

FACHHOCHSCHULE NORDWESTSCHWEIZ

What's That Error?

let's make the error messages actionable

Bachelor-Thesis IP6

Namen:

Lorin Desch, iCompetence, 8. Semester
Antonio Salvia, Informatik, 8. Semester

Betreuer:

Prof. Dr. Norbert Seyff
Dr. Nitish Patkar

Experte:

Claudia Wipf

Auftraggeber:

Institut für Interaktive Technologien FHNW

Projektdauer:

19. Februar 2024 bis 16. August 2024

16. August 2024

Abstract

In der heutigen Zeit, in der technologische Systeme immer komplexer werden, wird die klare und umsetzbare Darstellung von Fehlermeldungen zu einer zentralen Herausforderung. Diese Schwierigkeit betrifft nicht nur Experten, sondern insbesondere Laien, die oft Schwierigkeiten haben, technische Meldungen zu verstehen. Diese Arbeit zielt darauf ab, durch die Integration aktueller Designprinzipien und den Einsatz künstlicher Intelligenz die Benutzerfreundlichkeit und Handlungsfähigkeit von Fehlermeldungen zu verbessern.

Im Rahmen dieser Arbeit wurde ein Proof of Concept entwickelt, das die identifizierten Schwachstellen bestehender Systeme adressiert. Der Fokus liegt dabei auf der Gestaltung einer intuitiven Benutzeroberfläche, die sowohl für Experten als auch für Laien verständlich ist. Zusätzlich wurden KI-Funktionen implementiert, die es ermöglichen, technische Jargon zu entschlüsseln und dem Benutzer klare, umsetzbare Handlungsempfehlungen zu geben. Der Fokus lag zudem auf der Entwicklung und Implementierung eines Feedback-Mechanismus, durch den die Issues und die KI anhand der Benutzerinteraktionen kontinuierlich optimiert werden.

Die durchgeführten Tests und Evaluationen zeigen, dass die vorgeschlagenen Lösungen nicht nur die Benutzerfreundlichkeit steigert, sondern auch die Effizienz bei der Fehlerbehebung erhöht. Die Ergebnisse dieser Arbeit tragen zur Weiterentwicklung benutzerzentrierter Designansätze im technischen Umfeld bei und bieten Erkenntnisse für zukünftige Entwicklungen im Bereich der automatisierten Fehlerbehebung und Feedbacksysteme.

Diese Arbeit leistet somit einen Beitrag zur Diskussion über die Optimierung von Systemüberwachungssoftware und zeigt das Potenzial für die Integration von KI in die Benutzerinteraktion auf. Diese Arbeit setzt einen Impuls, Laien verstärkt in den Mittelpunkt zu stellen und ihnen durch benutzerfreundliche Lösungen sowie klare Handlungsempfehlungen eine effektivere Unterstützung bei der Problemlösung zu bieten.

Inhaltsverzeichnis

1	Einleitung	5
1.1	Hintergrund und Kontext des Projektes	5
1.2	Problemstellung	5
1.3	Zielsetzung	5
1.4	Forschungsfragen	5
1.5	Struktur des Berichts	6
2	Hintergrund Issue	7
2.1	Ziele und Funktionen von Issues	7
2.2	Kommunikationsmodell	7
3	State of the Art Analyse	9
3.1	Einführung	9
3.2	Perspektiven aus verschiedenen Domänen	9
3.3	Methodik der Literaturrecherche	10
3.4	Merkmale von Issues	11
3.4.1	Design	11
3.4.2	Auffälligkeit	11
3.4.3	Icons und Piktogramme	11
3.4.4	Farben	12
3.4.5	Layout	12
3.4.6	Inhalt	12
3.4.7	Textlänge	13
3.4.8	Informationsüberflutung	13
3.4.9	Wirksamkeit	13
3.4.10	Actionability	13
3.5	Zusammenfassung der Best Practices aus der Literaturrecherche	14
4	Marktanalyse	16
4.1	Auswahl der analysierten Systeme	16
4.2	Dynatrace	16
4.2.1	Design	17
4.2.2	Verständlichkeit	17
4.2.3	Satzschablonen	17
4.3	CuriX	17
4.3.1	Design	18
4.3.2	Verständlichkeit	18
4.3.3	Satzschablonen	18
4.4	IBM Maximo	19
4.4.1	Design	19
4.4.2	Verständlichkeit	19
4.4.3	Satzschablonen	20
4.5	Vergleich der Systeme	20
4.5.1	Methodik zur Bewertung der Best Practices	20
4.5.2	Tabelle Vergleich der Systeme	21
4.5.3	Schlussfolgerung des Vergleichs	21
5	Entwicklung eines konzeptuellen Modells für Issues in Systemen	22
5.1	Attribute von Issues	22
5.2	Lifecycle von Issues	23
5.3	Designfaktoren	24
5.4	Nicht-Designfaktoren	24
5.5	Wizard-Modus	25
5.6	Satzschablonen	25
5.6.1	Satzschablone für den Titel	25
5.6.2	Vorlage für die Beschreibung	25
5.7	Ersteller von Issues	26

5.8	Feedback-Zyklus für Issues	26
5.8.1	Wizard-Feedback	26
5.8.2	Initiales Feedback	27
5.8.3	Finales Feedback	27
5.9	Anwendung von KI zur Verbesserung der Issues	27
5.9.1	Hilfe bei der Erstellung von Issues	27
5.9.2	Integration von Terminal	27
5.9.3	Integration von Chat-Funktion	28
5.9.4	KI-gestützte Analyse und Optimierung	28
5.9.5	KI als Zwischenlayer	28
5.10	Wireframes	29
5.10.1	Listenansicht	29
5.10.2	Detailansicht	30
5.10.3	Feedbackansicht	31
6	Proof of Concept für das entwickelte Modell	33
6.1	Technologie-Stack	33
6.2	Listenansicht	34
6.3	Detailansicht aufgeklappt	37
6.4	Bearbeitung Detailansicht	40
6.5	Chat-Funktion	40
6.6	Terminal-Funktion	41
6.7	Erstellen von Issues	42
6.8	Wizard	43
6.9	Wizard-Feedback	45
6.10	Initial-Feedback	46
6.11	Final-Feedback	47
7	Evaluation des entwickelten Modells	49
7.1	Einleitung	49
7.2	Zielgruppe der Evaluation	49
7.3	Überblick über die angewandten Methoden und Vorgehensweise	49
7.3.1	Testmethoden	49
7.3.2	Vorgehensweise	49
7.3.3	Testaufgaben	50
7.4	Ergebnisse der Evaluation	50
7.4.1	Listenansicht	50
7.4.2	Detailansicht	50
7.4.3	Bearbeitungsansicht	51
7.4.4	Neues Issue-Ansicht	51
7.4.5	Design	51
7.4.6	Nicht-Design	51
7.4.7	KI-Funktionen	52
7.4.8	Wizard-Modus	52
7.4.9	Feedback	52
8	Schlussfolgerung	55
9	Ausblick	56
9.1	Automatisierte Fehlerlösung durch KI	56
9.2	Blacklist Satzschablone	56
9.3	Animationen	56
10	Anhang	57
10.1	Wizard-Feedback	57
10.2	Initial-Feedback	57
10.3	Final-Feedback	58
10.4	A/B-Testing	61
10.5	Feedbackformular Usertesting	64

10.5.1 Fragen zur Benutzeroberfläche und Navigation	64
10.5.2 Fragen zu den KI-Funktionen und Feedback-Mechanismen	65
10.5.3 Fragen zur Benutzerfreundlichkeit und allgemeinen Zufriedenheit	66
11 Hilfsmittelverzeichnis	68
12 Eigenständigkeitserklärung	68
Quellenverzeichnis	69

1 Einleitung

1.1 Hintergrund und Kontext des Projektes

Das Projekt „What’s That Error?“ zielt darauf ab, die Darstellung und Verständlichkeit von Fehlermeldungen zu verbessern. Es wird untersucht, wie Fehlermeldungen im Allgemeinen so gestaltet werden können, dass sie für Benutzer und vor allem auch für Laien klarer und handlungsorientierter sind.

Die gewonnenen Erkenntnisse werden in einem Proof of Concept (PoC) dargestellt, welches eine Systemüberwachungssoftware umfasst. Solche Softwarelösungen überwachen IT-Infrastrukturen um Abweichungen oder Anomalien als „Issues“ zu melden. Diese Fehlermeldungen werden in einem interaktiven Dashboard angezeigt. Aktuell ist die Darstellung jedoch oft ineffizient und für Benutzer schwer verständlich.

Zusätzlich wird untersucht, wie ein Feedbackmechanismus entwickelt werden kann, der die Verständlichkeit und Handlungsfähigkeit der Issues bewertet. Ausserdem wird das Potenzial von Künstlicher Intelligenz (KI) zur Unterstützung der Fehlerbehebung und Erstellung von Issues untersucht.

1.2 Problemstellung

Aktuelle Fehlermeldungen in IT-Systemen sind häufig unklar und schwer verständlich. Diese Unklarheit führt zu mehreren Herausforderungen, die sowohl die Benutzererfahrung als auch die Effizienz der Problemlösung beeinträchtigen. Unklare Fehlermeldungen erfordern zusätzliche Zeit und Ressourcen, um die Ursache des Problems zu identifizieren und geeignete Massnahmen zu ergreifen. Da die Fehlermeldungen nicht sofort verständlich sind, dauert es länger, bis Probleme behoben werden können, was zu höheren Ausfallzeiten und Beeinträchtigungen der IT-Infrastruktur führen kann.

Ein Benutzer, der mit unklaren Fehlermeldungen konfrontiert ist, fühlt sich frustriert und überfordert, da er mehr Zeit aufwenden muss, um die Ursache zu finden und das Problem zu beheben. Dies beeinträchtigt seine Fähigkeit, effektiv zu arbeiten, und verringert seine Zufriedenheit mit dem System. Es fehlt an klaren Handlungsempfehlungen, die dem Benutzer zeigen, wie er das Problem lösen kann.

Besonders herausfordernd sind diese Probleme für unerfahrene Benutzer, die im Gegensatz zu Experten oft nicht über das notwendige Hintergrundwissen verfügen, um komplexe Fehlermeldungen zu interpretieren. Während Experten möglicherweise in der Lage sind, die Bedeutung unklarer Fehlermeldungen schneller zu entschlüsseln, benötigen Laien mehr Unterstützung und klare Anweisungen.

1.3 Zielsetzung

Das Ziel des Projekts „What’s That Error?“ ist die Entwicklung eines Proof of Concept zur Verbesserung der Darstellung von Issues.

Um dieses Ziel zu erreichen, wurden folgende spezifische Ziele definiert:

- **Z-1:** Durchführung einer Analyse zur Fehlermeldungsdarstellung und -interaktion, um ein Verständnis für die aktuelle Benutzererfahrung zu entwickeln.
- **Z-2:** Entwicklung eines Konzepts für Fehlermeldungen, welches sich auf die Verständlichkeit, Design und Handlungsfähigkeit fokussiert.
- **Z-3:** Entwicklung eines Feedbackmechanismus, der es Benutzern ermöglicht, die Verständlichkeit und die Handlungsempfehlungen von Fehlermeldungen zu bewerten. Dieser ermöglicht ebenfalls fehlenden Kontext zu ergänzen, um die Qualität zu verbessern.
- **Z-4:** Untersuchung des Potenzials von KI in der Fehlerbehebung und Entwicklung eines konkreten Konzeptentwurfs, der darstellt, wie Lösungswege zur Behebung von Problemen vorgeschlagen werden können.

1.4 Forschungsfragen

Um die Zielsetzung zu erreichen und die Problemstellungen zu lösen, wurden folgende Forschungsfragen formuliert:

- **RQ1:** Wie können Fehlermeldungen so gestaltet werden, dass diese für Benutzer klar und handlungsorientiert sind?

- **RQ1.1:** Welche sprachlichen Elemente erhöhen die Verständlichkeit von Fehlermeldungen?
- **RQ1.2:** Welche visuellen Elemente erhöhen die Verständlichkeit von Fehlermeldungen?
- **RQ1.3:** Wie kann Kontextinformation effizient in Fehlermeldungen integriert werden?
- **RQ2:** Wie kann ein Feedbackmechanismus konzipiert werden, welcher Benutzerfeedback zu Fehlermeldungen sammelt?
 - **RQ2.1:** Welche Feedbackmechanismen (Multiple-Choice, Skala, etc.) sind am effektivsten für die Bewertung von Fehlermeldungen?
 - **RQ2.2:** Wann ist der Zeitpunkt für den Benutzer angenehm, um Feedback zu geben?
 - **RQ2.3:** Wie kann das Feedback geben für den Benutzer visuell ansprechend gemacht werden und nicht aufdringlich?
- **RQ3:** Wie kann Künstliche Intelligenz genutzt werden, um die Fehlerbehebung zu optimieren?
 - **RQ3.1:** Wie können Daten gesammelt werden, damit das Modell Issues und Benutzerreaktionen erfasst, um zukünftig Lösungsvorschläge anzubieten? (Diese Frage wurde teilweise beantwortet.)
 - **RQ3.2:** Wie können Daten gesammelt werden, damit das Modell den Kontext des Problems lernen kann? (Diese Frage wurde teilweise beantwortet.)
 - **RQ3.3:** Wie können die Vorschläge zur Problemlösung der KI benutzerfreundlich und verständlich präsentiert werden?
- **RQ4:** Wie kann die Wirksamkeit der verbesserten Fehlermeldungen und Feedbackmechanismen in Bezug auf Benutzerfreundlichkeit und Problemlösung bewertet werden?
 - **RQ4.1:** Welche Methoden können eingesetzt werden, um die Klarheit und Handlungsorientierung der optimierten Fehlermeldungen zu messen und zu bewerten? (Diese Frage wurde teilweise beantwortet.)

1.5 Struktur des Berichts

Dieser Bericht gliedert sich in mehrere Kernkapitel, die den Weg von der Recherche über die Konzeption bis zur Realisierung des Projekts erklären.

Zu Beginn wird im Kapitel 2 der Hintergrund zu den behandelten Themen dargestellt, um ein grundlegendes Verständnis für die weiteren Themen zu schaffen.

In der State-of-the-Art-Analyse im Kapitel 3 wird der aktuelle Stand der Forschung im Bereich der Fehlermeldungsdarstellung und -interaktion untersucht. Hierbei werden auch die Methodik der Literaturrecherche beschrieben und die gefundenen Erkenntnisse zusammengefasst.

Im Kapitel 4 wird untersucht, welche bestehenden Lösungen und Produkte es auf dem Markt gibt, wie diese Fehlermeldungen darstellen und welche Best Practices dabei angewendet werden.

Das Kapitel 5 beschreibt das Konzept für ein verbessertes Fehlermeldungssystem. Es umfasst verschiedene Aspekte wie die Definition und Abgrenzung sowie Wireframes.

Die verschiedenen Ansichten des PoC werden im Kapitel 6 präsentiert.

Kapitel 7 fasst die Ergebnisse der durchgeführten Usability-Tests mit Benutzern zusammen. Hierbei wird untersucht, wie die entwickelten Konzepte in der Praxis funktionieren und wie Benutzer mit dem PoC interagieren.

Abschliessend fasst das Kapitel 8 die Projektergebnisse zusammen und Kapitel 9 gibt einen Ausblick.

2 Hintergrund Issue

2.1 Ziele und Funktionen von Issues

Issues erfüllen mehrere wesentliche Funktionen und tragen dazu bei, dass die Systemüberwachung effizient und effektiv durchgeführt werden kann. Die Ziele eines Issues lassen sich wie folgt zusammenfassen:

- **Kommunikation:** Issues helfen Teams, effektiv zu kommunizieren, indem sie ein zentrales Medium für die Erfassung und Diskussion von Details bieten.
- **Dokumentation:** Issues dienen als Aufzeichnung von Entscheidungen, Problemen und deren Lösungen sowie von Änderungen, Anpassungen und Fortschritt im Projektverlauf. Dies bietet eine Referenz für zukünftige Projekte und hilft, ähnliche Probleme zu vermeiden.
- **Management:** Durch die Strukturierung von Issues können Projekte systematisch überwacht und gesteuert werden. Priorisierung und Zuweisung von Verantwortlichkeiten werden dadurch erleichtert.
- **Nachverfolgbarkeit:** Issues ermöglichen es, den Fortschritt und die Behandlung spezifischer Aspekte eines Projekts oder Systems nachzuverfolgen und zu analysieren.

2.2 Kommunikationsmodell

Durch ein geeignetes Kommunikationsmodell kann besser verstanden werden, wie ein Issue wahrgenommen wird und wie es effektiv kommuniziert und bearbeitet werden kann.

Ein typisches Kommunikationsmodell für Warnungen besteht aus vier Komponenten: Source (Sender der Warnung), Channel (Plattform zur Übermittlung, beispielsweise Computer, mobile Geräte, Meetings), Message (Warninhalt) und Receiver [1].

Das Communication-Human Information Processing (C-HIP) Modell wiederum definiert 5 Empfängerstufen: Attention, Comprehension, Attitudes/Beliefs, Motivation und Behavior. Source und Channel gelten nicht als Empfängerstufen. In Abbildung 1 wird veranschaulicht, in welchem Ablauf Informationen verarbeitet werden [2].

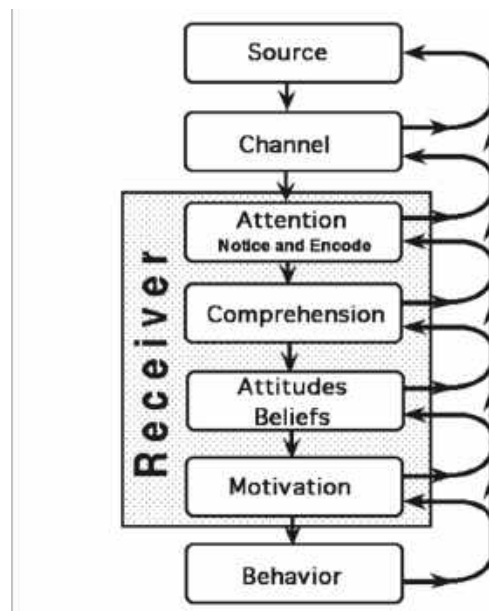


Abbildung 1: C-HIP Modell [2]

Hier sind die Phasen des C-HIP Modells und deren Anwendung auf Issues:

- **Source:** Die Quelle eines Issues kann ein System, Teammitglied, Kunde oder Stakeholder sein. Eine klare und glaubwürdige Quelle unterstreicht die Relevanz und Dringlichkeit des Issues.

- **Channel:** Issues sollten über geeignete Kanäle wie Dashboard, Ticketing-Systeme, E-Mails oder Meetings kommuniziert werden. Dies soll höhere Nachverfolgbarkeit und Transparenz sorgen.
- **Attention:** Das Issue muss die Aufmerksamkeit des Empfängers auf sich ziehen. Das kann durch klare Titel und detailreiche Beschreibungen erreicht werden. Die Aufmerksamkeit wird auf das Issue gelenkt, um sicherzustellen, dass es nicht übersehen oder auch vergessen wird.
- **Comprehension:** Nachdem die Aufmerksamkeit erlangt wurde, ist es wichtig, dass der Empfänger das Issue versteht. Wie schon bei Attention, soll eine detaillierte und klare Beschreibung dabei helfen, das Problem zu verstehen und die notwendigen Schritte zur Lösung zu erkennen.
- **Attitudes/Beliefs:** Die Wahrnehmung und Einstellung des Empfängers gegenüber einem Issue beeinflussen massgeblich, wie ernsthaft und effizient darauf reagiert wird. Ein als kritisch eingestuftes Issue erhält dabei deutlich mehr Aufmerksamkeit und Priorität. Deshalb ist es von Relevanz, die Attribute eines Issues möglichst korrekt anzugeben.
- **Motivation:** Um die Motivation zur Lösung eines Issues zu steigern, ist es wichtig, die Auswirkungen und die Dringlichkeit des Problems klar darzustellen. Ein motivierter Benutzer wird eher bereit sein, das Problem zu lösen, wenn er ausreichend und gut dargestellte Informationen hat.
- **Behaviour:** Auf das Issue sollte proaktiv reagiert werden und dieses sollte entsprechend seiner Priorität gelöst werden. Dies wird durch klare Anweisungen und definierte Handlungsrichtlinien erreicht.

Wie man in der Abbildung 1 sieht, befindet sich die Phase Attention bis und mit Motivation im Receiver-Bereich. Diese Schritte beschreiben die Art und Weise, wie der Empfänger das Issue wahrnimmt und darauf reagiert. Zunächst wird die Aufmerksamkeit des Empfängers geweckt, damit das Issue bemerkt wird. Danach muss der Empfänger das Issue verstehen, was durch klare und detaillierte Informationen gemacht wird. Die Einstellungen und Glaubenssätze des Empfängers beeinflussen dann, wie er das Issue priorisiert und als wie wichtig er es empfindet. Schliesslich motiviert das Verständnis der Auswirkungen und Dringlichkeit des Issues den Empfänger, aktiv zu handeln und das Problem zu lösen. Auch ist aus der Abbildung 1 zu nehmen, dass diese Phasen iterativ sein können, was durch die nach oben führenden Pfeile verdeutlicht wird. Dies zeigt, dass jede Phase des Modells wiederholt durchlaufen werden kann. Man nehme an, ein Entwickler stösst während der Problemlösung auf neue Aspekte des Problems, so kehrt er wieder auf eine frühere Phase.

3 State of the Art Analyse

3.1 Einführung

In diesem Kapitel wird der aktuelle Stand der Forschung im Bereich der Darstellung und Interaktion von Fehlermeldungen untersucht. Ziel ist es, relevante Literatur zu identifizieren, um bestehende Lücken aufzuzeigen und Verbesserungspotenziale für das Projekt „What’s That Error?“ herauszuarbeiten.

Für eine umfassendere Analyse und aufgrund der begrenzten Verfügbarkeit von Literatur, die sich direkt auf Issues im Sinne dieses Berichts bezieht, konzentriert sich die Recherche auf verschiedene Arten von Meldungen und nicht nur auf Fehlermeldungen. Folgende Arten wurden betrachtet: Alert Message, Warning Message, Error Message, Success Message und Bugs. Diese Arten von Meldungen können im weiteren Sinne ebenfalls als Issues betrachtet werden, wobei sich Ziel und Zeitpunkt der Anzeige unterscheiden. Beispielsweise werden Warning Messages vor einer Aktion angezeigt, um Informationen über mögliche Risiken zu geben, während Error Messages nach einer fehlgeschlagenen Aktion erscheinen und darüber informieren, warum die Aktion nicht erfolgreich war. Die folgende Tabelle zeigt die Definitionen der verschiedenen Meldungen:

Error Message	Informiert den Benutzer über ein aufgetretenes Problem [3].
Alert Message	Weist auf potenzielle Probleme oder Risiken hin, die nicht unbedingt kritisch sind [4].
Warning Message	Ähnlich wie Alert Message, betont jedoch das Risiko von Datenverlust oder unerwünschte Zustände [5].
Success Message	Bestätigt eine abgeschlossene Aktion oder Aufgabe [6].
Bug	Ein Fehler oder eine Unstimmigkeit im Quellcode, die zu unerwartetem Verhalten oder Fehlfunktionen führt [7].
Issue	Ein Oberbegriff für alle Arten von Aufgaben, Problemen oder Anfragen, die in einem System verfolgt werden (eigene Definition).

Tabelle 1: Verschiedene Arten von Meldungen

Die zugrundeliegenden Theorien und Gestaltungsprinzipien der verschiedenen Arten von Meldungen sind ähnlich und können zusammengefasst werden. Die bereits bekannten Best Practices für all diese Meldungstypen lassen sich weitgehend auch auf Issues anwenden. Im Bericht wird diese Annahme nicht direkt geprüft, aber die darauf basierenden Richtlinien werden validiert.

Um die Lesbarkeit und Konsistenz im weiteren Verlauf des Kapitels zu gewährleisten, wird ab jetzt immer der Begriff „Meldungen“ verwendet. Damit sind alle zuvor definierten Meldungsarten – Alerts, Success, Warnings, Errors und Bugs – gemeint.

3.2 Perspektiven aus verschiedenen Domänen

Der Begriff Issue findet sich nicht nur in der IT-Welt, sondern auch in anderen Domänen. Um ein allgemeines Verständnis von Issues zu entwickeln, wurden die verschiedenen Anwendungen des Begriffs in anderen Bereichen untersucht.

- **Softwareentwicklung:** In der Softwareentwicklung bezieht sich der Begriff Issue häufig auf Bugs, Störungen und Anforderungen. Diese Issues beeinflussen die Funktionalität und Qualität der Software und erfordern eine systematische Analyse und Behebung [8].
- **Projektmanagement:** Im Projektmanagement werden Issues oft als Features, Hindernisse, Risiken oder Abweichungen vom Plan betrachtet. Sie können den Fortschritt und den Erfolg eines Projekts erheblich beeinflussen und müssen daher effektiv gemanagt werden [9].
- **Geschäftswelt:** In der Geschäftswelt umfassen Issues Kundendienstprobleme, Betriebsunterbrechungen und Reputationsrisiken. Diese können die Kundenzufriedenheit und den Geschäftserfolg beeinträchtigen und erfordern proaktive Massnahmen zur Lösung [10].
- **Öffentliche Politik und Verwaltung:** In der öffentlichen Politik und Verwaltung beziehen sich Issues auf gesellschaftliche Herausforderungen und politische Streitfragen. Diese Themen erfordern oft komplexe Lösungen und umfassende politische Debatten [11].

- **Alltägliche Kontexte:** Im Alltag kann ein Issue jede Art von Streitpunkt oder Sorge darstellen, die das tägliche Leben beeinflusst. Diese können persönliche, soziale oder berufliche Aspekte betreffen und erfordern individuelle Lösungsansätze [12].

Die Sicht auf verschiedenen Domänen hat gezeigt, dass es stets Gemeinsamkeiten gibt. Im Folgenden sind die gemeinsamen Merkmale aufgeführt:

- **Auswirkungen und Priorisierung:** Jedes Issue hat eine Auswirkung auf ein Gesamtsystem und muss entsprechend priorisiert werden.
- **Stakeholder-Interessen:** Stakeholder, darunter sowohl Experten mit Fachkenntnissen als auch Laien, die Anweisungen benötigen, haben ein Interesse an der Lösung des Problems.
- **Informationsgehalt:** Alle Issues beinhalten auf irgendeiner Art Informationen.
- **Lebenszyklus:** Jedes Issue durchläuft einen Lebenszyklus.

Diese Perspektiven und Gemeinsamkeiten verdeutlichen, dass Issues je nach Kontext unterschiedliche Bedeutungen und Anforderungen haben können. Im Allgemeinen kann man für ein Issue in Betracht auf verschiedenen Domänen also sagen, es ist eine Angelegenheit, eine Fragestellung oder ein Problem, das Aufmerksamkeit, Diskussion oder eine Lösung erfordert. Daraus resultierend, konnte eine Definition für ein Issue festgelegt werden:

Ein Issue ist ein Element eines Systems oder Projekts, das dazu dient, Informationen strukturiert zu speichern, darzustellen und zu verwalten. Diese Informationen repräsentieren Aspekte, die Aufmerksamkeit erfordern und können in Form von Problemen, Herausforderungen oder möglichen Verbesserungen auftreten.

3.3 Methodik der Literaturrecherche

Die Suche nach relevanten Studien wurde in den folgenden verschiedenen wissenschaftlichen Datenbanken durchgeführt:

- Google Scholar
- IEEE Xplore
- ResearchGate
- ScienceDirect
- Scientific.net

Es wurden 43 Studien angeschaut, von denen 23 in dieser Arbeit verwendet und zitiert wurden. Folgende Suchbegriffe wurden angewendet:

- **ERROR MESSAGES**
- **ERROR MESSAGE OPTIMIZATION**
- **WARNING MESSAGES**
- **USABILITY DASHBOARDS**
- **ALERT MESSAGES**
- **SOFTWARE BUG**
- **ISSUE MONITORING BEST PRACTICE**
- **DESIGN AND TESTING WARNING MESSAGES**
- **UNDERSTANDABILITY SYSTEM MESSAGES**
- **DESIGN ON ERROR MESSAGES**

- **ERROR DESIGN PROGRAMMING GUIDELINES**
- **IMPLICIT EXPLICIT FEEDBACK**
- **SUCCESSION MESSAGES**
- **DESIGN HEURISTICS**
- **ACTIONABILITY SYSTEM MESSAGES**
- **ACTIONABLE MESSAGES**

3.4 Merkmale von Issues

Die in den folgenden Unterkapiteln behandelten Themen basieren auf einer Analyse der Literatur und heben die wichtigsten Merkmale von Meldungen hervor. Diese Merkmale umfassen unter anderem Grösse, Farbe, Form, Platzierung, Sprache und den Einsatz von Icons, ähnlich wie es in der Arbeit von Laughery und Wogalter [13] untersucht wurde. Die hier dargestellten Überlegungen greifen die zentralen Aspekte auf, die in der Forschung zu diesen und weiteren Merkmalen diskutiert werden.

3.4.1 Design

Das Design ist das erste, was Benutzer von einer Meldung wahrnehmen, bevor sie den eigentlichen Inhalt lesen. Der erste Eindruck ist hierbei entscheidend. Allgemein soll das Design von Meldungen Ausgewogenheit und Ansprechbarkeit beinhalten [14]. Das visuelle Design setzt sich aus verschiedenen Elementen zusammen, die im Folgenden näher erläutert werden.

3.4.2 Auffälligkeit

Eine Möglichkeit, das Verständnis und die Erinnerung an Meldungen zu verbessern, besteht darin, sicherzustellen, dass sie zunächst bemerkt und gelesen werden. Dies kann erreicht werden, indem zum Beispiel der Warntext auffälliger gestaltet wird als das restliche Textmaterial in einem Handbuch oder in einer Meldung. Dazu gehört das Vergrössern und/oder Fettdrucken des Textes sowie das Hinzufügen von Farben [15].

Zusätzlich tragen weitere Faktoren zur Auffälligkeit von Warnungen bei. Dazu gehören der Ort der Warnung, die verwendeten Farben, Icons, die Länge der Nachricht und die physische Interaktivität der Meldung. Diese Elemente sind entscheidend, um sicherzustellen, dass Meldungen effektiv wahrgenommen und verstanden werden [13].

Die Bedeutung von klarer und auffälliger Fehlermeldungen in der Softwareentwicklung wird durch eine Studie [16] hervorgehoben. Die Ergebnisse zeigen, dass Entwickler etwa 13 bis 25 Prozent ihrer Zeit auf das Lesen von Fehlermeldungen verwenden, was zeigt, dass man Meldungen so verständlich wie möglich gestalten soll.

„Wenn eine Warnung nicht beachtet wird, ist sie gleichbedeutend mit keiner Warnung [15].“

3.4.3 Icons und Piktogramme

Icons oder Piktogramme (beide Begriffe beziehen sich auf grafische Symbole, die zur Darstellung von Informationen oder zur Verdeutlichung von Meldungen verwendet werden) können in Meldungen gebraucht werden, um non-verbal Informationen zu kommunizieren wie zum Beispiel Konsequenzen, Gefahren und Schweregrad. Diese tragen signifikant zur Klarheit und Schnelligkeit der Informationsverarbeitung bei. In einer Untersuchung erleichterte das Vorhandensein von Symbolen nicht nur das Verständnis der Symbole selbst, sondern auch das Verständnis der mit dem Symbol verbundenen Meldung. Die Kombination von auffälligem Druck und anschauliche Piktogramme verbessern das Verständnis und das Erinnerungsvermögen des Inhalts von Meldungen [15].

In einer Umfrage wurden Benutzer gebeten, die wahrgenommene Kritikalität von Icons wie einem roten Kreuz (Error), einem gelben Ausrufezeichen (Warning) und einem blauen „i“ (Information) zu bewerten. Die Mehrheit der Benutzer empfand das rote Kreuz als das kritischste Symbol, gefolgt vom gelben Ausrufezeichen und schliesslich dem blauen „i“ [14]. Dies bedeutet, dass Benutzer mit diesen Symbolen gut vertraut sind und die Bedeutung davon auch kennen.

3.4.4 Farben

Wie bereits erwähnt, können Farben auch benutzt werden, um die Wahrscheinlichkeit zu erhöhen, dass die Meldungen wahrgenommen werden [15].

Farben werden in vielfältigen Zusammenhängen verwendet, um Aufmerksamkeit zu erregen. Daher ist es sinnvoll, sie auch in Verbindung mit Meldungen zu nutzen. Es wäre vorteilhaft, wenn die Farbe selbst darauf hinweist, dass eine Meldung vorliegt. Angesichts der Vielzahl von Objekten unterschiedlicher Farben in einer Umgebung ist es jedoch unwahrscheinlich, dass es möglich ist, einzigartige Farben für diesen Zweck auszuwählen. Sicherlich sind die gebräuchlichen Farben nicht einzigartig, aber es wurde vorgeschlagen, dass die Farben mit dem Risikoniveau der Gefahr abgestimmt werden sollten [15]. Daher sollte die Farbwahl das Risikoniveau der Gefahr reflektieren.

Farben können die Visualisierungen von Dashboards verbessern, aber eine übermäßige oder falsche Verwendung bestimmter Farbpaletten kann negative Auswirkungen auf die Entscheidungsfindung haben [17]. Mit Hilfe von Eye-Tracking-Technologie wird berichtet, dass der zufällige Einsatz von Farben in Dashboards zwar keine schlechten Entscheidungen verursacht, jedoch die Zeit verlängern kann, die für das Treffen einer angemessenen Entscheidung erforderlich ist [17].

Dies bedeutet, dass die sorgfältige Auswahl und Anwendung von Farben in Meldungen die Wahrnehmung und das Verständnis dieser Meldungen erheblich verbessern kann. Die richtige Farbauswahl kann nicht nur die Aufmerksamkeit erhöhen, sondern auch die Dringlichkeit und den Schweregrad einer Warnung verdeutlichen, was zu schnelleren und angemesseneren Reaktionen führen kann.

3.4.5 Layout

Ein gut gestaltetes Layout ist essenziell für die Effektivität von Meldungen. Das Layout beeinflusst die Lesbarkeit, Verständlichkeit und die Geschwindigkeit, mit der Benutzer die notwendigen Informationen aufnehmen und darauf reagieren können. Das Layout soll klar und einfach gehalten werden, um Verwirrung zu vermeiden. Überladene und komplexe Layouts erschweren es den Benutzern, die wichtigen Informationen schnell zu erfassen. Es wurde festgestellt, dass Fehlermeldungen oft uneinheitlich sind und verschiedene Dialoge für ähnliche Fehler angezeigt werden [14]. Ein konsistentes Layout über verschiedene Meldungen hinweg hilft den Benutzern, sich schnell zurechtzufinden. Dies bedeutet, dass die Platzierung von Titeln, Symbolen und Buttons standardisiert sein sollte.

3.4.6 Inhalt

Der Inhalt von Meldungen sollte spezifisch sein, detailliert und klar formuliert. Der Text darf nichts implizieren und der Benutzer sollte nicht noch selber logische Schlussfolgerungen ziehen müssen [13]. Eine Untersuchung zeigt, dass konkrete Informationen besser erinnert werden als abstrakte Informationen. Dies ist ein zentraler Aspekt des sogenannten Konkretheitseffekts. Konkretere Wörter führen zu einer stärkeren Aktivierung bestimmter Gehirnregionen [18]. Der Inhalt sollte auch von Personen verstanden werden, die als Laien definiert werden können. Vage Sicherheitswarnungen wie „Gefährliche Umgebung“ oder „Vorsichtsmassnahmen ergreifen“ sind kaum hilfreich, da sie keine konkrete Hinweise auf die Gefahren oder erforderlichen Massnahmen geben. Im Gegensatz dazu liefert eine klar formulierte Meldung detaillierte Informationen über die Gefahren, die möglichen Konsequenzen und die genauen Schutzmassnahmen. Dies ist wichtig, damit Benutzer fundierte Entscheidungen treffen und angemessen auf Gefahren reagieren können. Laughery und Wogalter [13] haben folgende Prinzipien zur Explizitheit eruiert:

- Nicht davon ausgehen, dass es jeder weiss
- Nicht auf Schlussfolgerungen verlassen
- Vorsichtig mit Annahmen, dass Gefahren und Konsequenzen offensichtlich sind
- Erinnerungen können notwendig sein
- Explizit bedeutet nicht unbedingt quantitativ
- Fachjargon ist nicht der beste Weg, um Explizitheit zu erreichen, besonders für ein allgemeines Publikum

Genauere Informationen über Gefahren und Konsequenzen ermöglichen es dem Benutzer, fundiertere Entscheidungen darüber zu treffen, ob die Einhaltung der Vorschriften notwendig ist. Forschungsergebnisse zeigen, dass explizite Informationen besonders wichtig sind, wenn die Konsequenzen schwerwiegender sind.

Im Falle einer Meldung, sollte diese den Fehler erklären, den Grund dafür nennen und dem Benutzer in intuitiver Weise mögliche Massnahmen zur Behebung aufzeigen.

„Fehlermeldungen sollten freundliche und hilfreiche Anweisungen geben, um dem Benutzer zu helfen, anstatt in einem anklagenden Ton zu sein [14].“

3.4.7 Textlänge

Eine kurze und prägnante Darstellung ist ein allgemein anerkanntes Kriterium für Meldungen, diese sollten so kurz wie möglich sein, um die notwendige Information effektiv zu vermitteln [19].

Bei der Erstellung solcher Meldungen sollten neben unbekannte Begriffe oder verwirrende Formulierungen unbedingt lange Nachrichten vermieden werden [20].

Lange und komplexe Meldungen können den Benutzer überfordern und dazu führen, dass die wichtige Nachricht übersehen oder missverstanden wird. Daher ist es entscheidend, dass die Sprache klar und direkt ist, um die Aufmerksamkeit der Benutzer zu gewinnen und die notwendigen Handlungsanweisungen effektiv zu vermitteln.

3.4.8 Informationsüberflutung

Informationsüberflutung ist ein anerkanntes Phänomen, das mit dem kontinuierlichen Anstieg von Daten und dem entsprechenden Bedarf zur Verarbeitung dieser Informationen zusammenhängt. Ein Überfluss an Informationen schafft einen Mangel an Aufmerksamkeit [21]. Die Verarbeitungsfähigkeit eines Benutzers wird bei einer grossen Menge der verfügbaren Informationen überstiegen. Das führt dazu, dass relevante von irrelevanten Informationen schwierig zu unterscheiden sind.

Ein weiterer Aspekt der Informationsüberflutung ist die sogenannte „Information Anxiety“ – eine stressbedingte Bedingung, die durch die Unfähigkeit verursacht wird, notwendige Informationen zuzugreifen, zu verstehen oder zu nutzen. Die Ursache dafür kann entweder eine Informationsüberflutung oder ein Mangel an Informationen sein, ebenso kann es durch schlecht organisierte oder präsentierte Informationen verursacht werden oder durch einem Mangels an Verständnis für das Informationsumfeld, in dem man arbeitet [22].

3.4.9 Wirksamkeit

Die Wirksamkeit von Meldungen kann auf verschiedene Weise und anhand unterschiedlicher Kriterien bewertet werden, aber viele Autoren sind sich einig, dass das letztlich Kriterium für die Wirksamkeit von Meldungen die Einhaltung dessen ist [23].

3.4.10 Actionability

Im Kontext der Software-Analyse bezieht sich die Actionability auf die Fähigkeit von Erkenntnissen und Vorhersagen, die durch analytische Systeme generiert werden, direkt praktische Entscheidungen und Handlungen mitzuteilen. Modernere Software-Systeme erfordern, dass analytische Erkenntnisse nicht nur korrekt und präzise sind, sondern auch in konkrete, handlungsfähige Massnahmen umgesetzt werden können. Dies ist wichtig, um Risiken wie Software-Fehler zu minimieren und die Wartbarkeit sowie die Qualität von Software-Projekten zu verbessern [24].

Ein Beispiel für umsetzbare Analysen könnte die Vorhersage von Codebereichen sein, die anfällig für Fehler sind. Während eine solche Vorhersage allein wertvolle Informationen liefert, wird sie erst dann wirklich nützlich, wenn sie auch spezifische Handlungsempfehlungen gibt, wie etwa „Refaktoriere diese Methode, um die Komplexität zu reduzieren“ oder „Erstelle zusätzliche Unit-Tests für diese Klasse“. Laut Tantithamthavorn et al. [24] sollten umsetzbare Analysen daher nicht nur Vorhersagen machen, sondern auch klare, kontextspezifische Anleitungen bieten, welche Massnahmen ergriffen werden sollten.

Die Actionability von Software-Analysen ist eng mit dem Konzept der Erklärbarkeit verbunden. Benutzer sollen nicht nur verstehen, was vorhergesagt wird, sondern auch, warum es vorhergesagt wird und wie sie darauf reagieren sollten. Denny et al. [25] weisen jedoch darauf hin, dass es nicht nur um

die Erklärbarkeit geht, sondern auch um die Lesbarkeit und Verständlichkeit der Anleitungen. Sie argumentieren, dass Analysen umsetzbar sein müssen, indem die Nachrichten und Empfehlungen sowohl verständlich als auch leicht umsetzbar sind, besonders für unerfahrene Benutzer.

Meldungen in Software sollten ebenso wie analytische Systeme nicht nur Probleme aufzeigen, sondern auch konkrete und umsetzbare Schritte zur Lösung bieten. Wenn diese klar und verständlich formuliert sind, können sie den Benutzer effektiv dazu leiten, wie das Problem zu beheben ist. Sie sollten daher nicht nur beschreiben, was falsch gelaufen ist, sondern auch konkrete Empfehlungen geben, wie der Benutzer das Problem lösen kann.

3.5 Zusammenfassung der Best Practices aus der Literaturrecherche

Die Literaturrecherche hat mehrere Best Practices zur Präsentation und Verbesserung von Meldungen in IT-Systemen hervorgebracht. Diese Best Practices sind essentiell für die Verbesserung der Benutzerfreundlichkeit und die Effizienz der Problemlösung:

Best Practice	Beschreibung	Anwendung
Farben	Einsatz von Farben, um die Dringlichkeit und den Schweregrad der Meldungen zu kennzeichnen.	Farben werden genutzt. Rot, Gelb und Blau wurden verwendet, da diese bereits weit akzeptiert sind.
Icons und Symbole	Verwendung von Icons und Symbolen, um die Kritikalität und Art der Meldungen schnell erkennbar zu machen. Symbole sollten intuitiv und allgemein verständlich sein.	Icons sind sinnvoll und werden integriert, um die Art der Meldung zu identifizieren. Es werden einfache und verständliche Icons gewählt.
Layout und Design	Ein gut gestaltetes Layout, das durch klare Struktur und visuelle Trennung der Informationen die Lesbarkeit und Verständlichkeit erhöht, ist entscheidend.	Das Layout wird so einfach wie möglich gehalten, ebenso das Design. Visuelle Trennung und klare Struktur werden priorisiert.
Actionability	Meldungen sollten zur Handlung motivieren. Klare Handlungsanweisungen und Hinweise erhöhen die Wahrscheinlichkeit, dass Benutzer die erforderlichen Schritte zur Problemlösung unternehmen.	Die Nachrichten werden so gestaltet, dass sie so actionable wie möglich sind, um dem Benutzer klare Aktionen zu ermöglichen.
Klarheit, Explizitheit und Prägnanz	Meldungen sollten explizit und prägnant formuliert sein, um Verwirrung zu vermeiden. Kurze und klare Aussagen erhöhen die Verständlichkeit.	Satzschablonen werden als Hilfsmittel verwendet, um Klarheit und Prägnanz zu gewährleisten.
Konsistenz	Konsistente Verwendung von Sprache und Design über alle Fehlermeldungen hinweg.	Satzschablonen sorgen für sprachliche Konsistenz, während gleiche Strukturen und Farben für Designkonsistenz sorgen.
Kontextuelle Informationen	Bereitstellung von ausreichendem Kontext, damit der Benutzer die Ursache und Lösung des Problems verstehen kann. Dazu gehören beispielsweise detaillierte Beschreibungen und relevante Hintergrundinformationen.	Es wird darauf geachtet, genügend kontextuelle Informationen bereitzustellen, unterstützt durch KI.
Benutzerfreundlichkeit	Gestaltung der Meldungen, die für alle Benutzer, sei es Experte oder Laie, leicht verständlich sind. Vermeidung von Fachjargon und komplexen technischen Begriffen.	Alle vorherigen Best Practices tragen zur Benutzerfreundlichkeit bei, die sowohl für Experten als auch für Laien gewährleistet wurde.
Informationsüberflutung vermeiden	Vermeidung von zu vielen Informationen auf einmal. Wichtige Informationen sollten hervorgehoben und weniger relevante Details ausgeblendet oder in sekundären Ansichten dargestellt werden.	Aufklappbare Spalten, White-space und visuelle Hierarchie wird berücksichtigt, um Informationsüberflutung zu vermeiden.

Tabelle 2: Best Practices für Meldungen mit getroffenen Entscheidungen

4 Marktanalyse

In diesem Kapitel wird der aktuelle Markt für Systemüberwachungssoftware analysiert. Ziel ist es, bestehende Lösungen zu identifizieren, deren Stärken und Schwächen herauszuarbeiten und mögliche Verbesserungspotenziale aufzuzeigen.

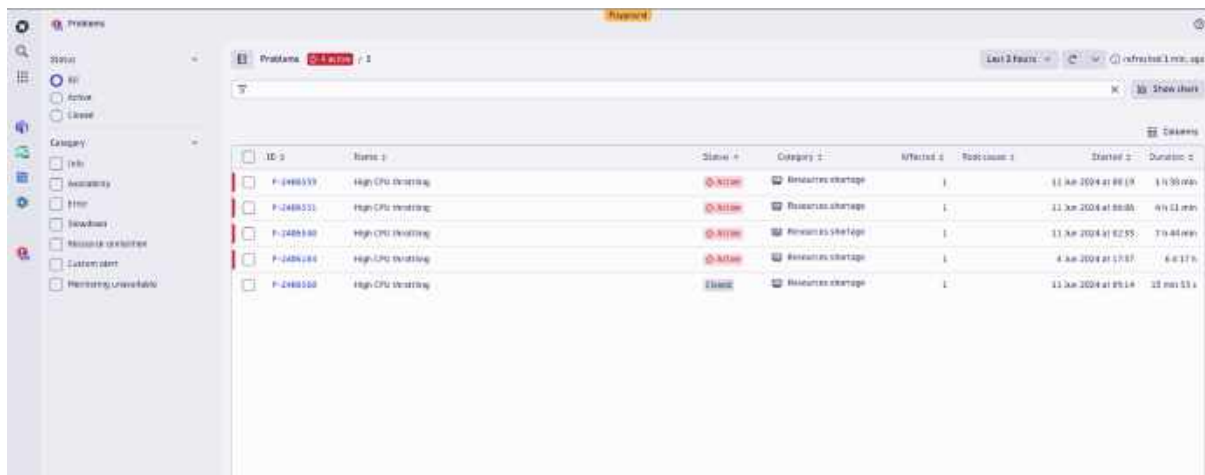
4.1 Auswahl der analysierten Systeme

Es wurden die drei bestehenden Systeme Dynatrace, CuriX und IBM untersucht, um ihre Ansätze zur Darstellung von Meldungen zu verstehen. Dabei wurde das Design und die Verständlichkeit der Meldungen genauer betrachtet, ebenso wie die Verwendung von Satzschablonen, sofern vorhanden. Die Arbeit mit Satzschablonen trägt dazu bei, dass Sätze einheitlich strukturiert sind und so gewisse Mängel, wie beispielsweise das Fehlen wesentlicher Informationen, von vornherein vermieden werden. Satzschablonen sind vordefinierte Textbausteine, die in Sätzen verwendet werden, um Konsistenz und Klarheit zu gewährleisten. Da diese Schablonen nicht in der offiziellen Dokumentation der Systeme zu finden waren, wurden diese aus Dokumentationen, Videos und Screenshots von vorhandene Issues der Systeme interpretiert. Die Systeme Dynatrace, CuriX und IBM Maximo wurden für die Analyse aus folgenden Gründen ausgewählt:

- **Etablierte Marktführer:** Dynatrace und IBM Maximo sind etablierte Marktführer im Bereich der Systemüberwachung und des Asset-Managements. Ihre breite Nutzung und umfassenden Funktionen machen sie zu wichtigen Referenzsystemen. IBM ist zudem weltweit bekannt.
- **Schweizer Startup:** CuriX ist ein Startup und wurde gewählt, da diese Software noch viel Raum für Verbesserungen und Anpassungen hat, was es zu einem interessanten Untersuchungsobjekt macht.

4.2 Dynatrace

Dynatrace ist eine Monitoring-Plattform, welche Unternehmen dabei hilft, ihre Anwendungen und IT-Infrastruktur zu überwachen und analysieren. Abbildung 2 zeigt die Listenansicht über aktuelle und vergangene Issues (bei Dynatrace „Problems“ genannt) in Dynatrace. Abbildung 3 zeigt die Detailansicht eines spezifischen Issues.



ID	Name	Status	Category	Affected	Root cause	Started	Duration
P-248833	High CPU throttling	Active	Resource shortage	1		11 Jun 2024 at 08:18	15:39 min
P-248832	High CPU throttling	Active	Resource shortage	1		11 Jun 2024 at 08:08	10:11 min
P-248830	High CPU throttling	Active	Resource shortage	1		11 Jun 2024 at 02:55	7:04 min
P-248828	High CPU throttling	Active	Resource shortage	1		4 Jun 2024 at 17:07	6:47 min
P-248820	High CPU throttling	Done	Resource shortage	1		11 Jun 2024 at 05:14	12 min 13 s

Abbildung 2: Dynatrace Listenansicht

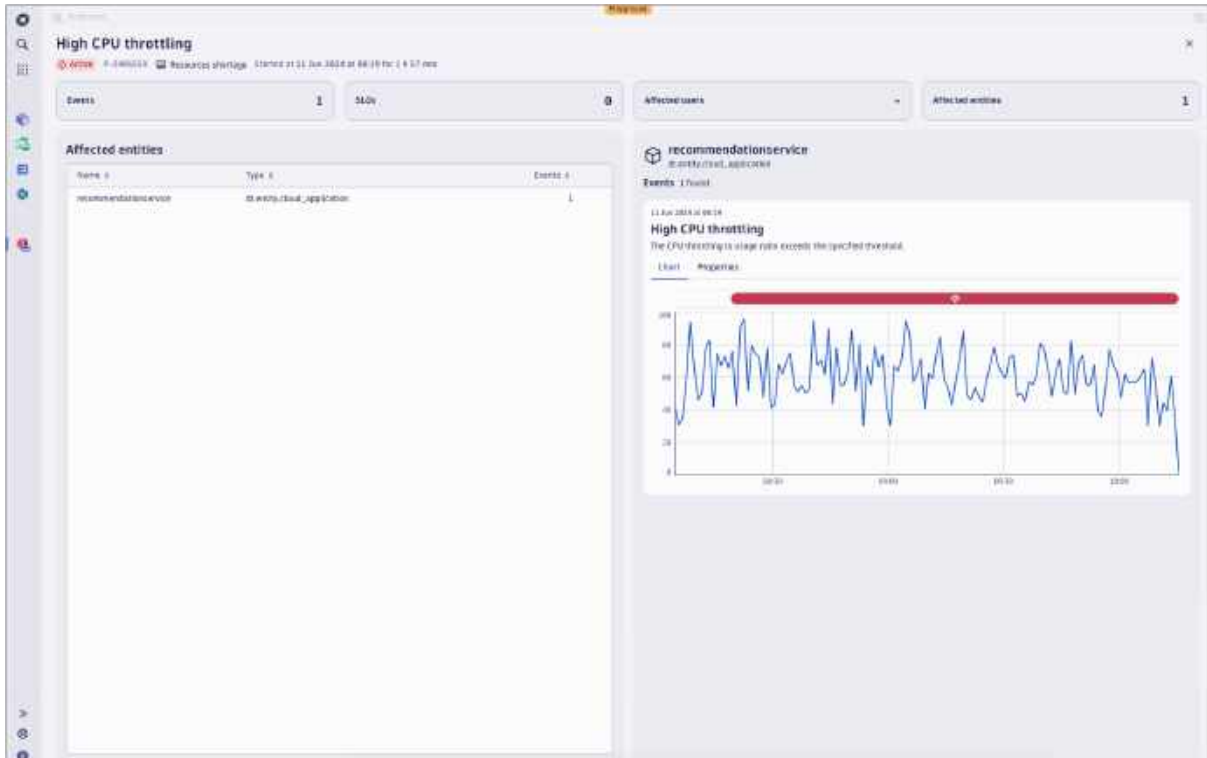


Abbildung 3: Dynatrace Detailansicht

4.2.1 Design

Die Listenansicht ist klar und übersichtlich gestaltet und teilt sich in drei Hauptbereiche auf. Eine Farbpalette, hauptsächlich in Grautönen, wird verwendet, wobei Akzente in Rot und Blau für Statusanzeigen gesetzt sind. Die Icons sind so gewählt, dass sie intuitiv sind, allerdings fehlen Hinweise auf interaktive Elemente oder Aktionen, die der Benutzer direkt aus dieser Ansicht heraus ausführen kann. Das Hinzufügen solcher interaktiven Hinweise könnte die Benutzerfreundlichkeit verbessern. Die Möglichkeit, direkt aus der Listenansicht heraus Aktionen für ein bestimmtes Issue auszuführen, könnte die Verarbeitungszeit verkürzen und die Effizienz steigern. Die Detailansicht ist ebenfalls klar strukturiert und visuell angenehm. Die „Affected Entities“ Box ist jedoch sehr gross und könnte effizienter genutzt werden und beispielsweise zusätzliche Informationen zu visualisieren.

4.2.2 Verständlichkeit

In Bezug auf die Verständlichkeit lässt sich feststellen, dass die Listenansicht eine klare Struktur bietet und wichtige Informationen eines Issues auf einen Blick sichtbar sind. Die Beschreibung des Issues ist nicht sehr hilfreich und zu kurz. Es wäre nützlich, detaillierte Informationen über beispielsweise das Problem, seine Ursachen oder mögliche Lösungen bereitzustellen. Eine erweiterte Beschreibung könnte zusätzlichen Kontext und spezifische Handlungsempfehlungen enthalten.

4.2.3 Satzschablonen

Bei der Untersuchung der Issues in Dynatrace wurden folgende Satzschablonen identifiziert:

- **The CPU throttling to usage ratio exceeds the specified threshold**
 - **Struktur:** [Subjekt] [Prädikat] [Objekt] [Konjunktion] [Bedingung/Ort/Aktion/Prozess]
 - **Beispiel:** The CPU throttling to usage ratio exceeds the specified threshold.

4.3 CuriX

CuriX ist eine Plattform zur Überwachung und Verwaltung von IT-Systemen. Sie verwendet maschinelles Lernen und Algorithmen, um Anomalien zu erkennen und Vorhersagen zu treffen. Die Detailansicht wurde

nicht berücksichtigt, da kein Zugriff darauf möglich war.

Creation	Severity	Title	Duration	Location
26.02.2024 21:17	High	Health Score reduced	4 min	10.103.10.101 VM-1 SHD
26.02.2024 21:17	High	Health Score reduced	4 min	10.102.10.13 Self Healing II (Linux Unix)
26.02.2024 21:17	High	Health Score reduced	4 min	curikurikresdev11
26.02.2024 21:17	High	Health Score reduced	4 min	Probe Device
26.02.2024 21:17	High	Health Score reduced	4 min	10.103.10.114 Curix Test Stack
26.02.2024 21:17	High	Health Score reduced	4 min	10.103.10.108 VM-2 PVE
26.02.2024 21:17	Medium	Component PRTG Core Server not in CURIX	4 min	PRTG Core Server
26.02.2024 21:17	Medium	Component Probe Device not in CURIX	4 min	Probe Device
26.02.2024 21:17	Medium	Component not in CMD9	4 min	ArtificialKpis
26.02.2024 21:17	Medium	Component is not managed	4 min	ArtificialKpis
26.02.2024 21:17	Medium	Component is not managed	4 min	Curix
26.02.2024 21:17	Medium	Component icv0071 not in CURIX	4 min	icv0071
26.02.2024 21:17	Medium	Component ELASTICSEARCH not in CURIX	4 min	ELASTICSEARCH

Abbildung 4: CuriX Listenansicht

4.3.1 Design

Die Listenansicht bei CuriX ist kompakt und bietet eine klare und übersichtliche Darstellung der Issues. Die Farben und Icons für die Schweregrade der Issues sind intuitiv und ermöglichen eine schnelle visuelle Erfassung der Kritikalität. Allerdings fehlt es an visueller Differenzierung zwischen den Zeilen, was das schnelle Scannen der Liste erschwert. Eine stärkere visuelle Trennung, beispielsweise durch abwechselnde Zeilenfarben, könnte die Lesbarkeit verbessern. Ein weiterer Punkt zur Verbesserung ist die Interaktivität: Es fehlen interaktive Elemente oder direkte Aktionsmöglichkeiten in der Listenansicht. Das Hinzufügen von Buttons oder Icons für häufige Aktionen wie das Bestätigen, Eskalieren oder Lösen eines Issues könnte die Benutzerfreundlichkeit steigern. Der Kontrast zwischen Schrift und Hintergrund bei der „Duration“-Spalte weist in einem Kontrastchecker einen schlechten Wert auf. Eine Anpassung des Kontrasts wäre hier notwendig, um die Lesbarkeit zu verbessern.

4.3.2 Verständlichkeit

In Bezug auf die Verständlichkeit bietet die CuriX-Listenansicht eine klare Struktur, bei der die wesentlichen Informationen eines Issues auf einen Blick sichtbar sind. Die Titel der Issues enthalten oft komplizierte technische Ausdrücke, die nicht für alle Benutzer immer verständlich sind. Eine vereinfachte Sprache oder zusätzliche Erklärungen könnten hier die Verständlichkeit erhöhen. Auch die Angaben in der „Location“-Spalte sind häufig komplex und schwer verständlich. Komplizierte technische Standortbeschreibungen könnten durch benutzerfreundlichere Begriffe oder zusätzliche Erklärungen verbessert werden.

4.3.3 Satzschablonen

Bei der Untersuchung der Issues in CuriX wurden folgende Satzschablonen identifiziert:

- **Component is not managed**
 - **Struktur:** [Subjekt] [Prädikat] [Bedingung/Ort/Aktion/Prozess]
 - **Beispiel:** Component is not managed in the system.

- Component ictv0071 not in CuriX

4.4 IBM Maximo

IBM Maximo ist eine Asset-Management-Plattform, die Unternehmen dabei unterstützt, ihre physischen Assets zu überwachen und zu verwalten. Sie bietet Funktionen zur Verwaltung von Wartung, Reparatur und Betrieb sowie zur Verfolgung von Anomalien und Problemen, um die Effizienz und Lebensdauer der Assets zu maximieren.

Asset	Site	Type	Location	Description	Status	Health	Installation Date	Total Cost	Age in Years	Days to failure	Failure pos
000_270200	EUDND	VACUUM_CIRCUIT...	000_270200	000_270200	OPERATING	40	08/10/1978	0.00	46.8		
000_270200	EUDND	AIR_HAUNDRYING...	000_270200	000_270200	OPERATING	37	08/10/1978	0.00	46.8		
000_200277	EUDND	VACUUM_CIRCUIT...	000_200277	000_200277	OPERATING	50	05/20/2004	0.00	20.2		
0T_107476	EUDND	DISTRIBUTION_TR...	0T_107476	Distribution Tr...	OPERATING	40	08/10/1978	0.00	53.2		
0T_107476	EUDND	DISTRIBUTION_TR...	0T_107476	Distribution Tr...	OPERATING	40	08/10/1978	0.00	53.2		
0T_134001	EUDND	DISTRIBUTION_TR...	0T_134001	Distribution Tr...	OPERATING	39	07/09/2004	0.00	23.8		

Abbildung 5: IBM Listenansicht

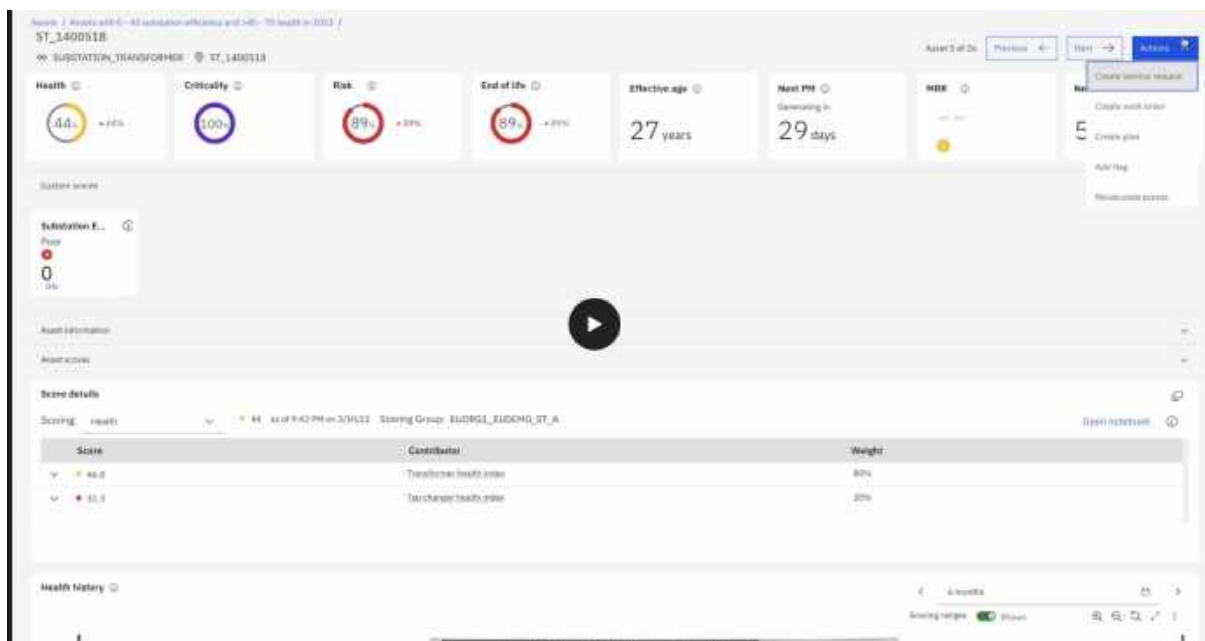


Abbildung 6: IBM Detailansicht

4.4.1 Design

Die Listenansicht von IBM Maximo ist klar und übersichtlich gestaltet. Es wird eine schlichte Farbpalette verwendet, die primär in Grautönen gehalten ist, wobei verschiedene Farben zur Darstellung von Risiko und Kritikalität eingesetzt werden. Die Icons sind intuitiv und tragen zur schnellen Erfassung der wichtigsten Informationen bei. Es fehlen interaktive Elemente oder Hinweise, die dem Benutzer sofort signalisieren, welche Aktionen von dieser Ansicht aus möglich sind. Das Hinzufügen solcher Elemente könnte die Benutzerfreundlichkeit erhöhen.

4.4.2 Verständlichkeit

In Bezug auf die Verständlichkeit bietet die Listenansicht eine klare Struktur und ermöglicht es, wichtige Informationen auf einen Blick zu erfassen. Assetnamen sind jedoch oft unklar und enthalten nur techni-

sche Bezeichnungen wie „BRK_272530“, die für viele Benutzer nicht unmittelbar verständlich sind. Eine klarere Benennung oder zusätzliche Erläuterungen könnten hier Abhilfe schaffen.

4.4.3 Satzschablonen

IBM Maximo ist hauptsächlich für das Asset-Management konzipiert und bietet keine spezifischen Satzschablonen für Meldungen.

4.5 Vergleich der Systeme

4.5.1 Methodik zur Bewertung der Best Practices

Um die verschiedenen Best Practices der analysierten Systeme objektiv zu bewerten, wurde ein einfaches Bewertungssystem verwendet:

- **Ja:** Vollständig vorhanden.
- **Nein:** Nicht vorhanden.
- **Teilweise:** Teilweise vorhanden.
- **- :** Nicht bewertbar.

Die folgende Liste beschreibt die Methoden, die zur Bewertung der einzelnen Best Practices angewendet wurden:

- **Kompakt und übersichtlich:** Horizontal als auch vertikal maximal eine halbe Seite scrollable. Verknüpfte Informationen stehen nahe beieinander.
- **Intuitive Farben:** Farben sind gut unterscheidbar und aus dem Alltag bekannt. Falls Farben verwendet werden, die nicht allgemein bekannt sind, müssen sie durch eine klare Beschreibung ergänzt werden, die ihre Bedeutung erklärt.
- **Intuitive Icons:** Gebräuchliche Icons mit Symbolen aus dem Alltag, welche auf den ersten Blick verstanden werden. Falls nicht, müssen sie durch eine Legende oder Beschreibung ergänzt werden, um ihre Funktion verständlich zu machen.
- **Visuelle Trennung zwischen Zeilen:** Klare visuelle Trennung zwischen den Zeilen in der Listenansicht. Abstände zwischen den Zeilen ausreichend, um eine klare Struktur zu gewährleisten.
- **Klare Hinweise auf interaktive Elemente:** Interaktive Elemente durch Hover-Effekte, Tooltips, Schatten, Formen oder Farben deutlich erkennbar.
- **Visuelle Hierarchie:** Wichtige Elemente fallen als erstes ins Auge, während weniger wichtige Elemente subtiler dargestellt werden.
- **Konsistenz:** Alle Designelemente verwenden eine einheitliche Designsprache, wobei keine Elemente nur einmal in einer spezifischen Form vorkommen.
- **Klare technische Bezeichnungen:** Es sollen möglichst keine oder nur wenige technische, komplizierte Bezeichnungen verwendet werden. Falls solche vorhanden sind, sind sie klar oder Möglichkeit besteht, Informationen dazu abzurufen.
- **Klarheit, Explizitheit und Prägnanz:** Die Länge der Texte sollte auf das Wesentliche beschränkt sein. Einfachheit der Ausdrucksweise im Vordergrund.
- **Kontextuelle Informationen:** Informationen, die den Kontext erklären, sind vorhanden.
- **Actionable:** Nach dem Lesen der Beschreibung oder des Titels muss mindestens ein klarer nächster Schritt erkennbar sein.
- **Informationsüberflutung:** Möglichkeit besteht, bei Bedarf zusätzliche Informationen abzurufen. Ausreichend whitespacing und grosszügige Abstände.

4.5.2 Tabelle Vergleich der Systeme

In der Tabelle 3 werden die drei analysierten Systeme CuriX, Dynatrace und IBM Maximo miteinander verglichen. Die Tabelle ist in die Kategorien Design und Nicht-Design aufgeteilt, wobei jede Kategorie mehrere Best Practices umfasst.

	CuriX	Dynatrace	IBM Maximo
Design			
Kompakt und übersichtlich	Ja	Ja	Ja
Intuitive Farben	Nein	Ja	Ja
Intuitive Icons	-	Ja	Ja
Visuelle Trennung zwischen Zeilen	Nein	Ja	Ja
Klare Hinweise auf interaktive Elemente	Nein	Nein	Nein
Visuelle Hierarchie	Teilweise	Ja	Teilweise
Konsistenz	-	Ja	Ja
Nicht-Design			
Klare technische Bezeichnungen	Teilweise	-	Teilweise
Klarheit und Prägnanz	Teilweise	Teilweise	Ja
Kontextuelle Informationen	Teilweise	Teilweise	Ja
Actionable	Teilweise	Teilweise	Teilweise
Informationsüberflutung	-	Nein	Teilweise
KI-unterstützter Lösungsvorschlag	Nein	Nein	Nein

Tabelle 3: Vergleich der Systeme

4.5.3 Schlussfolgerung des Vergleichs

Die Analyse der drei Systeme zeigt, dass jedes System Stärken und Schwächen in unterschiedlichen Bereichen aufweist. Während Dynatrace und IBM Maximo in Bezug auf das Design konsistenter und intuitiver erscheinen, zeigen alle drei Systeme Defizite bei der Berücksichtigung von Laien als Zielgruppe und in der Umsetzung von Handlungsempfehlungen.

Der Ansatz des Projekts zielt darauf ab, diese Lücken zu schliessen, indem ein System entwickelt wird, das sowohl eine intuitive Benutzeroberfläche als auch kontextsensitive, KI-gestützte Empfehlungen bietet. Zudem wird ein Feedback-Mechanismus integriert, um die Handlungsempfehlungen zu verbessern.

5 Entwicklung eines konzeptuellen Modells für Issues in Systemen

5.1 Attribute von Issues

Da der Fokus auf Issues im Bereich des Systemmonitorings liegt, ist es notwendig, Attribute zu identifizieren, die für diese Domäne relevant sind, damit ein PoC implementiert werden kann. Es wurde ein Klassendiagramm erstellt, welches zeigt, wie ein spezifisches Issue von dem allgemeinem Issue, welches als Grundstruktur dient, erben und erweitern kann.

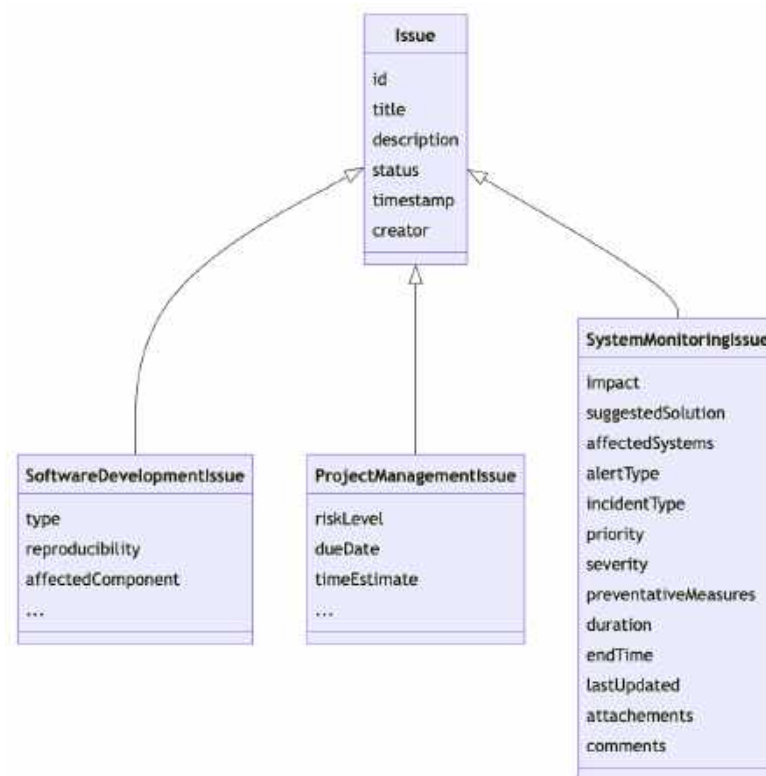


Abbildung 7: Klassendiagramm Issues

Wie in Abbildung 7 dargestellt, dient das allgemeine Issue als Grundlage und enthält Attribute wie Id, Title, Description, Status, Timestamp (Zeitpunkt der Erstellung des Issues) und Creator. Diese Grundstruktur wird für alle weiteren Issues verwendet und bietet eine einheitliche Basis.

Die spezifischen Issues, wie das `SoftwareDevelopmentIssue` und das `ProjectManagementIssue`, dienen als Veranschaulichung dafür, wie Issues in verschiedenen Domänen gestaltet werden können. Diese erben die Attribute vom allgemeinen Issue. Je nach Kontext und Anforderungen besitzen diese spezifischen Issues dann zusätzliche Attribute, die auf ihren jeweiligen Anwendungsbereich zugeschnitten sind.

Die Klasse `SystemMonitoringIssue` unten rechts auf der Abbildung 7 ist die spezifische Art von Issue, auf die sich das Projekt konzentriert. Die Attribute dieses Issues werden hier aufgelistet und erläutert.

- **Id:** Eindeutige Identifikationsnummer.
- **Title:** Der Titel des Issues bietet eine kurze und prägnante Beschreibung des Problems.
- **Description:** Detaillierte Erklärung des Fehlers. Diese Beschreibung enthält alle relevanten Informationen und Hintergründe zum Issue.
- **Status:** Zeigt den aktuellen Bearbeitungsstand des Issues an.
- **Timestamp:** Zeitpunkt der Erstellung des Issues.
- **Creator:** Name des Erstellers des Issues.

- **Impact:** Zeigt die Auswirkungen des Issues auf das System oder die betroffenen Benutzer an.
- **Suggested Solution:** Empfohlene Schritte zur Behebung des Problems. Wird im Bericht oft als „Lösungsvorschlag“ verwendet.
- **Affected Systems:** Listet die Systeme auf, die von dem Issue betroffen sind.
- **Alert Type:** Symbole für schnelle Erkennung der Art des Issues.
- **Incident Type:** Art des Vorfalls, beispielsweise Performance-Probleme, Speicherplatzmangel, Überhitzungen etc.
- **Priority:** Zeigt die Priorität des Issues an, die angibt, wie dringend es gelöst werden muss.
- **Severity:** Gibt den Schweregrad des Issues an.
- **Preventative Measures:** Vorbeugende Massnahmen, die ergriffen werden können, um das Auftreten ähnlicher Issues in der Zukunft zu verhindern.
- **Duration, Endtime, Last Updated:** Dauer, Zeitpunkt des Schliessens und Zeitpunkt der letzten Überarbeitung des Issues.
- **Attachments and Comments:** Möglichkeit um Kommentaren und Anhängen für zusätzliche Informationen hinzuzufügen.

5.2 Lifecycle von Issues

Der Lebenszyklus eines Issues beschreibt die verschiedenen Phasen, die ein Issue von der Erstellung bis zur abschliessenden Lösung durchläuft. Das soll eine strukturierte und nachvollziehbare Bearbeitung von Problemen gewährleisten. Die Phasen des Lifecycles sind wie folgt:

- **New:** Wenn das Issue erstellt wurde, erhält es direkt den Status **New**. Das Problem oder die Anfrage wurde dokumentiert. Dieses wird der Sammlung der neuen Issues hinzugefügt, getrennt von den **Open**-Issues.
- **Open:** Nachdem das Issue von einem Benutzer angeschaut wurde, oder er das Issue als „gelesen“ markiert hat, wird es in die **Open** Status überführt.
- **In Progress:** Zu Beginn der Bearbeitungsphase setzt der User den Status des Issues auf **In Progress**. Hier befindet sich das Issue in aktiver Bearbeitung. Die zuständigen Personen arbeiten daran, das Problem zu lösen.
- **Closed:** Nachdem das Issue vollständig bearbeitet und gelöst wurde, wird es geschlossen. Das Issue wird als **Closed** markiert. Nichtsdestotrotz kann ein Issue wieder in den **In Progress** Status übergeführt werden, falls dies beispielsweise nachbearbeitet werden muss.

Diese vier Phasen sorgen für Nachvollziehbarkeit und Transparenz, damit alle Beteiligten jederzeit den Status und Fortschritt eines Issues logisch nachvollziehen können. Das Modell wurde bewusst einfach gehalten, um die grundlegenden Prozesse zu verdeutlichen. Es kann jedoch bei Bedarf erweitert werden, beispielsweise durch Phasen für die Nachbearbeitung oder andere Zustände. Nachfolgend veranschaulicht Abbildung 8 die Status.



Abbildung 8: Lebenszyklus eines Issues

5.3 Designfaktoren

Die Gestaltung von Issues in Systemen spielt eine entscheidende Rolle für die Benutzerfreundlichkeit und die Effizienz der Problemlösung. In der durchgeführten State-of-the-Art-Analyse wurde bereits herausgearbeitet, wie diese Aspekte auf der Ebene der Recherche betrachtet werden. In diesem Abschnitt wird nun beschrieben, wie das konzeptuelle Modell in Bezug auf die folgenden Designfaktoren aussieht. Designfaktoren betreffen die visuelle und ästhetische Gestaltung. Sie entscheiden, wie eine Nachricht präsentiert, dargestellt und vom Benutzer wahrgenommen wird. Mit den folgenden Punkten wird **RQ1.2**: Welche visuellen Elemente erhöhen die Verständlichkeit von Fehlermeldungen? beantwortet.

- **Grösse und Positionierung**: Die Grösse und Positionierung von Meldungen beeinflusst ihre Sichtbarkeit und Wahrnehmung. Wichtige Informationen werden prominent platziert und in einer ausreichend grossen Schrift dargestellt, um sicherzustellen, dass sie leicht erkennbar sind. Ein Beispiel dafür ist der Titel. Bei der Positionierung von Elementen spielt die Leserichtung eine wesentliche Rolle. Wichtige Informationen sollten zuerst ins Auge fallen, indem sie oben und links platziert werden. Dadurch werden sie in der natürlichen Leserichtung, von oben nach unten und von links nach rechts, als erstes wahrgenommen. Ein Beispiel dafür ist, dass bei Issues der Alert Type zuerst angezeigt wird. Eine klare Trennung zwischen neuen und offenen Issues hilft zusätzlich, die Übersichtlichkeit zu bewahren und sicherzustellen, dass kritische Meldungen nicht übersehen werden.
- **Farbgestaltung**: Die Farbgestaltung spielt eine wesentliche Rolle bei der Kommunikation von Meldungen. Farben sollten verwendet werden, um unterschiedliche Arten von Nachrichten zu unterscheiden (beispielsweise Rot für Fehler, Gelb für Warnungen, Blau für Information). Dabei ist es wichtig, auf ausreichend Kontrast zu achten. Um den Schweregrad eines Issues zu verdeutlichen, werden ebenfalls Farben gebraucht.
- **Icons und Piktogramme**: Icons und Piktogramme können dazu beitragen, Meldungen schneller verständlich zu machen. Sie sollten intuitiv und universell verständlich sein, um die Benutzerfreundlichkeit zu erhöhen. Beispielsweise kann ein rotes Ausrufezeichen sofort als Hinweis auf ein Problem erkannt werden.
- **Konsistenz und Einheitlichkeit**: Konsistenz und Einheitlichkeit in der Darstellung von Meldungen sind entscheidend, um Verwirrung zu vermeiden. Die gleiche Terminologie, Farbgestaltung und Layout sollten im gesamten System verwendet werden, um den Benutzern ein einheitliches Erlebnis zu bieten. Als Hauptdesignelement wurden Cards gewählt.
- **Minimalistisches Design**: Das Design wird möglichst minimalistisch gehalten, nicht nur, weil es ein Proof of Concept ist, sondern auch um die Informationsüberflutung zu vermeiden. Bei Bedarf können zusätzliche Informationen angezeigt werden.
- **Visuelle Hierarchie**: Die visuelle Hierarchie in Meldungen sollte die wichtigste Information hervorheben. Durch die Verwendung von Grössenunterschieden, Leserichtung, Fett- oder Kursivdruck und Abständen können die relevanten Teile der Nachricht betont werden, sodass Benutzer schnell die notwendigen Informationen erfassen können.
- **Auffälligkeit**: Die Auffälligkeit von Issues wird durch die Kombination der vorherigen Faktoren wie Farbgestaltung, visuelle Hierarchie und Platzierung gewährleistet.

5.4 Nicht-Designfaktoren

Nicht-Designfaktoren beziehen sich auf funktionale und inhaltliche Aspekte. Diese sind nicht unmittelbar mit der visuellen Gestaltung in Zusammenhang sondern betreffen die Struktur, den Inhalt und die Benutzerinteraktion mit dem Issue. Die folgenden drei Punkte beantworten: **RQ1.1**: Welche sprachlichen Elemente erhöhen die Verständlichkeit von Fehlermeldungen?

- **Klarheit und Präzision**: Nicht-Designfaktoren wie Klarheit und Präzision in der Formulierung von Meldungen sind entscheidend. Meldungen sollten eindeutig und spezifisch sein, damit Benutzer genau verstehen, was das Problem. Technische Fachbegriffe, die Benutzer möglicherweise nicht verstehen, sollten vermieden werden.

- **Handlungsorientierung:** Meldungen sollten den Benutzern klare Anweisungen geben, wie sie das Problem beheben können. Statt nur auf das Problem hinzuweisen, sollten konkrete Lösungsvorschläge oder nächste Schritte angeboten werden, um die Problemlösung zu erleichtern. Hierzu wird als Hilfe die KI gebraucht.
- **Kontextbezogenheit:** Eine gute Meldung berücksichtigt den aktuellen Kontext des Benutzers und liefert relevante Informationen, die auf die spezifische Situation zugeschnitten sind. Der Kontext ist ebenfalls relevant, damit die KI angemessene Lösungen anbieten kann.
- **Tooltips:** Durch die Bereitstellung von zusätzlichen Informationen und Erklärungen direkt an der Benutzeroberfläche mit Hilfe von Tooltips werden Unklarheiten minimiert und die Benutzererfahrung verbessert.

5.5 Wizard-Modus

Der Wizard-Modus ist für Laien verfügbar, um sie durch die verschiedenen Bereiche eines Issues zu führen und ihnen eine schrittweise Erklärung zu bieten. Befindet sich ein Benutzer im Laien-Modus und wird ein Issue geöffnet, wird der Wizard-Modus automatisch gestartet, welcher den Benutzer durch die einzelnen Komponenten in der Detailansicht des Issues leitet.

Der Wizard-Modus hat eine Navigation, bei der der Benutzer auf „Weiter“ oder „Zurück“ klicken kann, um sich zu bewegen. Auch wird ein Fortschrittsbalken bestehend aus blauen Punkten angezeigt, der den Fortschritt in der Tour visualisiert. Ebenfalls ist in jedem Abschnitt, der gerade erklärt wird, eine Chat-Funktion verfügbar, um der KI Fragen zu stellen. Dieses Feature wird im Kapitel 5.9.3 genauer erklärt. Am oberen Bildschirmrand befindet sich eine Navigationsbar. Mit dieser kann der Benutzer direkt zum gewünschten Tourbereich springen.

Falls der Wizard erkennt, dass im angezeigten Bereich des Issues Informationen fehlen, wie beispielsweise eine unvollständige Beschreibung, erscheint ein Text „Es scheint als ob hier der Text fehlt. Möchtest du, dass die KI diesen Text ergänzt?“. Mit zwei Buttons „Ja“ und „Nein“ kann der Benutzer entscheiden, ob die KI die fehlenden Informationen generieren soll, oder nicht.

5.6 Satzschablonen

Satzschablonen sind vordefinierte Textbausteine, die in Sätzen verwendet werden, um Konsistenz und Klarheit zu gewährleisten. Diese Methode hilft, klare und handlungsorientierte Meldungen zu erstellen, indem sprachliche Elemente standardisiert werden. Dadurch wird die Verständlichkeit von Fehlermeldungen erhöht, was besonders wichtig für Laien ist. Dies beantwortet ebenfalls **RQ1.1**: Welche sprachlichen Elemente erhöhen die Verständlichkeit von Fehlermeldungen?

5.6.1 Satzschablone für den Titel

Die folgende Satzschablone wurde bewusst gewählt, um sicherzustellen, dass alle wichtigen Informationen auf den ersten Blick erkennbar sind.

- **Struktur:** [Subjekt] [Prädikat] [Objekt] [Konjunktion] [Bedingung/Ort/Aktion/Prozess]
- **Beispiel:** Die Datenbankverbindung schlägt fehl unter hoher Last.

Dadurch kann der Benutzer schnell erfassen, worum es in der Meldung geht. Obwohl dies zu längeren Titeln führen kann, wird die Verständlichkeit dadurch verbessert, da kurze Titel oft nicht ausreichend aussagekräftig sind.

5.6.2 Vorlage für die Beschreibung

Die Beschreibung folgt nicht einer Satzschablone, sondern einer Vorlage, die sicherstellt, dass alle wichtigen Punkte abgedeckt werden. Das macht die Beschreibungen verständlicher, besonders für Laien, die klare Informationen benötigen.

- **Einleitung:** Kurze Zusammenfassung des Problems.
- **Hintergrundinformationen:** Detaillierte Beschreibung des Kontexts.
- **Schritte zur Reproduktion:** Anleitung, wie das Problem reproduziert werden kann.

- **Erwartetes Verhalten:** Beschreibung dessen, was eigentlich passieren sollte.
- **Tatsächliches Verhalten:** Beschreibung dessen, was tatsächlich passiert.
- **Zusätzliche Informationen:** Weitere relevante Informationen, die bei der Problemlösung hilfreich sein könnten.

5.7 Ersteller von Issues

In herkömmlichen Systemüberwachungssoftware werden Issues automatisch durch das System selbst erstellt, basierend auf vordefinierten Schwellenwerten und Bedingungen. Die Erstellung von Issues kann durch den Einsatz von KI erweitert und verbessert werden. Die KI nutzt die vorgegebene Satzschablonen, um die erkannten Probleme in eine einheitliche Form zu bringen.

Während der Recherche wurden neben GitLab und GitHub auch verschiedene Ticketing-Systeme untersucht. Aus diesem Grund wurde die manuelle Erstellung von Issues ebenfalls in Betracht gezogen. Das System erlaubt daher auch die manuelle Erstellung von Issues mit Unterstützung durch KI.

5.8 Feedback-Zyklus für Issues

Im Rahmen dieser Arbeit wird untersucht, wie es möglich ist, ein Feedbackmechanismus zu entwickeln, mit welchem der Benutzer verschiedene Aspekte wie Verständlichkeit, Design und Handlungsempfehlungen des Issues bewerten kann. Im Ausblick (Kapitel 9.1) wird erläutert, wie dieses Benutzerfeedback mithilfe von KI möglicherweise genutzt werden kann, um zukünftige Issues zu optimieren. Dabei wird es drei Möglichkeiten geben, um das Issue zu bewerten: im Wizard-Modus, durch das Initiale Feedback und durch das Finale Feedback. In diesen drei verschiedenen Formulare wird es auch unterschiedliche Fragen geben.

Dieses Kapitel bespricht **RQ2.2:** Wann ist der Zeitpunkt für den Benutzer angenehm, um Feedback zu geben? Betrachtet man den Lebenszyklus eines Issues in Abbildung 9, so sieht man, dass ein Issue mit dem Status **New** startet.

Nachdem ein Benutzer ein Issue geöffnet und in den Status **Open** überführt hat, erhält er die Möglichkeit, ein initiales Feedback zu geben. Diese Option steht auch zur Verfügung, wenn das Issue den Status **In Progress** erreicht hat. Der Wizard-Modus kann sowohl bei neuen Issues (**New**), bei Issues mit Status **In Progress**, als auch bei bereits geöffneten Issues (**Open**) gestartet werden. Nach Abschluss des Wizard-Modus besteht ebenfalls die Möglichkeit, ein Feedback abzugeben.

Nun stellt sich die Frage, welche Feedbackfragen gestellt werden sollen, um ein sinnvolles und hilfreiches Feedback zu erhalten. Die Wahl der Fragen ist dabei entscheidend. Sie sollen nicht nur relevante Informationen liefern, sondern auch den Benutzer dazu motivieren, das Feedback bereitwillig zu geben. Um ein sinnvolles Feedbackformular zu gestalten, werden die wichtigsten Attribute des Issues sowie zusätzliche Fragen zur Verständlichkeit abgefragt. In einem initialen Feedback sollen nicht zu viele Fragen gestellt werden, denn zu Beginn eines Issues soll sich der Benutzer mehr um das Problem selbst kümmern, als um ein langes Feedback zu geben. Deshalb sind auch nicht alle Fragen Pflichtfelder. Dennoch sind die gestellten Fragen wichtig, um das KI-Modell zu trainieren und zu verbessern. So soll die Motivation mit Hintergedanke auf das C-HIP Modell in Abbildung 1 hoch bleiben.

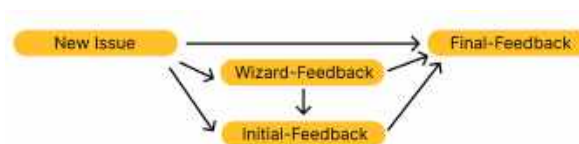


Abbildung 9: Feedbackzyklus

5.8.1 Wizard-Feedback

Im Wizard-Modus wird ein Laie durch die Detailansicht des Issues geführt. Nach Abschluss des Wizard-Modus werden drei wichtige Fragen gestellt. Diese Fragen sind Pflicht. Sie decken wesentliche Bereiche ab, die auch im Initial-Feedback enthalten sind. Allerdings ist das Initial-Feedback freiwillig, weshalb diese im Wizard-Modus obligatorisch sind. Sollten diese Fragen im Wizard-Modus beantwortet werden, werden sie im Initial-Feedback übersprungen. Die drei Fragen sind von Relevanz, weil sie helfen, folgende

Punkte zu klären: Klarheit des Issues, Verständnis des Problems und Bewusstsein der erforderlichen Schritte.

Jede Frage umfasst in diesem Dokument die Bereiche „Antwortmöglichkeiten“, „Zusätzliche Aktion“, „Ziel“ und „Warum erforderlich?“ oder „Warum optional?“. Im PoC wird lediglich die Antwortmöglichkeit dargestellt. Zudem steht jeweils ein (R) für Required oder ein (O) für optional nach der Frage.

Die Fragen befinden sich im Anhang unter Kapitel 10.

5.8.2 Initiales Feedback

Warum soll ein Benutzer ein initiales Feedback geben? Es ermöglicht, frühzeitig die ersten Eindrücke mitzuteilen, um Missverständnisse zu identifizieren und zu klären. Findet der Feedbackgeber also, dass etwas nicht verständlich ist, so erscheint ein Textfeld, wo seine Meinung geäußert werden kann. Nachfolgend kommen sieben Fragen, die im initialen Feedbackformular enthalten sind. Das initiale Feedback ist nicht Pflicht, um den Benutzer nicht zu überfordern oder zu belästigen. Wie erwähnt, erscheinen die ersten drei Fragen nur, falls im Wizard-Modus kein Feedback gegeben wurde.

5.8.3 Finales Feedback

Ist ein Issue fertig bearbeitet, so kann ein Benutzer das Issue schliessen, um es von **In Progress** auf **Closed** zu setzen. Wie man in der Abbildung 9 sieht, erscheint nach **In Progress** noch das finale Feedback. Das finale Feedback wird also dann ausgefüllt, wenn man das Issue schliessen möchte. Das Ziel des finalen Feedbacks ist es, eine detaillierte Rückmeldung zum gesamten Issue zu erhalten. Da das initiale Feedback optional ist und das Feedback im Wizard-Modus nur ein Laie abgeben muss, wird mit dem finalen Feedback sichergestellt, dass ein Feedback vorhanden ist. Fragen zu Beschreibung, Wissen zum Ablauf zur Lösung, Priorität und Severity werden nur aufgelistet, falls diese im initialen Feedback nicht beantwortet wurden. Nachfolgend kommen 13 Fragen, die im finalen Feedbackformular enthalten sind.

5.9 Anwendung von KI zur Verbesserung der Issues

Die Anwendung von KI im Bereich der Meldungen ist auch eine Fragestellung dieser Arbeit. Ziel ist es, das Potenzial von KI in der Verbesserung von Issues zu untersuchen und ein Konzept zu entwickeln, das darstellt, wie Lösungswege zur Behebung von Problemen vorgeschlagen werden können und wie dem Benutzer geholfen werden kann.

5.9.1 Hilfe bei der Erstellung von Issues

Es wurde untersucht, wie dem Benutzer bei der Erstellung eines neuen Issues geholfen werden kann. Die Idee stammt von GitHub Copilot, einem KI-gestützten Tool, das Entwicklern hilft, Code schneller und effizienter zu schreiben, in dem es dem Entwickler eine Code-Vervollständigung anbietet, welche er dann akzeptieren kann. GitHub Copilot verwendet hierfür maschinelles Lernen und der erzeugte Vorschlag basiert auf dem Kontext der vorherigen Eingabe des Benutzers. Diese Idee kann auf die Erstellung von Issues übertragen werden. Der Ablauf ist wie folgt:

1. Ein Benutzer möchte ein neues Issue erstellen und gibt den Titel möglichst nach Satzschablone an.
2. Darauf erscheint im Textfeld für das Attribut Description ein Vorschlag, basierend auf dem Kontext des Titels.
3. Diesen Vorschlag kann der Benutzer begutachten und akzeptieren. Im Nachhinein kann man den akzeptierten Text noch bearbeiten.

Diese Methode soll vor allem Laien helfen. Aber auch generell soll es ermöglichen, konsistente und genug detaillierte Issues zu erstellen.

5.9.2 Integration von Terminal

Eine Idee wurde entwickelt, wie man ein Terminal sinnvoll in die Detailansicht eines Issues integrieren könnte. Die KI soll somit nicht nur bei der Unterstützung, sondern auch direkt zur Lösung von Problemen einsetzbar sein. Bei Issues, die keine physische Interaktion erfordern, wird in der Detailansicht ein

Terminal angezeigt. In diesem Terminal werden dem Benutzer die Befehle angezeigt, in der Reihenfolge wie diese ausgeführt werden sollen, um das Problem zu beheben. Es ist möglich, diese Befehle direkt in diese Ansicht auszuführen oder zu überspringen.

Dies, sowie auch das nachfolgende Unterkapitel beantwortet **RQ3.3**: Wie können die Vorschläge zur Problemlösung der KI benutzerfreundlich und verständlich präsentiert werden?

5.9.3 Integration von Chat-Funktion

Um den Support und die Benutzerfreundlichkeit vor allem für Laien zu verbessern, wurde eine Chat-Funktion in die Detailansicht integriert. Dies soll als interaktiver Chatbot dienen, um dem Laien, aber auch dem Experten, bei der Lösung ihrer Probleme zu unterstützen. Beim Chatten werden nicht nur die vom Benutzer gesendeten Nachrichten, sondern auch alle relevanten Kontextinformationen des Issues an die KI übermittelt. Dadurch kann die KI präzise und hilfreiche Antworten geben, die genau auf das aktuelle Problem zugeschnitten sind. Diese Funktion kann beispielsweise nützlich sein, wenn kryptische oder unbekannte Gerätenamen in den Meldungen erscheinen. Benutzer können nachfragen und erhalten von der KI Erklärungen oder weiterführende Informationen, die helfen, das Problem besser zu verstehen.

5.9.4 KI-gestützte Analyse und Optimierung

Ein wesentlicher Bestandteil der Anwendung von KI zur Verbesserung der Fehlermeldungen ist die kontinuierliche Optimierung des KI-Modells durch das gegebene Feedback. Sobald ein Benutzer Feedback zur Vollständigkeit und Klarheit der bereitgestellten Informationen gibt, wird dieses Feedback genutzt, um das KI-Modell zu verfeinern. Die Idee ist also auch, dass das KI-Modell lernt, welche Informationen häufig fehlen oder unklar sind.

Dies beantwortet **RQ3.1**: Wie können Daten gesammelt werden, damit das Modell Issues und Benutzerreaktionen erfasst, um zukünftig Lösungsvorschläge anzubieten? so wie auch **RQ3.2**: Wie können Daten gesammelt werden, damit das Modell den Kontext des Problems lernen kann?

5.9.5 KI als Zwischenlayer

Um die Konsistenz und Einheitlichkeit der Issues sicherzustellen, wurde überlegt, wie man mit Hilfe von Large Language Model, kurz LLM, stets konsistente Issues erstellen kann. Ein LLM ist ein maschinelles Lernmodell, das auf umfangreichen Textdaten trainiert wurde und in der Lage ist, natürliche Sprache zu verstehen und zu generieren. Dieses LLM fungiert als Filter in einem Zwischenlayer, nämlich zwischen dem System, welches das Issue erstellt oder generiert, und dem Dashboard, welches die Issues dann anzeigt. Wenn ein Issue vom System kommt, wird es zunächst an das LLM übergeben. Das LLM bearbeitet den Titel und die Beschreibung des Issues nach vorgegebenen Satzschablonen, um die Einheitlichkeit und Verständlichkeit sicherzustellen. Darüber hinaus kann das LLM technischer Jargon übersetzen, sodass auch Laien die Informationen verstehen. Erst nach dieser Bearbeitung wird das Issue im Dashboard angezeigt.

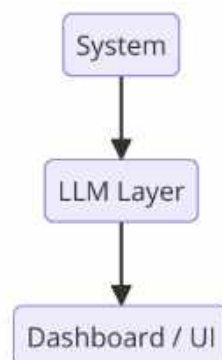


Abbildung 10: LLM als Zwischenlayer

5.10 Wireframes

Bereits früh im Projektverlauf wurden Wireframes entwickelt, um mögliche Richtungen und Gestaltungsmöglichkeiten der Anwendung zu erkennen. Diese visuellen Entwürfe halfen dabei, verschiedene Ideen zu testen. Durch das Erstellen von Wireframes konnten die Konzepte schnell und unkompliziert visualisiert werden, bevor sie in die Implementierung übergingen. Hierbei wurde beachtet, die ersten erkannten Best Practices und das Wissen aus der State-of-the-Art-Analyse zu integrieren. Die ersten Wireframes stellten die grundlegende Struktur und Navigation der Anwendung dar.

Der Schritt der Validierung der Wireframes wurde jedoch übersprungen um direkt mit der Umsetzung des PoC zu beginnen. Dieser Ansatz wurde gewählt, um reale Benutzererfahrungen zu sammeln, die Einblicke in die Anwendung der entwickelten Lösungen bieten.

5.10.1 Listenansicht

Die Listenansicht ist die erste Ansicht, die ein Benutzer sieht. Die ursprüngliche Idee war es, von einer Liste wegzukommen, da dies tatsächlich bei allen analysierten Systemen Standard ist. Jedoch hat sich schnell ergeben, dass diese Methode sehr bewährt ist und von allen Benutzern leicht verstanden wird, weshalb nicht ganz davon losgelassen wurde. Daher wurde eine Mischung ausgearbeitet, welche den Benutzer nicht an Information überwältigen soll, jedoch genug Informationen und Handlungsmöglichkeiten gibt.

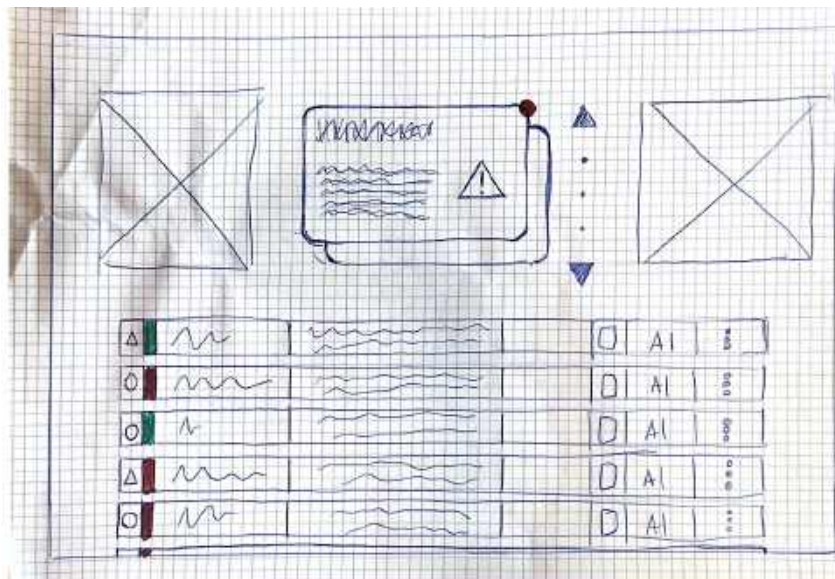


Abbildung 11: Listenansicht Wireframe

Designüberlegungen:

- **Klarheit und Struktur:** Die Listenansicht wurde so gestaltet, dass die wichtigsten Informationen auf einen Blick erfasst werden können. Im unteren Bereich befindet sich eine Liste, die alle Issues anzeigt, während im oberen Bereich sich die neusten Issues befinden. Hier sind die Issues in Cards gestaltet. Sind mehrere neue Issues vorhanden, so sind diese Cards übereinander gestapelt. Diese Cards sind somit dann auch scrollbar durch seitliche Pfeile.
- **Farben:** Farbe wurde eingesetzt, um zu sehen, wie und wo man diese integrieren kann. Die Farben wurden bei dem Alert Type Icon (Info, Warning, Error) in der Listenansicht beigelegt, wobei sich hier die Farben auf rot und grün begrenzen. Das lenkt die Aufmerksamkeit des Benutzers zu Beginn auf die wichtigen Elemente.
- **Icons und Symbole:** Icons sind wichtig, dies hat auch die Recherche gezeigt. Deshalb wurden diese wo möglich und wo es sinnvoll ist angewendet. In der Card im oberen Bereich ist der Alert Type als Icon, so wie auch in der unteren Liste deutlich erkennbar.

- **Handlungsorientierung:** Die Listenansicht ist so konzipiert, dass sie klare Handlungsanweisungen bietet. Das 3-Punkte Symbol soll dem Benutzer zeigen, dass er bereits hier erste Handlungen vornehmen kann. Auch ist die Idee, schon hier eine mögliche KI Aktion ausführen zu lassen.

5.10.2 Detailansicht

Die Detailansicht zeigt alle relevanten Informationen so wie Handlungsempfehlungen. Das Ziel bei dem Konzept war es, ebenfalls die Informationsüberflutung zu vermeiden und mit Farben so wie Icons zu arbeiten, um dem Benutzer helfen, schnell das Wichtigste zu erkennen.

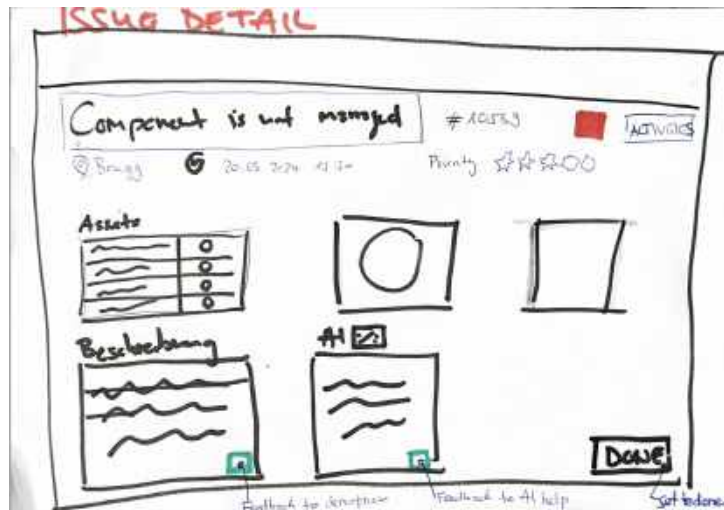


Abbildung 12: Detailansicht Wireframe 1

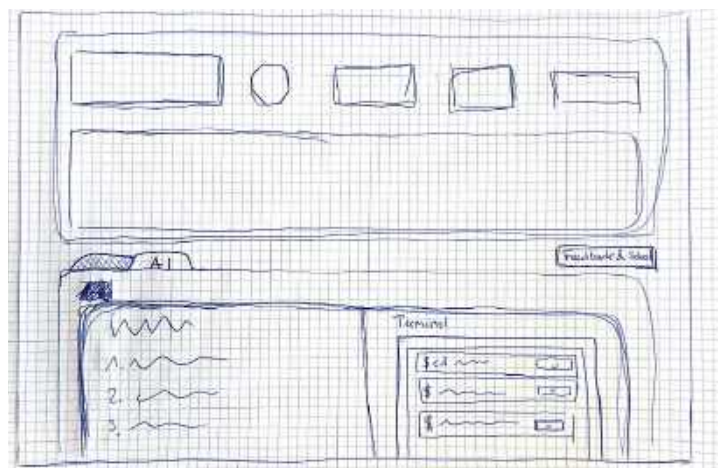


Abbildung 13: Detailansicht Wireframe 2

Designüberlegungen:

- **Informationshierarchie:** Die Detailansicht wurde so gestaltet, dass die wichtigsten Informationen in Leserichtung zuerst angezeigt werden. Im oberen Bereich befinden sich Titel sowie kleinere, aber wesentliche Informationen wie Severity und Alert Type. Weitere Attribute, wie beispielsweise die Beschreibung des Issues, sind in der Mitte der Ansicht in separaten Cards übersichtlich angeordnet. Der „Done“ Button, um das Issue zu schliessen, befindet sich unten rechts, während ein weiterer Button für zusätzliche Aktionen oben rechts positioniert ist.
- **Actionability:** Die Handlungsanweisungen sollen dem Benutzer klar und verständlich präsentiert werden, sodass alle notwendigen Informationen zur Lösung des Issues direkt verfügbar sind. In Abbildung 13 sieht man eine Idee, die entwickelt wurde, um dem Benutzer zu helfen, direkt in

einem integriertem Terminal ein Problem zu beheben. Deshalb befindet sich im unteren Teil der Detailansicht ein Tab mit der Bezeichnung „AI,, in dem die Schritte zur Lösung des Problems auf der linken Seite angezeigt werden und rechts daneben befindet sich ein Terminal, in dem die vorgeschlagenen Befehle direkt ausgeführt werden können. Auch der Lösungsvorschlag soll schriftlich hier in einer Card sichtbar sein.

- **Feedbackmechanismus:** Die Idee für den Feedbackmechanismus ist in der Detailansicht noch nicht stark ausgeprägt sichtbar. In Abbildung 12 sind zwei grüne Feedback-Buttons zu sehen, welche erlauben, bereits hier Feedback über die Beschreibung und den KI-Vorschlag zu geben.

5.10.3 Feedbackansicht

Die Feedbackansicht ermöglicht es dem Benutzer, Rückmeldungen zum Issue zu geben. Diese Ansicht erscheint, wenn man das Issue schliessen möchte. In der Abbildung 15 wurde das Feedbackformular als Pop-up gestaltet, also so, dass man die Detailansicht im Hintergrund noch sieht. Die beiden anderen Abbildungen 14 und 16 zeigen das Formular auf einer eigenen Seite.

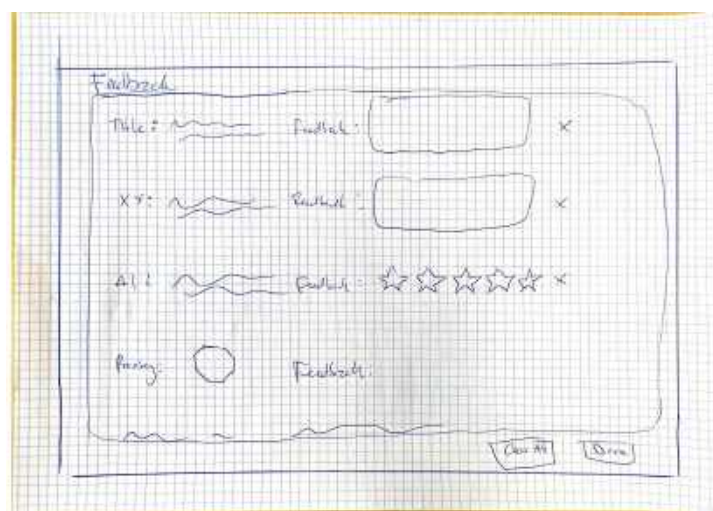


Abbildung 14: Feedbackansicht Wireframe 1

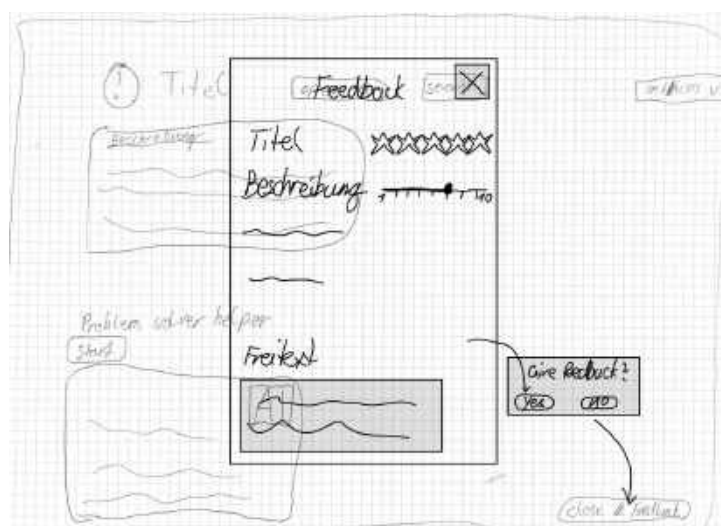


Abbildung 15: Feedbackansicht Wireframe 2



Abbildung 16: Feedbackansicht Wireframe 3

Designüberlegungen:

- **Einfachheit und Zugänglichkeit:** Das Feedbackformular wurde so gestaltet, dass es leicht zu verstehen und einfach auszufüllen ist. Dies soll die Hürde für die Benutzer senken, Feedback zu geben. Deshalb wurde das Formular so gestaltet, dass auf der linken Seite jeweils das zu bewertende Element angezeigt wird und rechts die entsprechende Bewertungsmethode.
- **Feedbackfragen:** Die konkreten Fragen im Feedbackformular wurden noch nicht vollständig ausgearbeitet, jedoch bestand die erste Idee darin, bestimmte Attribute abzufragen. Weitere Fragen helfen dabei, spezifisches und hilfreiches Feedback zu sammeln.
- **Antwortmöglichkeiten:** In den Abbildungen 14 und 15 sieht man erste Varianten, wie Feedback gegeben werden kann. Zur Bewertung wurden Slider, Sterne und Textfelder verwendet.

Diese Designüberlegungen sind die ersten Schritte, um **RQ2.3:** Wie kann das Feedback geben für den Benutzer visuell ansprechend gemacht werden und nicht aufdringlich? zu beantworten. Dies wird im Kapitel 6 weiter besprochen.

6 Proof of Concept für das entwickelte Modell

In diesem Kapitel wird die entwickelte Systemüberwachungssoftware anhand eines Proof of Concept vorgestellt. Das Proof of Concept dient dazu, die Funktionsweise und die verschiedenen Komponenten des Systems zu veranschaulichen. Die implementierten Funktionen basieren auf den zuvor entwickelten Wireframes, die als visuelle Entwürfe gedient haben.

6.1 Technologie-Stack

Für die Entwicklung des Proof of Concept wurde ein Technologie-Stack gewählt, der die Anforderungen an Flexibilität, Skalierbarkeit und Benutzerfreundlichkeit erfüllt. Der gewählte Stack umfasst:

- **Next.js:** Als React-basiertes Framework bietet Next.js serverseitiges Rendering, welches die Performance verbessert. Es ist geeignet für den Aufbau von Single Page Applications (SPAs) und ermöglicht die einfache Implementierung von Routing und API-Endpunkten.
- **React:** React wurde verwendet, um die Benutzeroberfläche zu erstellen. Es ermöglicht die Erstellung von wiederverwendbaren UI-Komponenten und sorgt für eine effiziente Darstellung und Aktualisierung der Benutzeroberfläche.
- **Tailwind CSS:** Tailwind CSS wurde für die Gestaltung der Benutzeroberfläche genutzt. Es bietet eine Vielzahl von vorgefertigten Klassen für Styling und Layout, was die Entwicklung einer konsistenten Benutzeroberfläche beschleunigt.
- **TypeScript:** TypeScript wurde anstelle von JavaScript verwendet, um die Codequalität zu verbessern. Durch statische Typisierung werden Fehler bereits zur Entwicklungszeit erkannt, was zu einem stabileren und wartbaren Code führt.
- **JSON:** Die Issues werden in JSON-Format gespeichert und abgerufen.
- **LocalStorage:** LocalStorage wird verwendet, um Benutzerdaten und Einstellungen lokal im Browser zu speichern. Dies ermöglicht es, den aktuellen Status und die Präferenzen des Benutzers zwischen den Sitzungen beizubehalten, ohne dass eine serverseitige Speicherung erforderlich ist.

6.2 Listenansicht

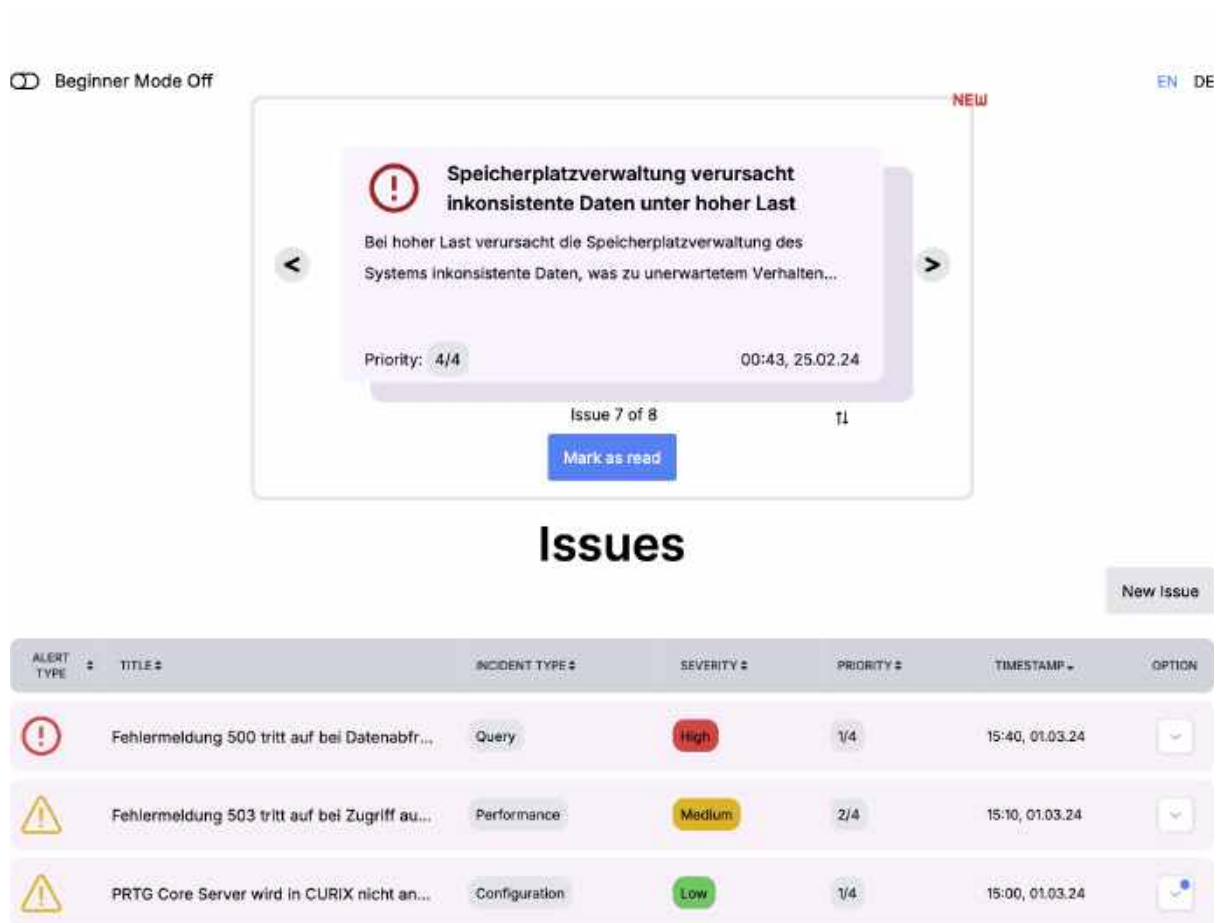


Abbildung 17: Listenansicht

Abbildung 17 stellt das zentrale Dashboard der Systemüberwachungssoftware dar, die darauf abzielt, eine klare und intuitive Übersicht über alle offenen Issues zu bieten. Die Benutzeroberfläche ist in zwei Hauptabschnitte unterteilt: eine Vorschau neuer Issues und eine tabellarische Übersicht aller Issues.

Vorschau neuer Issues

- **Alert Type-Icon:** Ein auffälliges Icon signalisiert den Alert Typ und ermöglicht eine schnelle visuelle Erkennung des Schweregrades des Problems.
- **Issue Title und Description:** Der Titel des Issues wird hervorgehoben, gefolgt von der Einleitung der Beschreibung, die den Kern des Problems erläutert.
- **New-Schriftzug:** Oben rechts in der Vorschau befindet sich ein roter Schriftzug mit der Aufschrift „NEW“. Dieser Schriftzug symbolisiert, dass es sich bei diesen Issues um neue Issues mit dem Status **New** handelt.
- **Priority und Timestamp:** Die Priorität des Issues wird klar angezeigt, zusammen mit einem Zeitstempel, der die genaue Uhrzeit und das Datum des Auftretens des Problems angibt.
- **Navigations-Buttons:** Links und rechts von der Issue-Vorschau befinden sich Navigationspfeile, mit denen Benutzer zwischen verschiedenen neuen Issues blättern können. Standardmässig wird das neuste Issue zuerst angezeigt, und die Issues sind entsprechend ihrer Aktualität sortiert.
- **Mark as read-Button:** Ein Button „Mark as read“ ermöglicht es den Benutzern, das Issue als gelesen zu markieren und den Status zu ändern. Somit wird das Issue in die tabellarische Ansicht übernommen.

- **Sortierfunktion:** Mit dem Pfeilsymbol rechts vom Mark as read-Button können die Issues nach verschiedenen Kriterien wie Alert Type, Priorität oder Zeitstempel sortiert werden. Dadurch wird eine individuelle und übersichtliche Darstellung der Issues ermöglicht.
- **Toggle-Button:** Mit dem Toggle-Button auf der oberen linken Seite kann man in den Laien-Modus wechseln, welcher den Wizard-Modus verfügbar macht.
- **Sprachwahl:** Auf der oberen rechten Seite kann man zwischen der deutschen und englischen Sprache wechseln. Dies verändert nur die Systemsprache und nicht die Inhalte der Issues.

Tabellarische Übersicht aller Issues

- **Alert Type-Icon:** Zeigt den Alert Type an, der durch ein passendes Icon visualisiert wird.
- **Title:** Der Titel des Issues.
- **Incident Type:** Klassifiziert das Problem nach Typen wie Storage, Performance, oder Configuration.
- **Severity:** Zeigt den Schweregrad des Problems an, farblich kodiert (rot für „Hoch“, gelb für „Mittel“, grün für „Niedrig“).
- **Priority:** Gibt die Priorität des Issues an, was die Dringlichkeit signalisiert.
- **Timestamp:** Zeigt das Datum und die Uhrzeit des Auftretens des Problems an.
- **Option:** Ein Dropdown-Menü (siehe Abbildung 18), das ermöglicht initiales Feedback zu geben oder direkt ein Ticket zu öffnen.
- **Sortierfunktion:** Die Tabelle ermöglicht es, die Issues durch Klicken auf die Spaltenüberschriften nach verschiedenen Kriterien zu sortieren. Dies umfasst den Alert Type, Title, Incident Type, Severity, Priority und Timestamp
- **New Issue-Button:** Auf der rechten Seite befindet sich ein Button, mit dem Benutzer ein neues Issue erstellen können.

Das Design der Seite setzt auf eine klare visuelle Hierarchie und eine intuitive Benutzerführung. Die wichtigsten Informationen werden prominent angezeigt, während zusätzliche Details leicht zugänglich sind. Die Farbcodierung der verschiedenen Schweregrade und Alert Types erleichtert die schnelle Identifikation kritischer Probleme.

Besonders wurde auch auf die Leserichtung geachtet, sodass die Informationen von oben nach unten und links nach rechts erfasst werden können. Am Ende jeder Zeile ermöglicht ein Button direkte Aktionen, wodurch die Handhabung der Issues effizienter gestaltet wird.

Eine ursprünglich geplante KI-Funktion, wie sie im Wireframe dargestellt ist, wurde vorerst nicht implementiert. Dies liegt daran, dass noch nicht alle relevanten Informationen über ein Issue vorliegen und Benutzer zusätzliche Informationen benötigen, um fundierte Entscheidungen treffen zu können.

Zwischen den einzelnen Issues in der Listenansicht wurde Abstand gehalten, damit diese visuell gut unterschieden werden können. Der Mark as read-Button ist blau, da diese Farbe typischerweise mit primären Aktionen assoziiert wird. Als Hauptfarbe wurde ein helles Lila oder Lavendel gewählt, welches sich gut mit Weiss kombinieren lässt.

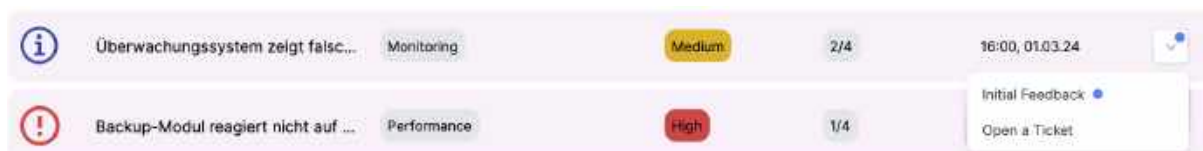


Abbildung 18: Ausführbare Aktionen



Abbildung 19: Tooltip

Abbildung 19 ist ein Beispiel für die Tooltips, die im System verwendet werden.

Beginner Mode Off EN DE

Issues

[New Issue](#)

ALERT TYPE	TITLE	INCIDENT TYPE	SEVERITY	PRIORITY	TIMESTAMP	OPTION
	Überwachungssystem zeigt falsche Alar...	Monitoring	Medium	2/4	16:00, 01.03.24	
	Backup-Modul reagiert nicht auf Fehler b...	Performance	High	1/4	15:50, 01.03.24	
	Fehlermeldung 500 tritt auf bei Datenabr...	Query	High	1/4	15:40, 01.03.24	
	Datenbankmodul verursacht Verbindungs...	Connection	Medium	2/4	15:30, 01.03.24	
	Speicherkomponente zeigt Fehlverhalten ...	Data Integrity	High	1/4	15:20, 01.03.24	
	Fehlermeldung 503 tritt auf bei Zugriff au...	Performance	Medium	2/4	15:10, 01.03.24	
	PRTG Core Server wird in CURIX nicht an...	Configuration	Low	1/4	15:00, 01.03.24	
	Server CT-20 verursacht Ausfall bei Netz...	Performance	High	1/4	15:00, 01.03.24	
	Component PRTG Core Server not in CURIX	Configuration	Low	1/4	15:00, 01.03.24	
	Speicherplatzverwaltung verursacht inko...	Storage	Medium	4/4	00:43, 25.02.24	
	Speicherplatzverwaltung Problem	Storage	Medium	4/4	00:43, 25.02.24	

Abbildung 20: Listenansicht ohne neue Issues

Die Abbildung 20 zeigt die Listenansicht aller Issues wenn keine neuen Issues vorhanden sind.

6.3 Detailansicht aufgeklappt

The screenshot shows the detail view of an issue titled "PRTG Core Server wird in CURIX nicht angezeigt". The interface includes a header with navigation icons, a title bar, and a metadata row. The main content is organized into several panels: Description, Suggested Solution, Affected Systems, Impact, Preventative Measures, Comments, Attachments, and Info. The Description panel provides a detailed explanation of the problem, including background information, reproduction steps, expected vs. actual behavior, and additional information. The Suggested Solution panel offers a clear instruction to check the CURIX configuration. The Affected Systems panel lists "DevelopmentServer-01" and "ERPSYSTEM-01". The Impact panel states that system monitoring reports cannot be generated. The Preventative Measures panel indicates that no preventive actions are required. The Info panel provides metadata such as Creator (User2), Issue Nr. (11), Duration (5 h), Timestamp (15:00, 01.03.24), Last updated (15:05, 01.03.24), and End time (---). A "Feedback and Close" button is located at the bottom right.

PRTG Core Server wird in CURIX nicht angezeigt

Alert Type: Severity: **Low** Status: **Open** Incident Type: Configuration Priority: 1/4

Description

Eineleitung
Beim Versuch, den PRTG Core Server in CURIX zu finden, wird dieser nicht angezeigt.

Hintergrundinformationen
Dieses Problem wurde von User2 gemeldet, der versuchte, den PRTG Core Server in der CURIX-Konfiguration zu finden.

Schritte zur Reproduktion
1. Öffnen Sie die CURIX-Konfigurationsseite.
2. Suchen Sie nach dem PRTG Core Server.
3. Überprüfen Sie die Liste der angezeigten Komponenten.

Erwartetes Verhalten
Der PRTG Core Server sollte in der CURIX-Konfigurationsliste angezeigt werden.

Tatsächliches Verhalten
Der PRTG Core Server wird in der CURIX-Konfigurationsliste nicht angezeigt.

Zusätzliche Informationen
- Log-Datei beigefügt
- Tritt auf in Version 3.4.1
- Keine Auswirkungen auf andere Systeme festgestellt.

Suggested Solution

Lösungsvorschlag Überprüfen Sie die CURIX-Konfiguration und stellen Sie sicher, dass der PRTG Core Server korrekt konfiguriert und hinzugefügt wurde.

Affected Systems

- DevelopmentServer-01
- ERPSYSTEM-01

Impact

Systemüberwachungsberichte können nicht generiert werden, was zu einem Mangel an Überwachung der Netzwerkperformance führt

Preventative Measures

Keine vorbeugenden Massnahmen erforderlich

Comments **Attachments**

Enter your comments here

Info

Creator:	User2
Issue Nr.:	11
Duration:	5 h
Timestamp:	15:00, 01.03.24
Last updated:	15:05, 01.03.24
End time:	---

Feedback and Close

Abbildung 21: Detailansicht

Abbildung 21 zeigt die Detailansicht eines spezifischen Issues. Alle relevanten Informationen und Details werden übersichtlich dargestellt. Zusätzlich zu den bereits erwähnten Informationen in der Listenansicht wie Title, Alert Type, Severity, Incident Typ, Priority und Timestamp umfasst diese Ansicht auch:

- **Description:** Detaillierte Erklärung des Fehlers. Diese Beschreibung enthält alle relevanten Informationen und Hintergründe zum Issue.
- **Affected Systems:** Listet die Systeme auf, die von dem Issue betroffen sind.
- **Suggested Solution:** Empfohlene Schritte zur Behebung des Problems.
- **Impact:** Beschreibt die Auswirkungen des Issues auf das System oder die betroffenen Benutzer.
- **Status:** Zeigt den aktuellen Bearbeitungsstand des Issues an.
- **Preventative Measures:** Massnahmen, um das Auftreten ähnlicher Probleme in der Zukunft zu verhindern.
- **Comments and Attachments:** Möglichkeit um Kommentaren und Anhängen für zusätzliche Informationen hinzuzufügen.

- **Info:** Zusätzliche Informationen zur Meldung, wie Ersteller, Id, Dauer, Zeitstempel, letzte Aktualisierung und Endzeit.
- **Edit-Button:** Ermöglicht das Bearbeiten des Issues.
- **Chat-Button:** Ermöglicht das Chatten mit der KI.
- **Initial-Feedback-Button:** Ermöglicht ein initiales Feedback zu geben.
- **Terminal-Button:** Ermöglicht das Ausführen von der KI empfohlenen Befehle.
- **Collapse-Button:** Ermöglicht das ein- und aufklappen der dritten Spalte der Detailansicht.
- **Feedback and Close-Button:** Ermöglicht ein finales Feedback zu geben und das Schliessen des Issues.
- **Schwarzes Fragezeichen-Icon:** Oben rechts auf jeder Card befindet sich ein schwarzes Fragezeichen-Icon. Wenn man mit der Maus darüber hovers, erscheint ein Tooltip, welches die Card beschreibt.

Die Auswahl von Cards zur Darstellung der verschiedenen Informationen innerhalb der Detailansicht wurde bewusst getroffen, um klare visuelle Abtrennungen zu schaffen. Jede Card stellt eine spezifische Informationseinheit dar, wie betroffene Systeme, Lösungsvorschläge oder Auswirkungen. Dies hilft Benutzern, die Informationen schnell zu erfassen und logisch zu verarbeiten. Die Cards-Struktur ermöglicht es zudem, die Informationen modular und flexibel anzuordnen, wodurch eine klare und übersichtliche Darstellung gewährleistet wird.

Dies beantwortet **RQ1.3:** Wie kann Kontextinformation effizient in Fehlermeldungen integriert werden?

Die Detailansicht verfügt über eine Ein- und Aufklappfunktion (Collapse-Button), die es ermöglicht, die drei Cards auf der rechten Seite bei Bedarf ein- und aufzuklappen. Diese Cards sind beim Aufrufen der Detailansicht standardmässig eingeklappt. Dies trägt dazu bei, dass eine visuelle Überladung vermieden wird. Diese Funktion stellt sicher, dass sowohl Experten als auch Laie, nicht überfordert werden, wenn sie die Seite öffnen. Abbildung 22 zeigt die eingeklappte Detailansicht.

Die Verwendung von Cards fügt sich auch in das Design der Listenansicht ein, wodurch ein konsistentes Benutzererlebnis gewährleistet wird. Die Farbgebung der Cards entspricht der der Listenansicht, was zur visuellen Einheitlichkeit beiträgt und dem Benutzer ein vertrautes Gefühl vermittelt. Diese Struktur wurde von den Wireframes übernommen und im finalen Design beibehalten und weiter ausgearbeitet.

←

📄
🔍
Edit
Give Initial Feedback
📄

PRTG Core Server wird in CURIX nicht angezeigt

Alert Type: ⚠️
Severity: Low
Status: Open
Incident Type: Configuration
Priority: 1/4

Description

Einleitung
Beim Versuch, den PRTG Core Server in CURIX zu finden, wird dieser nicht angezeigt.

Hintergrundinformationen
Dieses Problem wurde von User2 gemeldet, der versuchte, den PRTG Core Server in der CURIX-Konfiguration zu finden.

Schritte zur Reproduktion
1. Öffnen Sie die CURIX-Konfigurationsseite.
2. Suchen Sie nach dem PRTG Core Server.
3. Überprüfen Sie die Liste der angezeigten Komponenten.

Erwartetes Verhalten
Der PRTG Core Server sollte in der CURIX-Konfigurationsliste angezeigt werden.

Tatsächliches Verhalten
Der PRTG Core Server wird in der CURIX-Konfigurationsliste nicht angezeigt.

Zusätzliche Informationen
- Log-Datei beigefügt
- Tritt auf in Version 3.4.1
- Keine Auswirkungen auf andere Systeme festgestellt.

Suggested Solution

Lösungsvorschlag Überprüfen Sie die CURIX-Konfiguration und stellen Sie sicher, dass der PRTG Core Server korrekt konfiguriert und hinzugefügt wurde.

Comments
Attachments

Enter your comments here

Feedback and Close

Abbildung 22: Detailansicht eingeklappt

6.4 Bearbeitung Detailansicht

← Cancel Save [Print]

Title
PRTG Core Server wird in CURIX nicht angezeigt

Alert Type: Warning - Severity: Low - Status: Open - Incident Type: Configuration - Priority: T: Low -

Description
Einleitung
Beim Versuch, den PRTG Core Server in CURIX zu

Suggested Solution
Lösungsvorschlag
Überprüfen Sie die CURIX-Konfiguration und stellen

Affected Systems
Choose systems ▾
DevelopmentServer-01 Remove
ERPSYSTEM-01 Remove

Impact
Systemüberwachungsberichte können nicht generiert werden, was zu einem Mangel an Überwachung der

Preventative Measures
Keine vorbeugenden Massnahmen erforderlich

Comments Attachments
Enter your comments here

Info
Creator: User2
Issue Nr.: 11
Duration: 5 h
Timestamp: 15:00, 01.03.24
Last updated: 15:05, 01.03.24
End time: --:--

Abbildung 23: Bearbeitungsansicht

Abbildung 23 zeigt die Bearbeitungsansicht eines Issues. Auf diese gelangt man, indem man den Edit-Button betätigt. Diese Ansicht ermöglicht es dem Benutzer, das Issue zu bearbeiten. Mithilfe von Dropdown-Menüs und Textfeldern können alle Informationen des Issues angepasst und aktualisiert werden.

Die Anordnung der Informationen und die visuelle Struktur bleiben im Vergleich zur Detailansicht unverändert. Dies stellt sicher, dass Benutzer sich nicht an ein neues Layout gewöhnen müssen und sich sofort zurechtfinden.

6.5 Chat-Funktion

Send Refresh Edit Give Initial Feedback [Print]

angezeigt
Status: Open ▾
Priority: 1/4

Ask a question:
Type your message...
Send

Suggested Solution
orschlag Überprüfen Sie die
tion und stellen Sie sicher, dass der PRTG Core
rekt konfiguriert und hinzugefügt wurde.
- ERPSYSTEM-01

Abbildung 24: Chat-Funktion

Abbildung 24 zeigt die Chat-Funktion des Systems. Diese Funktion ermöglicht es den Benutzern, direkt mit der KI zu kommunizieren, um Unterstützung und Antworten auf allfälligen Fragen zu dem angezeigten Issue zu erhalten.

Die Chat-Funktion integriert sich in das bestehende Design der Anwendung. Die Farbpalette und die Designelemente entsprechen denen der restlichen Benutzeroberfläche für ein einheitliches Erscheinungsbild.

6.6 Terminal-Funktion

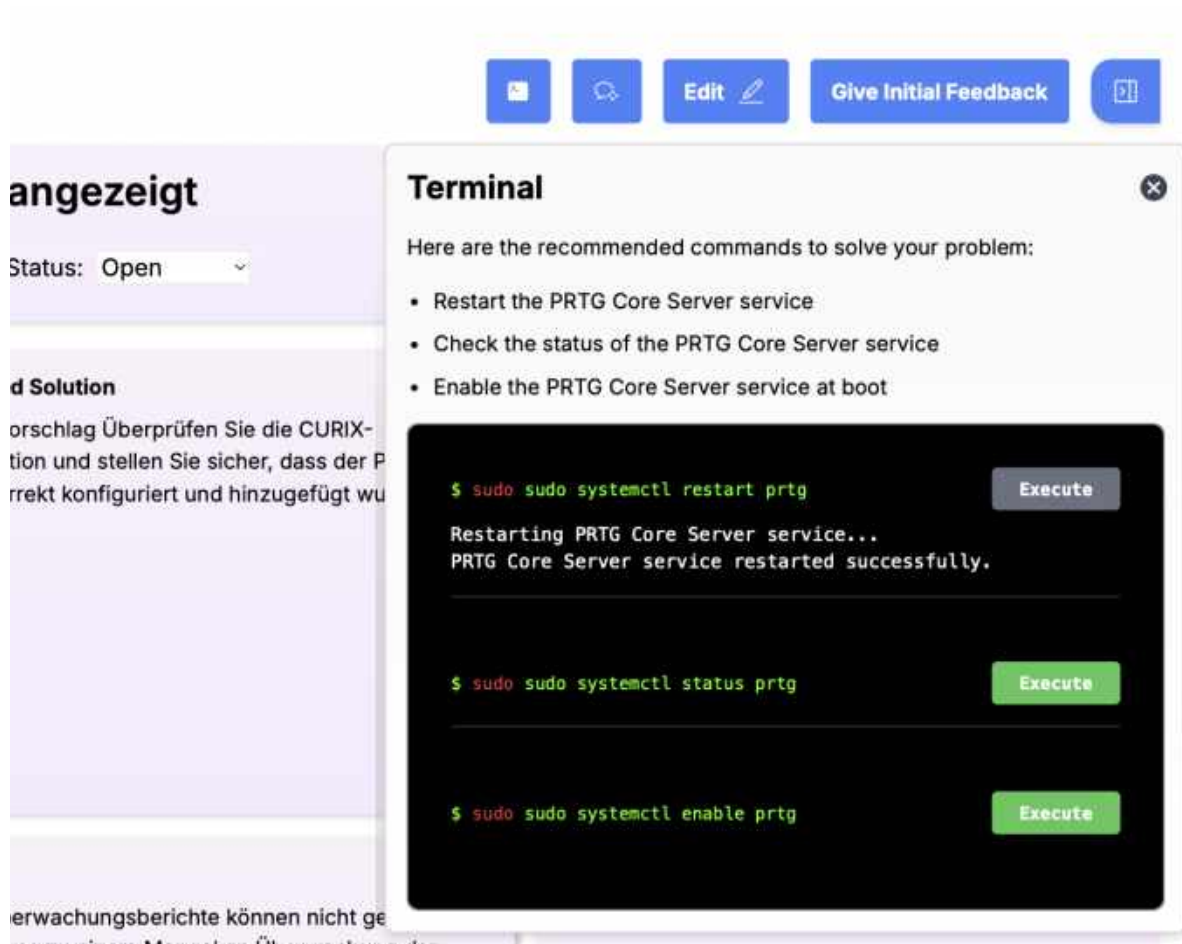


Abbildung 25: Terminal-Funktion

Abbildung 25 zeigt die Terminal-Funktion des Systems. Diese ermöglicht es den Benutzern, von der KI empfohlene Befehle auszuführen, um das Problem zu beheben. Die Terminal-Funktion kann direkt in der Detailansicht des Issues als Pop-up-Fenster geöffnet werden. Innerhalb dieses Fensters ist es möglich, die vorgeschlagenen Befehle direkt auszuführen. Jeder Befehl ist klar durch einen Divider getrennt und verfügt über einen eigenen Button zur Ausführung.

6.7 Erstellen von Issues

Neues Issue erstellen

Titel *

Titel Schablone ⓘ
Format: [Subjekt] [Prädikat] [Objekt] [Konjunktion] [Bedingung/Ort/Aktion/Prozess]
Beispiel: Server CT-10 überhitzt das System unter hoher Last.

Beschreibung *

Einleitung:
Der Datenbankserver CT-10 hat unter hoher Auslastung hohe Last- und Latenzprobleme verursacht, was die Leistung mehrerer Anwendungen beeinträchtigt.

Hintergrundinformationen:

Vorschlag annehmen

Alarmtyp **Schweregrad** **Incident-Typ** **Priorität**
Informationsmeldung ▾ Niedrig ▾ Abfrage ▾ 1: Niedrig ▾

Betroffene Systeme

Wähle die Systeme aus ▾ Keine Systeme ausgewählt

Auswirkung

Vorbeugende Massnahmen

Abbrechen **Issue erstellen**

Abbildung 26: Erstellungsansicht eines neuen Issues

Abbildung 26 zeigt die Ansicht für die Erstellung eines neuen Issues. In dieser Ansicht können Benutzer alle relevanten Informationen zu einem neuen Issue eingeben, das im System erfasst werden soll. Dabei wird auch die Satzschablone für den Titel gezeigt. Diese wird eingeblendet, wenn das Titelfeld fokussiert (onFocus) wird. Bei Bedarf kann diese auch ausgeblendet werden. Die Funktionalität gilt auch für die Beschreibung.

In Abbildung 26 wird ebenfalls gezeigt, wie die KI bei der Erstellung eines neuen Issues helfen kann. Nachdem der Titel eingegeben wurde, wird das Textfeld für die Beschreibung automatisch mit einem Textvorschlag befüllt. Dieser Text wird ausgegraut angezeigt, um zu signalisieren, dass es sich um einen provisorischen Vorschlag handelt. Der Benutzer kann nun den Vorschlag annehmen-Button klicken, um den Vorschlag zu übernehmen und bei Bedarf zu überarbeiten.

6.8 Wizard

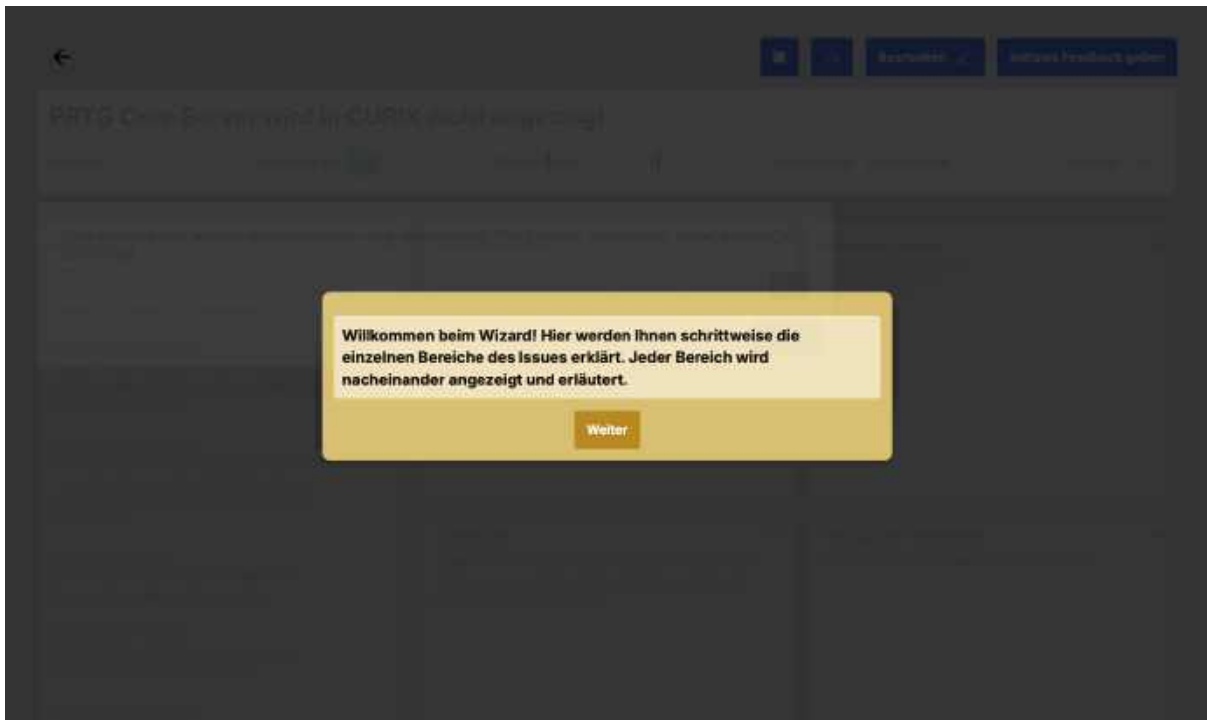


Abbildung 27: Wizard Startansicht

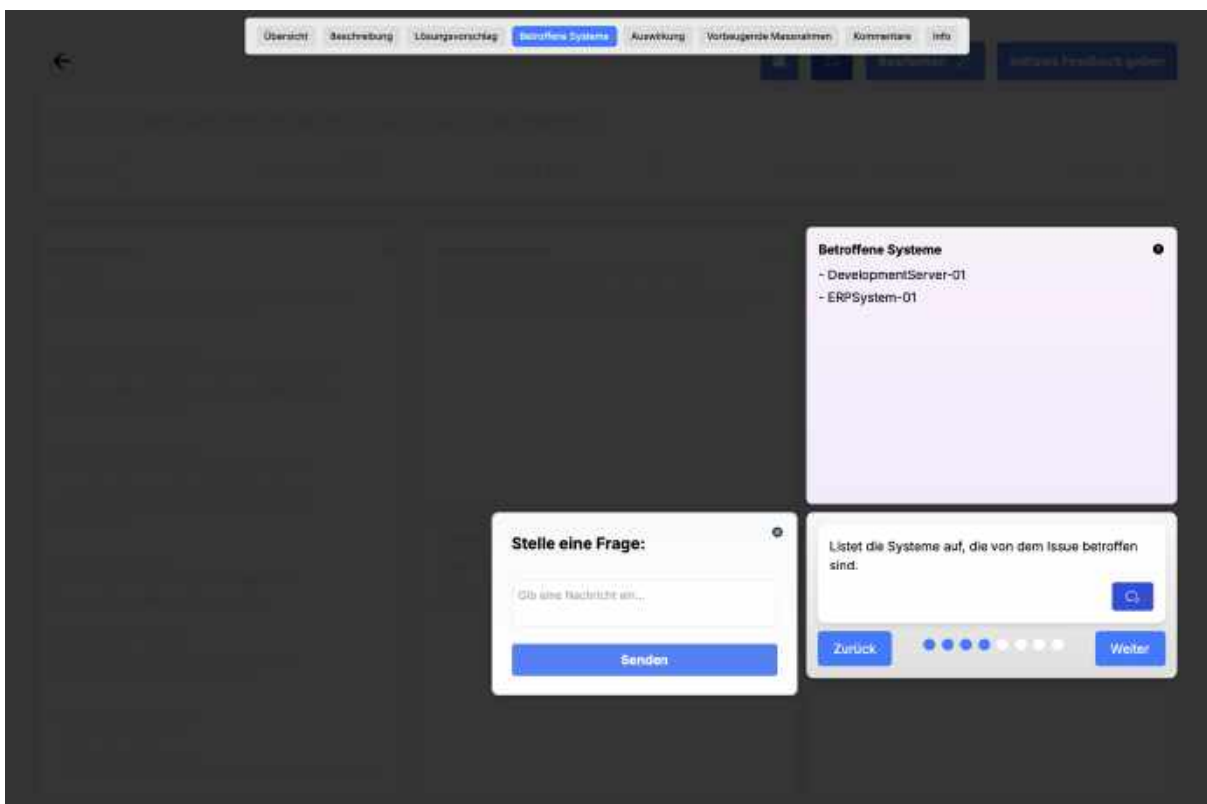


Abbildung 28: Wizard Zwischenschritt

Die Abbildungen 27 und 28 zeigen die Wizard-Funktion. Abbildung 27 zeigt eine kleine Einführung in die Funktion. Abbildung 28 gibt einen Einblick wie der Wizard-Modus in Aktion aussieht. Hier wird durch alle einzelnen Cards der Detailansicht iteriert. Der Hintergrund wird dunkler dargestellt, um den Fokus auf die aktuelle Card zu legen. Mit den Buttons „Weiter“ und „Zurück“ kann der Benutzer durch die Tour navigieren. Ein weiteres Textfeld, das entweder unterhalb, rechts oder links von der aktiven Card platziert ist, erklärt diese jeweils. Oben am Fenster befindet sich die Navigationsbar, dargestellt als horizontaler Balken. Darin sind die verschiedenen Bereiche, zu denen man übergehen kann, als Button dargestellt. In jedem Schritt ist ein Chat-Button verfügbar. Für Aktions-Buttons wurde wieder Blau als Primärfarbe gewählt.

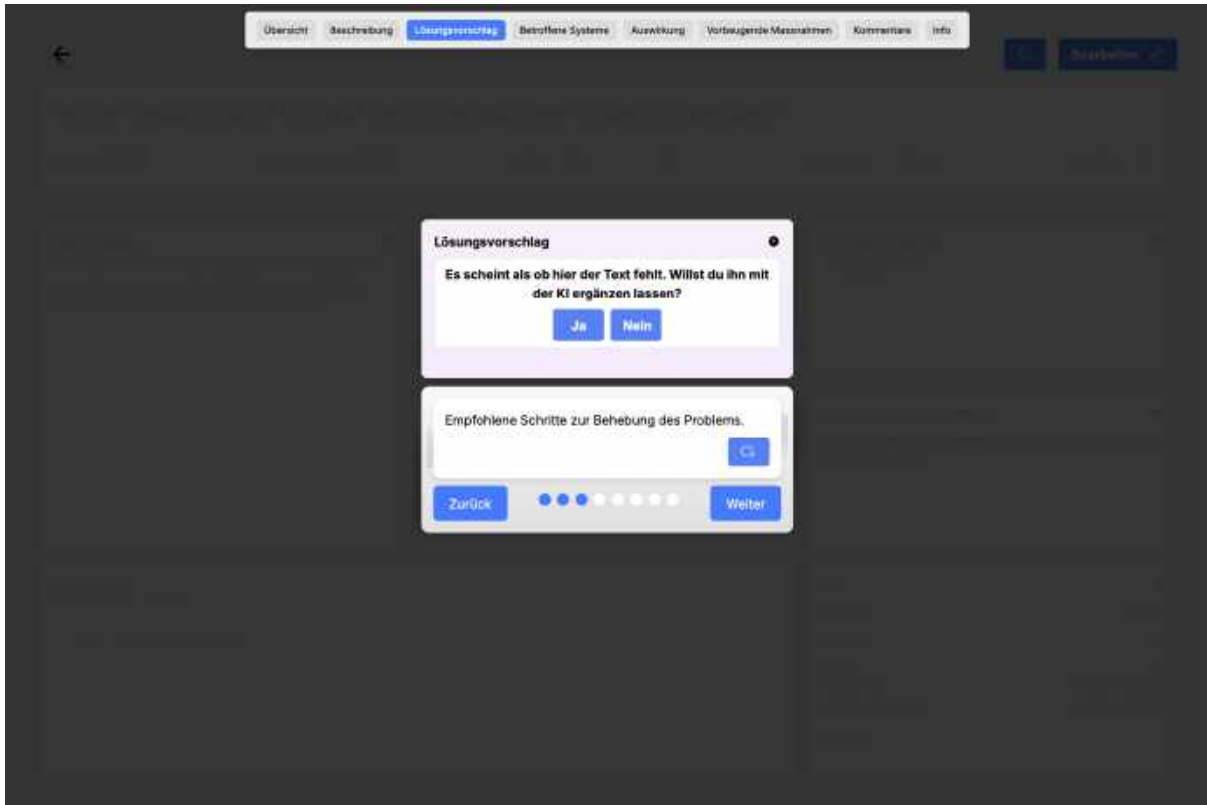
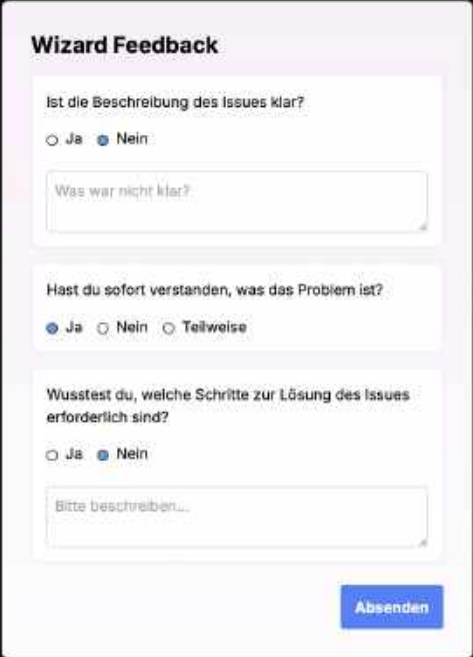


Abbildung 29: KI-Funktion zur Vervollständigung fehlender Texte

Auf der Abbildung 29 sieht man, wie im Wizard-Modus erkannt wird, dass noch kein Lösungsvorschlag vorhanden ist. Der Benutzer wird gefragt, ob die KI diesen Text generieren soll.

6.9 Wizard-Feedback



The image shows a feedback form titled "Wizard Feedback" with a light purple header. It contains three distinct sections, each with a question and radio button options, followed by a text input field. A blue "Absenden" button is located at the bottom right.

Wizard Feedback

Ist die Beschreibung des Issues klar?

Ja Nein

Was war nicht klar?

Hast du sofort verstanden, was das Problem ist?

Ja Nein Teilweise

Wusstest du, welche Schritte zur Lösung des Issues erforderlich sind?

Ja Nein

Bitte beschreiben...

Absenden

Abbildung 30: Wizard-Feedback

Die Abbildung 30 zeigt das Feedbackformular, das nach Abschluss der Wizard-Tour angezeigt wird. Es erscheint als ein Pop-up-Fenster, das nicht übersprungen werden kann. Da das Feedbackformular nur aus drei Fragen besteht, ist es einfach gehalten. Die Farben sind wie gewohnt und jede Frage ist mit der Antwortmöglichkeit separat in einer Card.

6.10 Initial-Feedback

Issue Details

Alarmtyp: Schweregrad: **Niedrig** Priorität: 1/4

PRTG Core Server wird in CURIX nicht angezeigt

Einleitung
Beim Versuch, den PRTG Core Server in CURIX zu finden, wird dieser nicht angezeigt.

Hintergrundinformationen
Dieses Problem wurde von User2 gemeldet, der versuchte, den PRTG Core Server in der CURIX-Konfiguration zu finden.

Schritte zur Reproduktion
1. Öffnen Sie die CURIX-Konfigurationssseite.
2. Suchen Sie nach dem PRTG Core Server.
3. Überprüfen Sie die Liste der angezeigten Komponenten.

Erwartetes Verhalten
Der PRTG Core Server sollte in der CURIX-Konfigurationsliste angezeigt werden.

Tatsächliches Verhalten
Der PRTG Core Server wird in der CURIX-Konfigurationsliste nicht angezeigt.

Zusätzliche Informationen
- Log-Datei beigefügt
- Tritt auf in Version 3.4.1
- Keine Auswirkungen auf andere Systeme festgestellt.

Initiales Feedback

Ist die Beschreibung des Issues klar? Ja Nein

Hast du sofort verstanden, was das Problem ist? Ja Nein Teilweise

Fehlen dir wichtige Informationen? Ja Nein

Wie würdest du die Verständlichkeit des Issues bewerten?

Wusstest du, welche Schritte zur Lösung des Issues erforderlich sind? Ja Nein

Wie nachvollziehbar ist die Priorität? 1 2 3 4
wenig sehr

Wie nachvollziehbar ist die Severity? 1 2 3 4
wenig sehr

Abbrechen **Absenden**

Abbildung 31: Initial-Feedback

Abbildung 31 zeigt die Ansicht für das initiale Feedback. Diese ähnelt nach wie vor den Wireframes. Auf der linken Seite sind die Attribute des Issues übersichtlich angeordnet, während auf der rechten Seite das Feedback-Formular platziert ist. Für das Feedback wurden verschiedene Elemente verwendet wie Bullet Points mit den Optionen „Ja“ und „Nein“ sowie bei einer Frage „Ja“, „Nein“ und „Teilweise“, eine Sternbewertung von 1 bis 5 und zwei Slider von 1 bis 4. Unten rechts befinden sich die Buttons „Abbrechen“ und „Absenden“.

6.11 Final-Feedback

Finales Feedback

War der Titel des Issues klar und verständlich?

PRTO Core Server wird in CURIX nicht angezeigt

Ja Nein

Wie war nicht klar?

War die Beschreibung des Issues klar und verständlich?

Ertitelung
Beim Versuch, den PRTO Core Server in CURIX zu finden, wird dieser nicht angezeigt.

Hintergrundinformationen
Dieses Problem wurde von User2 gemeldet, der versuchte, den PRTO Core Server in der CURIX-Konfiguration zu finden.

Schritte zur Reproduktion
1. Öffnen Sie die CURIX-Konfigurationssseite.
2. Suchen Sie nach dem PRTO Core Server.
3. Überprüfen Sie die Liste der angezeigten Komponenten.

Erwartetes Verhalten
Der PRTO Core Server sollte in der CURIX-Konfigurationsliste angezeigt werden.

Tatsächliches Verhalten
Der PRTO Core Server wird in der CURIX-Konfigurationsliste nicht angezeigt.

Zusätzliche Informationen
- Log-Daten beigefügt
- Title auf in Version 5.4.1
- Keine Auswirkungen auf andere Systeme festgestellt.

Ja Nein

Wie war nicht klar?

War der Alarmtyp der richtige?

Wähle den richtigen Typ

Informationenmeldung -

Begründe den neuen Alarm Type

War der Incident-Typ der richtigen Kategorie zugeordnet?

Konfiguration

Ja Nein

Wähle den richtigen Typ

Datenintegrität -

War der KI-Lösungsvorschlag für das Issue relevant?

Ja Nein

Weshalb waren die Vorschläge nicht relevant?

Wusstest du, was zu tun ist, um das Issue zu lösen?

Ja Nein

Wo fehlen dir Informationen?

War der Auswirkung verständlich?

Systemüberwachungsberichte können nicht generiert werden, was zu einem Mangel an Überwachung der Netzwerkperformance führt

Ja Nein

Wie war nicht klar?

War die Priorität nachvollziehbar?

1/4

Ja Nein

Wähle die korrekte PRIORITY aus.

3 -

Begründung für neue Priority

Waren die betroffenen Systeme korrekt?

DevelopmentServer-01
ERPSystem-01
StorageSystem-01
NetworkSwitch-01

Ja Nein

Wähle die Systeme aus -

War die Severity nachvollziehbar?

Low

Ja Nein

Wähle die richtige Severity.

Mittel -

Begründung für neue Severity

Hast du den Lösungsvorschlag tatsächlich umgesetzt?

Ja Nein

Wie einfach war die Umsetzung?

1 2 3 4
Schwer Einfach

Wie zufrieden bist du insgesamt mit den KI-Funktionen? (Terminal, Chat, Lösungsvorschlag)

☆☆☆☆☆

Bitte erkläre deine Bewertung

Abbildung 32: Final-Feedback

Abbildung 32 zeigt die Ansicht für das finale Feedback. Das finale Feedback ist ähnlich aufgebaut wie das initiale Feedback. Der Unterschied besteht darin, dass hier alle Fragen eigenständig sind und horizontal mit dem jeweiligen Feedback-Element verbunden sind. In einer horizontalen Card befindet sich entweder die Frage in Verbindung mit einem Attribut und die Antwortmöglichkeit oder nur die Frage mit der Antwortmöglichkeit.

7 Evaluation des entwickelten Modells

7.1 Einleitung

In diesem Abschnitt wird beschrieben, wie das entwickelte PoC auf Wirksamkeit und Benutzerfreundlichkeit getestet wurde. Das Ziel der Evaluation ist es, Erkenntnisse darüber zu gewinnen, wie gut das PoC die Anforderungen und Erwartungen der Benutzer erfüllt. Dazu werden verschiedene Testmethoden angewendet.

Des Weiteren wird **RQ4.1**: Welche Methoden können eingesetzt werden, um die Klarheit und Handlungsorientierung der optimierten Fehlermeldungen zu messen und zu bewerten? in diesem Kapitel beantwortet.

7.2 Zielgruppe der Evaluation

Es wurden sieben Personen befragt, um ein ausgewogenes Ergebnis der Tests zu erhalten. Unsere Zielgruppe besteht aus zwei Hauptgruppen:

- **Studenten**: Diese Gruppe repräsentiert Laien, die wenig bis keine Erfahrung im Systemadministrationsbereich haben. Sie wurden ausgewählt, um die Benutzerfreundlichkeit und Verständlichkeit der Fehlermeldungen und des Feedbackmechanismus aus der Perspektive von Anfängern zu bewerten. Da das System bewusst so entwickelt wurde, dass es nicht nur für Benutzer technischer Natur ist, hilft uns diese Gruppe festzustellen, ob unser System auch für unerfahrene Benutzer intuitiv und leicht verständlich ist.
- **Experten**: Diese Gruppe umfasst Personen, die bereits im Bereich der Systemadministration tätig sind und über Kenntnisse und Erfahrungen verfügen. Die Zielgruppe ermöglicht es, PoC aus der Perspektive erfahrener Benutzer zu beurteilen. Ihr Feedback ist deshalb wertvoll, da es sicherstellt, dass das entwickelte PoC auch den Anforderungen und Erwartungen von erfahrenen Systemadministratoren gerecht wird.

7.3 Überblick über die angewandten Methoden und Vorgehensweise

7.3.1 Testmethoden

Verschiedene Methoden wurden angewendet, um das erarbeitete Konzept zu evaluieren. Daraus sollen qualitative und quantitative Daten gesammelt werden, um diese auszuwerten. Die Teilnehmer wurden zu Beginn über die Ziele und den Ablauf des Tests informiert. Daraufhin erhielten sie die Aufgabe, die Anwendung selbstständig zu erkunden und ihre Gedanken laut auszusprechen. Die angewandten Methoden und deren Vorgehensweise werden im Folgenden beschrieben:

- **Guerilla-Testing**: Die Tests wurden in einem informellen Rahmen durchgeführt, bei dem die Testpersonen die Anwendung selbstständig erkunden und dabei ihre Gedanken laut aussprechen. Die Probanden sollten spezifische Aufgaben erledigen, wie das initiale Feedback geben oder ein neues Issue anlegen.
- **A/B-Testing**: Den Probanden wurden jeweils zwei Versionen einer Oberfläche gezeigt, wobei direkt eine Präferenzabfrage gemacht wurde, um zu wissen, welche Version sie besser finden und warum.
- **Interviews**: In einem kurzen Schlussinterview wurden Fragen gestellt wie beispielsweise ob der Benutzer einen roten Faden sieht, weitere Anmerkungen hat oder sonstige Hinweise.

7.3.2 Vorgehensweise

Zu Beginn erhielten die Testpersonen eine kurze Einführung in die Thematik. Anschliessend wurden die Teilnehmer gebeten, die Anwendung eigenständig zu erkunden und dabei ihre Gedanken gemäss der Think-Aloud-Methode [26] zu äussern. Während der Nutzung der Anwendung wurden Fragen gestellt, um die Denkweise der Benutzer zu verstehen, beispielsweise: „Was denkst du gerade?“ oder „Warum hast du dich entschieden, dort zu klicken?“.

Nach Abschluss der Aufgaben wurden die Teilnehmer mündlich nach Feedback gefragt um Einblicke in die Benutzererfahrung zu erhalten. Es wurde darauf geachtet, dass die Teilnehmer möglichst offen

und ehrlich ihre Meinung mitteilen können. Fragen wie „Gab es etwas, das verwirrend oder frustrierend war?“ sollten sie dazu ermutigen, über Schwierigkeiten zu sprechen. Jeder Test dauerte etwa 30 bis 45 Minuten.

7.3.3 Testaufgaben

Die Testpersonen wurden gebeten, Aufgaben durchzuführen. Zu diesen Aufgaben gehörten:

- Navigation durch die Anwendung.
- Erstellen von einem neuen Issue.
- Nutzung der KI-Funktionen wie Chat und Terminal.
- Geben von Feedback.
- Bearbeiten eines Issues.

7.4 Ergebnisse der Evaluation

7.4.1 Listenansicht

A/B-Testing:

Es wurden zwei unterschiedliche Darstellungsformen verglichen: eine Gitteransicht, die von konventionellen Designs abweicht, und eine klassische Listenansicht. Das Ziel war es, herauszufinden, welche der beiden Ansichten besser bei den Benutzern ankommt. Die Ergebnisse zeigten, dass die klassische Listenansicht bevorzugt wurde. Die Benutzer fanden diese Ansicht vertrauter und konnten sich leichter darin zurechtfinden. Abbildung 38 zeigt die Gitteransicht. Diese benötigt zudem deutlich mehr Platz als die Listenansicht, wodurch die Listenansicht für ein schlankeres Design bevorzugt wurde.

Guerilla-Testing und Interviews:

Es wurde bemängelt, dass die Cards nicht als klickbar erkannt wurden. Eine Änderung der Farbe beim Hover-Effekt könnte hier Abhilfe schaffen. Es wurde auch der Wunsch nach einer Animation beim Klick auf die Cards geäußert. Zudem sollten die Pfeile, um zwischen den neuen Issues zu navigieren, links und rechts neben der Card anstatt unterhalb positioniert werden.

Zusätzlich wurde bemängelt, dass der Bereich der neuen Issues nicht sofort erkannt wurde. Es wurde vorgeschlagen, die Anzahl neuer Issues als Nummer anzuzeigen. Ausserdem wurde eine Sortierfunktion gewünscht, die Issues sowohl in der Card als auch unten in der Listenansicht sortieren kann. Die Implementierung von Tooltips wurde als Verbesserungsmaßnahme vorgeschlagen.

7.4.2 Detailansicht

A/B-Testing:

Es wurden zwei Anordnungen der Cards getestet. Eine Variante platzierte den Lösungsvorschlag direkt in der Beschreibung, während eine andere diesen als separate Card anzeigte. Die Mehrheit fand es besser, wenn der Lösungsvorschlag alleine stehend ist. Dies sieht man in der Abbildung 12. Ebenfalls wurde präferiert, wenn die Card mit dem Lösungsvorschlag in der Mitte (bei aufgeklappter dritter Spalte) und die mit der Beschreibung links davon platziert ist. In den Abbildungen 21 und 39 sieht man diese Gegenüberstellung der beiden Versionen.

In der Detailansicht war ursprünglich die Hauptfarbe Weiss. Es wurde getestet, ob Weiss oder die gleiche Farbe wie in der Listenansicht (Lavendel/Lila) für eine höhere Konsistenz innerhalb der Applikation bevorzugt wird. Die Tests ergaben, dass Lavendel/Lila besser geeignet ist. Dies führte zu der Frage, wie der obere Bereich der Detailansicht gestaltet werden sollte, da der Titel sowie die Attribute Alert Type, Priorität, Status, Incident Type und Severity nicht in einer Card zusammengefasst waren und somit vom restlichen Design abwichen. Das Ergebnis war, dass der Titel zusammen mit den genannten Attributen in einer Card im oberen Bereich platziert werden kann, um ein konsistentes und einheitliches Design zu gewährleisten.

Guerilla-Testing und Interviews:

Zu Beginn war die Abtrennung der Cards im oberen Bereich nicht klar ersichtlich. Dies lag daran, dass die Cards keinen sichtbaren Rand hatten, der Schatten nach unten rechts geworfen wurde und die Hintergrundfarbe der Cards mit der des Hintergrund übereinstimmte. Dadurch war die obere Kante der

Cards optisch nicht vom Hintergrund abgetrennt. Eine Anpassung der Farbe der Cards zu Lavendel/Lila half, dieses Problem zu lösen. Zudem wurde die Detailansicht teilweise als überladen empfunden. Deshalb wurde die dritte Spalte so gestaltet, dass sie ein- und aufklappbar ist. Die Positionierung des Feedback und Schliessen-Buttons unten rechts wurde von allen Benutzern als intuitiv empfunden, während die übrigen Buttons oben rechts angeordnet werden sollten. Die Einführung von Tooltips in Form von Fragezeichen-Icons auf jeder Card, die deren Funktionalität erklärt, wurde als hilfreich bewertet.

7.4.3 Bearbeitungsansicht

A/B-Testing:

Für die Bearbeitungsansicht wurde erst eine formularähnliche, von oben nach unten fokussierte Version wie in Abbildung 40 ausgearbeitet. Diese Version wurde dann mit der neuen Ansicht wie man sie in der Abbildung 23 sieht evaluiert. Vor Allem für Laien war es verständlicher, wenn sich die Ansicht nicht änderte, wenn man in den Bearbeitungsmodus wechselt. Dadurch blieb die Orientierung erhalten und die Benutzer wussten intuitiv, wo sich die Elemente befanden.

7.4.4 Neues Issue-Ansicht

Guerilla-Testing und Interviews:

Es wurde vorgeschlagen, das Dropdown-Menü für die Auswahl von Incident-Typen alphabetisch zu sortieren. Wesentliche Attribute sollten mit einem Stern als Pflichtfelder markiert werden, und der Status sollte standardmässig auf „New“ gesetzt sein. Ausserdem wurde gewünscht, dass die Satzschablone für den Titel und die Vorlage die Beschreibung durch einen einfachen Klick ausserhalb des Fensters geschlossen werden können.

7.4.5 Design

- **Icons:** Die Icons wurden gut angenommen und waren für die Benutzer verständlich. In Kombination mit den Tooltips hat sich ihre Verständlichkeit weiter verbessert, was zu einer noch intuitiveren Bedienung geführt hat.
- **Farben:** Die verwendete Farbpalette wurde grösstenteils positiv bewertet, insbesondere die Konsistenz zwischen der Detailansicht und der Listenansicht, die beide gleiche Farben verwenden. Es wurde angeregt, diese Farbübereinstimmung auch bei den Buttons zu berücksichtigen, um ein einheitliches Erscheinungsbild zu gewährleisten.
- **Einheitlichkeit und Layout:** Die allgemeine Struktur wurde als logisch empfunden. Das Design der Anwendung wurde insgesamt als konsistent wahrgenommen. Jedoch gab es Feedback, dass die Farbgebung und das Layout in einigen Bereichen besser aufeinander abgestimmt sein könnten. Beispielsweise wurde angemerkt, dass die Farbkonsistenz zwischen der Detailansicht und der Listenansicht verbessert werden könnte, insbesondere in Bezug auf die Buttons und die Farbwahl für Buttons und wichtige visuelle Elemente.
- **Cards:** Die Entscheidung, Cards als grundlegendes Designelement zu verwenden, wurde positiv aufgenommen. Dessen Gestaltung sollte in allen Bereichen konsistent bleiben, insbesondere hinsichtlich des Rahmens, des Schattenwurfs und anderer visueller Details.

7.4.6 Nicht-Design

- **Actionability:**

Die Vorlage für die Beschreibung tragen zur allgemeinen Actionability bei, wodurch die Issues klar und handlungsorientiert sind. Der Lösungsvorschlag kam bei den Benutzern gut an und wurde als Unterstützung sinnvoll empfunden. Es ist jedoch wichtig zu beachten, dass die Wirksamkeit dieser Empfehlungen stark von der Qualität der KI abhängt. Je besser die KI trainiert ist und je umfangreicher die verfügbaren Daten sind, desto präziser und nützlicher sind die Empfehlungen, was letztlich die Umsetzbarkeit der Hinweise verbessert.

- **Satzschablone:** Die Verwendung von Templates und Vorlagen wurde durchweg positiv bewertet. Die Benutzer fanden sie leicht verständlich und empfanden sie als sinnvolles Hilfsmittel, um standardisierte Inhalte effizient zu erstellen.

- **Tooltips:** Es wurde kritisiert, dass es nicht genug Tooltips gab, um die Benutzer gut zu unterstützen. Es wurde vorgeschlagen, mehr Tooltips einzufügen, um die Benutzer besser zu führen und Unsicherheiten zu verringern.
- **Kontextbezogenheit:** Die Auswahl der Attribute und die Vorlage der Beschreibung sorgten für einen ausreichend reichhaltigen Kontext, der von den Benutzern als hilfreich empfunden wurde.

7.4.7 KI-Funktionen

- **Hilfe bei der Erstellung von Issues, Chat- und Terminal-Funktion:** Diese drei Funktionen wurden insgesamt positiv bewertet. Die Hilfe bei der Erstellung von Issues, sowie auch die Chat-Funktion wurde als nützlich empfunden. Die Terminal-Funktion überzeugte durch die direkte Ausführung von Befehlen in der Detailansicht und wurde als neuartig und praktisch empfunden. Die drei Funktionen sind gut in das Design integriert und es gab keine Beschwerden diesbezüglich.

7.4.8 Wizard-Modus

Aus zeitlichen Gründen war es nicht möglich, den Wizard-Modus zu testen.

7.4.9 Feedback

- **Design:** Zu Beginn waren alle Feedbackfragen zusammengefasst und nur im finalen Feedback vorhanden. Im Laufe des Feedback-Prozesses wurde das ursprüngliche finale Feedback in verschiedene Phasen unterteilt: Initial-Feedback, Final-Feedback und Wizard-Feedback. Das Problem war, dass der Benutzer mit zu vielen Fragen belästigt und überfordert wurde. Dies war nicht Benutzerfreundlich. Das Design mit den Cards wurde später hinzugefügt. Zuvor war es ein einfaches Formular. Dies wurde gewünscht, um das Design über die gesamte Plattform konsistent zu halten. Zudem sollten Textfelder zur Begründung nur dann erscheinen, wenn die Antwort „Nein“ ausgewählt wird. Die erhöht die Logik und die hält das Design schlank.
- **Bewertungsmethoden:** Es wurden verschiedene Bewertungsmethoden getestet, um herauszufinden, für welche Frage welche Methode am besten angenommen wird. Dieses Kapitel beantwortet **RQ2.1:** Welche Feedbackmechanismen (Multiple-Choice, Skala, etc.) sind am effektivsten für die Bewertung von Fehlermeldungen? Zu den getesteten Methoden gehören:
 - **Radio-Buttons:** Die Farbwahl fiel bei den Radio-Buttons auf Blau, um die Konsistenz im Design beizubehalten, da Blau in der gesamten Anwendung als primäre Farbe verwendet wird. Die Entscheidung für Radio-Buttons wurde getroffen, weil sie den Benutzern eine Möglichkeit bieten, zwischen vordefinierten Optionen zu wählen. Diese Methode ist nützlich, wenn nur eine Auswahl getroffen werden soll, beispielsweise geschlossene Fragen. Die Radio-Buttons wurden in das finale Design integriert.



Abbildung 33: Radio-Buttons

- **Textfelder:** Um dem Benutzer die Möglichkeit zu geben, detailliertere Antworten zu geben, wird ein Textfeld gebraucht. Ein Beispiel ist die Frage: „War der Titel des Issues klar und verständlich?“, wenn die Antwort „Nein“ lautet, wird ein Textfeld angezeigt, in dem die Frage „Was war unklar?“ gestellt wird. Dies ermöglicht es, präzisere Rückmeldungen zu erhalten und die spezifischen Probleme der Benutzer besser zu verstehen. Die Benutzer äusserten den Wunsch, dass das Textfeld erst dann erscheint, wenn man eine Antwort ausgewählt hat.



Abbildung 34: Textfeld

- **Slider:** Der Slider hat es ebenfalls in das finale Design geschafft. Im Gegensatz zur Sterne-Bewertung bietet der Slider eine Auswahl auf einer 4-Punkte-Skala (1-4), wobei bewusst keine Möglichkeit zur Auswahl einer neutralen Mitte gegeben wurde. Dies wurde so gestaltet, um die Benutzer zu einer eindeutigen Bewertung zu bewegen. Zu Beginn fehlten die Labels an den Endpunkten des Sliders, was bei den Benutzer zu Verwirrung führte, da nicht sofort ersichtlich war, welche Bedeutung die einzelnen Positionen hatten. Nach Rückmeldungen wurden die Endpunkte mit den Labels „wenig“ und „sehr“ versehen, was die Verständlichkeit deutlich verbesserte.



Abbildung 35: Slider

- **Sternen-Bewertung:** Die Sterne-Bewertung wurde sowohl in Blau als auch in Gelb getestet, da Blau überall sonst im Design verwendet wurde, jedoch Sterne normalerweise als gelb wahrgenommen werden. Die Wahl fiel letztendlich auf die gelbe Farbgebung, da diese mit bekannten Bewertungssystemen Erinnerung weckt und dadurch von den Benutzern gut angenommen wurde. Diese Bewertung wurde bis ins finale Design beibehalten. Sie wurde insbesondere für Fragen verwendet, bei denen die Benutzer eine qualitative Einschätzung geben können. Es wurde eine 5-Punkte-Skala gewählt, da sie als Standard in vielen Bewertungssystemen bekannt ist. Eine 5-Punkte-Skala bietet ausserdem eine ausgewogene Bewertung, die es dem Benutzer ermöglicht, eine neutrale Mittelpunktoption zu wählen. Somit drängt sie den Benutzer nicht zu einer positiven oder negativen Bewertung.



Abbildung 36: Sternen-Bewertung

- **Emoji-Bewertung:** Mit der Emoji-Bewertung war die Idee einen modernen Ansatz ausprobieren, um den Benutzern eine spielerische Möglichkeit zur Bewertung zu bieten. Allerdings stellte sich schnell heraus, dass diese Methode nicht geeignet ist. Die Benutzer empfanden die Emoji-Bewertung als weniger passend im Vergleich zu den klassischen Bewertungsmethoden. Diese boten eine klarere Möglichkeit zur Bewertung, weshalb die Emoji-Bewertung letztendlich nicht ins finale Design aufgenommen wurde.



Abbildung 37: Emoji-Bewertung

- **Zeitpunkt um Feedback zu geben:** Wie bereits erwähnt wurde zu Beginn das gesamte Feedback am Ende des Prozesses gesammelt. Diese Herangehensweise stellte sich jedoch für die Benutzer als arbeitsaufwendig heraus, da alle Fragen standardmässig als obligatorisch markiert waren. Es wurde daher überlegt, wie das Feedback zeitlich besser verteilt werden könnte. Die Fragen an sich wurden als relevant und wichtig erachtet, aber es wurde erkannt, dass sie zu unterschiedlichen Zeitpunkten gestellt werden könnten, um den Aufwand für den Benutzer zu reduzieren. Aus diesem

Grund wurde das Initial-Feedback eingeführt, was bei den Benutzern besser ankam. Viele Fragen überschritten sich in den verschiedenen Phasen, deswegen wurden ähnliche Fragen präziser formuliert oder entfernt. Im Zuge dessen wurde auch das Wizard-Feedback implementiert, um die Fragen weiter aufzuteilen. Zusätzlich wurden in den Feedbackformularen einige Fragen als obligatorisch und andere als freiwillig gekennzeichnet, um den Benutzer weiter zu entlasten. Die freiwilligen Fragen wurden bewusst einfach gestaltet, um dem Benutzer das Gefühl zu geben, dass er auch dieses Feedback noch ausfüllen kann.

8 Schlussfolgerung

Diese Arbeit hat sich mit der Verbesserung der Darstellung und Verständlichkeit sowie die Benutzerfreundlichkeit von Meldungen im Kontext des Systemmonitorings auseinandergesetzt. Durch die Entwicklung eines PoC wurden Lösungen erarbeitet, um die identifizierten Pain Points zu adressieren und die Benutzerfreundlichkeit zu steigern. Ausgangspunkt war die State-of-the-Art-Analyse sowie eine Marktanalyse bestehender Systeme.

Das entwickelte PoC fokussiert sich auf eine Darstellung, die sowohl für Experten als auch für Laien verständlich ist. Es wurde besonderer Fokus auf Laien gelegt, da diese oft mit komplexen Begriffen und Fachjargon konfrontiert sind, die ihnen das Verständnis von Fehlermeldungen erschweren. Hierbei wurden Designelemente wie Farbgebung, Icons und ein minimalistisches, aber intuitives Layout implementiert, um eine klare Benutzeroberfläche zu gewährleisten und sicherzustellen, dass die in den Forschungsfragen definierten Anforderungen berücksichtigt werden.

Das Konzept der KI-Funktionen wurde von allen Benutzern im Test positiv aufgenommen, was die Relevanz dieser Idee zeigt. Es erwies sich auch, dass KI in diesem Bereich, besonders um Laien zu unterstützen, noch Potenzial für zukünftige Entwicklungen bietet.

Die Einbindung eines Feedback-Mechanismus wurde von den Benutzern während der Tests gut verstanden. Ausserdem wurde in keinem System während der Marktanalyse ein solches Feature entdeckt. Eine der Herausforderungen bestand jedoch darin, zu entscheiden, an welchen Stellen im Prozess Fragen gestellt werden sollen, da das PoC drei unterschiedliche Phasen für das Einholen von Feedback bietet: nach dem Wizard-Modus, initiales Feedback und finales Feedback. Die erarbeiteten Lösungen könnten in zukünftigen Implementierungen und Forschungsprojekten als Grundlage für die Optimierung der Interaktion zwischen Mensch und Maschine dienen.

9 Ausblick

9.1 Automatisierte Fehlerlösung durch KI

Ein weiterer Schritt in der Entwicklung kann die vollständige Automatisierung der Fehlerlösung durch KI sein. In Zukunft könnte die KI in der Lage sein, nicht nur Vorschläge zur Fehlerbehebung zu machen, sondern diese auch direkt selbstständig auszuführen. Dies würde bedeuten, dass die KI nach der Identifikation eines Problems automatisch die erforderlichen Befehle im Terminal ausführt, um das Problem zu beheben, ohne dass menschliches Eingreifen erforderlich ist.

Nach der Identifikation eines Problems kann die KI die notwendigen Schritte zur Behebung eigenständig einleiten und ausführen. Ein Systemadministrator könnte Benachrichtigungen über die durchgeführten Aktionen erhalten und bei Bedarf eingreifen oder die Aktionen bestätigen. Die Umsetzung dieses Konzepts erfordert jedoch fortgeschrittene KI-Technologien und umfangreiche Tests, um die Zuverlässigkeit und Sicherheit der automatisierten Prozesse zu gewährleisten.

9.2 Blacklist Satzschablone

Ein weiteres zukünftiges Feature für das System ist eine Blacklist für Satzschablonen für den Titel. Dieses Feature würde es ermöglichen, bestimmte Begriffe oder Phrasen, die als unangemessen oder ineffektiv angesehen werden, automatisch aus den Titeln der Issues auszuschliessen.

Eine solche Blacklist würde sicherstellen, dass nur relevante und genug präzise Titel angewendet werden. In Umgebungen, wo viele Benutzer Zugang zum System haben und die Möglichkeit besteht, das unerwünschte oder zu ungenaue Titel verwendet werden, könnte dies sich als nützlich erweisen. Dies fördert indirekt wiederum die Aspekte wie die Effizienz bei der Bearbeitung des Issues und hilft der KI besser zu werden.

Ein weiteres Ziel der Blacklist wäre es, eine konsistente und professionelle Sprache in den Titeln zu fördern. Durch die Definition von unerwünschten Ausdrücken könnte das System die Benutzer dazu anregen, klarere und aussagekräftigere Titel zu verwenden.

9.3 Animationen

Der Einsatz von mehr Animationen könnte zur Verbesserung der Benutzererfahrung beitragen. Aktuell wirkt die Benutzeroberfläche in einigen Bereichen statisch. Durch einbeziehen von Animationen könnte die Interaktion mit dem System ansprechender gestaltet werden.

Animationen könnten beispielsweise verwendet werden, um den Benutzer auf wichtige Änderungen oder Aktionen hinzuweisen, wie das Erscheinen von Bestätigungsnachrichten. Auch beim Übergang zwischen verschiedenen Ansichten könnte durch fließende Animationen eine benutzerfreundlichere Navigation ermöglicht werden.

10 Anhang

10.1 Wizard-Feedback

- **Ist die Beschreibung des Issues klar? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, was unklar war.
 - **Ziel:** Festzustellen, ob die Beschreibung des Issues verständlich ist und ob es Unklarheiten gibt.
 - **Warum erforderlich?** Die Klarheit der Beschreibung ist entscheidend, um das Problem zu Beginn richtig zu verstehen und zu beheben.
- **Hast du sofort verstanden, was das Problem ist? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein / Teilweise
 - **Zusätzliche Aktion:** Falls „Nein“ oder „Teilweise“, erscheint ein Textfeld, in dem der Benutzer angeben kann, was nicht verstanden wurde.
 - **Ziel:** Sicherzustellen, dass der Benutzer das Problem nachvollziehen kann. Das hilft, Missverständnisse zu vermeiden.
 - **Warum erforderlich?** Diese Frage ist wichtig, um zu bewerten, ob die Beschreibung und die bereitgestellten Informationen ausreichend sind, damit der Benutzer das Problem sofort versteht.
- **Wusstest du, welche Schritte zur Lösung des Issues erforderlich sind? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, welche Schritte unklar sind.
 - **Ziel:** Herauszufinden, ob der Lösungsvorschlag des Issues ausreichend ist, damit der Benutzer die notwendigen Schritte zur Problemlösung kennt.
 - **Warum erforderlich?** Es ist wichtig zu wissen, ob der Benutzer die notwendigen Schritte zur Lösung des Issues versteht, da unklare Anweisungen zu Fehlern und längerer Problemlösung führen können.

10.2 Initial-Feedback

- **Ist die Beschreibung des Issues klar und verständlich? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, was unklar war.
 - **Ziel:** Festzustellen, ob die Beschreibung des Issues verständlich ist und ob es Unklarheiten gibt.
 - **Warum erforderlich?** Die Klarheit der Beschreibung ist entscheidend, um das Problem zu Beginn richtig zu verstehen und zu beheben.
- **Hast du sofort verstanden, was das Problem ist? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein / Teilweise
 - **Zusätzliche Aktion:** Falls „Nein“ oder „Teilweise“, erscheint ein Textfeld, in dem der Benutzer angeben kann, was nicht verstanden wurde.
 - **Ziel:** Sicherzustellen, dass der Benutzer das Problem nachvollziehen kann. Das hilft, Missverständnisse zu vermeiden.
 - **Warum erforderlich?** Diese Frage ist wichtig, um zu bewerten, ob die Beschreibung und die bereitgestellten Informationen ausreichend sind, damit der Benutzer das Problem sofort versteht.

- **Fehlen dir wichtige Informationen? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Ja“, erscheint ein Textfeld, in dem der Benutzer angeben kann, welche Informationen fehlen.
 - **Ziel:** Zu ermitteln, ob zusätzliche Informationen benötigt werden, um das Issue zu verstehen und zu lösen.
 - **Warum erforderlich?** Diese Frage hilft zu identifizieren, ob der Benutzer alle notwendigen Informationen hat, um das Issue zu verstehen und allenfalls zu lösen.
- **Wusstest du, welche Schritte zur Lösung des Issues erforderlich sind? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, welche Schritte unklar sind.
 - **Ziel:** Herauszufinden, ob der Lösungsvorschlag des Issues ausreichend ist, damit der Benutzer die notwendigen Schritte zur Problemlösung kennt.
 - **Warum erforderlich?** Es ist wichtig zu wissen, ob der Benutzer die notwendigen Schritte zur Lösung des Issues versteht, da unklare Anweisungen zu Fehlern und längerer Problemlösung führen können.
- **Wie würdest du die Verständlichkeit des Issues bewerten? (R)**
 - **Antwortmöglichkeiten:** 1 bis 5
 - **Ziel:** Eine allgemeine Bewertung der Verständlichkeit des Issues zu erhalten.
 - **Warum erforderlich?** Eine allgemeine Bewertung der Verständlichkeit gibt einen guten Überblick darüber, wie gut das Issue insgesamt verstanden wird.
- **Wie nachvollziehbar ist die Priorität? (O)**
 - **Antwortmöglichkeiten:** 1 (wenig) bis 4 (sehr)
 - **Ziel:** Zu bewerten, ob die festgelegte Priorität des Issues aus Sicht des Benutzers sinnvoll und nachvollziehbar ist.
 - **Warum optional?** Die Priorität eines Issues ist wichtig, aber es ist eine ergänzende Information, die nicht direkt die Verständlichkeit des Problems beeinflusst. Es kann hilfreich sein, diese Information zu haben, ist aber nicht zwingend erforderlich. Zudem wird die Antwort über einen Slider gegeben, was eine schnelle und benutzerfreundliche Möglichkeit bietet, sodass der Benutzer eher bereit ist, dieses Feedback zu geben.
- **Wie nachvollziehbar ist die Severity? (O)**
 - **Antwortmöglichkeiten:** 1 (wenig) bis 4 (sehr)
 - **Ziel:** Zu bewerten, ob der Schweregrad des Issues korrekt eingeschätzt wurde und ob dies für den Benutzer klar ist.
 - **Warum optional?** Ähnlich wie bei der Priorität ist die Severity wichtig, aber eher ergänzend. Wie bei der vorherigen Frage wird hier auch ein Slider gebraucht.

10.3 Final-Feedback

- **War der Titel des Issues klar und verständlich? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, was unklar war.
 - **Ziel:** Zu bewerten, ob der Titel des Issues verständlich war.
 - **Warum erforderlich?** Ein klarer und verständlicher Titel hilft für für das Verständnis. Es ist das erste, was man von einem Issue liest.

- **War die Beschreibung des Issues klar und verständlich? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, was unklar war.
 - **Ziel:** Zu bewerten, ob die Beschreibung des Issues verständlich war.
 - **Warum erforderlich?** Eine klare und verständliche Beschreibung ist entscheidend für das Verständnis und die Lösung des Problems.
 - **Anmerkung:** Erscheint nur, falls im Wizard-Feedback oder im initialen Feedback diese Frage nicht beantwortet wurde.
- **War der Alert Type der richtige? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Dropdown und ein Textfeld, in dem der Benutzer angeben kann, was der richtige Alert Type wäre und warum.
 - **Ziel:** Festzustellen, ob der Alert Type korrekt gewählt wurde.
 - **Warum erforderlich?** Der richtige Alert Type ist wichtig, um die Dringlichkeit und Art des Problems korrekt zu kommunizieren.
- **War der Incident Type der richtigen Kategorie zugeordnet? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Dropdown, in dem der Benutzer angeben kann, was der richtige Incident Type wäre.
 - **Ziel:** Zu bewerten, ob der Incident Type korrekt zugeordnet wurde.
 - **Warum erforderlich?** Ein korrekt zugeordneter Incident Type hilft, das Problem richtig zu klassifizieren.
- **War der KI-Lösungsvorschlag für das Issue relevant? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Ja“, erscheint ein Textfeld, in dem der Benutzer angeben kann, ob die KI-Vorschläge klar und verständlich sind. Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, warum die KI-Vorschläge nicht relevant waren.
 - **Ziel:** Zu bewerten, ob der KI-Lösungsvorschlag hilfreich und verständlich war.
 - **Warum erforderlich?** Diese Frage hilft, die Nützlichkeit und Relevanz der des KI-Lösungsvorschlag zu bewerten, was entscheidend für die Verbesserung der KI ist.
- **Wusstest du, was zu tun ist, um das Issue zu lösen? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, was unklar war.
 - **Ziel:** Zu bewerten, ob der Benutzer wusste, welche Schritte zur Lösung des Issues erforderlich sind.
 - **Warum erforderlich?** Das Wissen über die notwendigen Schritte zur Lösung ist wichtig für die Problemlösung.
 - **Anmerkung:** Erscheint nur, falls im Wizard-Feedback oder im initialen Feedback diese Frage nicht beantwortet wurde.
- **War die Auswirkung verständlich? (R)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, was nicht verständlich war.

- **Ziel:** Festzustellen, ob die Auswirkungen des Issues klar und verständlich sind.
- **Warum erforderlich?** Die Auswirkungen des Issues zu verstehen, hilft, die Dringlichkeit besser einzuschätzen.
- **War die Priorität nachvollziehbar? (O)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Dropdown und ein Textfeld, in dem der Benutzer angeben kann, was die richtige Priorität wäre und warum.
 - **Ziel:** Festzustellen, ob die Priorität des Issues verständlich und korrekt war.
 - **Warum optional?** Die Priorität eines Issues beeinflusst die Dringlichkeit der Lösung, aber ist für das Feedback nicht zwingend erforderlich.
 - **Anmerkung:** Erscheint nur, falls im initialen Feedback diese Frage nicht beantwortet wurde.
- **Waren die betroffenen Systeme korrekt? (O)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Dropdown, in dem der Benutzer angeben kann, welche Systeme betroffen waren. Hier kann man mehrere Systeme an- und abwählen.
 - **Ziel:** Festzustellen, ob die betroffenen Systeme korrekt identifiziert wurden.
 - **Warum optional?** Die richtigen betroffenen Systeme zu identifizieren, ist nützlich, aber nicht zwingend erforderlich für das Feedback.
- **War die Severity nachvollziehbar? (O)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Dropdown und ein Textfeld, in dem der Benutzer angeben kann, was die richtige Severity wäre und warum.
 - **Ziel:** Festzustellen, ob die Severity korrekt war.
 - **Warum optional?** Der Schweregrad des Issues ist wichtig für die Priorisierung, jedoch für das Feedback nicht zwingend erforderlich.
 - **Anmerkung:** Erscheint nur, falls im initialen Feedback diese Frage nicht beantwortet wurde.
- **Hast du den Lösungsvorschlag tatsächlich umgesetzt? (O)**
 - **Antwortmöglichkeiten:** Ja / Nein
 - **Zusätzliche Aktion:** Falls „Nein“, erscheint ein Textfeld, in dem der Benutzer angeben kann, wie das Issue tatsächlich gelöst wurde. Falls „Ja“, erscheint ein Slider von 1 (schwer) bis 4 (einfach), um zu bewerten, wie einfach die Umsetzung war.
 - **Ziel:** Zu bewerten, ob der vorgeschlagene Lösungsweg umgesetzt wurde oder ob eine andere, eigene Lösung genutzt wurde und zu erfahren, wie einfach die Umsetzung war.
 - **Warum optional?** Es ist hilfreich zu wissen, ob der Lösungsvorschlag nützlich war, aber es ist nicht zwingend erforderlich für die Bewertung des Issues.
- **Wie zufrieden bist du insgesamt mit den KI-Funktionen (Terminal, Chat, Lösungsvorschlag)? (O)**
 - **Antwortmöglichkeiten:** Skala von 1 bis 5 und Textfeld
 - **Ziel:** Eine allgemeine Bewertung der Zufriedenheit mit den KI-Vorschlägen zu erhalten.
 - **Warum optional?** Eine allgemeine Bewertung ist hilfreich, um die Zufriedenheit mit den KI-Vorschlägen zu verstehen.

10.4 A/B-Testing

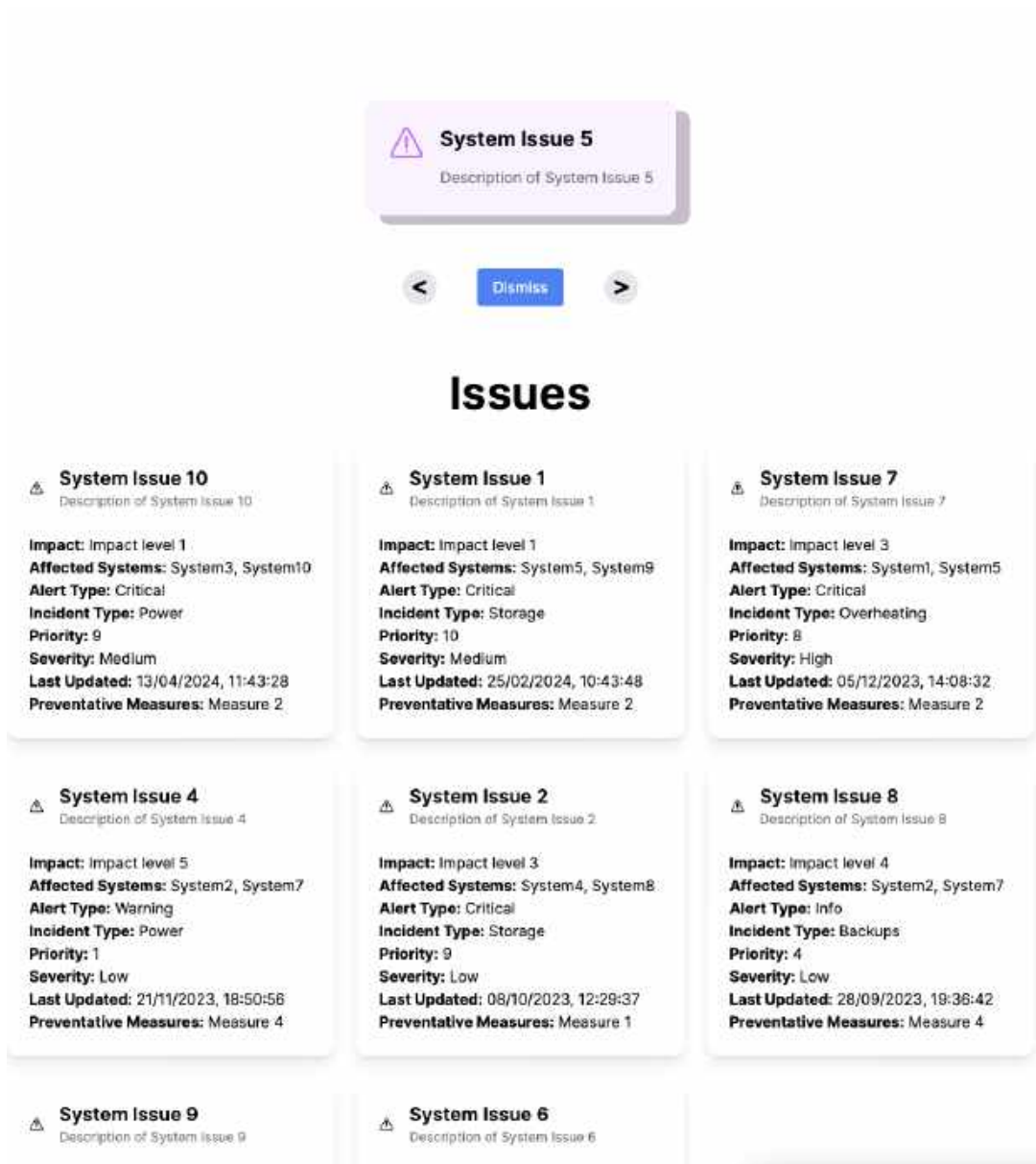


Abbildung 38: Mosaikansicht AB-Testing

←

📄
🔍
Edit
Give Initial Feedback

PRTG Core Server wird in CURIX nicht angezeigt

⚠️
Severity: Low
Status: Open
Incident type: Configuration
Priority: 1/4

Description

Erlebung:
Beim Versuch, den PRTG Core Server in CURIX zu finden, wird dieser nicht angezeigt.

Hintergrundinformationen:
Dieses Problem wurde von User2 gemeldet, der versuchte, den PRTG Core Server in der CURIX-Konfiguration zu finden.

Schritte zur Reproduktion:
1. Öffnen Sie die CURIX-Konfigurationsseite.
2. Suchen Sie nach dem PRTG Core Server.
3. Überprüfen Sie die Liste der angezeigten Komponenten.

Erwartetes Verhalten:
Der PRTG Core Server sollte in der CURIX-Konfigurationsliste angezeigt werden.

Tatsächliches Verhalten:
Der PRTG Core Server wird in der CURIX-Konfigurationsliste nicht angezeigt.

Zusätzliche Informationen:
- Log-Datei beigefügt
- Tritt auf in Version 3.4.1
- Keine Auswirkungen auf andere Systeme festgestellt

Lösungsvorschlag:
Überprüfen Sie die CURIX-Konfiguration und stellen Sie sicher, dass der PRTG Core Server korrekt konfiguriert und hinzugefügt wurde.

Affected Systems

- DevelopmentServer-01
- ERPSystem-01

Info

Creator:	User2
Issue Nr.:	11
Duration:	5 h
Timestamp:	15:00, 01.03.24
Last updated:	15:05, 01.03.24
End time:	---

Impact

Systemüberwachungsberichte können nicht generiert werden, was zu einem Mangel an Überwachung der Netzwerkperformance führt

Preventative Measures

Keine vorbeugenden Massnahmen erforderlich

Comments
Attachments

Enter your comments here

Feedback and Close

Abbildung 39: Detailansicht AB-Testing

←
Cancel Save

Title

PRTO Core Server wird in CURIX nicht angezeigt

Alert type: Warning

Severity: Low

Status: Open

Incident type: Configuration

Priority: Medium

Description

Einstellung
 Beim Versuch, den PRTO Core Server in CURIX zu finden, wird dieser nicht angezeigt.

Hintergrundinformationen
 Dieses Problem wurde von User2 gemeldet, der versuchte, den PRTO Core Server in der CURIX-Konfiguration zu finden.

Schritte zur Reproduktion
 1. Öffnen Sie die CURIX-Konfigurationssite.
 2. Suchen Sie nach dem PRTO Core Server.
 3. Überprüfen Sie die Liste der angezeigten Komponenten.

Erwartetes Verhalten
 Der PRTO Core Server sollte in der CURIX-Konfigurationssite angezeigt werden.

Affected Systems

Wähle alle Systeme aus

DevelopmentServer-01	Remove
ERPSystem-01	Remove

Info

Creator:	User2
Issue No.:	11
Duration:	5 h
Timestamp:	15 03, 01.03.24
Last updated:	15 03, 01.03.24
End time:	...

Impact

Systemüberwachungsberichte können nicht generiert werden, was zu einem Mangel an Überwachung der Netzwerkperformance führt

Preventative Measures

Keine vorbeugenden Massnahmen erforderlich

Assignments

Enter your comments here

Abbildung 40: Bearbeitungsansicht AB-Testing

10.5 Feedbackformular Usertesting

- Name: _____
- Datum: _____
- Beruf: _____

10.5.1 Fragen zur Benutzeroberfläche und Navigation

1. Wie intuitiv fandest du die Navigation durch die Anwendung?
 - O Sehr einfach
 - O Einfach
 - O Neutral
 - O Schwierig
 - O Sehr schwierig
2. Wie zufrieden bist du mit der Darstellung der Cards?
 - O Sehr zufrieden
 - O Zufrieden
 - O Neutral
 - O Unzufrieden
 - O Sehr unzufrieden
 - Kommentar: _____
3. Wie bewertest du den Informationsgehalt der Anwendung?
 - O Zu wenig Informationen, ich benötige mehr Details
 - O Genau richtig, die Menge an Informationen ist gut ausbalanciert
 - O Zu viel Informationen, es wirkt überladen

Wie empfindest du die Funktion zum Ein- und Ausklappen von Spalten?

 - Kommentar: _____
4. Gab es Bereiche der Benutzeroberfläche, die du als verwirrend empfandest?
 - O Navigation
 - O Buttons
 - O Icons
 - O Texte
 - O Keine
 - Kommentar: _____
5. Wie verständlich fandest du die verwendeten Icons in der Anwendung?
 - O Sehr verständlich
 - O Verständlich
 - O Neutral
 - O Unverständlich
 - O Sehr unverständlich
 - Kommentar: _____

6. War die Farbauswahl in der Anwendung angenehm und unterstützend für die Benutzererfahrung?
- Sehr angenehm
 - Angenehm
 - Neutral
 - Unangenehm
 - Sehr unangenehm
7. War das Layout der Anwendung verständlich und logisch strukturiert?
- Ja, vollständig
 - Teilweise
 - Nein, es war verwirrend
 - Kommentar: _____

10.5.2 Fragen zu den KI-Funktionen und Feedback-Mechanismen

1. Wie hilfreich fandest du die KI-Funktionen (z.B. Chat-Funktion)?
- Sehr hilfreich
 - Hilfreich
 - Neutral
 - Wenig hilfreich
 - Gar nicht hilfreich
 - Kommentar: _____
2. Waren die KI-Vorschläge klar und verständlich formuliert?
- Ja, vollständig
 - Teilweise
 - Nein
 - Kommentar: _____
3. War die Integration der KI in die Benutzeroberfläche konsistent und nützlich?
- Ja, konsistent und nützlich
 - Konsistent, aber nicht nützlich
 - Nicht konsistent, aber nützlich
 - Weder konsistent noch nützlich
 - Kommentar: _____
4. Fandest du den Zeitpunkt für das Feedback (Wizard, initiales und finales Feedback) passend gewählt?
- Ja, alle drei waren sinnvoll
 - Nur das Wizard und initiale Feedback war sinnvoll
 - Nur das Wizard und finale Feedback war sinnvoll
 - Nur das initiale und finale Feedback war sinnvoll
 - Nur das Wizard Feedback war sinnvoll
 - Nur das initiale Feedback war sinnvoll
 - Nur das finale Feedback war sinnvoll

- O Keines der drei war sinnvoll
5. Wie bewertest du den Nutzen des Feedback-Mechanismus insgesamt?
- O Sehr nützlich
 - O Nützlich
 - O Neutral
 - O Wenig nützlich
 - O Gar nicht nützlich

10.5.3 Fragen zur Benutzerfreundlichkeit und allgemeinen Zufriedenheit

1. Wie zufrieden bist du mit der allgemeinen Benutzerfreundlichkeit der Anwendung?
- O Sehr zufrieden
 - O Zufrieden
 - O Neutral
 - O Unzufrieden
 - O Sehr unzufrieden
2. Hast du während der Nutzung genügend Unterstützung durch Tooltips oder Anleitungen erhalten?
- O Ja, ausreichend
 - O Teilweise
 - O Nein, nicht ausreichend
3. Wie bewertest du die Konsistenz der Anwendung bezüglich Design und Benutzerführung?
- O Sehr konsistent
 - O Konsistent
 - O Neutral
 - O Inkonsistent
 - O Sehr inkonsistent
4. Wie hilfreich fandest du die Verwendung der Satzschablonen (z.B. für Titel und Beschreibung)?
- O Sehr hilfreich
 - O Hilfreich
 - O Neutral
 - O Wenig hilfreich
 - O Gar nicht hilfreich
 - O Kommentar: _____
5. Welche Aspekte der Anwendung hat dir besonders gut gefallen?
- O Design
 - O Benutzerfreundlichkeit
 - O Feedback-Funktion
 - O KI-Funktion
 - O Andere: _____
 - O Kommentar: _____

6. Welche Aspekte der Anwendung sollten deiner Meinung nach verbessert werden?

- Design
- Benutzerfreundlichkeit
- Feedback-Funktion
- KI-Funktion
- Andere: _____
- Kommentar: _____

7. Wie hilfreich fandest du die Anordnung und Funktion der Buttons in der Anwendung?

- Sehr hilfreich
- Hilfreich
- Neutral
- Wenig hilfreich
- Gar nicht hilfreich

8. Wie verständlich war die Terminologie (z.B. Begriffe und Benennungen) in der Anwendung?

- Sehr verständlich
- Verständlich
- Neutral
- Unverständlich
- Sehr unverständlich
- Kommentar: _____

9. Waren die Fehlermeldungen in der Anwendung klar und nachvollziehbar?

- Ja, vollständig
- Teilweise
- Nein
- Kommentar: _____

10. Hast du weitere Kommentare oder Verbesserungsvorschläge zur Anwendung?

- Kommentar: _____

11 Hilfsmittelverzeichnis

Hilfsmittel	Verwendung	Betroffene Stellen
ChatGPT	Verwendung für die Unterstützung bei der Umformulierung und Präzisierung von Textpassagen, zur Verbesserung der Verständlichkeit und Wissenschaftlichkeit der Ausdrucksweise. Verwendung für die effizientere Entwicklung und Fehlerbehebung von Programmierfehlern.	Fliesstext und Programmiercode

Tabelle 4: Hilfsmittelverzeichnis

12 Eigenständigkeitserklärung

Wir, Lorin Desch und Antonio Salvia, erklären hiermit, dass die folgende wissenschaftliche Arbeit mit dem Titel „What’s That Error?“ – let’s make the error messages actionable, unsere eigene Arbeit ist. Wir haben diese Arbeit eigenständig verfasst und unsere Aussagen sowie Ergebnisse basieren auf unseren eigenen Forschungen und Analysen.



Lorin Desch



Antonio Salvia

Quellenverzeichnis

- [1] Titin Rohayatin u. a. „Bureaucratic Reform Strategy In Improving The Quality Of Public Services By Implementing Bureaucratic Communication Model in Cimahi City“. In: *PERSPEKTIF* 11.3 (2022). [Online], S. 963–969. URL: <https://ojs.uma.ac.id/index.php/perspektif/article/view/6264> [26.03.2024].
- [2] Michael Wogalter. „Communication-Human Information Processing (C-HIP) Model in Forensic Warning Analysis: Volume IV: Organizational Design and Management (ODAM), Professional Affairs, Forensic“. In: [Online]. Springer, Cham, Jan. 2019, S. 761–769. URL: https://www.researchgate.net/publication/326850922_Communication-Human_Information_Processing_C-HIP_Model_in_Forensic_Warning_Analysis_Volume_IV_Organizational_Design_and_Management_ODAM_Professional_Affairs_Forensic/citations [26.03.2024].
- [3] Atlassian. *Writing Guidelines: Writing Error Messages*. [Online]. URL: <https://atlassian.design/content/writing-guidelines/writing-error-messages> [22.03.2024].
- [4] Meng Yan u. a. „Revisiting the Correlation Between Alerts and Software Defects: A Case Study on MyFaces, Camel, and CXF“. In: *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*. Bd. 1. [Online]. 2017. URL: <https://ieeexplore.ieee.org/document/8029597/metrics#metrics> [22.03.2024].
- [5] Atlassian. *Writing Guidelines: Writing Error Messages*. [Online]. URL: <https://atlassian.design/content/writing-guidelines/writing-a-warning-message> [22.03.2024].
- [6] Atlassian. *Writing Guidelines: Writing Error Messages*. [Online]. URL: <https://atlassian.design/content/writing-guidelines/writing-a-success-message> [22.03.2024].
- [7] Zhenmin Li u. a. „Have things changed now? an empirical study of bug characteristics in modern open source software“. In: *Proceedings of the 1st Workshop on Architectural and System Support for Improving Software Dependability*. [Online]. San Jose, California: Association for Computing Machinery, 2006. URL: <https://dl.acm.org/doi/abs/10.1145/1181309.1181314> [22.03.2024].
- [8] Dane Bertram. „The social nature of issue tracking in software engineering“. In: *University of Calgary* (2009). [Online]. URL: https://lsmr.org/docs/bertram_msc_2009.pdf [26.03.2024].
- [9] Suzanne Rivard Henri Barki und Jean Talbot. „An Integrative Contingency Model of Software Project Risk Management“. In: *Journal of Management Information Systems* 17.4 (2001). [Online], S. 37–69. URL: <https://doi.org/10.1080/07421222.2001.11045666> [26.03.2024].
- [10] Douglas M Lambert und Martha C Cooper. „Issues in Supply Chain Management“. In: *Industrial Marketing Management* 29.1 (2000). [Online], S. 65–83. URL: <https://www.sciencedirect.com/science/article/pii/S0019850199001133> [26.03.2024].
- [11] John R. Petrocik. „Issue Ownership in Presidential Elections, with a 1980 Case Study“. In: *American Journal of Political Science* 40.3 (1996). [Online], S. 825–850. URL: <http://www.jstor.org/stable/2111797> [26.03.2024].
- [12] Edward J. Banas und W. Frances Emory. „HISTORY AND ISSUES OF DISTANCE LEARNING“. In: *Public Administration Quarterly* 22.3 (1998). [Online], S. 365–383. URL: <http://www.jstor.org/stable/40862326> [26.03.2024].
- [13] Kenneth R. Laughery und Michael Wogalter. „Designing Effective Warnings“. In: *Reviews of Human Factors and Ergonomics* (Apr. 2006). [Online]. URL: <https://www.researchgate.net/publication/258183194> [24.03.2024].
- [14] Tejaswinee Kelkar, Prakash Gadepalli und Bipin Indurkha. „An Assistive Tool to Improve Usability of Error Dialogs“. In: *2013 Conference on Technologies and Applications of Artificial Intelligence*. [Online]. 2013. URL: <https://ieeexplore.ieee.org/document/6783900> [24.03.2024].
- [15] Stephen L. Young und Michael S. Wogalter. „Comprehension and Memory of Instruction Manual Warnings: Conspicuous Print and Pictorial Icons“. In: *Human Factors* 32.6 (1990). [Online], S. 637–649. URL: <https://doi.org/10.1177/001872089003200603> [10.04.2024].
- [16] Titus Barik u. a. „Do Developers Read Compiler Error Messages?“ In: *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*. [Online]. 2017, S. 575–585. URL: <https://ieeexplore.ieee.org/document/7985695> [29.04.2024].
- [17] Palash Bera. „How colors in business dashboards affect users’ decision making“. In: *2024-0Commun. ACM* 59.4 (2016). [Online]. URL: <https://doi.org/10.1145/2818993> [04.05.2024].

- [18] F. Jessen u. a. „The Concreteness Effect: Evidence for Dual Coding and Context Availability“. In: *Brain and Language* 74.1 (2000). [Online], S. 103–112. URL: <https://www.sciencedirect.com/science/article/pii/S0093934X0092340X> [17.05.2024].
- [19] Michael S. Wogalter u. a. „Effectiveness of elevator service signs: Measurement of perceived understandability, willingness to comply and behaviour“. In: *Applied Ergonomics* 28.3 (1997). [Online], S. 181–187. URL: <https://www.sciencedirect.com/science/article/pii/S0003687096000634> [05.05.2024].
- [20] Max-Emanuel Maurer, Alexander De Luca und Sylvia Kempe. „Using data type based security alert dialogs to raise online security awareness“. In: *Proceedings of the Seventh Symposium on Usable Privacy and Security*. [Online]. 2011. URL: <https://doi.org/10.1145/2078827.2078830> [01.04.2024].
- [21] Michael Donhost und Vincent Anfara. „Data-Driven Decision Making“. In: *Middle School Journal* 42 (Nov. 2010). [Online], S. 56–63. URL: https://www.tandfonline.com/doi/pdf/10.1080/00940771.2010.11461758?casa_token=ff2NjKYOnekAAAAA:cLLb22Q4y0j208C6IBqGpBGG7rU309aZU9RvYvLGf-rya2Xd8TTItZ-vtXdWeMu3CYvulK4pZmArmkc [15.04.2024].
- [22] David Bawden und Lyn Robinson. „The dark side of information: overload, anxiety and other paradoxes and pathologies“. In: *Journal of Information Science* 35.2 (2009). [Online], S. 180–191. URL: <https://doi.org/10.1177/0165551508095781> [17.05.2024].
- [23] Emília Duarte u. a. „What should I do? - A study about conflicting and ambiguous warning messages“. In: *Work (Reading, Mass.)* 41 (Jan. 2012). [Online], S. 3633–40. URL: https://www.researchgate.net/publication/220038719_What_should_I_do_-_A_study_about_conflicting_and_ambiguous_warning_messages [01.05.2024].
- [24] Chakkrit Tantithamthavorn, Jirayus Jiarpakdee und John Grundy. „Actionable Analytics: Stop Telling Me What It Is; Please Tell Me What To Do“. In: *IEEE Software* 38.4 (2021). [Online], S. 115–120. URL: <https://ieeexplore.ieee.org/document/9460977> [30.06.2024].
- [25] Paul Denny u. a. „On Designing Programming Error Messages for Novices: Readability and its Constituent Factors“. In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. [Online]. Association for Computing Machinery, 2021. URL: <https://doi.org/10.1145/3411764.3445696> [07.07.2024].
- [26] M. W. van Someren, Y. F. Barnard und J. A. C. Sandberg. *The Think Aloud Method: A Practical Approach to Modelling Cognitive Processes*. [Online]. London, UK: Academic Press, 1994. URL: <https://hdl.handle.net/11245/1.103289> [13.07.2024].