

Schriftliche Dokumentation der App

Fachhochschule Aachen
Fachbereich 5 – Elektrotechnik und Informationstechnik



Dokumentation der Scheibner-App

Mobile Informationssysteme
- Dr. ir. Kris Cardinaels -

03.07.2019

Björn Kops
David Neuroth
Felix Bengsch
Florentin Bürvenich

Inhaltsverzeichnis

1 Anforderungen (David)	1
1.1 Funktionale Anforderungen	2
1.2 Optionale Anforderungen	2
1.3 Mögliche Erweiterungen	3
2 Navigation (David)	4
2.1 Ablauf einer Simulation	4
2.2 “...”-Button zur Navigation	5
2.3 Normaler Workflow	5
3 Funktionalität (Kps)	7
3.1 Profilverwaltung	7
3.2 Simulationsdaten laden	8
3.2.1 Laden vom Server	9
3.2.2 Laden aus QR-Code	10
3.3 Daten bearbeiten	10
3.4 Simulation	11
3.4.1 Diagramme	11
3.4.2 Tabelle	12
3.4.3 Kommentar	13
4 Einstellungen	15
4.1 Sprache	16
4.2 Eingabemethode	16
5 Benutzertests- und Interviews (Felix)	18
5.1 Benutzertest- und Interview 1	18
5.2 Benutzertest- und Interview 2	20
5.3 Benutzertest- und Interview 3	22

6 Technische Umsetzung	24
6.1 Speicherung der Profildaten	24
6.2 Speicherung der Einstellungen	26
6.3 Verwaltung des App-States	26
6.4 Daten hinzufügen	26
6.5 Diagramme	27

1 Anforderungen

Die Firma Scheibner m-tec GmbH vertreibt das Rahmenmesssystem mega-m.a.x. zur Diagnose von Motorrädern. Dieses System wird beispielsweise nach Unfällen eingesetzt, um die Maßhaltigkeit eines Fahrzeugs zu überprüfen. Dazu werden bestimmte Messungen am betreffenden Motorrad durchgeführt. Anschließend werden die Ergebnisse mit den entsprechenden Herstellervorgaben abgeglichen.

Als Ergänzung zum mega-m.a.x.-System bietet Scheibner m-tec GmbH das Chassis-Maximizing-System (CMS) an. Dieses ermittelt in Kombination mit mega-m.a.x. zusätzliche Daten zum Motorrad. Darüber hinaus ist in CMS ein Simulationsprogramm enthalten, mithilfe dessen die Auswirkungen verschiedener Änderungen am Motorrad im Vorhinein berechnet werden können. Auf diese Weise können Kunden ohne großen Aufwand unterschiedliche Parametrierungen simulieren und die optimalen Einstellungen für ihre Fahrzeuge und ihre individuellen Anforderungen ermitteln.

Bisher ist CMS für Laptops und Desktop-Computer verfügbar und wird in erster Linie von Werkstätten und Vermessungsservices genutzt. Diese stellen die Mess- und Simulationsergebnisse den Eigentümern der Fahrzeuge zur Verfügung. Ohne Hilfe mit dem jeweiligen Betrieb kann der Eigentümer eines Motorrades daher bisher keine eigenen Simulationen durchführen.

Um auch den Fahrzeugeigentümern die Möglichkeit zu bieten, verschiedene Änderungen am Motorrad zu simulieren und so ein besseres Verständnis für den Optimierungsvorgang zu erlangen, soll eine App für mobile Endgeräte entwickelt werden. Diese soll die mit mega-m.a.x. und CMS gemessenen Daten abrufen und anzeigen können. Anschließend sollen, wie in der Simulationssoftware des CMS, Änderungen an einigen Einstellungen simuliert werden können. Durchgeführte Simulationen sollen dabei für eine weitere Bearbeitung gespeichert werden können.

1.1 Funktionale Anforderungen

Die Hauptfunktion der zu entwickelnden App besteht darin, die Simulationssoftware des CMS für Änderungen an einem Motorrad auch auf mobilen Endgeräten zur Verfügung zu stellen. Daher müssen in erster Linie die für die Simulation notwendigen Berechnungen implementiert werden. Da die App die benötigten Messdaten nicht direkt vom mega-m.a.x.- und CMS-System erhalten kann, müssen diese stattdessen von einem Server der Scheibner m-tec GmbH heruntergeladen werden. Zur Identifikation des korrekten Datensatzes muss eine Messungs-ID eingegeben werden. Um die Simulation durchführen zu können, muss die App eine Eingabemaske enthalten, über die die fünf parametrierbaren Werte eingestellt werden können. Im Anschluss an eine Simulation müssen die Ergebnisse in einer Tabelle angezeigt und den gemessenen Werten gegenübergestellt werden. Die berechneten Werte müssen gespeichert und zu einem späteren Zeitpunkt wieder abgerufen werden können. Dabei muss die App die Möglichkeit bieten, eine Simulation mit erklärenden Notizen zu versehen.

Da die App insbesondere auch für Eigentümer ohne viel Hintergrundwissen oder Erfahrung mit CMS gedacht ist, muss die Oberfläche der App intuitiv bedienbar und so weit wie möglich selbsterklärend sein. Da die meiste Zeit mit dem Parametrieren von Einstellungen und der anschließenden Betrachtung der Simulationsergebnisse verbracht werden wird, muss dieser Teil der App besonders leicht und schnell bedienbar sein.

Zusätzlich zu diesen Anforderungen müssen für die App ein Icon und ein Launch Screen erstellt werden.

1.2 Optionale Anforderungen

Neben den oben beschriebenen funktionalen Anforderungen sind einige Ergänzungen erstellenswert, die die App leichter bedienbar oder optisch ansprechender machen, um so insgesamt das Nutzungserlebnis zu verbessern. Für eine fehlerfreie Funktion der App sind diese Erweiterungen nicht notwendig.

Neben dem Download der Messdaten vom Scheibner-Server stellt das Auslesen aus einem Barcode oder einem QR-Code eine weitere Möglichkeit dar. Ein solcher Code könnte von anderen Anwendungen erzeugt und dann genutzt werden, um mit der App schnell und einfach Messdaten einzulesen. Die Veränderung der Simulationsparameter erfolgt standardmäßig wie im CMS über die direkte Eingabe der gewünschten Werte mittels eingeblendeter numerischer Tastatur. Alternativ stellt eine graphische Eingabeviante, beispielsweise in

Form von Slidern, eine angenehmere und benutzerfreundlichere Form der Werteeingabe dar. Die Simulationsergebnisse sollen zunächst in einer einfachen Tabelle dargestellt werden. Ergänzend können Grafiken generiert werden, die die wichtigsten Veränderungen deutlicher und übersichtlicher anzeigen.

1.3 Mögliche Erweiterungen

Einige Anforderungen müssen zunächst nicht erfüllt werden, sind aber eventuell für eine zukünftige Weiterentwicklung der App vorgesehen. Einige dieser Features können auch deshalb noch nicht implementiert werden, weil die nötige Serverinfrastruktur noch nicht eingerichtet ist. Dazu gehört zum Beispiel der direkte Zugriff auf die Datenbank von Scheibner, die zu vielen Fahrzeugmodellen die entsprechenden Herstellervorgaben liefert. Um nur authentifizierten Benutzern den Zugriff zu erlauben, wäre dann auch eine Überprüfung der Zugriffsberechtigungen notwendig. Außerdem könnte in Zukunft ein Server eingerichtet werden, auf den die Simulationsergebnisse aus der App hochgeladen werden können.

2 Navigation

Die Hauptfunktionalität der Scheibner-App besteht darin, anhand vorhandener Messdaten die Auswirkungen von weiteren Änderungen am Fahrzeug zu simulieren. Um diese Funktionalität einfach und übersichtlich zur Verfügung zu stellen, besteht die App im wesentlichen aus vier Seiten. Die erste davon ist die Profilseite. Hier können Profile angelegt und gelöscht werden. Durch Auswahl eines existierenden Profils gelangt der Nutzer auf die zweite Seite. Dort werden, falls verfügbar die aktuellen Messdaten des Fahrzeugs angezeigt. Auf dieser Seite können auch neue Messdaten geladen werden, vom Scheibner-Server oder per QR-Code. Über einen weiteren Button wird die Simulationsseite geöffnet. Dort können für fünf der Messwerte Änderungen simuliert werden. Wenn die gewünschten Werte eingestellt wurden, kann die Simulation über einen Button ausgeführt werden. Daraufhin wird die letzte Seite geöffnet, die die Ergebnisse der Simulation zeigt. Diese Seite besteht aus drei separaten Tabseiten. Auf der ersten werden die simulierten Werte tabellarisch angezeigt und den gemessenen Werten gegenübergestellt. Auf der zweiten Tabseite werden die Simulationsergebnisse graphisch dargestellt, und auf der letzten Seite können ergänzende Notizen zur Simulation gespeichert werden. Um ein leichtes und schnelles Navigieren zu ermöglichen, gibt es auf jeder Seite (außer der Profilseite) einen „Zurück“-Button, um zur vorherigen Seite zurückzukehren. Auf der Ergebnisseite wurden mehrere Tabseiten verwendet, um schnell zwischen der tabellarischen und der graphischen Ergebnisdarstellung hin und her wechseln zu können.

2.1 Ablauf einer Simulation

Die meiste Zeit werden Anwender voraussichtlich damit verbringen, Werte zur Simulation einzugeben und sich das entsprechende Simulationsergebnis anzuschauen. Daher wurde darauf Wert gelegt, dass diese Arbeitsschritte möglichst einfach erledigt werden können. Daher können die Simulationsseite und die Ergebnisseite von der jeweils anderen aus direkt über einen Button erreicht werden.

2.2 “...”-Button zur Navigation

Um auch einige weniger häufig besuchte Seiten schnell erreichen zu können, wurde in der AppBar ein “...”-Button eingefügt, der ein PopUp mit weiteren Buttons öffnet. Über die Buttons in diesem PopUp können die Profilseite und die Einstellungsseite direkt geöffnet werden.

Ein Profil repräsentiert in dieser App ein Fahrzeug. Daher ist nicht zu erwarten, dass während einer Benutzung häufig zwischen verschiedenen Profilen gewechselt wird. Die Einstellungsseite ermöglicht das Wechseln der Sprache oder der Eingabemethode auf der Simulationsseite. Wie bei der Profilseite ist auch hier nicht zu erwarten, dass häufig zwischen dieser und anderen Seiten hin und her gewechselt wird. Über den “...”-Button können diese beiden Seiten von jeder anderen Seite aus dennoch leicht und schnell erreicht werden, ohne in der AppBar zuviel Platz einzunehmen.

2.3 Normaler Workflow

Der typische Workflow eines Anwenders dieser App besteht aus dem Auswählen eines Profils, gegebenenfalls dem Laden neuer Daten und dem – unter Umständen wiederholten – Simulieren von Änderungen am Fahrzeug. Der genaue Ablauf wurde in einem Aktivitätsdiagramm dargestellt (Abschnitt 2.3).

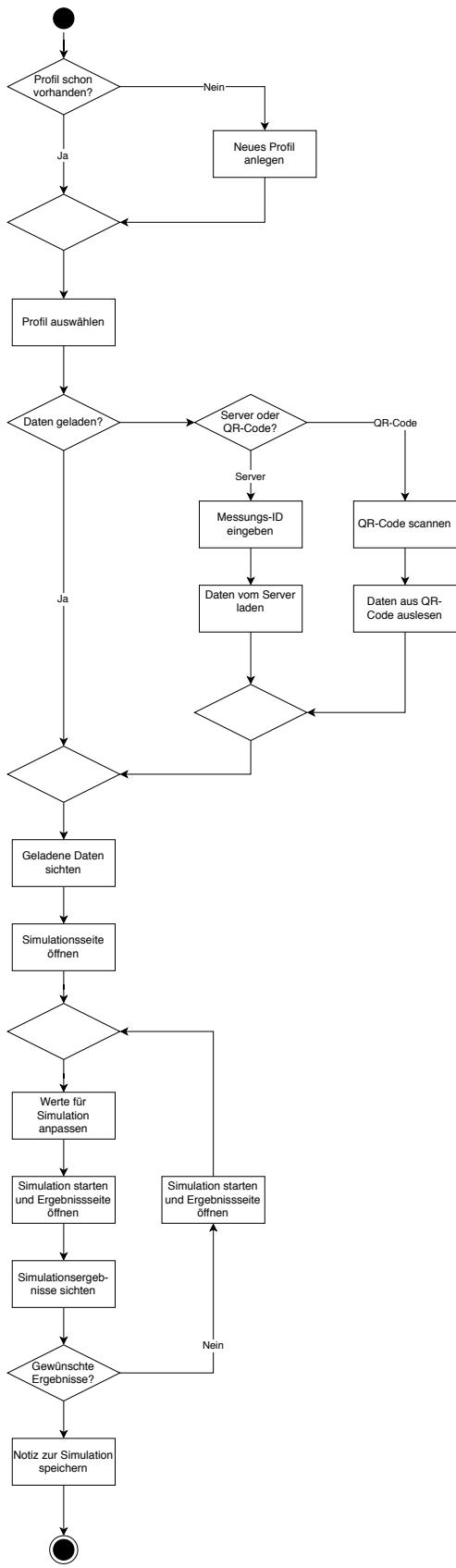


Abbildung 2.1: Typischer Workflow eines Anwenders

3 Funktionalität

Im Folgenden werden die Funktionalitäten der App hinsichtlich der Anwendernutzung beschrieben. Dabei wird erklärt, wie man Profile anlegt und Daten in ein Profil lädt. Zudem werden die Auswertungsseiten der Simulation kurz erläutert.

3.1 Profilverwaltung

Die Profilverwaltung ist die Startseite der App. Dort können Profile angelegt und gelöscht werden.

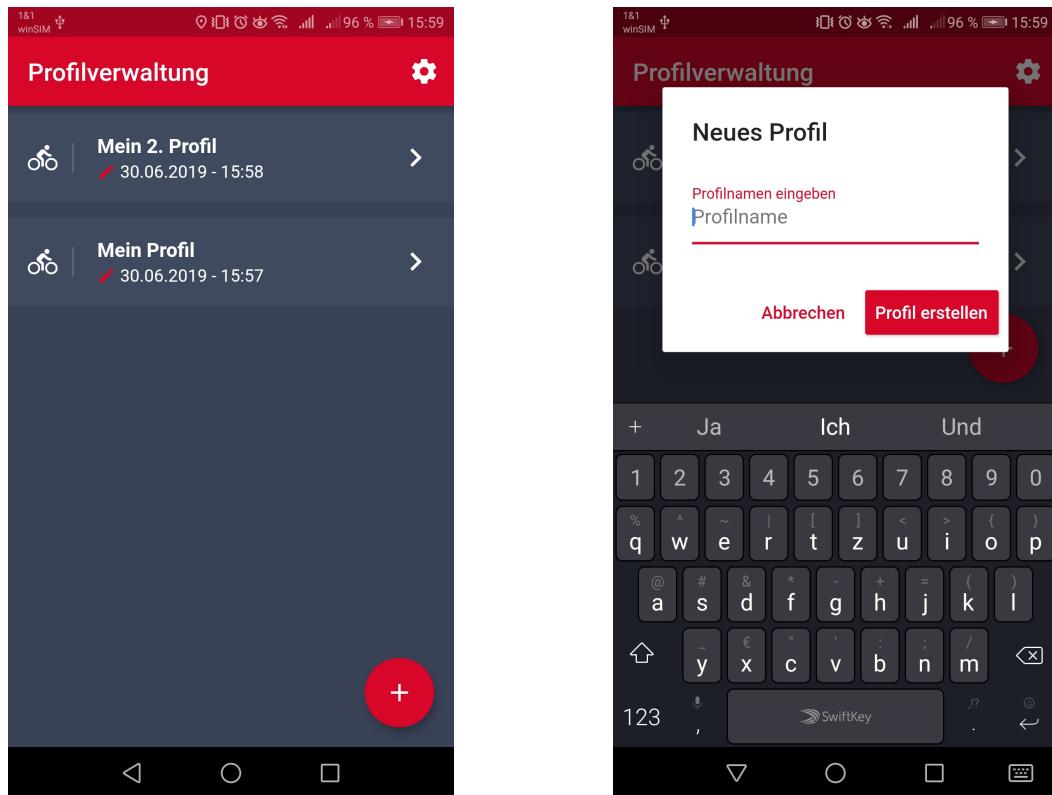


Abbildung 3.1: Umsetzung der Profilverwaltung

Anlegen eines Profils

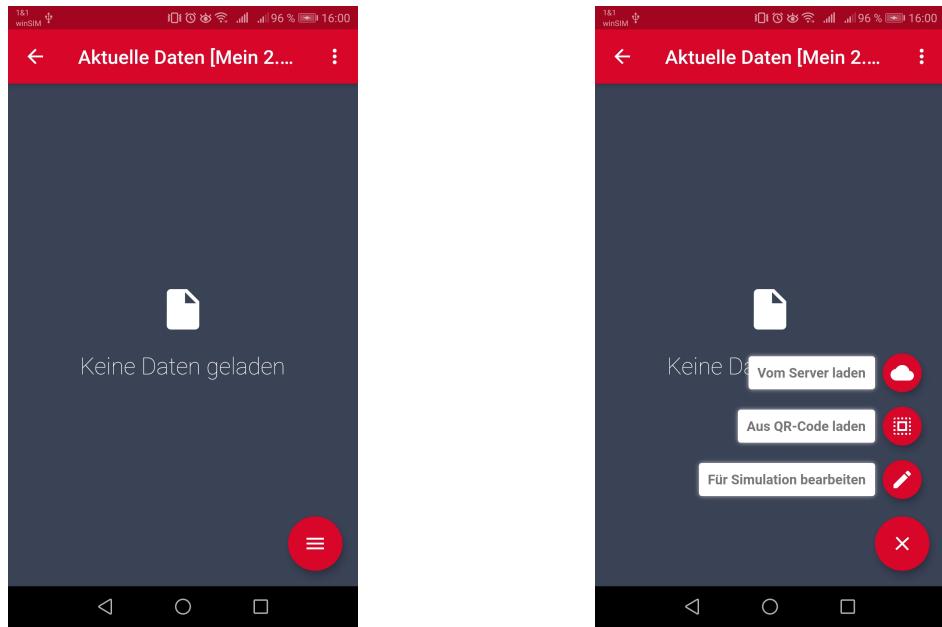
Das Anlegen eines Profils wird durch Drücken des mit einem „+“ versehenden Floating-Action-Buttons eingeleitet. Nach drücken des Buttons öffnet sich ein Fenster, wie in Abschnitt 3.1 zu sehen. Dort kann über ein Eingabefeld der Profilname hinzugefügt werden. Anschließend kann über den Button „Profil erstellen“ oder den Button „Abbrechen“ das hinzufügen eines Profils beendet werden.

Löschen eines Profils

Ein Profil kann gelöscht werden indem man eine Wischbewegung (Swipen) nach links auf dem entsprechenden Profil ausführt. Beginnt man die Wischbewegung nach links so, wird das Profil zur Seite geschoben und ein Mülleimericon, sowie der Schriftzug löschen angezeigt, um den Nutzer davon zu unterrichten, dass seine Aktion zur Löschung des Profils führt. Stoppt man die Wischbewegung bevor das Profil ganz aus dem Bildschirm verschwunden ist, so wird das Löschen abgebrochen.

3.2 Simulationsdaten laden

Das Laden der Daten für die Simulation wird auf einer eigenen Seite ausgeführt. Sind keine Daten vorhanden, so wird dies angezeigt. Über einen Floating-Action-Button ist es möglich ein Buttonmenü zu öffnen. Dies erlaubt dem Nutzer, Daten vom Server zu oder aus einem QR-Code zu laden oder Daten für die Simulation zu bearbeiten. Sind keine Daten geladen, so ist die Bearbeitung der Daten für die Simulation auch nicht nutzbar. Sind bereits Daten für dieses Profil geladen worden, so werden sie direkt geladen, da sie mit dem Profil verbunden sind. Die zu geladenen Daten bestehen aus Kennzahlen, die die in einer Liste angezeigt werden.



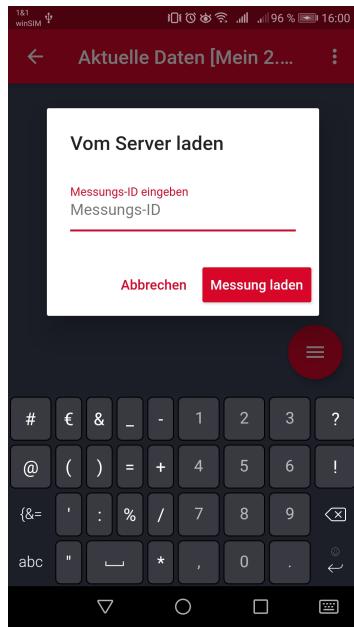
(a) Seite ohne Daten

(b) Buttonsmenü

Abbildung 3.2: Die Laden Daten Ansicht

3.2.1 Laden vom Server

Drückt man den Button „Vom Server laden“ so öffnet sich ein Fenster ähnlich zum Anlegen eines neuen Profils. Über ein Eingabefeld muss eine Messungs-ID übergeben werden. Diese bestimmt welche Daten vom Server geladen werden. Durch einen Klick auf „Messung laden“ oder „Abbrechen“ kann die Aktion entsprechend beendet werden. Das Fenster schließt sich und während des Ladens kommt eine kurze Ladeanimation. Sind die Daten geladen, so werden diese nun angezeigt.



(a) Daten vom Server laden

Aktuelle Daten [Mein 2....]	
Radumfang vorne	1889.0 mm
Radumfang hinten	1948.0 mm
Gabellänge	525.2 mm
Heckhöhe	550.0 mm
Schwingenlänge	533.7 mm

(b) Nach Laden der Daten

Abbildung 3.3: Die Laden Daten Ansicht

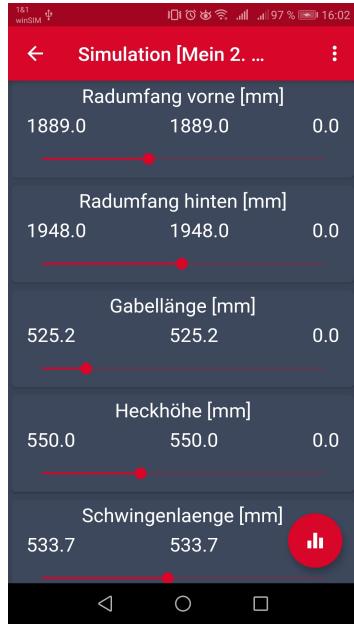
3.2.2 Laden aus QR-Code

Wird der Button „Aus QR-Code laden“ gedrückt, so öffnet sich die Kamera zum Scannen eines QR-Codes. Sobald der QR-Code erfasst wird, werden die Daten geladen und angezeigt. Ist der QR-Code wiederum ungültig, wird ein Fehler in Form einer Snackbar angezeigt.

3.3 Daten bearbeiten

Um eine Simulation durchzuführen, müssen gewisse Werte steuerbar sein. In der Daten bearbeiten Ansicht, werden die Kennzahlen dargestellt, die für die Simulation steuerbar sind. Dabei handelt es sich um folgende Kennzahlen:

Diese werden in einer Liste dargestellt. Zu jeder Kennzahl werden je drei Werte angezeigt. Links ist der alte Wert und in der Mitte der neue Wert der Kennzahl angezeigt. Auf der rechten Seite sieht man die Differenz zwischen dem neuen und den alten Wert. Ist die Differenz positiv, so wird sie grün, ist sie negativ so wird sie rot dargestellt. Über Slider oder wahlweise auch Inputfelder kann der neue Wert je Kennzahl verändert werden. Die Einstellung welche Eingabevariante man nutzten möchte, kann in den Einstellungen umgestellt werden.



(a) Mit Slidern



(b) Als Inputfeld

Abbildung 3.4: Daten bearbeiten Ansicht

3.4 Simulation

Die Simulation wird auf einer Seite mit drei Tabs dargestellt. Der Tab in der Mitte beinhaltet Diagramme. Dieser wird standardmäßig als erstes geöffnet. Durch eine Wischbewegung (swipen) nach links oder rechts, kann man auf die anderen Tabs gelangen. Der Tab auf der linken Seite stellt die Ergebnisse als Tabelle dar. Zusammen mit dem Diagramm-Tab werden dort die Ergebnisse der Simulation angezeigt. Der Tab auf der rechten Seite hingegen, ist nur um zusätzliche Informationen zur Simulation hinzuzufügen.

3.4.1 Diagramme

Der Diagrammtab beinhaltet fünf Diagramme. Das erste Diagramm ist ein Balkendiagramm, das eine Übersicht über die Veränderungen geben soll. Es stellt die größten prozentuellen Veränderungen der Kennzahl dar. Dies bietet dem User die Möglichkeit auf einem Blick zu sehen, welche Kennzahlen sich am meisten verändert haben. Zur Vergleichbarkeit zwischen Winkeln und Längenmaßen wurde hier die prozentuelle Veränderung angezeigt. Kennzahlen, die durch die Simulation kleiner werden, werden durch rote Balken gekennzeichnet. Sich vergrößernde Kennzahlen hingegen werden grün dargestellt. Gleichermaßen gilt auch für die vier folgenden Diagramme. Die Single Kennzahl Diagramme, zeigen

die Veränderung einer Kennzahl im Detail an. Dabei werden der alte und neuer Wert, sowie die Differenz angezeigt. Zusätzlich wird anhand eines Säulendiagramms der alte und neue Wert der Kennzahl nochmal grafisch dargestellt. Single Kennzahl Diagramme werden für die vier Kennzahlen mit der größten absoluten Änderung dargestellt.

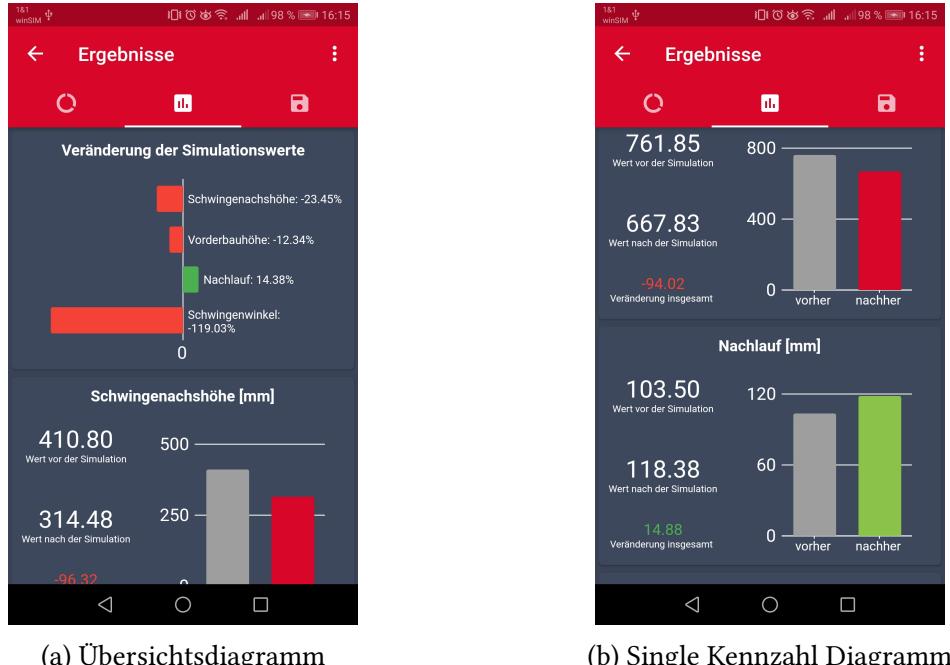


Abbildung 3.5: Diagrammtypen

3.4.2 Tabelle

Die Tabellenansicht gibt einen guten Überblick über die alle Werte an. Dabei werden zunächst die aus der Simulation hervorgehenden Werte angezeigt. Anschließend werden die bearbeiteten Eingabewerte angezeigt. So kann der Nutzer erkennen aus welchen Eingabewerten, welches Simulationsergebnis vorher geht. Es werden, wie auch in der Dateneingabe der alte und neue Wert der Kennzahl, sowie die Differenz angezeigt.

(a) Simulationswerte

Lenkkopfwinkel [°]		
25.1	25.1	0.0
Nachlauf [mm]		
103.5	98.5	-5.0
Offset [mm]		
33.8	33.8	0.0
Schwingenwinkel [°]		
10.9	-0.8	-11.7

(b) Eingabewerte

Radumfang vorne [mm]		
1889.0	1889.0	0.0
Radumfang hinten [mm]		
1948.0	2076.4	+128.4
Gabellänge [mm]		
525.2	525.2	0.0
Heckhöhe [mm]		
550.0	911.5	+361.5

Abbildung 3.6: Daten bearbeiten Ansicht

3.4.3 Kommentar

Die Kommentarseite bietet dem User die Möglichkeit eine Notiz oder einen Kommentar für die Simulation zu speichern. Ein Textfeld ermöglicht die Eingabe des Kommentars. Der Kommentar wird dabei automatisch gespeichert.

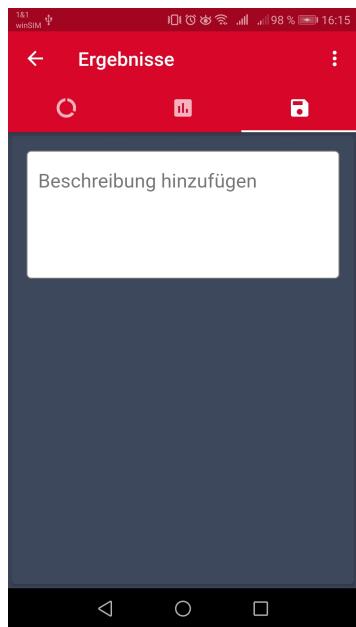


Abbildung 3.7: Kommentarseite

4 Einstellungen

Über das Einstellungsmenü lässt sich die App in zwei Punkten konfigurieren:

1. Einstellung der Sprache (s. Abschnitt 4.1)
2. Einstellung der Eingabemethode (s. Abschnitt 4.2)

Die Einstellungsseite wird je nach Ansicht, in der sich der Nutzer aktuell befindet, auf verschiedene Arten geöffnet:

1. **Profilverwaltung:** Die AppBar der Profilverwaltung enthält einen Button, der aus einem Zahnrad-Icon besteht und per direktem Klick die Einstellungsansicht öffnet (s. Abb. 4.1 links).
2. **Sonstige Ansichten:** Alle weiteren Ansichten (außer der Profilverwaltung) enthalten in der AppBar ein Dropdown (s. Abb. 4.1 Mitte), mithilfe dessen über den zweiten Menüpunkt die Einstellungsseite geöffnet werden kann (s. Abb. 4.1 rechts).

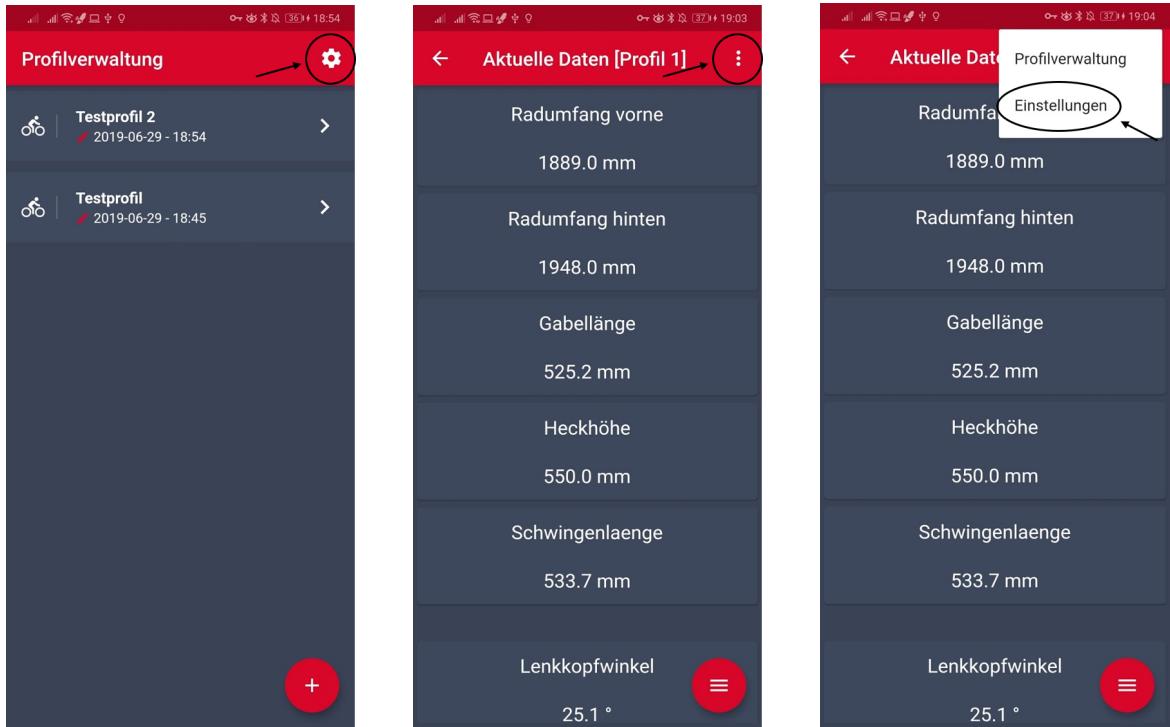


Abbildung 4.1: Links: Profilverwaltung, Mitte: Button für Dropdown-Menü, Rechts: Dropdown-Menü

4.1 Sprache

Die App ist in den Sprachen Deutsch und Englisch verfügbar. Die Einstellung wird standardmäßig auf die im Gerät eingestellte Sprache gesetzt und kann manuell verändert werden (s. Abb. 4.2 rechts)

4.2 Eingabemethode

Neue Werte, die von der App simuliert werden sollen, werden über die Ansicht "Simulation" eingegeben. Hierfür kann in den Einstellungen angepasst werden, ob die Werte mit Textfeldern oder Slidern eingegeben werden sollen (s. Abb. 4.2 links).

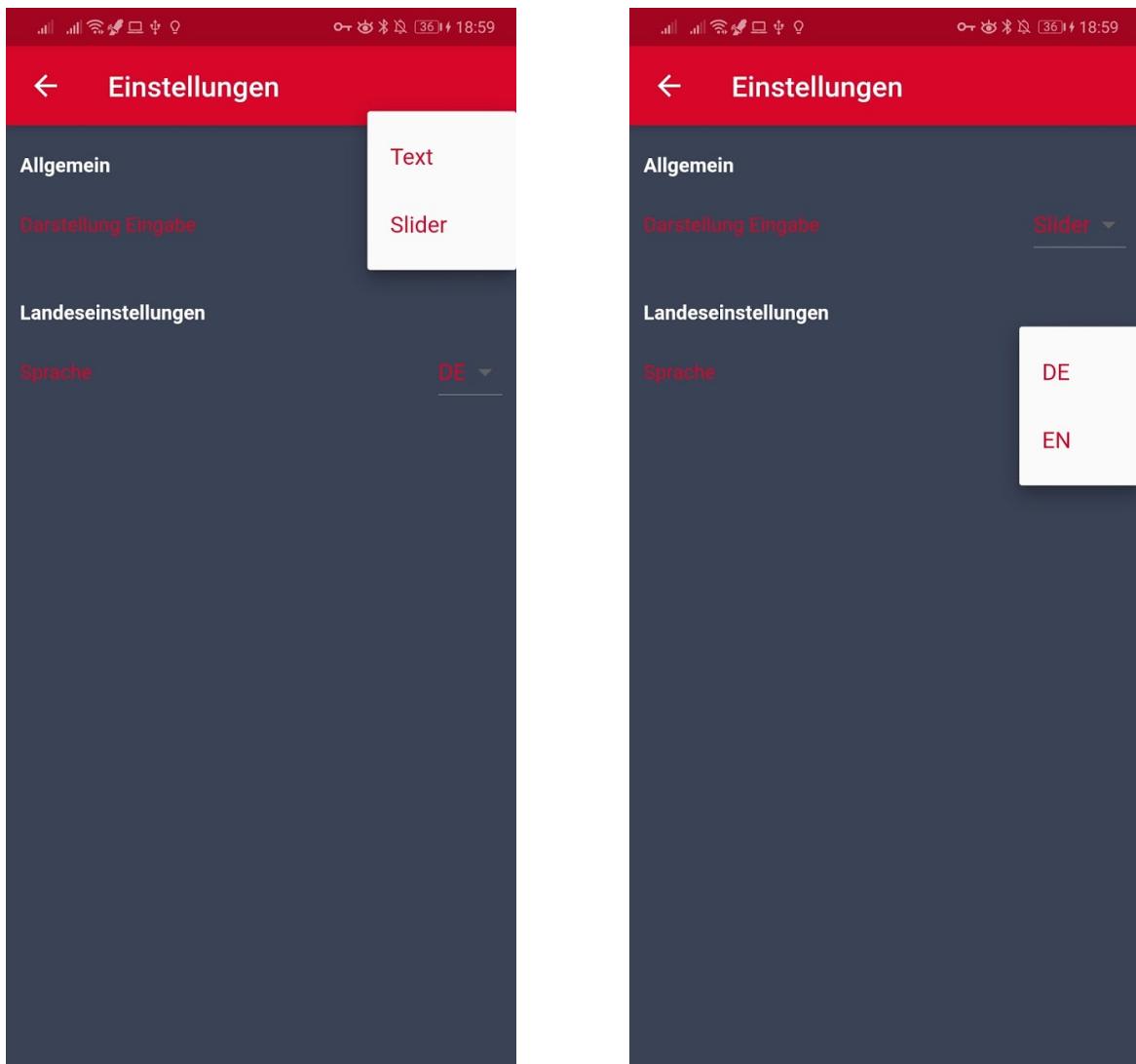


Abbildung 4.2: Links: Einstellung der Eingabemethode, Rechts: Einstellung der Sprache

5 Benutzertests- und Interviews

Es wurden drei Interviews mit Personen durchgeführt, welche die App bewerten sollen. Den Probanden wurde dabei jeweils nur erklärt, was der Nutzen einer Messung und Simulation von *MEGA m.a.x.* und wofür die App dienen soll, alles weitere mussten sich die Personen selbst erarbeiten. Damit sollte getestet werden, ob die App intuitiv benutzbar und gut zu bedienen sind. Die Probanden sollten die App jeweils an Hand dieser vier Punkte bewerten:

- Wie ist der erste Eindruck?
- Was ist positiv aufgefallen?
- Was ist negativ aufgefallen?
- Ist die App intuitiv bedienbar?

Nach der Beantwortung dieser Fragen wird jeweils noch beschrieben, welche Anpassungen auf Grund des Feedbacks des jeweiligen Interviews durchgeführt wurden.

5.1 Benutzertest- und Interview 1

Wie ist der erste Eindruck? Mein erster Eindruck der App ist sehr positiv. Die Corporate Identity ist konsequent umgesetzt worden und die Farben sind untereinander sehr stimmig. Des Weiteren ist die Geschwindigkeit der App sehr hoch, sodass man nie lang warten muss. Sogar die Simulation von Eingabewerten geht sehr schnell, obwohl ich mir vorstellen kann, dass sehr komplexe Berechnungen dahinterstehen.

Was ist positiv aufgefallen? Wie bereits erwähnt, ist mir die hohe Geschwindigkeit der App aufgefallen. Auch die übersichtliche Darstellung, sowohl der veränderbaren Werte, als auch der simulierten Werte, sorgt meiner Meinung nach dafür, dass die App in sich sehr aufgeräumt wirkt.

Was ist negativ aufgefallen? Ein Punkt, der mir nicht so gut gefallen hat ist, dass das Textfeld für den Kommentar nur eine Zeile hat. Drückt man innerhalb dieses Textfeldes

auf "Enter", wo wird dieses um eine Zeile länger. Ich finde dies sehr unintuitiv, weil sonst die Gefahr besteht, dass der Benutzer denkt, dass dort nur einzeilige Kommentare möglich sind.

Ist die App intuitiv bedienbar? Nach der Einführung in MEGA *m.a.x.* und eine Erläuterung, wofür diese App dienen soll, konnte ich direkt loslegen und Profile anlegen sowie Simulationen durchführen. Es war keine weitere Anleitung notwendig, was sehr für die Intuitivität spricht.

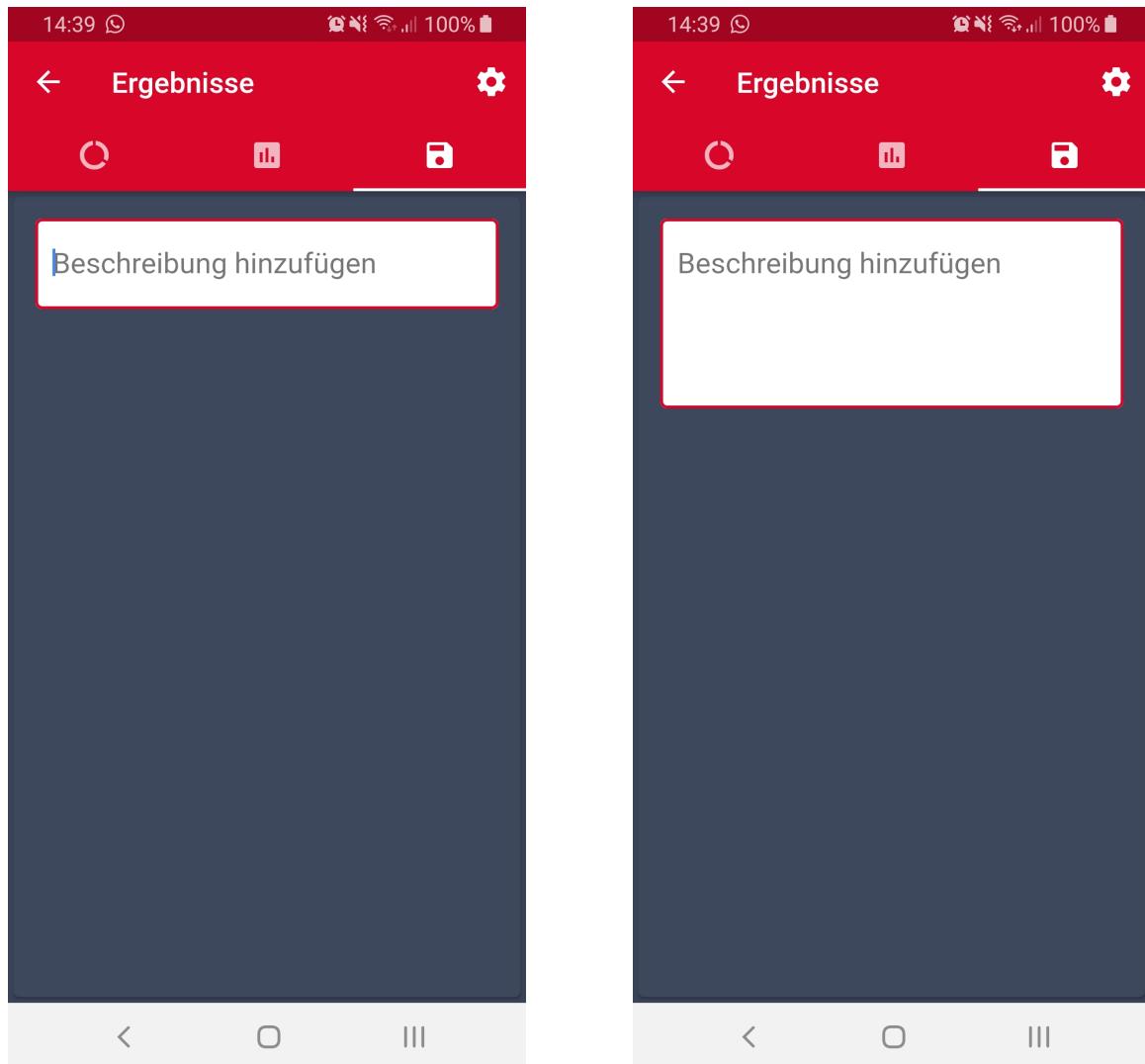


Abbildung 5.1: Links: Kommentarfeld vor dem Interview, rechts nach dem Interview

Nach Durchführung des Interviews wurde das Kommentarfeld so angepasst, dass es etwas größer ist, um Verwirrung zu vermeiden. Dies ist in Abschnitt 6.1 zu sehen. Um diesen

Effekt zu erreichen, wurde die Eigenschaft `minLines` des `TextField`s auf den Wert 4 gesetzt, vorher war dieser nicht gesetzt.

5.2 Benutzer- und Interview 2

Wie ist der erste Eindruck? Die App wirkt auf den ersten Blick sehr strukturiert und leicht zu bedienen. Wäre ich ein potentieller Kunde von Scheibner, würde es mir auf jeden Fall Spaß machen, mit dieser App zu arbeiten. Nach einiger Zeit habe ich die bildliche Darstellung der Simulationsergebnisse in Form von Diagrammen entdeckt, welche die Ergebnisse sehr anschaulich darstellen.

Was ist positiv aufgefallen? Mir haben die Einstellungsmöglichkeiten sehr gut gefallen. Es ist beispielsweise für einen Benutzer immer angenehmer, wenn die App automatisch in seiner Landessprache gestaltet ist und nicht in einer “Standardsprache” wie englisch. Weiterhin bietet die Einstellung, die Eingabe der veränderbaren Werte wahlweise durch Textfelder oder per Slider vorzunehmen, sehr viel Komfort. Will man einen sehr genauen Wert eingeben, so ist die Eingabe per Textfeld von Vorteil, soll nur eine ungefähre Simulation durchgeführt werden, so eignen sich die Slider besser, weil die Werte so schnell verändert werden können.

Ein weiterer Punkt, der mir sehr positiv aufgefallen ist, dass mehrere Fahrzeuge, sprich Profile in der App angelegt und verwaltet werden können. Dies bietet insbesondere Besitzern von mehreren Motorrädern die Möglichkeit, all ihre Fahrzeuge in einer App zu simulieren.

Was ist negativ aufgefallen? Was mir nicht gut gefallen hat, ist die Tatsache, dass die Einstellungen auf der Einstellungsseite nicht sehr gut zu erkennen waren. Das lag daran, dass die Texte schwarz und der Hintergrund ein sehr dunkles grau sind.

Ist die App intuitiv bedienbar? Da ich keinerlei weitere Anleitung benötigt habe, ist die App aus meiner Sicht intuitiv bedienbar. Für die Anwender, die vorher immer in eine Werkstatt vor Ort gehen mussten, um eine neue Simulation auszuführen, wird sie sicherlich einen hohen Mehrwert haben. Für die Werkstätten hat dies auch den Vorteil, dass sich die Mitarbeiter nicht mehr mit der stupiden Arbeit der Simulation befassen müssen: Es reicht dann, die Messung durchzuführen, dem Kunden die ID der Messung mitzuteilen und dieser kann die Simulation bequem selbst zu Hause durchführen. Schließlich kann er

mit seinen angepassten und fertig simulierten Werten zurück zur Werkstatt kommen, um sein Motorrad entsprechende umzurüsten.

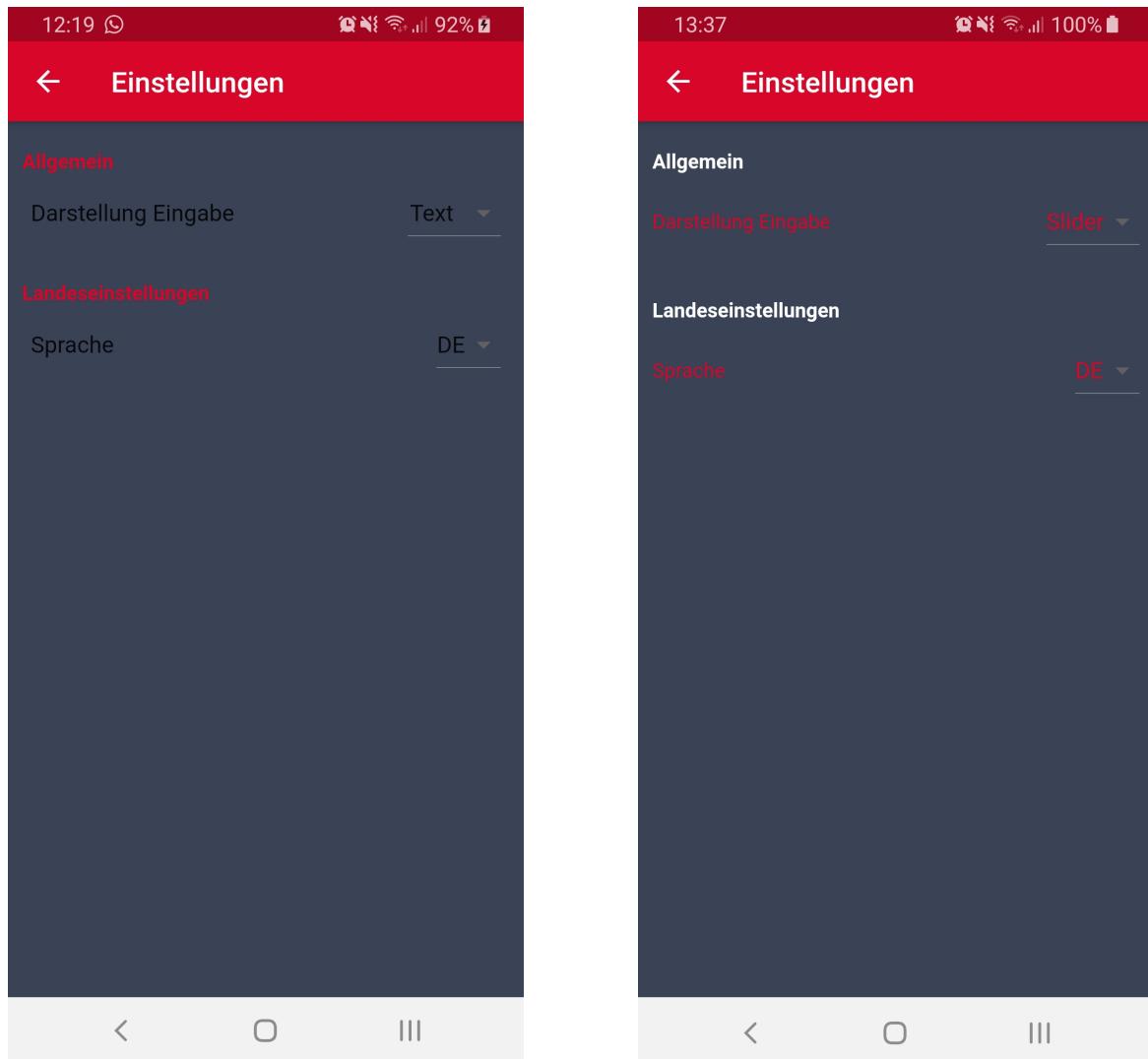


Abbildung 5.2: Links: Einstellungsseite vor dem Interview, rechts nach dem Interview

Nach dem Feedback wurde die Einstellungsseite so angepasst, dass der Text der einzelnen Kategorien weiß und der Text der einzelnen Einstellungen dunkelrot ist. Diese Anpassung zog einige Anpassungen nach sich, weil zuvor das Flutter-Paket *preferences* verwendet wurde. Dies lässt aber leider keine Konfiguration der Farben zu, sodass die komplette Einstellungsseite selbst programmiert werden musste. Der zu Grunde liegende *PrefService*, welcher auf die *SharedPreferences* zugreift, wird weiterhin verwendet. Das dynamische Setzen der Bedienungselemente wie die Auswahlliste und das automatische Speichern der neuen Werte musste nachprogrammiert werden. Abb. 5.2 zeigt die Einstellungs-

seite vor und nach den Anpassungen

5.3 Benutzertest- und Interview 3

Wie ist der erste Eindruck? Die App macht auf mich einen sehr guten Eindruck, es gibt keinen Punkt, der mich extrem gestört hat. Die Darstellung der einzelnen Daten ist sehr strukturiert, außerdem beschränkt sich die App auf das Wesentliche und ist nicht überladen. Dadurch war es sehr angenehm, sie zu benutzen. Insbesondere gefällt mir die Farbgebung sehr gut, das rot wirkt angenehm und auch die weiße Schrift in Kombination mit dem dunkelgrauen Hintergrund wirkt sehr stimmig.

Was ist positiv aufgefallen? Insgesamt hat mir die App sehr gut gefallen, sodass es keinen einzelnen Punkt gibt, den ich besonders hervorheben möchte. Die App als Ganzes wirkt in sich abgeschlossen und es gibt kein Element, welches in ihr "fremd" wirkt. Sie bietet einige Komfortfunktionen, beispielsweise das Einlesen der Simulationsdaten über einen zuvor generierten QR-Code, aber auch das Einlesen über die ID der Simulation ist sehr gut gelungen. Hat man die Daten als QR-Code zur Verfügung, wo ist die App komplett ohne Internet-Zugriff verwendbar, da alle Daten lokal auf dem Gerät gespeichert werden, auch dies ist ein positiver Aspekt.

Was ist negativ aufgefallen? Trotz aller zuvor erwähnten positiven Aspekte gibt es einen Punkt, den ich als störend empfunden habe: Befindet man sich auf der Seite, auf der die Simulationsergebnisse angezeigt werden, so ist es nicht möglich, das Profil zu wechseln, ohne den ganzen Weg "zurückzugehen". Dies ist insbesondere dann sehr nervig, wenn man mehrere Profile hat, zwischen denen man oft wechselt muss. Ich würde mir deshalb eine Funktion wünschen, durch die ich das Profil auch dann schnell wechseln kann, wenn ich mich auf der Seite mit den Simulationsergebnissen befinde.

Ist die App intuitiv bedienbar? Durch die Verwendung des *Materialdesigns* passt sich die App nahtlos an bereits bestehende Apps an und wirkt nicht wie ein Fremdkörper. Dadurch fiel mir die Bedienung sehr leicht und ich habe keine weitere Hilfe benötigt. Außerdem werden bereits durch andere Apps etablierte Gesten wie das Wischen über ein Listenelement zum Löschen verwendet, sodass sich die App auch in diesem Punkt gut eingliedert.

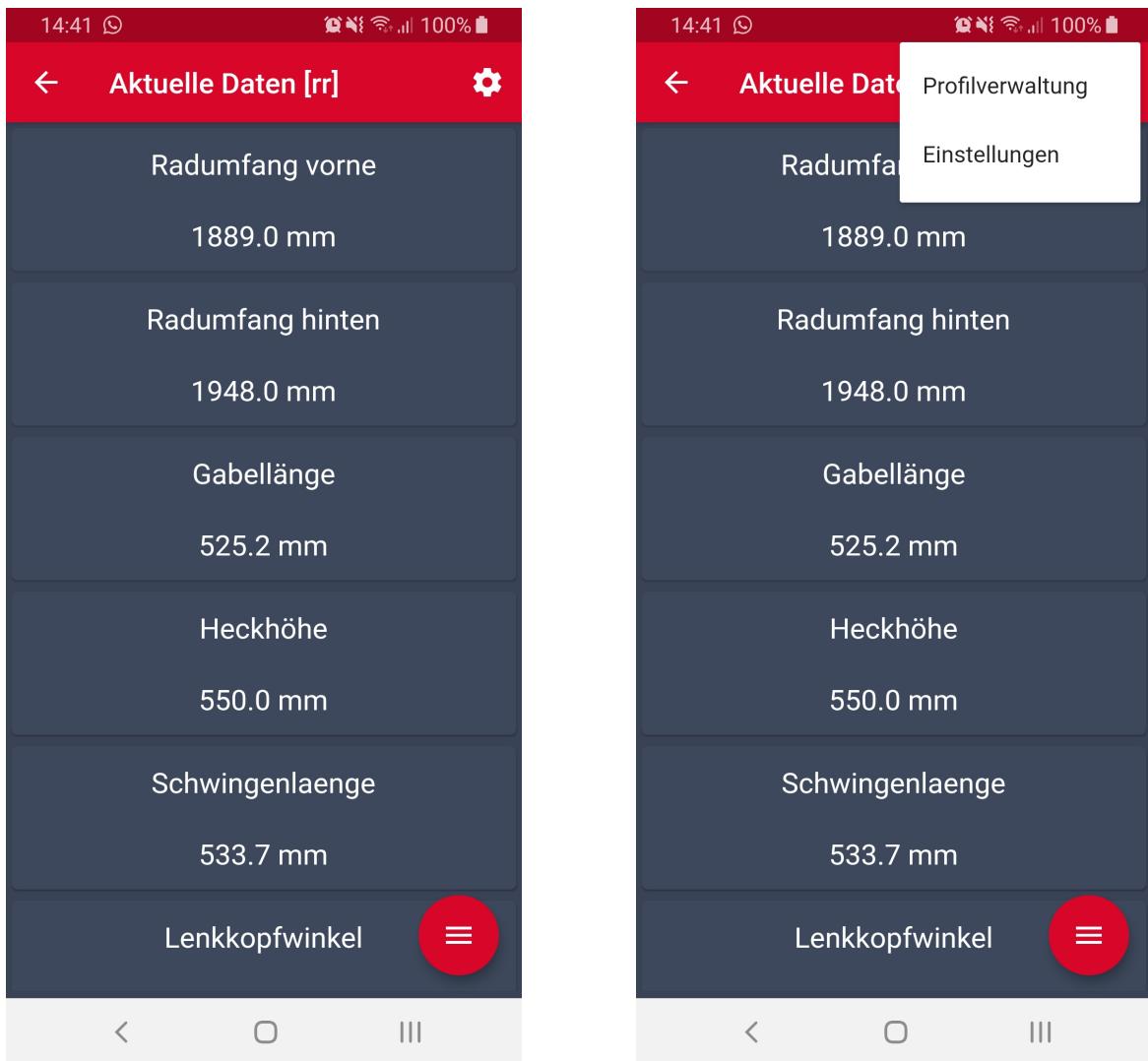


Abbildung 5.3: Links: Übersichtsseite vor dem Interview, rechts nach dem Interview. Zu sehen ist das Drei-Punkt-Menü, was eine verbesserte Navigation ermöglicht.

Die Kritik des Probanden wurde zum Anlass genommen, die Navigation der App zu verändern: Befand man sich vorher auf der Seite, auf der die Ergebnisse der Simulation dargestellt werden, so musste man drei mal zurück gehen, um das Profil zu ändern. Nach der Anpassung ist dies nun über das Drei-Punkt-Menü möglich, die ist in ... zu sehen.

6 Technische Umsetzung

6.1 Speicherung der Profildaten

Profildaten, die der Nutzer in der App eingibt, werden persistent gespeichert. Die Daten werden mit dem Flutter Plugin SQFLite (<https://pub.dev/packages/sqlite>) gespeichert. Das Plugin speichert Daten in einer SQLite Datenbank.

Profildaten werden bei jeder Änderung automatisch gespeichert. Hierzu gehört nicht nur das Laden neuer Daten, sondern auch das Verändern von EingabevARIABLEN und dem Kommentarfeld.

Die Datenverwaltung basiert im wesentlichen auf zwei Komponenten:

- **Profilklasse (*profile.dart*):** Die Profilklassse repräsentiert Profile, die in der App gespeichert werden können. Jedes Profil besteht dabei aus einer ID, einem Namen, dem letzten Änderungsdatum, der ID, mit der die Messung vom Server geladen wurde (falls die Messung nicht per QR-Code gescannt wurde), einem Satz von Messwerten, einem Satz von Simulationswerten und dem optionalen Kommentar.

Die Klasse verfügt außerdem über die Methode *toMap()*, die ein Profil in eine Map konvertiert, die dann in der Datenbank gespeichert werden kann.

- **Datenbank-Hilfsklasse (*database_helpers.dart*):** Die Hilfsklassse stellt sämtliche Funktionalitäten zur Verfügung, die genutzt werden, um mit der SQLite Datenbank zu interagieren. Die Klasse verwaltet die Erstellung der Datenbank-Tabelle und kann Daten zur Datenbank hinzufügen, löschen und bestehende Daten aktualisieren.

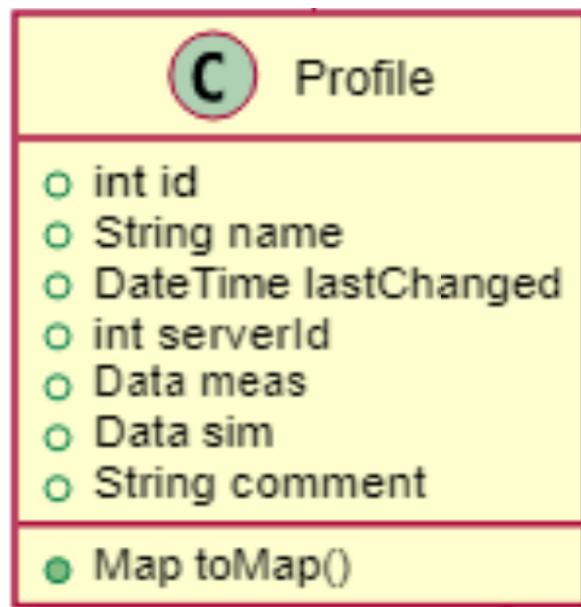


Abbildung 6.1: Aufbau der Profilkasse (*profile.dart*)

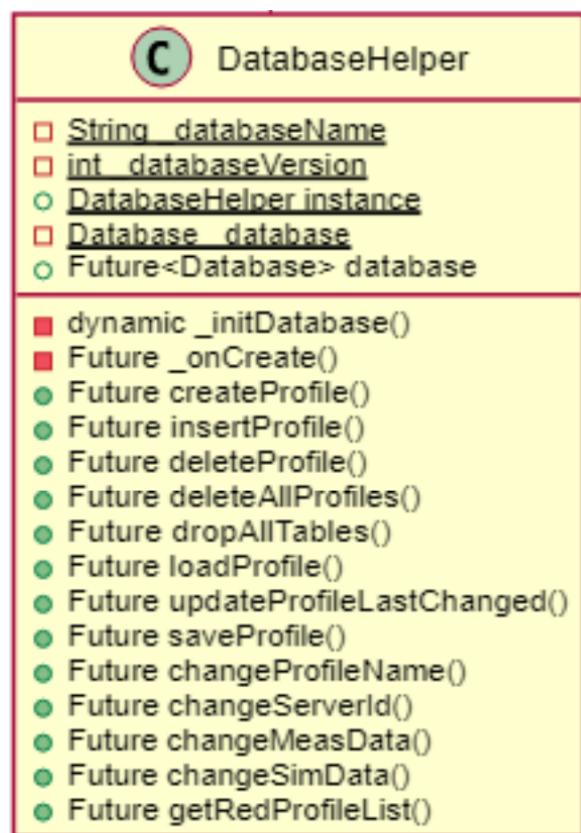


Abbildung 6.2: Aufbau der Hilfsklasse (*database_helpers.dart*)

6.2 Speicherung der Einstellungen

Die Daten zur Einstellung der App bestehen aus Präferenzen zur Sprache und zur Eingabemethode. Diese werden mithilfe des *Shared preferences* Plugins (<https://flutter.dev/docs/cookbook/persistence/key-value>) persistent gespeichert. Da diese Daten nur sehr wenig Speicherplatz benötigen und eine Speicherung mithilfe von Shared preferences sehr einfach ist, wurde hierfür diese Technik gewählt.

6.3 Verwaltung des App-States

Um den aktuellen Zustand der App über mehrere Ansichten verteilt nutzen zu können, wurde das *scoped_model* (https://github.com/brianegan/scoped_model) genutzt. Dieses ermöglicht es, ein aktuelles Modell des Zustandes an seine Kinderelemente weiterzugeben. Außerdem kann mit der statischen Methode **ScopedModel.of** das aktuelle Modell gefunden werden.

Der Zustand, der in der App verwaltet werden muss, ist das aktuell ausgewählte Profil. In der Profilverwaltung werden lediglich Daten in der SQLite Datenbank verändert, dies verändert also noch nicht den Zustand der App. Sobald ein Nutzer ein Profil als aktuelles Profil auswählt, werden die Daten des entsprechenden Profils aus der Datenbank geholt und in den App-State geladen. Dadurch ist dann das aktuelle Profil auf jeder weiteren Ansicht verfügbar und kann genutzt werden. Weitere Daten sind für den Zustand der App im App-State nicht nötig.

6.4 Daten hinzufügen

Um einem Profil neue Messdaten hinzuzufügen, können zwei Methoden genutzt werden:

1. **Scannen eines QR-Codes:** Der Nutzer scannt einen QR-Code ein, in dem jegliche Messdaten in Form eines JSON-Objektes encodiert sind. Zum Scannen und verarbeiten des QR-Codes wurde die Bibliothek *barcode_scan* (https://pub.dev/packages/barcode_scan) verwendet.
2. **Laden der Daten vom Server:** Die Daten der Messung können des weiteren von einem Server geladen werden. Hierzu wird ein HTTP-Get-Request an den entsprechenden Server geschickt, der ebenfalls ein JSON-Objekt mit den Messdaten zurückgibt.

6.5 Diagramme

Um die Diagramme in der App anzeigen zu können wurde die Bibliothek *charts_flutter* (https://github.com/google/charts/tree/master/charts_flutter) benutzt. Diese ermöglicht es Diagramme darzustellen. Bei der Umsetzung der App wurden einige dieser Diagramme ausgewählt und in den einzelnen Diagrammdarstellung genutzt. Dabei wurden Achsen und Beschriftungen, wie in der Bibliothek beschrieben, individualisiert. Diese Diagrammdarstellungen können als Bausteine für die Diagrammseite gesehen werden, da jede Darstellung über eine *Row* zugibt, die den jeweiligen Baustein für die Diagrammseite darstellt. Dabei erben alle diese Diagrammdarstellungen von der Klasse *ChartFactory*. Die *ChartFactory* stellt Methoden bereit, um einheitliche Abstände und Überschriften zu gewährleisten.

Es wurden zwei Diagrammdarstellungen programmiert. Das *simOverviewChart* und das *singleMeasChangeChart*. Das *simOverviewChart* gibt eine Übersicht über die prozentuellen Veränderungen der Kennzahlen an. Dabei werden die Kennzahlen mit der betragsmäßig größten prozentuellen Veränderung genommen und angezeigt. Das *singleMeasChangeChart* hingegen zeigt nur die Veränderung einer Kennzahl an. Dabei werden die absoluten Werte vor und nach der Simulation, sowohl als Zahl als auch als Säulendiagramm dargestellt.

Um alle Diagrammdarstellungen zu ordnen, existiert die Klasse *ChartInitializer*. Diese nutzt die einzelnen Diagrammdarstellungen, um daraus die Diagrammseite bauen zu können. Damit alle Diagramme auch auf Smartphones verschiedener Displaygrößen laufen kann, wird ein *LayoutBuilder* verwendet.