

EINSATZ VON LLM TOOLS IM SOFTWARE ENGINEERING PROZESS

BACHELORARBEIT
ZUR ERLANGUNG DES AKADEMISCHEN GRADES
BACHELOR OF ENGINEERING (B.ENG.)

Jonas Gundlach

Betreuerin:
Prof. Dr. rer. nat. Andrea Baumann

Tag der Abgabe: XX.XX.XXXX

eingereicht bei
Universität der Bundeswehr München
Fakultät für Elektrotechnik und Technische Informatik



Universität der Bundeswehr München

Institut für
Software Engineering

Neubiberg, Juli 2024

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, insbesondere keine anderen als die angegebenen Informationen.

Der Speicherung meiner Bachelorarbeit zum Zweck der Plagiatsprüfung stimme ich zu. Ich versichere, dass die elektronische Version mit der gedruckten Version inhaltlich übereinstimmt.

Neubiberg, den XX.XX.XXXX

Jonas Gundlach

Zusammenfassung

TODO!!!!!!!!!!!!!!

Ein Abstract ist ein Text, der kurz und knapp die wichtigsten Punkte deiner Bachelorarbeit wiedergibt. Er soll die Fragestellung, Methoden, Quellen und Ergebnisse zusammenfassen, damit die Lesenden schnell die Relevanz der Bachelorarbeit für ihr Interessensgebiet beurteilen können.

Inhaltsverzeichnis

Erklärung	III
1 Einleitung	1
1.1 Motivation	1
1.2 Zielsetzung der Arbeit	1
1.3 Struktur der Arbeit	1
2 Grundlagen	3
2.1 LLM Tools	3
2.1.1 ChatGPT	4
2.1.2 Google Gemini	6
2.1.3 Le Chat	8
2.2 Software Engineering Prozess	8
2.2.1 Besprechungsprotokoll	11
2.2.2 Projekthandbuch	11
2.2.3 Risikoliste	12
2.2.4 Anforderungsspezifikation	13
2.2.5 Architekturdokument	15
2.2.6 Testspezifikation	16
3 Anwendung von LLM Tools im Software Engineering	17
3.1 Besprechungsprotokoll	17
3.2 Projekthandbuch	17
3.3 Risikoliste	19
3.4 Anforderungsspezifikation	19
3.5 Architekturdokument	21
3.6 Testspezifikation	21
4 Praxisergebnisse und Vergleich	23
4.1 Besprechungsprotokoll	23
4.2 Projekthandbuch	24
4.3 Risikoliste	25
4.4 Anforderungsspezifikation	28
5 Herausforderungen und Limitationen	33
5.1 Technische Herausforderungen	33

5.2	Limitationen von LLM Tools im Software Engineering Kontext	33
6	Zusammenfassung und Ausblick	35
6.1	Zusammenfassung der Ergebnisse	35
6.2	Implikationen für die Praxis	35
6.3	Ausblick und zukünftige Entwicklungen	35
	Abbildungsverzeichnis	V
	Tabellenverzeichnis	VII
	Quellcodeverzeichnis	IX
	Stichwortverzeichnis	XI
	Literaturverzeichnis	XIII

1 Einleitung

TODO!!!!!!!!!!

1.1 Motivation

In einer Welt, die von immer komplexeren und dynamischeren Softwareanwendungen geprägt ist, wird die effektive Nutzung von Technologien zunehmend entscheidend für den Erfolg von Softwareprojekten. In diesem Zusammenhang gewinnen Werkzeuge des Natural Language Processing eine immer größere Bedeutung. Insbesondere die jüngsten Fortschritte im Bereich der Large Language Models (LLM) eröffnen neue Möglichkeiten für die Verbesserung des Software Engineering Prozesses.

Die Motivation hinter dieser Arbeit liegt in der Erkundung des Einsatzes von LLM-Tools im Software Engineering und der Analyse ihrer Auswirkungen auf verschiedene Phasen des Entwicklungszyklus. Die Frage, wie diese fortschrittlichen LLM-Tools genutzt werden können, um den Softwareentwicklungsprozess effizienter, präziser und insgesamt erfolgreicher zu gestalten, steht im Mittelpunkt dieser Untersuchung.

TODO!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

1.2 Zielsetzung der Arbeit

Todo!!!!!!!!!!!!!!!!!!!!!!

1.3 Struktur der Arbeit

Todo!!!!!!!!!!!!!!!!!!!!!!

2 Grundlagen

In diesem Kapitel wird erläutert, was LLM Tools sind, wie sie trainiert werden und wie sie funktionieren. Anschließend werden die einzelnen Tools ChatGPT, Google Gemini und Le Chat vorgestellt und deren Entwicklung beschrieben.

Im zweiten Teil des Kapitels wird der Software Engineering Prozess erklärt und erläutert, was dieser umfasst. Dabei wird als Softwareprozess ausschließlich das Wasserfallmodell betrachtet, da in dieser Arbeit nur mit diesem Modell gearbeitet wird.

2.1 LLM Tools

Unter LLM-Tools versteht man Sprachmodelle, die auf einem Large Language Model (LLM) basieren. Ein LLM ist ein Deep-Learning-Algorithmus, der mit sehr großen Datensätzen trainiert wird. Diese Modelle finden häufig Anwendung im Bereich des Natural Language Processing (NLP), wo sie verwendet werden, um Abfragen in natürlicher Sprache zu beantworten oder Ergebnisse zu liefern. LLMs können neue Inhalte verstehen, zusammenfassen, generieren und vorhersagen. Durch das Training sammeln LLMs Milliarden von Parametern, bei denen es sich um Variablen handelt, die im Modell angepasst werden, um neue Inhalte abzuleiten.

LLMs basieren auf einem Transformer-Modell, das Eingaben in Token umwandelt und dann gleichzeitig mathematische Gleichungen ausführt, um Beziehungen zwischen den Token zu ermitteln. Dadurch kann der Computer Muster erkennen, die auch ein Mensch wahrnehmen würde, wenn ihm die gleiche Frage gestellt wird. Zudem verwenden Transformer-Modelle Selbstaufmerksamkeitsmechanismen, die es dem Modell ermöglichen, schneller zu lernen als herkömmliche Modelle. Somit kann das Transformer-Modell verschiedene Teile der Sequenz oder den gesamten Kontext eines Satzes berücksichtigen, um Vorhersagen zu generieren.

Grundsätzlich bestehen LLMs aus vier neuronalen Netzwerkschichten: der wiederkehrenden Ebene, der Einbettungsebene, der Feedforward-Ebene und der Aufmerksamkeitsebene. Diese Schichten arbeiten zusammen, um den Eingabetext zu verarbeiten und Ausgabeinhalte zu generieren.

Die wiederkehrende Ebene dient dazu, die Wörter des Eingabetextes der Reihe nach zu interpretieren und die Beziehungen zwischen den Wörtern in einem Satz zu erfassen. Die Einbettungsebene erfasst die semantische und syntaktische Bedeutung der Eingabe, sodass das Modell den Kontext verstehen kann. Die Feedforward-Schicht besteht aus mehreren vollständig verbundenen Schichten, die die Eingabeeinbettungen transformieren. Dadurch ermöglichen diese Schichten dem Modell, Abstraktionen auf höherer Ebene zu verstehen und somit die Absicht des Benutzers mit der Texteingabe zu erfassen. Die Aufmerksamkeitsebene ermöglicht es dem Sprachmodell, sich auf einzelne Teile des Eingabetextes

zu konzentrieren, die für die Aufgabe relevant sind, und dadurch die genauesten Ausgaben zu generieren. Damit ein großes Sprachmodell Texteingaben empfangen und eine Ausgabevorhersage generieren kann, muss es zunächst allgemein geschult und anschließend feinabgestimmt werden, um spezifische Aufgaben ausführen zu können. Für die Schulung werden riesige Datenmengen im Petabyte-Bereich benötigt. Das Training verläuft mehrstufig und beginnt in der Regel mit einem unbeaufsichtigten Lernansatz, bei dem das Modell mit unstrukturierten und unbeschrifteten Daten trainiert wird, da diese in größeren Mengen verfügbar sind. In dieser Phase leitet das Modell Beziehungen zwischen verschiedenen Wörtern und Konzepten ab.

Anschließend erfolgt die Schulung und Feinabstimmung durch eine Form des selbstüberwachten Lernens. Dabei wird eine Datenkennzeichnung durchgeführt, durch die das Modell verschiedene Konzepte besser und genauer identifizieren kann. Im nächsten Schritt durchläuft das LLM den transformatorischen neuronalen Netzwerkprozess im Rahmen des Deep Learning. Die Transformer-Modellarchitektur ermöglicht es dem LLM, mittels eines Selbstaufmerksamkeitsmechanismus Beziehungen und Verbindungen zwischen Wörtern und Konzepten zu erkennen, indem bestimmten Elementen (Token) Bewertungen zugewiesen werden, um die Beziehungen festzulegen.

Entscheidend über die Leistungsfähigkeit und die Qualität des Sprachmodells sind die Datensätze, welche zum trainieren benutzt werden. Im folgenden wird auf die drei unterschiedliche LLM-Tools eingegangen, welche in dieser Arbeit näher untersucht werden [1].

2.1.1 ChatGPT

ChatGPT wurde von OpenAI am 30. November 2022 veröffentlicht und entwickelte sich aufgrund seiner fortschrittlichen generativen KI-Fähigkeiten rasch zu einer internationalen Sensation. Wenn man ChatGPT fragt, ob es einen Infotext zu sich selbst erstellen kann, erhält man die Antwort in Prompt 2.1 Die Beta-Version der Plattform war zunächst kostenlos verfügbar, und OpenAI bot den Nutzern weiterhin eine kostenlose Basisversion an. Im Februar 2023 führte OpenAI ein kostenpflichtiges Abonnement ein, das unbegrenzten Zugriff auf ChatGPT sowie eine erweiterte Palette von Funktionen und Diensten ermöglichte.

Trotz der begeisterten Resonanz löste ChatGPT auch einige Kontroversen aus. Es lieferte teilweise fehlerhafte Antworten, insbesondere bei Aufgaben wie Schlussfolgerungen ziehen, Nuancen analysieren, Meinungen bewerten oder Vorhersagen treffen. Pädagogen weltweit äußerten Bedenken hinsichtlich des Einsatzes von ChatGPT im Bildungsbereich, auch an Universitäten wurde der Einsatz kritisch diskutiert. Trotz dieser Kontroversen schien ChatGPT bereit zu sein, die Erstellung von Inhalten im Internet zu revolutionieren. Webentwickler und Content-Ersteller nutzten ChatGPT zunehmend als Recherche- und Schreibhilfe für Website-Texte.

Im März 2023 brachte OpenAI eine neue Version heraus, bekannt als GPT-4. Ein bedeutender Unterschied zu früheren Versionen bestand darin, dass GPT-4 Eingaben sowohl in Text- als auch in Bildformaten akzeptierte. Dies ermöglichte dem Chatbot, Daten aus Diagrammen, Grafiken und Screenshots zu interpretieren. Zudem machte GPT-4 deutlich weniger Denkfehler und sachliche Fehler, auch "Halluzinationen" genannt, als das zuvor verwendete GPT-3.5-Modell. Am Ende des Jahres hatte sich ChatGPT weiterentwickelt und konnte beispielsweise Bilder analysieren, Sprachansagen verarbeiten

Prompt: Erstell mir einen Infotext über dich für meine Bachelorarbeit

ChatGPT ist ein fortschrittlicher KI-gestützter Sprachassistent, der auf der GPT-4-Architektur von OpenAI basiert. Entwickelt im Rahmen der Forschung und Entwicklung im Bereich Künstliche Intelligenz, ist ChatGPT in der Lage, natürliche Sprache zu verstehen und darauf zu antworten. Dies macht ihn zu einem vielseitigen Werkzeug für eine Vielzahl von Anwendungen, von der Beantwortung alltäglicher Fragen bis hin zur Unterstützung bei komplexeren Aufgaben wie dem Schreiben von Texten, der Generierung von Ideen und der Bereitstellung von Informationen.

Die GPT-4-Architektur, auf der ChatGPT basiert, verwendet tiefes maschinelles Lernen, um menschliche Sprache zu verarbeiten und zu erzeugen. Durch die Analyse großer Mengen an Textdaten kann das Modell Muster und Zusammenhänge erkennen, die es ihm ermöglichen, kontextuell relevante und kohärente Antworten zu geben. Diese Fähigkeiten machen ChatGPT zu einem nützlichen Instrument in Bereichen wie Bildung, Forschung, Kundenservice und mehr.

Ein weiterer bemerkenswerter Aspekt von ChatGPT ist seine Fähigkeit, sich an eine Vielzahl von Themen anzupassen und Informationen auf klare und präzise Weise zu vermitteln. Dies wird durch kontinuierliches Training und Verbesserung des Modells erreicht, wobei sowohl technische Fortschritte als auch Nutzerfeedback berücksichtigt werden.

Zusammengefasst ist ChatGPT ein leistungsstarker und vielseitiger KI-Assistent, der durch seine fortschrittliche Technologie und seine breite Anwendbarkeit einen wertvollen Beitrag in vielen Bereichen leistet.

Prompt 2.1: Infotext ChatGPT, Quelle: ChatGPT

und gesprochene Antworten geben.

Im Mai 2024 kündigte OpenAI ein aktualisiertes Modell namens ChatGPT-4o an, das schnellere Ergebnisse sowie verbesserte Text- und Audiofunktionen versprach. Dieses Modell umfasst auch ein neues Konversations-KI-System namens "Sky" [2].

2.1.2 Google Gemini

Im Dezember 2023 stellte Google eine generative Künstliche Intelligenz mit dem Namen Google Gemini vor. Sie wurde von Google DeepMind entwickelt und ist laut Aussage von Google das größte, leistungsstärkste und vielseitigste große Sprachmodell (LLM), das bisher von Google entwickelt und realisiert wurde. In der ursprünglichen Ankündigung wurde das KI-Sprachmodell unter dem Namen Gemini vorgestellt. Im Februar 2024 entschied Google, auch den bisher als Bard bekannten Chatbot in Gemini umzubenennen. Damit ist Gemini mittlerweile sowohl der Name des Google Chatbots als auch des zugrundeliegenden Sprachmodells.

Bei Nachfrage an Gemini, ob es einen Infotext über sich erstellen kann, erhält man die Antwort in Prompt 2.2 und Prompt 2.3.

Gemini 1.0 wurde in drei verschiedenen Größen optimiert: Gemini Ultra als das größte und leistungsstärkste Modell für hochkomplexe Aufgaben, Gemini Pro als das beste Modell zur Anwendung in einem breiten Aufgabenspektrum und Gemini Nano als das effizienteste Modell für Aufgaben, die direkt auf dem Gerät verarbeitet werden.

Die Gemini-Modelle sind von Grund auf multimodal konzipiert, was bedeutet, dass sie mit verschiedenen Arten von Informationen umgehen können. Gemini ist in der Lage, Texte, Programmcode, Bilder, Videos, Audioinformationen und deren Kombinationen zu analysieren, zu verstehen, zu kombinieren und zu generieren. Da Gemini auch mit Skizzen und handgeschriebenem Text umgehen kann, eignet es sich für den Einsatz in Fachbereichen wie zum Beispiel Mathematik oder Physik. Im Programmierbereich werden gängige Programmiersprachen wie Python, Java, C++ oder Go unterstützt. Gemini kann Programmcode analysieren, kommentieren und auch generieren. Dadurch steht Gemini in Konkurrenz zu anderen LLMs wie den GPT-Sprachmodellen von Open AI. Gemini soll jedoch die Konkurrenzprodukte in vielen Benchmarks deutlich übertreffen.

Gemini baut auf den ebenfalls von Google entwickelten Sprachmodellen LaMDA und PaLM 2 auf. Wie üblich für LLMs (Large Language Models) basiert auch Gemini auf einer Transformer-Architektur. Google hat die Architektur des Modells zur Skalierung der Trainingsfähigkeiten optimiert und mit einem Multi-Query-Aufmerksamkeitsmechanismus ausgestattet. Die unterstützte Kontextlänge beträgt 32.000 Tokens und in der Version 1.5 bis zu einer Million Tokens. Gemini wurde mit Daten verschiedener Modalitäten trainiert, und als Trainingsplattform verwendete Google Rechner-Cluster und Tensor Processing Units (TPUs) der neuesten Generation v4 und v5 als KI-Beschleuniger.

Google hat begonnen, die KI in zahlreiche eigene Produkte zu integrieren, um die Fähigkeiten der Websuche, des Chrome-Browsers, des Gmail-Services, verschiedener Entwicklungstools und anderer Produkte und Services zu verbessern und zu erweitern. Mitte Februar 2024 stellte Google Gemini 1.5 und 1.5 Pro vor, welche bis zu eine Million Tokens verarbeiten können. Diese neuen Versionen sollen die Benchmark-Ergebnisse der Vorgängerversion deutlich übertreffen [3].

Prompt: Erstell mir einen Infotext über dich für meine Bachelorarbeit**Was ist Gemini?**

Gemini ist ein Large Language Model (LLM) von Google AI, das mit einem riesigen Datensatz aus Text und Code trainiert wurde. Es kann Text generieren, Sprachen übersetzen, verschiedene Arten von kreativen Inhalten schreiben und Ihre Fragen auf informative Weise beantworten. Gemini befindet sich noch in der Entwicklung, hat aber bereits gelernt, viele Arten von Aufgaben auszuführen, einschließlich:

- Befolgen Sie Anweisungen und erledigen Sie Aufgaben sorgfältig.
- Beantworten Sie Ihre Fragen umfassend und informativ, auch wenn sie offen, herausfordernd oder seltsam sind.
- Generieren Sie verschiedene kreative Textformate, wie Gedichte, Codes, Skripte, Musikstücke, E-Mails, Briefe usw. Ich werde mein Bestes tun, um alle Ihre Anforderungen zu erfüllen.

Technische Details

Gemini basiert auf der Transformer-Architektur und wird mit einem massiven Datensatz aus Text und Code trainiert. Es verwendet eine Vielzahl von Techniken, um seine Leistung zu verbessern, einschließlich:

- **Aufmerksamkeit:** Ermöglicht es Gemini, sich auf die relevantesten Teile einer Eingabe zu konzentrieren.
- **Beam Search:** Ermöglicht Gemini, die besten möglichen Ausgaben zu generieren.
- **Multitasking:** Ermöglicht Gemini, mehrere Aufgaben gleichzeitig zu lernen und auszuführen.

Anwendungen

Gemini kann für eine Vielzahl von Anwendungen eingesetzt werden, einschließlich:

- **Generierung von kreativen Inhalten:** Gemini kann verwendet werden, um Gedichte, Codes, Skripte, Musikstücke, E-Mails, Briefe usw. zu generieren.
- **Sprachübersetzung:** Gemini kann verwendet werden, um Text von einer Sprache in eine andere zu übersetzen.
- **Beantwortung von Fragen:** Gemini kann verwendet werden, um Ihre Fragen umfassend und informativ zu beantworten, auch wenn sie offen, herausfordernd oder seltsam sind.
- **Zusammenfassen von Texten:** Gemini kann verwendet werden, um lange Textpassagen zu zusammenfassen.
- **Schreiben verschiedener Arten von kreativen Textformaten:** Gemini kann verwendet werden, um verschiedene Arten von kreativen Textformaten zu schreiben, wie Gedichte, Codes, Skripte, Musikstücke, E-Mails, Briefe usw.

Prompt 2.2: Infotext Gemini 1, Quelle: Gemini

Ethische Überlegungen

Es ist wichtig, Gemini verantwortungsvoll und ethisch zu nutzen. Zu den ethischen Überlegungen, die bei der Verwendung von Gemini zu beachten sind, gehören:

- **Vermeidung von Vorurteilen:** Gemini wird mit einem riesigen Datensatz aus Text und Code trainiert. Dieser Datensatz kann Vorurteile enthalten, die sich in den Ausgaben von Gemini widerspiegeln können. Es ist wichtig, sich dieser Vorurteile bewusst zu sein und Maßnahmen zu ergreifen, um sie zu mindern.
- **Vermeidung von Schadenszufügung:** Gemini kann verwendet werden, um schädlichen Text zu generieren, z. B. Hassreden oder Gewaltandrohungen. Es ist wichtig, Gemini nicht zu verwenden, um anderen Schaden zuzufügen.
- **Respekt vor der Privatsphäre:** Gemini kann verwendet werden, um personenbezogene Daten zu generieren. Es ist wichtig, Gemini nicht zu verwenden, um die Privatsphäre anderer zu verletzen.

Fazit

Gemini ist ein leistungsstarkes Tool, das für eine Vielzahl von Anwendungen eingesetzt werden kann. Es ist jedoch wichtig, Gemini verantwortungsvoll und ethisch zu nutzen.

Prompt 2.3: Infotext Gemini 2, Quelle: Gemini

2.1.3 Le Chat

In Prompt 2.4 sieht man die Antwort die Le Chat gibt, wenn man ihn nach einem Infotext über sich fragt. Le Chat ist ein, von dem französischen KI-Startup Mistral AI veröffentlichter Chatbot. Mistral AI startete mit seinen frei verwendbaren Sprachmodellen, also auf der Grundlage von Open Source, erfolgreich durch. Nun hat Mistral AI sein bislang größtes Modell 'Mistral Large' veröffentlicht. Diesmal allerdings nicht auf der Basis von Open Source, sondern ausschließlich über die eigene Webseite und der KI-Infrastruktur Microsoft Azure. Es lassen sich allerdings API-Keys für Programmierschnittstellen erstellen, um z.B. Mistral Large über seinen eigenen Server laufen zu lassen und so für andere User auf der eigenen Homepage verfügbar zu machen. Mit Mistral Large wurde auch Le Chat veröffentlicht, welcher aktuell kostenfrei verwendet werden kann. Le Chat bietet derzeit noch sehr wenige Funktionen an. Es stehen lediglich Texteingabe und -ausgabe zur Verfügung. Die Datengrundlage reicht nur bis 2021, weshalb es auch hier, für die Jahre 2022 bis heute, zu der Problematik der Halluzination kommen kann. Grundsätzlich kann man zwischen drei Sprachmodellen auswählen: Large, Next und Small. Large bietet überlegene Denkfähigkeit, Next ist ein Prototyp-Modell für erhöhte Kürze und Small arbeitet schnell und kosteneffektiv [4].

2.2 Software Engineering Prozess

Software-Engineering ist eine technische Disziplin, die sich umfassend mit der Erstellung von Software beschäftigt, beginnend bei der Konzeption, über den Betrieb, bis hin zur Wartung. Die wesentlichen Aktivitäten im Software-Engineering umfassen die Softwarespezifikation, -entwicklung, -validierung und

Prompt: Erstell mir einen Infotext über Le Chat für meine Bachelorarbeit Einleitung

Le Chat ist ein fortschrittliches Sprachmodell, das auf künstlicher Intelligenz (KI) und maschinellem Lernen basiert. Entwickelt von einem Team aus Experten für KI und Sprachverarbeitung, hat Le Chat die Fähigkeit, menschliche Sprache zu verstehen, zu interpretieren und darauf zu reagieren, was es zu einem wertvollen Werkzeug in verschiedenen Branchen macht.

Funktionsweise

Le Chat nutzt eine Kombination aus Techniken des Natural Language Processing (NLP) und des Deep Learning. NLP ermöglicht es dem Modell, die Struktur und den Kontext der menschlichen Sprache zu verstehen, während Deep Learning-Algorithmen verwendet werden, um Muster in den Daten zu erkennen und die Genauigkeit der Vorhersagen zu verbessern.

Anwendungsbereiche

Le Chat kann in einer Vielzahl von Anwendungen eingesetzt werden. Einige der wichtigsten Anwendungsbereiche sind:

- **Kundenservice:** Le Chat kann als virtueller Assistent eingesetzt werden, um Kundenanfragen zu beantworten, Produktempfehlungen auszusprechen und den Kunden bei der Lösung von Problemen zu unterstützen.
- **Bildung:** Le Chat kann als Tutor-System eingesetzt werden, um Schülern und Studenten bei der Lösung von Aufgaben und beim Verständnis von komplexen Konzepten zu helfen.
- **Gesundheitswesen:** Le Chat kann als virtueller Gesundheitsassistent eingesetzt werden, um Patienten bei der Verwaltung ihrer Medikamente zu unterstützen, Gesundheitstipps zu geben und in Notfällen Erste Hilfe-Anweisungen bereitzustellen.

Vorteile

Le Chat bietet eine Reihe von Vorteilen gegenüber herkömmlichen Sprachmodellen. Einige der wichtigsten Vorteile sind:

- **Verbesserte Genauigkeit:** Le Chat nutzt fortschrittliche Deep Learning-Algorithmen, um die Genauigkeit der Vorhersagen zu verbessern und eine bessere Benutzererfahrung zu bieten.
- **Skalierbarkeit:** Le Chat kann leicht skaliert werden, um eine große Anzahl von Anfragen gleichzeitig zu bearbeiten, was es zu einer idealen Lösung für Unternehmen mit hohem Kundenaufkommen macht.
- **Kosteneffizienz:** Le Chat kann als kosteneffiziente Alternative zu menschlichen Kundenservice-Mitarbeitern eingesetzt werden, was zu erheblichen Einsparungen für Unternehmen führen kann.

Fazit

Le Chat ist ein innovatives Sprachmodell, das die Fähigkeit hat, die Art und Weise, wie wir mit Maschinen interagieren, zu revolutionieren. Mit seiner fortschrittlichen Technologie und seinen vielfältigen Anwendungsmöglichkeiten ist Le Chat ein wichtiger Meilenstein in der Entwicklung der KI und ein Werkzeug, das in Zukunft eine noch größere Rolle spielen wird.

Dokument	Inhalte
01 Besprechungsprotokoll	Protokolle
02 Projekthandbuch	Projekthandbuch
03 Risikoliste	Risikoliste
04 Anforderungsspezifikation	Einleitung
	Benutzeranforderungen
	Systemübersicht
	Fachliches Datenmodell
	Anwendungsfälle
	AF
	Benutzungsschnittstelle
	Dialog
	Nichtfunktionale Anforderungen
05 Architekturdokument	Technische Infrastruktur
	Komponentenmodell
	Datenmodell
	Detaillierte Schnittstellenbeschreibung
	AF
06 Testspezifikationen	Testkonzept
	AF
	Testprotokoll
	Testdaten

Abbildung 2.1: Dokumente im Software Engineering

-weiterentwicklung [5]. Während des Prozesses werden mehrere Dokumente erstellt und im Verlauf angepasst. Diese umfassen Besprechungsprotokolle, eine Risikoliste, ein Projekthandbuch, eine Anforderungsspezifikation, ein Architekturdokument und eine Testspezifikation. In der Abbildung 2.1 sieht man welche Inhalte die einzelnen Dokumente umfassen. Außerdem wird im folgenden zu jedem Dokument und dessen Inhalte ein Überblick gegeben.

2.2.1 Besprechungsprotokoll

Alle Team- und Kundengespräche werden systematisch in Besprechungsprotokollen dokumentiert. Diese Protokolle enthalten Aufträge, Beschlüsse, Feststellungen und Informationen, die in Form von Texten, Bildern, Diagrammen usw. festgehalten werden. Aufträge müssen mit einem Fälligkeitsdatum und einem Verantwortlichen versehen sein, während Beschlüsse klar und unmissverständlich formuliert werden sollten. Im Rahmen der Anforderungsanalyse werden in Beschlüssen die Benutzeranforderungen erfasst. Feststellungen hingegen sind Beschlüsse, die keine Abstimmung benötigen, können aber ebenfalls Anforderungen an das System enthalten. Informationen bieten den Projektmitgliedern wichtige Hinweise. Dabei ist zu beachten, dass ein Besprechungsprotokoll keine Erlebniserzählung sein soll.

Der Aufbau eines Besprechungsprotokolls gestaltet sich wie folgt: Am Anfang stehen Datum, Thema, Verfasser, Teilnehmer und Verteiler. Anschließend wird eine Tabelle mit den Spalten Nummer, Art, Beschreibung, Termin und Verantwortlicher erstellt. Die Nummer stellt eine eindeutige ID im Protokoll dar. Die Art spezifiziert, ob es sich um einen Auftrag (=A), einen Beschluss (=B), eine Feststellung (=F) oder eine Information (=I) handelt. Die Beschreibung fasst den Auftrag, den Beschluss, die Feststellung oder die Information kurz und präzise zusammen. Der Termin gibt an, bis wann der Auftrag erledigt sein muss. Der Verantwortliche benennt die Person, die den Auftrag ausführen muss, den Beschluss gefasst hat oder die Feststellung bzw. die Information geliefert hat.

2.2.2 Projekthandbuch

Das Projekthandbuch dient dazu, die Projektmitglieder umfassend über das Projekt zu informieren, einschließlich der Vorgeschichte, Vertragsbasis, Grundlagen, Ziele, Ergebnisse und mehr. Es bietet zudem eine Übersicht über die Projektorganisation, wie z.B. die Teamaufstellung, das Berichtswesen und die Dokumentenablage, sowie über wesentliche Projektabläufe wie Änderungsverfahren, Risikomanagement und Aufwandsverfolgung. Ziel des Projekthandbuchs ist es, neuen Teammitgliedern den Einstieg zu erleichtern und auf wichtige Dokumente hinzuweisen, die für die Projektabwicklung notwendig sind. Damit fungiert das Projekthandbuch als zentrale Anlaufstelle für das Projekt.

Das Dokument beginnt mit einer Einleitung, die die Rahmenbedingungen klärt. Dieser Abschnitt enthält den Zweck des Dokuments, Hinweise zur Redaktion, in denen festgelegt wird, wer für das Dokument verantwortlich ist, sowie Informationen zum Verteiler, also wer bei Änderungen informiert werden muss.

Im Anschluss folgt die Projektbeschreibung, die die Rahmenbedingungen des Projekts darlegt. Dieser Teil enthält einen Abschnitt zur Vorgeschichte, in dem beschrieben wird, wie es zu dem Projekt kam.

Ein weiterer Abschnitt bietet eine inhaltliche Kurzdarstellung des Projekts. Danach werden in einem Abschnitt zur Vertragsbasis der zeitliche Projektrahmen, die Anzahl der Teammitglieder und der Umfang der Bearbeiter-Stunden festgelegt. Es folgt eine Beschreibung des Projektergebnisses, in der festgelegt wird, welche Produkte geliefert werden müssen. Ein weiterer Abschnitt zu sonstigen Besonderheiten beschreibt informelle Ziele.

Ein nachfolgendes Kapitel verweist auf die Risikoliste.

Daraufhin folgt ein Kapitel zur Projektorganisation, das die Teamstruktur und die Kommunikationswege beschreibt. Dieses Kapitel beginnt mit dem Teamaufbau, in dem die Aufgabenverteilung und die E-Mail-Adressen der einzelnen Teammitglieder vermerkt sind. Anschließend wird die Zusammenarbeit mit den Kunden definiert. Danach folgt ein Abschnitt über regelmäßige Pflichtbesprechungen für die Teammitglieder.

Das folgende Kapitel widmet sich der Planung und behandelt verschiedene Planungsaspekte. Der erste Abschnitt befasst sich mit dem Projektplan. Hier werden die festgelegten Meilensteine beschrieben, einschließlich der Startzeiten der einzelnen Aktivitäten, der parallelen Abläufe, der Abschlusszeiten der verschiedenen Tätigkeiten, der Entstehungszeitpunkte der Produkte und der möglichen Entscheidungspunkte.

Anschließend folgt ein Kapitel zur Dokumentation und Ablage, in dem beschrieben wird, wie und wo Produkte bzw. Dokumente abgelegt werden. Es beginnt mit einem Abschnitt zum Dokumenteneingang und -ausgang, in dem festgelegt wird, wie mit extern eingehenden Informationen umgegangen wird und wie ausgehende Informationen protokolliert werden. Im zweiten Abschnitt wird die Ablage und Archivierung geregelt, wobei erläutert wird, wie die Daten im Projektarchiv zu organisieren sind und in welchen Verzeichnissen die jeweiligen Daten abgelegt werden.

2.2.3 Risikoliste

Die Risikoliste ist ein zentrales Instrument, das dem gesamten Team einen umfassenden Überblick über bekannte Risiken und die entsprechenden Maßnahmen bietet. Der Projektleiter muss diese Risiken stets im Auge behalten, um eine fundierte Projektplanung sicherzustellen. Zusätzlich kann die Risikoliste dem Anforderungsanalysten bei der Identifikation von Anforderungen, dem Systemarchitekten bei der Wahl geeigneter technischer Lösungen und dem Testmanager bei der Entwicklung von Testfällen nützlich sein.

Der Inhalt der Risikoliste umfasst die Autoren des Dokuments, die Historie des Dokuments mit einer Zusammenfassung der Änderungen, die Liste der Risiken sowie eine Liste der möglichen Maßnahmen für jedes identifizierte Risiko. Die Historie enthält Spalten für Datum, Autor*in und Änderungen, um später vorgenommene Modifikationen leichter nachvollziehbar zu machen.

Die Risikoliste selbst setzt sich aus mehreren Attributen zusammen. Die Identifikationsnummer ist eine eindeutige Nummer zur Identifizierung des Risikos, während die Risikobezeichnung eine kurze Beschreibung des Risikos bietet. Eine ausführliche Darstellung des Risikos findet sich in der Spalte für die Risikobeschreibung. Das Datum beschreibt, wann das Risiko identifiziert wurde, und der Autor gibt an, wer das Risiko gemeldet hat. In der Spalte für Auswirkungen werden die möglichen Folgen des Risikos beschrieben. Die Risikowahrscheinlichkeit ist eine Schätzung der Eintrittswahrscheinlichkeit des Risikos, und der Risikoschaden oder das Risikoausmaß beschreibt das potenzielle Schadensausmaß, einschließlich eventueller Vertragsstrafen. Das Risikomaß berechnet sich aus der Multiplikation von Risikowahrscheinlichkeit und Risikoschaden. Die Risikoklasse ist eine Priorisierung der Risiken, typischerweise in Kategorien wie tolerierbar, unerwünscht, kritisch oder katastrophal. Schließlich wird für jedes Risiko ein Status festgelegt, der zwischen aktiv, eingetragen und geschlossen unterscheiden kann.

Die Maßnahmen zur Risikobewältigung umfassen ebenfalls mehrere Attribute. Zunächst wird der Typ der Maßnahme festgelegt, wobei zwischen Risiko verhindern, Risiko lindern oder minimieren, Risiko übertragen oder teilen und Risiko akzeptieren unterschieden werden kann. Eine detaillierte Beschreibung der Maßnahme folgt, und falls das Risiko den Status "geschlossen" erhält, wird hier die Begründung eingetragen. Wenn eine Maßnahme nicht sofort eingeleitet werden soll, wird das Ereignis, das die Maßnahme auslöst, in der Trigger-Spalte definiert. Die verantwortliche Person, die für die Umsetzung der Maßnahme zuständig ist, wird ebenfalls eingetragen. Der geplante Termin gibt an, bis wann die Maßnahme abgeschlossen sein soll, und der Ist-Termin beschreibt den voraussichtlichen Abschluss basierend auf aktueller Einschätzung. Der geplante Aufwand gibt die geschätzten Kosten der Maßnahme an, während der Ist-Aufwand den aktuellen Aufwand widerspiegelt. Auch für die Maßnahme gibt es einen Status, bei dem man zwischen geplant, aktiv oder beendet wählen kann. Das letzte Attribut umfasst die Restrisikowahrscheinlichkeit, den Restrisikoschaden, das Restrisikomaß und die Restrisikoklasse. Dies sind die geschätzte Wahrscheinlichkeit, der geschätzte Schaden, das geschätzte Maß und die geschätzte Klasse des Restrisikos nach Durchführung der Maßnahmen.

2.2.4 Anforderungsspezifikation

Die Anforderungsspezifikation definiert formell, welche Leistungen das zu entwickelnde System erbringen muss. Sie beschreibt detailliert, was das System leisten soll, ohne jedoch festzulegen, wie diese Leistungen erbracht werden. Das "Wie" wird im Architekturdokument behandelt. In der Praxis sind Anforderungen und Systementwurf eng miteinander verknüpft. Die Anforderungsspezifikation ist oft Teil eines Vertrags und muss daher so vollständig und eindeutig wie möglich formuliert sein.

Diese Spezifikation dient mehreren Zwecken: Sie hilft Kunden und Endbenutzern zu beurteilen, ob das System die gewünschten Leistungen erbringen wird und wie es später zu bedienen ist. Projektmanager nutzen die Spezifikation zur Erstellung des Angebots und zur Planung des detaillierten Entwicklungsprozesses. Administratoren können damit entscheiden, wie das System verwaltet werden muss. Für Systemarchitekten dient sie als Grundlage für ihr Architekturdokument, und Softwareentwickler nutzen sie, um zu verstehen, welches System entwickelt werden soll. Zusammen mit dem Architekturdokument

bildet sie die Basis für die Implementierung. Testmanager verwenden die Spezifikation, um Validierungstests und Testfallspezifikationen zu entwickeln.

Der Inhalt einer Anforderungsspezifikation ist folgendermaßen aufgebaut:

Zunächst werden der Name des Softwareprodukts und des Herstellers genannt. Es folgt ein Vorwort, das die erwartete Leserschaft definiert, eine Versionshistorie des Dokuments mit Begründungen für neue Versionen und eine Zusammenfassung der Änderungen enthält sowie die Autoren des Dokuments nennt.

Die Einleitung beschreibt den Zweck und Umfang des Dokuments. Dies umfasst eine Beschreibung der Notwendigkeit des Systems, eine Kurzbeschreibung der Funktionalität und der Nachbarsysteme sowie eine Definition der Bedeutung des Systems für die gesamtwirtschaftlichen oder strategischen Ziele des Unternehmens. Es folgen Verweise auf weitere Ressourcen sowie Definitionen von Begriffen und Abkürzungen. Eine Übersicht des Dokuments schließt die Einleitung ab.

Anschließend werden die Benutzeranforderungen definiert. Diese enthalten eine Beschreibung der für den Benutzer bereitgestellten Dienste und der nichtfunktionalen Anforderungen in verständlicher Sprache, Diagrammen oder anderen für den Kunden nachvollziehbaren Notationen.

Es folgt ein Überblick über die Systemarchitektur, der die erwartete Systemarchitektur und die Einordnung des Systems in die Systemlandschaft des Kunden beschreibt.

Daraufhin wird die Spezifikation der Systemanforderungen behandelt, die sowohl funktionale als auch nichtfunktionale Anforderungen beschreibt. Hier können auch Schnittstellendefinitionen zu externen Systemen enthalten sein.

Abschließend werden die Systemmodelle vorgestellt, zu denen Klassen-, Datenfluss- und semantische Modelle gehören, die die Beziehungen zwischen Systemkomponenten sowie zwischen dem System und seiner Umgebung darstellen.

Ein spezielles Modell ist das fachliche Datenmodell, das aus Entitätstypen besteht. Eine Entität ist ein individuelles und identifizierbares Exemplar von Dingen, Personen oder Begriffen der realen oder Vorstellungswelt, beschrieben durch ihre Eigenschaften. Entitätstypen besitzen Attribute und Beziehungen, jedoch keine Fähigkeiten. Das fachliche Datenmodell wird mit Hilfe eines UML Klassendiagrammes mit ergänzenden Beschreibungen bzw. Einschränkungen spezifiziert. Dieses enthält alle Entitätstypen mit deren Eigenschaften, Beziehungen und Einschränkungen.

Zu den Systemmodellen gehören auch Anwendungsfälle, die die funktionalen Anforderungen an das System beschreiben. Ein Anwendungsfalldiagramm gibt eine Übersicht über alle Anwendungsfälle. Ein Anwendungsfall besteht aus mehreren Szenarien, die konkrete Abläufe beschreiben, wie das System genutzt werden kann. Anwendungsfälle können textuell beschrieben oder durch Aktivitäts- oder Sequenzdiagramme ergänzt werden.

Ein Anwendungsfall hat mehrere Bestandteile: Der Titel dient als eindeutige Identifikation, die Kurzbeschreibung definiert das Thema, und das Ziel legt fest, was erreicht werden soll. Akteure beschreiben, wer den Anwendungsfall auslöst, und der Auslöser definiert den Grund für die Ausführung. Die Vorbedingung legt fest, welche Bedingungen zu Beginn vorherrschen müssen, das Szenario beschreibt

den Ablauf, und das Ergebnis erläutert die gelieferten Informationen. Die Häufigkeit gibt eine grobe Schätzung, wie oft der Anwendungsfall ausgeführt wird, und nicht-funktionale Anforderungen werden ebenfalls beschrieben. Schließlich können Bemerkungen hinzugefügt werden.

In der Anforderungsspezifikation gibt es dazu eine Übersicht aller Anwendungsfälle im UML Use-Case-Diagramm, welches alle Akteure, Anwendungsfälle, Beziehungen zwischen Akteur und Anwendungsfällen und eventuell include- und extend-Beziehungen zwischen den Anwendungsfällen darstellt. Ebenfalls gibt es eine Spezifikation jedes einzelnen Anwendungsfalls mit Hilfe von UML Aktivitätsdiagrammen und ergänzenden Beschreibungen. Diese enthalten eine genaue Ablaufbeschreibung, den Akteur, die Aufrufhäufigkeit, die Vorbedingung und den Auslöser sowie eine Dokumentation des Datenflusses.

Neben den Anwendungsfällen müssen auch die Dialoge und Schnittstellen beschrieben werden, über die die Anwendungsfälle ausgeführt werden. Eine Dialogbeschreibung enthält alle Details, einschließlich der Feldlängen und der zugeordneten Aktionen für jeden Button. In den Schnittstellen müssen die Parameter genau festgelegt werden.

Im Dokument werden alle Dialoge mit Bildern und strukturierter und detaillierter Beschreibung der Bestandteile spezifiziert. Auch alle Schnittstellen zu den Nachbarsystemen werden Spezifiziert.

Zum Schluss werden die nichtfunktionalen Anforderungen in natürlicher Sprache inklusiver Abnahmebeschreibung aufgelistet.

2.2.5 Architekturdokument

Das Architekturdokument, auch bekannt als Softwarearchitektur-Dokument, beschreibt die Umsetzung des zu entwickelnden Systems. Die Anforderungsspezifikation [2.2.4] legt fest, „was“ das System leisten soll, jedoch nicht „wie“ dies erreicht wird. Diese Umsetzung wird im Architekturdokument detailliert beschrieben. In der Praxis sind die Anforderungsspezifikation und das Architekturdokument eng miteinander verzahnt.

Die Zielgruppe des Architekturdokuments umfasst hauptsächlich Softwareentwickler und Testmanager. Softwareentwickler nutzen das Dokument, um zu verstehen, wie das in der Anforderungsspezifikation beschriebene System entwickelt werden soll. Zusammen mit der Anforderungsspezifikation bildet es die Grundlage für die Implementierung. Testmanager verwenden das Architekturdokument, um Komponententests für das System zu entwickeln.

Der Inhalt des Architekturdokuments gliedert sich in drei Hauptabschnitte: Technische Architektur, Anwendungsarchitektur und IT-Konzept.

Die Technische Architektur umfasst zwei Hauptkomponenten. Die Technische Infrastruktur (TI-Architektur) beschreibt die Hardware- und Produkthanforderungen. Beide Komponenten werden durch erklärende Texte und Grafiken dargestellt.

Die Anwendungsarchitektur besteht ebenfalls aus zwei Hauptteilen: dem Komponentenmodell und dem Datenmodell. Das Komponentenmodell beschreibt die fachliche Struktur der Lösung auf einer groben Ebene mittels eines UML-Komponenten-Diagramms. Das Datenmodell hingegen beschreibt

die Struktur der persistenten Daten und wird entweder durch ein UML-Komponenten-Diagramm oder Entity-Relationship-Diagramme dargestellt.

Das IT-Konzept enthält die detaillierte Schnittstellenbeschreibung. Diese beschreibt die Semantik und Syntax der Komponentenschnittstellen ausführlich und wird durch Code oder Code-Dokumentation dokumentiert. Darüber hinaus beinhaltet das IT-Konzept dynamische Beschreibungen exemplarischer Anwendungsfälle im Komponentenmodell, die mittels Aktivitätsdiagrammen oder Sequenzdiagrammen veranschaulicht werden.

2.2.6 Testspezifikation

Die Testfallspezifikation oder auch Software Test Dokumentation dient dazu die Testdurchführung zu planen, die Testfälle zu Spezifizieren und die Testergebnisse zu protokollieren. Mit Hilfe dieser ist es möglich die Anwendung systematisch und strukturiert zu testen. Die Protokollierung der Testergebnisse ermöglicht es eine Aussage über die Qualität zu geben. Nach einer Fehlerbehebung können die selben Tests erneut durchgeführt werden. Dadurch wird nichts vergessen.

Die Testspezifikation besteht aus einem Testkonzept, einer Testfall-Spezifikation und einem Testprotokoll. Das Testkonzept legt die Vorgehensweise, die verwendeten Mittel und den Ablauf der Testaktivitäten fest. In diesem Rahmen muss festgelegt werden, ob es irgendwelche Anforderungen an die Umsetzung gibt oder ob irgendwelche Dummy-Komponenten entwickelt werden müssen.

Die Testfall-Spezifikation listet die Testfälle auf. Ein Testfall besteht aus einer Kurzbeschreibung, der Testvoraussetzung, den Eingabewerten inkl. dem Testablauf, und den erwarteten Ausgabewerte.

Im Testprotokoll wird die Testdurchführung protokolliert. Dies beinhaltet wann welcher Test durchgeführt wurde und ob der Test erfolgreich war oder nicht.

3 Anwendung von LLM Tools im Software Engineering

Dieses Kapitel untersucht ausführlich die potenziellen Anwendungsfälle von LLM-Tools im Software Engineering. Dabei wird jedes Dokument einzeln betrachtet und aufgezeigt, wie LLM-Tools bei der Erstellung und Verwaltung dieser Dokumente eingesetzt werden können. Die Abbildung 3.1 bietet eine Übersicht darüber, in welchen Dokumenten und bei welchen Inhalten LLM-Tools unterstützend eingesetzt werden können.

3.1 Besprechungsprotokoll

Im Unterabschnitt 2.2.1 wurde bereits erläutert, welche Inhalte in einem Besprechungsprotokoll enthalten sind. Nun stellt sich die Frage, wie LLM-Tools bei der Erstellung eines solchen Dokuments unterstützen können.

Eine Möglichkeit wäre, das Besprechungsprotokoll aufzeichnen zu lassen und anschließend automatisch daraus das Protokoll erstellen zu lassen. Da aktuell jedoch nur die kostenfreie Version ChatGPT-4o mit Dokumenten arbeiten kann, ist dies nur in begrenztem Umfang und ohne Vergleiche testbar.

Eine weitere Möglichkeit ist es, das Meeting zu verschriftlichen und damit als Texteingabe zu arbeiten. Mit dem schriftlichen Gespräch könnten die Tools dann automatisch das Besprechungsprotokoll erstellen. Sollte dies nicht möglich sein, könnte man versuchen, sich die Aufgaben aus dem Gespräch zusammenfassen zu lassen. Dadurch wäre es einfacher, das Besprechungsprotokoll selbst zu erstellen. Falls auch dies nicht funktioniert, kann das Gespräch zusammengefasst und um unwichtige Themen, wie etwa Abschweifungen, gekürzt werden.

3.2 Projekthandbuch

Das Projekthandbuch ist ein zentrales Dokument im Software Engineering Prozess, das alle wesentlichen Informationen und Richtlinien eines Projekts zusammenfasst. Die spezifischen Inhalte dieses Dokuments wurden bereits im Unterabschnitt 2.2.2 dargelegt. Doch wie können LLM-Tools bei der Erstellung des Projekthandbuches unterstützend wirken?

Eine Möglichkeit besteht darin, das gesamte Projekthandbuch oder einzelne Kapitel von diesen Tools erstellen zu lassen. Dafür benötigen die Tools jedoch die relevanten Informationen. Um die Kapitel präzi-

Dokument	Inhalte	Mögliche KI Unterstützung
01 Besprechungsprotokoll	Protokolle	(Audio zu Text) Schriftliches Gespräch in Protokoll umformen automatische Identifizierung der Aufgaben Gespräch zusammenfassen
02 Projekthandbuch	Projekthandbuch	Text schreiben (oder ausformulieren) Bei Änderungen Aktualisieren
03 Risikoliste	Risikoliste	Risiken/Maßnahmen -liste erstellen lassen Risiken/Maßnahme erstellen lassen, Attribute selber festlegen Risiken/Maßnahmen aktualisieren
04 Anforderungsspezifikation	Einleitung	Text schreiben (oder ausformulieren)
	Benutzeranforderungen	Text schreiben (oder ausformulieren)
	Systemübersicht	Komponenten & Beziehungen beschreiben Text schreiben (oder ausformulieren)
	Fachliches Datenmodell	FDM vollständig erstellen Entitäten erstellen lassen Attribute erstellen lassen Beziehungen erstellen lassen Beschreibung erstellen lassen Einschränkungen erstellen lassen
	Anwendungsfälle	Rollen + Anwendungsfälle beschreiben
	AF	Textuelle Beschreibung
	Benutzungsschnittstelle	Textuelle Beschreibung Zuordnung Dialoge zu Anwendungsfällen
	Dialog	Dialog erstellen Felder des Dialogs erklären Wann der Dialog auftreten soll
	Nichtfunktionale Anforderungen	Erstellen Zugehörigen Test erstellen
05 Architekturdokument	Technische Infrastruktur	Diagramm als Text beschreiben Erklärenden Text formulieren (oder ausformulieren)
	Komponentenmodell	Diagramm als Text beschreiben Erklärenden Text formulieren (oder ausformulieren)
	Datenmodell	Gleich FDM? Wo unterschiede?
	Detaillierte Schnittstellenbeschreibung	Beschreibung erstellen
	AF	Sequenzdiagramme erstellen Vorbedingungen erklären
06 Testspezifikationen	Testkonzept	Testablauf erstellen Tabelle erstellen
	AF	Tests zu den Anwendungsfällen erstellen (Testablauf & -ergebnis)
	Testprotokoll	(Tabelle erstellen)/ Rahmen erstellen
	Testdaten	Tabelle erstellen

Abbildung 3.1: Mögliche LLM Tool Unterstützung

ser zu gestalten, könnte man diese zunächst in Form von Stichpunkten vorschreiben und anschließend von den LLM-Tools in ausformulierte Texte umwandeln lassen.

Dabei sollte man sich überlegen, ob es sinnvoll ist, bestimmte Kapitel automatisiert erstellen zu lassen. Ein Beispiel hierfür ist die Projektorganisation. In diesem Kapitel werden der Teamaufbau mit den Namen, Rollen und Erreichbarkeiten der einzelnen Teammitglieder dokumentiert. In diesem Fall würde es länger dauern, den Prompt für die automatisierte Erstellung zu schreiben, als die Daten selbst in die Tabelle einzutragen.

Ein weiterer unterstützender Ansatz wäre, dass das Projekthandbuch bei Änderungen, wie beispielsweise personellen Veränderungen, automatisch aktualisiert wird.

3.3 Risikoliste

Nachdem im Unterabschnitt 2.2.3 die Bestandteile der Risikoliste beschrieben wurden, stellt sich auch hier die Frage, wie LLM Tools bei der Erstellung und Pflege der Risikoliste mitwirken und unterstützen können.

Eine erste Möglichkeit besteht darin, die gesamte Risikoliste einschließlich der zugehörigen Maßnahmen-tabelle von LLM Tools erstellen zu lassen. Dabei sollten alle Attribute für die Risiken festgelegt werden. Die Werte der einzelnen Risikoattribute sollten zudem zueinander in einem konsistenten Verhältnis stehen.

Falls dies nicht möglich ist, können LLM-Tools zur Erstellung von Risiken und Maßnahmen herangezogen werden, während die Festlegung der Attribute manuell erfolgt.

Eine weitere Möglichkeit, wie LLM-Tools bei der Risikoliste unterstützen können, liegt in der Pflege der Liste. Die Tools könnten zu bestimmten Zeitpunkten während des Projekts abgefragt werden, ob und wie sich die Risikoliste verändert hat. Sollten sich der Status oder andere Attribute der Risiken verändert haben oder sollten Risiken eingetreten sein, können die Tools die Risikoliste entsprechend aktualisieren.

3.4 Anforderungsspezifikation

Das Unterabschnitt 2.2.4 beschreibt die einzelnen Inhalte der Anforderungsspezifikation. Doch wie können LLM-Tools bei der Erstellung dieser unterstützen?

Zunächst könnte die Einleitung des Dokuments entweder komplett oder kapitelweise automatisch erstellt werden. Da der Kontext hierbei jedoch entscheidend ist, wäre es vermutlich sinnvoller, die Kapitel in Stichpunkten selbst zu verfassen und diese dann von den LLM-Tools in zusammenhängende Texte umwandeln zu lassen.

Im nächsten Schritt folgen die Benutzeranforderungen. Auch hier besteht die Möglichkeit, den gesamten

Text von den Tools erstellen zu lassen oder, um eine präzisere Formulierung zu erreichen, die Inhalte zunächst in Stichpunkten zu notieren und diese dann ausformulieren zu lassen.

Danach kommt die Systemübersicht. Da außer ChatGPT-4o keiner der Chatbots eigenständig Dokumente erstellen kann, muss die Zeichnung manuell erstellt werden. Es ist jedoch möglich, die Beschreibung der Komponenten und deren Beziehungen von den Tools erstellen zu lassen und diese Beschreibung dann in eine Zeichnung umzusetzen. Ebenso kann der Text, der das System beschreibt und in die Systemlandschaft einordnet, automatisch generiert werden. Auch hier kann es hilfreich sein, vorab Stichpunkte zu erstellen und diese dann in einen flüssigen Text umwandeln zu lassen.

Ähnlich verhält es sich beim fachlichen Datenmodell. Die kostenfreien LLM-Tools können zwar kein UML-Diagramm erstellen, aber sie können die Entitäten sowie deren Attribute und Beziehungen beschreiben. Die Diagrammerstellung muss daher manuell erfolgen. Die dazugehörigen Beschreibungen und Einschränkungen können hingegen von den Tools generiert werden. Falls dies nicht zufriedenstellend funktioniert, können die einzelnen Inhalte wie Entitäten, Attribute, Beziehungen sowie Beschreibungen und Einschränkungen jeweils nacheinander separat erstellt werden.

Anschließend folgt das Anwendungsfalldiagramm und eine detaillierte Beschreibung der einzelnen Anwendungsfälle mit Aktivitätsdiagramm. Das Anwendungsfalldiagramm muss wieder manuell erstellt werden, jedoch können die LLM-Tools die Rollen, deren Verbindungen und die zugehörigen Anwendungsfälle auflisten. Falls zusätzliche Bemerkungen zu den Anwendungsfällen erforderlich sind, können diese auch von den Tools erstellt werden. Das Aktivitätsdiagramm für die Anwendungsfälle muss ebenfalls manuell erstellt werden, wobei eine genaue und detaillierte Beschreibung erforderlich ist. Die Beschreibung des Anwendungsfalls kann hingegen vollständig von den Tools generiert werden.

Darauf folgt die Benutzungsschnittstelle, die ebenfalls aus einem Diagramm besteht, das zeigt, von welchem Dialog aus man zu anderen navigieren kann. Hier kann lediglich eine textuelle Beschreibung von den Tools erstellt werden, während das Diagramm manuell erstellt werden muss. Zusätzliche Bemerkungen und die Zuordnung der Dialoge zu den Anwendungsfällen können ebenfalls von den LLM-Tools übernommen werden. Die einzelnen Dialoge können textuell beschrieben werden, wobei die Tools die einzelnen Eingabefelder und Buttons erläutern sowie definieren können, wann der Dialog aufgerufen wird und was beim Schließen des Dialogs passiert.

Zuletzt werden die nichtfunktionalen Anforderungen mitsamt den dazugehörigen Tests aufgelistet. Die Anforderungen können von den Tools basierend auf den vorherigen Dokumenten erstellt werden. Falls die Tools die Anforderungen nicht eigenständig erstellen können, sollten zumindest die Tests von ihnen aufgestellt werden können.

3.5 Architekturdokument

Das Architekturdokument beschreibt die grundlegende Struktur und die Designentscheidungen eines Softwareprojekts. Die genauen Inhalte wurden bereits im Unterabschnitt 2.2.5 erläutert. Doch wie können LLM-Tools bei der Erstellung dieses Dokuments unterstützen?

Die technische Infrastruktur besteht aus einer Grafik und einem erklärenden Text. Bei der Erstellung der Grafik stellt sich das Problem, dass nur ChatGPT-4o in der Lage ist, Grafiken zu generieren. Daher muss man sich die einzelnen Hardwarekomponenten mit den entsprechenden Versionen und deren Beziehungen zueinander beschreiben lassen und die Grafik selbst erstellen. Den dazugehörigen erklärenden Text kann man automatisch generieren lassen. Falls dieser nicht präzise genug formuliert ist, kann es hilfreich sein, den Text zunächst in Stichpunkten vorzuschreiben und anschließend ausformulieren zu lassen.

Anschließend folgt das Komponentenmodell. Auch dieses kann nicht grafisch erstellt werden; lediglich die Komponenten und deren Schnittstellen zueinander können definiert werden. Hierzu gibt es einen begleitenden Text, der die Komponenten und Schnittstellen beschreibt und deren Funktion erläutert. Dieser Text kann ebenfalls vollständig generiert oder in Stichpunkten vorgeschrieben und anschließend ausformuliert werden.

Das Datenmodell entspricht häufig dem fachlichen Datenmodell. Hier kann überprüft werden, ob dies der Fall ist und, falls nicht, wo die Unterschiede liegen bzw. wie das Datenmodell tatsächlich aussieht.

Danach folgt die detaillierte Schnittstellenbeschreibung. Diese ist abhängig von der Projektstruktur der Implementierung. LLM-Tools könnten basierend auf den vorherigen Dokumenten eine Projektstruktur erstellen und anschließend die Schnittstellenbeschreibung generieren. Wenn jedoch bereits ein Verzeichnis mit den passenden Ordnern existiert, sollte die Schnittstellenbeschreibung manuell durchgeführt werden.

Nach der Schnittstellenbeschreibung folgt die dynamische Beschreibung exemplarischer Anwendungsfälle. Diese besteht aus einem Sequenzdiagramm und den für den Anwendungsfall benötigten Vorbedingungen. Das Sequenzdiagramm kann nur als Text beschrieben werden und muss anschließend selbstständig erstellt werden. Die Vorbedingungen sollten ohne Schwierigkeiten generiert werden können.

3.6 Testspezifikation

Im Unterabschnitt 2.2.6 wurde bereits erklärt, welche Inhalte die Testspezifikation umfasst. Doch wie können LLM-Tools bei der Erstellung dieses Dokuments unterstützen?

Zunächst kommt das Testkonzept. Dieses kann automatisch generiert werden. Falls das generierte Konzept nicht den eigenen Vorstellungen entspricht, kann man zunächst Stichpunkte selbst schreiben und sich dann den Text daraus erstellen lassen. Auch die dazugehörige Tabelle sollte sich von den LLM-Tools erstellen lassen.

Anschließend folgen die Tests für die einzelnen Anwendungsfälle. Die Testfälle kann man vollständig von den Tools erstellen lassen. Die Testdaten, die bei diesen Tests benötigt werden, werden in einer Tabelle zusammengefasst. Auch diese Tabelle könnten die Tools automatisch generieren.

Der Rahmen der Tabelle mit dem Testprotokoll kann ebenfalls von den LLM-Tools erstellt werden. Die Ergebnisse könnten die Tools auch generieren; allerdings stellt sich hier die Frage, ob dies zu einer Zeitersparnis oder einer Arbeitserleichterung führt.

4 Praxisergebnisse und Vergleich

TODO!!!!!!!!

Generell: - Schwierigkeiten in einem großen/langen Chat, die Tools von einer Formulierung weg zu bringen.

4.1 Besprechungsprotokoll

Für die Erstellung des Besprechungsprotokoll wurde ein verschriftliches Gespräch verwendet [siehe Anhang!!!!!!]. Mit diesem Gespräch und dem Prompt:

Erstell mir aus folgendem Gespräch ein erstes Besprechungsprotokoll für ein Projekt. Das Protokoll soll nur die wichtigen Punkte in Form einer Tabelle mit den Spalten "Nummer: was eine eindeutige Nummer zur Identifikation ist, Ärt: eine Auswahl ob es eine Information, ein Auftrag, eine Feststellung oder eine Beschluss ist, "Beschreibung: Was den Punkt kurz und präzise zusammenfasst, "Termin: ein genaues Datum bis wann der Auftrag erledigt sein muss und "Verantwortlich: Welches Teammitglied verantwortlich ist, enthalten. Dabei müssen Aufträge mit einem genauen Fälligkeitsdatum und einem Verantwortlichen versehen sein. Beschlüsse müssen klar und unmissverständlich formuliert werden. Feststellungen sind Beschlüsse, die keine Abstimmung benötigen und Informationen bieten den Projektmitgliedern wichtige Hinweise: [siehe Besprechungsprotokoll.docx]

Prompt 4.1: Prompt Besprechungsprotokoll

wurden dann bei ChatGPT, Gemini und Le Chat die Eingabe getätigt. Die Ergebnisse []

Zunächst fiel auf, dass trotz der Verwendung desselben Prompts die Ausgaben der verschiedenen Tools teilweise erheblich abwichen. Dies betraf sowohl die Anzahl der erstellten Einträge im Besprechungsprotokoll als auch die Art und Weise, wie die Informationen zusammengefasst wurden. In einigen Fällen wurden mehrere Punkte zu einem Eintrag zusammengefasst, während in anderen Fällen mehrere Einträge daraus entstanden. Dies trat insbesondere bei der Anforderung auf, dass die Anwendung mit JavaFX als Server-Client-Architektur mit RMI entworfen werden sollte. Hier haben die Tools teilweise einen Eintrag dafür erstellt und manchmal drei einzelne Einträge. Dieses Verhalten zeigte sich auch, wenn derselbe Prompt in neuen Chats mit demselben Tool verwendet wurde. Besonders bei Gemini trat dieses Problem sehr häufig und extrem auf.

Ein wichtiger Aspekt bei der Erstellung des Prompts war die Notwendigkeit einer genauen Beschreibung der Spalten und der verschiedenen Arten (Information, Auftrag, Feststellung, Beschluss). Ohne diese genaue Beschreibung tendierten die Tools dazu, lediglich Stichpunktlisten anstelle einer strukturierten Tabelle zu erstellen. Doch auch die genaue Definition der Arten führte nicht immer zu konsistenten Ergebnissen, da die Zuweisung der Kategorien unterschiedlich vorgenommen wurde.

Auf die Spalte "Termin" musste ebenfalls ein besonderes Augenmerk gelegt werden. Wenn im Prompt nicht explizit angegeben war, dass ein genaues Datum erforderlich ist, arbeiteten die Tools oft mit relativen Angaben wie "+2 Wochen". Selbst mit der Klarstellung, dass der Termin ein genaues Datum sein sollte, traten bei Le Chat Probleme auf: Es wurden immer Termine gewählt, die in der Vergangenheit lagen. Dazu wurde angegeben, dass die Termine beispielhaft gewählt und auf das aktuelle Datum, den 29.03.2023, bezogen seien.

Google Gemini hatte, wie bereits erwähnt, erhebliche Schwierigkeiten, nur die wichtigsten Punkte aus dem Gespräch herauszufiltern. Häufig wurden Besprechungsprotokolle mit etwa 30 Punkten erstellt, wobei jede einzelne Information separat und auch unwichtige Informationen aufgeführt wurden. Gemini fügte zudem eine neue Art, "Frage", selbstständig hinzu, wodurch im Besprechungsprotokoll teilweise mehrere Punkte für eine Information erstellt wurden. Bei erneuten Eingaben variierte die Länge des Besprechungsprotokolls stark. Trotz des Hinweises, nur die wichtigen Punkte in das Protokoll aufzunehmen, neigte Gemini weiterhin dazu, sehr detaillierte und kleinteilige Protokolle zu erstellen.

Le Chat hingegen hielt sich häufig zu knapp. Dadurch wurden immer wieder wichtige Anforderungen nicht in das Besprechungsprotokoll aufgenommen, und es musste besonders darauf geachtet werden, ob alle relevanten Informationen enthalten waren.

4.2 Projekthandbuch

Das Projekthandbuch wurde in zwei Schritten erstellt. Zunächst wurde die Einleitung mit dem Zweck des Dokuments, der Redaktion und dem Verteiler verfasst. Hierzu wurde der folgende Prompt im selben Chat eingegeben, in dem auch das Besprechungsprotokoll erstellt wurde:

Erstell mir für dieses Projekt die Einleitung für das Projekthandbuch. Die Einleitung besteht aus einem Abschnitt für den Zweck des Dokuments, einen Abschnitt zur Redaktion, in welchem geklärt wird, wer für das Dokument verantwortlich ist und einen Abschnitt zu dem Verteiler, also wer bei Änderungen zu informieren ist. Verantwortlich für das Dokument ist der Projektleiter und über Änderungen wird das gesamte Team informiert. Dazu wird eine entsprechende Nachricht in den Discord Channel geschrieben.

Prompt 4.2: Prompt Einleitung Projekthandbuch

Für die Einleitung sind spezifische Informationen erforderlich, die im Prompt angegeben werden müs-

sen. In diesem Projekt sind das die Verantwortlichkeit des Projektleiters und der Verteiler über den Discord-Channel. Die Tools konnten die Abschnitte gut erstellen, wobei jedoch auffiel, dass ChatGPT im Abschnitt Verteiler alle Teammitglieder nannte, was nicht unbedingt nötig ist. Ansonsten wurden die Abschnitte gut erstellt [Verweis!!!!!!!!!!!!!!!!!!!!!!].

Der zweite Teil betrifft das Kapitel „Projektdefinition“ mit den Abschnitten „Vorgeschichte“ und „Inhaltliche Kurzdarstellung“. Dazu wurde der folgende Prompt verwendet, welcher nach der Ausgabe der Einleitung im Chat eingegeben wurde:

Erstelle mir nun das Kapitel "Projektdefinition" des Projekthandbuches. Der erste Abschnitt soll die Vorgeschichte des Projekts beschreiben und anschließend soll ein Abschnitt eine inhaltliche Kurzdarstellung beschreiben.

Prompt 4.3: Prompt Projektdefinition Projekthandbuch

Diese zweite Eingabe erforderte keine weiteren Informationen, da diese im Gespräch bereits vorgegeben waren und die Tools darauf zugreifen können sollten. Auch hier wurden die Abschnitte gut erstellt, und die Tools konnten die benötigten Informationen aus dem Gespräch gut extrahieren [Verweis!!!!!!!!!!!!!!!!!!!!!!].

Bei der Erstellung der Einleitung zeigte sich, dass eine genaue Beschreibung der benötigten Abschnitte entscheidend war. Ohne diese klare Vorgabe neigten die Tools dazu, eigene Strukturen und Inhalte zu erstellen, die nicht den Anforderungen entsprachen.

Außerdem traten bei Gemini teilweise Fehler in den Namen auf. Einmal wurde beispielsweise geschrieben, dass Frau Schmidt (Teamchefin) mit Herrn Müller (Kunde) Kontakt aufnahm, was eine fehlerhafte Zuordnung darstellt.

4.3 Risikoliste

Auch die Risikoliste wurde in zwei Schritten erstellt. Zunächst wurde nur die Risikotabelle erstellt und im zweiten Schritt die Tabelle mit den Maßnahmen. Die beiden Ergebnisse sind [Verweis!!!!!!!!!!!!!!!!!!!!!!].

Der Prompt für die Risikotabelle

Erstell mir für diese Projekt eine Risikoliste. Diese soll aus mehreren Spalten bestehen: "ID" für eine eindeutige Identifikationsnummer, "Beschreibung" für eine ausführliche Beschreibung des Risikos und der Auswirkungen, "Datum" für den Zeitpunkt, wann das Risiko identifiziert wurde, "Autor" für die Person die das Risiko gemeldet hat, "Wahrscheinlichkeit (in%)" für einen Schätzwert der Eintrittswahrscheinlichkeit des Risikos, "Schaden (in €)" für eine Schätzung wie groß der Schaden ist, "Maß (in €)" was das Produkt aus Wahrscheinlichkeit und Schaden ist, "Risikoklasse" für eine Priorisierung der potentiellen Risiken wo zwischen Tolerierbar, Unerwünscht, Kritisch und Katastrophal Unterschieden wird und "Status" wo zwischen aktiv, eingetreten und geschlossen unterschieden wird. Das Risiko ist Tolerierbar wenn das Risikomaß geringer als 0,1% des Projektvolumen ist, Unerwünscht wenn es größer als 0,1% ist, Kritisch wenn es größer als 1% ist und Katastrophal wenn es größer als 10% ist. In der Risikoliste sollen Team-, Technische-, Methodische-, Kunden-, Fachliche-, Produkt-, Management- und Planungsrisiken betrachtet werden. Diese sollen mit einer leer Zeile getrennt werden, in welchen die Oberbegriffe stehen. Das Projektvolumen beträgt 500000€.

Prompt 4.4: Prompt Risikotabelle

ist sehr umfangreich formuliert. Damit die Tabelle mit den richtigen Spalten erstellt wird, wurde im Prompt jede einzelne Spalte aufgezählt und beschrieben. Ebenfalls ist es wichtig, die möglichen Werte für die Risikoklasse zu definieren, da hier sonst immer "Hoch", "Mittel" und "Niedrig" von den Tools verwendet wird. Es musste auch festgelegt werden, wann die einzelnen Risikoklassen auftreten, da die Zuweisung ansonsten recht schwammig ausfällt und schwerwiegende Risiken eine geringere Risikoklasse erhalten als eher unwichtige Risiken. Außerdem sollte beschrieben werden, welche Arten von Risiken betrachtet werden sollen und dass diese Arten in einer leer Zeile stehen, welche die dazugehörigen Risiken von den anderen Arten trennen. Ansonsten kann es passieren, dass die Risiken durcheinander geschrieben werden und damit nicht den Arten zuzuordnen sind. Damit der Risikoschaden zwischen den Tools vergleichbar ist, sollte das Projektvolumen definiert werden. Ansonsten wird auch dies von Tools festgelegt und führt zu Ungenauigkeiten.

Grundsätzlich war die Erstellung der Risiken kein Problem. Schwierig war es eher die gewünschte Formatierung zu erhalten, sowie eine richtige Zuweisung der restlichen Attribute. Besonders Gemini und Le Chat haben dabei Schwierigkeiten. Außerdem war bei allen drei Tools auffällig, dass der Autor immer die Person ist, zu dem das Risiko in den Tätigkeitsbereich fällt. Also die Person, die auch der Verantwortliche ist. Fraglich ist dabei teilweise, wenn Kundenrisiken vom Kunden, also Herr Müller, erstellt werden, da dieser an der Erstellung der Risikoliste überhaupt nicht beteiligt ist.

ChatGPT hat die Tabelle sehr gut erstellt. Lediglich die Zuweisung der Risikoklasse war falsch. Nach einem Hinweis diesbezüglich wurden diese jedoch richtig korrigiert.

Ein Problem bei Gemini ist, dass das Risikomaß nicht richtig berechnet wird. Das Komma der Werte

ist um eine Stelle zu weit links. Es muss alles einmal $\cdot 10$ gerechnet werden, damit die Werte stimmen. Kritisch bei Gemini ist, dass die Tabelle nicht in einem Zug erstellt werden kann. Es wird während der Generierung einfach aufgehört. Auch wenn man Gemini fragt, ob die vollständige Tabelle generiert werden kann, wird mitten drin aufgehört. Man muss explizit nur nach den noch offenen Risikoarten fragen. Diese werden dann erstellt, jedoch soll man den Schaden und das Maß selbst eintragen und berechnen. Nach anschließender Frage, ob er den Risikoschaden und das Schadensmaß festlegen kann, sagt er, dass er detaillierte Informationen über das Projekt und die potenziellen Risiken benötigt um genaue Werte festzulegen.

Le Chat hatte ein ähnliches Problem wie bereits im Besprechungsprotokoll [Abschnitt 4.1], dass das Datum immer auf den 01.04.2023 gesetzt wird. Außerdem hatte Le Chat Probleme damit, die Tabelle in das gewünschte Format zu bringen, dass die Risikoarten in einer Zeile stehen und die dazugehörigen Risiken darunter aufgelistet werden. Auch hier wurden die Risikoklassen falsch bestimmt und auch nach einem Hinweis waren diese falsch. Zudem wurde dabei der Risikoschaden der einzelnen Risiken geändert.

Die Maßnahmentabelle wurde mit folgendem Prompt erstellt:

Erstelle mir nun eine dazu passende Maßnahmentabelle. Auch diese besteht aus mehreren Spalten: "Typ" beschreibt ob die Maßnahme das Risiko verhindert, lindert oder überträgt. In "Beschreibung" wird die Maßnahme beschrieben. Falls ein Risiko als nicht mehr relevant eingestuft wird, wird in der Spalte "Beschreibung" eine Begründung eingetragen und die Maßnahme auf "beendet" gesetzt. "Trigger" ist das Ereignis, das den Start der Maßnahme veranlasst, falls diese nicht sofort eingeleitet werden soll. "Verantwortlicher" ist die zuständige Person für die Durchführung der Maßnahme. "Status" unterscheidet zwischen geplant, aktiv und beendet. Anschließend gibt es jeweils eine Spalte für "Restwahrscheinlichkeit (in %)", "Restschaden (in €)", "Restmaß (in €)" und "Restklasse" was die geschätzte Wahrscheinlichkeit, geschätzter Schaden, Maß und Klasse des Restrisikos, nach Durchführung der Maßnahme entsprechen. Für jedes Risiko sollen zwei Maßnahmen erstellt werden. Dazu wird über die zwei Maßnahmen eine Zeile mit der ID von dem Risiko beschrieben, auf die sich die Maßnahmen beziehen.

Prompt 4.5: Prompt Risikotabelle

Der Prompt für die Maßnahmen ist ähnlich wie der, für die Risikotabelle. Es wird jede Spalte der Tabelle einmal beschrieben, damit das Format der Tabelle mit den Vorgaben übereinstimmt. Anschließend wird gesagt, dass die ID des dazugehörigen Risikos mit in die Tabelle übernommen werden soll, damit man die Maßnahmen den Risiken zuordnen kann. Die Probleme sind hier ähnlich zu denen bei der Risikotabelle, jedoch erstellen alle drei Tools grundsätzlich anständige Maßnahmen für die Risiken. Bei allen dreien fühlt sich allerdings die Zuweisung der Attribute zu den Maßnahmen repetitiv an. Häufig wechseln sich die auswählbaren Parameter ab und auch die Wahrscheinlichkeiten und der Restschaden sind meistens immer wieder die gleichen Zahlen.

ChatGPT erstellt auch die Maßnahmentabelle sehr gut. Lediglich die Zuweisung der Restklasse stimmt

nicht überein.

Bei Gemini wird für die Begründung, warum ein Risiko als nicht mehr relevant eingestuft wird, eine eigene Spalte erstellt. Auch wenn man die Beschreibung dafür im Prompt ändert, wird die Spalte erstellt. Außerdem ist die Formatierung bei Gemini manchmal nicht so gut, da bei jeder 2. Zeile alle Daten ab der Spalte "Begründung" um eine Stelle nach links verschoben wird. Lässt man sich die Antwort neu generieren und auch wenn man Gemini sagt er soll die Formatierung korrigieren bleibt das Problem bestehen. Ein weiteres Problem ist, dass die Spalte "Trigger" nicht gefüllt wird und auch wieder das Problem, dass die Tabelle nicht ganz vollständig generiert wird sondern mitten drin aufhört, tritt wieder auf. Teilweise ist das Restmaß falsch berechnet und auch die Zuweisung der Restklasse ist nicht immer richtig.

Le Chat [%%]

- Prompt für Maßnahmen ähnlich, dass jede Spalte der Tabelle einmal beschrieben wird und anschließend gesagt wird, dass die ID des dazugehörigen Risikos übernommen werden soll.
- ChatGPT nur ein Fehler bei Restklasse. Sonst passt alles.
- Gemini erstellt für die Begründung, warum ein Risiko als nicht mehr relevant eingestuft wird eine eigene Spalte. Ist aber nicht so schlimm, da zu Beginn eh leer. Formatierung schlecht, da bei jeder 2. Zeile alle Daten ab Spalte "Begründung" um eine Stelle nach links verschoben wird. Neu generierung und auch Hinweis darauf ändert nichts. Trigger auch nicht gefüllt. Außerdem ist der Typ immer abwechselnd "Verhindern" und "Lindern". Auch hier ist teilweise das Restmaß falsch berechnet. Hört bei Risiko 4.2 auf mit der generierung der Maßnahmen. Maßnahmen passen aber zu Risiken.
- Le Chat erstellt die Maßnahmen nicht immer zu den Risiken in der Risikotabelle, sondern zu den Oberbegriffen Team-, Technische-, Methodische-, Kunden-, Fachliche-, Produkt-, Management- und Planungsrisiken. Maßnahme 1-4 scheinen sehr allgemein formuliert, Maßnahmen 5-8 würden zu den 1. Risiken des jeweiligen Oberbegriffes passen. Auch hier scheinen die Attribute sehr repetitiv. Wahrscheinlichkeit immer bei 5%, die Risikoklasse außer bei der ersten Maßnahme immer Unerwünscht und der Restschaden auch meistens bei den 2 Maßnahmen für ein Risiko gleich.

4.4 Anforderungsspezifikation

Einleitung: - Beschreibung, welche Absätze erstellt werden sollen. Da "Zweck und Umfang des Dokuments" recht spezifisch, genauer definieren was für ein Inhalt benötigt wird.

- ChatGPT stellt nur textuellen Verweis auf Spielregeln. Müssen noch selber eingefügt werden, falls gewünscht. Fraglich ob Begriff "Punktesystem" erklärt werden muss. Sonst gut.
- Gemini: ??????????????????????????????????????
- Le Chat verweist auf Link aus dem Internet. Macht man das in einem offiziellen Dokument? Sonst gut.
- Systemarchitektur: - Prompt: Inhalt der Systemarchitektur spezifizieren. Mit Nachrichten vorher im Chat wird diese gut erstellt. (Gemini ausstehend.....)
- ChatGPT stellt die Client-Server und Server-Datenbank Verbindung zunächst unidirektional da, nach Hinweis korrigiert er dies aber richtig. Hat in die Komponenten schon genauere Teile definiert, schon zu

detailliert? Beschreibung gut und hat in Chat Skizze "gezeichnet".

- Gemini: []

- Le Chat: Kann die Abbildung nicht zeichnen. Gibt Text nach erstem Prompt aber gut aus. Nach Eingabe "Beschreib mir, wie die Abbildung aussehen soll.Ërhält man beschreibung, welche Komponenten vorhanden sind, jedoch nicht die Beziehungen zueinander. Bei Frage "Wie ist die Beziehung zwischen der Client-Anwendungs, Server-Anwendung und der Datenbank?"Wird erklärt wie der Datenfluss abläuft (Bidirektional). Infos reichen um skizze selber zu erstellen.

Fachliches Datenmodell: - Prompt: Information, das ein UML-Klassendiagramm verwendet werden soll, um entsprechendes Layout zu erhalten. Außerdem beschreibung, welche Infos benötigt werden. Inhalt muss nicht beschrieben werden.

- ChatGPT: - Pfeile zeigen in gleiche Richtung -> sagt Btimmt sollten Bidirektional sein,...ünd generiert neu. Fehler bleibt aber - Spiel-Spieler Beziehung 4..*? Sollen doch 0..4 Teilnehmen oder garkeine. -> Ändert auf 1..4, richtige Bezeichnung? weiß ich auch nicht. - spielBrett als String? - Wofür zugNummer? -> Reihenfolge der Spielzüge innerhalb eines Spiels festzuhalten - Beziehungen von Spielfigur im Diagramm nicht richtig dargestellt. -> Bei Hinweis wird nicht korrigiert - Feld braucht Attribut ob belegt oder nicht -> Korrigiert - Chat nicht richtig im UML dargestellt, Beziehungen falsch -> Bei Hinweis wird nicht korrigiert - Spieler in User und Player aufteilen? -> Macht er richtig. Frage nur ob dann pos in Spieler oder figur? - status in Spiel zu einem Enum statt String? - farbe in Spieler zu Enum statt String? - Spielzug: figurID statt spielfigur? Beziehung zu Spieler? - Spielfigur: braucht farbe wenn Spieler schon farbe hat? -

- Hat keine Einschränkungen zu den Attributen festgelegt. -> Wird nach Hinweis gemacht - Legt nicht alle Beziehungen fest.

Prompts: - Erstell mir nun für die Anforderungsspezifikation das Fachliche Datenmodell. Dieses soll mit Hilfe eines UML Klassendiagrammes mit ergänzenden Beschreibungen bzw. Einschränkungen spezifiziert werden. Das Modell soll alle Entitätstypen mit deren Eigenschaften, Beziehungen und Einschränkungen besitzen. - Im UML-Diagramm zeigen immer beide Pfeile in eine richtung, müssen diese nicht bidirektional zeigen? - Du hast die Beziehung `Ein Spiel hat mehrere Spieler auf 4..*` festgelegt. Sollte diese nicht 1 - 4 betragen? Es sollen ja maximal 4 Spieler teilnehmen können. - Braucht das Feld nicht noch ein Attribut ob es belegt ist oder nicht? - Macht es nicht auch sinn Spieler in zwei Entitäten aufzuteilen? Einmal `User` und einmal `"Player"`. `User` beinhaltet dabei den Namen, die Email, das Passwort und die Punkte. `SSpieler` beinhaltet dann die nötigen Attribute zum Spielen wie `SpielerID`, die Farbe und die position seiner Spielfiguren. - Erstell mir zu den Attributen noch Einschränkungen, in dem du zu jedem Attribut eine Erklärung schreibst und ein Format definierst.

- Gemini: []

- Le Chat: - Nur beschreibung - Hat auch nur Spieler, über "player" würden sinn machen. - Spielfigur an einem oder mehreren Spielen teilnehmen? - Spieler-Spielfigur: Ein Spieler hat genau 4 Figuren, nicht weniger und nicht mehr - Beziehung zwischen Spiel-Spielfigur? wenn ja, Spieler * 4 Spielfiguren. - Punkte werden nicht gespeichert. - Player hat position? Player kann an mehreren Spielen teilnehmen? -> Nach Hinweis geändert - Player und Spielfigur haben positionen. - Spiel-Spielfigur Beziehung sinnvoll?
- Probleme mit der Beschreibung der Beziehungen zwischen den Entitäten - Ein paar grundsätzliche Einschränkungen aufgelistet, aber nicht zu jedem Attribut -> Sagt nur ob Ganzzahl oder Text, keine

Begrenzung in der Länge.

Prompts: - Erstell mir nun für die Anforderungsspezifikation das Fachliche Datenmodell. Dieses soll mit Hilfe eines UML Klassendiagrammes mit ergänzenden Beschreibungen bzw. Einschränkungen spezifiziert werden. Das Modell soll alle Entitätstypen mit deren Eigenschaften, Beziehungen und Einschränkungen besitzen. - Macht es nicht auch sinn SSpieler*in zwei Entitäten aufzuteilen? Einmal Üser*und einmal "Player". Üser* beinhaltet dabei den Namen, die Email, das Passwort und die Punkte. SSpieler* beinhaltet dann die nötigen Attribute zum Spielen wie SpielerID, die Farbe und die position seiner Spielfiguren. - Ein Player hat 4 Figuren. Außerdem hat der Player keine Positionen, diese haben ja die Figuren. Ein Player soll auch nicht an mehreren Spielen teilnehmen, sondern der User kann mehrere Player sein und ein Player spielt ein Spiel. Ein Spiel besteht aus 1 bis 4 Playern.

Anwendungsfalldiagramm:

- Prompt: Reicht zu sagen, dass das Diagramm alle Anwendungsfälle beinhalten soll und den Rollen zugewiesen werden soll.

- ChatGPT: - Spiel erstellen: wird zum Gastgeber. Besondere rechte? wenn ja, warum keine neue Rolle.
- > Nach Hinweis neue Rolle - Als User Chat nutzen? In Anforderungen wurde nur von Chat während des Spiels gesprochen. -> Nach Hinweis wurde entfernt. - Was gehört zu Zug machen, wenn Spielfigur bewegen eigener Anwendungsfall ist -> gesamte Ablauf wenn Spieler am Zug ist (Würfeln, Auswählen und Bewegen der Spielfigur, Schlagen gegnerischer Figuren und Beenden des Zuges), AF "SSpielfigur bewegen" war doppelt. Wurde aber nach Hinweis entfernt. Sehr großer Anwendungsfall, vielleicht lieber in kleinere AF aufteilen??
- Erstellt Code um in einem PlantUML-Editor oder einem Online Renderer das Diagramm als Bild zu erstellen
- Erstellt Skizze in PlantUML ganz gut, sieht lediglich aus als ob auch "Spieler" von "User" erbt.

- Gemini: []

- Le Chat: - Rollen werden gut erstellt. - Passende Anwendungsfälle, stimmen auch zu den Rollen - "Benutzer verwalten" von Admin vielleicht etwas ungenau? -> Wird geändert und entsprechend ergänzt - möglich mit PlantUML erstellen zu lassen, Diagramm jedoch recht durcheinander und Anwendungsfall "Passwort vergessen" wurde "User" zugewiesen und nicht wie in der Beschreibung "Guest"

AF:

- Prompt: Beschreibung, welcher Anwendungsfall beschrieben werden soll, welche Informationen zunächst erstellt werden soll und anschließend das ein Aktivitätsdiagramm erstellt werden soll. Beschrieben das die Aktivitäten dem Server oder dem Client zugewiesen werden sollen, um die spätere implementierung der Server-Client-Applikation einfacher fällt. Außerdem erwähnt das keine Dialog eigenschaft beschrieben werden sollen.

- ChatGPT-3.5: - Häufigkeit keine Zahl: "Häufig verwendet (mehrmals täglich) - Erstellt diagramm direkt mit, kein Endzustand. Verbindungen führen nach "Anmeldung erfolgreich" und "Anmeldung fehlgeschlagen" wieder zusammen. - Beschreibungen gut. - ChatGPT-4o macht zu viel. Implementierung wird bisschen mit beschrieben (Benutzersession erstellen) sowie Dialog eigenschaften wie Anzeigen von Fehlermeldungen werden beschrieben. GPT-3.5 hier tatsächlich besser. Außerdem wurde Code von PlantUML falsch erstellt (Syntax error)

- Gemini: []

- Le Chat: - Beschreibt für die Endzustände trotz Hinweis, das keine Dialog eigenschaften beschrieben

werden sollen, was der Dialog anzeigt. - Häufigkeit keine Zahl - Kann auf Nachfrage Code für PlantUML erstellen. - Diagramm hat nur einen Endzustand, wo Verbindung von "Anmeldung erfolgreich" und "Anmeldung fehlgeschlagen" enden.

Benutzungsschnittstelle:

- Prompt:

- ChatGPT: - Man startet im Hauptmenü und sieht alle Buttons. "Spiel erstellen" und "Spiel beitreten" können jedoch nur genutzt werden, wenn man Angemeldet ist. - Erstellt Skizze und Textuelle Beschreibung. Skizze passt aber nicht ganz zu Beschreibung. Auch nach Hinweis wird selber Fehler nochmal gemacht. Wenn man auf die wiederholt Falsche Antwort negativ Reagiert mit dem Feedback, dass die Hinweise nicht richtig umgesetzt wurden, und dann im als Prompt die Eingabe "Die Skizze ist unverändert. Setzt den SSpiel beitretenDialog mit dem Hauptmenü in Verbindung." gibt, wird die Skizze richtig erstellt. - Es wird auch beschrieben, welche Buttons und Eingabefelder die einzelnen Dialoge besitzen. Dies ist jedoch an dieser Stelle nicht nötig. - Skizze kann auch nicht zu einem Dialog zurück gehen. Wird dann nochmal aufgeführt (ohne Buttons,...)

- Gemini:

- Le Chat: - Erstellt für jeden Anwendungsfall einen Dialog, grundsätzlich gut. - Spielbrett benötigt auch ein Dialog? - Verbindungen zwischen den Dialogen macht keinen Sinn -> Nach Hinweis wird erneut ausgegeben, Beschreibung besser aber Skizze immer noch unsinnig. - Beschreibung beinhaltet aber nicht alle Anwendungsfälle (Admin wird ausgelassen). Außerdem werden Dialoge erstellt, für die es keine Anwendungsfälle gibt (z.B. "Profil bearbeiten um Benutzerdaten selber zu ändern) - Auch nach genauer Beschreibung wird Skizze nicht richtig erstellt.

Dialog:

- Prompt: Beschreibung des Inhalts reicht.

- ChatGPT: - Erstellt Skizze - beschreibt Validierung. Richtigen Stelle dafür?

- Gemini:

- Le Chat: - kleiner Formulierungsfehler bei "Abbrechen"-Button. "Nutzer bleibt angemeldet. -> Wird bei Hinweis korrigiert - Erklärt Skizze gut bei Nachfrage.

Nichtfunktionale Anforderungen:

- Prompt:

- ChatGPT: - passt

- Gemini:

- Le Chat: - passt

5 Herausforderungen und Limitationen

TODO!!!!!!!

5.1 Technische Herausforderungen

TODO!!!!!!!!!!!!

5.2 Limitationen von LLM Tools im Software Engineering Kontext

6 Zusammenfassung und Ausblick

TODO!!!!!!!!!!

6.1 Zusammenfassung der Ergebnisse

TODO!!!!!!!!!!!!

6.2 Implikationen für die Praxis

Alles Produkte von Unternehmen, welche Gewinnorientiert sind. Daher Diebstahl von Infos und Herangehensweisen der Projekte wenn man diese mit dem Tool zusammenfassen lässt. Grade im militärischen Bereich.

Lokale LLM Tools

Immer unterschiedliche Ausgaben trotz gleicher Eingabe. Erfahrung hat gezeigt, das falls mal nur mist rauskommt, einfach selbe eingabe immer wieder machen, bis was anständiges rauskommt.

Gemini merkt sich infos aus dem gleichen Chat nicht. Wenn man am nächsten Tag weiter arbeitet, kann Gemini nicht mehr auf die Infos vorher zugreifen.

Le Chat sagt, dass es ein Bild zu den Diagrammen erstellen kann. Jedoch sieht man zunächst den Link nicht und wenn man ihn danach fragt liefert er ein Link zu imgur.com, wo error 404 angezeigt wird.

TODO!!!!!!!!!!!!

6.3 Ausblick und zukünftige Entwicklungen

TODO!!!!!!!!!!!!

Anhang

Abbildungsverzeichnis

2.1	Dokumente im Software Engineering	10
3.1	Mögliche LLM Tool Unterstützung	18

Tabellenverzeichnis

Quellcodeverzeichnis

Stichwortverzeichnis

Anforderungsspezifikation, 13, 19, 28
Anwendung von LLM Tools im Software Engineering, 17
Architekturdokument, 15, 21
Ausblick und zukünftige Entwicklungen, 35
Besprechungsprotokoll, 11, 17, 23
ChatGPT, 4
Einleitung, 1
Google Gemini, 6
Grundlagen, 3
Herausforderungen und Limitationen, 33
Implikationen für die Praxis, 35
Le Chat, 8
Limitationen von LLM Tools im Software Engineering Kontext, 33
LLM Tools, 3
Motivation, 1
Praxisergebnisse und Vergleich, 23
Projekthandbuch, 11, 17, 24
Risikoliste, 12, 19, 25
Struktur der Arbeit, 1
Technische Herausforderungen, 33
Testspezifikation, 16, 21
Zielsetzung der Arbeit, 1
Zusammenfassung der Ergebnisse, 35
Zusammenfassung und Ausblick, 35

Literaturverzeichnis

- [1] Milfad Sadar. *Large Language Models - Grundlagen KI-getriebener Kommunikation*. Zugriff am 27. Mai 2024. 2023. URL: <https://weissenberg-group.de/was-ist-ein-large-language-model/#:~:text=Zum%20IPA%20Whitepaper-,Wie%20funktioniert%20ein%20LLM%3F,zwischen%20den%20Token%20zu%20ermitteln> (siehe Seite 4).
- [2] Jim Greene. *ChatGPT(software)*. Zugriff am 28. Mai 2024. 2024. URL: <https://eds.p.ebscohost.com/eds/detail/detail?vid=3&sid=1983b266-2f53-4090-a0f2-1aef00f51d1f%40redis&bdata=Jmxhbm9ZGUmc2l0ZT1lZHMtYGl2ZQ%3d%3d#AN=163030996&db=ers> (siehe Seite 6).
- [3] Dipl.-Ing. (FH) Stefan Luber. *Was ist Google Gemini?* Zugriff am 29. Mai 2024. 2024. URL: <https://www.bigdata-insider.de/was-ist-google-gemini-a-f855ef105c02f4533c18e913d4643007/> (siehe Seite 6).
- [4] Matthias Kindt.
Mistral AI: Französisches KI-Startup stellt Chatbot "Le Chat" und Kooperation mit Microsoft Azure vor. Zugriff am 16. Mai 2024. 2024. URL: <https://www.unidigital.news/mistral-ai-franzoesisches-ki-startup-stellt-chatbot-le-chat-und-kooperation-mit-microsoft-azure-vor/> (siehe Seite 8).
- [5] Ian Sommerville. *Software Engineering*. 9. Auflage. Harlow, England: Addison-Wesley, 2010. ISBN: 978-0-13-703515-1 (siehe Seite 11).

