Gottfried Wilhelm
Leibniz Universität Hannover
Fakultät für Elektrotechnik und Informatik
Institut für Praktische Informatik
Fachgebiet Software Engineering

Erhebung von Erklärbarkeitsanforderungen für die Integration von Spracheingaben in einer Navigations-App

Elicitation of Explainability Requirements for the Integration of Voice Input in a Navigation App

Bachelorarbeit

im Studiengang Informatik

von

Lukas Nickel

Prüfer: Prof. Dr. rer. nat. Kurt Schneider
Zweitprüfer: Dr. rer. nat. Jil Ann-Christin Klünder
Betreuer: Martin Obaidi

Hannover, 23.08.2024

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbständig und ohne fremde Hilfe verfasst und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel verwendet habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 23.08.2024

Lukas Nickel

Zusammenfassung

Im Zeitalter der Digitalisierung und des zunehmenden Einsatzes von künstlicher Intelligenz (KI) gewinnt die Verständlichkeit und Nachvollziehbarkeit von Technologien zunehmend an Bedeutung. Gerade bei Sprachsteuerungen in Navigationssystemen, die eine intuitive und effiziente Benutzerinteraktion ermöglichen sollen, ist es essenziell, dass die Systeme dem Nutzer verständliche Rückmeldungen geben und ihre Entscheidungsprozesse transparent gestalten.

Die Arbeit untersucht daher die spezifischen Anforderungen an Erklärbarkeit, die bei der Implementierung von Spracheingaben in Navigationsanwendungen berücksichtigt werden müssen. Auf Basis einer Literaturrecherche wird ein Konzept zur systematischen Erhebung und Implementierung von Erklärbarkeitsanforderungen vorgestellt. Mittels Nutzerbefragungen und Experteninterviews werden relevante Faktoren für ein Fallbeispiel im Kontext von Navigationssoftware identifiziert und analysiert. Die gewonnenen Erkenntnisse dienen der Entwicklung von Leitlinien für Entwickler und Designer, die die Benutzerfreundlichkeit und Akzeptanz solcher Systeme fördern sollen. Die Ergebnisse dieser Untersuchung zeigen, dass Erklärungen in heutigen Systemen unerlässlich sind. Sie erhöhen die Akzeptanz und Benutzerfreundlichkeit der Nutzer für diese komplexen Systeme. Ebenso tragen sie dazu bei, die Interaktion zwischen Mensch und Maschine zu verbessern.

Abstract

Elicitation of Explainability Requirements for the Integration of Voice Input in a Navigation App

In the age of digitalization and the increasing use of artificial intelligence (AI), the comprehensibility and traceability of technologies are becoming increasingly important. This is especially true for voice-controlled navigation systems, which aim to facilitate intuitive and efficient user interaction. It is essential that these systems provide users with understandable feedback and ensure transparency in their decision-making processes.

This thesis examines the specific requirements for explainability that must be considered when implementing voice inputs in navigation applications. Based on a literature review, a systematic concept for gathering and implementing explainability requirements is presented. Through user surveys and expert interviews, relevant factors for a case study in the context of navigation software are identified and analyzed. The insights gained are used to develop guidelines for developers and designers to enhance the user-friendliness and acceptance of such systems. The findings of this study indicate that explanations in modern systems are essential. They increase user acceptance and usability for these complex systems. Moreover, they contribute to improving the interaction between humans and machines.

Inhaltsverzeichnis

Kapitel 1

Einleitung

In der heutigen digitalen Welt spielt die Verarbeitung natürlicher Sprache (*Natural Language Processing*, NLP) eine immer wichtigere Rolle. Ein zentraler Aspekt dieser Technologie ist die Erklärbarkeit. Erklärbarkeit bezieht sich auf die Fähigkeit eines Systems, seine Entscheidungen und Prozesse auf eine für Menschen verständliche Weise darzulegen [8]. Besonders wichtig ist dies bei Spracheingaben, da die Nutzer hier oft ein intuitives Verständnis der Funktionsweise eines Systems erwarten und zugleich nicht mit den zugrunde liegenden Algorithmen vertraut sind [30]. Dies führt schnell zu komplexen und undurchsichtigen Systemen. Daher ist es essentiell, dass diese Diskrepanz strukturiert abgebaut und verbessert werden kann [12].

Die Erhebung von Erklärbarkeitsanforderungen für Spracheingaben mittels NLP umfasst verschiedene Dimensionen. Dazu gehören technische Aspekte wie die Transparenz der Algorithmen, aber auch benutzerzentrierte Überlegungen wie Verständlichkeit und Nachvollziehbarkeit der Systemantworten. Bezieht man nun auch noch eine künstliche Intelligenz (KI) in diesen Prozess mit ein, so vervielfältigt sich die Komplexität eines solchen Systems [30].

1.1 Problemstellung

Die zunehmende Integration von Spracheingaben in Navigations-Apps ermöglicht neue innovative Wege wie der Nutzer mit dieser interagieren kann. Allerdings entstehen auch neue Herausforderungen bei der Implementierung eines solchen Systems. Diese Herausforderungen lassen sich in folgende zentrale Aspekte unterteilen: *Transparenz*, *Benutzerfreundlichkeit* und *Datenschutz* [6].

Diese Arbeit legt den Fokus auf die Transparenz und Benutzerfreundlichkeit von Erklärungen in einer solchen Navigations-App. Dafür wird ein Konzept zum Aufstellen von Erklärbarkeitsanforderungen vorgestellt und dieses anschließend mittels einer prototypischen Umsetzung an einem

konkreten Beispiel aus der Wirtschaft angewandt.

1.2 Lösungsansatz

In dieser Arbeit wird ein methodischer Ansatz entwickelt, um die Anforderungserhebung mit besonderem Fokus auf die Erklärbarkeit von Systemen systematisch durchzuführen. Der Lösungsansatz zielt darauf ab, Transparenz und Benutzerfreundlichkeit in den Vordergrund zu stellen, um die Akzeptanz und das Verständnis der Nutzer für komplexe Technologien zu fördern.

Zunächst wird ein Konzept für die Anforderungserhebung erarbeitet, das speziell auf die Identifikation von Erklärbarkeitsmerkmalen abzielt. Dieses Konzept basiert auf bewährten Methoden des Requirements Engineering, erweitert diese jedoch um einige spezielle Techniken, die es ermöglichen, die spezifischen Bedürfnisse und Erwartungen der Nutzer im Hinblick auf Erklärbarkeit zu erfassen. Dies umfasst beispielsweise die Identifikation von individuellem Erklärungsbedarf, in denen Erklärungen besonders wichtig sind, sowie die Erhebung von Anforderungen, die sicherstellen, dass die erklärenden Elemente sowohl verständlich als auch zugänglich sind.

Nach der Anforderungserhebung folgt die Implementierung der identifizierten Erklärbarkeitsanforderungen. Anhand eines konkreten Beispiels im Kontext von Navigationssoftware wird der entwickelte Ansatz veranschaulicht. Anhand dieses Beispiels werden die erarbeiteten Anforderungen in einem Prototypen umgesetzt. Der Schwerpunkt liegt auf der transparenten Gestaltung von Entscheidungsprozessen bei der KI-gestützten Adresserkennung für die Planung von Tourstopps entlang einer Navigationsroute. Zudem wird eine benutzerfreundliche Oberfläche geschaffen, die es den Nutzern ermöglicht, die zugrunde liegenden Mechanismen einfach nachzuvollziehen.

1.3 Ergebnisse der Arbeit

Die durchgeführte Studie zur Erhebung von Erklärbarkeitsanforderungen (siehe 5.2) zeigt, dass die Anforderungen an ein sprachgesteuertes Navigationssystem äußerst komplex und vielschichtig sind. Diese Komplexität ergibt sich aus der Notwendigkeit, unterschiedliche Nutzerbedürfnisse, technische Einschränkungen und die Balance zwischen intuitiver Bedienung und detaillierter Erklärung in Einklang zu bringen. Ein solcher Ansatz erfordert nicht nur technisches Know-how, sondern auch ein tiefes Verständnis der menschlichen Interaktion mit digitalen Systemen.

In dieser Arbeit wird die Notwendigkeit für ein strukturiertes Vorgehen mit einem wohldefinierten Konzept erkannt und umgesetzt (siehe 4). Das vorgestellte Konzept ermöglicht es Entwicklern, das System kontinuierlich zu verbessern und somit die Akzeptanz für das System bei den Nutzern zu erhöhen.

1.4 Struktur der Arbeit

Diese Arbeit beschreibt zunächst in Kapitel 2 die Grundlagen von Erklärbarkeit und Requirements Engineering, die der Ausarbeitung eins strukturierten Konzepts zur Erhebung und Implementierung von Erklärbarkeitsanforderungen dienen. Außerdem wird in 3 auf vorausgegangene Arbeiten eingegangen, auf deren Erkenntnissen diese Arbeit aufbaut. In Kapitel 4 wird das besagte Konzept zum systematischen Vorgehen für die Erhebung und Implementierung von Erklärbarkeitsanforderungen entwickelt. Dieses Konzept wird in Kapitel 5 umgesetzt, in welchem die Anforderungen für den konkreten Sachverhalt mithilfe eines Interviews und einer Umfrage erhoben werden. Anschließend wird auf Basis der erkannten Anforderungen ein Prototyp in Kapitel 6 vorgestellt, welcher diese implementiert. In Kapitel 7 werden die Ergebnisse dieser Arbeit zusammengefasst, Einschränkungen in der Validität identifiziert und ein Ausblick auf mögliche Weiterführungen dieser Arbeit gegeben.

Kapitel 2

Grundlagen

In diesem Kapitel werden die fundamentalen Konzepte des Requirements Engineerings (RE) sowie der Erklärbarkeit umfassend analysiert, da diese essenziell für die Entwicklung eines Konzepts zur systematischen Erhebung von Erklärbarkeitsanforderungen sind. Besonderes Augenmerk wird auf die spezifischen Aspekte des RE gelegt, die im Kontext der Erklärbarkeit eine besondere Relevanz besitzen.

2.1 Requirements Engineering

Das RE stellt einen essenziellen Bestandteil des Softwareentwicklungsprozesses dar und zeichnet sich durch einen systematischen und disziplinierten Ansatz aus, bei dem eine formale Spezifikation zur Implementierung erstellt wird [4]. Dabei ist es wichtig zu verstehen, was Anforderungen (*Requirements*) eigentlich sind.

Das IEEE definiert **Anforderungen** folgendermaßen:

> „Eine **Bedingung oder Fähigkeit**, die von einem Benutzer (Person oder System) zur **Lösung eines Problems** oder zur Erreichung eines Ziels benötigt wird. [...] [10]“

Demnach dienen Anforderungen der Lösungsfindung für ein zuvor definiertes Problem. Das genaue Verständnis und die präzise Dokumentation dieser Anforderungen sind für den Erfolg eines Projekts von entscheidender Bedeutung, da sie direkt mit der Lösung der festgelegten Problemstellungen verknüpft sind [24]. Anforderungen spezifizieren daher nach der bisherigen Definition, *was* ein System leisten soll, jedoch nicht *wie* dies zu erfolgen hat [23]. In 2.1.3 wird erläutert, welche Arten von Anforderungen auch das *wie* eines Systemverhaltens definieren können.

2.1.1 Stakeholder

Die Identifikation der verschiedenen Stakeholder ist dabei von zentraler Bedeutung. Stakeholder umfassen alle, die das Projekt beeinflussen, insbesondere die späteren Nutzer und die administrativen Verantwortlichen [18, 23, 24]. Es umfasst also alle Personen und Personengruppen, die ein potentielles Interesse an einem zukünftigen System haben. Mit diesem Interesse verbunden stellen sie in der Regel auch Anforderungen an dieses System [23]. Dabei kann eine Person auch mehrere Interessen vertreten und die Rolle verschiedener Stakeholder einnehmen. So trägt beispielsweise der Projektleiter die Verantwortung, das Projekt erfolgreich zu leiten. Er ist für die Durchführbarkeit und Umsetzung von Anforderungen verantwortlich. Aber er agiert auch als Teammitglied, welches das Projektteam unterstützt und das Bindeglied zwischen den Entwicklern und dem restlichen Vertrieb darstellt. Laut Balzert seien 77% der nachträglichen Änderungen von Anforderungen auf Missverständnisse in der Kommunikation der Stakeholder zurückzuführen [2].

Da die Stakeholder gänzlich unterschiedliche Personengruppen angehören können, ist eine Kommunikation nicht immer einfach. Jargonausdrücke und Schlüsselwörter mit unterschiedlicher Semantik können die Verständigung beeinträchtigen und die Anforderungserhebung erschweren [23]. Es gilt für den Requirements Engineer, diese Kommunikations- und Interessenskonflikte zu identifizieren und zwischen den Stakeholdern zu vermitteln, um spezifische Anforderungen erheben und Kommunikationsprobleme verhindern zu können[18].

2.1.2 Vorgehensmodelle

Es gibt verschiedene Ansätze zur Ausarbeitung von Anforderungen, die sich in schwergewichtige und leichtgewichtige Vorgehensweisen unterteilen lassen.

Schwergewichtige Modelle wie das Wasserfallmodell [25] oder das V-Modell [13] zeichnen sich dadurch aus, dass alle Anforderungen im Vorfeld ermittelt werden. RE wird für das gesamte System im Voraus durchgeführt und vor Beginn der ersten Entwicklungsphase abgeschlossen [24]. Das Wasserfallmodell wird in Abbildung 2.1 dargestellt. Hier wird ersichtlich, dass Anforderungen ganz zu Beginn eines Projekts erhoben werden. Nach dem Paradigma dieses Modells kann nur zu dem zuletzt durchgeführten Schritt zurückgekehrt werden, eine Anforderungserhebung im fortlaufenden Projekt kann also nicht ohne erhebliche Umstände durchgeführt werden [25, 18]. Das V-Modell ist kein eigenständiges Vorgehensmodell im eigentlich Sinn, sondern erweitert das Wasserfallmodell um zusätzliche Strukturen, um das Vorgehen mit weiteren Tests für die unterschiedlichen Ebenen abzusichern und Rücksprünge in die vorherige Ebene zu vermeiden [18, 23].

Leichtgewichtige Methoden wie eXtreme Programming [3] ermitteln die

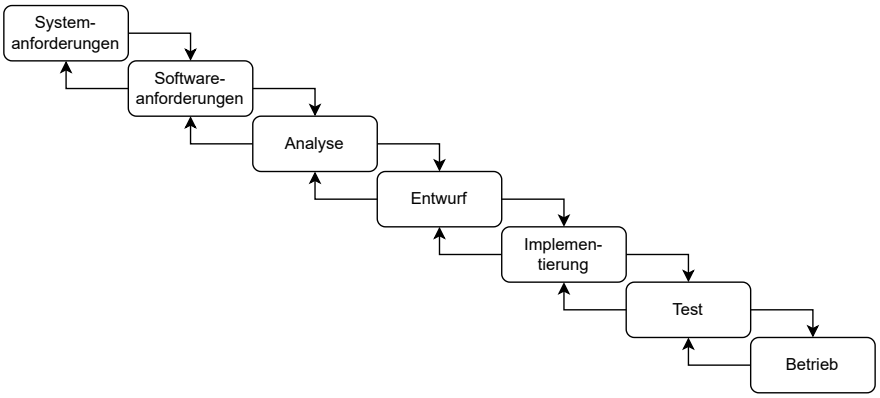


Abbildung 2.1: Das Wasserfallmodell nach Winston W. Royce [25]

benötigten Anforderungen erst dann, wenn sie konkret implementiert werden sollen. Hier wird Requirements Engineering als ein phasenübergreifender Prozess in die Entwicklung integriert, sodass dynamisch auf neue oder angepasste Anforderungen reagiert werden kann [24, 18]. Hier liegt ein großer Vorteil darin, auf Änderungen der Wünsche und Vorstellungen der Stakeholder besser reagieren zu können als bei den schwergewichtigen Modellen. Denn ein Entwicklungsprozess kann viel Zeit in Anspruch nehmen und nicht immer entsprechen die Wünsche zu Beginn des Projekts dem Soll-Zustand [18]. Indem nur Anforderungen erhoben werden, welche akut von den Stakeholdern gewünscht sind, ist eine engere Zusammenarbeit der Entwickler und des Kunden zu erreichen und es kann schneller auf Änderungswünsche und äußere Umstände reagiert werden [18, 3].

2.1.3 Arten von Requirements

Anforderungen lassen sich zudem in funktionale und nichtfunktionale Anforderungen (*functional requirements*, FRs, und *nonfunctional requirements*, NFRs) unterteilen. Da Erklärbarkeit den nichtfunktionalen Voraussetzungen zugeordnet ist [9], liegt der Fokus dieser Arbeit auf den NFRs.

FRs beziehen sich auf die erwarteten Ergebnisse eines Systems, welche auf Funktionalitäten eben dieses zurückzuführen sind [24]. Sie beantworten die Frage, *was* ein System tut. Dabei werden FRs häufig in drei Bereiche unterteilt [23]:

1. **Eingaben:** Daten, Ereignisse
2. **Funktionen:** Umformung von Daten, Verhaltensweisen
3. **Ausgaben:** Daten, Fehlermeldungen, Reaktionen des Systems

Nichtfunktionale Anforderungen definieren die gewünschten Qualitäten des Systems und beeinflussen häufig die funktionalen Anforderungen. NFRs sollten stets getrennt von FRs definiert werden, da sie oft in direkter Abhängigkeit von diesen stehen [24]. Sie beantworten die Frage, *wie* ein System seine Aufgaben erfüllen soll. Beispiele für NFRs sind Qualitätsanforderungen an Bedienbarkeit (*Usability*), Wartbarkeit (*Maintainability*) oder auch Eklärbarkeit (*Explainability*, siehe 2.2), [23]. NFRs haben einen großen Einfluss auf die Softwarearchitektur. Zudem können sie untereinander im Konflikt stehen [2]. Zum Beispiel stehen Sicherheitsanforderungen (*Security Requirements*) oft im Konflikt mit der Bedienbarkeit von Software. Qualitätsanforderungen legen fest, welche der vielen Qualitätsmerkmale als relevant eingestuft werden sollten und auch in welcher Qualitätsstufe sie erreicht werden sollten [2].

2.2 Erklärbarkeit

Erklärbarkeit (*Explainability*) ist ein zentrales Thema in der modernen Datenverarbeitung und der KI. Sie beschreibt die Fähigkeit eines Systems, seine Entscheidungen und Handlungen auf eine für Menschen nachvollziehbare Weise zu erläutern [12]. In Zeiten, in denen Algorithmen immer komplexer und undurchsichtiger werden, gewinnt Erklärbarkeit zunehmend an Bedeutung [9, 12]. Sie ist nicht nur für das Vertrauen der Nutzer in die Technologie entscheidend, sondern auch für die Einhaltung regulatorischer Vorgaben und ethischer Standards. Definiert wird Erklärbarkeit von Chazette et al. wie folgt:

> „Ein System S ist in Bezug auf einen Aspekt X von S relativ zu einem Adressaten A im Kontext C nur dann erklärbar, wenn es eine Entität E (den Erklärer) gibt, die es A durch Angabe eines Informationskorpus I (die Erklärung von X) ermöglicht, X von S in C zu verstehen“[8].

Dies heißt also, dass ein System S nur dann erklärbar für den Adressaten A sein kann, wenn es einen expliziten Teil des Systems namens E gibt, der diese Erklärung bereitstellt. Dabei ist diese Erklärung I vom Kontext C abhängig, kann also beispielsweise für unterschiedliche Adressaten anders aussehen. I ermöglicht es dem Adressaten A, mit dem System S in gewünschter Weise zu interagieren. Dabei kann der Aspekt X ganz unterschiedliche Dimensionen annehmen. Oft wird Erklärbarkeit im Kontext der KI betrachtet. Hier ist es für den Nutzer nur schwer nachvollziehbar, auf welche Art und Weise die KI zu einem Ergebnis kommt. Allerdings werden auch sonstige Softwaresysteme immer komplexer in ihrer Funktion, so dass auch hier Erklärungsbedarf besteht [12].

Ein wichtiger Aspekt ist ebenso die Individualisierung des Systems, welches sich so verschiedenen Nutzergruppen anpassen kann. Nutzergruppen

haben nicht alle den gleichen Erklärungsbedarf [9]. Sie sind unterschiedlich vertraut mit dem System. So kann die Umfänglichkeit, mit der das System verstanden wird, von der Häufigkeit der Nutzung oder dem Alter der Nutzer abhängen [9].

2.2.1 Erklärbarkeits als nichtfunktionelle Anforderung

Erklärbarkeit wird den nichtfunktionellen Anforderungen (NFRs) zugeordnet, weil sie sich auf die Qualität und Benutzerfreundlichkeit eines Systems bezieht, nicht auf dessen direkte Funktionalität [12]. NFRs betreffen die Art und Weise, wie ein System seine Aufgaben erfüllt, wie z. B. Sicherheit, Leistung, Wartbarkeit und Benutzerfreundlichkeit (siehe Abbildung 2.1.3). Die Erklärbarkeit eines Systems ist entscheidend dafür, wie gut ein Benutzer oder Entwickler die Funktionsweise des Systems, seine Entscheidungen oder Ergebnisse nachvollziehen kann [28, 9]. Diese Nachvollziehbarkeit ist besonders wichtig in komplexen oder sicherheitskritischen Anwendungen, wo das Verständnis der Systemlogik für die Vermeidung von Fehlern und für das Vertrauen in das System essenziell ist. Da Erklärbarkeit die Interaktion des Nutzers mit dem System erleichtert, ohne die eigentliche Funktion des Systems zu beeinflussen, wird sie als eine nichtfunktionale Anforderung betrachtet.

2.2.2 Bedeutung der Erklärbarkeit

Dabei sind einige weitere Aspekte besonders hervorzuheben. So dient Erklärbarkeit der Vertrauensbildung von Nutzern für KI-Systeme. Sie können so die Funktionsweise der Algorithmen verstehen und deren Entscheidungen nachvollziehen. Dies ist gerade in den heutigen komplexen Systemen essentiell [12]. Denn die immer komplexer werdenden Algorithmen stellen die Nutzer vor eine große Blackbox [26]. Dabei reicht oft eine nicht verstandene Antwort des Systems, um Unklarheit über die Wirkungsweise des ganzen Systems hervorzurufen.

Außerdem trägt Erklärbarkeit zur Fehlererkennung und -behebung bei. Fehler können so mit Hilfe von erklärbaren Systemen leichter identifiziert und korrigiert werden. Dies verbessert die Zuverlässigkeit und Effizienz des Systems. Entwickler und Benutzer können so schneller und präziser erkennen, wo ein Problem liegt. Beispielsweise können unerwartete Ergebnisse oder abweichende Systementscheidungen durch eine nachvollziehbare Erklärung auf ihre Ursachen hin untersucht werden. Erklärbare Systeme ermöglichen es, unzureichende Modullierungen oder fehlerhafte Algorithmen besser zu identifizieren [26].

Zudem ermöglichen erklärbare Systeme die Schaffung von transparenten Entscheidungen der KI. Dies hilft dabei, Diskriminierungen vorbeugen [5]. Die Erklärbarkeit schafft ein Verständnis der Nutzer dafür, wie ein System

entscheidet [14]. Dies ist beispielsweise bei der Kreditvergabe oder im Gesundheitswesen von besonderer Bedeutung. Eine Aufklärung über die getroffenen Entscheidungen stärkt nicht nur das Vertrauen der Nutzer in das System, sondern kann auch aufzeigen, welche Aspekte verbessert werden könnten, um bei der nächsten Benutzung erfolgreich zu sein. Langfristig fördert Erklärbarkeit also nicht nur das Vertrauen in KI-Systeme, oder auch andere komplexe Systeme, sondern unterstützt auch die Entwicklung fairer und ethisch verantwortlicher Technologien [5]. Sie hilft Diskriminierung zu minimieren, indem sichergestellt wird, dass die getroffenen Entscheidungen nachvollziehbar, überprüfbar und nach ethischen Richtlinien getroffen werden [14].

Kapitel 3

Verwandte Arbeiten

In der vorangegangenen Literaturrecherche wurden die Grundlagen des *Requirements Engineerings* und der *Erklärbarkeit* erläutert. Darauf aufbauend werden in diesem Kapitel verwandte Arbeiten vorgestellt, deren Erkenntnisse ebenso das Fundament des in dieser Arbeit entwickelten Konzepts bereitstellen.

Dehn führt in ihrer Arbeit „Anforderungen verschiedener Stakeholdergruppen an die Stimmungsanalyse in Softwareprojekten“ [11] eine Identfizifzierung von Stakeholdern durch, um mit diesen eine Erhebung und Priorisierung von Anforderungen vorzunehmen. Die Arbeit zeigt, dass zwischen den unterschiedlichen Stakeholdergruppen sehr unterschiedliche Bedürfnisse an die gefundenen Anforderungen vorliegen. Die Priorisierung der Anforderungen wurde von den Stakeholdern sehr auf die eigenen Vorstellungen und Wünsche abgestimmt, so dass zwischen den Gruppen beachtliche Unterschiede in der Priorisierung festgestellt werden konnten. In Abgrenzung zu Dehns Arbeit werden in dieser Arbeit Erklärbarkeitsanforderungen von Experten priorisiert. Hierbei versetzen sich diese Experten mit Hilfe ihrer Erfahrung in die Rolle des Nutzers, um aus dessen Sicht gefundene Erklärbarkeitsanforderungen zu priorisieren.

Einen grundlegenden Überblick über den Prozess des Requirement Engineerings geben Rupp et al. in ihrer Arbeit „Requirements Engineering und Management“ [27]. Hier werden praxisorientiert die einzelnen Schritte des Requirement Engineerings erklärt. Diese werden in sechs Schritten unterteilt. Beginnend mit der Vorbereitung werden die grundlegenden Ziele und Stakeholder identifiziert. Diese werden im nächsten Schritt spezifiziert. Dazu werden bereits vorhandene Dokumente geprüft sowie Interviews und Umfragen zur Erhebung vorbereitet. Diese werden dann mit den verschiedenen Stakeholdergruppen durchgeführt. In Schritt vier werden die Daten ausgewertet und priorisiert. Auch können hier schon erste Modellierungen zur Veranschaulichung erfolgen. Anschließend können die gefundenen Anforderungen validiert werden, bevor sie im letzten Schritt schriftlich festgehalten

werden können. Die in dieser Arbeit durchgeführten Anforderungserhebung orientiert sich stark an den hier vorgestellten Schritten. Dabei wird vor allem der iterative Ansatz der Anforderungserhebung aufgegriffen.

Balci stelt in seiner Arbeit „Entwurf eines Requirements Engineering Workflows für erklärbare Systeme“ [1] die endnutzerzentrierte Perspektive auf die Anforderungserhebung von Erklärbarkeiten in den Vordergrund. Dabei gelangt er zu der Erkenntnis, dass ein endnutzerzentrierter Fokus entscheidend für den Erfolg der Anforderungserhebung sein kann. Die Erkenntnisse der wissenschaftlichen Literatur sowie die Ergebnisse der durchgeführten Experteninterviews belegen eine Eignung für die Erhebung von Erklärbarkeitsanforderungen. Diese Arbeit nutzt diese Erkenntnisse und legt einen Fokus auf die Sicht des Endnutzers, um Erklärbarkeiten zu identifizieren und zu priorisieren.

Ein wichtiger Grundstein dieser Arbeit stellt die von Chazette durchgeführte Analyse von Anforderungen an ein System in Bezug auf Verständlichkeit und Nachvollziehbarkeit für das Verhalten dieses Systems. In ihrer Arbeit „Requirements Engineering von erklärbaren Systemen“ [7] beschreibt sie die enge Kommunikation sowie den Aufbau eines gemeinsamen Verständnisses zwischen Stakeholdern und Entwicklern als essentiell für den Projekterfolg. Diese Arbeit nutzt die gewonnenen Erkenntnisse und bindet diese in das entwickelte Konzept zur Erhebung von Erklärbarkeitsanforderungen mit ein.

Mistler beschreibt in seiner Arbeit die immer weiter zunehmende Komplexität von Systemen [21]. Er erkennt die Lösung in einem strukturierten Ansatz, welcher auf agile Art die Komplexität modellieren kann und auf diese Weise einen systematischen Ansatz bietet, um die Erhebung von Anforderungen durchführen zu können. Auch hier liegt der Fokus sehr auf der Identifizierung der einzelnen Stakeholder, sowie der engen Zusammenarbeit mit diesen. Die agile Denkweise wurde in dieser Arbeit übernommen und fließt in den Ablauf des vorgestellten Konzepts mit ein.

Herzog entwickelt in seiner Arbeit „Konzeption und Evaluation eines Modells zur Unterstützung des Designs von Erklärungen in erklärbaren Systemen“ [19] einen Leitfaden zur strukturierten Erhebung von Erklärungen. Dieser erleichtert eine systematische Erhebung von Erklärbarkeitsanforderungen. Herzog entwickelt Heuristiken, die für eine allgemeine Erhebung anwendbar sein sollen. So sei auszugsweise etwa *Context-Sensitivity* wichtig, um den Nutzern nur erforderliche Erklärungen bereitzustellen. *Minimally Complete Explanations* verhindern einen Informationsüberfluss. Erklärungen müssen vollständig sein, sollen aber so kurz und präzise wie möglich sein. Ein anderer wichtiger Aspekt sei die *User Perception*. Die Wahrnehmung des Nutzers sollte stets im Zentrum von Erklärungen stehen. Dies sind wichtige Erkenntnisse, die in dieser Arbeit für die Konzeptuierung des Erhebungsprozesses von Erklärbarkeitsanforderungen einfließen.

Droste et al. haben in ihrem Paper „Explanations in Everyday Software Systems: Towards a Taxonomy for Explainability Needs“ [12] eine Taxonomie

entwickelt, mit deren Hilfe sich einzelne Felder des Erklärungsbedarfs identifizieren lassen. Dabei wurden verschiedene Softwaresysteme identifiziert, welche unterschiedliche Strategien für Erklärungen bevorzugen. In dieser Arbeit wird ein Prototyp entwickelt, welcher unter *Consumer-oriented Software* fällt. Es wurden insgesamt fünf *explainability needs* identifiziert.

1. System Behavior
2. Interaction
3. Domain Knowledge
4. Privacy & Security
5. User Interface

Die Ergebnisse dieser Arbeit wurden für die Ausarbeitung eines Experteninterviews für die Erhebung von Erklärbarkeitsanforderungen in abgewandelter Form verwendet.

Kapitel 4

Konzept

Die Erklärbarkeit von Software und Algorithmen, insbesondere von KI-Systemen, ist unerlässlich, um Vertrauen und Transparenz sicherzustellen. Im Folgenden wird ein Konzept vorgestellt, das eine systematische Erhebung und Implementierung von Erklärbarkeitsanforderungen ermöglicht.

Dieses Konzept geht sowohl auf die Erhebung als auch auf die Implementierung detailliert ein, bevor die gewonnenen Erkenntnisse zu einem strukturierten Gesamtbild zusammengeführt werden. Das entwickelte Konzept basiert auf der Flow-Methode [29]. Anschließend werden Anforderungen an eine Navigations-App am Beispiel dieses Konzept in 5 erhoben und in 6 praktisch umgesetzt.

4.1 Erhebung

Um die Erklärbarkeitsanforderungen für das zu entwickelnde System zu erfüllen, müssen diese zunächst identifiziert und detailliert ausgearbeitet werden. Dieses Vorgehen sollte klar strukturiert erfolgen, um die Wahrscheinlichkeit von Fehlern und Missverständnissen zu minimieren. Abbildung 4.1 veranschaulicht diesen Erhebungsprozess in seiner Gesamtheit.

Der Requirements Engineer spielt in diesem Prozess eine zentrale Rolle. Seine Aufgabe besteht darin, alle Nutzergruppen zu identifizieren, welche das System verwenden werden. Werden Nutzergruppen übersehen, können wesentliche Erklärbarkeitsaspekte bei der Implementierung unberücksichtigt bleiben. Nachträgliche Änderungen sind deutlich teurer und aufwändiger als die direkte Berücksichtigung dieser Anforderungen von Beginn an. Sind alle Nutzergruppen identifiziert, gilt es mit diesen Interviews und Umfragen durchzuführen. Der Requirements Engineer erstellt dafür die erforderlichen Schriftstücke. Dabei bringt er seine Erfahrung ein, um die Erhebung möglichst effizient zu gestalten. Im Rahmen der Gespräche mit den verschiedenen Nutzergruppen werden diese auch aufgefordert, ihre genannten und bereits identifizierten Erklärbarkeitsanforderungen zu priorisieren. Dies

wird im Implementierungsprozess relevant, da Deadlines und Kapazitäten die Entwicklung einschränken können. So wird sichergestellt, dass die wichtigsten Anforderungen zuerst implementiert werden können.

Die Interviews und Umfragen werden anschließend in einer Erklärbarkeitsanalyse ausgewertet. In diesem Schritt werden die erkannten Erklärbarkeitsanforderungen konkret benannt und verschriftlicht. Diese Formalisierung ist essentiell für die Fertigstellung eines Anforderungsdokuments. Anschließend werden die schriftlich formulierten Anforderungen vom Requirements Engineer evaluiert. Es entsteht ein Anforderungsdokument, welches mit größter Sorgfalt auf Missverständnisse und Kommunikationsfehlern mit den Nutzergruppen untersucht wird. Bevor der Entwicklungsprozess beginnen kann, wird dieser Prozess bestehend aus der Analyse der Interviews und Umfragen und dessen Evaluierung zyklisch wiederholt, bis letztlich ein finales Anforderungsdokument entsteht, welches alle identifizierten Erklärbarkeitsformulierungen konkret benennt [27].

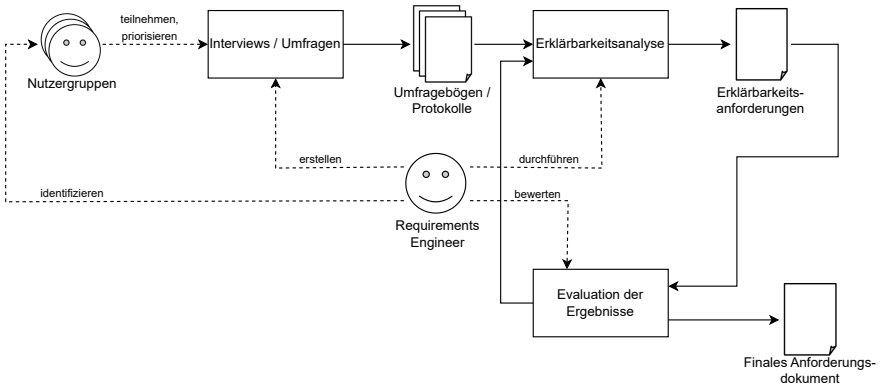


Abbildung 4.1: Konzept zur Erhebung von Erklärbarkeitsanforderungen

4.2 Implementierung

Sobald ein vom Product Owner abgenommenes Anforderungsdokument mit allen identifizierten Erklärbarkeitsanforderungen vorliegt, kann mit der Implementierung begonnen werden. Abbildung 4.2 zeigt diesen Ablauf konzeptuell.

Hier wird vom Projektleiter der Workload auf die Entwicklerteams aufgeteilt. Dabei kann es auch passieren, dass es nur einen einzelnen Entwickler gibt. Da die Erklärbarkeitsanforderungen priorisiert vorliegen (siehe 4.1), können die wichtigsten Erklärungen zuerst implementiert werden. Der Entwicklungsprozess kann auf verschiedene Art und Weise durchgeführt werden (siehe 2.1.2). Letztlich wird von den Entwicklern Code generiert,

welcher die herausgearbeiteten Erklärbarkeitsanforderungen an das System erfüllt. Dieser Code wird im nächsten Schritt von Testern auf Fehler überprüft.

Abhängig vom Ergebnis der Stakeholder und Tester, kann das System in die nächste Entwicklungsphase des Projekts geschoben werden, etwa in die Produktionsphase. Diese ist nicht mehr in Abbidung 4.2 abgebildet. Allerdings wird das System auch vom Requirements Engineer auf Erklärbarkeitsmängel untersucht. Der Erhebungsprozess kann bei groben Fehlern von Neuem beginnen. Dieser baut dann auf den bisherigen Erkenntnissen auf und versucht die bislang übersehenen Missverständnisse und fehlerhaften Implementierungen zu berücksichtigen. Es kann auch nur der Implementierungsprozess von neuem beginnen.

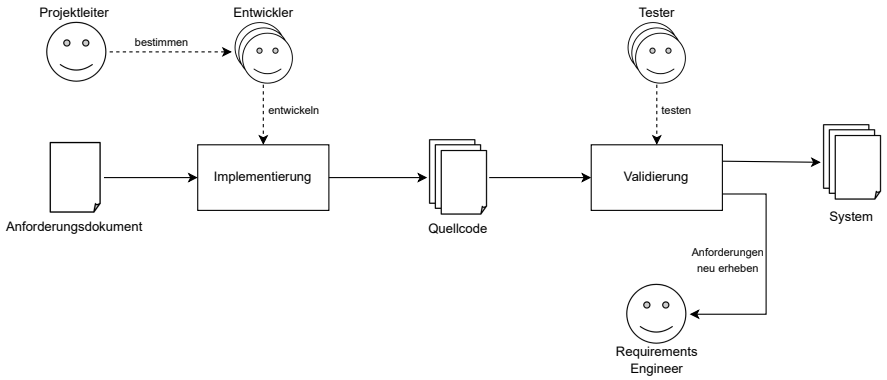


Abbildung 4.2: Konzept zur Implementierung von Erklärbarkeitsanforderungen

4.3 Gesamtkonzept

Aus den beiden vorgestellten Arbeitsschritten Erhebung und Implementierung von Erklärbarkeitsanforderungen ergibt sich ein Prozess, mit dessen Hilfe ein systematisches Erarbeiten von Erklärbarkeitsanforderungen möglich ist. Abbildung 4.3 zeigt einen gröberen Ablauf des Gesamtprozesses als die zuvor vorgestellten Konzepte.

Besonders hervorzuheben ist dabei, dass dieser Gesamtprozess iterativ mehrfach durchlaufen werden kann. Wie genau und umfassend bereits im ersten Zyklus die Erhebung der Erklärbarkeitsanforderungen durchgeführt wird, hängt auch von der Strategie des Entwicklungsprozesses ab. So nimmt die Erhebung etwa beim Wasserfallmodell deutlich mehr Zeit in Anspruch [25], hingegen wird bei agileren Strategien der Zyklus kürzer gehalten und stattdessen öfter durchlaufen, um schnell auf Veränderungen reagieren zu können[3].

Auffällig ist vor allem die zentrale Rolle der Requirement Engineers. Dies zeigt, wie wichtig die sorgfältige Erhebung von Erklärbarkeiten ist. Außerdem bedeutet die Einbindung der Nutzergruppen für den Erfolg des Projekts, dass die nötigen Erklärungen bereits im Vorfeld identifiziert werden. Denn ohne die erforderlichen Erklärbarkeiten sind heutige Systeme oft zu komplex und unverständlich für den einzelnen Nutzer, was solche Systeme praktisch unbrauchbar werden lässt.

Zusammenfassend lässt sich sagen, dass sowohl Erhebung als auch Implementierung von Erklärbarkeiten einen strukturierten Prozess bieten, der die systematische Entwicklung solcher Anforderungen erlaubt. Besonders hervorzuheben ist die iterative Natur dieses Prozesses, die es erlaubt, ihn mehrfach zu durchlaufen. Der Requirements Engineer spielt dabei vor allem im Erhebungsprozess eine zentrale Rolle, aber auch in der Implementierungsphase wird er eingebunden, um die Erklärungen zu überprüfen. Die Berücksichtigung von Erklärbarkeitsanforderungen ist also nicht nur ein „Add-On“ für ein zu entwickelndes System, sondern ein essentieller Bestandteil des Erhebungs- und Entwicklungsprozesses.

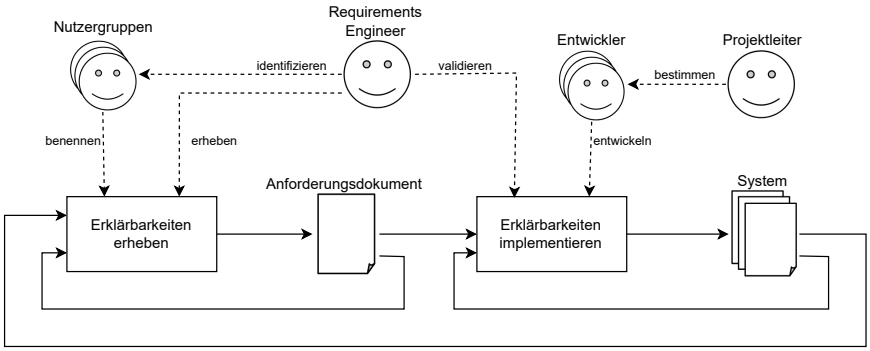


Abbildung 4.3: Überblick über den Gesamtprozess zur Erhebung und Implementierung von Erklärbarkeitsanforderungen

Kapitel 5

Anforderungserhebung

Die Entwicklung einer benutzerfreundlichen und verständlichen Navigationsanwendung erfordert eine sorgfältige Anforderungserhebung, um die Bedürfnisse und Erwartungen der Nutzer optimal zu berücksichtigen. In diesem Kapitel wird der Prozess der Erhebung und Priorisierung von funktionellen (*functional Requirements*, FRs) und nichtfunktionellen (*nonfunctional Requirements*, NFRs) Anforderungen detailliert beschrieben. Dafür wird das in 4.1 vorgestellte Konzept verwendet. Dabei liegt der Fokus auf den nichtfunktionellen Erklärbarkeitsanforderungen. Dafür wird zunächst in 5.1 die Software vorgestellt, bei der die erhobenen Anforderungen prototypisch umgesetzt werden sollen. Anschließend wird die Gestaltung 5.2 und Auswertung 5.3 der Studie zur Erhebung dieser Anforderungen beschrieben, bevor diese letztlich in 5.4 formuliert werden.

5.1 Nunav Courier

Nunav Courier ist eine spezialisierte Navigationsanwendung der Firma *Graphmasters GmbH* in Hannover, die darauf abzielt, die Effizienz und Produktivität von Kurierdiensten und Lieferunternehmen zu maximieren [16]. Die App bietet eine Vielzahl von Funktionen, die eine optimale Routenplanung und Navigation ermöglichen. Eine der zentralen Funktionen ist die Routenoptimierung, bei der die besten Routen für Zustellfahrzeuge unter Berücksichtigung aktueller Verkehrbedingungen, Straßensperrungen und anderer relevanter Faktoren berechnet werden. Diese Funktion führt zu kürzeren Fahrzeiten und einer effizienteren Auslastung der Fahrzeuge. Insgesamt stellt Nunav Courier ein Werkzeug für Kundendienste und Lieferunternehmen dar, um Paketzustellungen effizient und verlässlich durchführen zu können.

Im Zuge dieser Arbeit wurde ein Prototyp entwickelt, welcher die Adresseneingabe der einzelnen Tourstopps mithilfe von künstlicher Intelligenz (KI) verbessern soll. In der aktuellen Version der App kann der Nutzer sich

eine Tour mit einer beliebigen Anzahl an Stopps zusammenstellen. Diese werden dann von Nunav Courier zu einer optimierten Tour zusammengefasst. Dabei werden die Stopps bislang händisch durch ein Textfeld oder einzeln durch Speech-To-Text vom Nutzer in die App eingetragen (siehe A.1). Dieser Prozess gestaltet sich oft als sehr langwierig, sobald mehrere Dutzend Adressen für eine einzelne Tour eingegeben werden müssen. Die prototypisch implementierte KI-basierte Adresseneingabe ermöglicht es dem Nutzer, frei mit Nunav Courier zu sprechen und so den Prozess der Adresseneingabe zu beschleunigen. Die in dieser Funktion verwendete KI nutzt *Natural Language Processing* (NLP), um die gesprochene Sprache zu verstehen und korrekt zu interpretieren. Die KI hört die Adressen, die der Nutzer nennt, heraus und fügt diese als einen neuen Tourstop der Tour hinzu. Ein Geocoder ordnet reale Geokoordinaten den Tourstopps zu, um eine optimale Navigationsroute für den Nutzer zu erstellen. Für die Erhebung der konkreten Anforderungen für diesen Prototypen wurde eine Studie durchgeführt.

5.2 Gestaltung der Studie

Um ein umfassendes Verständnis für die Nutzeranforderungen zu gewinnen, wurde zunächst ein Experteninterview durchgeführt. Aufgrund der schwer erreichbaren Nutzerbasis der App, wurde dieses Interview und die darauf aufbauende Umfrage nicht mit tatsächlichen Endnutzern durchgeführt. Stattdessen wurden Experten, darunter erfahrene Entwickler, ehemalige Kurierfahrer und Fachleute aus dem Bereich der Navigationssoftware, einbezogen. Diese Experten verfügen über tiefgehende Kenntnisse und Erfahrungen im Bereich der Nutzung und Entwicklung von Navigationslösungen, was sie zu wertvollen Teilnehmern für die Anforderungserhebung macht. Die Ergebnisse der Umfrage bilden die Grundlage für die formale Erhebung der Anforderungen, welche in den folgenden Abschnitten dieses Kapitels systematisch analysiert und dokumentiert werden.

Diese Arbeitsweise ermöglicht die Identifikation relevanter Sektionen und Aspekte für die Identifizierung von Erklärungsbedarf. Sie werden benötigt, um eine strukturierte Umfrage gestalten und durchführen zu können. Dabei können auf diese Weise die Limitierungen und zeitliche Beschränkung dieser Arbeit besser berücksichtigt werden. In dem Interview sollten FRs und NFRs in ihrer Wichtigkeit bewertet werden. Dazu wurden zu verschiedenen Aspekten Anforderungen genannt und nach ihrer Wichtigkeit bewertet 5.1.

Auf Basis dessen haben sich fünf Themengebiete herauskristallisiert, welche im Kontext der Erklärung von besonderer Bedeutung sind.

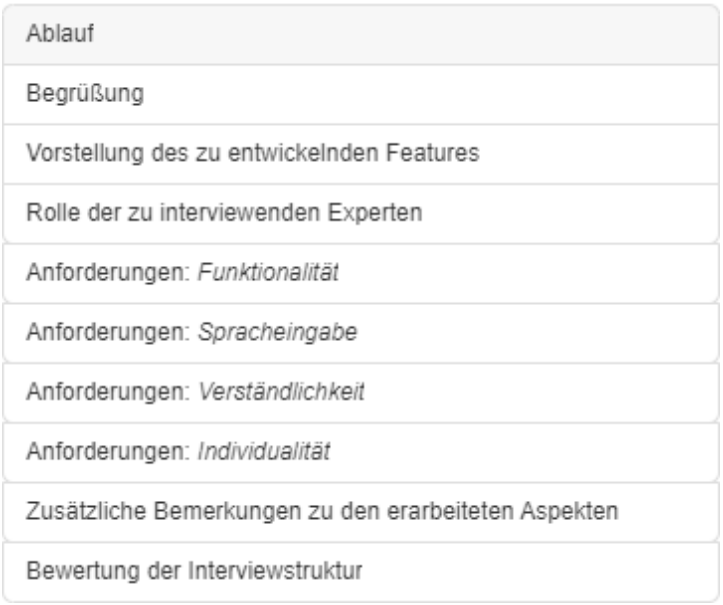


Abbildung 5.1: Ablauf des Experteninterviews

1. Interaktion
2. Spracheingabe
3. Hilfestellung
4. Transparenz
5. Erklärungen

Für diese wurden jeweils die vier Anforderungen herausgearbeitet, welche in dem Interview als am wichtigsten bewertet wurden (siehe B). Diese Erkenntnisse stellen die Basis für eine Umfrage dar, aus welcher anschließend die formellen Anforderungen herausgearbeitet werden.

In der Umfrage sollten die Teilnehmer dann potentielle FRs und NFRs des zu entwickelnden Prototypen mit Hilfe einer fünfstufigen Likert-Skala bewerten (siehe 5.2). Außerdem sollten die einzelnen Anforderungen einer Gruppe zusätzlich von den Teilnehmern priorisiert werden. Dafür mussten die vier dort genannten Aspekte in eine nach der vom Teilnehmer empfundenen Wichtigkeit in Reihenfolge gebracht werden. Dazu wurden im folgenden Abschnitt die Ergebnisse der einzelnen Gruppen ausgewertet, um anschließend diese Ergebnisse für die Formalisierung der erhobenen Anforderungen zu verwenden.

5.3 Auswertung der Studie

Für die Auswertung der ordinalen Daten, welche sich aus den Likert-Skalen ergeben haben, wird der Median betrachtet, um eine Aussage über die mittlere Bewertung der Teilnehmer treffen zu können. Für die Auswertung der Priorisierung der Anforderungen wurde ein Punktbewertungssystem verwendet. Bei diesem werden Reihenfolgen in Punktwerte umgewandelt. Die von den Teilnehmer vorgenommene Platzierung wird aufsummiert, um eine Gewichtung der einzelnen Anforderungen vornehmen zu können. Bei gleicher Punktzahl wird die Anzahl der besten Platzierung berücksichtigt. Insgesamt haben neun Personen an der Umfrage teilgenommen.

5.3.1 Interaktion

Die folgenden Anforderungen wurden unter der Gruppe *Interaktion* zusammengefasst. Diese zielen darauf ab, die Benutzerfreundlichkeit und die Bedienbarkeit der Anwendung zu verbessern. Eine intuitive und reibungslose Interaktion ist entscheidend, um die Effizienz und Zufriedenheit der Benutzer zu steigern. Dabei spielen Feedback-Mechanismen und eine intuitive Bedienung eine besondere Rolle. Die Nutzung einer sprachgesteuerten Adresseneingabe wirft für die Interaktion viele Fragen auf.

- **F1:** Die bisher erkannten Adressen sollten mit einer Swipebewegung wieder entfernbar sein.
- **F2:** Mir ist wichtig, dass ich Adressen auch manuell noch abseits der Spracheingabe hinzufügen kann.
- **F3:** Ich sollte das Smartphone für den gesamten Bedienungsprozess der Adresseneingabe aus der Hand legen können.
- **F4:** Bereits gefundene Adressen sollten schon während der sprachgesteuerten Eingabe modifizierbar sein.

Insgesamt ist die Zustimmung für Aussage F2 am größten von den vier ausgearbeiteten Interaktions-Aussagen mit einem Median von 4 (siehe Abbildung 5.2). Auffällig ist hier auch, dass keiner der Teilnehmer einer manuellen Addresseneingabe neutral oder ablehnend gegenüber steht. Auch wenn die fokussierte Methodik der Adresseneingabe die sprachgesteuerte und KI-interpretierte Eingabe ist, scheint es für Nutzer wichtig zu sein, dass ihnen ebenfalls bereits bekannte und bewährte Art und Weisen zur Verfügung stehen, mit denen sie alternativ Adressen hinzufügen können. F3 wird ebenfalls mit einem Median von 4 zu großen Teilen zugestimmt. Hier zeigt sich, dass der Use Case der Adresseneingabe stark davon profitieren

kann, wenn der Fahrer währenddessen das Gerät aus der Hand legen kann, um etwa Pakete für die Adressenfindung zu greifen. F1 und F4 wurden von den Teilnehmern sehr differenziert bewertet. So wurden den Aussagen sowohl von zwei Teilnehmern stark zugestimmt, wie aber auch jeweils von einem Teilnehmer stark abgelehnt. Dies deutet darauf hin, dass hier individualisierte Funktionalitäten bereitgestellten werden sollten. Die Nutzer haben unterschiedliche Interaktionsbedürfnisse, um Änderungen an den bereits gefundenen Adressen vornehmen zu können. Dass diese Adressen schon während der Spracheingabe modifizierbar sein sollten, stößt am meisten auf Ablehnung. Allerdings liegt der Median auch für F1 und F4 bei 4, was auf insgesamt großen Erklärungsbedarf für Interaktionen bei der Spracheingabe hindeutet.

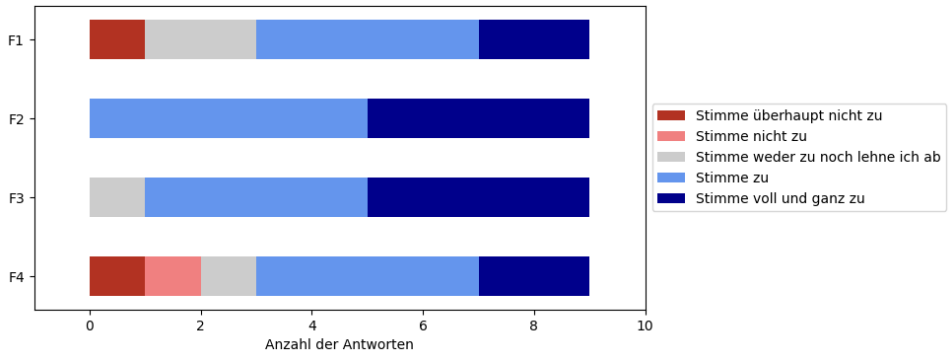


Abbildung 5.2: Bewertung von Interaktionsanforderungen mit Hilfe einer Likert-Skala

Abbildung 5.3 zeigt, dass bei der Priorisierung ebenfalls die manuelle Eingabe der Adressen an erste Stelle gesetzt wird. Interessanterweise weicht die Reihenfolge aber von den Bewertungen der einzelnen Aussagen leicht ab. So wird F1 vor F3 priorisiert, obwohl F1 ablehnender und neutraler bewertet wurde als F3. Dies könnte man mit dem für jeden Nutzer unterschiedlichen Kontext erklären. Adressen schnell mit einer Swipebewegung entfernen zu können, könnte in den Augen der Nutzer häufiger relevant und essentieller für den Gesamtablauf der Adresseneingabe sein. Nicht jeder Nutzer hält es für persönlich wichtig, dass das Endgerät während der Adresseneingabe aus der Hand gelegt werden kann.

5.3.2 Spracheingabe

Bei der Anforderungserhebung in diesem Kontext auch funktionelle Aspekte der Spracheingabe zu erheben, ist essentiell, da diese eines der zentralen Bausteine für den zu entwickelnden Prototypen darstellen. Da der Fokus dieser Arbeit aber auf der Durchführung des in 4.1 vorgestellten Konzepts liegt, werden hier nur vier Anforderungen betrachtet.

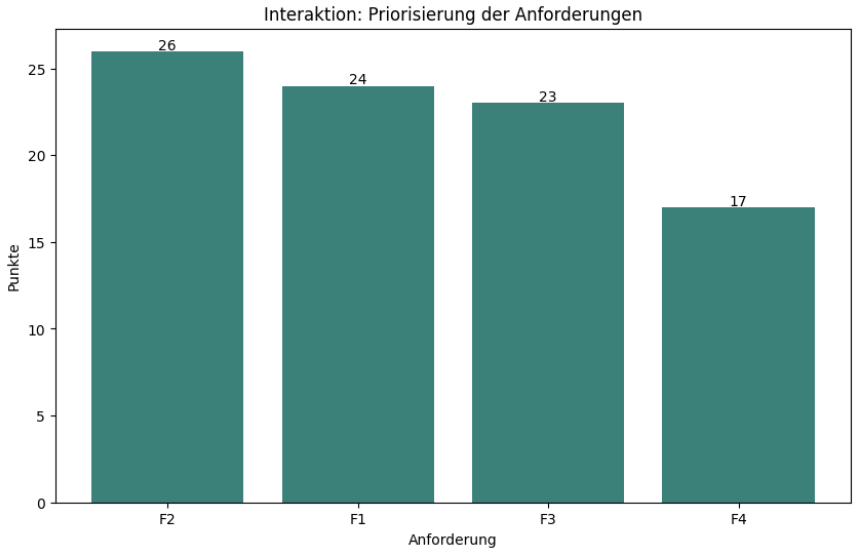


Abbildung 5.3: Priorisierung der Interaktionsanforderungen

- **F5:** Die Spracheingabe sollte Sprachen abseits von Deutsch verstehen, auch wenn nur deutsche Adressen eingegeben werden.
- **F6:** Die Spracheingabe sollte aktiv Feedback geben, wenn sie meine Worte nicht verstanden hat.
- **F7:** Die App sollte meine gesprochenen Worte in Form von Untertiteln visuell wiedergeben.
- **F8:** Die Spracheingabe sollte auch verbal gestoppt werden können, z.B. durch ein gesprochenes „Stop“.

F5 wurde in 5.4 zu großen Teilen mit „Stimme zu“ bewertet, was die konsistenten Antworten um den Median 4 zeigen. Eine zentrale Nutzergruppe der in 5.1 vorgestellten Navigationsapp sind berufliche Kurierfahrer. Hier zeigt sich, dass diesen Beruf nicht nur deutschsprachige Personen wahrnehmen und ein Bedarf für die Unterstützung verschiedener Sprachen existiert. Am höchsten wird F6 bewertet mit einem Median von 5. Hier gaben zwei Drittel der Teilnehmer die Bewertung „Stimme voll und ganz zu“ ab. Aktives Feedback ist ein zentraler Weg, um Erklärungen in einem System verständlich zu machen. Dies zeigt auch die hohe Punktzahl in 5.5 mit insgesamt 31 Punkten, bei der direktes und aktives Feedback als am wichtigsten priorisiert wurde.

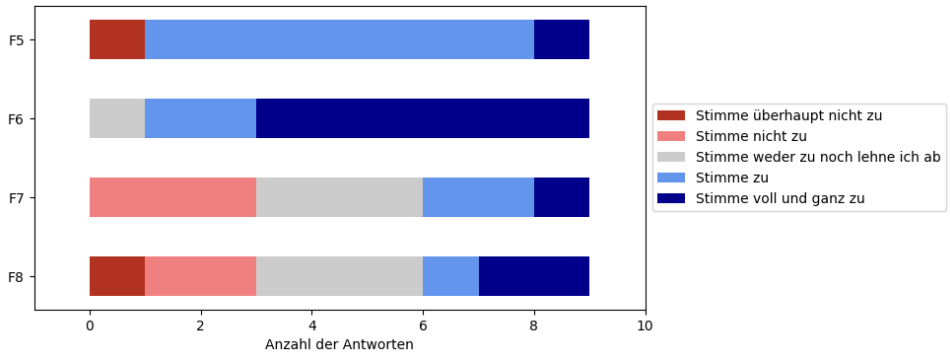


Abbildung 5.4: Bewertung von funktionellen Anforderungen an die Spracheingabe mit Hilfe einer Likert-Skala

Den Aussagen F7 und F8 stehen die Teilnehmer insgesamt neutral gegenüber. Dies zeigt, dass Erklärungen wie zum Beispiel ein visuelles Feedback der gesprochenen Worte nicht als notwendig angesehen wird. Es gilt also, dass mehr Erklärungen nicht gleich bessere Erklärungen für die Gesamtheit des Systems bedeuten. Die Wiedergabe von Untertiteln wird mit 14 Punkten als am unwichtigsten priorisiert.

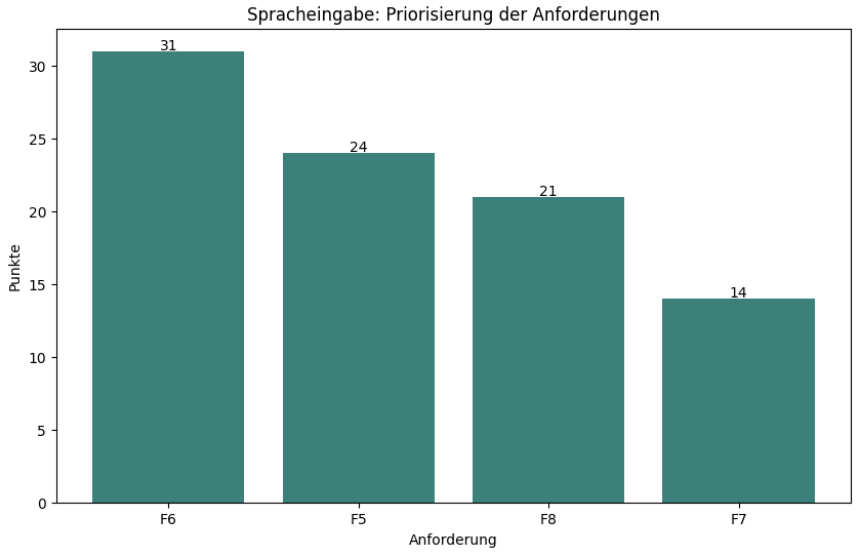


Abbildung 5.5: Priorisierung der funktionellen Anforderungen an die Spracheingabe

5.3.3 Hilfestellung

Hilfestellungen und Erklärungen sind essenziell, um die Benutzerfreundlichkeit, Effizienz und Akzeptanz einer sprachgesteuerten Adresseneingabe zu gewährleisten. Sie tragen dazu bei, das System zugänglicher und benutzerfreundlicher zu machen, Vertrauen aufzubauen und die Gesamtqualität der Benutzererfahrung zu verbessern. Daher wurden vier Aspekte herausgearbeitet, welche sich auf mögliche Hilfestellungen beziehen, um diese gewünschten Effekte zu erreichen.

- **F9:** Die App sollte bei der erstmaligen Benutzung der sprachgesteurten Adresseneingabe die einzelnen Arbeitsschritte ausführlich erklären.
- **F10:** Mir ist wichtig, dass die App eine gefundene Adresse z.B. durch einen sichtbaren grünen Haken bestätigt.
- **F11:** Die App sollte einen hörbaren Ton bei jeder hinzugefügten Adresse abspielen.
- **F12:** Die App sollte einen „Help Button“ bereit stellen, welcher die sprachgesteuerte Adresseneingabe erklärt.

Mit einem Median von 5 ist die Zustimmung für F10 am höchsten (siehe Abbildung 5.6). Im Gegensatz dazu bewegt sich F11 zwischen einer neutralen Haltung und voller Zustimmung. Die Teilnehmer sehen daher einen größeren Nutzen in visuellem als in auditivem Feedback. F12 wurde ausschließlich zugestimmt. Eine allgegenwärtige Möglichkeit, gezielte Hilfestellungen zur Funktionalität der sprachgesteuerten Adresseingabe zu erhalten, scheint den Teilnehmern sehr wichtig zu sein. Einzig F9 wurde explizit von einem Teilnehmer abgelehnt. Eine einmalige Einführung kann für Nutzer als umständlich und unnötig angesehen werden. Auch hier zeigt sich, dass zu viele Erklärungen das Nutzererlebnis beeinträchtigen können.

Die Priorisierung der Aussagen F9 bis F12 deckt sich hier sehr mit deren Bewertung. Eine permanente Möglichkeit zur Einholung von Erklärungen empfinden die Teilnehmer mit einer Punktzahl von 26 als wichtig. Im Gegensatz dazu wird mit 18 Punkten die einmalige detaillierte Einführung in diese Funktionalität als am unwichtigsten empfunden.

5.3.4 Transparenz

Transparenz in der Anforderungserhebung und -umsetzung ist essenziell, um Vertrauen zu schaffen, die Benutzererfahrung zu verbessern und ethische sowie rechtliche Standards zu erfüllen. Sie ermöglicht es Benutzern, das System

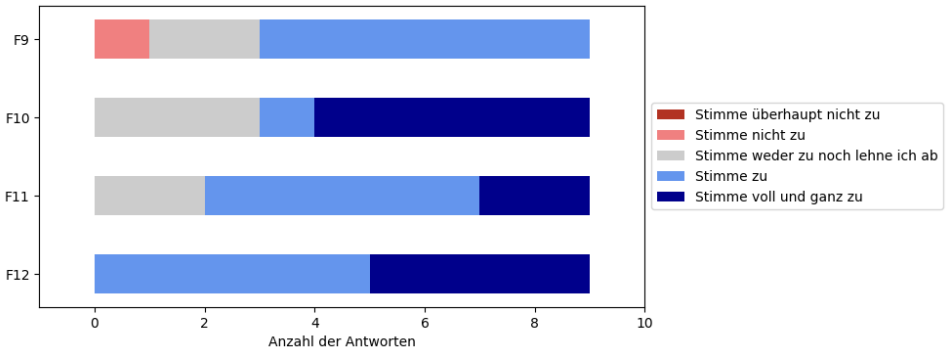


Abbildung 5.6: Bewertung von Hilfestellungskriterien mit Hilfe einer Likert-Skala

besser zu verstehen, sicher zu nutzen und konstruktives Feedback zu geben, was letztlich zur Entwicklung eines robusteren und benutzerfreundlicheren Systems führt.

- **F13:** Die App soll durch einen verständlichen Wert anzeigen, wie gut die Spracheingabe meine Worte verstanden hat.
- **F14:** Die App sollte vor der finalen Tourerstellung auf Adressen-Duplikate hinweisen.
- **F15:** Ich möchte zu jeder Zeit wissen, wie gut gerade meine Internetverbindung ist, um die Adresseneingabe vornehmen zu können.
- **F16:** Die App sollte mir erklären, dass ich mit einer KI kommuniziere.

Der Aussage F13 wird in 5.8 mit einem Median von 2 insgesamt nicht zugestimmt. Es scheint kein Interesse daran zu bestehen, technisches Feedback der Spracheingabe zu erhalten, um das Verständnis für gefundene Adressen zu erhöhen. Ebenfalls werden die Aussagen F15 und F16 mit einem Median von 3 neutral bewertet. Der Status der Internetverbindung sollte also nur bei unzureichender Verbindung erkenntlich sein. Andernfalls liegt in dieser Information kein Mehrwert für den Nutzer vor. Hier haben vier Teilnehmer explizit eine dauerhafte Anzeige abgelehnt. Ebenso wird die Information, mit einer KI zu reden, als nicht relevant erachtet. F14 wird jedoch zu weiten Teilen zugestimmt. Mit einem Median von 4 zeigt sich, dass Adressenduplikate für den Nutzer relevant sind.

F14 wurde mit einer Punktzahl von 33 als sehr wichtig eingestuft (siehe

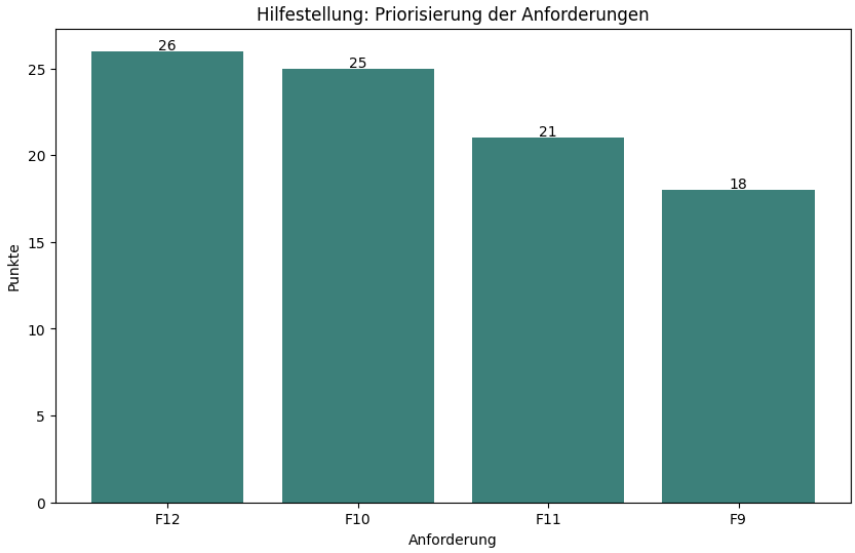


Abbildung 5.7: Priorisierung der Hilfestellungskritierien

Abbildung 5.9), gefolgt von F16 mit 23 Punkten. Dass diese beiden Aussagen wichtiger als F13 und F15 empfunden werden, zeigt dass die Sichtbarkeit von technischen Details als unwichtig empfunden wird. F13 und F15 wurden beide mit einer Punktzahl von 17 priorisiert. Da F13 von einem Teilnehmer aber auch als wichtigste Priorität angesehen wurde, wird hier ein Feedback zur Genauigkeit der Spracheingabe höher priorisiert als der Status des Internets.

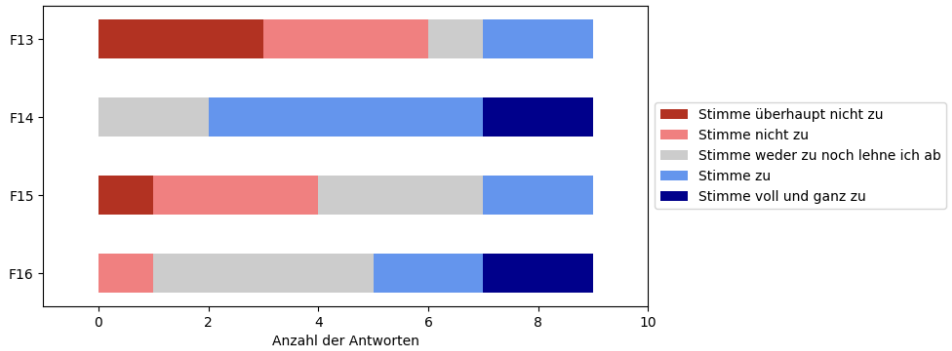


Abbildung 5.8: Bewertung von Transparenzanforderungen mit Hilfe einer Likert-Skala

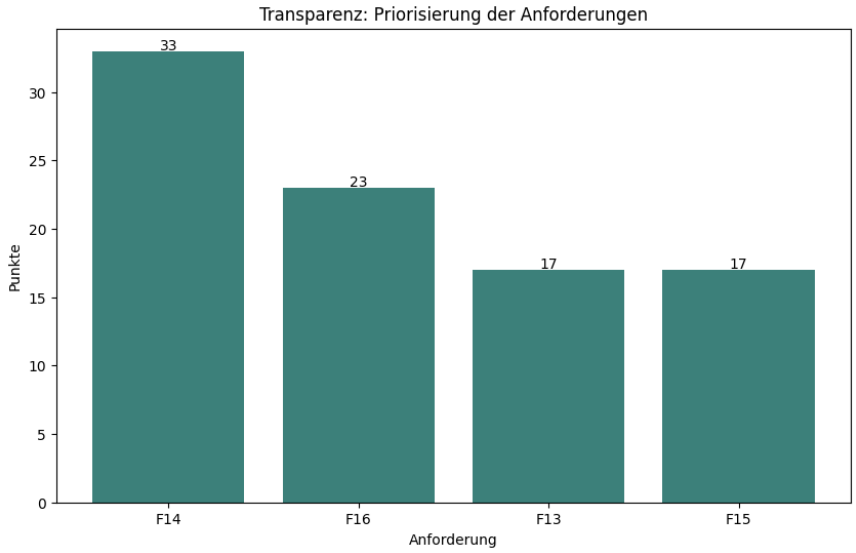


Abbildung 5.9: Priorisierung von Transparenzanforderungen

5.3.5 Erklärungen

Erklärungen sind unerlässlich, um die Benutzerfreundlichkeit, Effizienz, Sicherheit und Akzeptanz eines sprachgesteuerten Adresseneingabesystems zu gewährleisten. Sie fördern das Vertrauen der Benutzer, erleichtern die Selbsthilfe, machen das System zugänglicher und tragen zur kontinuierlichen Verbesserung bei. Insgesamt sind Erklärungen ein wesentlicher Bestandteil, um ein erfolgreiches und nutzerzentriertes System zu entwickeln.

- **F17:** Die sprachgesteuerte Adresseneingabe sollte erklären, welche Adressinformationen zwingend benötigt werden.
- **F18:** Nach jeder erkannten Adresse sollte die App die gefundene Adresse verbal wiederholen.
- **F19:** Das System sollte in der Lage sein, den Kontext der Adressangaben zu verstehen, um logische Fehler zu minimieren (z.B. unplausible Kombinationen von Stadt und Postleitzahl).
- **F20:** Das System greift auf frühere Eingaben des Benutzers zurück, um die Adresseneingabe zu vereinfachen.

Abbildung 5.10 zeigt, dass F19 mit einem Median von 5 insgesamt am höchsten zugestimmt wurde. Kontexterkennung in Systemen kann eine deutlich gesteigerte Nutzerzufriedenheit bewirken, da das System intelligenter

wird und die Ziele der Nutzer so oft schneller erreicht werden können. Zudem können falsch genannte Daten vom Nutzer eventuell korrigiert werden. F18 wird im Gegensatz dazu mehrheitlich abgelehnt. Hier scheint der Mehrwert für das Verständnis an das System einer Einschränkung des Nutzererlebnisses zu unterliegen. Eine Wiederholung der Adresse würde es dem Nutzer deutlich vereinfachen, die Resultate der KI zu verstehen, den Prozess der Adresseneingabe jedoch erheblich verlängern. F17 wird mit einem Median von 4 zugestimmt. Der Nutzer möchte wissen, welche Informationen zwingend benötigt werden und welche vielleicht optional auch weggelassen werden können. Ebenso wird mehrheitlich Aussage F20 zugestimmt. Wenn eine Eingabe schon einmal erfolgt ist, kann das System mit weniger Daten potentiell die vom Nutzer gewollte Adresse erkennen.

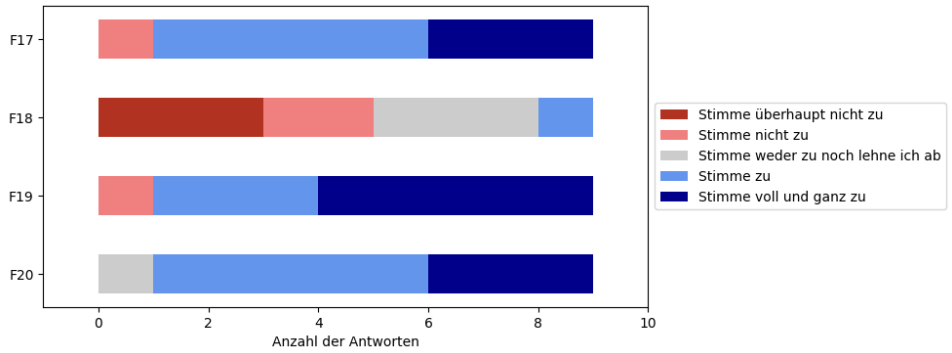


Abbildung 5.10: Bewertung der Aussagen von Erklärungsbedarf mittels einer fünfstufigen Likert-Skala

Priorisiert wird von den Teilnehmern F19 als wichtigste Anforderung mit 30 Punkten (siehe 5.11). F17 wird mit 26 Punkten als leicht wichtiger empfunden als F20 mit 22 Punkten. Als am unwichtigsten wird F18 mit 12 Punkten empfunden, was sich mit der Bewertung aus 5.10 deckt.

5.4 Formale Anforderungen

Auf Basis des durchgeführten Experteninterviews und der darauf abgestimmten Umfrage werden nachfolgend alle Anforderungen, welchen mit einem Median über 3 zugestimmt wurden, hier formal aufgestellt. Dabei bestimmt sich die Prioriserung durch den Median bei der Bewertung über die Zustimmung. Bei gleichem Median erhält die Anforderung mit der höheren Priorität innerhalb der Gruppe den Vorrang.

[R01] Die Spracheingabe gibt aktives Feedback, wenn die gesprochenen Worte nicht genau verstanden wurden.
Ab einer Unsicherheit von 80%.

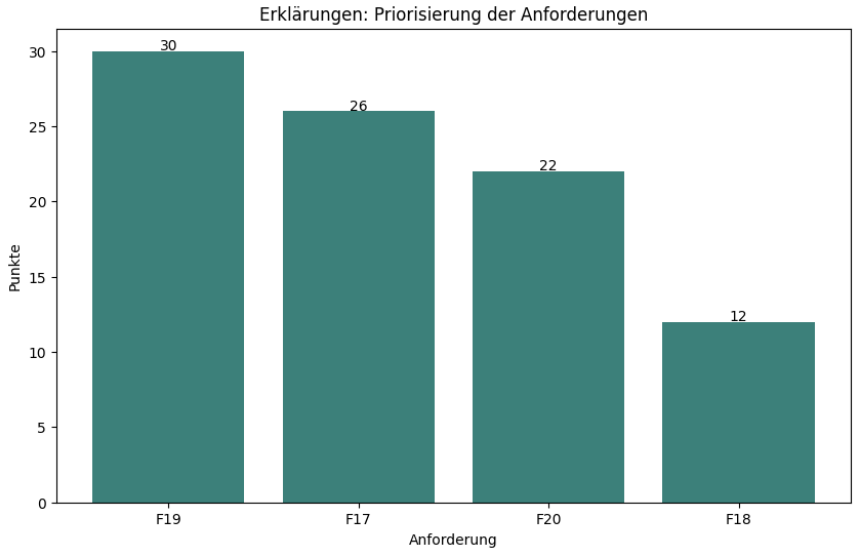


Abbildung 5.11: Priorisierung der Aussagen von Erklärungsbedarf nach der von den Teilnehmern empfundenen Wichtigkeit

[R02] Das System sollte logische Fehler in der Adresse erkennen.
Es werden nicht plausible Kombinationen von Postleitzahl und Stadt, sowie ungültige Hausnummern einer Straße erkannt.

[R03] Gefundene Adressen werden visuell bestätigt.
Ein grüner Haken markiert erfolgreich gefundene Adressen.

[R04] Das System weist vor der Erstellung der Tour auf Adressenduplikate hin.
Ein Dialog wird geöffnet, welcher die Duplikate benennt und die Option des Löschens dieser Duplikate anbietet.

[R05] Es gibt einen Help Button, welcher die Funktionsweise der sprachgesteuerten Adresseneingabe erklärt.
Der Help Button ist über die Topbar dauerhaft anklickbar.

[R06] Adressen lassen sich auch manuell hinzufügen.
Jede Adresse hat einen Edit Button, mit welchem die Adresse über ein Textfeld angepast werden kann.

[R07] Der Nutzer sieht, welche Adressinformationen einer unvollständigen Adresse benötigt werden.
Es werden mit einem roten Ausrufezeichen unvollständige Adressen markiert.

[R08] Bereits gefundene Adressen lassen sich mit einer Swipebewegung wieder entfernen.

Mit einer Swipebewegung nach links wird die Adresse aus der Tourliste gelöscht.

[R09] Die Spracheingabe erkennt, wenn der Nutzer Sprachen abseits von Deutsch spricht.
Die Spracheingabe übersetzt ins Deutsche oder zeigt eine Fehlermeldung, wenn die Sprache von der Spracherkennung nicht unterstützt wird.

[R10] Das Gerät kann während der gesamten sprachgesteuerten Adresseneingabe aus der Hand gelegt werden.
Zwischen dem Klick „Starte Spracheingabe“ und „Stoppe Spracheingabe“ muss keine Interaktion mit dem Gerät durchgeführt werden.

[R11] Das System greift auf frühere Eingaben des Nutzers zurück, um unvollständige Adressen potentiell vervollständigen zu können.
Eine Datenbank speichert alle Adressen, sobald der Nutzer auf „Starte Tour“ klickt.

[R12] Das System spielt einen hörbaren Ton bei jeder hinzugefügten Adresse ab.
Bei einer erfolgreichen Hinzufügung wird ein positiv assoziierter Ton und bei fehlgeschlagener Hinzufügung analog ein negativ assoziierter Ton abgespielt.

[R13] Das System erklärt bei der erstmaligen Benutzung der sprachgesteurten Adresseneingabe die Funktionsschritte.
Ein Dialog erscheint, welcher die einzelnen Schritte verständlich erklärt.

[R14] Bereits gefundene Adressen sind bereits während der Spracheingabe modifizierbar.
siehe [R06]

[R15] Das System erklärt dem Nutzer, dass er mit einer KI kommuniziert.
Der Button, der die Spracheingabe startet, hat als Icon zusätzlich einen Roboter. Ebenso wird in den Hilfsdialogen darauf hingewiesen.

[R16] Die Spracheingabe kann auch verbal gestoppt werden.
Die Spracheingabe bricht ab, wenn der Nutzer „Stoppe Spracheingabe“ sagt.

[R17] Der Nutzer erhält die Möglichkeit, jederzeit den Status der Internetverbindung zu sehen.
Auf dem Display zeigt ein vierstufiger Verbindungsbalken die Qualität der Internetverbindung an.

[R18] Das System kann die gesprochenen Worte in Form von Untertiteln visuell wiedergeben.
Die Untertitel erscheinen auf dem Display, können aber in den Einstellungen auch deaktiviert werden. Standardmäßig sind sie deaktiviert.

Kapitel 6

Implementierung

In diesem Kapitel wird der Prototyp vorgestellt, welcher die Implementierung der in 5.4 erhobenen Anforderungen demonstriert. Dieser ist in die Android App *Nunav Courier*(siehe 5.1) integriert [17]. Auszugsweise werden einige der umgesetzten Anforderungen aus 5.4 vorgestellt.

6.1 Datenfluss

Das in 5.1 vorgestellte Feature ermöglicht es dem Nutzer, durch Spracheingabe neue Adressen zur geplanten Tour hinzuzufügen. Wie in Abbildung 6.1 dargestellt, durchläuft das System mehrere interne Schritte, um dies zu realisieren. Zunächst wird die Spracheingabe des Nutzers mithilfe des *Google SpeechRecognizers* [15] in Text umgewandelt. Diese Umwandlung erleichtert die Interaktion mit der KI und ermöglicht zugleich eine Bewertung der Genauigkeit der Spracherkennung.

Das Ergebnis der *Speech-to-Text* Umwandlung wird anschließend von einer KI analysiert, um die im Text genannten Adressen zu identifizieren. Der Prototyp nutzt hierfür eine Schnittstelle zu ChatGPT [22]. Die KI liefert als Ergebnis ein JSON-Objekt (JavaScript Object Notation), das alle erkannten Adressen umfasst. Im folgenden Schritt müssen diesen Adressen reale Geokoordinaten zugeordnet werden, um eine präzise Navigationsroute planen zu können. Dazu wird der Open-Source-Geocoder *Photon* von Komoot [20] verwendet. Schließlich werden die mit Geokoordinaten versehenen Adressen in einer Datenbank gespeichert, um sie für die Tourplanung zu nutzen.

6.2 User Interface

Abbildung 6.2a zeigt das User Interface, wie es dem Nutzer präsentiert wird, wenn noch keine Adressen zur Tour hinzugefügt wurden. Am oberen Rand des Bildschirms befindet sich die Topbar (siehe Abbildung A.2b), die dem

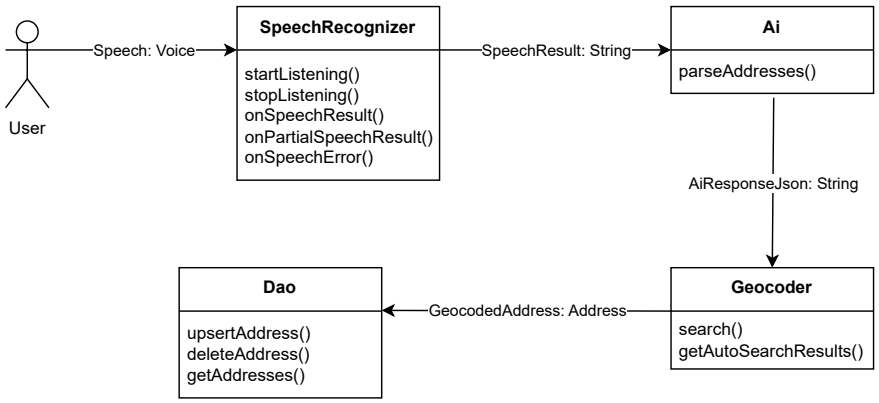


Abbildung 6.1: Der Datenfluss vom gesprochenen Wort des Users hin zur abgespeicherten Adresse

Nutzer ermöglicht, das Feature zu verlassen, Hilfe anzufordern oder das Einstellungsmenü zu öffnen.

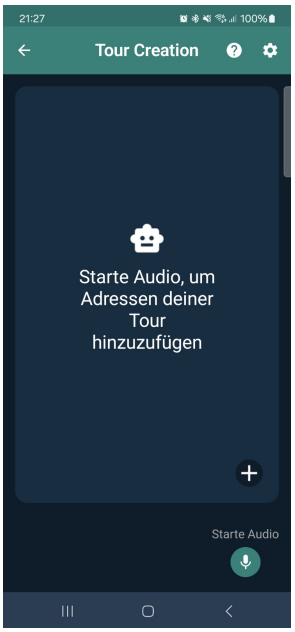
Der Hauptbereich des Bildschirms wird von der leeren Adressenliste dominiert. Anstelle der fehlenden Einträge wird ein Informationstext angezeigt, der die Funktionsweise des Features erklärt (siehe Abbildung A.2a). Hier besteht auch die Möglichkeit, Adressen manuell hinzuzufügen, indem der Nutzer den Plus-Button betätigt (**[R06]**).

Am unteren Rand des Bildschirms befindet sich die Bedienfläche für die Spracheingabe. Ein Mikrofon-Button ermöglicht es dem Nutzer, die Spracheingabe zu starten. Das gesamte User Interface ist bewusst minimalistisch gestaltet, um den Nutzer nicht mit der Funktionalität zu überfordern. Abbildung 6.2b zeigt das Interface während einer aktiven Spracheingabe. Ein pulsierendes rotes Mikrofonsymbol auf der linken Seite und sich typisch für Audioaufnahmen bewegende Balken signalisieren dem Nutzer, dass seine Worte momentan aufgenommen werden.

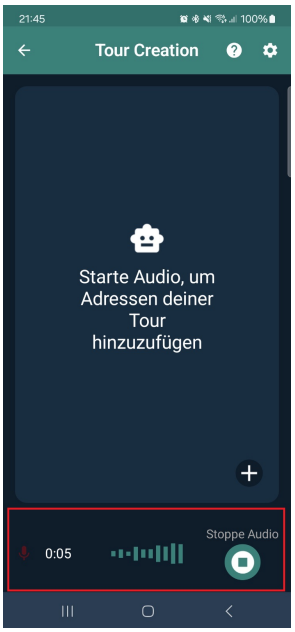
Der Fokus des Interfaces liegt auf dem zentral platzierten Informationstext und dem beschrifteten Mikrofon-Button, um die Aufmerksamkeit des Nutzers auf das Starten der Spracheingabe zu lenken.

Sobald die erste Adresse erkannt wird, verschwindet der Informationstext, da der Nutzer die Funktionalität der Spracheingabe erfolgreich verstanden und genutzt hat. Abbildung 6.3a zeigt die bereits erkannten Adressen. Jede Adresse ist dabei mit einem Badge versehen, der den jeweiligen Status der Adresse anzeigt (siehe Abbildung 6.3).

Erst nach dem Hinzufügen mindestens einer Adresse erscheint ein Button zur finalen Erstellung der Tour. Mit diesem Button kann die Navigation der Route, bestehend aus den erkannten Adressen, gestartet werden. Zusätzlich wird die Anzahl der bereits gefundenen Adressen angezeigt, um dem Nutzer



(a) Spracheingabe ist deaktiviert



(b) Spracheingabe ist aktiv

Abbildung 6.2: Startbildschirm des Prototypen

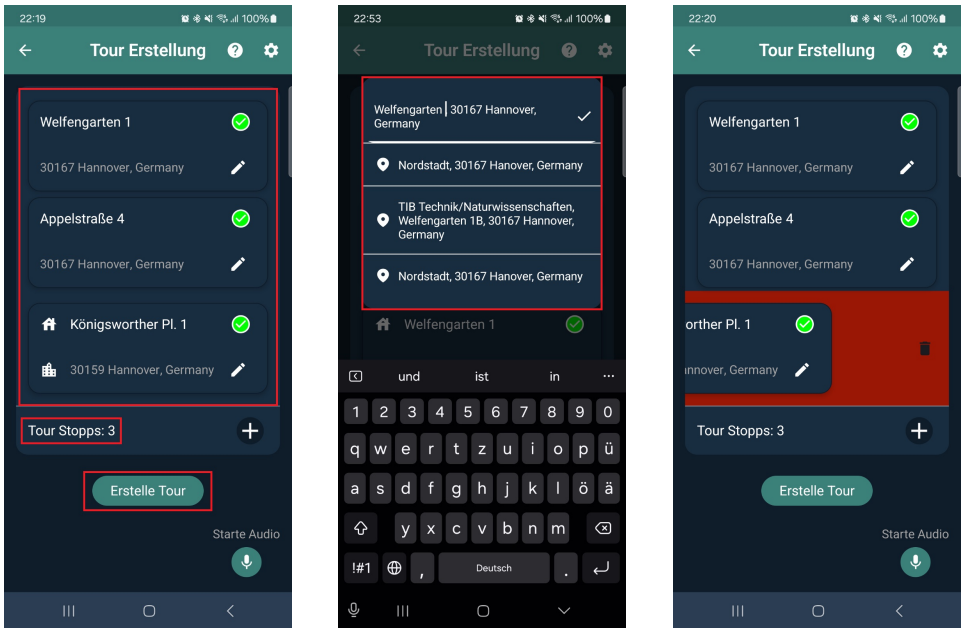
einen Überblick darüber zu geben, wie viele Stopps er schon hinzugefügt hat.

Darüber hinaus können erkannte Adressen bearbeitet werden (**[R14]**), wie in Abbildung 6.3b dargestellt. Dabei werden dem Nutzer die nächstliegenden Ergebnisse vorgeschlagen. Ebenso lassen sich bereits gefundene Adressen mit einer Swipe-Geste wieder entfernen (**[R08]**), wie in Abbildung 6.3c gezeigt.

Diese Handhabung ermöglicht es dem Nutzer, das Gerät während der gesamten Spracheingabe aus der Hand zu legen (**[R10]**), was eine besonders relevante Anforderung für Kurierfahrer darstellt. Für diese Zielgruppe ist es entscheidend, dass die Bedienung möglichst freihändig und effizient erfolgt, um die Sicherheit und den Arbeitsfluss nicht zu beeinträchtigen. Dies stellt einen wesentlichen Vorteil in der täglichen Nutzung dar, insbesondere unter den dynamischen und oft stressigen Bedingungen des Kurierdienstes.

Darüber hinaus bietet das manuelle Editieren von Adressen eine alternative Eingabemethode, die auf traditionellen, bewährten Praktiken basiert. Diese Funktion stellt sicher, dass Nutzer, die möglicherweise weniger vertraut oder weniger komfortabel mit der Spracheingabe sind, weiterhin in der Lage sind, ihre Ziele zuverlässig in die Liste der Tourstopps einzutragen.

Die Kombination beider Eingabemethoden – der innovativen Spracheingabe und der klassischen manuellen Adressbearbeitung – unterstützt nicht nur eine flexible Bedienung, sondern trägt auch zur Vertrauensbildung gegen-



(a) List der bereits gefundenen Adressen

(b) Manuelles Editieren von Adressen

(c) Swipe-To-Delete von gefundenen Adressen

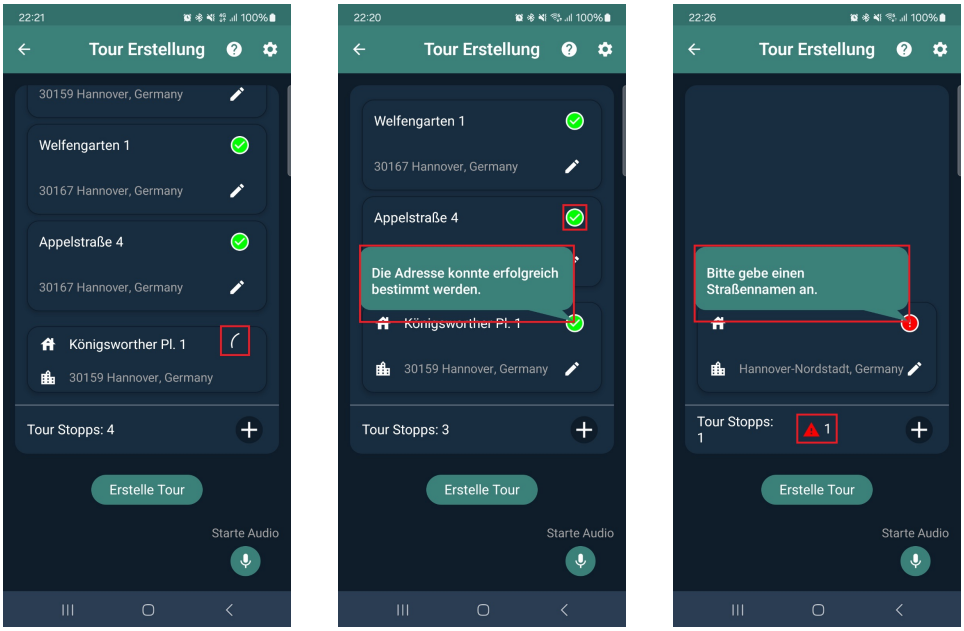
Abbildung 6.3: Adressliste mit bereits gefunden Adressen

über der neuen Technologie bei. Indem bekannte Mechanismen beibehalten und nahtlos mit neuen Funktionen integriert werden, wird die Akzeptanz der Technologie bei einer breiten Nutzergruppe gefördert. Dies ist besonders relevant, da der Übergang zu neuen Technologien häufig mit Widerständen oder Bedenken hinsichtlich der Zuverlässigkeit und Benutzerfreundlichkeit verbunden ist. Die Dualität der Eingabemöglichkeiten bietet hier eine Brücke zwischen traditioneller Bedienung und innovativer Technologie, was letztlich die Barriere für die Einführung und Nutzung der neuen Funktionalitäten verringert. Diese strategische Gestaltung des User Interfaces verfolgt das Ziel, neue Technologien unter Berücksichtigung bestehender Nutzungsgewohnheiten einzuführen, um die Akzeptanz und das Vertrauen der Nutzer nachhaltig zu fördern.

6.3 Erklärungen

Das User Interface der Applikation wurde so gestaltet, dass sie eine intuitive Navigation ermöglicht und dem Nutzer dabei hilft, die verschiedenen Funktionen der Anwendung schnell zu verstehen. Ein zentrales Element dieser Gestaltung sind die eingebauten Erläuterungen und Hilfetexte, die den Nutzer bei der Adressverarbeitung und Tourplanung unterstützen. Wie in Abbildung 6.4 gezeigt, bietet die Applikation visuelle Rückmeldungen

zum Status der Adressverarbeitung. Jede identifizierte Adresse zeigt ihren aktuellen Bearbeitungsstatus an, sodass der Nutzer den Fortschritt der Geolokalisierung in Echtzeit verfolgen kann. Abbildung 6.4a veranschaulicht den Vorgang der Geolokalisierung einer Adresse. Sobald eine Adresse erfolgreich lokalisiert wurde, wie in Abbildung 6.4b dargestellt, kann sie vom System zur Routenplanung verwendet werden (**[R03]**). Sollte die Lokalisierung einer Adresse fehlschlagen, wird dies dem Nutzer durch eine deutliche Fehlermeldung angezeigt, wie in Abbildung 6.4c zu sehen ist (**[R07]**). Zusätzlich erscheint in der Adressliste ein Warnsymbol, das die Anzahl der aktuell fehlgeschlagenen Lokalisierungsversuche anzeigt. Der Nutzer kann dieses Warnsymbol anklicken, um die fehlgeschlagenen Adressen manuell zu editieren. Zudem ist der Informationstext zu den einzelnen Zuständen jederzeit für den Nutzer zugänglich, indem er einfach auf das entsprechende Statussymbol klickt.



(a) Adresse wird geparsed

(b) Adresse wurde erfolgreich erkannt

(c) Erkennung ist fehlgeschlagen

Abbildung 6.4: Status der identifizierten Adressen

Falls der Nutzer auf Schwierigkeiten stößt oder weitere Informationen benötigt, steht ihm ein Hilfetext zur Verfügung, der durch einen Klick auf das entsprechende Symbol in der Topbar (siehe Abbildung A.2b) aufgerufen werden kann. Dieser Hilfetext ist in Abbildung 6.5 abgebildet und enthält detaillierte Anweisungen und Erklärungen zu den möglichen Problemen, die während der Adressverarbeitung auftreten können, sowie Tipps zur Fehlerbehebung (**[R05]**). Diese Funktion ist besonders wertvoll,

um sicherzustellen, dass der Nutzer auch ohne externe Unterstützung in der Lage ist, auftretende Probleme zu lösen. Durch diese integrierten Erläuterungen und Hilfetexte wird der gesamte Prozess der Tourstopp-Planung und Adressverarbeitung deutlich nutzerfreundlicher gestaltet. Sie fördern nicht nur das Verständnis der zugrunde liegenden Mechanismen, sondern reduzieren auch die Notwendigkeit für externe Unterstützung, was zu einer effizienteren Nutzung der Anwendung führt.

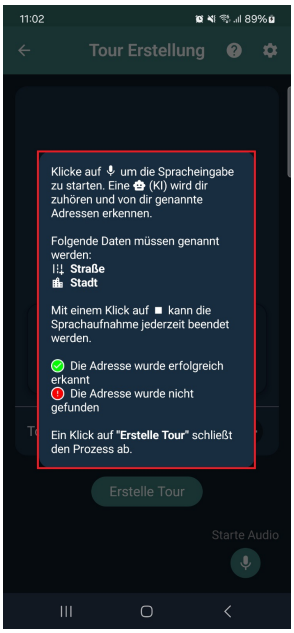


Abbildung 6.5: Informationstext, wenn der Nutzer auf Hilfe klickt

Kapitel 7

Zusammenfassung und Ausblick

In diesem Kapitel werden die Ergebnisse der Arbeit zusammengefasst. Anschließend werden diese auf ihre Validität hin untersucht. Ferner wird ein Ausblick auf mögliche weiterführende Arbeiten gegeben, die die in dieser Arbeit gewonnenen Erkenntnisse vertiefen und weiterentwickeln könnten.

7.1 Zusammenfassung

Die durchgeführte Literaturrecherche verdeutlicht die wachsende Bedeutung von Erklärungen in modernen Systemen, insbesondere angesichts ihrer zunehmenden Komplexität. Diese Komplexität erschwert den Nutzern den Zugang zu den Systemen, weshalb Erklärungen unerlässlich sind, um die Nutzbarkeit und das Verständnis zu gewährleisten. Eine zentrale Erkenntnis der Recherche ist die Notwendigkeit, den Erklärungsbedarf in solchen Systemen gesondert zu betrachten. Hieraus ergibt sich die Bedeutung, Erklärungen nicht erst im Nachhinein zu integrieren, sondern diesen Bedarf bereits frühzeitig im Projektverlauf zu identifizieren und systematisch in den Erhebungs- und Entwicklungsprozess einzubinden.

Das vorgestellte Konzept adressiert diesen Erklärungsbedarf durch einen iterativen Ansatz, der eine enge und kontinuierliche Kommunikation mit den relevanten Stakeholdern erfordert. Diese iterative Vorgehensweise ermöglicht es, den Erklärungsbedarf fortlaufend zu überprüfen und anzupassen, wodurch die Entwicklung eines Systems erfolgt, das den individuellen Bedürfnissen der Nutzer besser gerecht wird.

Die durchgeführte Nutzerstudie untermauert die Notwendigkeit einer differenzierten Betrachtung des Erklärungsbedarfs. Sie zeigt deutlich, dass die Bedürfnisse der Nutzer in Bezug auf Erklärungen sehr unterschiedlich sein können. Eine nutzerzentrierte Herangehensweise ist daher essenziell, um die Zugänglichkeit des Systems für unterschiedliche Nutzergruppen sicher-

zustellen. Diese Ausrichtung trägt dazu bei, dass das System verständlicher und somit effektiver genutzt werden kann.

Die entwickelten Leitlinien bieten Entwicklern und Designern eine wertvolle Orientierungshilfe, um die Erklärbarkeit in Systemen gezielt zu verbessern. Sie liefern praxisnahe Richtlinien, wie Erklärbarkeitsanforderungen systematisch erhoben und in den Entwicklungsprozess integriert werden können. Der entwickelte Prototyp dient hierbei als exemplarisches Modell, das aufzeigt, wie diese Anforderungen in der Praxis umgesetzt werden können. Er zeigt, dass durch die frühzeitige und systematische Berücksichtigung des Erklärungsbedarfs ein System entsteht, das den Nutzer in den Mittelpunkt stellt und dadurch zugänglicher und benutzerfreundlicher wird.

7.2 Validität

Die Validität ist ein zentraler Aspekt in der empirischen Forschung. Die kritische Überprüfung auf Validitätseinschränkungen dieser Arbeit orientiert sich dabei an dem Werk von Wohlin et al. [31]. In dieser werden vier spezifische *Threads of Validity* identifizert: *„Conclusion Validity“*, *„Internal Validity“*, *„Construct Validity“* und *„External Validity“*.

7.2.1 Conclusion Validity

Die Studie hat mit neun Teilnehmern eine eher geringe Anzahl an Partitionen. Dies liegt unter anderem an der schwer erreichbaren Nutzerbasis der App. Zu Rate wurden daher Fachleute aus dem Gebiet der Entwicklung von Navigationssoftware, sowie ehemalige Kurierfahrer, gezogen. Durch die geringe Teilnehmerzahl steigt die Wahrscheinlichkeit, dass Ausreißer das Gesamtergebnis unverhältnismäßig stark beeinflussen. Zudem wird es schwieriger, statistisch signifikante Ergebnisse zu erzielen. Dem wurde durch die Betrachtung des Medians bei der Messung der Zustimmung entgegengewirkt. Trotzdem wäre es sinnvoll, die Erhebung mit einer größeren Teilnehmerzahl zu wiederholen, sollte sich die Erreichbarkeit der Nutzerbasis verbessern.

7.2.2 Internal Validity

Das Experteninterview wurde mit zwei ausgewählten Experten auf dem Gebiet der Softwareentwicklung durchgeführt. Die Ergebnisse dieses Interviews sind stark abhängig von den subjektiven Empfinden der Experten. Andere Personen, die im gleichen Maße auf dem Fachgebiet bewandert sind, könnten gänzlich andere Meinungen vertreten und durch das Interview vorgegebene Themen anders interpretieren. Es wurde daher darauf geachtet, zwei Teilnehmer mit langjähriger Erfahrung in der Softwareentwicklung zu wählen.

Zudem sollten Teilnehmer der Studie die vorgeschlagenen Anforderungen sowohl bewerten, als auch priorisieren. Dadurch könnten semantische Ungereimtheiten entstehen, wenn beispielsweise einer Aussage mehr zugestimmt wird als einer anderen, diese dann aber wiederum höher vom Teilnehmer priorisiert wird. Diese Herangehensweise ermöglicht aber auch ein umfassenderes Bild der Meinungen der Teilnehmer. Während die Zustimmung eine generelle Akzeptanz oder Ablehnung einer Anforderung widerspiegelt, zeigt die Priorisierung, welche Anforderungen die Teilnehmer als besonders wichtig oder dringlich empfinden. Diese Dualität ermöglicht es, nicht nur zu verstehen, was die Teilnehmer für eine Meinung vertreten, sondern auch, welche Punkte sie in der Praxis als essentiell betrachten. Unterschiede zwischen Zustimmung und Priorisierung haben außerdem wertvolle Informationen zur Klärung von Missverständnissen geboten. Die gesammelten Anforderungen konnten so einer genauen Überprüfung unterzogen werden. In der Folge wurden zwei Anforderungen (F13 & F18), welche sich aus dem Interview entwickelt haben, wieder verworfen.

7.2.3 Construct Validity

Als Bewertungsskala für die Fragebögen der Umfrage wurde eine Likert-Skala gewählt. Diese hat den Vorteil, dass Teilnehmer Abstufungen in ihrer Bewertung vornehmen können. Es wurde dabei darauf geachtet, dass die Skala klare und leicht verständliche Formulierungen verwendet. So wird den Teilnehmern eine klare Vorstellung davon verschafft, wie stark sie eine Aussage unterstützen oder ablehnen, was die Genauigkeit der Antworten erhöht. Durch die Verwendung von fünf Antwortoptionen, die von starker Ablehnung bis zu starker Zustimmung reichen, ermöglicht die Skala eine feine Differenzierung der Meinungen der Teilnehmer. Zudem wurde darauf geachtet, dass die Skala ausgewogen ist. Sie enthält gleich viele positive als auch negative Antwortmöglichkeiten.

Die Repräsentativität der Stichprobe ist gefährdet durch die Tatsache, dass die Teilnehmer aus einem sehr spezifischen Kreis gewählt wurden. Es stellt sich die Frage, ob diese Personen die relevanten Konstrukte genauso verstehen und interpretieren wie die gesamte Zielgruppe der App. Unterschiedliches Hintergrundwissen und Erfahrungen könnten dazu führen, dass die Teilnehmer die Anforderungen unterschiedlich bewerten, was die Konstruktivität beeinträchtigen könnte. Dem wurde entgegengewirkt, indem zu Beginn der Umfrage der zu entwickelnde Prototyp beschrieben wurde. Zudem wurde allen Teilnehmern die Aufgabe gestellt, sich in die Lage eines Kurierfahrers zu versetzen, was ihnen durch den beruflichen Kontext möglich war.

7.2.4 External Validity

Eine Herausforderung für die externe Validität stellt die Zusammensetzung der Stichprobe dar. Die Teilnehmer der Studie stammen zu einem großen Teil aus einer spezifischen Gruppe. Das sehr ähnliche berufliche Umfeld könnte die Generalisierbarkeit der Ergebnisse einschränken. Es ist möglich, dass die teilnehmenden Personen Ansichten, Erfahrungen und Hintergründe teilen, die nicht repräsentativ für die gesamte Zielgruppe sind. Dadurch könnten die Ergebnisse nicht ohne Weiteres auf die gesamte Nutzergruppe übertragen werden.

Um die externe Validität zu erhöhen, könnte die Stichprobe diversifiziert werden für die gesamte Zielgruppe der Navigations-App. Dies könnte durch die Einbeziehung von Teilnehmern unterschiedlicher Altersgruppen, Berufe und technischer Erfahrungen erreicht werden. Zudem wäre es wichtig, die Studie in verschiedenen Kontexten durchzuführen, um zu überprüfen, ob die Ergebnisse auch unter anderen Bedingungen konsistent sind.

Darüber hinaus spielt die Auswahl der Untersuchungsbedingungen eine wichtige Rolle. Wenn die Studie unter sehr spezifischen oder künstlichen Bedingungen durchgeführt wird, könnte dies die Übertragbarkeit auf reale Nutzungsszenarien einschränken. Es war den Teilnehmern freigestellt, wann und unter welchen Bedingungen sie an der digitalen Umfrage teilnehmen. Es besteht die Möglichkeit, dass einzelne Teilnehmer in anderen Bedingungen anders entschieden hätten. Dem wurde entgegen gewirkt, indem allen Teilnehmern eine erwartete Bearbeitungsdauer von etwa zehn Minuten mitgeteilt wurde. Außerdem konnte die Umfrage über einen Zeitraum von 14 Tagen bearbeitet werden.

7.3 Ausblick

Um den entwickelten Prototypen weiter zu optimieren und dessen praktischen Nutzen zu evaluieren, bietet es sich an, eine umfassende Studie zur Messung der Effizienz und Effektivität des identifizierten Erklärungsbedarfs durchzuführen. Diese Studie könnte wertvolle Erkenntnisse darüber liefern, welche spezifischen Aspekte der Erklärbarkeit nicht nur in dem entwickelten Prototypen, sondern auch in anderen, allgemeineren Systemen von Bedeutung sind und sich gegebenenfalls darauf übertragen lassen. Dabei könnten Unterschiede in der Wahrnehmung und im Verständnis verschiedener Nutzergruppen hervortreten, die für die Weiterentwicklung und Anpassung des Modells relevant sind.

Ein weiterer wichtiger Aspekt wäre die Entwicklung und der Einsatz von Personas. Durch die Erstellung detaillierter und differenzierter Nutzerprofile könnte der individuelle Erklärungsbedarf besser identifiziert und gezielt adressiert werden. Personas würden es ermöglichen, die Erklärungsbedarfe verschiedener Zielgruppen systematisch zu erfassen und zu analysieren. Dies

könnte die Personalisierung und somit die Wirksamkeit der Erklärungen erheblich verbessern, da die Erklärungen auf die spezifischen Bedürfnisse und Wissensstände der unterschiedlichen Nutzergruppen abgestimmt werden könnten.

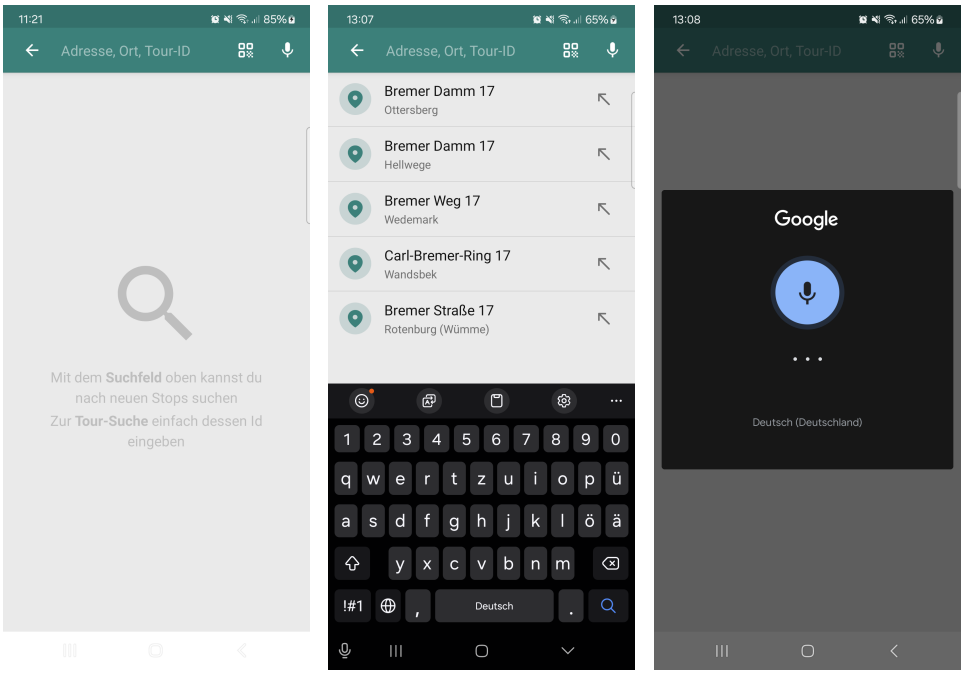
Zusätzlich zur theoretischen Untersuchung ist eine Testphase in der Praxis unerlässlich, um das entwickelte Modell in einem realen wirtschaftlichen Umfeld zu validieren. Diese Testphase könnte in Form von Pilotprojekten mit mehreren Kurierfahrern durchgeführt werden, um das Modell unter realen Bedingungen zu erproben. Hierdurch könnten praxisnahe Erkenntnisse über die Anwendbarkeit, Nutzerakzeptanz und den tatsächlichen Mehrwert des Modells gewonnen werden.

Letztlich bietet diese integrative Vorgehensweise die Möglichkeit, das in dieser Arbeit entwickelte theoretische Modell durch iterative Anpassungen weiter zu verfeinern und in der Praxis zu etablieren. Dies würde nicht nur den wissenschaftlichen Diskurs bereichern, sondern auch einen praktischen Nutzen für die Wirtschaft und weitere Anwendungsbereiche bieten.

Anhang A

Screenshots

Hier werden alle in der Arbeit referenzierten Screenshots aufgeführt.

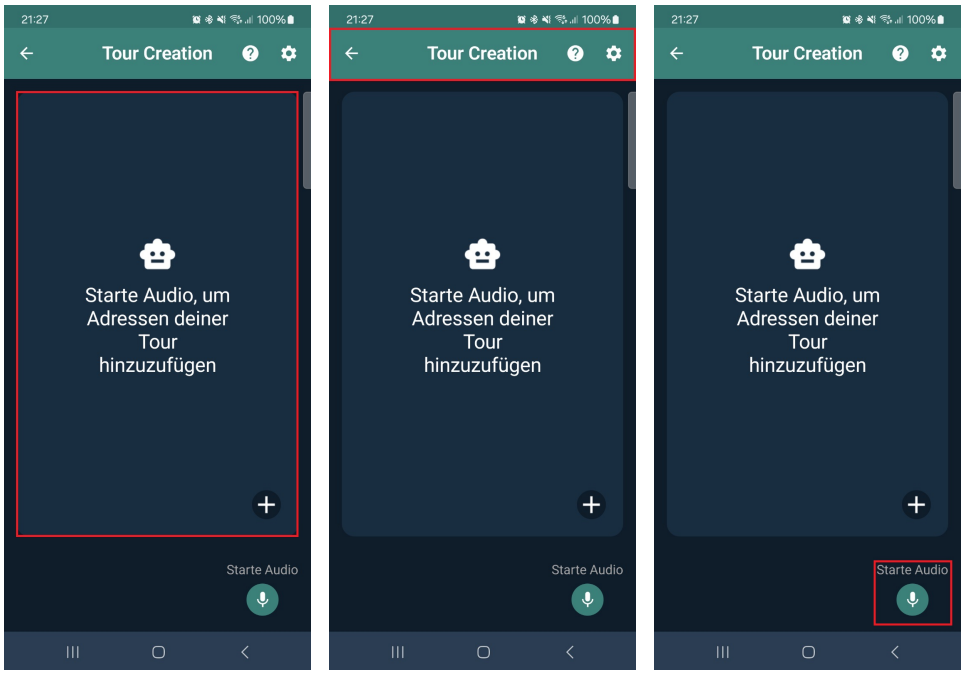


(a) Nunav Courier - leeres Suchfeld

(b) Nunav Courier - Textfeldeingabe

(c) Nunav Courier - Spracheingabe

Abbildung A.1: Vergleich der verschiedenen Eingabemöglichkeiten in Nunav Courier



(a) Platzhalter bei Leere Adressliste

(b) Topbar-Elemente des Prototypen

(c) Knopf zum Starten der Spracheingabe

Abbildung A.2: UI-Elemente des Startbildschirms des Prototypen

Anhang B

Studie zur Erhebung und Priorisierung von Anforderungen

Willkommen. Vielen Dank dafür, dass du dir die Zeit nimmst, mich durch deine Teilnahme bei dieser Umfrage bei meiner Bachelorarbeit zu unterstützen.

Es geht um die Erhebung und Priorisierung von Anforderungen an einen von mir entwickelten Prototypen in Nunav Courier.

Feature-Beschreibung

Der Prototyp soll eine sprachgesteuerte Adresseneingabe für die Zusammenstellung einer Tour ermöglichen. Dafür kann der User in freier Sprache mit der App kommunizieren. Eine Ki analysiert die gesprochenen Worte und erkennt, wenn der User eine Adresse zu der Tour hinzufügen möchte. Auf diese Weise entsteht eine Liste von Adressen, die anschließend vom User für die Erstellung einer optimierten Tour verwendet wird.

Du sollst in dieser Umfrage dich in die Rolle des Users versetzen und deine beruflichen Erfahrungen und Vorstellungen in die Bewertung mit einfließen lassen. Mit den erhobenen Daten werden anschließend Anforderungen formal erhoben und priorisiert.

Teil A: Interaktion

A1.

	Stimme überhaupt nicht zu	Stimme nicht zu	Stimme weder zu noch lehne ich ab	Stimme zu	Stimme voll und ganz zu
Die bisher erkannten Adressen sollten mit einer Swipe-Bewegung wieder entfernbar sein.	☐	☐	☐	☐	☐
Mir ist wichtig, dass ich Adressen auch manuell noch abseits der Spracheingabe hinzufügen kann.	☐	☐	☐	☐	☐
Ich sollte das Smartphone für den gesamten Bedienungsprozess der Adresseneingabe aus der Hand legen können.	☐	☐	☐	☐	☐
Bereits gefundene Adressen sollten schon während der sprachgesteuerten Eingabe modifizierbar sein.	☐	☐	☐	☐	☐

A2. **Bitte priorisiere die auf dieser Seite genannten Anforderungen nach deiner empfundenen Wichtigkeit. Die Reihenfolge der Priorisierung gibt an, welche Anforderungen an das Feature für dich relevanter sind im Vergleich untereinander. Diese Priorisierung kann unabhängig von den oben vergebenen Scores stattfinden.**

Oben steht die relevanteste Anforderung - Unten die am wenigsten relevante Anforderung

Die bisher gefundenen und in einer Liste gespeicherten Adressen sollten mit einer Swipe-Bewegung wieder entfernbar sein. ☐

Mir ist wichtig, dass ich Adressen auch manuell noch abseits der Spracheingabe hinzufügen kann. ☐

Ich sollte das Smartphone für den gesamten Bedienungsprozess der Adresseneingabe aus der Hand legen können. ☐

Bereits gefundene Adressen sollten schon während der sprachgesteuerten Eingabe modifizierbar sein. ☐

Teil B: Spracheingabe

B1.

	Stimme überhaupt nicht zu	Stimme nicht zu	Stimme weder zu noch lehne ich ab	Stimme zu	Stimme voll und ganz zu
Die Spracheingabe sollte Sprachen abseits von Deutsch verstehen, auch wenn nur deutsche Adressen eingegeben werden.	☐	☐	☐	☐	☐
Die Spracheingabe sollte aktiv Feedback geben, wenn sie meine Worte nicht verstanden hat.	☐	☐	☐	☐	☐
Die App sollte meine gesprochenen Worte in Form von Untertiteln visuell wiedergeben.	☐	☐	☐	☐	☐
Die Spracheingabe sollte auch verbal gestoppt werden können, z.B. durch ein gesprochenes "Stop".	☐	☐	☐	☐	☐

B2. **Bitte priorisiere die auf dieser Seite genannten Anforderungen nach deiner empfundenen Wichtigkeit. Die Reihenfolge der Priorisierung gibt an, welche Anforderungen an das Feature für dich relevanter sind im Vergleich untereinander. Diese Priorisierung kann unabhängig von den oben vergebenen Scores stattfinden.**

Oben steht die relevanteste Anforderung - Unten die am wenigsten relevante Anforderung

Die Spracheingabe sollte Sprachen abseits von Deutsch verstehen, auch wenn nur deutsche Adressen eingegeben werden. ☐

Die Spracheingabe sollte aktiv Feedback geben, wenn sie meine Worte nicht verstanden hat. ☐

Die App sollte meine gesprochenen Worte in Form von Untertiteln visuell wiedergeben. ☐

Die Spracheingabe sollte auch verbal gestoppt werden können, z.B. durch ein gesprochenes "Stop". ☐

Teil C: Hilfestellung

C1.

	Stimme überhaupt nicht zu	Stimme nicht zu	Stimme weder zu noch lehne ich ab	Stimme zu	Stimme voll und ganz zu
Die App sollte bei der erstmaligen Benutzung der sprachgesteurten Adresseneingabe die einzelnen Arbeitsschritte ausführlich erklären.	☐	☐	☐	☐	☐
Mir ist wichtig, dass die App eine gefundene Adresse z.B. durch einen sichtbaren grünen Haken bestätigt.	☐	☐	☐	☐	☐
Die App sollte einen hörbaren Ton bei jeder hinzugefügten Adresse abspielen.	☐	☐	☐	☐	☐
Die App sollte einen "Help Button" bereit stellen, welcher die sprachgesteuerte Adresseneingabe erklärt.	☐	☐	☐	☐	☐

C2. **Bitte priorisiere die auf dieser Seite genannten Anforderungen nach deiner empfundenen Wichtigkeit. Die Reihenfolge der Priorisierung gibt an, welche Anforderungen an das Feature für dich relevanter sind im Vergleich untereinander. Diese Priorisierung kann unabhängig von den oben vergebenen Scores stattfinden.**

Oben steht die relevanteste Anforderung - Unten die am wenigsten relevante Anforderung

Die App sollte bei der erstmaligen Benutzung der sprachgesteurten Adresseneingabe die einzelnen Arbeitsschritte ausführlich erklären. ☐

Mir ist wichtig, dass die App eine gefundene Adresse z.B. durch einen sichtbaren grünen Haken bestätigt. ☐

Die App sollte einen hörbaren Ton bei jeder hinzugefügten Adresse abspielen. ☐

Die App sollte einen "Help Button" bereit stellen, welcher die sprachgesteuerte Adresseneingabe erklärt. ☐

Teil D: Transparenz

D1.

	Stimme überhaupt nicht zu	Stimme nicht zu	Stimme weder zu noch lehne ich ab	Stimme zu	Stimme voll und ganz zu
Die App soll durch einen verständlichen Wert anzeigen, wie gut die Spracheingabe meine Worte verstanden hat.	☐	☐	☐	☐	☐
Die App sollte vor der finalen Tourerstellung auf Adressen-Duplikate hinweisen.	☐	☐	☐	☐	☐
Ich möchte zu jeder Zeit wissen, wie gut gerade meine Internetverbindung ist, um die Adresseneingabe vornehmen zu können.	☐	☐	☐	☐	☐
Die App sollte mir erklären, dass ich mit einer KI kommuniziere.	☐	☐	☐	☐	☐

D2. **Bitte priorisiere die auf dieser Seite genannten Anforderungen nach deiner empfundenen Wichtigkeit. Die Reihenfolge der Priorisierung gibt an, welche Anforderungen an das Feature für dich relevanter sind im Vergleich untereinander. Diese Priorisierung kann unabhängig von den oben vergebenen Scores stattfinden.**

Oben steht die relevanteste Anforderung - Unten die am wenigsten relevante Anforderung

Die App soll durch einen verständlichen Wert anzeigen, wie gut die Spracheingabe meine Worte verstanden hat. ☐

Die App sollte vor der finalen Tourerstellung auf Adressen-Duplikate hinweisen. ☐

Ich möchte zu jeder Zeit wissen, wie gut gerade meine Internetverbindung ist, um die Adresseneingabe vornehmen zu können. ☐

Die App sollte mir erklären, dass ich mit einer KI kommuniziere. ☐

Teil E: Erklärungen

E1.

	Stimme überhaupt nicht zu	Stimme nicht zu	Stimme weder zu noch lehne ich ab	Stimme zu	Stimme voll und ganz zu
Die sprachgesteuerte Adresseneingabe sollte erklären, welche Adressinformationen zwingend benötigt werden.	☐	☐	☐	☐	☐
Nach jeder erkannten Adresse sollte die App die gefunde Adresse verbal wiederholen.	☐	☐	☐	☐	☐
Das System sollte in der Lage sein, den Kontext der Adressangaben zu verstehen, um logische Fehler zu minimieren (z.B. unplausible Kombinationen von Stadt und Postleitzahl).	☐	☐	☐	☐	☐
Das System greift auf frühere Eingaben des Benutzers zurück, um die Adresseneingabe zu vereinfachen.	☐	☐	☐	☐	☐

E2. **Bitte priorisiere die auf dieser Seite genannten Anforderungen nach deiner empfundenen Wichtigkeit. Die Reihenfolge der Priorisierung gibt an, welche Anforderungen an das Feature für dich relevanter sind im Vergleich untereinander. Diese Priorisierung kann unabhängig von den oben vergebenen Scores stattfinden.**

Oben steht die relevanteste Anforderung - Unten die am wenigsten relevante Anforderung

Die sprachgesteuerte Adresseneingabe sollte erklären, welche Adressinformationen zwingend benötigt werden. ☐

Nach jeder erkannten Adresse sollte die App die gefunde Adresse verbal wiederholen. ☐

Das System sollte in der Lage sein, den Kontext der Adressangaben zu verstehen, um logische Fehler zu minimieren (z.B. unplausible Kombinationen von Stadt und Postleitzahl). ☐

Das System greift auf frühere Eingaben des Benutzers zurück, um die Adresseneingabe zu vereinfachen. ☐

Teil F: Bemerkungen

F1. **Ich habe Bemerkungen zu den in dieser Umfrage genannten Anforderungen.**

F2. **Ich habe Bemerkungen zu Aspekten, die in dieser Umfrage nicht genannt wurden.**

F3. **Ich habe Feedback zu dieser Umfrage, unabhängig von den hier besprochenen Inhalten.**

Vielen Dank für deine Teilnahme!

Literaturverzeichnis

[1] M. Balci. Entwurf eines requirements engineering workflows für erklärbare systeme, 2021.

[2] H. Balzert and C. Ebert. *lehrbuch der software-Technik*. Spektrum, Akad. Verlag, 1996.

[3] K. Beck. *Extreme programming explained: embrace change*. addison-wesley professional, 2000.

[4] J. Bergsmann. *Requirements Engineering für die agile Softwareentwicklung: Methoden, Techniken und Strategien*. dpunkt. verlag, 2023.

[5] R. Binns. Fairness in machine learning: Lessons from political philosophy. In *Conference on fairness, accountability and transparency*, pages 149–159. PMLR, 2018.

[6] A. Brennen. What do people really want when they say they want "explainable ai?" we asked 60 stakeholders. In *Extended abstracts of the 2020 CHI conference on human factors in computing systems*, pages 1–7, 2020.

[7] L. Chazette. Requirements engineering für erklärbare systeme, 2023.

[8] L. Chazette, W. Brunotte, and T. Speith. Exploring explainability: A definition, a model, and a knowledge catalogue. In *2021 IEEE 29th International Requirements Engineering Conference (RE)*, pages 197–208, 2021.

[9] L. Chazette and K. Schneider. Explainability as a non-functional requirement: challenges and recommendations. *Requirements Engineering*, 25(4):493–514, 2020.

[10] I. C. S. S. E. T. Committee. *IEEE Standard Glossary of Software Engineering Terminology*, volume 729. IEEE, 1983.

[11] N. Dehn. Anforderungen verschiedener stakeholdergruppen an die stimmungsanalyse in softwareprojekten. *Institut für Praktische Informatik der Leibniz Universität Hannover*, 2020.

[12] J. Droste, H. Deters, M. Obaidi, and K. Schneider. Explanations in everyday software systems: Towards a taxonomy for explainability needs. *arXiv preprint arXiv:2404.16644*, 2024.

[13] J. Friedrich, M. Kuhrmann, M. Sihling, and U. Hammerschall. *Das v-modell xt*. Springer, 2009.

[14] B. Goodman and S. Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI magazine*, 38(3):50–57, 2017.

[15] Google. `https://developer.android.com/reference/android/speech/SpeechRecognizer`. [Online; accessed 10-August-2024].

[16] Graphmasters. `https://graphmasters.net`. [Online; accessed 27-June-2024].

[17] Graphmasters. `https://play.google.com/store/apps/details?id=com.nunav.logistics&hl=de&pli=1`. [Online; accessed 27-June-2024].

[18] T. Grechenig and M. Köhle. *Software engineering mit UML und dem unified process*. Pearson Studium, 2004.

[19] F. Herzog. Konzeption und evaluation eines modells zur unterstützung des designs von erklärungen in erklärbaren systemen, 2021.

[20] Komoot. `https://photon.komoot.io/`. [Online; accessed 10-August-2024].

[21] M. Mistler. *Entwicklung eines Vorgehenskonzeptes zum modellbasierten agilen Anforderungsmanagement (Requirements Engineering und Requirements Management) für Organisationen–REMOt*. PhD thesis, Dissertation. Wuppertal: Bergische Universität Wuppertal, 2021.

[22] OpenAI. `https://openai.com/chatgpt/`. [Online; accessed 10-August-2024].

[23] H. Partsch. *Requirements-Engineering systematisch: Modellbildung für softwaregestützte Systeme*. Springer-Verlag, 2010.

[24] K. Pohl and C. Rupp. *Basiswissen requirements engineering: Aus- und Weiterbildung nach IREB-Standard zum certified professional for requirements engineering foundation level*. dpunkt. verlag, 2021.

[25] W. W. Royce. Managing the development of large software systems (1970), 2021.

[26] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

[27] C. Rupp, M. Simon, and F. Hocker. Requirements engineering und management. *HMD Praxis der Wirtschaftsinformatik*, 46(3):94–103, 2009.

[28] N. Schaaf and P. Wagner. Blick in die black box: Erklärbarkeit maschineller lernverfahren, 2021.

[29] K. Stapel and K. Schneider. Flow-methode-methodenbeschreibung zur anwendung von flow. *arXiv preprint arXiv:1202.5919*, 2012.

[30] I. P. Tussyadiah and F. J. Zach. The role of geo-based technology in place experiences. *Annals of Tourism Research*, 39(2):780–800, 2012.

[31] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, A. Wesslén, et al. *Experimentation in software engineering*, volume 236. Springer, 2012.