Fachhochschule Nordwestschweiz
Hochschule für Technik

Fachhochschule Nordwestschweiz

# IP5 - Fachbericht

Innovative Prompt Management for LLMs

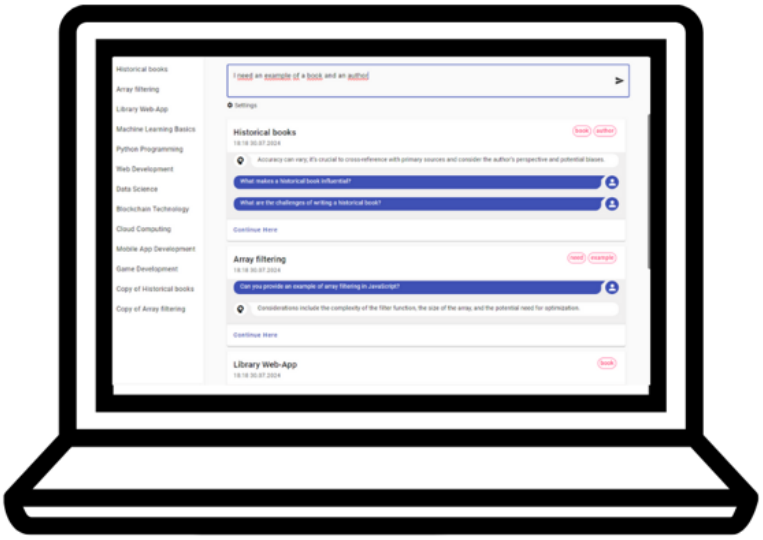



*Abbildung 1 - PromptRecall*

Studierende: Simon Lüscher, Jack Gläser

Betreuung: Prof. Martin Kropp, Dr. Nitish Patkar

Vereinbarung: vertraulich

Projektnummer: 24HS_IMVS10

Abgabe: 16.8.2025

# Inhaltsverzeichnis

# Abstract

In diesem Projekt haben wir eine neue, innovative Funktionalität im Bereich des Prompt-Managements entwickelt. Wir nennen diese Funktionalität: «Similar Prompts». Ihr Ziel ist es die Benutzerinteraktionen mit grossen Sprachmodellen (LLMs) zu verbessern. Diese Innovation zeigt den Benutzern ähnliche oder verwandte Gespräche respektive Prompts an, während der Benutzer einen Prompt verfasst, sodass sie nahtlos im richtigen Kontext einsteigen oder schnell bereits gestellte Prompts und deren Antworten wiederfinden und nutzen können. Dies macht ein erneutes Prompten des Modells überflüssig und ermöglicht es den Benutzern schneller auf relevante Informationen zuzugreifen.

**Ziel**

Das Hauptziel dieses Projekts war die Einführung einer innovativen Funktion im Bereich des "Prompt Managements" für KI-gesteuerte Chatbots. Durch die Analyse bestehender Probleme (Pain Points) wollten wir ein Werkzeug schaffen, das die Benutzererfahrung und die Handhabung von Prompts optimiert sowie die Effizienz und Effektivität bei der Nutzung von KI-gesteuerten Chatbots steigert.

**Problemstellung**

Aktuelle KI-gesteuerte Chatbots fehlen innovative Funktionen zur Verwaltung von Prompts. Benutzer müssen oft Fragen erneut stellen oder haben Schwierigkeiten, relevante vergangene Gespräche abzurufen, was zu einer ineffizienten Interaktionserfahrung führt.

**Methodik**

Unser Ansatz begann mit einer Marktanalyse bestehender KI-gesteuerten Chatbots, um verfügbare Funktionen im Bereich des Prompt Managements zu identifizieren. Anschliessend führten wir eine Umfrage durch, um Einblicke in die aktuellen Paint Points und gewünschten Funktionen der Benutzer zu gewinnen. Die Ergebnisse wurden priorisiert, und auf dieser Grundlage entwickelten wir mehrere Prototypen der "Similar Prompts"- / Suchfunktion.
Zur Validierung unserer Prototypen führten wir mehrere Runden von Benutzertests durch. Diese Tests halfen uns, das User Interface zu verbessern. Basierend auf dem Feedback nahmen wir Anpassungen vor, um Layout und Benutzerführung weiter zu verfeinern.

**Ergebnisse**

Das Endprodukt, ein Minimum Viable Product (MVP), wurde basierend auf validiertem Benutzerfeedback entwickelt und als funktionsfähige Funktion implementiert. Die "Similar Prompts"-Funktionalität zeigt erfolgreich eine verbesserte Effizienz und Benutzerzufriedenheit, indem relevante Prompt-Vorschläge bereitgestellt wurden und es den Benutzern ermöglicht wurde, Gespräche im richtigen Kontext fortzusetzen.

# Danksagung

Wir möchten an dieser Stelle unseren aufrichtigen Dank an Prof. Martin Kropp und Dr. Nitish Patkar aussprechen, die uns während unserer gesamten Arbeit begleitet und unterstützt haben. Ihre fachliche Expertise, wertvollen Ratschläge und stetige Ermutigung haben massgeblich zu unserem Erfolg beigetragen.

Professor Kropp hat uns durch seine tiefgründigen Einblicke und seinen Einsatz nicht nur in fachlicher Hinsicht weitergebracht, sondern auch unsere Herangehensweise und unser Verständnis für das Themengebiet entscheidend geprägt. Seine Fähigkeit, komplexe Sachverhalte verständlich zu vermitteln, hat uns stets inspiriert und motiviert, unser Bestes zu geben.

Dr. Patkar hat uns mit seinen detaillierten Rückmeldungen und seinem umfangreichen Wissen geholfen, unser Projekt auf ein höheres Niveau zu heben. Seine Unterstützung bei der Bewältigung technischer Herausforderungen und Brainstorming zu offenen Themen war von unschätzbarem Wert und hat uns über schwierige Phasen hinweggeholfen.

Wir sind dankbar für die Zeit und Mühe, die beide Coaches in unsere Betreuung investiert haben. Ohne ihre Unterstützung wäre unser Projekt nicht in dieser Form realisierbar gewesen. Wir freuen uns darauf, das Gelernte in zukünftigen Projekten anwenden zu können und hoffen, dass wir auch weiterhin auf ihre Expertise und ihren Rat zählen dürfen.

Auch den zahlreichen Teilnehmenden an unserer Umfrage und den durchgeführten User Tests sei ein Dankeschön auszusprechen.

# 1 Einleitung

In der vorliegenden Arbeit wurde die innovative Funktion nachfolgend „Similar Prompts“ genannt, entwickelt, die den Umgang mit KI-basierten Chatbots durch fortschrittliches Prompt-Management signifikant verbessert. Ziel war es, eine neuartige Funktion zu schaffen, die den Benutzern einen Mehrwert bietet, indem sie den Kontext vergangener Interaktionen besser nutzt.

## 1.1 Ausgangslage

In der Welt der grossen Sprachmodelle (LLMs) wie ChatGPT [1] von OpenAI oder Llama [2] von Meta spielt die Benutzerfreundlichkeit in Bezug auf die Handhabung aller Prompts und Chats eine entscheidende Rolle. Derzeit bieten diese Modelle jedoch lediglich eine grundlegende Historie vergangener Konversationen, ohne erweiterte Such- oder Kontextfunktionen.

Es fehlt an einer effizienten Möglichkeit, vergangene Konversationen oder individuelle Prompts schnell wiederzufinden und in den relevanten Kontext einzutauchen, ohne das LLM erneut mit Informationen versorgen zu müssen.

Diese Einschränkungen führen oft zu wiederholten Fragen und einem ineffizienten Informationsfluss, da Benutzer manuell in früheren Konversationen suchen müssen, um relevante Antworten zu erhalten.

## 1.2 Problemstellung

Um eine innovative Funktion im Bereich des Prompt-Managements zu entwickeln, müssen wir spezifische Herausforderungen bewältigen:

1. **Klassifikation von Prompts:** Entwicklung einer Methode zur Klassifikation nach Inhalt, Kontext und Relevanz.
2. **Effiziente Suchfunktion:** Implementierung eines robusten Suchalgorithmus basierend auf Ähnlichkeit und Kontext.
3. **Benutzerfreundliche Präsentation:** Darstellung der Ergebnisse in einer intuitiven Art und Weise.
4. **Nahtlose Benutzerführung:** Gestaltung eines Workflows, der eine effiziente Nutzung der Funktionalität ermöglicht, damit unsere Funktionalität auch einen wirklichen Mehrwert bieten kann.

## 1.3 Zielsetzung

Das Ziel dieser Arbeit ist es, ein System zu entwickeln, das den Benutzern ermöglicht, vergangene und ähnliche Prompts effizient und benutzerfreundlich zu finden. Dies soll verhindern, dass bereits gestellte Fragen erneut gestellt werden müssen und es den Benutzern ermöglichen, ihre Fragen im richtigen Kontext zu platzieren. Dadurch wird die Notwendigkeit reduziert, das LLM erneut mit Informationen zu versorgen, und die Zeit, die für die Beantwortung relevanter Fragen benötigt wird, erheblich verkürzt.

Durch die Implementierung dieser Funktionen streben wir folgendes an:

- **Effizienzsteigerung:** Reduzierung der Suchzeit für relevante Informationen.
- **Usability:** Schaffung einer intuitiven und benutzerfreundlichen Oberfläche.
- **Kontextbewusstheit:** Verbesserung der Antwortgenauigkeit durch Nutzung relevanter vergangener Konversationen.
- **Redundanzvermeidung:** Minimierung der Anzahl wiederholter Fragen, was zu einer effizienteren Nutzung der LLM-Ressourcen führt.

Mit diesen Zielen vor Augen wollen wir die Interaktion mit grossen Sprachmodellen signifikant verbessern und den Benutzern ein leistungsfähiges Werkzeug zur Verfügung stellen, welches die Nutzung von vergangenen Konversationen optimiert und die Wiederauffindbarkeit von Daten erleichtert.

Daraus resultieren folgende Forschungsfragen für diese Arbeit:

1. Wie lässt sich ein spezifischer Prompt der Vergangenheit möglichst einfach und effizient wiederfinden?

2. Wie können wir es dem Benutzer erleichtern im richtigen Kontext seine Frage dem LLM zu stellen?

3. Wie lassen sich LLM-Prompts bewerten, klassifizieren und besser visualisieren?

## 1.4 Berichtaufbau

Zu Beginn des vorliegenden Berichts findet eine Problemanalyse statt, wobei die momentan vorhandenen Pain Points mittels einer Umfrage untersucht werden und mit einer Marktanalyse der momentan bestehenden Chatbots verglichen werden („Momentaner Stand der Forschung“).

Es folgt eine genaue Erläuterung des eigentlichen Use-Cases unserer zu entwickelnden Funktionalität sowie das Konzept und Design (Prototypen).

Im Kapitel Implementierung wird auf die technische Umsetzung einzelner wichtiger Bausteine punktuell eingegangen.

Zum Schluss des Berichts werden die Ergebnisse im Kontext des gesamten Projekts diskutiert, indem die zu Beginn gestellten Forschungsfragen analysiert werden und ein Fazit gezogen wird.

Im Anhang findet man alle unsere wichtigen Artefakte und zusätzliche Informationen zu unserer Arbeit.

## 1.5 Methodik

Um unsere Ziele zu erreichen, haben wir einen systematischen und iterativen Ansatz gewählt, der auf Recherche, Validierung und schrittweiser Implementierung basiert. Unsere Methodik umfasst folgende Schritte:

1. **Recherche:**
    - Zu Beginn des Projekts haben wir umfassende Recherchen durchgeführt, um bestehende Methoden und Technologien zur Klassifizierung, Speicherung und Suche von Prompts zu verstehen. Dabei haben wir auch Best Practices und Fallstudien analysiert, um ein solides Fundament für unsere Arbeit zu schaffen.
2. **Datenvalidierung:**
    - Nach der Recherchephase haben wir relevante Daten gesammelt und validiert. Dies beinhaltete die Analyse von Prompts aus verschiedenen Quellen, um ihre Struktur, Häufigkeit und Kontext zu verstehen. Wir haben verschiedene Ansätze zur Datenaufbereitung und -bereinigung getestet, um eine qualitativ hochwertige Datenbasis zu gewährleisten.
3. **Prototyping:**
    - Basierend auf unseren Erkenntnissen haben wir erste Prototypen entwickelt. Diese umfassten Entwürfe für die Klassifizierung und Suche von Prompts sowie erste Ansätze für die Benutzeroberfläche. Die Prototypen wurden intern getestet und anhand von Feedback überarbeitet.
4. **Validierung:**
    - Jeder Prototyp durchlief einen Validierungsprozess, bei dem wir Feedback von Nutzern und Experten einholten. Diese Rückmeldungen flossen in die Überarbeitung der Prototypen ein, um sicherzustellen, dass unsere Ansätze sowohl technisch machbar als auch benutzerfreundlich sind.
5. **Iterative Implementierung:**
    - Die Implementierung erfolgt in iterativen Schritten. In jeder Iteration wird ein Teil der Funktionalität implementiert und erneut validiert. Dieser Zyklus aus Implementierung und Validierung stellt sicher, dass wir kontinuierlich Verbesserungen einführen und Probleme frühzeitig identifizieren und beheben können.
6. **Kontinuierliche Validierung:**
    - Während des gesamten Projekts setzen wir auf kontinuierliche Validierung. Nach jeder Implementierungsphase wird das System getestet und Feedback eingeholt, um sicherzustellen, dass wir auf dem richtigen Weg sind und die Anforderungen der Benutzer erfüllen.

Diese Methodik ermöglicht es uns, flexibel auf Herausforderungen zu reagieren und gleichzeitig eine hohe Qualität und Benutzerfreundlichkeit unseres Systems zu gewährleisten. Durch die iterative Vorgehensweise stellen wir sicher, dass unser Endprodukt sowohl technisch robust als auch optimal auf die Bedürfnisse der Benutzer abgestimmt ist.

# 2 Analyse

Dieses Kapitel behandelt die theoretische Analyse der Arbeit und umfasst eine detaillierte Problemanalyse, die Ergebnisse einer durchgeführten Umfrage, den aktuellen Stand der Forschung sowie die Beschreibung einer typischen Persona für den Use-Case der implementierten Lösung.

## 2.1 Problemanalyse

In diesem Kapitel wird das zentrale Problem, welches die Entwicklung der neuen Funktionalität aufzeigt, detailliert analysiert.
Zunächst wird der aktuelle Stand der Technik und die Bedürfnisse der Benutzer erhoben, um die Lücken in bestehenden Lösungen zu identifizieren und die Anforderungen an die neue Funktion zu definieren.

Um eine innovative, neue Funktionalität im Bereich des Prompt-Managements zu entwickeln, stehen wir vor folgenden Herausforderungen:

**Klassifikation von Prompts: Wie klassifizieren wir Prompts?**
Um eine effiziente Suche und Wiederverwendung von Prompts zu ermöglichen, müssen wir eine Methode entwickeln, die Prompts nach Inhalt, Kontext und Relevanz klassifiziert. Dies könnte durch die Analyse von Schlüsselwörtern, Themen oder semantischen Ähnlichkeiten geschehen. Eine präzise Klassifikation ist entscheidend, um relevante Ergebnisse schnell und akkurat zu liefern.

**Suchalgorithmus: Wie suchen wir nach Prompts?**
Um den Benutzern relevante Prompts basierend auf Ähnlichkeit und Kontext zur Verfügung zu stellen, muss ein robuster Suchalgorithmus entwickelt werden. Dieser könnte durch Vektorraumsuche oder andere fortschrittliche Suchtechniken realisiert werden.
Die Herausforderung besteht darin, den Algorithmus sowohl leistungsfähig als auch benutzerfreundlich zu gestalten, sodass es asynchron im Hintergrund geschehen kann und der Benutzer keine langen Wartezeiten aushalten muss.

**Präsentation der Suchergebnisse: Wie zeigen wir die Resultate dem User?**
Die Art und Weise, wie die Suchergebnisse präsentiert werden, ist entscheidend für die Benutzerfreundlichkeit der Funktion. Die Ergebnisse müssen klar und intuitiv dargestellt werden, beispielsweise durch eine übersichtliche Auflistung ähnlicher Prompts oder durch Empfehlungen, die direkt in die laufende Konversation integriert werden.

**Gestaltung des User Interfaces / User Experience: Wie sieht die User Journey aus?**
Ein nahtloser und intuitiver Workflow ist entscheidend für die Akzeptanz und Nutzung der neuen Funktion. Die User Journey umfasst die Schritte von der Eingabe eines neuen Prompts bis zur Anzeige ähnlicher vergangener Prompts.

### 2.1.1 Umfrage

Um die Bedürfnisse und Präferenzen sowie die Pain Points der Nutzer zur momentanen Lage besser zu verstehen, haben wir eine Umfrage durchgeführt. An der Umfrage nahmen insgesamt circa 65 Personen teil, darunter Studierende, Forschende und viele Fachkräfte welche hauptsächlich aus dem IT-Bereich stammen.

Die Umfrage haben wir in der Fachhochschule Nordwestschweiz sowie mit Arbeitskollegen und Freunden geteilt.
Des Weiteren wurde unsere Umfrage auf LinkedIn gepostet und auch von Dr. Nitish Patkar in seinem Netzwerk geteilt.

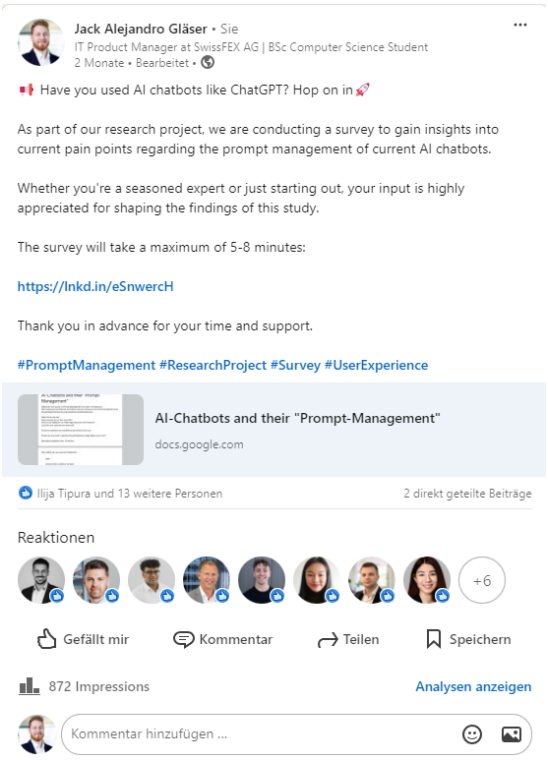


*Abbildung 2 - Beitrag mit Umfrage auf LinkedIn*

**Wichtige Erwähnung**

Neben unserer Umfrage, welche sich im Anhang befindet, hat Dr. Nitish Patkar ebenfalls parallel eine sehr ähnliche Umfrage [3] für seine Forschung durchgeführt. Wir haben die bis dahin gesammelten Informationen in unsere Ergebnisse der Umfrage mit einfliessen lassen, sobald unsere Umfrage beendet war, sodass sich ***die Resultate hier auf beide Umfragen beziehen***.

**Ziel der Umfrage**
Das Ziel der Umfrage bestand darin, unsere Ersten Ideen zu neuen Prompt-Management Funktionalitäten anhand der potenziellen Benutzer dieser Funktionen zu validieren, um daraus Priorisierungen machen zu können, unsere Zielgruppe besser kennenzulernen, potenzielle Benutzer für User Tests einladen zu können, sowie unsere Ersten Prototypen bauen zu können.

Die Fragen zielten darauf ab, die Nutzungsgewohnheiten und Anforderungen der Teilnehmer an einer Suchfunktion sowie Tagging/Labelling und weitere Funktionalitäten wie eine andere Ansicht der Prompts in einem "Prompt Management"-System zu analysieren.

**Zielgruppe**
Es haben insgesamt **63 Personen** an der Umfrage teilgenommen, davon waren **90% in der IT tätig**, die restlichen 10% sind im Bereich Finanzen, Bildung, Gesundheit oder Andere tätig.
Gut 50% der Teilnehmenden beschreiben sich als Fortgeschrittene oder Experten mit der Handhabung von herkömmlichen AI-Chatbots.
Die Restlichen 50% sind normale Benutzer und ein paar wenige würden sich als Anfänger einstufen.

**Durchführungsweise**
Online via Google Forms, hauptsächlich Fragen mit einer Skala von 1-5 und ein paar offene Fragen, um individuelle Pain Points oder Ideen sammeln zu können.

**Zusammenfassung der Ergebnisse aus der Umfrage**
Es folgt eine Zusammenfassung der wichtigsten Resultate aus der Umfrage geordnet nach Funktionalität.

**Suchfunktion**
Häufige Antworten:

- 85% der Befragten gaben an, dass sie eine erweiterte Suchfunktion benötigen, um ähnliche Prompts zu finden.
- 75% der Teilnehmer wünschen sich eine Funktion zur Anzeige von Suchergebnissen basierend auf semantischen Ähnlichkeiten.

Erkenntnisse:

- 60% der Befragten wussten nicht, dass sie die vorhandene Konversationshistorie nutzen können, um ihre Fragen zu präzisieren.
- Die Mehrheit der Nutzer (80%) hat Schwierigkeiten, relevante frühere Konversationen schnell zu finden.
- 70% der Befragten bevorzugen eine Suchfunktion, die mehr kontextbezogene Ergebnisse liefert.

**Tagging/Labelling von einzelnen Prompts und Konversationen**
Häufige Antworten:

- 78% der Teilnehmer halten es für wichtig, dass Prompts durch Tags oder Labels kategorisiert werden können, um die Auffindbarkeit zu verbessern.
- 65% der Befragten nutzen bisher keine Tags oder Labels, sehen aber den Bedarf.

Erkenntnisse:

- 55% der Teilnehmer wissen nicht, wie sie Tags oder Labels effizient nutzen können, obwohl sie den Nutzen erkennen.
- Es wurde deutlich, dass 70% der Nutzer spezifische Tags für verschiedene Themenbereiche wünschen, um die Suche zu erleichtern.
- 68% der Befragten gaben an, dass eine automatisierte Tagging-Funktionalität nützlich wäre.

**Möglichkeit das Layout von vergangenen Prompts zu ändern am Beispiel einer Ordnerstruktur**
Häufige Antworten:

- 45% der Teilnehmer geben an, dass sie sich eine Ordnerstruktur der Prompts anstelle einer einfachen Liste wünschen
- Die restlichen Teilnehmer denken es ist nicht nötig (26%) oder sind sich unsicher, ob es einen Mehrwert bringen würde (29%)

**Interpretation**
Die Umfrageergebnisse zeigen, dass es ein starkes Bedürfnis nach verbesserten Such- und Tagging-Funktionen gibt sowie eine Möglichkeit einfacher in vergangene Konversationen einzusteigen, um an eine Information zu gelangen oder im gleichen Kontext bleiben zu können. Viele Nutzer haben Schwierigkeiten, relevante Informationen schnell zu finden und wünschen sich eine effizientere Verwaltung und Kategorisierung von Prompts.
Eine Möglichkeit zur Personalisierung des Layouts oder einer Ordnerstruktur anstelle einer einfachen Liste für die Darstellung der vergangenen Konversationen ist drittrangig.

**Implikation**
Durch die Resultate der Umfrage priorisieren wir folgende potenzielle Funktionalitäten:

- Suchfunktion jeglicher Art
- Tagging / Labelling (Kategorisierung von Prompts)
- Einstellbares Layout

### 2.1.2 Marktanalyse / Konkurrenzanalyse bestehender LLM-Chatbots

Um einen umfassenden Überblick über die aktuelle Marktsituation zu erhalten, haben wir mehrere populäre KI-basierte Chatbots untersucht und deren vorhandene Prompt-Management-Funktionalitäten analysiert.

Die in der Umfrage am häufigsten gewünschten Funktionen wurden hauptsächlich in die Analyse einbezogen.
Zu den am häufigsten gewünschten Funktionen gehören: eine umfassende Suchfunktion, Tagging oder Labelling von Prompts, Prompt-Vorlagen sowie ein anpassbares Layout.

| | OpenAI ChatGPT 3.5 | Meta Llama 2 | Gemini | AnonChatGPT |
|---|---|---|---|---|
| Prompt Verlauf | x | x | x | |
| Datum Sortierung | x | x | x | |
| Suchfunktion jeglicher Art | | | | |
| Tagging / Labelling von Prompts | | | | |
| Prompt Vorlagen | | | x | |
| Einstellbares Layout | | | | |
| Einfaches Design | x | x | x | x |

*Abbildung 3 - Marktanalyse bestehender LLM AI-Chatbots in Bezug auf gewünschte Prompt-Management Features*

**Fazit und Erkenntnisse**

Basierend auf dieser Analyse lassen sich folgende Erkenntnisse ziehen: Es gibt sehr viele verschiedene LLM Chatbots. Viele sind granular für spezifische Aufgabengebiete abgestimmt. Beispiel: AnonChatGPT [4], welches sehr schlicht und den Fokus auf Anonymität zieht.
Alle diese Chatbots unterscheiden sich in ihren Einsatzzwecken oder weisen kleinere Unterschiede in den Funktionalitäten auf.

Was klar heraussticht: Alle haben so gut wie keine der aus unserer Umfrage resultierenden Funktionalitäten im Bereich von Prompt-Management Feature verfügbar.
Während Prompt-Vorlagen teilweise verfügbar sind, fehlen Suchfunktionen sowie die Möglichkeit zum automatischen oder manuellen Tagging von Prompts oder Konversationen in den meisten Fällen.

## 2.2 Use Case: Similar Prompts

In diesem Kapitel wird der Anwendungsfall beschrieben, aus dem unsere Idee der Similar Prompts entstanden ist und in welchen Szenarien diese Funktionalität von Nutzen ist. Ausserdem wird kurz die Persona vorgestellt, die unserer Meinung nach am meisten von dieser Funktionalität profitiert.

Aus der Problemanalyse haben sich insbesondere die folgenden Funktionalitäten herauskristallisiert, die für die Benutzer den grössten Nutzen bieten würden: eine umfassende Suchfunktion, Tagging/Labelling von Prompts und eine allgemeine Klassifizierung der Prompts.

**Beschreibung**

Die Idee der Similar Prompts besteht darin, den Nutzern während des Verfassens eines Prompts ähnliche, bereits vergangene Konversationen und Prompts anzuzeigen. Dies ermöglicht es ihnen, schneller und effizienter die gewünschte Information zu erhalten, ohne den gesamten Prompt erneut ausschreiben oder zusätzliche Informationen für den richtigen Kontext liefern zu müssen.
Ein weiterer positiver Effekt Nebeneffekt besteht darin, die $CO_2$-Emissionen zu reduzieren, da ein Datenbankaufruf deutlich ressourcenschonender ist als eine erneute Prompt-Generierung, da laut Studien zu dem Thema eine Anfrage zwischen 0.1 und 10 Gramm CO2 [5] verbraucht.

**Vorteile**

- Schnellerer Zugang zu den gewünschten Informationen
- Genauere Antworten bei nachfolgenden Prompts (die KI hat den Kontext bereits durch die bestehende Konversation)
- Wiederverwendung von Informationen, Strukturen und Vorlagen (z. B. Newsletter, interne wiederkehrende E-Mails, ähnliche Fragen, YouTube-Beschreibungen)
- Bereits gestellte Fragen können schnell wiedergefunden und die Lösung ohne erneute Generierung angezeigt werden
- Ökologischere Nutzung von LLMs

**Nachteile**

- User muss sich die Vorschläge anschauen und durchlesen welches für den User mühsam sein kann

### 2.2.1 Szenario «Softwareentwicklung» - korrekter Kontext

**Persona**

Jonas ist Softwareprogrammierer, welcher an mehreren Projekten gleichzeitig arbeitet. Beispielsweise hat er ein Projekt auf der Arbeit in Angular, eines in der Schule mit Java und zu Hause arbeitet er noch an ein weiteres in der Programmiersprache Rust.
Da der Entwickler ständig zwischen verschiedenen Projekten wechseln muss, ist es für ihn besonders wichtig, dass die KI den jeweiligen Kontext seiner Anfrage versteht, um ihm gezielt weiterhelfen zu können. (Details zur Persona befinden sich im Anhang)

**Code-Snippets & Kontext**

Entwickler können in früheren Konversationen nach spezifischen Code-Snippets, Lösungsansätzen oder dem richtigen Kontext suchen, was die Effizienz bei der Problemlösung erheblich steigert.

**Fehlerbehebung**

Wenn ein Entwickler auf ein Problem stösst, kann er schnell nach ähnlichen früheren Konversationen suchen, in denen das Problem bereits gelöst wurde oder in denen der richtige Kontext vorliegt.

**Beispiel**

Ein Entwickler sucht nach „Wie kann ich meine Bücher nach Datum filtern?“ und erhält eine Antwort, die sich auf physische Bücher bezieht. Dabei handelt es sich jedoch um eine Webanwendung, in der Bücher digital in einer Tabelle gefiltert werden sollen.
Da der Kontext nicht klar ist, könnte die KI möglicherweise folgend antworten: „Fang damit an alle Bücher aus deinem Bücherregal zu holen...“.

Unsere Idee ist es, anhand definierter Schlüsselwörter in früheren Konversationen nach ähnlichen Fragestellungen zu suchen und diese anzuzeigen.
Entweder findet der Benutzer eine ähnliche Frage und ist damit zufrieden, oder es gibt wenigstens eine Konversation, die den richtigen Kontext bietet. In diesem Fall erhält der Benutzer eine neue, sinnvolle Antwort von der KI. Falls keine passenden Vorschläge vorhanden sind, kann der Benutzer eine neue Konversation starten.

### 2.2.2 Szenario «Content Creation» - Wiederverwendung von Inhalten und Strukturen

**Persona**

Lisa arbeitet an mehreren Content-Projekten gleichzeitig, darunter Blog-Posts über aktuelle Trends in der IT, Whitepapers für Produkteinführungen und Social-Media-Kampagnen.
Sie nutzt regelmässig ihre bevorzugte KI, um Inspiration und Ideen für neue Inhalte zu finden. Allerdings fehlt der KI oft der Kontext zu ihren früheren Arbeiten, was bedeutet, dass Lisa immer wieder nach ähnlichen Themen suchen muss, um Strukturen und Formulierungen manuell wiederzuverwenden.
Diese mühsame und zeitraubende Arbeit stört ihren kreativen Fluss und führt dazu, dass sie weniger Zeit für die Entwicklung neuer Ideen hat. (Details zur Persona befinden sich im Anhang)

**Wiederverwendung von Inhalten**

Autoren oder Marketing-Teams können auf bereits erstellte Inhalte zugreifen, um ähnliche Themen oder Formulierungen zu finden und wiederzuverwenden, was die Content-Erstellung beschleunigt. Beispielsweise die Wiederverwendung von Einleitungssätzen, Schlusspassagen oder spezifischen Formatierungen und Strukturen.

**Inspiration und Ideenfindung**

Ähnliche Prompts können als Inspirationsquelle dienen, um neue Inhalte zu erstellen oder bestehende Ideen weiterzuentwickeln.

**Beispiel**

Wiederverwendung von Blog-Post-Strukturen, ein Content Creator schreibt regelmässig Blog-Posts über technische Themen und möchte eine ähnliche Struktur wie bei einem früheren Beitrag verwenden.

**Vorteil**

Der Autor kann nicht nur die Struktur übernehmen, sondern auch spezifische Formulierungen und Ideen wiederverwenden, die besonders gut funktioniert haben.

**Beispielhafte Konversation (mit Funktionalität der Similar Prompts)**

- **Autor:** „Ich brauche die Struktur für den Blog-Post über die neuesten JavaScript-Frameworks.“
- **System:** Zeigt ähnliche frühere Blog-Posts an, darunter einige mit der gesuchten Struktur für den neuen Blog-Post.
- **Autor:** „Ich werde die Struktur des Posts über die 'Top 10 Python Libraries' wiederverwenden.“

# 3 Konzept & Design

Im folgenden Kapitel wird das Konzept sowie das Design der entwickelten Lösung detailliert beschrieben. Es wird erläutert, wie der Prototyp entstanden ist und wie er durch eine Reihe von User-Tests weiterentwickelt wurde, um eine benutzerfreundliche und effektive Lösung zu gewährleisten. Der Abschnitt geht insbesondere auf die wichtigsten Erkenntnisse aus den User-Tests ein, die den Designprozess massgeblich beeinflusst haben.

## 3.1 User Testing

Um sicherzustellen, dass unsere Lösung den Bedürfnissen der Benutzer gerecht wird, haben wir mehrere Runden von User-Tests durchgeführt. Diese Tests hatten das Ziel, wertvolles Feedback zu sammeln, Designentscheidungen zu validieren und den Prototypen kontinuierlich zu verbessern. Die gesammelten Erkenntnisse aus den Tests flossen direkt in die Weiterentwicklung des Prototyps ein, wodurch wir einen MVP-Prototypen entwickeln konnten, der auf echten Nutzeranforderungen basiert.

**Ziel**

Das Ziel der User-Tests war es, einen Prototyp zu entwickeln, der durch echte Nutzer validiert wurde und für die Implementierung eines MVP bereit ist.

**User Testing 1**

In der ersten Testphase erhielten wir wertvolles Feedback zu den grundlegenden Funktionen des Prototyps. Nutzer fragten nach mehr Transparenz darüber, auf welchen Kriterien die Vorschläge basieren, wie z. B. durch die Anzeige von Tags, die das Thema der Übereinstimmung indizieren. Auch das Datum und der Name des Chats wurden als wichtige Informationen hervorgehoben.

Fazit:

- Variante - Prototyp 2.1:
    - Der Input wurde als Nachrichten-Editor interpretiert.
    - Die Nachricht wurde als Prompt formuliert.
    - Die Vorschläge waren nicht klar als Suchresultate erkennbar.
    - Es gab keinen signifikanten Zeitgewinn im Vergleich zu einem normalen Chat-Bot.
- Variante - Prototyp 2.2:
    - Der Input wurde als Suchfeld interpretiert.
    - Die Nachricht wurde eher als Suchtext formuliert.
    - Die Suchresultate waren klar und als solche erkennbar.
    - Ein kleiner Pain-Point war der Such-Button, der sich am Ende der Seite befand, was einen geringen Zeitverlust verursachte.
    - Die Suchvorschläge wurden häufiger genutzt.

**User Testing 2**
In der zweiten Testphase konzentrierten wir uns auf die Optimierung des Layouts und die Benutzerführung. Der Input wurde erfolgreich als "Message/Text-Input" erkannt, und die Positionierung des Eingabefelds oben im Bildschirm wurde von den Nutzern positiv bewertet.
Fazit:

- Startscreen:
  - Das Eingabefeld sollte oben positioniert sein.
  - Der "Senden"-Button sollte sich im oder direkt beim Eingabefeld befinden.
- Similar Prompts/Vorschläge:
  - Vorschläge wurden besser erkannt, wenn das Eingabefeld oben positioniert war.
  - Eine vertikal gestreckte Ansicht der Vorschläge wurde leicht bevorzugt.
  - Titel wie "Vorschläge" oder "Chats" wurden gewünscht.
- Chat-Verlauf:
  - Die Funktion "Continue here" sollte klarer oder besser beschrieben werden.
  - Enter sendet die Nachricht ganz unten.

Die Ergebnisse dieses Tests bestätigten, dass eine klare Kennzeichnung der Suchresultate sowie eine intuitive Positionierung der Eingabefelder und Senden-Buttons entscheidend für die Benutzerfreundlichkeit ist. Diese Erkenntnisse flossen direkt in das Design des Prototyps ein.

**User Testing 3**
In der dritten Testphase lag der Fokus auf der Feinabstimmung des Layouts und der Lesbarkeit der Nachrichten. Die Tester fanden das Tool vielversprechend, stellten jedoch fest, dass einige Nachrichten schwer zu lesen waren, wenn viele Nachrichten angezeigt wurden.
**Fazit**

- Das Layout und die Struktur wurden als klar und verständlich empfunden.
- Die Funktion „Hier senden“ könnte noch prominenter hervorgehoben werden.
- Nachrichten waren teilweise schwer zu lesen, da relativ viele angezeigt wurden.
- Gesamturteil des Testers: Ein spannendes Tool, das tatsächlich in bestimmten Situationen eine Erleichterung bieten könnte.

**Zusammenfassung**
Die User-Tests haben uns wertvolle Einblicke in die Präferenzen und Erwartungen der Benutzer gegeben. Die Tests zeigten, dass die klare Kennzeichnung von Suchresultaten, eine intuitive Navigation sowie eine prominente Platzierung der wichtigen Bedienelemente wesentlich für die Akzeptanz und Nutzung des Tools sind. Basierend auf diesem Feedback wurde der Prototyp kontinuierlich angepasst, um die Benutzerfreundlichkeit zu maximieren und die Effizienz des Content-Erstellungsprozesses sowie der Entwicklungstätigkeiten zu steigern.

Weitere Details sowie der genaue Ablauf der User Tests befinden sich im Anhang.

## 3.2 Prototyp für MVP

In diesem Abschnitt wird der finale, für die Implementierung validierte Prototyp als Wireframe dargestellt und die dahinterstehende Idee erläutert. Die vorangegangenen Prototypen, die aus den User-Tests hervorgegangen sind, befinden sich im Anhang.

Der finale Prototyp wurde insgesamt dreimal angepasst und durchlief drei User-Tests.

Der Fokus lag dabei auf einem schlichten, übersichtlichen Design, das die Funktionalität der "Similar Prompts" in den Vordergrund rückt.

**Dashboard: Initiale Ansicht**

Dies ist die Startansicht. Auf der linken Seite befindet sich, wie gewohnt, ein Verlauf der vorherigen Konversationen. Im Zentrum steht ein einzelnes Eingabefeld, das beim Tippen nach oben wandert, um Platz für die asynchron angezeigten Vorschläge (Similar Prompts Vorschläge) zu schaffen.

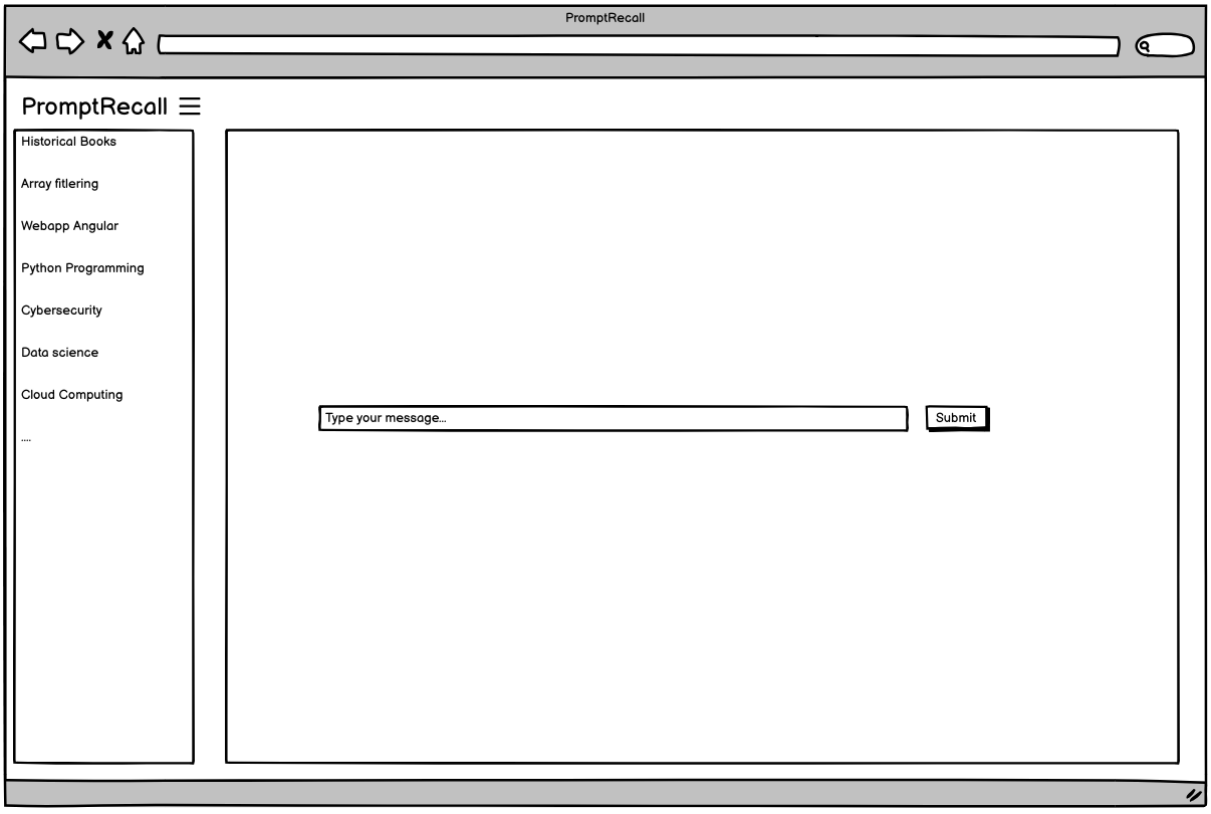



*Abbildung 4 - Wireframe der initialen Ansicht*

**Similar Prompts: Ansicht der Live-Vorschläge beim Verfassen des Prompts**

Sobald ein Benutzer beginnt, einen Prompt zu verfassen, durchsucht das System im Hintergrund automatisch frühere Konversationen nach ähnlichen Inhalten. Die Treffer werden geordnet nach Relevanz unter dem Eingabefeld angezeigt. Die Schlüsselwörter werden dabei generalisiert und als Tags in den Vorschlägen hervorgehoben.

Der Benutzer kann dann entscheiden, ob er in eine der vorgeschlagenen Konversationen einsteigt (durch Klicken auf "Continue here"), um den Kontext zu vertiefen oder eine bereits erhaltene Antwort zu nutzen, oder ob er eine neue Konversation startet, indem er auf "Submit" klickt.

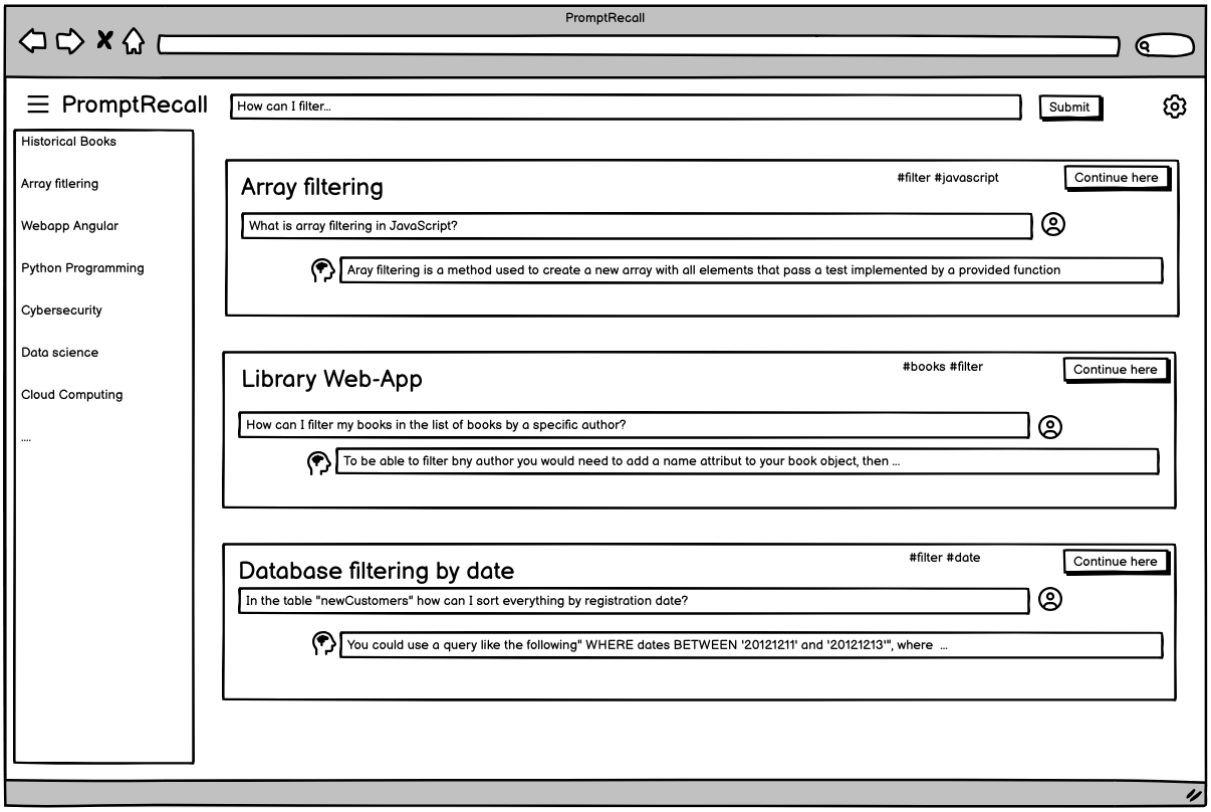



*Abbildung 5 - Wireframe Vorschläge*

**Chat: Ansicht in einem geöffneten Chat**

Entscheidet sich der Benutzer für einen Vorschlag, wird er in den entsprechenden Chatverlauf weitergeleitet. Hier wird der gesamte Verlauf der Konversation angezeigt, und der Benutzer kann die Unterhaltung fortsetzen oder direkt auf eine spezifische Antwort des LLMs reagieren. Wenn auf eine spezifische Nachricht des LLMs geantwortet wird, wird im Hintergrund eine neue Konversation ab dem Zeitpunkt der gewählten Antwort eröffnet.

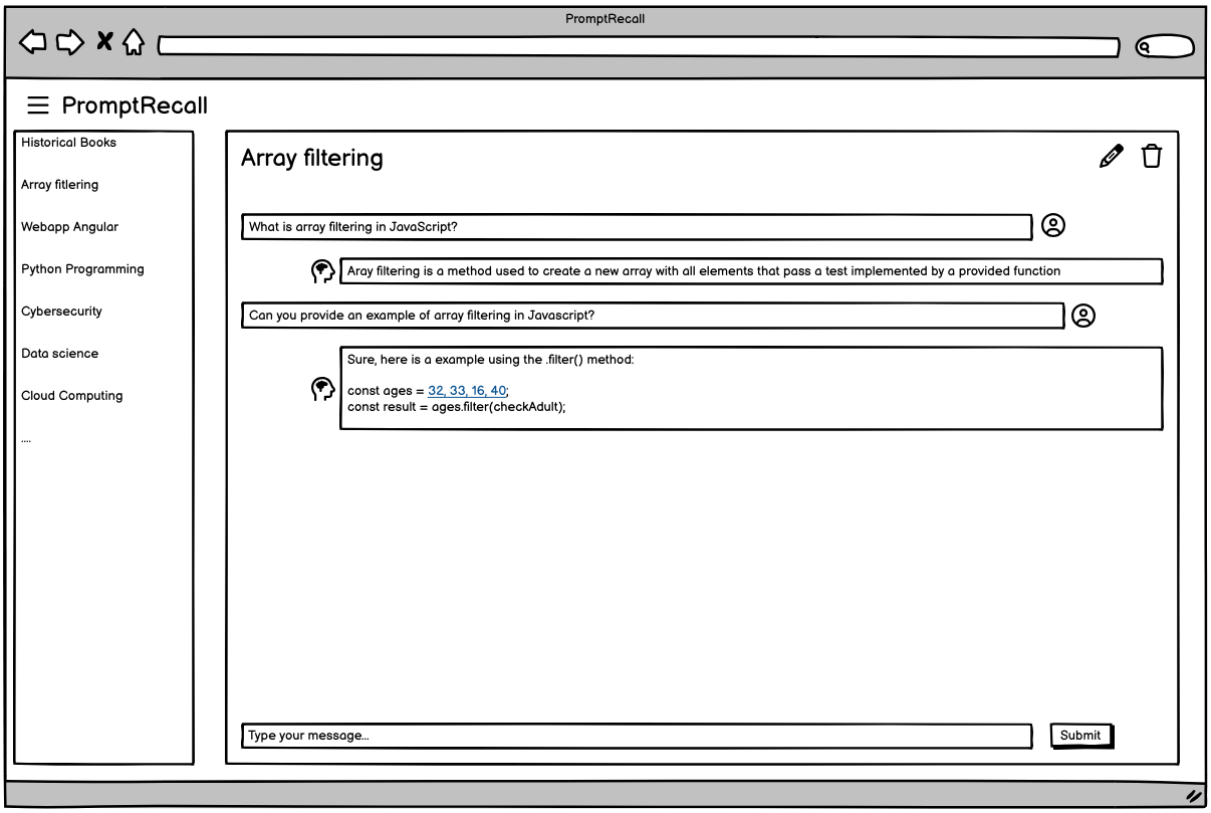



*Abbildung 6 - Wireframe geöffneter Chat*

# 4 Implementierung

In diesem Kapitel werden die zentralen Punkte der technischen Umsetzung beschrieben. Zuerst wird der Aufbau des Systems gezeigt und die Interkationen zwischen den einzelnen Komponenten. Danach werden die einzelnen Teile genauer erläutert.
Zum Schluss wird auf die einzelnen Features und die Limitierungen der Implementierung eingegangen.

## 4.1 System

Die folgende Grafik gibt einen kurzen Überblick über die Architektur.

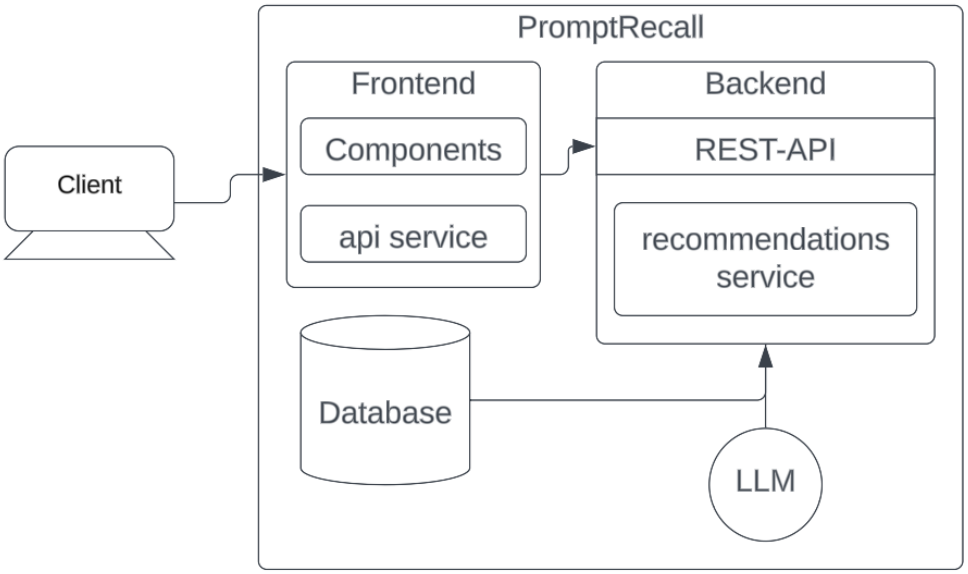



*Abbildung 7 - Systemaufbau*

Die Architektur und die verschiedenen Technologien wurden einerseits anhand der vorhandenen Kompetenzen im Team wie auch ihrer Einfachheit ausgewählt. Ziel war es Technologien zu nutzen, mit welchen der Prototyp umgesetzt werden konnte und die Überlegungen und Resultate aus der vorhergegangenen Forschung und Analyse verifiziert werden konnte. Die verschiedenen Teile des Systems wurden nicht hinsichtlich leistungsorientierter Gesichtspunkte wie Performance gewählt.

## 4.2 Frontend

Das Frontend der Applikation ist umgesetzt mit Angular. Für die Gestaltung des Designs wird das Design- und Komponenten-Framework Angular Material verwendet. Es dient als einzige Schnittstelle zu den Anwendern.

## 4.3 Backend

Das Backend wurde mit Python entwickelt. Es bietet eine REST-Schnittstelle für das Frontend, regelt den Zugriff auf die den Chat-Bot und dessen History und generiert Vorschläge anhand von Benutzerprompts. Die wichtigen Frameworks, die hier eingesetzt werden, sind Flask für die REST-Schnittstelle, Marshmallow für die Serialisierung der DTO und SQLAlchemy als Datenbank-Toolkit.

### 4.3.1 REST-API

Die folgende Tabelle gibt alle Routen an, die die API zur Verfügung stellt.

| **Pfad** | **Methode** | **Beschreibung** |
|---|---|---|
| **/chat** | POST | Erstellt einen neuen Chat |
| **/chat/<cid>** | GET | Gibt den Chat mit der ID <cid> zurück |
| **/chat/<cid>** | DELETE | Löscht den Chat mit der ID <cid> |
| **/chats** | GET | Gibt alle Chats zurück |
| **/message** | POST | Erstellt eine neue Nachricht |
| **/messages/<cid>** | GET | Gibt alle Nachrichten für den Chat mit der ID <cid> zurück |
| **/recommendations** | POST | Gibt Vorschläge zurück |

Alle Endpoints, die Contracts und der dazugehörige Code befinden sich im File api.py.

## 4.4 Datenbank

Für das Speichern der Chat-Nachrichten wird SQLite genutzt. Diese simple Methode wurde gewählt, da der Fokus mehr auf der Generierung der Vorschläge liegt und weniger auf der Optimierung des Datenmodells und der Speicherperformance.

### 4.4.1 Model

Das Datenmodell besteht aus zwei Entitäten. Das ist einmal die Entität «Chat» und einmal die «Message».

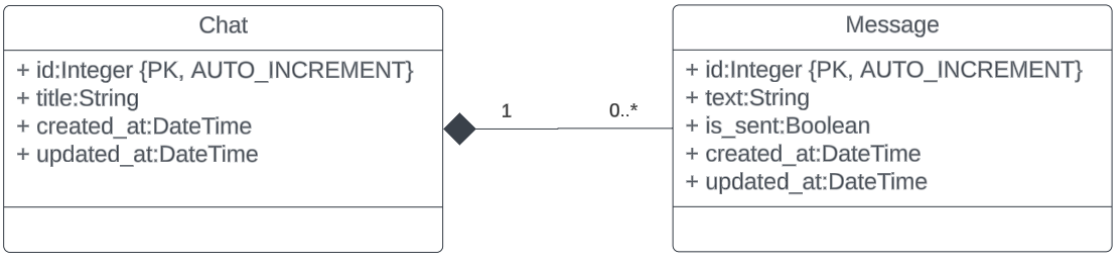



*Abbildung 8 – Datenmodell*

## 4.5 LLM

Für den Chat wird das Conversational-Model «facebook/blenderbot-400M-distill» verwendet. (https://huggingface.co/facebook/blenderbot-400M-distill) Das Model ist leichtgewichtig und eignet sich daher zur lokalen Nutzung auf dem Desktop.

## 4.6 Feature «Chats»

Ein Benutzer kann beliebig neue Chats erstellen, diese Anzeigen, Bearbeiten und Löschen. Eine Auflistung aller Chats befindet sich in der Seitennavigation, die oben Links geöffnet werden kann. In der nächsten Abbildung ist die geöffnete Seitennavigation mit den vergangenen Chats zu sehen.

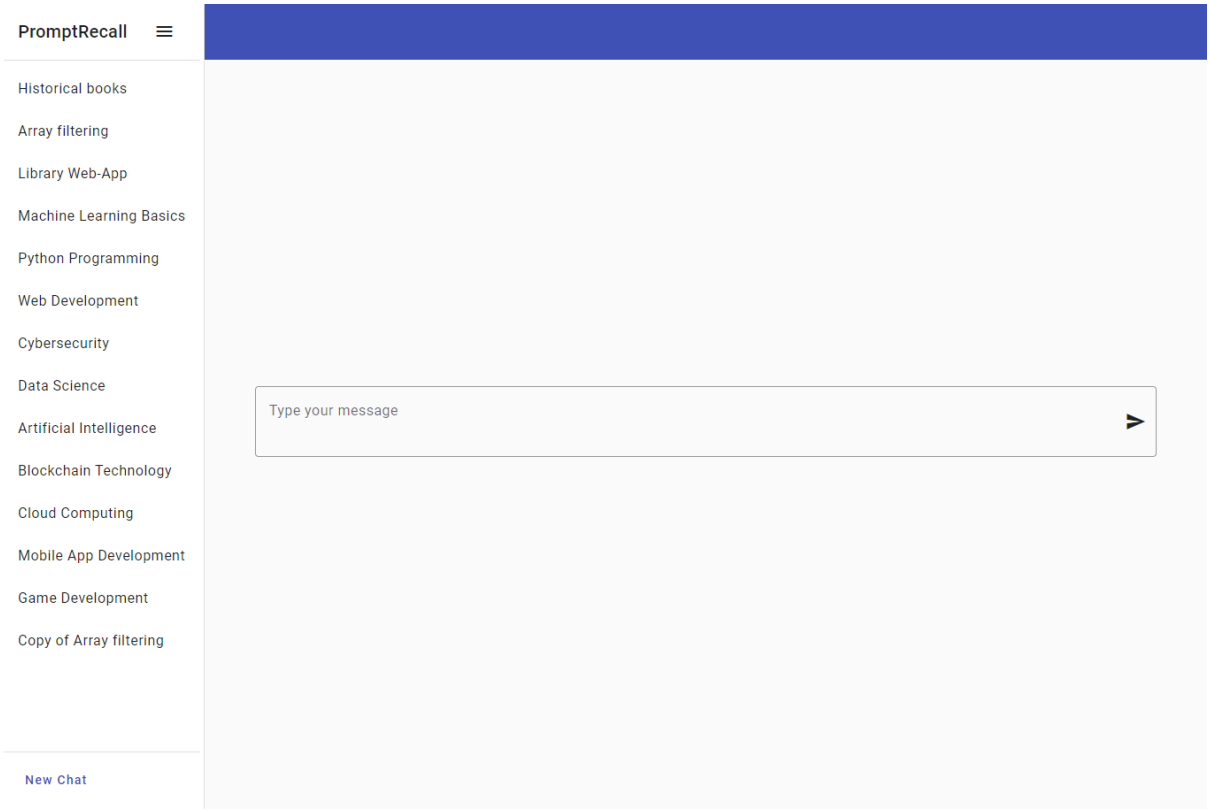



*Abbildung 9 - Startseite mit geöffneter Navigation*

Mit einem Klick auf den Namen können Chats geöffnet und eingesehen werden. Ist ein Chat geöffnet, kann oben rechts über die beiden Icons ein neuer Name gewählt werden oder der ganze Chat gelöscht werden. Die folgende Abbildung zeigt den Chat „Array filtering“ mit den Icons für das Bearbeiten und Löschen.

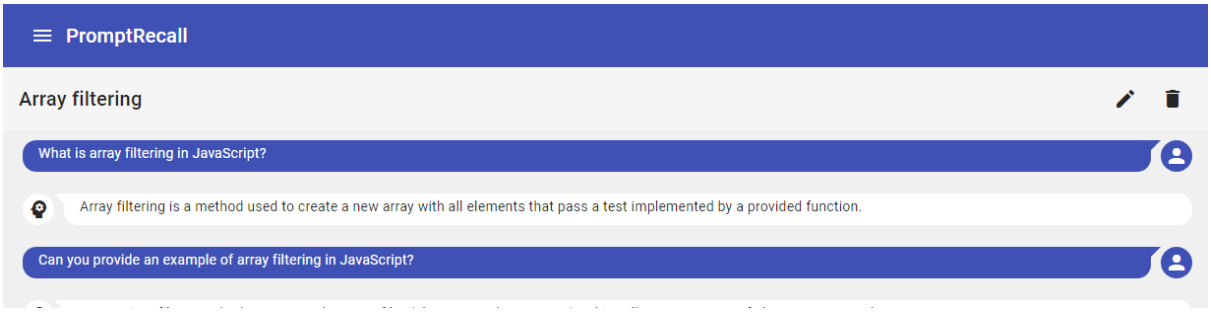



*Abbildung 10 - Action-Buttons eines Chats*

## 4.7 Feature «Similar Prompts»

Damit ein User möglichst keinen Mehraufwand hat für das Durchsuchen von existierenden Chats zu dem aktuellen Prompt, werden die Vorschläge automatisch anhand des Prompts generiert. Wenn der User noch keinen Text eingegeben hat, dann befindet sich die Texteingabe zentral auf dem Bildschirm. Erst wenn der User beginnt zu schreiben, verschiebt sich das Textfeld nach oben und die Vorschläge erscheinen unterhalb des Textfeldes.

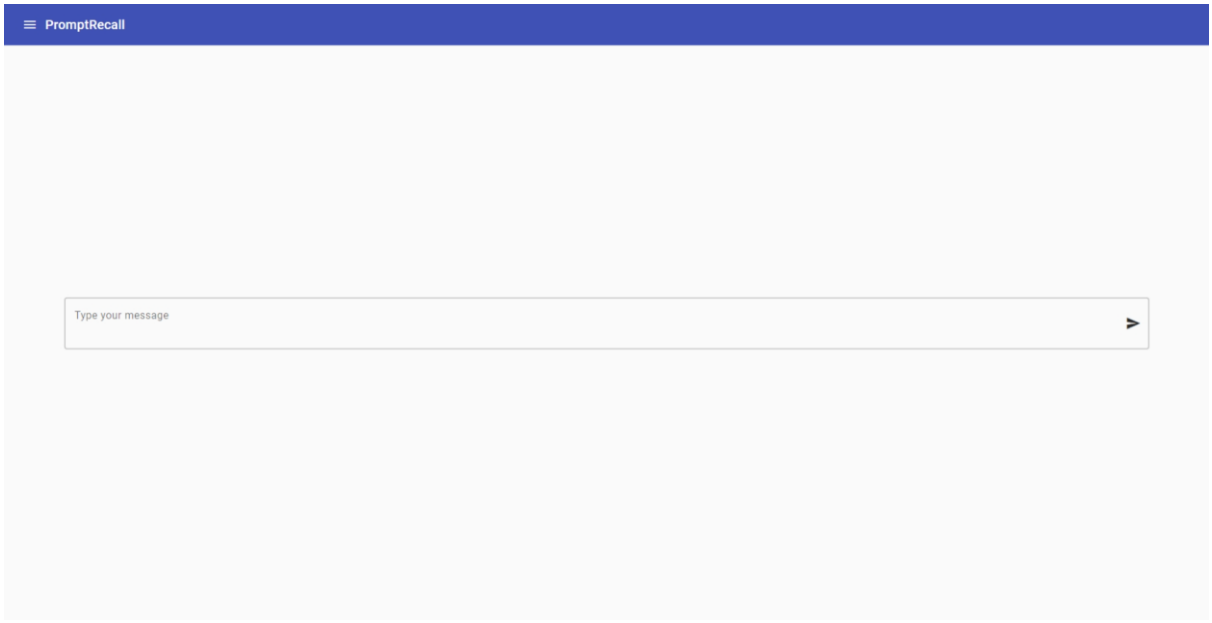



*Abbildung 11 - Startseite mit Eingabefeld für Prompts*

Die vorherige Abbildung zeigt den initialen Zustand, bevor der User einen Prompt eingibt. Sobald ein Zeichen eingegeben wird, ändert sich das Layout und die Vorschläge werden unterhalb des Feldes angezeigt, wie in der nächsten Abbildung zu sehen ist.

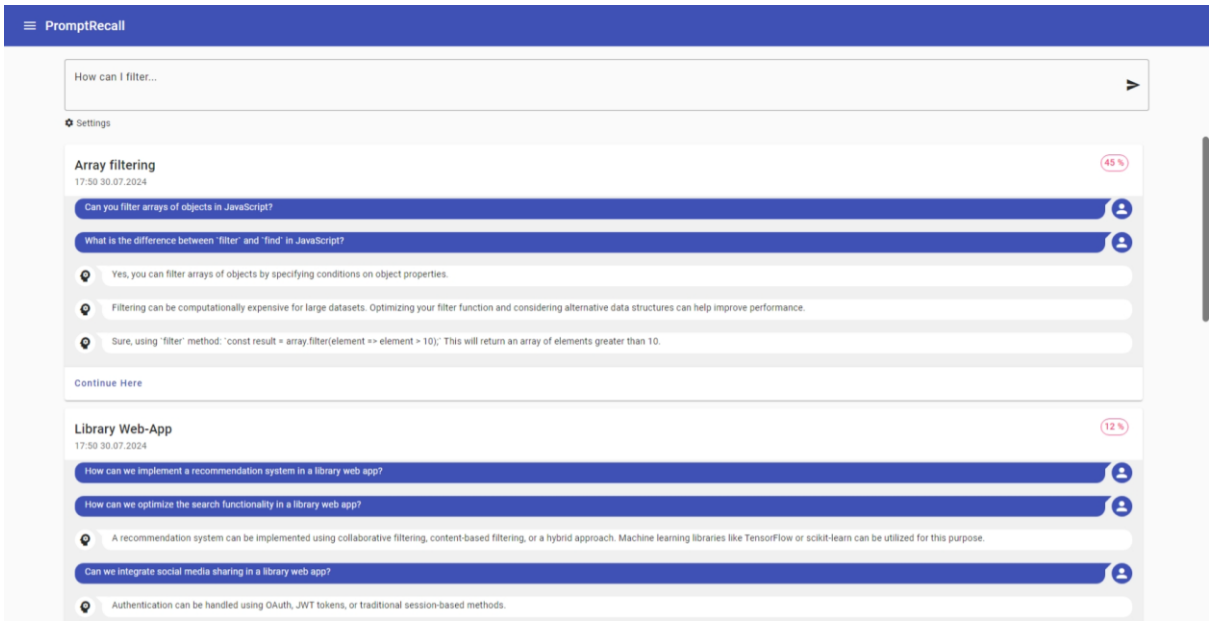



*Abbildung 12 - Vorschläge*

Je nach Suchmethode werden auf jedem Vorschlag nun entweder eine prozentuale Ähnlichkeit des Prompts zu diesem Vorschlag oder die übereinstimmenden Schlüsselwörter angezeigt. Die folgenden beiden Abbildungen zeigen jeweils ein Beispiel für eine prozentuale Ähnlichkeit und eine Auflistung der Schlüsselwörter.

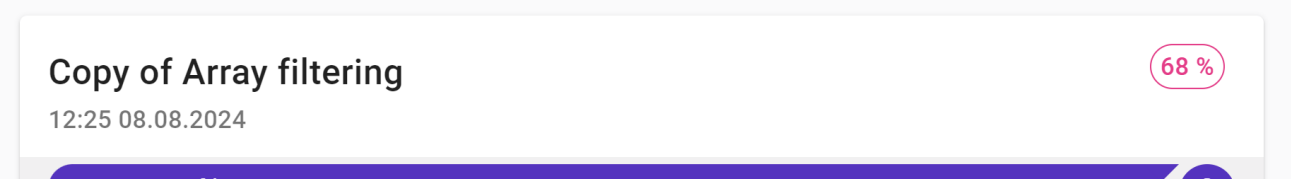



*Abbildung 13 - Prozentuale Übereinstimmung*

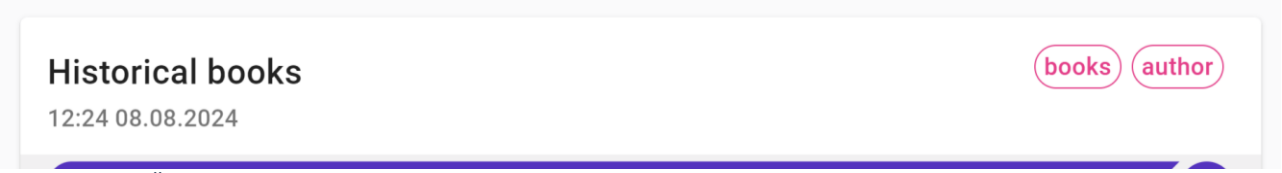



*Abbildung 14 - Übereinstimmende Schlüsselwörter*

Die Methode kann in den Einstellungen ausgewählt werden. Hier können zudem auch die Anzahl der Vorschläge, Anzahl Nachrichten pro Vorschlag und einige weitere Einstellungen zur Darstellung vorgenommen werden, wie man in der nächsten Abbildung sehen kann.

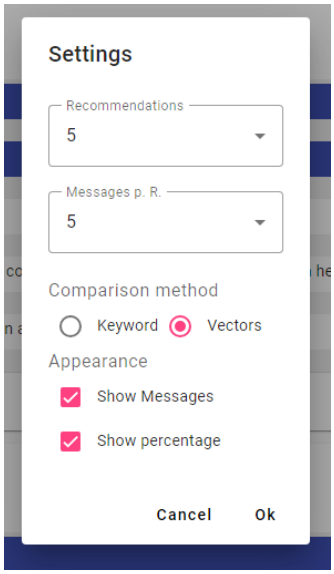



*Abbildung 15 - Einstellungen zu den Vorschlägen*

**Umsetzung**

Die grösste Komplexität liegt in der Generierung der Vorschläge. Anhand vom Prompt vom User sollen alle Nachrichten und die dazugehörigen Chats herausgefiltert werden, die möglicherweise für den Benutzer von Interesse sind. Das bedeutet, diese Nachrichten…

- Enthalten ähnliche Wörter, wie der Prompt
- Enthalten ähnlichen Themen, wie der Prompt

Im Folgenden wird genauer darauf eingegangen, wie diese beiden Kriterien bestimmt werden können. Der Source-Code dazu befindet sich im File recommendations.py.

**Wort-Vergleich**
Die Prüfung auf gleiche Schlüsselwörter ist trivial. Voraussetzung für einen Vergleich ist, dass die Texte in normalisierter Form und als Tokens separiert vorliegen. Ein Token ist in diesem Fall jedes Wort oder Teilwort, das durch Leer- oder Sonderzeichen von anderen Wörtern getrennt wird.
normalisiert bedeutet, dass Wörter aus Flexionsformen falls möglich in ihre Grundform gesetzt werden. Das Wort «iss» beispielsweise würde zu «essen». Die Methode, die dazu verwendet wird, nennt sich Lemmatisierung.

**Algorithmus:**

1. Alle Texte, das heisst der Prompt sowie die existierenden Nachrichten, werden in Tokens aufgeteilt
2. Sonderzeichen und Füllwörter werden aus den Tokens entfernt
3. Die Tokens werden lemmatisiert mit dem WordNetLemmatizer von nltk.
4. Die Tokens aus den Nachrichten werden verglichen mit den Tokens des neuen Prompts und alle Übereinstimmungen pro Nachricht bestimmt
5. Die Nachrichten werden gruppiert nach dem Chat, zu dem sie gehören und alle Schlüsselwörter pro Chat vereint
6. Diese Chats werden dem User, absteigend sortiert nach der Anzahl übereinstimmender Schlüsselwörter, vorgeschlagen

**Themen-Vergleich**
Die Ähnlichkeit von zwei Texten wird mit dem Vergleich der jeweiligen Text-Vektoren gemacht. Um die Nachrichten mit dem Prompt abzugleichen, wird wie folgt vorgegangen:

1. Alle Texte, das heisst der Prompt sowie die existierenden Nachrichten, werden in Tokens aufgeteilt
2. Sonderzeichen und Füllwörter werden aus den Tokens entfernt
3. Die Tokens werden lemmatisiert mit dem WordNetLemmatizer von nltk.
4. Die Tokens werden mit dem TfidfVectorizer von sklearn in Vektoren transformiert
5. Für jede Nachricht wird die Cosinus-Ähnlichkeit zwischen der Nachricht und dem Prompt berechnet. Dazu wird die Funktion cosine similarity von sklearn verwendet.
6. Die Nachrichten werden gruppiert nach dem Chat, zu dem sie gehören und die maximale Ähnlichkeit pro Chat bestimmt.
7. Diese Chats werden dem User, absteigend sortiert nach der berechneten Ähnlichkeit, vorgeschlagen

## 4.8 Feature „Im Kontext fragen“

Wenn die Frage des Users nicht allein durch die Vorgeschlagene Nachricht oder den weiteren Verlauf des Chats beantwortet werden kann, dann besteht die Möglichkeit einen Prompt direkt in diesem Kontext zu stellen. Dazu muss nur die bereits eingefügte Nachricht angeklickt werden (siehe Abbildung 16) und die Nachricht wird abgeschickt (siehe Abbildung 17)

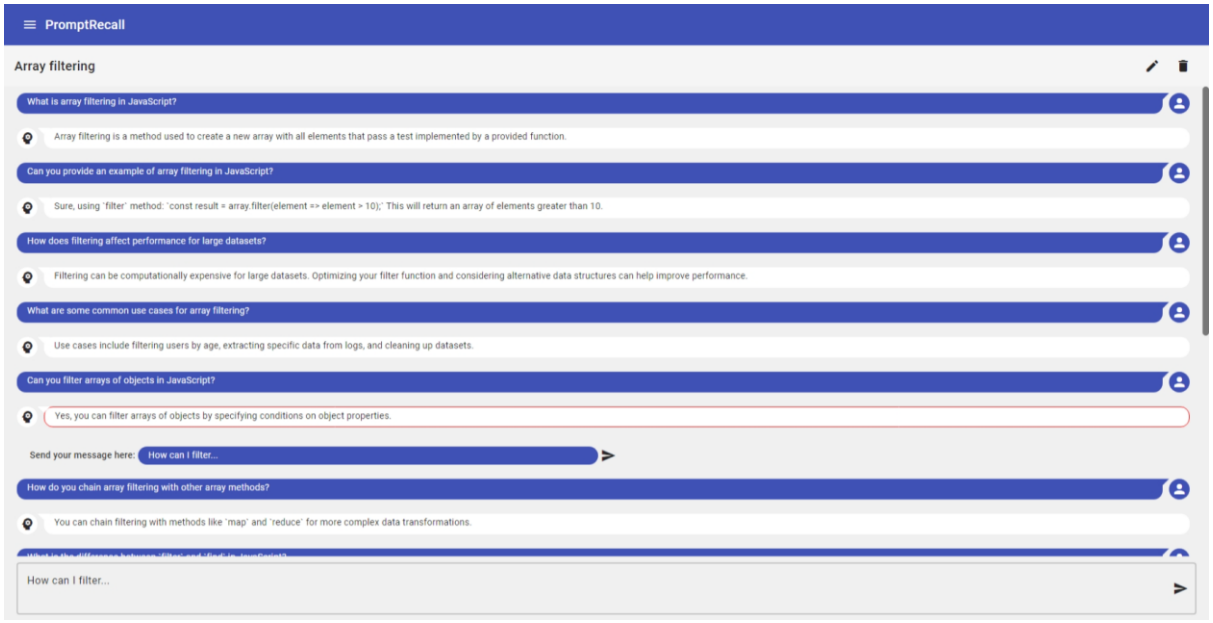



*Abbildung 16 - Chat-Verlauf mit einer Nachricht im Fokus*


*Abbildung 17 - Chat-Verlauf nach dem Senden einer Nachricht im Kontext*

Die kopiert die bestehende Konversation inklusive aller Nachrichten bis und mit der Vorgeschlagenen Nachricht und fügt den neuen Prompt als neue Nachricht ein. Dies löst auch direkt eine Antwort des LLM aus, die auch unmittelbar nach dem Prompt angezeigt wird.

## 4.9 Technische Einschränkungen

- Die Lemmatisierung der Texte funktioniert nur für die englische Sprache
- Der Bot funktioniert nur in Englisch
- Der Chat-Bot kann nur die letzten 128 Zeichen verarbeiten und nicht die ganze Historie
- Die Performance ist nicht ausreichend für grosse Datenmengen, da die Analyse der Nachrichten jedes Mal beim Abfragen der Vorschläge durchgeführt wird.

# 5 Diskussion

Das Ziel dieser Arbeit war es die Pain Points der User in Bezug auf die momentan vorhandenen LLM-Chatbots zu untersuchen und daraus eine innovative, neue Lösung zu finden, welche mit vielem Prototyping und User Tests validiert werden.
Nachfolgend befinden sich die Zusammenfassung der Ergebnisse sowie die Limitationen zur implementierten Lösung und ein möglicher Forschungsausblick.

## 5.1 Zusammenfassung der Ergebnisse

**Forschungsfrage A**
Wie können wir es dem Benutzer erleichtern im richtigen Kontext seine Frage dem LLM zu stellen?

Ja, mit der Funktionalität der «Similar Prompts» ist es für den Benutzer nun möglich jegliche vergangene Konversation wiederzufinden, um im korrekten Kontext weiterzuführen. Nicht möglich ist, direkt einen «generierten und korrekten» Kontext zu haben, ohne jemals etwas in der Richtung des Prompts gefragt zu haben, was aber mehr in Richtung LLM-Training anstelle von Prompt-Management geht.
Folge dessen haben wir unserer Meinung nach die Forschungsfrage A vollkommend beantwortet.

**Forschungsfrage B**
Wie lässt sich ein spezifischer Prompt der Vergangenheit möglichst einfach und effizient in der Historie finden?

Die Umfrage hat gezeigt, dass vor allem das Bedürfnis einer Suche besteht.
Es ist möglich, asynchron nach vergangenen Prompts und Chats zu suchen während des Verfassens eines Prompts respektive einer Suchanfrage. Die User Tests haben gezeigt, dass sobald eine Suche verbunden wird mit dem klassischen Eingeben eines Prompts, relevante Prompts sehr schnell gefunden wird, sodass wir auch diese Forschungsfrage vollkommend beantwortet haben.

**Forschungsfrage C**
Wie lassen sich LLM-Prompts bewerten, klassifizieren und besser visualisieren?

Zur Klassifizierung haben wir in unseren Prototypen auf eine Klassifizierung in Schlüsselwörter und eine Erfassung des Themas durch Vektorisierung gesetzt.
Schlüsselwörter können trivial als Tags und Übereinstimmung von Themen durch Prozente ausgedrückt werden.
Die einzelnen Prompts werde nach Relevanz angezeigt und klassifiziert, sodass eine Klassifizierung stattfindet.
Die Prompts werden in einer Liste, mit Tags versehen und in einem einstellbaren User Interface visualisiert. Diese Forschungsfrage haben wir unserer Meinung nach zu etwa der Hälfte beantwortet, da wir der Meinung sind, dass noch etwas mehr Arbeit in die User Experience und User Interface Design machbar wäre.

## 5.2 Forschungsausblick / Limitationen

Für einen produktiven Einsatz des Prototyps wären noch die folgenden Punkte zu implementieren:

- Das LLM sollte ausgetauscht werden, gegen einen ChatBot. Das aktuelle Model dient lediglich als ein Proof of Concept.
- Für einen Einsatz im deutschsprachigen Raum muss die Sprachverarbeitung und der Chat erweitert werden, dass auch Deutsch verwendet werden kann.
- Als Datenbank fungiert aktuell eine SQLite-DB, diese müsste durch eine geeignetere Datenstruktur – z.B. eine Vektordatenbank, ersetzt werden.
- Die Performance lag nicht im Fokus bei der Generierung der Vorschläge. Für einen Einsatz mit grösseren Datenmengen wäre es nötig Informationen wie die Schlüsselwörter und Themen bereits beim Senden oder asynchron zu berechnen.

Zudem gibt es viele Features, die für eine produktive Nutzung sehr interessant sein könnten. Einige Beispiele dazu wären:

- Manuelles Tagging der Chats und Prompts
- Qualifizierung der Vorschläge durch den Benutzer
- Löschen von Vorschlägen
- Darstellung der Hierarchie der einzelnen Chats, die durch das „Nachrichten im Kontext senden“ entstehen

Weiter untersuchen müsste man die Generierung der Vorschläge. Aktuell kann man wählen zwischen einem Textabgleich „Wort per Wort“ und einem Vergleich der Vektorisierten Texten. Für ein optimales Ergebnis müsste genauer ermittelt werden, welche dieser Methoden besser geeignet ist und welche Schwellenwerte und Vergleichsfunktionen hier am besten geeignet sind. Zudem sind auch weitere Methoden zur Generierung der Vorschläge denkbar. Beispielsweise eine Klassifizierung und Vergleich der Themen einer Nachricht durch AI oder auch eine laufende Anpassung der Vorschläge durch Aufzeichnen und Lernen vom Benutzerverhalten, wären spannende Ansätze.

Eine weitere Frage, die sich, während dem Projektablauf gestellt hat, wäre, ob mit diesem Konzept auch Energie eingespart werden könnte. Dazu müsste der Energieverbrauch des Vorschläge-Generierens verglichen werden mit dem des LLM unter Berücksichtigung des effektiven Benutzerverhaltens.

Zuletzt wäre es auch interessant festzustellen, ob allein durch das Erscheinen der Vorschläge die Prompt-Eingabe eines Benutzers beeinflusst wird und allenfalls sogar verbessert wird.

# 6 Fazit

Das Feature «Similar Prompts» ermöglicht es den Usern, thematisch relevante Chats ohne zusätzlichen Aufwand zu durchsuchen. Das positive Feedback und die ersten Erfahrungen mit dieser Funktion unterstreichen das Potenzial, das durch weitere Forschung und Entwicklung noch gesteigert werden könnte.

Auffällig ist, dass es derzeit nur wenige Tools gibt, die Funktionen zur Verwaltung von Prompts, wie eine Suchfunktion, beinhalten. Diese Arbeit hat gezeigt, dass es sowohl Anwendungsfälle als auch ein starkes Bedürfnis nach solchen Features gibt. Dies spricht dafür, die Entwicklung in diesem Bereich voranzutreiben.

Der Proof of Concept funktioniert gut und zeigt das Potenzial der entwickelten Lösung. Für einen produktiven Einsatz sind jedoch weitere Tests, Ausbauten und der Austausch des LLMs und der Datenbank notwendig. Auch an den Algorithmen zur Generierung der Vorschläge kann je nach Aufgabengebiet noch Verbesserung angebracht werden.

Die eingesetzten Methoden und Techniken haben sich als effektiv erwiesen. Besonders wertvoll war die iterative Validierung der Prototypen durch User-Tests, die es ermöglichte, Annahmen direkt zu überprüfen. Weniger erfolgreich war der Versuch, mehrere Features gleichzeitig zu entwickeln, was aufgrund der begrenzten Ressourcen nicht in der gewünschten Qualität realisiert werden konnte. Daher wurde der Fokus auf wenige, aber qualitativ hochwertige Features gelegt.

Diese Arbeit legt den Grundstein für zukünftige Entwicklungen im Bereich des Prompt Managements, die durch weitere Forschung und Optimierung zu einem nützlichen und produktiven Tool ausgebaut werden können.

# 7 Verzeichnisse

Folgend sind alle Verzeichnisse der Arbeit abgelegt.

## 7.1 Literaturverzeichnis


[1] „ChatGPT,“ OpenAI, 2024. [Online]. Available: https://chatgpt.com/.

[2] Meta, „Llama 2,“ 2024. [Online]. Available: https://llama.meta.com/.

[3] A. F. a. M. K. Nitish Patkar, «Challenges and Opportunities for Prompt Management: Empirical Investigation,» University of Applied Sciences and Arts, Switzerland, 2024.

[4] „AnonChatGPT,“ 2024. [Online]. Available: https://anonchatgpt.com/.

[5] „utopia.de,“ 23 Februar 2023. [Online]. Available: https://utopia.de/chat-gpt-und-die-klimakrise-experten-warnen_474199/#:~:text=Studien%20zu%20dem%20Thema%20kommen,10%20Gramm%20CO2e%20pro%20Anfrage..

## 7.2 Abbildungsverzeichnis

# Anhang

Der Anhang unserer Arbeit enthält zusätzliche Informationen und Materialien, die zur Unterstützung und Vertiefung der in der Arbeit dargestellten Inhalte dienen. Diese Anhänge bieten detaillierte Einblicke und ergänzende Daten, die für das Verständnis unserer Forschung und Methodik von Bedeutung sind.

# A1 Projektausschreibung

Fachhochschule Nordwestschweiz

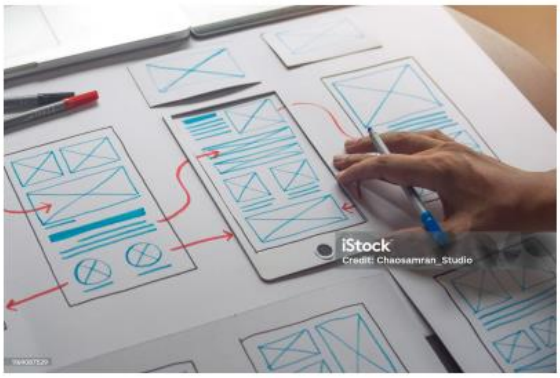
Design UX | ML Alg

## 24FS_IMVS10: Innovative Prompt Management for LLMs

| | | | Priority 1 | Priority 2 |
|---|---|---|---|---|
| **Advisor:** | Martin Kropp<br>Nitish Patkar | **Work scope:** | P5 oder P6 | --- |
| | | **Team size:** | 2er Team | --- |
| **Languages:** | German or English | | | |
| **Study course:** | Computer Science and Data Science | | | |

**Initial position**

Our daily interactions with generative AI tools result in hundreds of prompts. It is easy to lose track of all the prompts (some of them are useful, but many of those are exploratory). Popular and heavily used chatbot tools, such as ChatGPT or Bard, provide almost no functionality to manage prompts. There are a few other ways of managing prompts, e.g., tools like prompt box, or even more primitive ways like using document management systems. These tools only allow storage and edit prompts without any possibility of executing those.




**Objective**

The project's overarching goal is to build a proof-of-concept chatbot for LLMs with an innovative user interface. The UI should help end-users to manage and execute prompts more efficiently than the existing ones. To begin with, we want to understand current challenges with prompt management so that appropriate solution ideas can be generated. For the implementation, we can focus on open-source LLMs. Validating the solution ideas and the implementation is equally important for the project's successful conclusion.

**Problem statement**

The concrete tasks include:

- Conduct a survey and interviews to explore the challenges with prompt management. Apart from exploring obvious factors, such as efficient ways of managing prompts (e.g., having a folder structure, search functionality), students can delve into further interesting dimensions, such as cognitive efforts required vs. number of consecutive prompts in a single chat thread, effects of being or leaving the chat context, etc.)
- Conduct a market analysis of similar tools.
- Develop an innovative concept for prompt management and execution. Explore which metaphors are interesting for (partial) replication, like managing prompts as a book with chapters or themes, or an interactive notebook like Jupyter, using labels, etc. Students are expected to be creative with this task.
- Create an MVP by implementing the proposed concept.
- Validate the solution to determine its promises and shortcomings.

**Technologies/Technical emphasis/References**

To be decided together with students

## A2 Projektvereinbarung

Fachhochschule Nordwestschweiz

Windisch, 13.08.24

**Projektvereinbarung**
**IP5, 24FS_IMVS10: Innovative Prompt Management for LLMs**

**Betreuer:** **Martin Kropp**
**Nitish Patkar**

**Auftraggeber:** Fachhochschule Nordwestschweiz FHNW

**Projektdauer:** 19.02.2024 bis 16.08.2024 (Abgabetermin)

| Version | Bemerkungen | Autor |
|---|---|---|
| 1 | Draft | Jack Gläser, Simon Lüscher |
| 2 | Draft überarbeitet | Jack Gläser, Simon Lüscher |
| 3 | Finale version | Jack Gläser, Simon Lüscher |

## 1. Ausgangslage

In der heutigen digitalen Kommunikationslandschaft und der täglichen Nutzung von LLMs sammeln sich schnell eine Vielzahl von Prompts an, insbesondere in der Interaktion mit Large Language Models (LLMs). Diese Prompts, die häufig in Form von Texteingaben oder Sprachanfragen auftreten (Neuerdings auch Bilder / Videos), werden derzeit jedoch nur in der Reihenfolge ihres Eingangs gespeichert und können lediglich nach Datum abgerufen werden. Diese Einschränkung führt zu einer erschwerten Handhabung und einem ineffizienten Management von Prompts, insbesondere wenn diese in großen Chatverläufen oder Kommunikationsverläufen auftreten. Ausserdem hat man als Benutzer solcher LLMs keinerlei Features zur Sortierung, Filterung oder Klassifizierung der vergangenen Prompts.

Bei der Analyse von unterschiedlichen LLM-Interfaces und Tools in diesem Bereich stellt man fest, dass es keine grossen Abweichungen oder Innovationen von diesem Weg zur Prompt-Organisierung gibt. Auch gibt es keine nennenswerten wissenschaftliche Werke, die sich mit Prompt Management auseinandersetzen. Eine Studie aus dem Jahr 2023, die sich mit dem Prompting durch Non-AI-Experten befasst, hat aber gezeigt, dass bereits das Formulieren von Prompts und Interagieren mit einem Chat-Bot für diese eine Herausforderung darstellt. Eine mögliche Begründung, die dazu genannt wird, ist, dass falsche Annahmen getroffen werden aus der Erfahrung von realer Mensch-zu-Mensch-Interaktion. (Zamfirescu-Pereira, J. D., Wong, R. Y., Hartmann, B., & Yang, Q. (2023, April). Why Johnny can't prompt: how non-AI experts try (and fail) to design LLM prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (pp. 1-21).)

Wir haben verschiedene momentan aktuelle AI-Chatbots (OpenAI ChatGPT 3.5, Meta Llama 2, Google Gemini und AnonChatGPT) analysiert und kamen zu folgendem Fazit:
Viele der AI-Chatbots sind «fine-tuned» für spezifische Aufgabengebiete. Alle haben etwas andere Einsatzzwecke oder kleine Unterschiede in den Funktionalitäten, alle haben jedoch ein klares Problem: So gut wie kein Prompt-Management Feature ist vorhanden.

Die wichtigsten Probleme, die bei bestehenden LLM-Anwendungen vorherrschen wären:

**Mangelnde Filtermöglichkeiten:** Da man momentan Prompts nicht nach spezifischen Kriterien filtern kann ist es für Benutzer schwierig, relevante Informationen schnell finden zu können oder wenigstens den Rahmen einzugrenzen.

**Fehlende Organisationsoptionen:** Das Fehlen von Funktionen wie Tags oder Labels macht es schwierig, Prompts zu kategorisieren und zu organisieren, was zu einem unstrukturierten Prompt Management führt.

**Unvollständige oder unklare Prompts:** Da der erste Prompt selten perfekt ist, führen längere Kommunikationsverläufe häufig zu einer Fragmentierung von Informationen und einem erhöhten Bedarf an Rückverfolgbarkeit. Hier wäre es Ideal, wenn man aus den verschiedenen Prompts letztendlich einen zusammengefassten Prompt oder eine Prompting-Struktur zu dieser Thematik zur Ablage erhalten würde.

**Variabilität der Themen und Antworten:** Prompts können verschiedene Themen umfassen, und die gesuchten Antworten können von Text über Code bis hin zu Korrekturen variieren, was die Suche von Informationen erschwert. Eine einfache Suche wäre hier zwar ein Anfang aber auch nicht ideal im Hinblick auf die verschiedenen Output-Varianten.

In Anbetracht dieser Herausforderungen ist es offensichtlich, dass ein innovatives Prompt-Management-System benötigt wird, welches die Effizienz und Benutzerfreundlichkeit verbessern, indem es die Möglichkeit bietet, Prompts beispielsweise zu filtern, zu organisieren oder gezielt aufzurufen.
Durch die Einführung einer solchen Lösung können Benutzer ihre Interaktionen mit LLMs optimieren und die Produktivität mit ihnen kann gesteigert werden.

## 2. Projektvision

Unsere Vision ist, eine intuitive, innovative und verbesserte Interaktion mit LLM-Prompts zu erschaffen, welche die Effizienz steigert, die Nachvollziehbarkeit erhöht und kein spezielles Vorwissen dafür erfordert.

Durch die Entwicklung einer Plattform, die eine reibungslose und organisierte Handhabung von Prompts ermöglicht, streben wir folgende Ziele an:

**Effizienzsteigerung:** Unsere Lösung soll es den Benutzern ermöglichen, Prompts inklusive des Verlaufs schnell und effizient zu durchsuchen, zu filtern und relevante Informationen zum Inhalt und der Semantik zu extrahieren/zu finden.
Durch eine verbesserte Navigation und Suchfunktion wird die Suchzeit reduziert, wodurch die Gesamteffizienz der Nutzer gesteigert wird.

**Erfassung der Semantik**: Durch die Implementierung von Funktionen zur Kategorisierung (Tagging und Organisation) von Prompts streben wir an, die Nachvollziehbarkeit von Diskussionen mit den LLMs zu verbessern. Benutzer sollen in der Lage sein, den Verlauf von Interaktionen leicht zu verfolgen und vergangene Informationen schnell wiederzufinden, was zu einer besseren Transparenz und Dokumentation führt.

**Benutzerfreundlichkeit:** Unser Ziel ist es, eine Plattform zu bauen, welche intuitiv, benutzerfreundlich und einfach zu bedienen ist. Das Ziel ist es, dass die User Experience auf einem Level ist, welche den Nutzern erlaubt unsere Funktionalitäten zu nutzen ohne spezifisches Vorwissen mitzubringen.

Durch die Realisierung dieser Vision streben wir danach, eine Lösung zu schaffen, die die Arbeitsweise mit LLM-Prompts grundlegend verbessert.
Wir wollen die momentanen «pain-points» mildern und die Zufriedenheit der Benutzer von LLMs steigern. Unsere Qualitätskriterien werden sich auf die Messung der Effizienz der Suche, gemessen an der Zeit und der Länge der User Journey des Durchsuchens der Prompt-Verläufen und Benutzerfreundlichkeit konzentrieren, um sicherzustellen, dass die gesteckten Ziele erreicht wurden. Dies werden wir mit User und anhand Literatur in diesem Bereich validieren.

## 3. Ziele und Fragestellungen

Das Ziel dieser Arbeit ist es bestehende LLMs in Bezug auf ihr Prompt-Management zu analysieren und nach Möglichkeiten zu suchen, dieses innovativer und benutzerfreundlicher zu gestalten. Es gilt herauszufinden, welche Features bei bestehenden Lösungen vermisst werden und welche Pains bestehen.

Aus dieser Analyse soll ein Prototyp entstehen, der anhand dieser Erkenntnisse ein innovatives Prompt-Management-Konzept veranschaulicht und demonstriert.

### 3.1 Ziele

1. Klar strukturierte Prompt-Historie, die das Durchsuchen erleichtert.
2. Eine Klassifizierung von Prompts anhand ihres Inhalts und dem Charakter ihrer Antworten.
3. Eine Schnellere Identifikation und Erreichbarkeit des optimalen Prompt-Verlaufs für eine gegebene Fragestellung.
4. Optionales Ziel: Prompts bearbeiten zu können.

### 3.2 Forschungsfragen:

A. Wie können wir es dem Benutzer erleichtern im richtigen Kontext seine Frage dem LLM zu stellen?
B. Wie lässt sich ein spezifischer Prompt der Vergangenheit möglichst einfach und effizient in der Historie finden?
C. Wie lassen sich LLM-Prompts bewerten, klassifizieren und besser visualisieren?

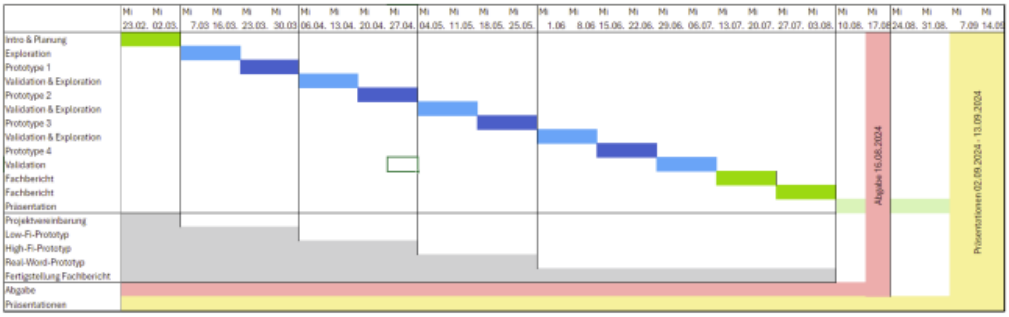
## 4. Methodik

Da diese Arbeit ein Hohes Mass an Flexibilität und Reaktivität erfordert, um auf Ergebnisse aus Analysen und Testings einzugehen, haben wir ein iterativer Ansatz zur Durchführung gewählt. Dieser eignet sich auch deswegen, weil beide Mitglieder im Projektteam professionelle Erfahrung im Einsatz solcher Methoden haben.

Da auch die Analyse und das Verifizieren von möglichen Lösungskonzepten eine wichtige Rolle spielen, werden auch Techniken aus dem Design Thinking eingesetzt werden. So soll dem konzeptuellen Teil der Arbeit genügend Raum gegeben werden. Konkret werden zweiwöchige Sprints durchgeführt, die immer abswechselnd einen Fokus auf Research, Prototyping und dann Validierung haben. Nach jedem Sprint werden bei einem Review-Meeting mit den Coaches, Fragen geklärt und es findet ein kurzer Austausch statt. Im Anschluss daran findet dann bilateral ein Planning für den nächsten Sprint und eine kurze Reflektion statt.

Als Plattform für die Arbeitsorganisation wird Github eingesetzt, da auch hier bereits einiges an Vorerfahrung vorhanden ist. Für die Erarbeitung des Prototyps werden diverse Techniken zum Einsatz kommen, wie zum Beispiel Low- und High-Fi-Prototypen, Umfragen und User-Testings.
Erkentnisse die aus der Validierung der Prototypen und aus den Interviews resultieren werden im Fachbericht dokumentiert werden.

## 5. Planung




**Beispiel unserer «Sprints» in den verschiedenen Iterationen:**

| **Sprint X.1: Research** | |
|---|---|
| Ziele | - Erkentnisse gewinnen und<br>- Erkenntnisse dokumentieren |
| Aktivitäten | Recherche, Umfragen |
| Deliverables | Resultate in jeglicher Form sowie Dokumentation |

| **Sprint X.2: Prototyping** | |
|---|---|
| Ziele | - Gewonnene Erkenntnisse umsetzen<br>- Fortschritt dokumentiert |
| Aktivitäten | Prototyp Desing, Implementierung |
| Deliverables | Applikation, Dokumentation |

| **Sprint X.3: Validation** | |
|---|---|
| Ziele | - Erkenntnisse sammeln, wie sich der Prototyp die Problemstellung löst<br>- Erkenntnisse in Bezug auf die Forschungsfragen finden |
| Aktivitäten | - Prototyp validieren mit Tools wie Usability-Testing, Umfragen, Interviews |
| Deliverables | - Dokumentation der Erkenntnisse |

## 6. Risiko Assessement

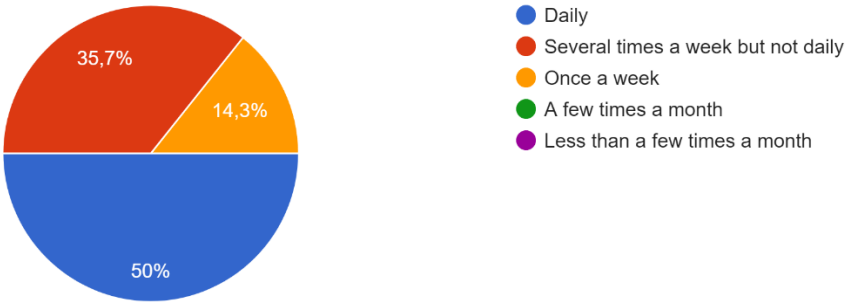
| Risiko | Eintrittsgefahr | Auswirkung | Präventionsmassnahme / Strategie |
|---|---|---|---|
| Zu hohes Arbeitsvolumen | Mittel | Überlastung der Teammitglieder, Qualitätseinbußen bei Arbeitsergebnissen | Realistische Ressourcenplanung und Aufgabenverteilung; regelmäßige Überprüfung der Arbeitslast und ggf. Neuzuweisung von Aufgaben; Priorisierung von Aufgaben nach ihrer Dringlichkeit und Wichtigkeit |
| Ausfall eines Teammitglieds | Klein | Verzögerungen bei der Projektumsetzung, erhöhter Arbeitsdruck für verbleibendes Teammitglied | Wissenstransfer durch Dokumentation der Arbeitsprozesse und Aufgabenverteilung sowie den Scope verkleinern. |
| Kommunikationsprobleme zwischen Teammitgliedern | Klein - Mittel | Missverständnisse, ineffiziente Zusammenarbeit | Regelmäßige Besprechungen zur Klärung von Fragen und zur Sicherstellung eines reibungslosen Informationsaustauschs |
| Unzureichende Kommunikation mit Stakeholder / Betreuer | Klein | Missverständnisse, unklare Anforderungen, fehlende Unterstützung | Regelmäßige Meetings und Statusupdates mit den Stakeholdern/Betreuern vereinbaren; Klare Kommunikationswege und -protokolle festlegen; Offene und transparente Kommunikation |
| Technische Herausforderungen | Mittel | Verzögerungen bei der Entwicklung | Stets am Ball bleiben, Literatur lesen und sich gegenseitig unterstützen neues zu lernen |
| Zeitmanagementprobleme | Mittel | Verzögerungen bei der Fertigstellung, erhöhter Zeitdruck | Realistische Zeitpläne und Meilensteine setzen; Priorisierung der Aufgaben entsprechend ihrer Bedeutung und Dringlichkeit, Scope verkleinern oder etwas in die nächste Iteration mitnehmen |
| Thema ist relatives Neuland, Fokus zu finden ist schwer | Mittel – hoch | Motivation sinkt, Arbeit stagniert | Bei Blockaden eine Pause machen und Betreuer um Rat fragen |

## A3 Umfrage - Resultate

Folgend findet man die komplette Umfrage zur Analyse der momentanen Pain Points und deren Resultate für die genauere Betrachtung.

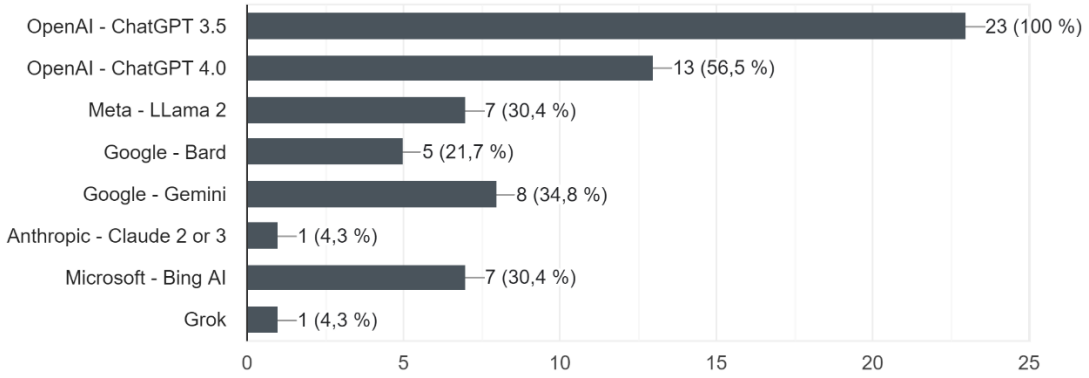
How often do you use AI-Chatbots?

14 Antworten



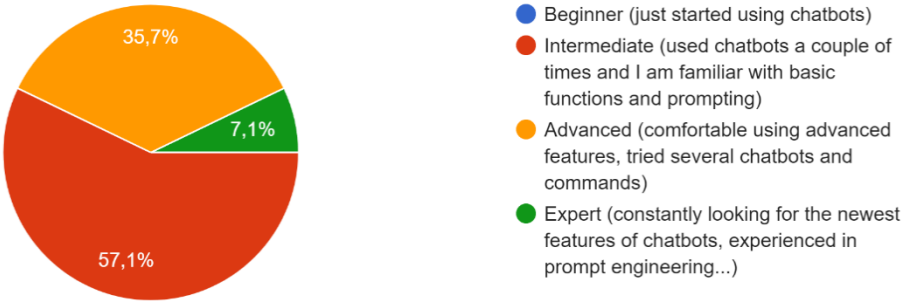

Which of these tools did you already try out? (please select all that apply)

23 Antworten




What would you say is your experience with AI-Chatbots?

14 Antworten

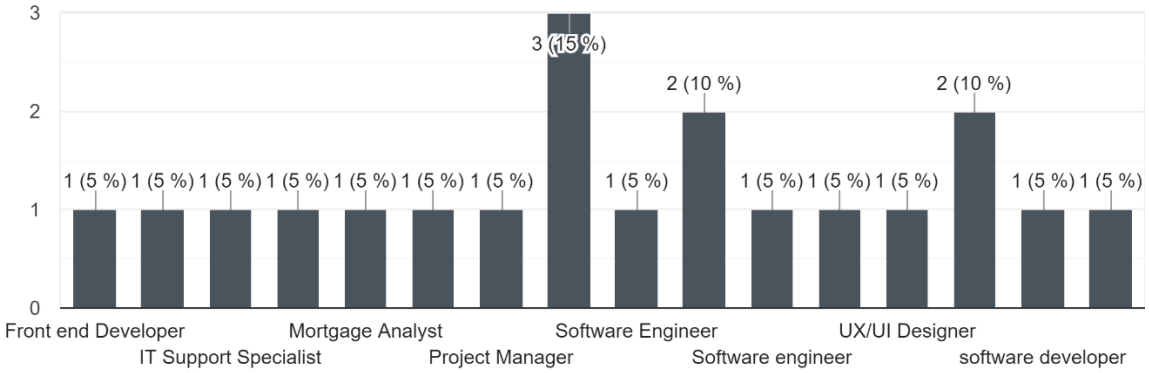


## For what industry do you work for?

14 Antworten

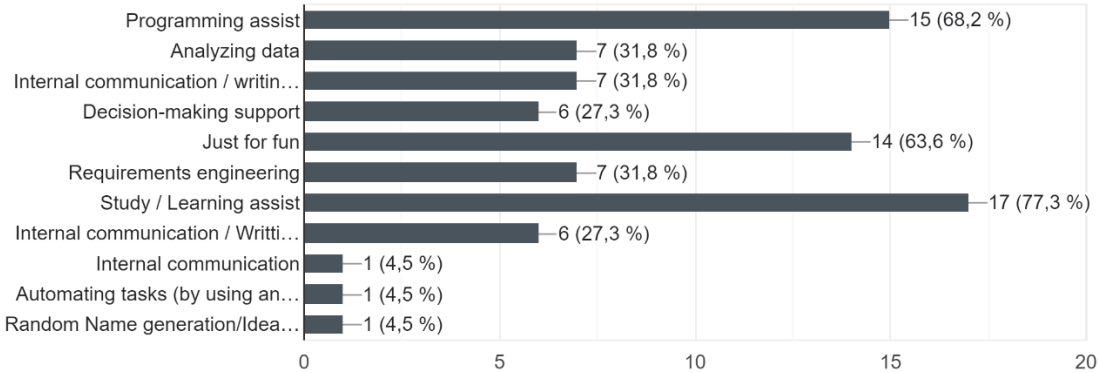

## What is your role? (e.g. "software developer", "research scientist", "student and data scientist"...)

20 Antworten



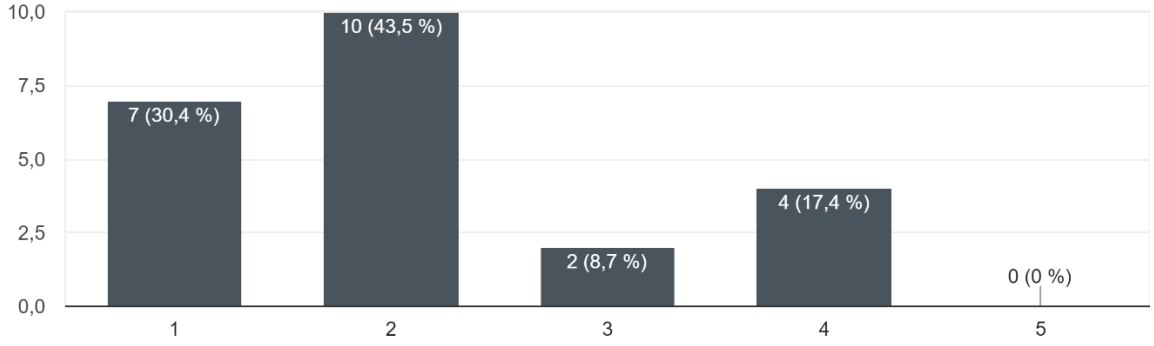

## For what purposes do you use AI-Chatbots?

22 Antworten



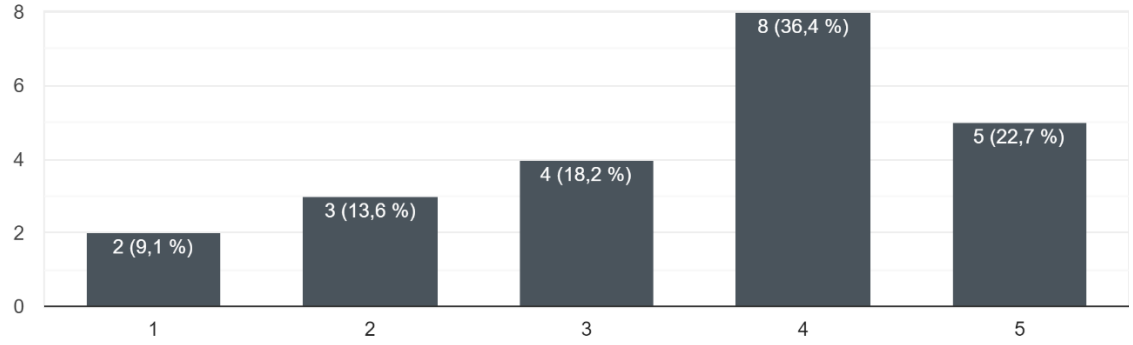
Let's talk about pain points: How often do you find yourself browsing through your past conversations?

23 Antworten



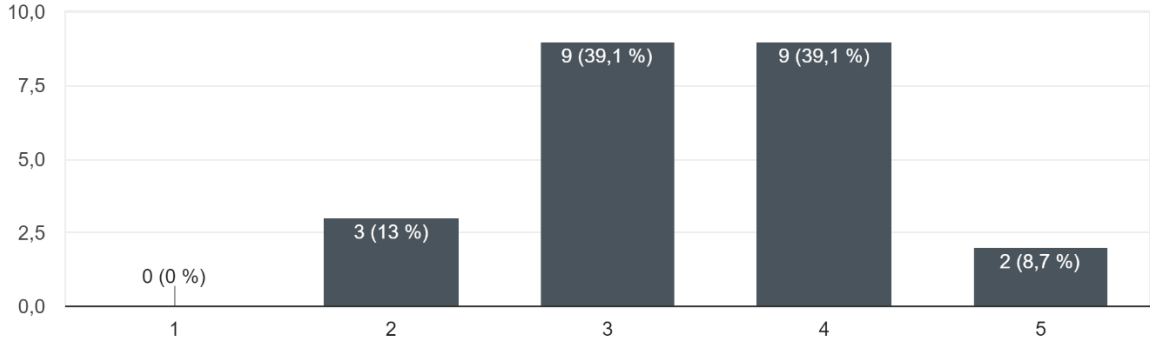

Is it hard for you to find some specific data in your past conversations with the AI or even the right conversation?

22 Antworten



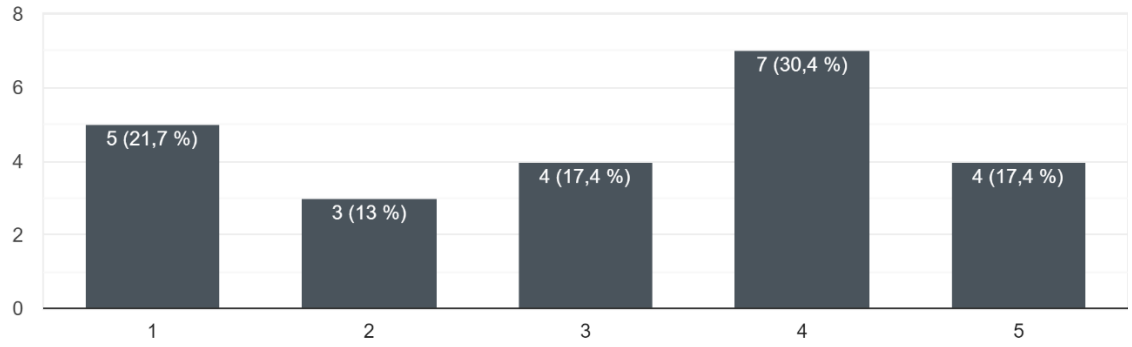
As you may know, you can not classify or order your past prompts in any way, do you find it hard to find a specific prompt you had in the past by going over large discussions you had with the AI?

23 Antworten



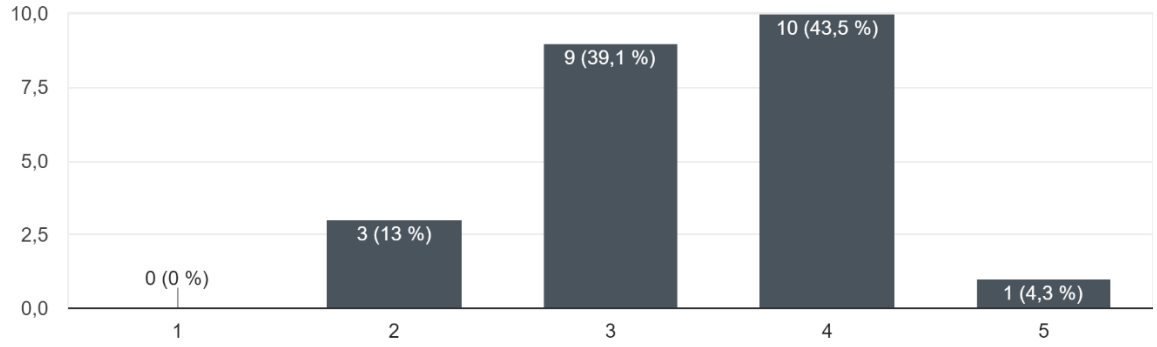

Do you feel restricted in the sense of not being able to sort / filter conversation histories / prompts by your own rules?

23 Antworten



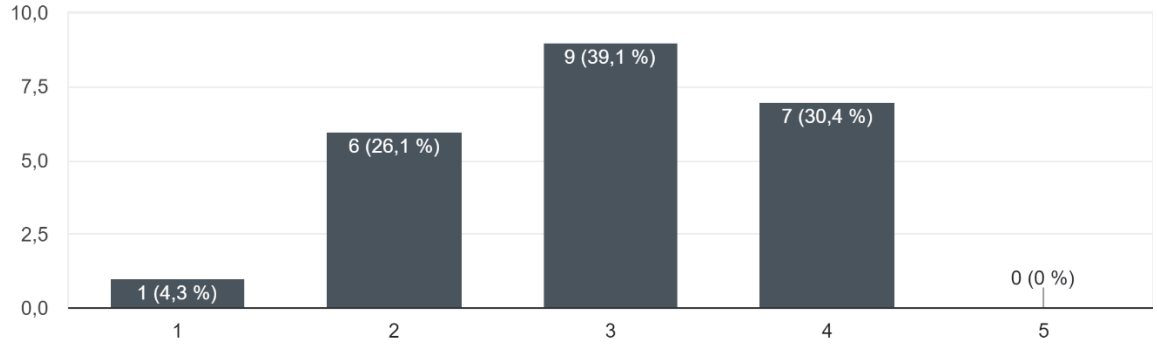
How happy are you with the User Experience (in general) of current AI-Chatbots?

23 Antworten



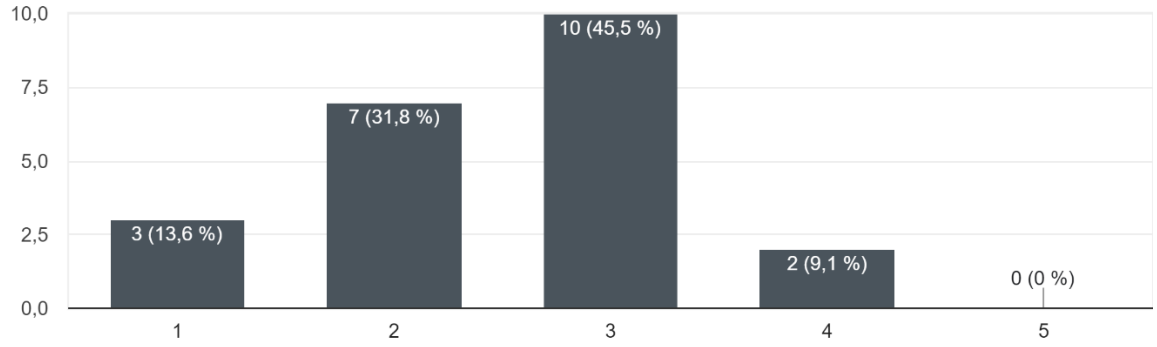

How useful is the classic "past conversation history" sorted by date for you?

23 Antworten



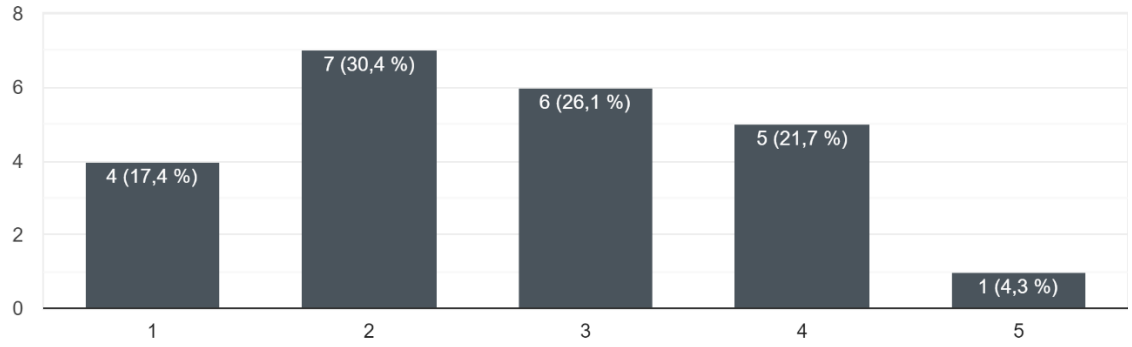

How well can you currently search for a past conversation?

22 Antworten



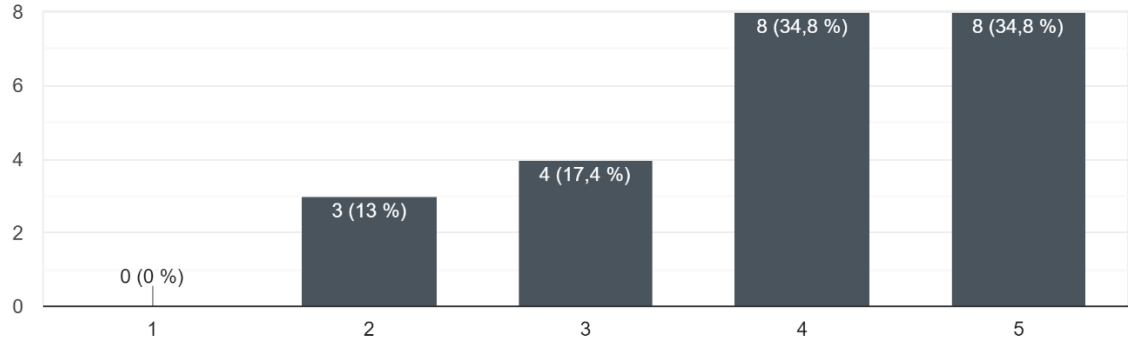
How often do you find yourself searching for some past conversation?

23 Antworten



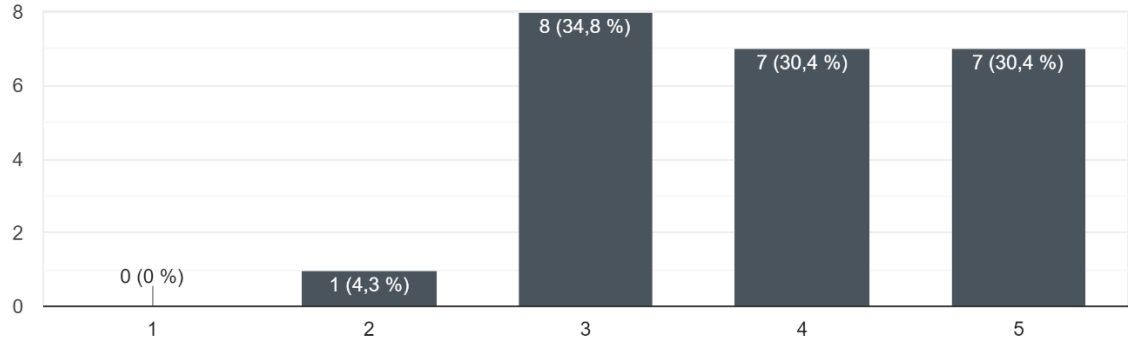

How useful do you think it would be to have a full text search over all your previous prompts instead of just a history with the AI?

23 Antworten



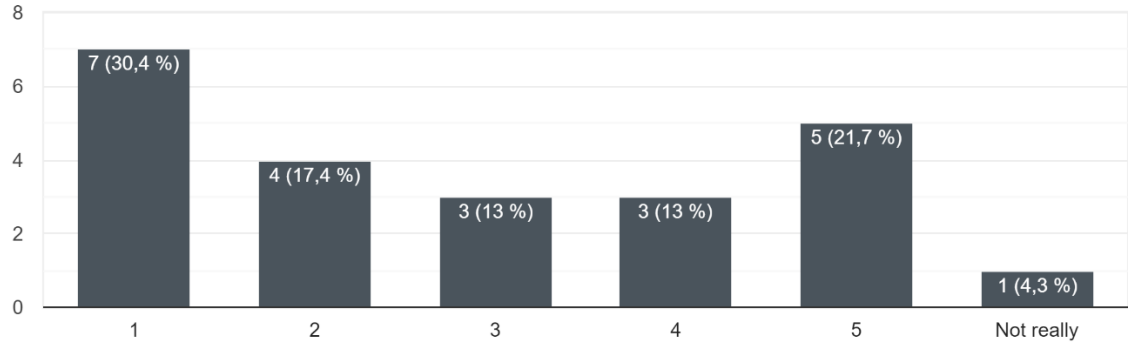
How useful do you think it would be to be able to tag or label prompts with the AI? (to help with your prompt-management and history)

23 Antworten



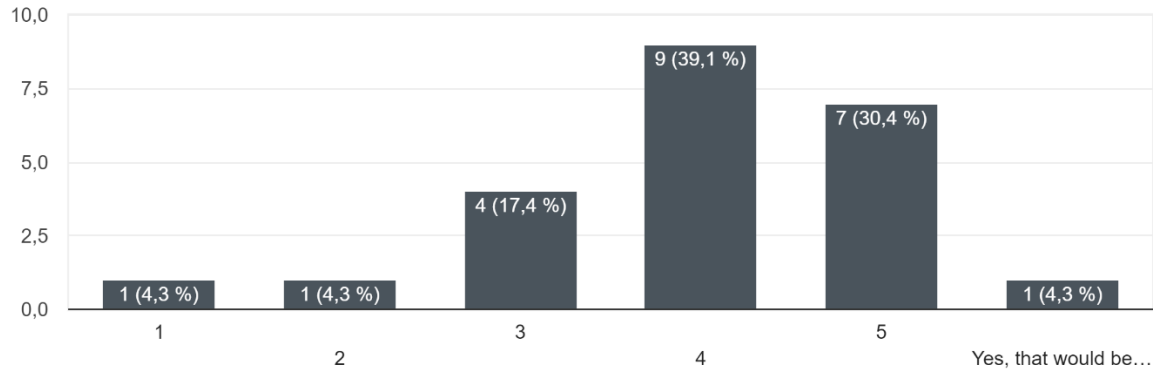

Would you like to receive a summary of the conversation you had with the AI at the end of it?

23 Antworten



Would you like to have your prompts in a more organized way instead of just a history sorted by date as it's now? (Maybe with a table, more filtering options etc..)

23 Antworten

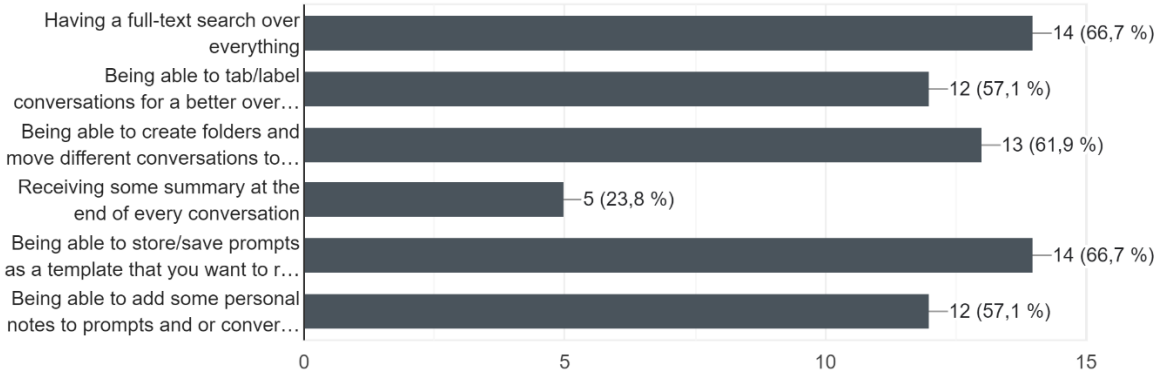



Would you like to have your past conversations in a more structured way like folders?

23 Antworten


Which of these prompt-management features do you think are the most useful?

21 Antworten



Do you have any additional comments or suggestions for improving the user experience / prompt management of AI-Chatbots?

5 Antworten

Martin

AI Chatbox should always automatically double check themself before answering.
Different chats where chat gbt role is assigned. Example Chat gbt should give answers like a teacher by default in the "teacher chat" when I am using it for studying.

Automatic tagging/labeling would be very nice and perhaps also automated grouping (for example in the way that arc allows automatically grouping browser-tabs)

Some retrospective tagging would be nice I guess. Like in ChatGPT you have the convo on the left hand side with a short summary sentence. Now imagine github labels tagging the convo as well „java“ „development“ „math“ „research“. This would help create a folder structure.

Editing in the Answer-Prompt (sometime Prompts need some min. Changes or adjustsm.);
Changing Font, Font-Size;
instead of a Summary it would be nice if it could assist me with further question for example: after answer-prompt: something like most other useres searched also for xxx xxxxx (giving hints for further investigation or explanation) and if I click this hint, then it would auto. give me a answer prompt to this;

## A4 User Testings

In diesem Abschnitt befinden sich die detaillierten Notizen und Abläufe der User Tests. Zusätzlich zu den schriftlichen Aufzeichnungen haben wir die Tests auch per Audio aufgenommen. Diese Aufnahmen finden jedoch keinen Platz in diesem Bericht.

### User Testing 1

**Ziel**

Das Ziel dieses User Testings war es herauszufinden, wie die Positionierung des Suchfeldes bzw. des Nachrichten-Editors die Benutzer dazu motivieren könnte, die vorgeschlagenen Prompts zu verwenden, anstatt eine neue Konversation zu starten. Ein weiteres Ziel war es zu verstehen, wie der User die Vorschläge interpretiert und ob er diese nutzen würde.

**Szenario**

Der User soll mit dem Prototyp versuchen eine Frage zu stellen und während er beschreibt, was er denkt und wie er die verschiedenen Felder und Informationen interpretiert.

**Ablauf zu Prototyp 2.1**

1. Der User soll versuchen seine Frage zu beantworten.
*Der User findet ohne Probleme das Input-Feld und gibt dort eine Nachricht ein, wie er es sich gewöhnt ist von anderen Chat-Bots.*

2. Der User soll sagen, was er denkt, was die Vorschläge auf der Seite bedeuten / welche Informationen er vorgeschlagen kriegt.
*Er erwartet, dass die Vorschläge ihm helfen sollen seine Frage besser zu formulieren. Zum Beispiel könnten Vorschläge kommen, die per AI generiert sind und versuchen seine Frage zu vervollständigen.*
Dem User wird mitgeteilt, dass dort Prompts/Chats/Antworten angezeigt werden, die seine Frage beantworten könnten

3. Der User soll sagen, was er erwartet, dass passiert, wenn er nun auf einen Vorschlag klickt.
*Per Klick erwartet er, dass sich ein Chat öffnet an genau der Stelle, an der die Suche eine Übereinstimmung findet.*

4. Was Passiert, wenn der User nun seine Nachricht abschickt?
*Der User würde erwarten, dass nun entweder ein Prompt gemacht wird mit einem Bezug zu der vorgeschlagenen Nachricht (er beschreibt es als „Antworten auf die vorherige Nachricht“), oder dass nur der Kontext bis dahin bekannt wäre.*

5. Der User soll eine Aussage treffen, ob er diese Vorschläge nützen würden und falls ja, in welchen Situationen.
*Eher weniger*

**Prototyp 2.2**

1. Der User soll versuchen seine Frage zu beantworten.
*Der User findet ohne Probleme das Input-Feld und gibt dort eine Nachricht ein, er würde seine Nachricht aber vielleicht anders formulieren als im ersten Beispiel, weil ihn die Positionierung eher an eine Suche erinnert.*

2. Der User soll sagen, was er denkt, dass die Vorschläge auf der Seite bedeuten / welche Informationen er vorgeschlagen kriegt.
*Der User sagt, nun ist ihm klar, dass die Vorschläge Resultate einer Suche sind, die über die bestehenden Chats erfolgt ist. Er findet diese Art der Darstellung der Vorschläge intuitiver.*

3. Der User soll eine Aussage treffen, ob er diese Vorschläge nützen würden und falls ja, in welchen Situationen.
*Nun würde der User die Vorschläge öfters nutzen, da er verständlicher und übersichtlicher findet, was diese bedeuten. Auch sagt er, dass er so gezwungen wird mit den Augen über die Vorschläge zu gehen, um den Senden Button zu klicken. Dies sei aber auch negativ, dass er nun einen längeren Weg zurücklegen muss. Bei Enter würde er erwarten, dass die Suche durchgeführt/fixiert wird und nicht, dass eine Nachricht geschickt wird.*

**Weitere Fragen und Inputs**

Kann man sehen auf Grund welcher Kriterien ein Vorschlag ausgewählt wurde?
*Interessant wäre Tags anzuzeigen, die indizieren welches „Thema" gematcht wurde* sowie *die Möglichkeit nach der Form der Antwort zu suchen wäre spannend (Text, Code, File etc.)*. *Bei den Suchvorschlägen wäre das Datum wichtig und der Name des Chats*

**Fazit**

Prototyp 2.1

- Input wird als Nachrichten-Editor interpretiert
- Die Nachricht wird als Prompt formuliert
- Die Vorschläge sind nicht klar als Suchresultate erkennbar
- Kein Zeitverlust im Vergleich zum normalen Chat-Bot

Prototyp 2.2

- Input wird als Such-Feld interpretiert
- Die Nachricht wird eher als Suchtext formuliert
- Die Resultate sind klar und werden auch als Suchresultate interpretiert
- Kleiner Pain/Zeitverlust, da der Such-Button am Ende der Seite ist
- Suchvorschläge würden öfters benutzt werden

## User Testing 2

**Ziel**

Das Ziel dieses User Testings war es, die optimale Positionierung der Suchleiste/Text-Input (oben/unten), das Layout der Vorschläge (horizontal/vertikal) und die Navigation durch die Empfehlungen und Chats zu ermitteln.

**Szenario**

Benutzer, in diesem Fall Informatiker, sollten eine Frage zu einem Projekt stellen und dabei zwei Prototypen vergleichen, um zu entscheiden, welche Ansicht sie bevorzugen und warum.

**Ablauf**

Die Benutzer testen Prototyp 3.1 und 3.2 und werden gebeten, ihre Präferenzen zu geben sowie die Gründe dafür zu erklären.

**Ergebnisse zum Startscreen**

Input wird erfolgreich als "Message/Text-Input" interpretiert.

**Ergebnisse zum «Similar Prompts / Vorschläge» Screen**

- Vorschläge werden besser erkannt, wenn das Eingabefeld oben positioniert ist.
- Eine vertikal gestreckte Ansicht der Vorschläge wird leicht bevorzugt.
- Titel wie "Vorschläge" oder "Chats" werden gewünscht.

**Ergebnisse zum «Chat-Verlauf» Screen**

- Enter sendet den Chat ganz unten.
- Der Klick auf "Continue here" sollte klarer sein oder besser beschrieben werden.

**Fazit**

- Das Eingabefeld sollte oben positioniert sein.
- Der "Senden"-Button sollte sich im oder direkt beim Eingabefeld befinden.
- Titel wie "Recommendation" wären nützlich.
- Die Navigation ist klar verständlich.
- "Continue Here" im Chat-Verlauf sollte klarer beschrieben werden.

Die Ergebnisse des User-Tests liefern wertvolle Einblicke in die Benutzerpräferenzen und -erwartungen. Insbesondere zeigt sich, dass die klare Kennzeichnung von Suchresultaten und die intuitive Positionierung von Eingabefeldern und Senden-Buttons entscheidend für die Akzeptanz und Nutzung der vorgeschlagenen Funktionen sind. Basierend auf diesem Feedback werden wir die Positionierung des Suchfeldes optimieren und klare Titel sowie eine intuitivere Navigation implementieren.

## User Testing 3

**Ziel**

Das Ziel dieses User Testings ist es, die Erkenntnisse aus den vorherigen User Testings zu verifizieren, die Darstellung der Empfehlungen «Similar Prompts» zu validieren und weiteres Feedback zu sammeln.

**Szenario**

Der Benutzer, ein Informatiker, sollte eine Frage zu einem Projekt stellen und sich dabei selbstständig durch den Prototyp klicken, während er seine Gedanken formuliert.

**Ablauf**

1. Startscreen:
    - Animation flüssig
    - Ist klar, dass es sich um einen Chat handelt
    - Enter und Icon Rechts zum Senden war verständlich
2. Vorschlag-Screen:
    - Recommendations waren auch klar erkennbar als solche
    - Nachrichten waren eindeutig erkennbar
    - Tags und Datum waren auch klar
3. Chat-Verlauf
    - Kleine Unsicherheit bei dem “Hier senden”, was genau passieren würde. Nach kurzem Überlegen war es klar
    - Nach dem Senden hat sich diese Überlegung dann bestätigt

**Fazit**

- Layout und Struktur sind klar und verständlich
- “Hier senden” könnte noch prominenter sein
- Nachrichten waren ein wenig schwer zu lesen, da es relativ viele angezeigt hat
- Fazit vom Tester: Spannendes Tool, das tatsächlich eine Erleichterung sein könnte in gewissen Situationen

## A5 Prototypen

Hier befinden sich alle unsere Prototypen welche wir teilweise per Hand in eigenen kleinen Workshops erstellt haben.

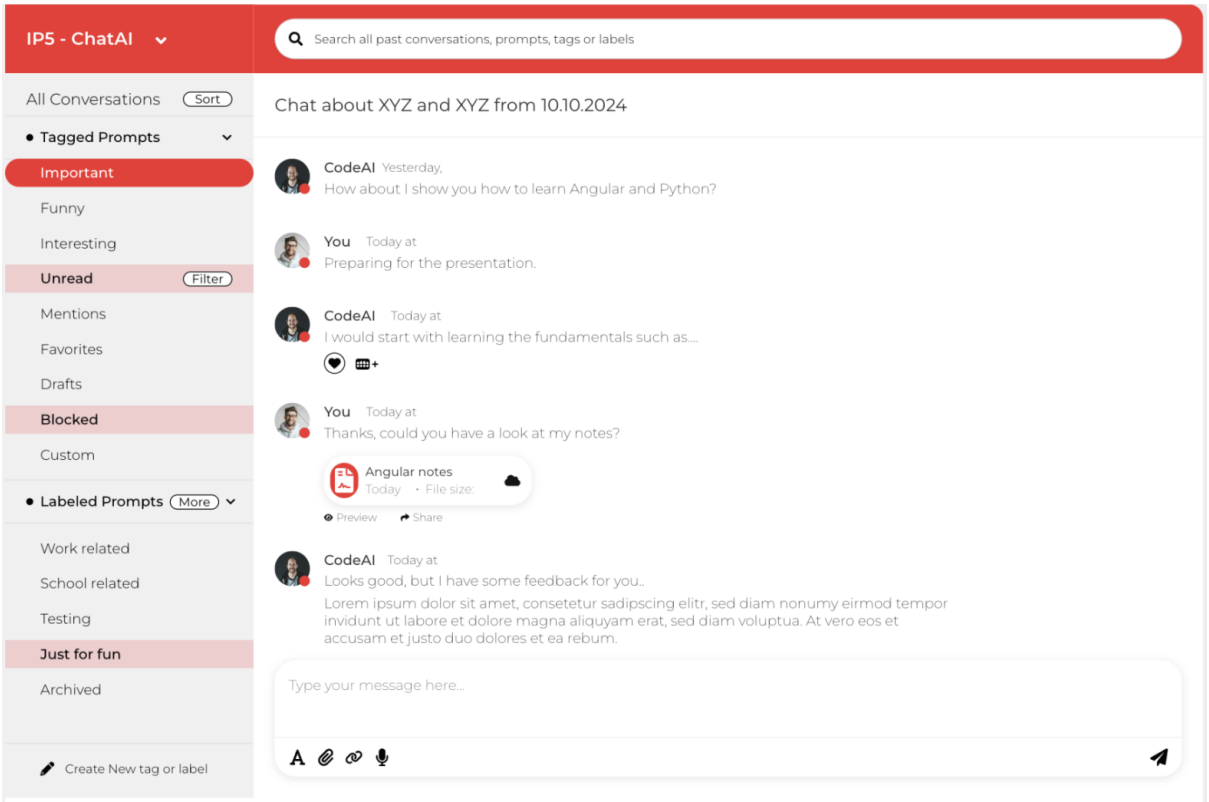
### Prototyp 0 - Inspiration

**Erste Inspirationen durch AI-generierte Prototypen**

Der Erste Prototyp haben wir nur für die Inspiration gebastelt mithilfe einer AI «UiWizard». Ziel war es unsere Ideen für die neuen Funktionalitäten zusammen mit den Erkenntnissen aus der Umfrage in einen Prototyp zu verwandeln, um daraus Inspiration und neue Ideen gewinnen zu können.

Priorisierung der Funktionalitäten für diesen Prototyp:

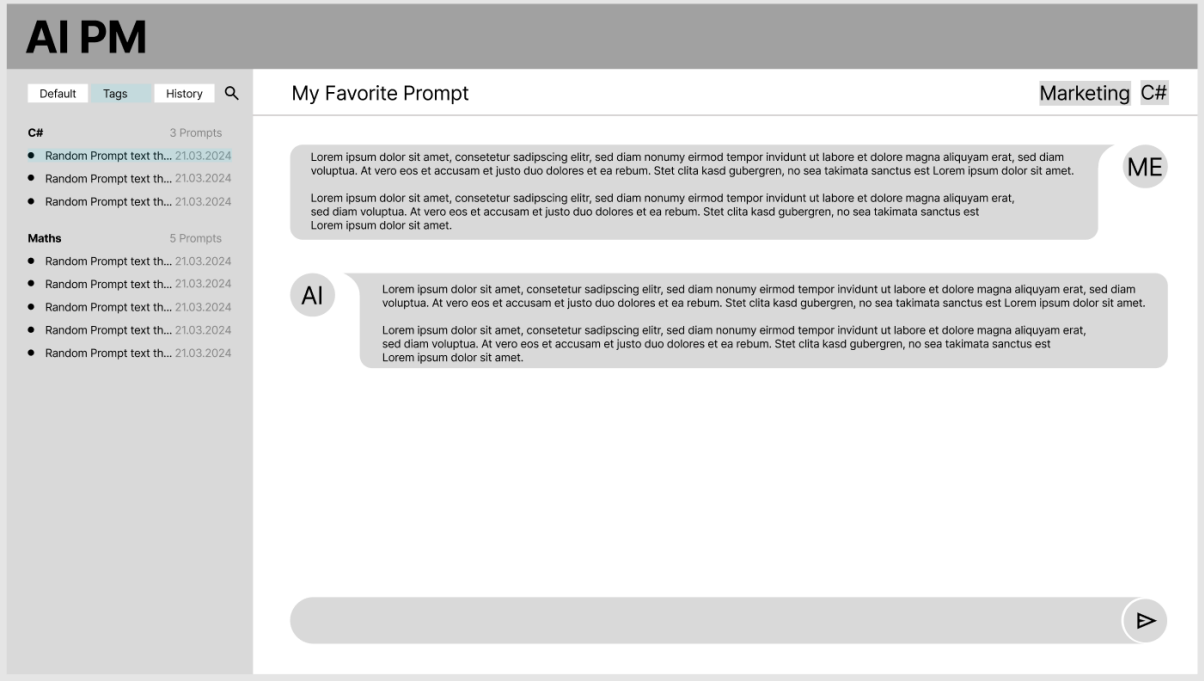
- Suchfunktion
- Tagging Funktionalität für Konversationen



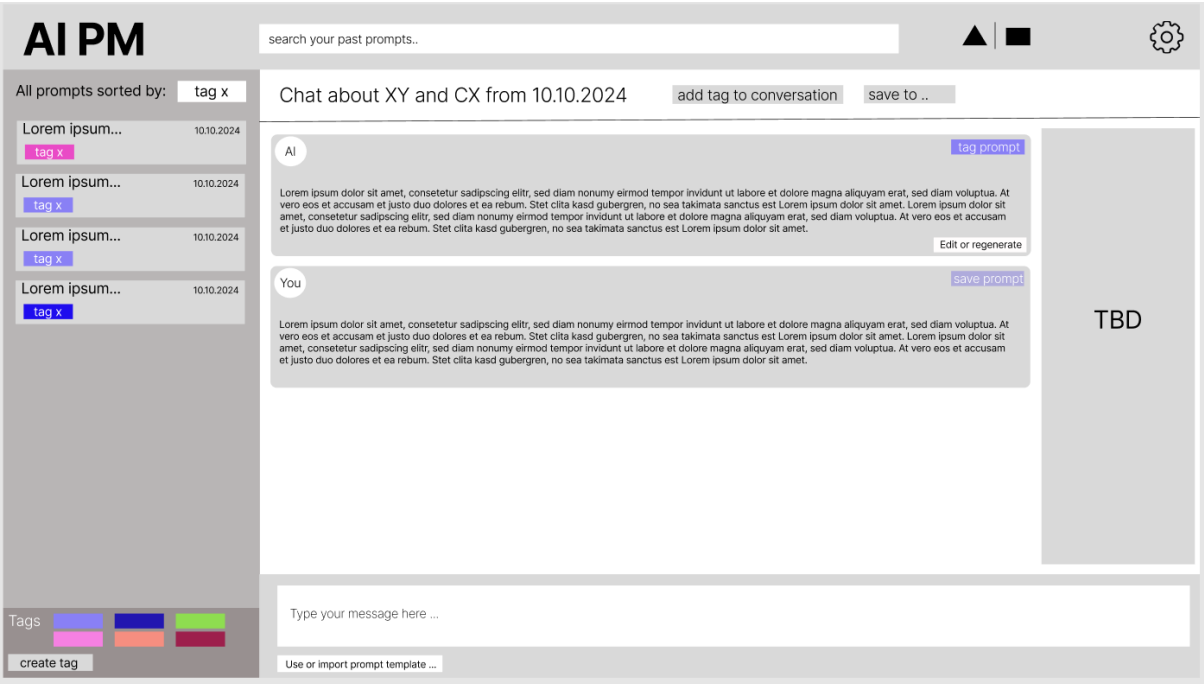
## Prototyp 1 – Ein erster Ansatz

Nachdem wir genügen Inspiration aus unserem generierten Prototyp, der Umfrage sowie weiterer Recherche sammeln konnten haben wir in Figma ein paar Prototypen erstellt. Auch hier lag unser Fokus hauptsächlich auf die Suchfunktion sowie das Tagging.

### Version 1.1




### Version 1.2



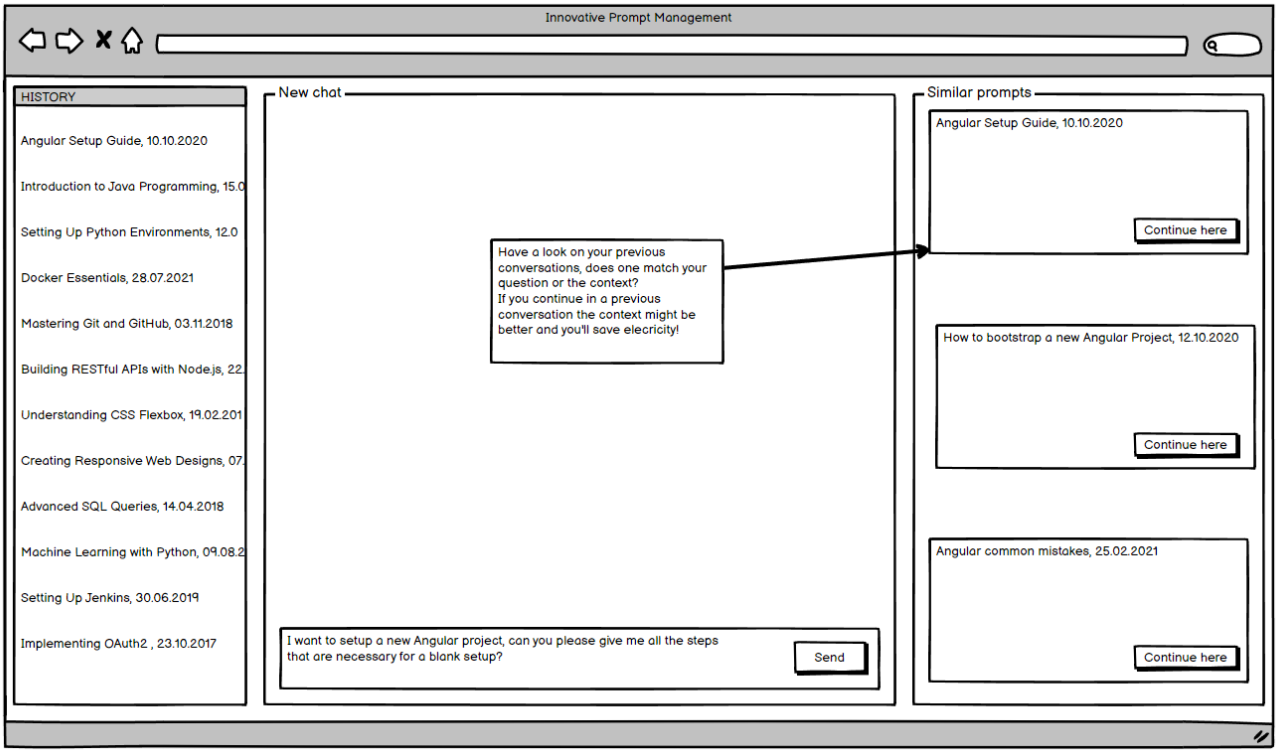
## Prototyp 2 – Erster validierter Prototyp

Nach dem Ersten Prototyp ist uns aufgefallen, dass wir uns zu fest auf spezifische UI basierte Features fokussiert haben und dabei die User-Journey etwas in Vergessenheit geraten ist.

Daraufhin haben wir mit Stift und Papier einen neuen Prototyp gebastelt, diesen in einen Low-Fi Wireframe übertragen und ihm einen User-Test unterzogen (siehe User Testing 1)

Die wichtigste Änderung war hier ein Shift von einer einfachen Volltext-Suche und einem Tagging/Labelling Feature auf einen neuen Fokus mit einer Suche im Hintergrund, welche direkt ähnliche «Similar Prompts» auf der rechten Seite anzeigen soll, um den Benutzer auf potenzielle schon gestellte Fragen hinzuweisen oder direkt in den richtigen Kontext einsteigen zu können.

**Prototyp 2 - Wireframe**




Im Linken Bereich sollte wie gewohnt ein Verlauf vergangener Konversationen stehen, in der Mitte der eigentliche Chat und auf der rechten Seite neu die von uns gefundenen ähnlichen Konversationen, welche während des Verfassens eines Prompts aktualisiert würden.

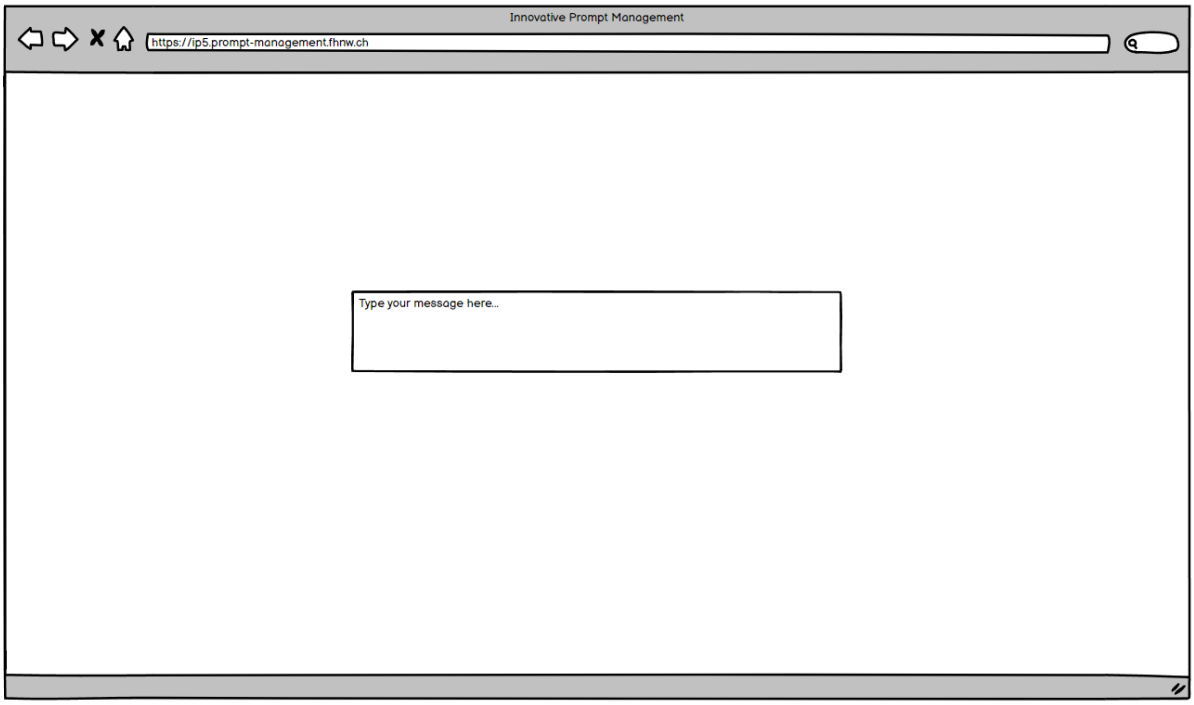
## Prototype 3 – Zweiter validierter Prototyp

Aus dem Prototyp 2 gefiel uns die Idee der «Similar Prompts» sehr, sodass wir darauf aufbauend den dritten Prototyp entworfen haben.

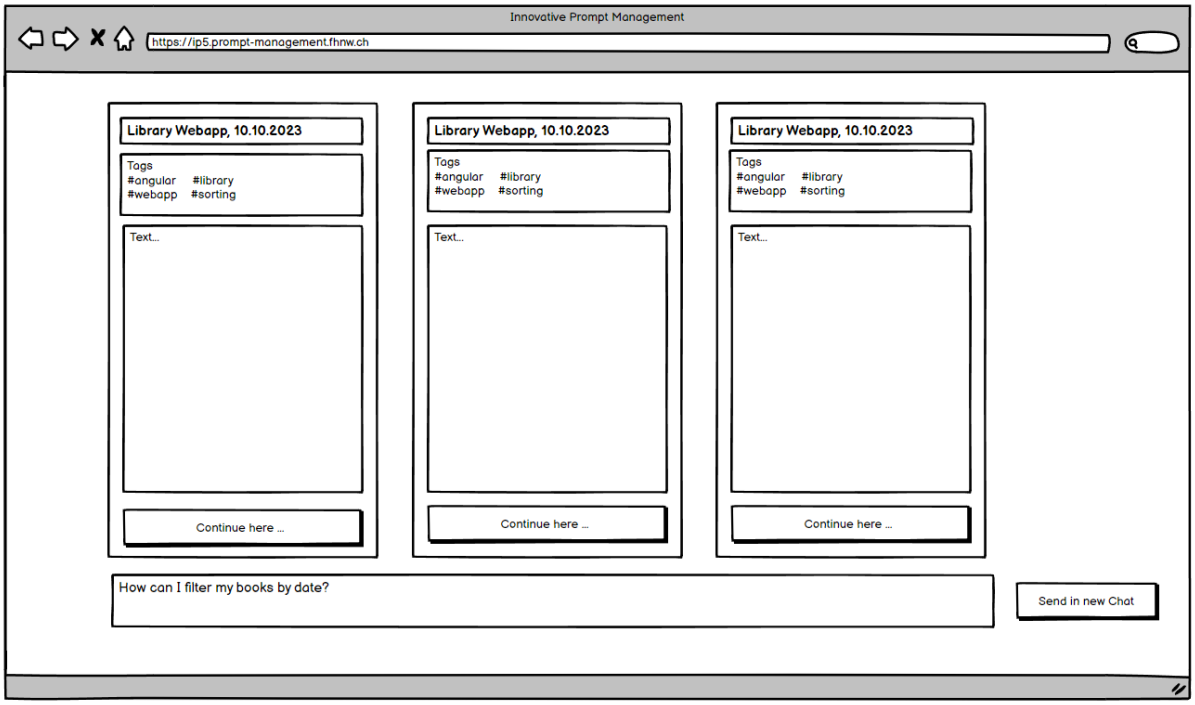
Die Idee während des Verfassens eines Prompts parallel «Ergebnisse» anzuzeigen, kam bei dem Ersten User Testing sehr gut an, sodass wir gleich einen weiteren etwas durchdachteren Prototyp auf ein Whiteboard zeichnen mussten.

Die Frage, welche noch offengeblieben ist, war, ob wir das initiale Input-Feld oben oder unten positioniert haben möchten, wo der «Senden» Button positioniert sein sollte und ob unsere User-Journey auch für einen externen Benutzer Sinn ergibt, sodass wir unseren zweiten User Test gemacht haben (siehe User Testing 2).

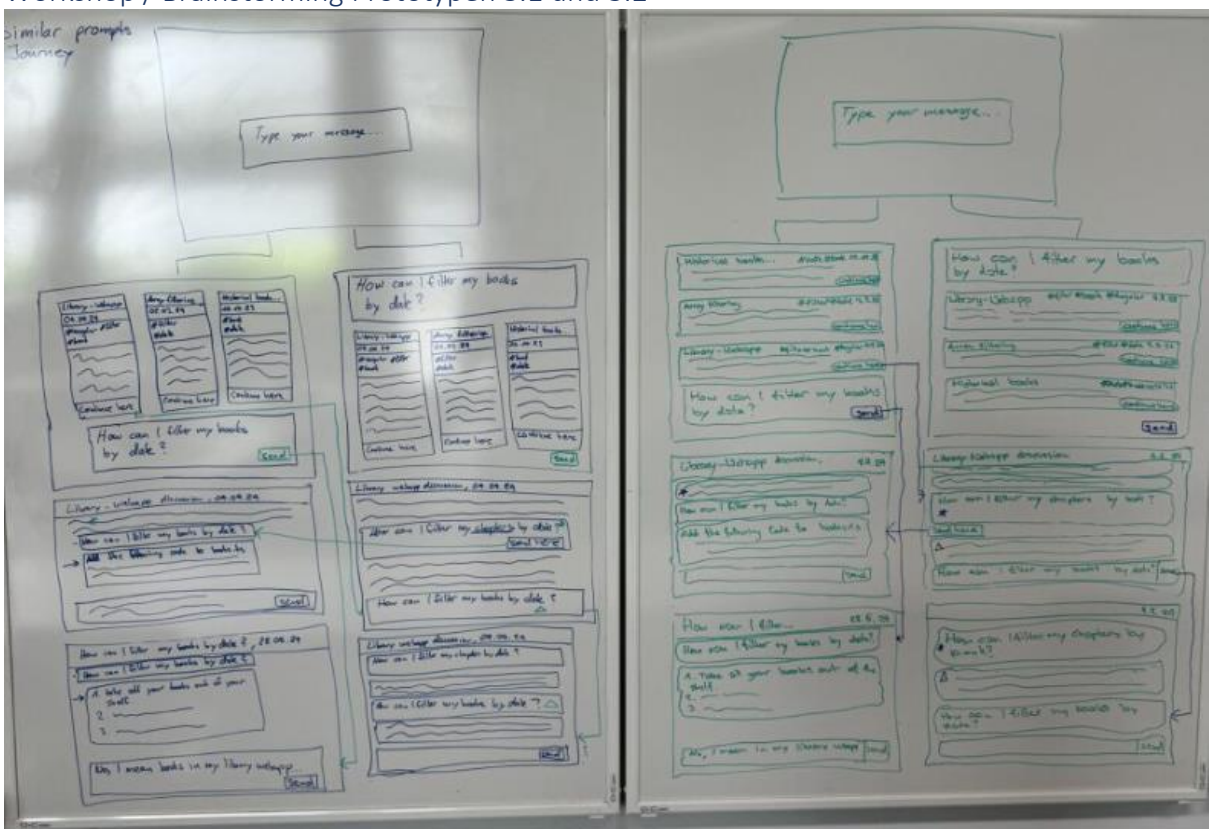
Start auf Dashboard:



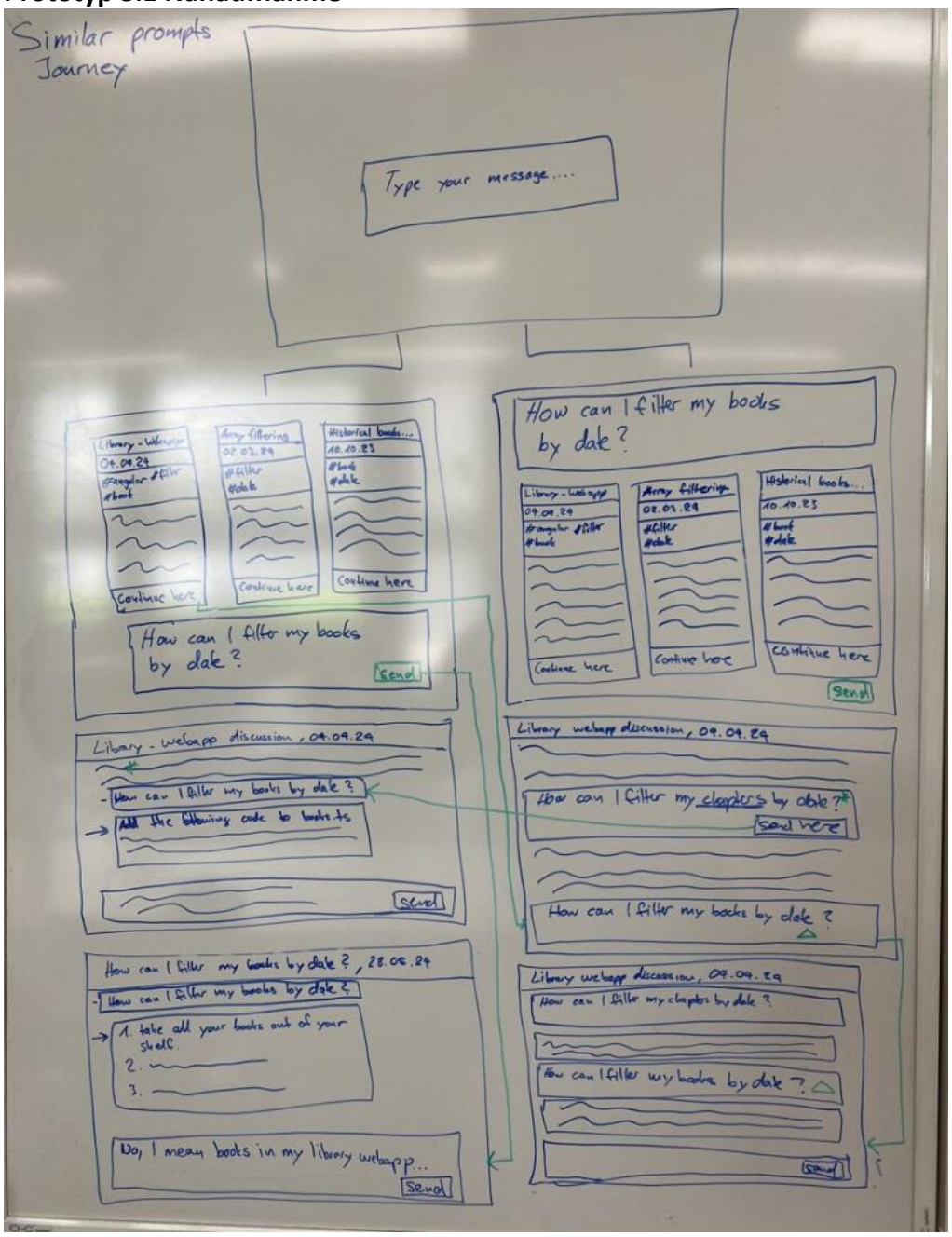
Similar Prompts View:



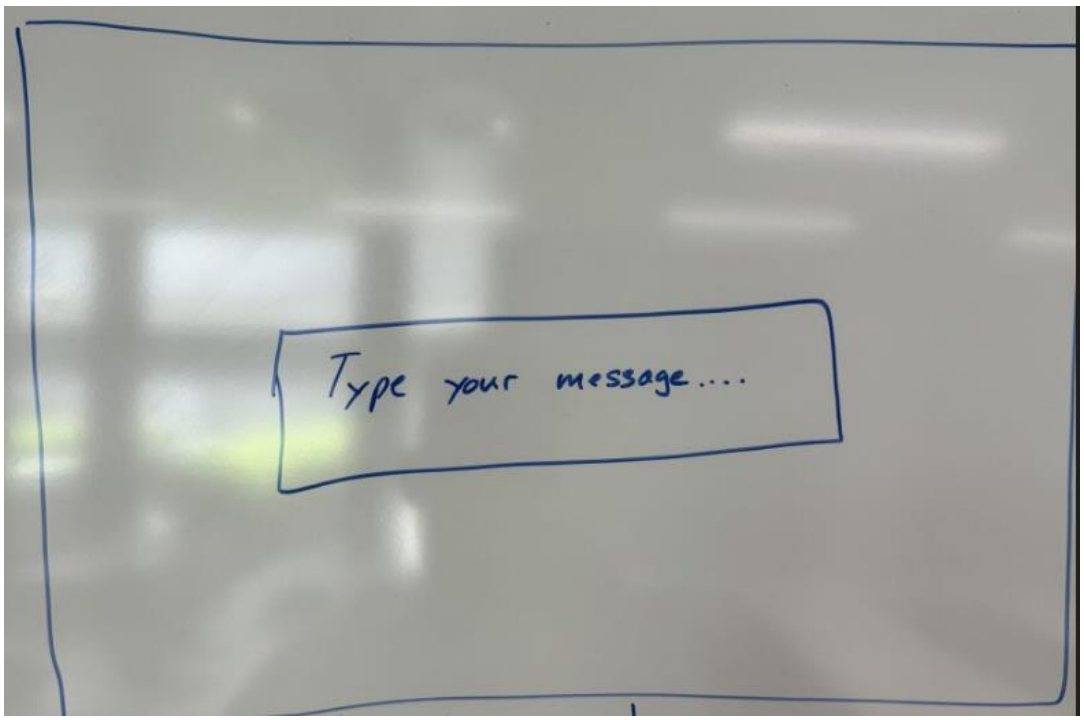
Workshop / Brainstorming Prototypen 3.1 und 3.2



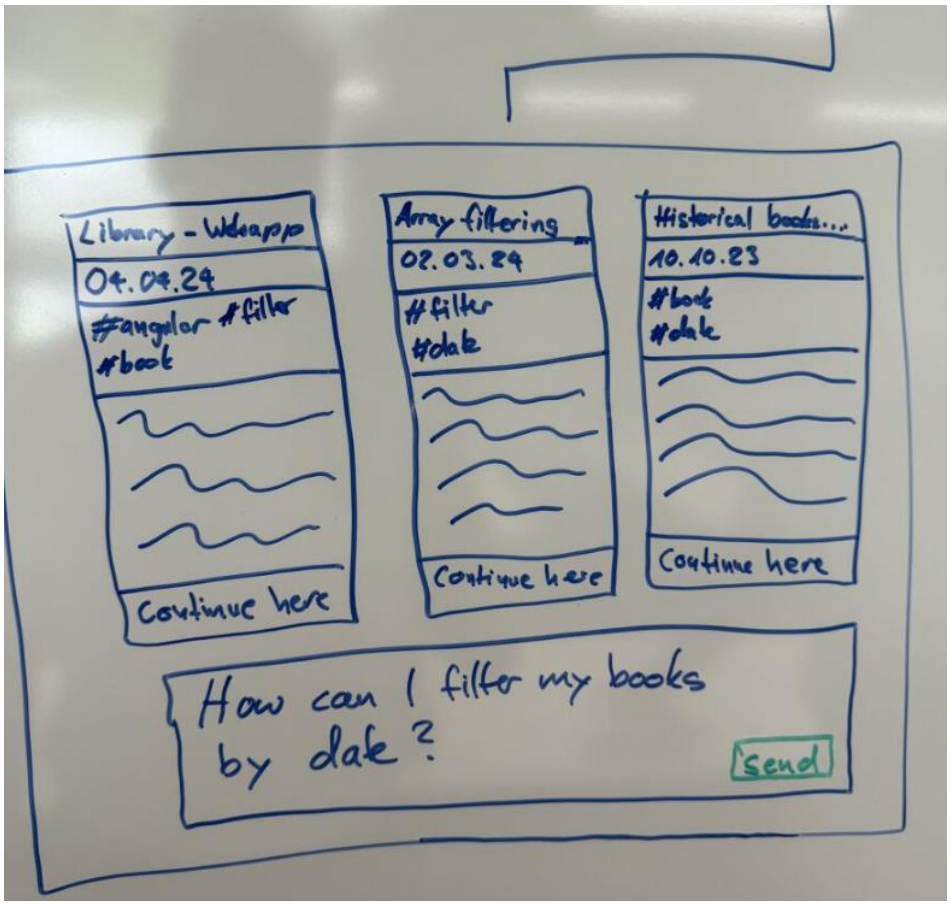
**Prototyp 3.1 Nahaufnahme**



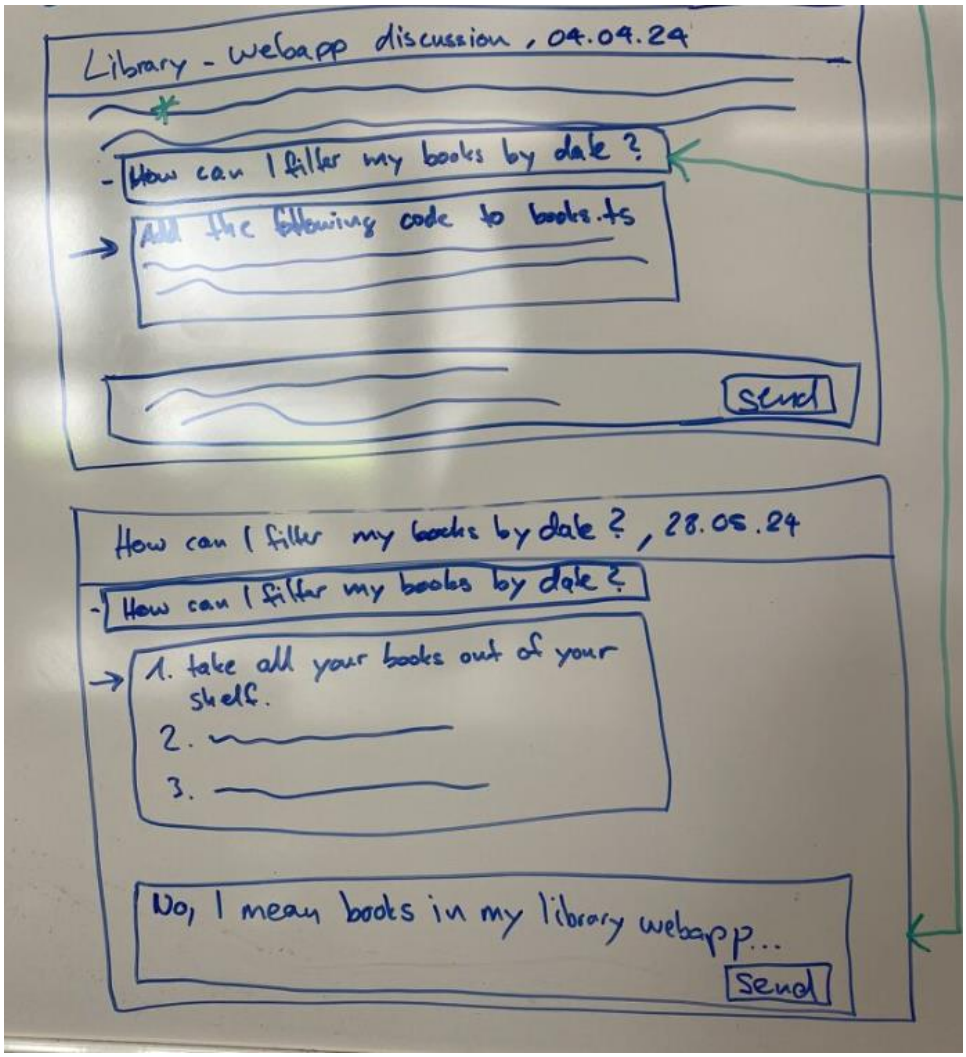
Der User startet auf einem Dashboard mit dem Input-Feld in der Mitte und verfasst seinen Prompt:




Im Hintergrund wird die Datenbank durchsucht und passende Konversationen und/oder spezifische Prompts angezeigt:



Der User kann daraufhin in eine Konversation einsteigen und direkt im richtigen Kontext seine Frage stellen, eine bereits beantwortete Frage nachschlagen oder auf eine passende gefundene Antwort/Frage antworten.



## A6 Personas

Aus den gewonnenen Erkenntnissen der Umfrage sowie den vielen Brainstormings zusammen mit unseren Betreuern haben wir eine Persona erstellt, um unseren potenziellen Benutzer stets vor Augen zu haben.

### Persona – Softwareentwickler

Softwareprogrammierer welcher an mehreren Projekten gleichzeitig arbeitet. Beispielsweise hat er ein Projekt auf der Arbeit in Angular, eines in der Schule mit Java und zu Hause arbeitet er noch an ein weiteres in der Programmiersprache Rust.
Er stellt seiner präferierten AI immer wieder Fragen rund um seine Projekte, doch die AI weiss nicht in welchem Kontext es sich gerade befinden muss, weswegen er die AI immer wieder mit etwas Zusatzinformationen füttern muss, welches mühsam und Zeitverschwendung ist.

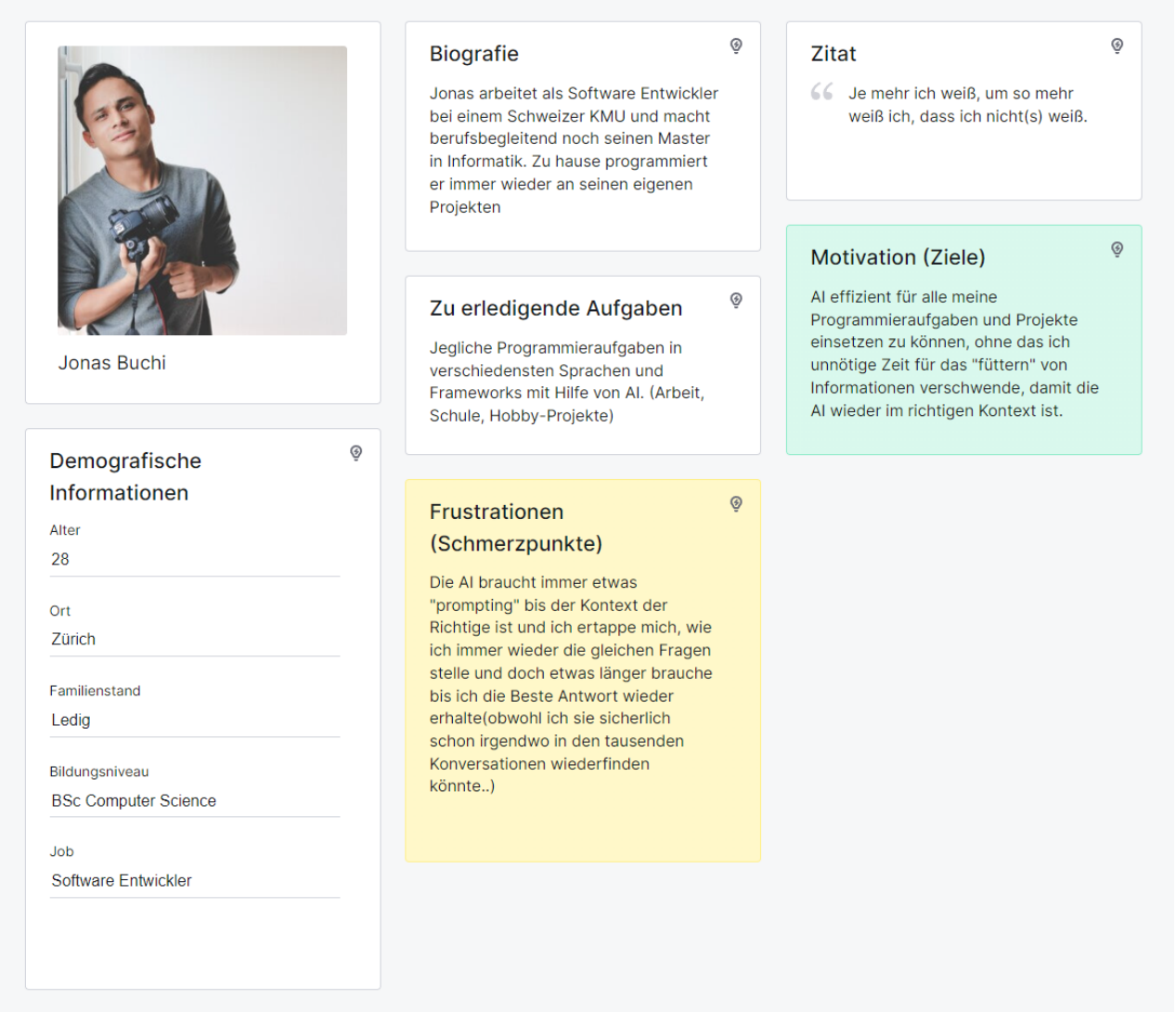



*Abbildung 18: Persona Softwareentwickler für Szenario 1*

## Persona – Content Creator

Lisa arbeitet an mehreren Content-Projekten gleichzeitig, darunter Blog-Posts über aktuelle Trends in der IT, Whitepapers für Produkteinführungen und Social-Media-Kampagnen. Sie nutzt regelmässig ihre bevorzugte KI, um Inspiration und Ideen für neue Inhalte zu finden. Allerdings fehlt der KI oft der Kontext zu ihren früheren Arbeiten, was bedeutet, dass Lisa immer wieder nach ähnlichen Themen suchen muss, um Strukturen und Formulierungen manuell wiederzuverwenden. Diese mühsame und zeitraubende Arbeit stört ihren kreativen Fluss und verlangsamt den Content-Erstellungsprozess.

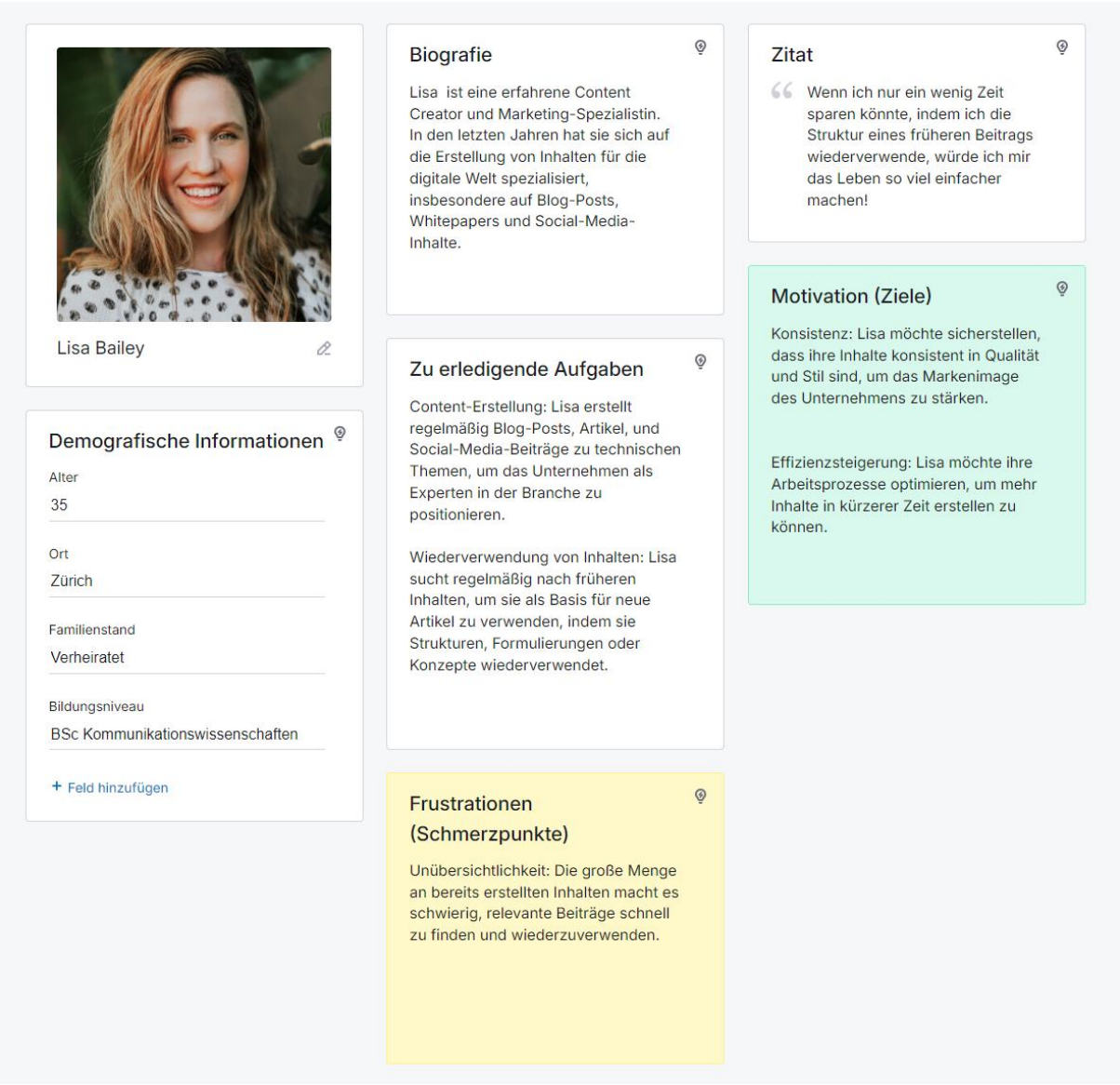



*Abbildung 19: Persona Content Creator für Szenario 2*