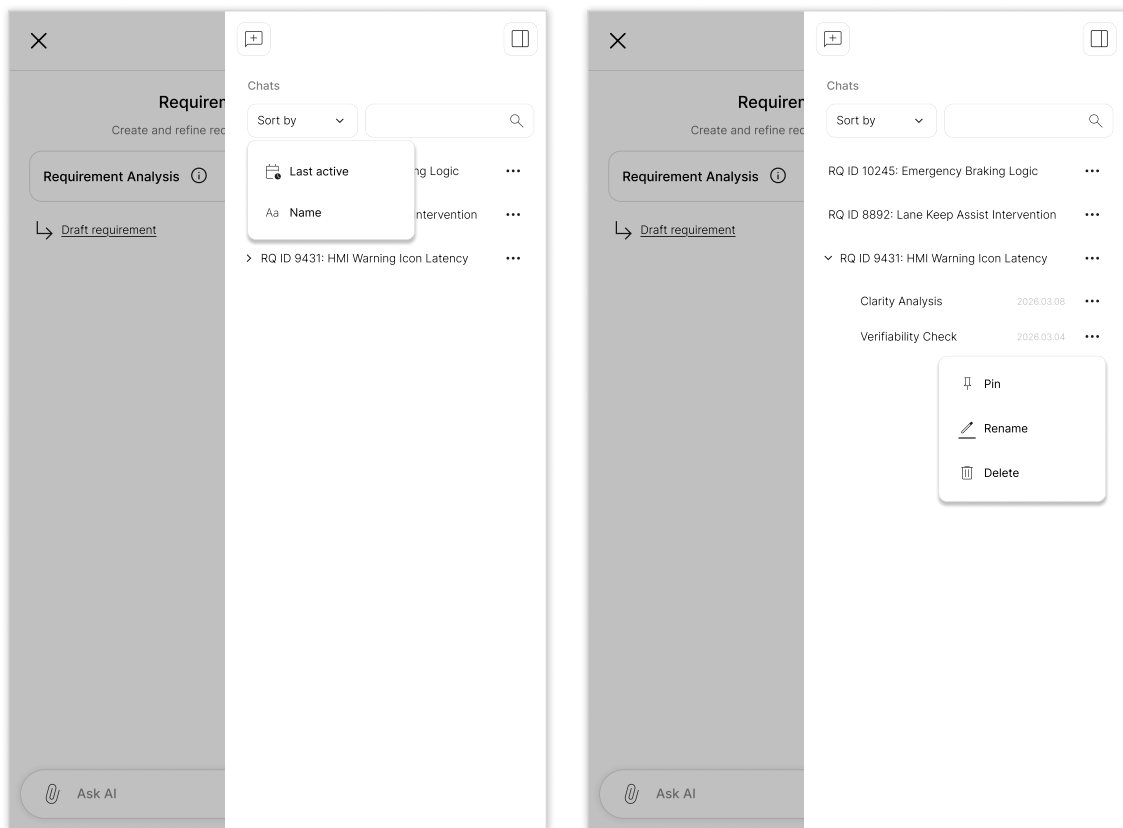


Figure 6.7: Sidebar navigation for chat history, featuring search and sorting filters (left) and the management menu for specific chat sessions (right).

6.3.5 Review and Apply

When the user is satisfied with the refined requirement, it can be saved back into SystemWeaver through the Review and apply button, which opens a dialog. The dialog presents the original requirement and the AI-generated suggestion side by side, supporting a final check before changes are committed, aligning with **G12** and **G5**. The user is given two options: save as new version or replace current version. Saving as a new version preserves the original requirement while storing the updated one separately. Replacing the current version overwrites the existing requirement with the updated text, in line with **G13**. If the AI has suggested splitting a requirement into multiple parts, each suggestion can be reviewed individually. The user can choose Create new requirement, which opens separate requirement entries in SystemWeaver, or Review and apply to save a suggestion as a new version or replace the existing requirement, as described above.

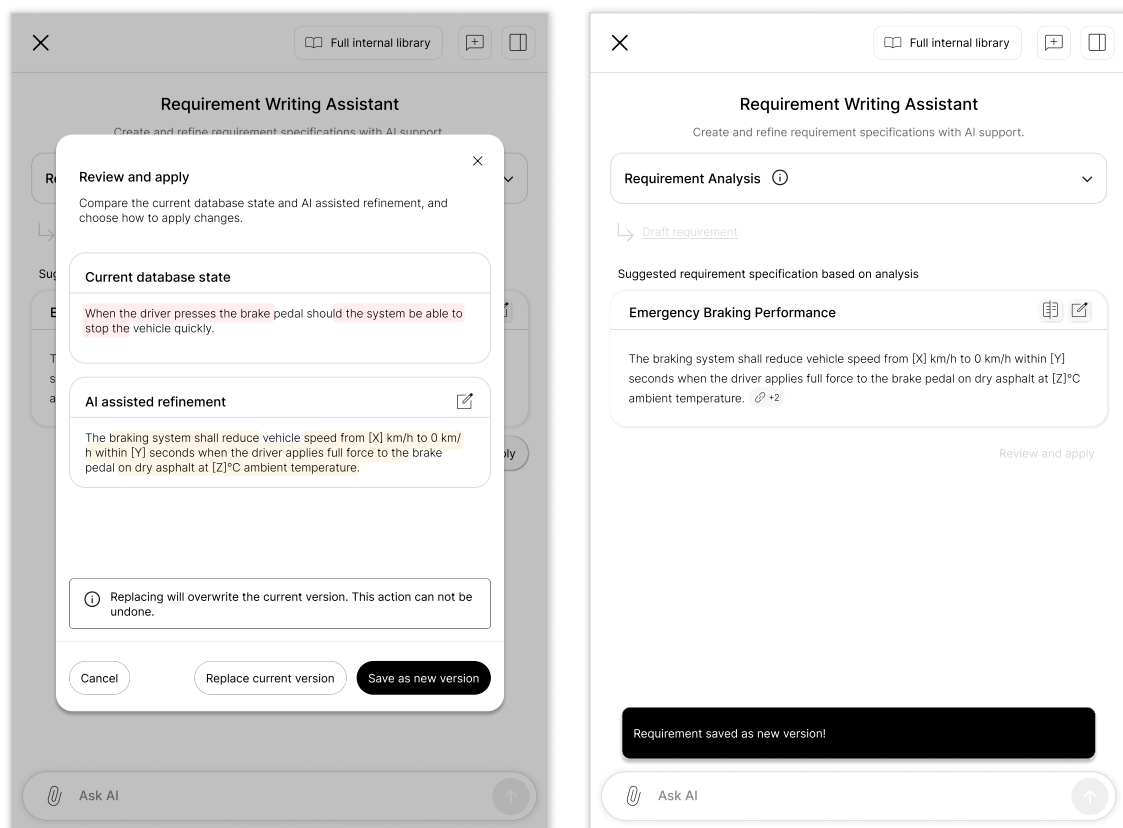


Figure 6.8: The "Review and apply" process: a side-by-side comparison dialog (left), and the confirmation snackbar displayed after successfully saving the changes (right).

7

Discussion

This chapter discusses the results of the study alongside reflections on the research process. It also addresses the scope and limitations of the study, ethical considerations, and outlines directions for future work.

7.1 Reflecting on the Results

The following sections reflect on selected aspects of the results where tensions emerge, including balancing trust and over-reliance, efficiency versus human control, flexibility versus consistency, and the dependency on AI model performance.

7.1.1 Balancing Trust and Over-reliance

The results indicate that trust in AI is closely tied to system performance, as well as transparency and the ability to verify outputs. Accurate outputs increase trust over time, while inconsistent or incorrect outputs reduce it. However, high performance introduces a different challenge: as trust increases, so does the risk of over-reliance, where users may accept AI output without sufficient critical evaluation. This concern was recurring throughout the study and is particularly important in requirements engineering, where errors can have significant downstream consequences. These findings align with previous research in human-AI interaction [32], underscoring the importance of balancing trust with critical evaluation.

7.1.2 Efficiency vs Human Control

Building on this, the results also reveal a tension between efficiency and human control. While users seek to reduce effort, they still need to remain actively involved in reviewing AI-generated output. This introduces a potential trade-off, as the need for oversight can increase cognitive load and partially offset time savings. However, this should be understood in relation to current practice. Participants described how significant time is already spent interpreting and correcting unclear requirements. From this perspective, AI support may reduce overall effort by improving clarity earlier in the process, even if it introduces an additional verification step.

7.1.3 Flexibility vs Consistency

Extending this need for balance, the results also reveal a tension between flexibility and consistency. Requirement writing is often characterized by personal variation and lack of structure, which creates a need for more consistent practices. At the same time, participants emphasized that requirements differ in type, level of detail, and context, which makes flexibility necessary. This creates a clear design challenge. While standardization can improve clarity and consistency, excessive structure risks reducing usability across diverse contexts. To address this, the design avoids rigid constraints and instead focuses on supporting general principles of requirement quality, such as improving language clarity, ensuring testability, and encouraging single-concept requirements. However, the extent to which these design choices improve requirement quality remains to be seen.

7.1.4 Dependency on AI model Performance

One of the biggest challenges during the project was the difficulty of isolating the user interface design from the performance of the underlying LLM. It became clear in both interviews and the evaluation that the user experience is heavily connected to the quality of the underlying AI model and the output it provides. This presented a challenge when creating our design guidelines, as the user experience is influenced by several different factors at once. Essentially, even a well-designed interface cannot fully compensate for the fact that an AI sometimes gives inconsistent, unexpected, or incorrect results.

7.2 Reflecting on Methodologies and Process

This section reflects on the practical design process carried out during the study and evaluates how it may have influenced the project's outcome. Furthermore, it explores the role of AI as a collaborative tool within the design process, reflecting on how such technologies are shifting the role of designers and the way interactive systems are prototyped and evaluated.

7.2.1 Thematic analysis

A significant part of our methodology was the use of thematic analysis to interpret the qualitative data gathered from our interviews and user tests. During the project, we shifted between different coding approaches, which provided different experiences and challenges. In our first thematic analysis, of the interview data, we used an inductive approach, meaning that codes were generated directly from the interview data rather than from predefined categories. While this was helpful for discovering new insights, it also resulted in a large and overwhelming amount of codes. We found that having too many scattered codes made the analysis difficult to manage, as data with essentially the same underlying meaning were sometimes assigned unique codes. This lack of structure led to an overwhelming amount of information that was hard to synthesize into clear themes.

In the second thematic analysis, focusing on the user test data, we instead adopted a more deductive approach by using our evaluation guide and specific scenarios to organize the findings. This provided a structured foundation that allowed us to more effectively identify inductive themes as they emerged from the data. This proved to be much more efficient, and it allowed us to focus on how the participants actually interacted with and understood the prototype rather than becoming lost in a high volume of individual codes.

Reflecting on this, we found it beneficial to start with a more structured or deductive approach when handling large amounts of qualitative data. While an inductive start is valuable for exploring a new problem area, a deductive framework helps prevent the data from becoming too overwhelming. This shift in strategy was essential for us to distill our findings into clear, actionable themes in an effective way.

7.2.2 User testing

The user tests and cooperative evaluation of the prototype involved four participants, which is a relatively small sample size considering that the system is intended for more than 100 potential users. The low number of participants was primarily due to the limited time available for the evaluation phase, which made it difficult to find and recruit available employees from the organization within this time span. This is a significant limitation, as it may have affected the range of feedback we received on the prototype.

With more participants, we could have reached more nuanced feedback and a better understanding of certain needs, challenges, or preferences that are unique to specific departments or roles within the company. On the other hand, in highly specialized fields like requirements engineering, feedback from a few expert users can be more valuable than a large number of tests with non-experts. The participants in our study had extensive experience working in SystemWeaver, which meant they could provide deep professional insights and understanding of the current workflow. Even with this small group of test users, we saw consistent patterns in the feedback, such as confusion over unclear terminology.

Nevertheless, we realize that a larger number of tests would have provided more nuanced feedback. More testing would have allowed us to explore how different engineers interact with the AI tool in their individual workflows. Because of this limitation, our final design should be seen as a concept that identifies core needs rather than a fully validated product.

7.2.3 Prototyping

The use of high-fidelity prototyping was a successful method for quickly visualizing and evaluating the interface during the project. However, reflecting on the process, more iterations would have been necessary to present and validate a final prototype. Additional cycles of testing and refinement would have allowed us to further validate that the design effectively fulfills user needs and preferences, while also confirming the effectiveness of the guidelines. Furthermore, to achieve a more realistic and

final user experience, the tool would require a higher level of implementation and integration within SystemWeaver. Since the AI assistant and the core requirement management platform are intended to be closely linked, it was difficult for users to fully understand the consequences of certain actions without seeing them reflected in the live system. While this level of technical integration was outside the scope of our project and timeline, we recognize that it is a critical factor for the overall user experience and for the user to fully understand the interface and interactions. Without a seamless connection between the AI and the existing database, the prototype remains a proof-of-concept rather than a final functional tool.

7.2.4 AI in the Design Process

The use of AI-powered tools like Lovable in this project highlights a shift in how designers can approach prototyping and development. Traditionally, creating a functional prototype with real-time API integration would require deep technical knowledge and a significant amount of time. However, by leveraging AI, we were able to transform a static design in Figma to a fully interactive prototype much faster and with little technical or development knowledge. This suggests that AI acts as an "accelerator", allowing designers to focus more on user interaction, interface design, and system behavior rather than getting stuck in manual coding.

Another important reflection is how the role of designers changes when using AI. Instead of manually building every component, its different interaction states and transitions, the process becomes more about steering the AI through precise prompting and visual references. For example, ensuring that the prototype precisely matched our Figma designs required us to provide the AI with detailed instructions on color codes and layout to achieve the final desired look, even when we provided it with screenshots of the designs. This shows that AI can handle the execution, but the designer's eye for detail and understanding of user needs, design details, and usability are still essential to ensure a high-quality result.

Finally, using a functional AI prototype during user testing provided a level of realism that is hard to achieve with standard tools without requiring extensive coding. Because participants could interact with a real AI powered chat, their feedback on trust and efficiency was based on actual system behavior rather than simulated "fake" experiences. This more realistic experience is key in human-AI interaction research, as it reveals the true nuances of how users react to AI suggestions and errors. Taken together, using AI in the design process is not just about efficiency and saving time, it also enables designers to create and test more complex, realistic systems with less effort, leading to better user insights.

7.3 Ethical Concerns

The integration of AI-assisted writing into a professional engineering environment introduces several ethical challenges that must be addressed to ensure responsible implementation.

7.3.1 Job Displacement and Professional Pride

A central ethical concern is the tension between AI as a supporting tool and the risk of it replacing jobs or leading to skill degradation. If LLMs become highly proficient at formulating requirements and acquiring the contextual knowledge in-house engineers currently hold, a threat to engineers' own technical expertise arises. As participants described, the act of writing requirements is a cognitive process used to build a "mental picture" and a deep understanding of the product development and requirement structure. If this creative step is bypassed by delegating the formulation to an AI, it may lead to a state of cognitive passivity. Over time, this makes it increasingly difficult for engineers to critically engage with the technical context, creating a cycle of over-reliance where the user lacks the necessary context to even spot AI errors. This potential loss of deep domain expertise not only threatens job security but also undermines professional pride, as the engineer's role shifts from a creative expert to a passive monitor of automated output. If the sense of ownership and mastery over the technical content diminishes, the profession as a whole may lose value.

To address these concerns and protect professional pride, we have been certain about adopting a Human-Centered AI (HCAI) approach when designing the AI interface. As mentioned in section 3.2.2, Shneiderman [33] promotes that AI should not be defined as an autonomous agent but as a tool designed to enhance human control and mastery. We have strived to reflect this in our design proposal by preventing the AI from becoming an independent actor, for example, by requiring the user to actively initiate suggestions (G3) and by using intuitive mechanisms for review and manual correction (G4, G7) to support user agency. By prioritizing human judgment in the design, we aim to maintain the engineer's role as an active expert.

7.3.2 Accountability

Ensuring that the engineer remains an active user is not only essential for upholding professional integrity and justifying the need for human expertise but also serves as a foundation for accountability. The principle of accountability in AI systems raises critical questions about who is responsible when a system produces an error with severe consequences [40]. In the context of requirements engineering, a single misinterpreted constraint can lead to significant production costs and safety risks, raising serious concerns when AI is integrated and entrusted with the formulation of requirements.

To address this, our guidelines and proposed design strive to ensure that the human remains in the loop throughout the interaction, primarily by establishing the user as the sole actor capable of final actions. In accordance with G5, the AI should be strictly incapable of making autonomous entries or modifications directly into the database. By ensuring that the final action always belongs to the user, the interface reinforces the principle that the human, not the system or AI, owns the requirement. This design choice is intended to provide the user with a clear sense of agency and accountability for the data being implemented. Furthermore, the interface introduces friction points, as stated in G12, to counteract the risk of passive

implementation of changes. These points of reflection, such as the mandatory review step before applying any change into the database, aim to keep the engineer engaged and encourage active evaluation of the AI's suggestions. By intentionally slowing down the final step, the design forces a moment of verification, making it clear that when a user clicks "Replace current version" or "Save as new version", they are explicitly accepting responsibility for the AI-generated content.

7.4 Future Work

While this study provides a foundational framework for AI-assisted requirements engineering, the transition from a proof-of-concept to a production-ready tool involves several paths for further exploration.

A primary focus for future work should be more extensive user testing to validate the guidelines and the design proposal. In this project, cooperative evaluation was primarily used as a rapid iterative method, and further iterations are required to achieve a more rigorous validation. To enable such testing in a more realistic environment, a critical next step is to integrate the prototype of the AI tool into the core SystemWeaver platform. Moving beyond a standalone prototype to an integrated high-fidelity version within the core system would allow evaluation of the technical interplay and ensure that the collaboration feels seamless to the user. This is essential for creating a full understanding of how the systems influence each other and to ensure that actions performed within the AI extension affect the core system in a way that aligns with user expectations. Validating this technical interplay is necessary to guaranty that the integrated environment functions as a unified system rather than as separate components.

Another relevant direction for future work is to explore accessibility aspects of AI-supported writing tools in requirements engineering. While not a primary focus of this project, accessibility considerations are important to ensure that AI assistance does not unintentionally exclude or disadvantage certain user groups, thereby strengthening the applicability of the solution in broader organizational contexts.

Future work could also focus on extending the AI support to cover additional parts of the requirements engineering process. Throughout the project, users expressed a need for broader support beyond writing requirements, including tasks such as generating test cases and assisting with documentation. In practice, these activities are closely interconnected, where requirements, test cases, and documentation influence and depend on each other. Expanding the scope in this way could therefore contribute to a more coherent and useful tool.

7.5 Rethinking AI as the Primary Solution

A question that arose during this project is whether AI is the most appropriate solution to improve requirements quality, or whether similar results can be achieved through alternative methods.

There appears to be a tendency to overuse AI as a general solution to quality-related problems, where technical tools are expected to address challenges that are often organizational or knowledge-related in nature.

The interviews indicated that issues in requirements quality are largely related to a lack of shared practices for how requirements are written, suggesting that these challenges could also be addressed through internal training or organizational standards rather than solely through AI-based solutions. At the same time, AI provides value by offering real-time support in the workflow, which traditional training cannot replace in the moment of writing.

Whether AI is the best solution for improving requirements quality, or whether a combination of AI support and traditional training would be more appropriate, is beyond the scope of this study. However, it remains an open question that may be worth exploring further.

7.6 Use of AI

Beyond the use of AI described in 5.3.5 Functional Prototype and discussed in 7.2.4 AI in the Design Process, AI tools have also been used to support the writing of this report. Tools such as Gemini and ChatGPT were used to improved the linguistic quality, like assisting spelling, grammar, and sentence structure, to ensure that the concepts were communicated with greater clarity and precision. Furthermore, AI assisted in maintaining a consistent professional tone and optimizing the overall flow of the content.

While AI tools offer significant advantages in terms of linguistic refinement, their use is not without risks. A primary concern is the potential that the original meaning is subtly altered. Because AI models prioritize fluency and stylistic flow, they may occasionally rephrase technical descriptions in a way that subtly changes the original intent or compromises scientific accuracy.

Another risk involves the loss of the author's unique voice or the introduction of hallucinations, where the AI might suggest factual additions that are not supported by the actual research data.

To mitigate these risks, a rigorous human-in-the-loop approach was maintained. It is important to note that all ideas, arguments, and conclusions remain the author's own, and that all AI-generated suggestions were critically reviewed and edited by the authors to ensure accuracy, academic integrity, and the preservation of the original intent.

8

Conclusion

This thesis, conducted in collaboration with Geely Technology Europe, explored the intersection of AI-assisted writing and usability in the context of requirements engineering. It was guided by the following research questions:

RQ1: *What factors are important for achieving strong usability when integrating AI-assisted writing tools into requirements management systems?*

RQ2: *What guidelines should be followed when designing interfaces for AI-assisted writing tools to support the creation of well-formed requirements?*

The study followed a Research through Design approach, structured using the Double Diamond model. The project included a literature review, semi-structured interviews, thematic analysis, ideation, prototyping, and usability testing. This resulted in four key factors important for usability when integrating AI-assisted writing tools into requirements management systems: Trust, Control, Flexibility, and Efficiency (see Section 6.1). Based on these findings, fourteen design guidelines across interaction, functionality, and design were developed, as outlined in Section 6.2. In addition, a design proposal based on these guidelines was created (see Section 6.3).

These findings contribute both practically and academically. From a practical perspective, the results provide guidance for the integration and design of AI-assisted writing tools in requirements management systems. From an academic perspective, the study contributes to the field of interaction design by extending research on AI-assisted writing to industrial contexts, where the need for precision and consistency is particularly high. Future work could build on these findings by further evaluating the proposed design guidelines with a broader group of users and by exploring how AI support can be extended to additional parts of the requirements engineering workflow.

In conclusion, this thesis provides insights into how AI-assisted writing tools can be designed and integrated to better support requirements engineering in practice.

Bibliography

- [1] M. Raza, Z. Jahangir, M. B. Riaz, M. J. Saeed, and M. A. Sattar, “Industrial applications of large language models,” *Scientific Reports*, vol. 15, no. 1, p. 13755, 2025.
- [2] K. Pohl, *Requirements engineering: An overview*. RWTH, Fachgruppe Informatik Aachen, 1996.
- [3] Q. K. Shams-ul-Arif and S. Gahyyur, “Requirements engineering processes, tools/technologies, & methodologies,” *International Journal of reviews in computing*, vol. 2, no. 6, pp. 41–56, 2009.
- [4] S. Lim, A. Henriksson, and J. Zdravkovic, “Data-driven requirements elicitation: A systematic literature review,” *SN Computer Science*, vol. 2, no. 1, p. 16, 2021.
- [5] L. Montgomery, D. Fucci, A. Bouraffa, L. Scholz, and W. Maalej, “Empirical research on requirements quality: A systematic mapping study,” *Requirements Engineering*, vol. 27, no. 2, pp. 183–209, 2022.
- [6] D. Helgesson, E. Engström, P. Runeson, and E. Bjarnason, “Cognitive load drivers in large scale software development,” in *2019 IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, IEEE, 2019, pp. 91–94.
- [7] M. A. Zadenoori, J. Dbrowski, W. Alhoshan, L. Zhao, and A. Ferrari, “Large language models (llms) for requirements engineering (re): A systematic literature review,” *arXiv preprint arXiv:2509.11446*, 2025.
- [8] J. Wang, W. Ma, P. Sun, M. Zhang, and J.-Y. Nie, “Understanding user experience in large language model interactions,” *arXiv preprint arXiv:2401.08329*, 2024.
- [9] Geely Technology Europe, *Geely technology europe - innovation and product development*, <https://geelytech.eu/en>, Accessed: 2026-04-21, 2026.
- [10] Systemite AB, *Systemweaver - software plm for complex product development*, Accessed: 2026-01-28, 2025. [Online]. Available: <https://systemweaver.com/>.
- [11] E. Wolfgang, *Introduction to artificial intelligence*. Springer, 2017.
- [12] D. Khurana, A. Koli, K. Khatter, and S. Singh, “Natural language processing: State of the art, current trends and challenges,” *Multimedia tools and applications*, vol. 82, no. 3, pp. 3713–3744, 2023.
- [13] N. Karanikolas, E. Manga, N. Samaridi, E. Tousidou, and M. Vassilakopoulos, “Large language models versus natural language understanding and generation,” in *Proceedings of the 27th Pan-Hellenic Conference on Progress in Computing and Informatics*, 2023, pp. 278–290.

- [14] S. Feuerriegel, J. Hartmann, C. Janiesch, and P. Zschech, “Generative AI,” *Business & Information Systems Engineering*, vol. 66, no. 1, pp. 111–126, 2024. DOI: 10.1007/s12599-023-00834-7.
- [15] J. Rezwana and M. L. Maher, “Designing creative ai partners with cofi: A framework for modeling interaction in human-ai co-creative systems,” *ACM Trans. Comput.-Hum. Interact.*, vol. 30, no. 5, Sep. 2023, ISSN: 1073-0516. DOI: 10.1145/3519026. [Online]. Available: <https://doi.org/10.1145/3519026>.
- [16] OpenAI, *Introducing canvas*, Accessed: 2026-05-07, 2024. [Online]. Available: <https://openai.com/sv-SE/index/introducing-canvas/>.
- [17] OpenAI, *Chatgpt overview*, Accessed: 2026-05-07, 2024. [Online]. Available: <https://chatgpt.com/sv-SE/overview/>.
- [18] Google, *About gemini*, Accessed: 2026-05-07, 2024. [Online]. Available: <https://gemini.google/about/>.
- [19] Google, *Gemini canvas overview*, Accessed: 2026-05-07, 2024. [Online]. Available: <https://gemini.google/overview/canvas/>.
- [20] Anthropic, *Claude product overview*, Accessed: 2026-05-07, 2024. [Online]. Available: <https://www.claude.com/product/overview>.
- [21] D. Pandey, U. Suman, and A. Ramani, “An effective requirement engineering process model for software development and requirements management,” in *2010 International Conference on Advances in Recent Technologies in Communication and Computing*, 2010, pp. 287–291. DOI: 10.1109/ARTCom.2010.24.
- [22] D. M. Fernández et al., “Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice,” *Empirical Software Engineering*, vol. 22, no. 5, pp. 2298–2338, 2017. DOI: 10.1007/s10664-016-9451-7.
- [23] M. Shahid, S. Ibrahim, and M. N. Mahrin, “An evaluation of requirements management and traceability tools,” *World Academy of Science, Engineering and Technology, WASET*, vol. 1, no. 1, pp. 1–6, 2011.
- [24] SystemWeaver, *Systemweaver software plm for complex product development*, <https://systemweaver.com/>, Besökt 2026-01-28, 2025.
- [25] M. A. Abbasi, P. Ihantola, T. Mikkonen, and N. Mäkitalo, “Towards human-ai synergy in requirements engineering: A framework and preliminary study,” in *2025 Sixth International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, IEEE, 2025, pp. 81–88.
- [26] INCOSE, *Guide to writing requirements*, Version 4, INCOSE-TP-2010-006-04, International Council on Systems Engineering, San Diego, CA, 2023. [Online]. Available: <https://www.incose.org>.
- [27] M. Lou, J. Crowley, S. Dodson, and D. Yoon, “Anchoredai: Contextual anchoring of ai comments improves writer agency and ownership,” *arXiv preprint arXiv:2509.16128*, 2025.
- [28] M. Reza et al., “Abscribe: Rapid exploration & organization of multiple writing variations in human-ai co-writing tasks using large language models,” in *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, 2024, pp. 1–18.

- [29] F. Lehmann and D. Buschek, “Functional flexibility in generative ai interfaces: Text editing with llms through conversations, too lbars, and prompts,” *arXiv preprint arXiv:2410.10644*, 2024.
- [30] H. Sharp, J. Preece, and Y. Rogers, *Interaction Design: Beyond Human-Computer Interaction*, 5th ed. John Wiley & Sons, Incorporated, 2019.
- [31] G. Sinha, R. Shahi, and M. Shankar, “Human computer interaction,” in *2010 3rd International Conference on Emerging Trends in Engineering and Technology*, IEEE, 2010, pp. 1–4.
- [32] S. Amershi et al., “Guidelines for human-ai interaction,” in *Proceedings of the 2019 chi conference on human factors in computing systems*, 2019, pp. 1–13.
- [33] B. Shneiderman, *Human-centered AI*. Oxford University Press, 2022.
- [34] J. Auernhammer, “Human-centered ai: The role of human-centered design research in the development of ai,” 2020.
- [35] M. Hassenzahl, “User experience and experience design,” *The encyclopedia of human-computer interaction*, vol. 2, pp. 1–14, 2013.
- [36] *Iso-9241-11:1998(en), ergonomic requirements for office work with visual display terminals (vdt) part-11: Guidance on usability*. [Online]. Available: <https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-1:v1:en:sec:D>.
- [37] P. W. Jordan, *An introduction to usability*. Crc Press, 2020.
- [38] L. Moreno, P. Martínez, and B. Ruiz-Mezcua, “A bridge to web accessibility from the usability heuristics,” in *Symposium of the Austrian HCI and Usability Engineering Group*, Springer, 2009, pp. 290–300.
- [39] K. K. Fu, M. C. Yang, and K. L. Wood, “Design principles: The foundation of design,” in *International design engineering technical conferences and computers and information in engineering conference*, American Society of Mechanical Engineers, vol. 57175, 2015, V007T06A034.
- [40] D. Leslie, “Understanding artificial intelligence ethics and safety,” *arXiv preprint arXiv:1906.05684*, 2019.
- [41] J. Morrison, C. Na, J. Fernandez, T. Dettmers, E. Strubell, and J. Dodge, “Holistically evaluating the environmental impact of creating language models,” *arXiv preprint arXiv:2503.05804*, 2025.
- [42] N. Jegham, M. Abdelatti, C. Y. Koh, L. Elmoubarki, and A. Hendawi, “How hungry is ai? benchmarking energy, water, and carbon footprint of llm inference,” *arXiv preprint arXiv:2505.09598*, 2025.
- [43] G. Su, “Unemployment in the ai age,” *AI Matters*, vol. 3, no. 4, pp. 35–43, Feb. 2018. DOI: 10.1145/3175502.3175511. [Online]. Available: <https://doi.org/10.1145/3175502.3175511>.
- [44] C. Frayling, “Research in art and design (royal college of art research papers, vol 1, no 1, 1993/4),” 1994.
- [45] Design Council, *The double diamond: A universally accepted depiction of the design process*, Hämtad: 2024-05-22, 2024. [Online]. Available: <https://www.designcouncil.org.uk/our-resources/the-double-diamond/>.
- [46] H. Snyder, “Literature review as a research methodology: An overview and guidelines,” *Journal of business research*, vol. 104, pp. 333–339, 2019.
- [47] J. Preece, Y. Rogers, H. Sharp, D. Benyon, K. Holland, and T. Carey, *Interaction Design: Beyond Human-Computer Interaction*, 5th. Wiley, 2019.

- [48] B. Hanington and B. Martin, *Universal Methods of Design, Expanded and Revised*. Beverly, MA: Quarto Publishing Group USA, 2019.
- [49] V. Braun and V. Clarke, “Using thematic analysis in psychology,” *Qualitative research in psychology*, vol. 3, no. 2, pp. 77–101, 2006.
- [50] K. Österlin, *Design i fokus: Varför ser saker ut som de gör?* Swedish, 3:2. Malmö: Liber AB, 2011, ISBN: 9789147094431.
- [51] Google Design Sprint Kit, *Crazy 8’s*, Google, Hämtad: 2026-03-30, u.å. [Online]. Available: <https://designsprintkit.withgoogle.com/methodology/phase3-sketch/crazy-8s>.
- [52] Interaction Design Foundation, *What are Wireframes?* Accessed: 2026-01-27, Sep. 2016. [Online]. Available: <https://www.interaction-design.org/literature/topics/wireframe>.
- [53] Interaction Design Foundation, *What are Prototypes?* Accessed: 2026-01-27, Oct. 2019. [Online]. Available: <https://www.interaction-design.org/literature/topics/prototypes>.
- [54] S. Gibbons, *Design critiques: Encourage a positive culture to improve products*, Nielsen Norman Group, Hämtad: 2026-03-30, Oct. 2016. [Online]. Available: <https://www.nngroup.com/articles/design-critiques/>.
- [55] S. Riihiahho, “Experiences with usability testing: Effects of thinking aloud and moderator presence,” 2015.
- [56] M. Nii Laryeafio and O. C. Ogbewe, “Ethical consideration dilemma: Systematic review of ethics in qualitative data collection through interviews,” *Journal of Ethics in Entrepreneurship and Technology*, vol. 3, no. 2, pp. 94–110, 2023.

A

Semi-structured Interview Guide

General

- Can you describe your professional role?

Use of SystemWeaver

- How often do you use SystemWeaver in your daily work? For which tasks?
- How experienced would you say you are in using SystemWeaver? Please rate on a scale from 1 to 4, where 1 = beginner and 4 = very experienced.
- What do you think about SystemWeaver?
- Is there anything in its use that does not work well? For example, tasks that are time-consuming, difficult, or frustrating?

Requirements formulation in SystemWeaver

- How often do you work with reading and/or writing requirements in SystemWeaver?
- Can you describe your workflow when writing requirements?
- Do you follow any frameworks or guidelines when writing requirements?
- Do you experience any challenges when writing requirements? How do you usually handle such challenges?
- Do you experience any challenges when reading requirements? How do you usually handle such challenges?
- Are there parts of the requirements writing process where additional support or guidelines would be helpful?

Attitude toward AI

- Are you using AI? How?
- Do you think AI could support the work of writing requirements? In what way?
- What level of AI support would you prefer when writing requirements?
- Would you trust that support? What factors would affect your trust?

B

Cooperative Evaluation Protocol

Scenario 1

You are tasked with writing a new requirement. As a starting point, you are given the following initial description: "The car should stop quickly when the driver presses the brake pedal."

1. Open the AI tool
 - (a) What do you see here?
 - (b) What do you think you can do here?
2. Open and read each analysis category
 - (a) What do you think about the information in the analysis?
 - (b) Is the information useful?
 - (c) Is any information missing?
3. Change something in the AI suggestion
 - (a) Is it clear how to change the suggestion?
 - (b) Would you use this feature?
4. Publish the AI suggestion to the database
 - (a) What do you think about this feature?
 - (b) Was anything difficult or confusing?
 - (c) Would this have simplified your work?

Scenario 2

You are tasked with writing a new requirement. As a starting point, you are given the following initial description: "The car should automatically brake at maximum and turn on the hazard lights when an obstacle is detected."

1. Review the "Focus" analysis category
 - (a) What information do you get here?

- (b) Is anything difficult to understand?
 - (c) Is any information missing?
- 2. Open one of the suggestions as a new requirement
 - (a) What do you think about this feature?
 - (b) Is anything unclear?
 - (c) Would you use this feature?
- 3. Publish the second suggestion to the database
 - (a) Is any functionality missing to handle the split?
 - (b) Is anything unnecessary?

Scenario 3

You are tasked with writing a new requirement for an automatic emergency braking system in a car.

- 1. Change the source to: EU Vehicle Safety Regulations.
 - (a) What do you think about this feature?
 - (b) Is it clear how to change the source?
- 2. Ask the AI about relevant laws
 - (a) Is it clear where the information comes from?
 - (b) Would you trust this information?
 - (c) Would you use this feature?

General Questions

- What was the best part of the prototype?
- What was the worst part of the prototype?
- What do you think needs the most improvement?
- Is there anything else you would like to add or comment on?