

Entwicklung einer App zur Analyse des Fahrverhaltens von PKW-Fahrern

Studienarbeit

Studiengang Informatik

Duale Hochschule Baden-Württemberg Karlsruhe

Joscha Heid

Max Joch

Eingereicht am: 13.01.2026

Matrikelnummer, Kurs: 7654321, TINF23B6
2298512, TINF23B6

Betreuer an der DHBW: Prof. Dr. Roland Schätzle

Inhalt

1	Einleitung	1
1.1	Umfeld	1
1.2	Motivation	2
1.3	Problemstellung	2
1.4	Aufgabenstellung	2
1.4.1	Ziele	2
1.4.2	Nicht-Ziele	3
1.5	Vorgehensweise	3
2	Namensgebung und Logo-Findung	4
3	Fahrverhalten und Ablenkung	6
3.1	Analyseverfahren für Fahrstildaten	6
3.1.1	Statistische Kennzahlen	6
3.1.2	Segmentierung der Strecke	7
3.1.3	Relevante Kennzahlen in der Fahrstilanalyse	7
3.2	Ablenkung von Autofahrern durch mobile Endgeräte und ihre Auswirkungen auf DriveWise Live-Feedback	8
4	Smartphone Sensorik	10
4.1	Global Positioning System (Global Positioning System (GPS))	10
4.2	Bewegungssensoren und Fahrzeugdynamik	10
4.3	Abtastrate und Energieverbrauch	11
5	iOS App Entwicklung	12
5.1	iOS App Entwicklung mit Swift	12
5.2	Zugriff auf Sensorik unter iOS	13
5.3	Lokale Speicherung und Datenformat	13
5.4	Benutzeroberfläche mit SwiftUI	13
5.5	Verwendung von Firebase Authentication für Kontofunktionen	14
6	Gamification als Motivator zur Steigerung von App-Nutzung	15
7	User Experience (User Experience (UX))	17
7.1	UX im Kontext der Fahrstilanalyse-App	18
7.2	Figma	19
8	Projektmanagement	20

8.1	Projektorganisation	20
8.2	Anforderungsanalyse	21
8.2.1	Vorgehensweise der Anforderungsanalyse im Projekt DriveWise	22
8.3	Projektplan	24
8.4	Qualitätssicherungsmaßnahmen	24
8.4.1	Allgemeine QS-Maßnahmen	24
8.4.2	Spezifische QS-Maßnahmen nach Bereichen und Meilensteinen	25
9	Umsetzung	27
9.1	Vorgehensweise	27
9.2	Benchmarking bestehender Lösungen zur Fahrverhaltensanalyse	27
9.2.1	Telematik im automobilen Kontext	27
9.2.2	Telematik-basierte Systeme mit externer Hardware (Versicherungstarife)	28
9.2.3	Smartphone-basierte Fahrten-Tracker ohne explizite Fahrstilanalyse. ...	28
9.2.4	Einordnung der eigenen Lösung	29
9.3	Entwicklung	31
9.3.1	Prototyping mit Figma	31
9.4	Umsetzung des UX-Prototyps in SwiftUI	37
9.4.1	Navigationsstruktur mit TabView	37
9.4.2	Startseite: Mapping des Figma-Layouts nach SwiftUI	39
9.4.3	Fahrtenübersicht und Fahrtdetails	39
9.4.4	Statistik- und Erfolgsansichten	42
9.4.5	Zwischenfazit zur User Interface (UI)-Implementierung	45
10	Code	47
10.1	Features einbauen	47
11	Anlysen (Joscha, Max)	48
12	Gamification (Joscha)	49
13	Testen	50
14	Schlussbetrachtung	51
14.1	Learnings	51
14.2	Fazit	51
14.3	Ausblick	51
A	Literatur	52

B	Glossar	57
C	Anhang	60
C.1	UX-Prototyp in Figma	60
C.2	A.1 Einstiegs- und Authentifizierungsansichten	60
C.3	A.2 Profil- und Kontoverwaltung	63



Einleitung

Diese Studienarbeit wird im 5. und 6. Semester an der Dualen Hochschule Baden-Württemberg ([Duale Hochschule Baden-Württemberg \(DHBW\)](#)) in Karlsruhe durchgeführt und trägt den Arbeitstitel „Entwicklung einer App zur Analyse des Fahrverhaltens von PKW-Fahrern“. Ziel ist es, Fahrdaten mit dem Smartphone zu erfassen, auszuwerten und verständlich aufzubereiten. Die App kombiniert Standort- und Bewegungsdaten, erkennt auffällige Fahrmanöver (z. B. starkes Beschleunigen, abruptes Bremsen, aggressives Kurvenfahren) und gibt kontextbezogenes Feedback – sowohl während der Fahrt als auch in einer nachträglichen Zusammenfassung.

1.1 Umfeld

Die Arbeit liegt im Bereich mobiler Sensorik und nutzerzentrierter Gestaltung. Die geplante Umsetzung erfolgt auf [iOS](#) und nutzt etablierte System-Schnittstellen wie [Core Location](#) (für Standortdaten über das Global Positioning System (GPS)) und [Core Motion](#) (für Bewegungsdaten wie [G-Sensor](#) und [Gyroskop](#)). Zusätzlich werden Anforderungen an Datenschutz und [Datensparsamkeit](#) berücksichtigt, damit die Auswertung nachvollziehbar bleibt und nur notwendige Daten verarbeitet werden. Ein [UX-Konzept](#) dient dabei als Grundlage für Navigation, Zustände (Start/Stopp, Fahrt aktiv, Auswertung) sowie Hinweise zu Fehlerfällen und Datenschutz.

1.2 Motivation

Fahrverhalten wirkt sich unmittelbar auf Sicherheit, Komfort und Umwelt aus. Gleichzeitig erhalten viele Fahrer wenig objektives, zeitnahes Feedback. Eine rein smartphonebasierte Lösung senkt die Einstiegshürden, da keine zusätzliche Fahrzeughardware nötig ist. Die App soll verständliche, situationsnahe Hinweise liefern und so Selbstreflexion fördern (z. B. „zu starkes Bremsen/ Beschleunigen“) – mit dem Ziel ein bewussteren und sichereren Fahrstils zu fördern. Auf diese Weise können Fahrer ihre Fahrweise eigenverantwortlich verbessern, was zu mehr Sicherheit und Umweltbewusstsein beiträgt. Besonders jüngere Fahrer, die häufig Smartphones nutzen, profitieren von objektivem Feedback, da in dieser Zielgruppe oft noch Unsicherheiten und wenig Fahrerfahrung bestehen.

1.3 Problemstellung

Bestehende Systeme sind häufig an fahrzeuginterne Hardware gebunden (z. B. [On-Board Diagnostics \(OBD\)](#)), flottenorientiert und damit für Privatanutzer überdimensioniert oder liefern schwer interpretierbare Kennzahlen ohne klare Handlungsempfehlung. Gesucht ist eine leicht zugängliche App, die ohne zusätzliche Fahrzeugtechnik auskommt, während der Fahrt robust läuft, Ereignisse zuverlässig erkennt und verständliche Rückmeldungen gibt – bei maximaler Transparenz und [Datensparsamkeit](#).

1.4 Aufgabenstellung

Gegenstand der Arbeit ist die Konzeption, prototypische Implementierung und Evaluation einer App zur Fahrstilanalyse. Die App soll Fahrten aufzeichnen, auffällige Manöver erkennen und die Ergebnisse so darstellen, dass Nutzende sowohl während der Fahrt (dezent und optional) als auch im Nachgang konkrete Hinweise erhalten. Die prototypische Umsetzung erfolgt auf [iOS](#).

1.4.1 Ziele

- Die App erfasst kontinuierlich und energieeffizient Standort- und Bewegungsdaten (u. a. [GPS](#)) und speichert diese zuverlässig über eine robuste Start-/Stopp-Logik (inklusive geeignetem [Logging](#)).

- Die App erkennt auffällige Ereignisse wie starke Beschleunigungen, Bremsungen und Kurvenfahrten mithilfe nachvollziehbarer, erklärbarer Regeln (z. B. Schwellwerte als [Heuristik](#)), sodass Ergebnisse für Nutzende transparent bleiben.
- Die App bietet optionales Live-Feedback mit geringer Ablenkung und stellt nach der Fahrt eine verständliche Zusammenfassung bereit, in der Ereignisse markiert und relevante Kennzahlen sowie Trends dargestellt werden.
- Die Interaktionsflüsse sind konsistent umgesetzt und orientieren sich am vorhandenen [Figma](#)-Design; erste Nutzertests liefern dokumentierte Erkenntnisse zur Gebrauchstauglichkeit.

1.4.2 Nicht-Ziele

- Eine Fahrzeugdiagnose oder eine Integration über [OBD](#) bzw. [Controller Area Network \(CAN\)](#) ist nicht Bestandteil der Arbeit, um die Lösung bewusst hardwareunabhängig und niedrigschwellig zu halten.
- Ein versicherungsrelevantes [Scoring](#) oder eine risikobewertende Analyse auf Basis von [Machine Learning \(ML\)](#) ist nicht Teil des Projekts, da der Fokus auf Transparenz, Nachvollziehbarkeit und Nutzerfeedback liegt.

1.5 Vorgehensweise

Die Arbeit gliedert sich in einen theoretischen Teil, die Konzeption und prototypische Umsetzung der App sowie eine abschließende Evaluation. Zunächst werden die relevanten Grundlagen kapitelweise erarbeitet. Darauf aufbauend werden Anforderungen abgeleitet, der Prototyp implementiert und anschließend durch Tests und erste Nutzungsrückmeldungen bewertet. Den Abschluss bildet eine Zusammenfassung mit Grenzen und Ausblick.



Namensgebung und Logo-Findung

Um den Namen und das Logo für die App passend zu wählen wurde sich im Projektteam getroffen und eine Brainstorming Session betrieben. Hierbei wurde sich auf den Namen „DriveWise“ geeinigt. Der Name wurde gewählt, um gleichzeitig den Fahrkontext der App hervorzuheben („Drive“ übersetzt „Fahren“), als auch das Ziel der App mitaufzunehmen, nämlich die angestrebte Verbesserung der Fahrweise („Wise“ übersetzt „weise“). Der Nutzer soll also eine weise und somit sichere und intelligente Fahrweise erlangen, da er auf Problemzonen seiner Fahrweise aufmerksam gemacht wird. Der Name soll dies direkt in Grundzügen beinhalten und deutlich machen.

Für das Logo wurde das [Künstliche Intelligenz \(KI\)](#)-Tool ChatGPT verwendet um einen geeigneten Entwurf zu generieren. Auch das Logo soll die gleiche Idee aufgreifen, wie schon davor der gewählte Name. Der untere Teil des Logos stellt eine Straße dar, um den bereits erwähnten Fahrkontext hervorzuheben. Der obere Teil fokussiert sich auf den Kontext der Fahrweise. Die Eule ist weit verbreitet für Weisheit, Aufmerksamkeit und Intelligenz bekannt. Alles Eigenschaften, die für ein verantwortungsvolles Fahren wichtig sind. Die Eule schaut auf dem Logo wachend über die Straße, was verdeutlichen soll, dass der Nutzer durch eine verbesserte Fahrweise sicherer fährt.



Abbildung 1 – DriveWise Logo

Die Kombination aus Name und Logo symbolisiert somit deutlich Sinn und Zweck der App und vermittelt ein einprägsames Bild, an dem ein Nutzer sich schon selbst grob ableiten kann, für was die App gedacht ist.



Fahrverhalten und Ablenkung

3.1 Analyseverfahren für Fahrstildaten

Die Analyse von Fahrstildaten dient der objektiven Bewertung des Fahrverhaltens auf Basis messbarer Größen. Durch die systematische Auswertung von Sensordaten können Rückschlüsse auf Sicherheit, Effizienz und Umweltverträglichkeit gezogen werden. Hierzu werden sowohl klassische statistische Kennzahlen als auch segmentierte und maschinell ausgewertete Datensätze eingesetzt.

3.1.1 Statistische Kennzahlen

Zu den grundlegenden Metriken zählen Durchschnitts- und Spitzengeschwindigkeit, Beschleunigungs- und Bremsvorgänge, Dauer einzelner Fahrten sowie G-Kraft-Spitzen während Kurven-, Beschleunigungs- oder Bremsmanövern. Diese Parameter erlauben eine quantitative Einschätzung des Fahrverhaltens in Bezug auf Fahrdynamik und Verkehrssicherheit ([1]).

Nach Angaben der YellowFox GmbH werden insbesondere starke Beschleunigungs- und Bremsvorgänge, abrupte Lenkmanöver sowie Geschwindigkeitsüberschreitungen als Indikatoren für einen „aggressiven“ Fahrstil gewertet ([1]). Auch in wissenschaftlichen Ansätzen werden solche Schwellenwerte herangezogen, um Fahrverhalten zu kategorisieren – beispielsweise die Häufigkeit starker Querschleunigungen oder die Varianz der Geschwindigkeit ([2]).

Neben der Ermittlung von Einzelwerten erfolgt häufig eine Normierung der Messgrößen über die gesamte Strecke, um verschiedene Fahrten vergleichbar zu machen. Eine hohe Varianz in der Beschleunigung deutet auf ein inkonsistentes Fahrverhalten hin, während gleichmäßige Werte auf defensive Fahrweise schließen lassen ([3]).

3.1.2 Segmentierung der Strecke

Für eine differenzierte Bewertung werden Fahrten häufig in Teilsegmente unterteilt, die jeweils ein homogenes Fahrverhalten aufweisen. Diese Segmentierung kann zeit- oder distanzbasiert erfolgen, etwa in Abschnitten von Sekunden oder Metern. Je kleiner die Segmente, desto präziser kann das Fahrverhalten analysiert werden. Alternativ kann die Segmentierung auch auf Veränderungen in Geschwindigkeit, Richtung oder Beschleunigung basieren, um Phasen gleichartigen Fahrverhaltens zu identifizieren ([4]).

Die Technische Hochschule Ingolstadt beschreibt im Projekt Fahrerverhaltens-Schätzer, dass durch Segmentierung auffällige Fahrabschnitte automatisch erkannt und klassifiziert werden können, etwa starkes Beschleunigen vor Kurven oder abrupte Bremsungen vor Ampeln ([5]). Solche Methoden erlauben eine gezielte Auswertung einzelner Situationen anstatt nur summarischer Werte über die gesamte Fahrt.

3.1.3 Relevante Kennzahlen in der Fahrstilanalyse

Zur objektiven Bewertung des Fahrverhaltens werden in der Praxis und Forschung typischerweise folgende Kennzahlen erhoben:

- **Durchschnittsgeschwindigkeit (\bar{v}):** misst das generelle Fahrtempo über die Strecke.
- **Spitzengeschwindigkeit (v_{\max}):** identifiziert Geschwindigkeitsüberschreitungen und risikoreiches Verhalten.
- **Beschleunigung (a):** erfasst Längsdynamik, etwa starkes Beschleunigen oder Abbremsen.
- **Bremsintensität (a_{neg}):** starke negative Beschleunigung als Indikator für abruptes Bremsen.
- **Querbeschleunigung (a_{lat}):** misst Kurvendynamik und Stabilität bei Richtungswechseln.
- **G-Kraft-Spitzen:** charakterisieren dynamische Manöver (z. B. schnelle Kurvenfahrt).

- **Fahrzeit / Stillstandszeit:** zeigt Anteil aktiver Fahrphasen und Leerlauf.
- **Fahrdistanz:** zur Relativierung anderer Kennzahlen.
- **Varianz der Geschwindigkeit und Beschleunigung:** Indikator für gleichmäßiges oder hektisches Fahrverhalten.
- **Fahrerevents:** Anzahl harter Bremsungen, Beschleunigungen oder Kurvenvorgänge pro Kilometer.

Diese Kennzahlen bilden die Grundlage für die algorithmische Bewertung des Fahrstils und sind Voraussetzung für weiterführende Analysen, etwa Klassifikation oder Feedback-Systeme ([1]).

3.2 Ablenkung von Autofahrern durch mobile Endgeräte und ihre Auswirkungen auf DriveWise Live-Feedback

Smartphones werden heutzutage für eine Vielzahl von Anwendungszwecken während der Fahrt genutzt, sei es zum Hören von Musik oder zur Nutzung eines Navigationssystems, aber auch zum Telefonieren. Diese Nutzung stellt laut [World Health Organization \(WHO\)](#) ein zunehmend ernstzunehmendes Problem dar. Laut eigener Aussage verwenden 60% bis 70% der Autofahrer ein Smartphone zu irgendeinem Zeitpunkt während der Fahrt ([6] S.15). Die Wahrscheinlichkeit in einen Unfall verwickelt zu sein, wird durch die Nutzung eines Smartphones rund um das vierfache erhöht. Hierbei zählt sogar das Nutzen einer Freisprechanlage zu den Verursachern und nicht nur das direkte halten und benutzen des Smartphones mit den Händen ([6] S. 3-4). Dies liegt vorallem daran, dass zur Hauptaufgabe des Fahrens eine sekundäre Aufgabe hinzukommt, nämlich das Verwenden des Smartphones in beliebiger Art und Weise. Die Aufmerksamkeit des Fahrers steht also nicht mehr uneingeschränkt für die Hauptaufgabe zur Verfügung sondern wird sich mit der sekundären Aufgabe geteilt und beeinflusst somit das Ausführen der Hauptaufgabe ([6] S. 7).

Hierbei wird in der Wissenschaft zwischen verschiedenen Arten der Ablenkung unterschieden ([7] S. 4):

- **Visuelle Ablenkung** liegt vor, wenn der Fahrer die Augen von der Straße entfernt, beispielsweise um eine Nachricht auf seinem Handy zu lesen, die auf dem Display erscheint.
- Bei der **Manuellen Ablenkung** nimmt der Fahrer seine Hände vom Lenkrad um direkt mit dem Smartphone zu interagieren, um beispielsweise auf eine Textnachricht zu antworten.

- **Kognitive Ablenkung** beschreibt das bereits zuvor beschriebene Phänomen, dass die Aufmerksamkeit des Fahrers auf eine sekundäre Aufgabe gelenkt wird, die in vielfältigen Variationen durch das Smartphone auftreten können. Der Fahrer kann hierbei noch auf die Straße schauen und auch die Hände am Lenkrad haben, verwendet jedoch nicht mehr die volle Aufmerksamkeit für das Fahren. Hierzu kann zum Beispiel das Verwenden einer Freisprechanlage sein, da der Fahrer sich auf das Gespräch konzentriert.

Da unser Live-Feedback den Fahrer während der Fahrt auf seine aktuelle Fahrweise aufmerksam machen soll, gilt es, die oben genannten Ablenkungen zu minimieren, um eine optimale Sicherheit zu gewährleisten. Im Folgenden wird erläutert, durch welche Maßnahmen die drei Ablenkungsarten minimiert werden sollen.

- Um die **Visuelle Ablenkung** zu reduzieren haben wir uns dazu entschieden, eine reine farbliche Darstellung zu verwenden. Der Nutzer bekommt keine ausformulierten Sätze oder Diagramme während der Fahrt angezeigt, sondern lediglich eine farbliche Einordnung seiner aktuellen Fahrweise. Gleichzeitig muss der Nutzer nicht aktiv auf sein Handy schauen, da er die aktuelle Farbe aus dem Augenwinkel bereits sehen kann.
- Die **Manuelle Ablenkung** wollen wir reduzieren, indem wir dem Kunden durch das Live-Feedback keine Möglichkeiten zur Interaktion geben. Das Feature wird nicht anklickbar sein, sondern nur eine Anzeige darstellen. Auch blockieren wir während eine Fahrt getrackt wird alle anderen Interaktionen mit der App, um dem Nutzer so gut es geht die Möglichkeit zu nehmen während der Fahrt aktiv mit unserer App zu interagieren.
- Die **Kognitive Ablenkung** versuchen wir durch ein einfaches Ampelsystem zu minimieren. Der Nutzer sieht im Augenwinkel einen grünen, gelben oder roten Bildschirm, je nach aktueller Fahrweise. Die Farbkodierung am Beispiel einer Verkehrsampel ist bei Autofahrern bereits angelernt und erfordert somit eine kleinstmögliche kognitive Leistung. Der Nutzer erhält sofort die Information anhand der aktuellen Farbe und muss dafür nicht nachdenken oder eine eigene Einschätzung durchlaufen.

4

Smartphone Sensorik

Smartphones verfügen über mehrere integrierte Sensoren, mit denen sich physikalische Größen wie Position, Geschwindigkeit und Beschleunigung erfassen lassen. Diese Sensorik bildet die technische Grundlage vieler Mobilitäts- und Analyseanwendungen. Für eine Fahrstilanalyse sind insbesondere satellitengestützte Positionsdaten sowie Bewegungsdaten aus Beschleunigungs- und Drehsensoren relevant.

4.1 Global Positioning System (GPS)

Das Global Positioning System (GPS) ist ein satellitengestütztes System zur Positionsbestimmung. Ein Empfänger (z. B. das Smartphone) berechnet seine Position aus Laufzeit- bzw. Zeitdifferenzen von Signalen mehrerer Satelliten. Aus aufeinanderfolgenden Positionen lassen sich Strecke und Geschwindigkeit ableiten ([8], S. 4 f.). In der Praxis wird die Genauigkeit durch verschiedene Einflüsse begrenzt, etwa Abschattungen, Signalreflexionen und Mehrwegeeffekte in „urbanen Schluchten“ sowie atmosphärische Störungen. Dadurch kann die Präzision insbesondere in Städten deutlich sinken und es kommt zu sprunghaften Messwerten (sog. Drift) ([9], S. 15 ff.). Für Anwendungen, die Fahrmanöver erkennen sollen, ist daher eine robuste Verarbeitung (z. B. Glättung, Plausibilitätsprüfungen) notwendig.

4.2 Bewegungssensoren und Fahrzeugdynamik

Neben GPS liefern Bewegungssensoren zusätzliche Informationen zur Dynamik. Beschleunigungssensoren messen lineare Beschleunigungen, während Gyrosko-

pe Rotationsraten erfassen. Aus diesen Signalen lassen sich Manöver ableiten: Hohe positive Längsbeschleunigungen können auf starkes Beschleunigen, negative Werte auf Bremsvorgänge und erhöhte Querschleunigungen auf dynamische Kurvenfahrten hindeuten ([10], S. 1971 ff.). Bewegungsdaten sind jedoch ebenfalls fehlerbehaftet (z. B. Rauschen, Bias). Eine Kombination mehrerer Quellen im Sinne der **Sensorfusion** kann die Robustheit erhöhen, weil Ausfälle oder Messfehler einer Quelle durch die andere teilweise kompensiert werden ([11], S. 37 ff.; [12], S. 2422).

4.3 Abtastrate und Energieverbrauch

Die Erfassung hochfrequenter Sensorwerte verbessert zwar die zeitliche Auflösung, erhöht jedoch den Energieverbrauch. Für mobile Anwendungen ist daher ein Kompromiss zwischen Messqualität und Batterieverbrauch erforderlich. Untersuchungen zeigen, dass moderate Genauigkeitseinstellungen häufig ein gutes Verhältnis zwischen Energiebedarf und Nutzbarkeit bieten ([8], S. 4 f.). Für eine spätere Auswertung kann zudem eine Segmentierung der Zeitreihen in kurze Zeitfenster sinnvoll sein, da sie Analyse und Visualisierung erleichtert ([4]).



iOS App Entwicklung

5.1 iOS App Entwicklung mit Swift

Die prototypische Umsetzung der Fahrstilanalyse-App erfolgt auf Basis des Betriebssystems **iOS** und der Programmiersprache **Swift**. **Swift** wurde von Apple im Jahr 2014 eingeführt und hat sich als Standard für die Entwicklung nativer Anwendungen auf **iOS**, **iPadOS**, **macOS** und **watchOS** etabliert ([13]). Die Sprache bietet eine moderne Syntax, hohe Typsicherheit und gute Performance und wurde mit dem Ziel entwickelt, sichere und effiziente Programmierung zu unterstützen ([14]).

Swift kombiniert Eigenschaften kompilierter Sprachen mit gut lesbarer Syntax und unterstützt objektorientierte, funktionale und protokollbasierte Paradigmen. Mechanismen wie striktere Typprüfung und optionale Typen reduzieren typische Fehler (z. B. Nullzeiger-Probleme) und erhöhen Stabilität und Sicherheit der Anwendung ([14]).

Die Entwicklung erfolgt in der integrierten Entwicklungsumgebung (Integrated Development Environment, **Integrierte Entwicklungsumgebung (IDE)**) **Xcode**, die Werkzeuge für Code-Erstellung, Debugging und Performance-Analyse bereitstellt. Zusätzlich können Benutzeroberflächen gestaltet und mit Logik verknüpft werden, was eine konsistente Umsetzung des im **Figma**-Prototypen definierten Designs unterstützt ([15]).

5.2 Zugriff auf Sensorik unter iOS

Für die Datenerfassung nutzt die App primär die Frameworks [Core Location](#) und [Core Motion](#) ([16]; [17]). Standortdaten werden über [Core Location](#) bereitgestellt, typischerweise über Klassen wie `CLLocationManager` und `CLLocation` ([16]; [18]). Einstellungen wie `desiredAccuracy` und `distanceFilter` steuern Genauigkeit und Aktualisierungsverhalten und beeinflussen damit auch den Energieverbrauch ([18]). Für Fahrten, die im Hintergrund weiterlaufen, kann die Standortaktualisierung über `allowsBackgroundLocationUpdates` fortgesetzt werden, sofern die App entsprechend konfiguriert ist ([16]).

Bewegungsdaten werden über [Core Motion](#) erfasst, z. B. über `CMMotionManager` oder `CMDeviceMotion` ([17]). Durch systemnahe Verarbeitung und (je nach Quelle) Sensorfusion stehen häufig stabilere Signale zur Verfügung, die für eine robuste Ereigniserkennung hilfreich sind ([11], S. 37 ff.). Insgesamt ermöglichen die nativen Frameworks eine ressourcenschonende Erfassung, die für eine kontinuierliche Fahrtdatenerhebung erforderlich ist.

5.3 Lokale Speicherung und Datenformat

Die erfassten Daten werden lokal auf dem Gerät gespeichert, um Offline-Funktionalität zu gewährleisten und Anforderungen an Datenschutz sowie [Datensparsamkeit](#) zu unterstützen. Für Dateizugriffe innerhalb des App-Containers stellt [iOS](#) z. B. `FileManager` bereit ([19]). Die Speicherung kann in einem strukturierten Textformat wie JavaScript Object Notation ([JavaScript Object Notation \(JSON\)](#)) erfolgen, da es eine kompakte Serialisierung und eine einfache Weiterverarbeitung ermöglicht ([20], S. 160). Nach Abschluss einer Fahrt können Daten gebündelt und asynchron an eine Server- oder Webanwendung übertragen werden, um Energieverbrauch und Datenverlust zu reduzieren ([8], S. 3 f.).

5.4 Benutzeroberfläche mit SwiftUI

Für die Benutzeroberfläche kann [SwiftUI](#) genutzt werden, ein deklaratives Framework, das die Umsetzung responsiver und interaktiver Layouts unterstützt ([21]). Durch wiederverwendbare Komponenten lassen sich konsistente Ansichten erstellen und das [UX-Konzept](#) effizient umsetzen.

5.5 Verwendung von **Firebase Authentication** für Kontofunktionen

Für Kontofunktionen wird **Firebase Authentication** eingesetzt. **Firebase** ist eine Entwicklungsplattform von Google, die verschiedene Dienste bereitstellt, um Anwendungen zu erstellen und zu betreiben ([22]). **Firebase Authentication** unterstützt die schnelle Integration eines Benutzerverwaltungssystems mit vorgefertigten Authentifizierungsverfahren, z. B. E-Mail/Passwort, Telefonnummer sowie Anmeldungen über Google-Konto oder Apple-ID ([23]; [24]).

Für den Prototyp ist die kostenlose Stufe geeignet, da sie bis zu 50.000 monatliche aktive Nutzer ohne zusätzliche Kosten abdeckt; Einschränkungen bestehen dabei insbesondere bei SMS-basierter Authentifizierung ([25]). Da im Rahmen der Studienarbeit keine große Nutzerbasis erwartet wird, ist dieser Umfang ausreichend. Dadurch kann ein realistisches Kontosystem umgesetzt werden, ohne eigene Backend-Infrastruktur entwickeln und betreiben zu müssen.



Gamification als Motivator zur Steigerung von App-Nutzung

Gamification beschreibt den Prozess, Videospiel-ähnliche Elemente, wie zum Beispiel Herausforderungen, Abzeichen oder Level-Systeme mit gewissen Belohnungen, in einem Umfeld zu verwenden, das selbst nichts mit Videospielen zu tun hat ([26] S.371). Man will den Nutzer ähnlich durch Belohnung motivieren, wie es Videospiele seit langem tun und ihn so an das eigene Produkt binden ([26] S.372).

Studien zeigen, dass Gamification die Nutzerbindung steigert. Nicht nur durch Belohnungen wie Abzeichen oder Punkte, sondern auch durch Ansporn mit Hilfe von Challenges. Diese geben dem Nutzer die Möglichkeit Fortschritt zu erzielen und aktiv zu sehen, inwiefern er sich in gewissen Feldern verbessert hat und vorangekommen ist. Dies steigert sein Gefühl von Kompetenz ([27] S.181). Es wird ebenfalls hervorgehoben, dass es von Vorteil sein kann, Challenges mit voranschreitender Schwierigkeit zu implementieren, damit der Nutzer seinen eigenen Fortschritt in Echtzeit verfolgen und einsehen kann und stets eine neue Herausforderung erhält, um nicht von gleichbleibender Schwierigkeit gelanweilt zu werden ([27] S.181).

Die Grundlage hierfür findet sich in der **Motivationspsychologie**. Durch positives Feedback, beispielsweise in Form von Belohnungen in Videospielen, wird

die intrinsische Motivation des Nutzers gesteigert ([28] S.70). Gleichzeitig trägt dieses Feedback auch dazu bei, dem Nutzer ein gewisses Gefühl von Kompetenz und Bestätigung zu vermitteln was wiederum seine Motivation für die betroffene Handlung steigert ([28] S.70).

Die Self-Determination Theory (SDT) beschreibt die Motivation eines jeden Menschen als ein nicht starres Konstrukt, welches stark von äußeren Einflüssen abhängen kann ([28] S.68). Ryan und Deci leiten hierzu drei grundlegende Bedürfnisse ab, die erfüllt sein müssen, um die Motivation eines Menschen zu steigern: Kompetenz, Autonomie und Soziale Eingebundenheit ([28] S.68). Die Einschränkung dieser Bedürfnisse führt hingegen zum gegenteiligen Effekt, also weniger Motivation.

Man spornt den Nutzer also durch die Erfüllung der oben genannten Bedürfnisse dazu an, eine größere Motivation für die Verwendung des eigenen Produkts zu entwickeln und somit eine stärkere Bindung und Nutzung zu bezwecken.



User Experience (UX)

User Experience (UX) ist ein zentrales Konzept der Mensch-Computer-Interaktion und beschreibt, wie Menschen die Nutzung eines Produkts, einer Dienstleistung oder eines Systems subjektiv erleben ([29], S. 327–337). Im Mittelpunkt stehen dabei nicht nur funktionale Aspekte wie Zuverlässigkeit, Effizienz und Bedienbarkeit, sondern auch emotionale Dimensionen wie Freude, Vertrauen oder Frustration. Eine positive UX entsteht, wenn beide Ebenen – funktional und emotional – im Einklang stehen und die Erwartungen der Nutzer erfüllt oder sogar übertroffen werden.

Das Ziel der UX-Gestaltung besteht darin, digitale Produkte so zu entwerfen, dass sie nicht nur zweckmäßig, sondern auch angenehm und intuitiv nutzbar sind ([30]). Eine gute UX steigert die Zufriedenheit, reduziert Fehlbedienungen und führt zu einer nachhaltig positiven Wahrnehmung der Anwendung ([31]).

Im Unterschied zur Benutzeroberfläche (User Interface, UI), die sich auf die sichtbaren Interaktionselemente konzentriert, betrachtet die UX den gesamten Nutzungskontext – also die Erfahrungen, die ein Nutzer vor, während und nach der Interaktion macht. Dadurch umfasst UX neben der visuellen Gestaltung auch Informationsarchitektur, Usability, Erwartungshaltung und emotionale Reaktionen ([29], S. 327–337).

Eine gelungene UX ist somit ein wesentliches Qualitätsmerkmal digitaler Systeme und trägt entscheidend zum Erfolg eines Produkts bei. Insbesondere bei mobilen Anwendungen, wie der in dieser Arbeit entwickelten iOS-App zur Fahrstilanalyse, beeinflusst eine positive UX das Vertrauen der Nutzer in die Genauigkeit der Analyse, die Bereitschaft zur kontinuierlichen Nutzung und letztlich die Akzeptanz des Gesamtsystems.

7.1 UX im Kontext der Fahrstilanalyse-App

Die User Experience (UX) spielt im Kontext der Fahrstilanalyse-App eine zentrale Rolle, da die Qualität der Nutzererfahrung maßgeblich über die Akzeptanz und langfristige Nutzung der Anwendung entscheidet. Das Ziel besteht darin, komplexe Fahrdaten so aufzubereiten, dass sie für den Nutzer verständlich, transparent und motivierend dargestellt werden.

Ein wesentlicher Aspekt ist dabei die Klarheit der Informationsdarstellung. Fahrverhaltensdaten wie Beschleunigung, Bremsintensität oder G-Kräfte müssen so visualisiert werden, dass sie auch ohne technisches Vorwissen nachvollziehbar sind. Eine übersichtliche Darstellung mit intuitiven Diagrammen und farblichen Hervorhebungen trägt dazu bei, dass Nutzer ihr Fahrverhalten schnell einordnen und daraus Handlungsoptionen ableiten können ([31]).

Darüber hinaus beeinflusst die emotionale Dimension der UX das Nutzungserlebnis entscheidend. Positives Feedback, Gamification-Elemente und Fortschrittsanzeigen fördern Motivation und Lernbereitschaft, während zu viele Warnungen oder negative Bewertungen Frustration auslösen könnten. Ein ausgewogenes UX-Design muss daher zwischen konstruktiver Rückmeldung und motivierender Belohnung differenzieren ([29], S. 332).

Auch die Benutzerführung spielt eine zentrale Rolle: Eine klare Informationsarchitektur, kurze Ladezeiten und eine einheitliche Navigation ermöglichen eine reibungslose Interaktion und reduzieren kognitive Belastung. Die App sollte sowohl während der Fahrt als auch in der Nachanalyse eine intuitive Bedienung bieten, um Ablenkungen zu vermeiden und die Sicherheit zu gewährleisten ([30]).

Schließlich trägt eine gute UX wesentlich zur Vertrauensbildung bei. Da die App Fahrverhaltensdaten aufzeichnet und auswertet, ist eine transparente Kommunikation über Datennutzung, Berechnungsmethoden und Datenschutz essenziell. Nur wenn Nutzer das Gefühl haben, dass ihre Daten sicher und sinnvoll verarbeitet werden, entsteht eine nachhaltige positive Nutzererfahrung.

Insgesamt stellt die UX der Fahrstilanalyse-App einen entscheidenden Erfolgsfaktor dar: Sie verbindet funktionale Präzision mit emotionaler Akzeptanz und macht technische Analysen zu einem verständlichen und motivierenden Erlebnis.

7.2 Figma

Figma ist ein cloubasiertes Design- und Prototyping-Tool, das sich insbesondere in der **UX**- und **UI**-Entwicklung als vielseitiges und kollaboratives Werkzeug etabliert hat ([32]). Es ermöglicht die gemeinsame Erstellung und Bearbeitung von Designentwürfen direkt im Browser, wodurch Entwickler, Designer und Stakeholder gleichzeitig an einem Projekt arbeiten können. Durch diese Echtzeit-Kollaboration wird die Kommunikation im Team vereinfacht und der Designprozess deutlich beschleunigt ([33]).

Besonders in agilen Entwicklungsprozessen unterstützt **Figma** die schnelle Iteration von Designideen und fördert eine enge Abstimmung zwischen Design und Umsetzung. Designsysteme können zentral gepflegt und konsistent über alle Ansichten hinweg angewendet werden, was die Qualität und Einheitlichkeit der Benutzeroberfläche verbessert ([32]).

In dieser Arbeit wird **Figma** als zentrales Werkzeug zur Entwicklung und Visualisierung des **UX**-Konzepts eingesetzt. Es dient dazu, Benutzerinteraktionen realitätsnah zu simulieren und frühes Feedback in den Gestaltungsprozess zu integrieren. Dadurch können Prototypen iterativ optimiert und potenzielle Usability-Probleme bereits vor der eigentlichen Implementierung identifiziert werden. **Figma** trägt somit wesentlich zur benutzerzentrierten Gestaltung der in dieser Arbeit entwickelten Fahrstilanalyse-App bei und unterstützt die Überführung theoretischer **UX**-Prinzipien in ein praxisnahes, interaktives Design.



Projektmanagement

8.1 Projektorganisation

Die Studienarbeit wird im Rahmen des 5. und 6. Semesters im Studiengang Informatik an der [DHBW](#) Karlsruhe durchgeführt. Der Bearbeitungszeitraum ist in zwei Präsenzphasen gegliedert: vom 29.09.2025 bis 21.12.2025 sowie vom 16.02.2026 bis 17.05.2026. Die Arbeit wird von Prof. Dr. Roland Schätzle betreut.

Das Projektteam besteht aus zwei Studierenden: Max Joch und Joscha Heid. Beide übernehmen sowohl Entwicklungs- als auch Designaufgaben und sind darüber hinaus in die Projektplanung eingebunden. Joscha Heid fungiert zusätzlich als Projektleiter und koordiniert Termine, Aufgabenverteilung und Kommunikation mit dem Betreuer. Die Betreuung durch Prof. Schätzle erfolgt insbesondere über regelmäßige Statusgespräche und Feedback zu Zwischenergebnissen.

Die Zusammenarbeit findet überwiegend in selbstorganisierten Arbeitssitzungen statt, die je nach Verfügbarkeit in Räumen der [DHBW](#) oder remote durchgeführt werden. Da Quellcode und Dokumentation vollständig digital vorliegen, ist eine ortsunabhängige Kollaboration problemlos möglich.

Zur Entwicklung wird die integrierte Entwicklungsumgebung (IDE) [Xcode](#) eingesetzt. Die App wird nativ in [Swift](#) bzw. [SwiftUI](#) für [iOS](#) umgesetzt. Für die Auswertung und Speicherung der Fahrdaten kommen die System-[Application Programming Interface \(API\)s](#) [Core Motion](#) und [Core Location](#) sowie eine lokale Datenhaltung (z. B. im [JSON](#)-Format) zum Einsatz. Die Studienarbeit selbst wird mit Typst erstellt, wodurch eine klare Trennung zwischen Inhalt und Layout sowie eine konsistente Zitations- und Abbildungsverwaltung ermöglicht wird.

Als zentrale Plattform für Versionsverwaltung und Zusammenarbeit dient ein privates GitLab-Repository. Dort werden Quellcode, Projektdokumente und Issues gemeinsam verwaltet. Die Nutzung von Git ermöglicht eine lückenlose Nachverfolgbarkeit von Änderungen, die parallele Bearbeitung durch beide Teammitglieder sowie eine strukturierte Organisation von Aufgaben (z. B. über Branches, Merge Requests und Issue-Boards).

Das Vorgehen im Projekt ist iterativ und an klassische Softwareentwicklungsprozesse angelehnt. Zunächst werden Anforderungen erhoben und konsolidiert (Anforderungsanalyse). Darauf aufbauend erfolgt eine technische Konzeption, in der Frameworks, Datenmodell, Speicherkonzept und grundlegende UX-Strukturen festgelegt werden. Anschließend werden die einzelnen Views auf UX Ebene umgesetzt, bevor die Datenerfassung (GPS-Tracking, Sensorintegration) mit den Analysealgorithmen implementiert und Feedbackmechanismen (Karten, Kennzahlen, Gamification) ausgearbeitet werden. Die Implementierung wird durch Testfahrten, Validierungsschritte und Usability-Tests begleitet und abschließend durch eine schriftliche Dokumentation mit Reflexion und Ausblick ergänzt.

Im Rahmen der Projektorganisation wurde zudem ein einfaches Risikomanagement etabliert. Zentrale Risiken – etwa eine eingeschränkte technische Umsetzbarkeit bestimmter Funktionen, zeitliche Verzögerungen, unzureichende Datenqualität, Krankheit von Teammitgliedern oder Änderungen in iOS-Frameworks – wurden identifiziert und hinsichtlich Eintrittswahrscheinlichkeit und Projekteinfluss bewertet. Daraus wurde ein Risiko-Faktor (Wahrscheinlichkeit \times Einfluss) abgeleitet und für jedes Risiko konkrete Minderungsmaßnahmen definiert (z. B. frühe Machbarkeitsanalysen, Zeitpuffer, frühzeitige Testfahrten, Dokumentation der Aufgabenverteilung). Dieses strukturierte Vorgehen unterstützt die Planungssicherheit und erleichtert den Umgang mit unerwarteten Projektverläufen.

8.2 Anforderungsanalyse

Die Anforderungsanalyse ist ein Schritt der Software-Entwicklung und umfasst grundlegend die Erfassung und Dokumentation der Anforderungen, die an ein Software-System gestellt werden. Durch eine solche Dokumentation wird sichergestellt, dass alle am Projekt Beteiligten auf dem selben Stand sind, was das erwartete Ergebnis betrifft ([34] S.102). Hierbei werden sowohl funktionale Anforderungen, als auch nicht funktionale Anforderungen (sogenannte Quality-Requirements) aufgenommen und festgehalten. Diese nicht-funktionalen Anforderungen gehen über funktionale Features hinaus und beschreiben oft Aspekte,

wie Performance oder Verfügbarkeit etc. ([35] S.8). Weiterführend können auch Nicht-Ziele definiert werden, die genauer abstecken, welche Features nicht Teil der Entwicklung sein werden.

All diese festgehaltenen Punkte dienen dazu, spätere Missverständnisse zu verhindern und das Risiko für zeitkritische oder kostspielige Änderungswünsche zu minimieren. Die Wahrscheinlichkeit wird deutlich erhöht, dass das Endprodukt den Vorstellungen aller zu Projektbeginn Beteiligten Personen entspricht ([36] S.12-13).

Das Lastenheft spielt bei diesem Prozess eine zentrale Rolle. Es beschreibt die Anforderungen und Vorstellungen aus Sicht des Auftraggebers. Hierbei ist wichtig, dass noch keine technische Beschreibung der Lösung aufgenommen wird. Diese Ausarbeitung wird erst später im Pflichtenheft erforderlich.

8.2.1 Vorgehensweise der Anforderungsanalyse im Projekt DriveWise

Da im Projekt DriveWise die Projektmitglieder gleichzeitig auch die Stakeholder sind, handelt es sich um selbstaufgelegte Anforderungen. Diese Anforderungen wurden gemeinsam festgelegt und anschließend in einem Lastenheft festgehalten. Hierbei wurden sowohl funktionale, als auch nicht funktionale Anforderungen erarbeitet.

Hierbei wurde damit begonnen die Hauptanforderungen an das Projekt in einem Satz gebündelt zu formulieren, um auf den ersten Blick einen guten Überblick zu haben. Anschließend wurden der Anwendungsbereich und die Zielgruppe definiert (Privatpersonen mit iOS-Gerät, die ihr Fahrverhalten verbessern wollen). Danach wurden die funktionalen Anforderungen definiert, darunter:

1. Nutzerregistrierung und Login:

Um individuelle Fahrdaten sicher speichern zu können, soll ein Login-System bereitgestellt werden, über das sich Nutzer mit einer Email Adresse und einem Passwort anmelden können. Nach erfolgreicher Anmeldung werden Daten Accountbasiert gespeichert, sodass eine individuelle Analyse, sowie Geräteübergreifendes Verwenden der App möglich ist. Für den Fall, dass ein Nutzer seine Daten vergisst, soll es eine einfache Möglichkeit geben, sein Passwort zurückzusetzen. Der gesamte Prozess sollte auch für technisch unerfahrene Nutzer einfach verständlich sein.

2. Erfassung von Fahr- und Sensordaten:

Die App soll iPhone-interne Sensoren verwenden, um Bewegungs- und Positionsdaten zu ermitteln. Diese Daten sollen gespeichert und als Grundlage für die Bewertung des Fahrstils verwendet werden. Die Analyse umfasst die Durchschnittsgeschwindigkeit und Fahrzeit, die Erkennung von starkem Beschleunigen, abruptem Bremsen und riskantem Kurvenverhalten, sowie G-Kraft-Spitzen und einer Segmentierung der Strecke in Abschnitte. Durch diese Daten soll dem Nutzer sichtbar gemacht werden, wie er seinen Fahrstil verbessern kann, da er anschaulich sieht, wo seine Problemzonen im Bereich des Straßenverkehrs liegen.

3. Echtzeit-Feedback während der Fahrt:

Während der Fahrt soll das Fahrverhalten in Echtzeit analysiert werden, sodass der Nutzer noch während der Fahrt vor gefährlichem Fahrverhalten gewarnt werden kann. Diese Warnung soll entweder visuell oder akustisch erfolgen, jedoch dezent und sicher, sodass der Fahrer nicht während der Fahrt abgelenkt wird. Mögliche Warnfälle umfassen zu starkes Beschleunigen, abruptes Bremsen oder auch übermäßige Kurvenbeschleunigung.

4. Auswertung einer Route nach der Fahrt inklusive Kartenansicht:

Nach jeder Fahrt soll der Nutzer eine detaillierte Auswertung über seine Fahrt erhalten. Hierbei soll es eine Kartenansicht geben, auf der auffällige Bereiche der Route farblich hervorgehoben werden. Zusätzlich soll der Nutzer statistische Kennzahlen erhalten, anhand derer ein Score für die einzelne Fahrt errechnet wird.

5. Generierung von Wochen- und/oder Monatsberichten:

Um die langfristige Entwicklung eines Nutzers zu erkennen soll die App automatisch wöchentliche oder monatliche Reports erstellen. Diese sollen als Zusammenfassung dienen, um dem Nutzer sein Fahrverhalten und mögliche Änderungen dessen anschaulich zu machen. Hierbei können die Änderungen positiv oder auch negativ sein.

6. Gamification-Elemente zur spielerischen Motivationssteigerung:

Um dem Nutzer eine spielerische Motivation zu geben, die App weiter zu verwenden und seinen Fahrstil zu verbessern, soll die App auf Gamification Elemente setzen. Hierzu zählen beispielsweise Fortschrittsbalken und Challenges, aber auch Belohnungen für ein gutes Fahrverhalten.

Darüber hinaus wurden nicht-funktionale Anforderungen an Usability und Design formuliert, die alle Altersklassen abdecken sollen.

1. Benutzerfreundlichkeit

Die App soll ein modernes, aber klares Design aufweisen mit einer intuitiven Navigation. Statistiken sollen klar veranschaulicht sein, statt lange Zahlenreihen zu verwenden. Hierdurch sollen Nutzer aller Altersklassen angesprochen werden, unabhängig von technischem Vorwissen.

Zu guter Letzt wurden Nicht-Ziele definiert, die explizit nicht Teil des Projekts sind. Diese enthalten folgende Punkte. Die App soll keine Verbindung zu OBD-Systemen oder anderen fahrzeuginternen Sensoren herstellen. Ebenfalls soll sie nicht dazu dienen, eine technische Fahrzeugdiagnose zu erhalten, da die App lediglich für den Fahrstil und nicht das Fahrzeug selbst gedacht ist. Auch soll es keine Integration mit anderen externen Geräten oder Fahrzeugsteuerungen geben.

8.3 Projektplan

in Arbeit (TODO)

8.4 Qualitätssicherungsmaßnahmen

Um die Qualität der entwickelten App sowie der Studienarbeit sicherzustellen, wurden sowohl allgemeine als auch phasen- und bereichsspezifische Qualitätssicherungsmaßnahmen definiert. Ziel ist es, funktionale Korrektheit, Stabilität, Benutzerfreundlichkeit, Sicherheit und Nachvollziehbarkeit der Ergebnisse systematisch abzusichern.

8.4.1 Allgemeine QS-Maßnahmen

Zu den übergreifenden Maßnahmen gehören regelmäßige Code-Reviews, bei denen Quellcode hinsichtlich Lesbarkeit, Struktur und Funktionalität überprüft wird. Die Nutzung von GitLab als Versionsverwaltungssystem stellt sicher, dass Änderungen versioniert, nachvollziehbar dokumentiert und bei Bedarf wiederhergestellt werden können.

Für alle wesentlichen Tests (Funktions-, Performance- und später Usability-Tests) werden Testprotokolle geführt, in denen Vorgehen, Testdaten, Ergebnisse und ggf. identifizierte Fehler festgehalten werden. Zusätzlich sind Feedbackrunden mit Kommiliton:innen sowie dem Betreuer vorgesehen, insbesondere in der Abschlussphase, um externe Perspektiven auf [UX](#), Verständlichkeit und Funktionalität zu integrieren.

Die Projektarbeit wird durch wöchentliche Team-Meetings begleitet, in denen Fortschritte, offene Punkte und Risiken besprochen werden. Monatliche Treffen mit dem Betreuer dienen der Abstimmung zentraler Entscheidungen und der Überprüfung des Projektstands. Als Qualitätskriterien werden u. a. die Erfüllung der Anforderungen, die Stabilität der App, die Nachvollziehbarkeit der Analyseergebnisse sowie die formale Korrektheit der Studienarbeit (z. B. Zitation, Quellenverzeichnis) festgelegt.

8.4.2 Spezifische QS-Maßnahmen nach Bereichen und Meilensteinen

Die Qualitätssicherung wird zusätzlich entlang definierter Meilensteine und Themenfelder konkretisiert.

Projekt & Zusammenarbeit (M1–M3). Zu Projektbeginn wird sichergestellt, dass alle Teammitglieder Zugriff auf das GitLab-Repository haben und grundlegende Git-Kenntnisse vorhanden sind. Es wird geprüft, ob das gemeinsame Arbeiten über Branches und Merge Requests funktioniert und eine aktuelle README mit Setup-Anleitung (z. B. [Xcode](#)-Version, benötigte Abhängigkeiten) vorliegt. Außerdem wird der Build-Prozess auf den Entwicklungsgeräten getestet, um eine konsistente Entwicklungsumgebung sicherzustellen.

UI & Design (M4–M5). In dieser Phase wird die Konformität der implementierten Oberfläche mit den [Figma](#)-Mockups überprüft. Dabei werden Farben, Icons, Layout, Abstände und die Funktionsweise von Navigation Bar und Tab Bar mit dem Prototyp abgeglichen. Zudem wird getestet, ob das Layout auf unterschiedlichen Gerätegrößen (z. B. iPhone SE bis iPhone 15 Pro Max) korrekt skaliert und ob Animationen sowie Übergänge flüssig sind. Falls ein Dark-Mode-Design vorgesehen ist, wird dessen Vollständigkeit ebenfalls geprüft.

Funktionalität & Logik (M4–M5). Für die fachliche Logik werden spezifische Tests definiert: Es wird kontrolliert, ob Fahrten korrekt gestartet, pausiert und beendet werden, ob [GPS](#)-Daten zuverlässig – auch im Hintergrundbetrieb – aufgezeichnet werden und ob die Logik zur Erkennung von Beschleunigungs-, Brems- und Lenkereignissen valide und reproduzierbar ist. Ebenso werden Ga-

mification-Funktionen (Punktesystem, Badges, Level) sowie Authentifizierungs- und Datenhaltungsmechanismen getestet.

Benutzererlebnis & Stabilität (M5–M6). In dieser Phase stehen realitätsnahe Tests im Vordergrund. Die App wird auf verschiedenen iOS-Geräten gestartet und auf Absturzfreiheit überprüft. Testfahrten unter realen Bedingungen (Stadt, Landstraße, Autobahn) dienen dazu, das Verhalten der App während längerer Nutzung zu bewerten und potenzielle Probleme hinsichtlich Speicherverbrauch, Energieeffizienz und Robustheit aufzudecken. Zusätzlich wird auf Zugänglichkeit und Lesbarkeit geachtet (Kontraste, Schriftgrößen, Touch-Ziele), um die App auch für unterschiedliche Nutzergruppen gut nutzbar zu machen.

todo muss noch geschaut werden ob wir das machen wollen

Sicherheit & Datenschutz (M5–M6). Da GPS- und Fahrdaten personenbezogene Informationen enthalten können, werden spezielle Maßnahmen zur Sicherstellung von Sicherheit und Datenschutz vorgesehen. Dazu gehören die Überprüfung von Zugriffsregeln (z. B. dass nur eingeloggte Nutzer auf ihre eigenen Daten zugreifen können), die Vermeidung unverschlüsselter Speicherung sensibler Daten und die Bereitstellung einer verständlichen Datenschutzerklärung mit Einwilligungsabfrage beim ersten Start. Zudem werden Prozesse für den Export bzw. die Löschung persönlicher Daten im Sinne der DSGVO dokumentiert und getestet.

Durch die Kombination aus organisatorischen Regelungen, klar definierten Meilensteinen und detaillierten Qualitätssicherungsmaßnahmen wird gewährleistet, dass die entwickelte App sowohl den fachlichen Anforderungen als auch den wissenschaftlichen und formalen Anforderungen der Studienarbeit gerecht wird.



Umsetzung

9.1 Vorgehensweise

9.2 Benchmarking bestehender Lösungen zur Fahrverhaltensanalyse

Das Benchmarking bestehender Lösungen dient dazu, die im Rahmen dieser Studienarbeit entwickelte Fahrstilanalyse-App im Markt- und Technologiekontext zu verorten. Untersucht werden insbesondere Anwendungen, die Fahrten aufzeichnen und Fahrverhalten bewerten. Dabei lassen sich zwei grundlegende Ansätze unterscheiden: telematikbasierte Systeme mit externer Hardware (insbesondere Versicherungsprodukte) und rein smartphonebasierte Tracking-Apps.

9.2.1 Telematik im automobilen Kontext

Der Begriff **Telematik** bezeichnet die Verbindung von Telekommunikation und Informatik zur Erfassung, Übertragung und Auswertung von Daten über räumlich verteilte Objekte ([37]; [38]). Im Fahrzeugbereich kommen typischerweise **GPS**-Empfänger, Beschleunigungssensoren und eine Kommunikationsschnittstelle (z. B. Mobilfunk) zum Einsatz, um Positions-, Bewegungs- und Zustandsdaten kontinuierlich zu erfassen und an Backend-Systeme zu übertragen ([37]).

In der Kfz-Versicherung wird Telematik im Rahmen von **Usage-Based Insurance** (**Usage-Based Insurance (UBI)**) eingesetzt. Fahrdaten wie gefahrene Kilometer,

Tageszeit, Position, starkes Beschleunigen, hartes Bremsen oder Kurvenfahrten werden über fest installierte Geräte oder Smartphone-Apps aufgezeichnet und in Echtzeit an den Versicherer bzw. dessen Dienstleister übermittelt ([39]; [38]). Auf Basis dieser Verhaltensdaten werden Risikoprofile erstellt und Prämien stärker am individuellen Fahrstil ausgerichtet ([40]). Studien zeigen, dass Telematikdaten nicht nur zur Risikobewertung, sondern auch zur Verbesserung von Sicherheit und Umweltverträglichkeit im Straßenverkehr genutzt werden können, etwa durch Feedback und Coaching-Ansätze ([41]; [42]).

9.2.2 Telematik-basierte Systeme mit externer Hardware (Versicherungstarife)

Ein typisches Beispiel für telematikbasierte Fahrstilanalyse ist der Tarif **Telematik Plus** der HUK24. Das System besteht aus einem Telematik-Sensor, der innen an der Frontscheibe angebracht wird, und der Smartphone-App „HUK Mein Auto,, ([43]). Der Sensor erfasst während der Fahrt Daten zu Geschwindigkeit, Beschleunigung sowie Brems- und Lenkverhalten und übermittelt diese über die App an einen Dienstleister (HDD GmbH), der daraus Fahrwerte berechnet ([43]). Aus den aggregierten Fahrwerten wird ein Gesamtfahrwert zwischen 0 und 100 Punkten abgeleitet, der als Grundlage für einen Bonus von bis zu 30 % auf den Kfz-Beitrag dient ([43]).

Telematik-Plus-Systeme verfolgen damit primär zwei Ziele:

- **Risikoadäquate Tarifierung:** Fahrverhalten wird detailliert erfasst und in risikobasierte Prämien übersetzt (Pay-how-you-drive / Pay-as-you-drive) ([40]; [38]).
- **Verhaltensbeeinflussung durch Feedback:** Fahrerinnen und Fahrer erhalten über die App Rückmeldungen zu ihrem Fahrstil und werden zu defensivem, sicherem und zunehmend auch umweltbewusstem Fahren motiviert ([43]; [42]).

Nachteilig ist aus Anwendersicht, dass zusätzliche Hardware im Fahrzeug installiert werden muss und die Auswertung stark an den Versicherungszweck gebunden ist. Die individuelle Transparenz über die Rohdaten bleibt begrenzt, da dem Versicherer meist nur aggregierte Fahrwerte und nicht die vollständigen Fahrdaten zur Verfügung stehen ([43]; [40]).

9.2.3 Smartphone-basierte Fahrten-Tracker ohne explizite Fahrstilanalyse

Neben versicherungsgetriebenen Telematiklösungen existiert eine Vielzahl von Apps, die Fahrten primär zu Dokumentations- oder Navigationszwecken

aufzeichnen. Diese Anwendungen nutzen ausschließlich die Sensorik des Smartphones (**GPS**, Beschleunigungssensoren), liefern aber in der Regel **keine** explizite Fahrstilanalyse oder Fahrerbewertung.

Ein Beispiel ist die App **Geo Tracker – GPS Tracker**. Sie zeichnet Routen **GPS**-basiert auf, arbeitet auch offline und stellt detaillierte Statistiken wie Streckenlänge, Aufzeichnungsdauer, maximale und durchschnittliche Geschwindigkeit, Höhenunterschiede und Steigung bereit ([44]; [45]). Der Fokus liegt auf der präzisen Aufzeichnung von Bewegungsdaten und der Anzeige von Geschwindigkeits- und Höhenprofilen, etwa für Outdoor-Aktivitäten wie Wandern oder Radfahren, nicht auf der sicherheitsorientierten Bewertung des Fahrstils.

Die App **Driversnote** adressiert vor allem das Führen eines digitalen Fahrtenbuchs. Fahrten können automatisch oder manuell über das **GPS** des Smartphones getrackt und anschließend steuerkonform dokumentiert werden ([46]). Im Vordergrund stehen Funktionen wie automatische Fahrtenerkennung, nachträgliche Erfassung von Fahrten und die Erstellung detaillierter Fahrtenbücher für Erstattungen und steuerliche Abzüge, nicht jedoch die Analyse von Beschleunigungs- oder Bremsmustern zur Einstufung des Fahrverhaltens ([46]).

Eine weitere Kategorie stellen Apps dar, die Geschwindigkeit und G-Kräfte visualisieren. Die Android-App **Speedometer with G-FORCE meter** kombiniert beispielsweise einen **GPS**-Tachometer mit einem G-Kraftmessgerät und nutzt den Beschleunigungssensor des Smartphones, um Geschwindigkeit sowie Beschleunigungs- und Querkräfte in Echtzeit darzustellen ([47]). Ziel ist die Visualisierung von Geschwindigkeit und G-Kräften für unterschiedliche Fahr- und Bewegungskontexte (z. B. Auto, Motorrad, Boot), nicht jedoch eine umfassende, regelbasierte Bewertung des Fahrverhaltens im Sinne von „sicher„ oder „risikoreich“ ([47]).

Diese Beispiele zeigen, dass viele bestehende smartphonebasierte Lösungen zwar Fahrten und relevante physikalische Größen wie Geschwindigkeit, Beschleunigung und G-Kräfte erfassen, aber überwiegend auf die Darstellung von Rohdaten und einfachen Statistiken fokussiert sind. Eine systematische Fahrstilanalyse mit qualitativer Bewertung (z. B. Fahrcores, Kategorien wie „defensiv,, „moderat“, „aggressiv,,) ist in diesen Anwendungen typischerweise nicht vorgesehen ([44]; [46]; [47]).

9.2.4 Einordnung der eigenen Lösung

Im Vergleich zu den betrachteten Systemen positioniert sich die in dieser Studienarbeit entwickelte App wie folgt:

- **Sensorik und Infrastruktur:** Die App nutzt ausschließlich die im Smartphone integrierten Sensoren (GPS, Beschleunigungs- und ggf. Gyroskopsensoren) und kommt ohne zusätzliche Telematik-Hardware oder OBD-Schnittstelle aus. Damit folgt sie technisch den smartphonebasierten Tracking-Ansätzen ([44]; [46]), vermeidet aber die Notwendigkeit externer Sensoren, wie sie bei klassischen Versicherungs-Telematiktarifen zum Einsatz kommen ([43]).
- **Funktionsumfang:** Ähnlich zu bestehenden Tracking-Apps werden Fahrten mit Parametern wie Strecke, Dauer, Geschwindigkeitsprofil und G-Kräften erfasst ([44]; [47]). Darüber hinaus zielt die App explizit auf die Auswertung von Fahrverhalten (z. B. starkes Beschleunigen, abruptes Bremsen, hohe Querbeschleunigung) und die Ableitung eines Fahrstils ab. Damit nähert sie sich funktional den telematikbasierten Versicherungslösungen an, bleibt aber technisch auf die Smartphone-Sensorik beschränkt.
- **Zweck und Nutzerfokus:** Während Telematik-Tarife wie **Telematik Plus** vorrangig auf risikobasierte Prämiengestaltung und Bonus-Modelle ausgerichtet sind ([39]; [43]; [42]), verfolgt die hier entworfene App einen nutzerzentrierten Ansatz: Der Schwerpunkt liegt auf Transparenz, Selbstreflexion und einer eigenverantwortlichen Verbesserung des Fahrstils. Die Fahrdaten verbleiben zunächst lokal auf dem Gerät und werden primär zur individuellen Analyse und Visualisierung genutzt.
- **Potenzial im Kontext Sicherheit und Nachhaltigkeit:** Die Literatur zu Fahrzeugtelematik weist darauf hin, dass telematikbasierte Systeme einen Beitrag zu mehr Verkehrssicherheit und geringeren Emissionen leisten können, insbesondere wenn Fahrende kontinuierliches Feedback zu ihrem Fahrverhalten erhalten ([41]; [42]). Die entwickelte App knüpft an diese Erkenntnisse an, indem sie auf Basis der Smartphone-Daten Hinweise zu sicherem und effizientem Fahren geben soll, ohne den Umweg über Versicherungsprodukte oder zusätzliche Hardware zu gehen.

Insgesamt zeigt das Benchmarking, dass es bereits etablierte Lösungen zur Fahrenaufzeichnung und telematikbasierte Versicherungstarife mit umfassender Fahrstilanalyse gibt. Eine Lücke besteht jedoch bei Anwendungen, die **allein mit Smartphone-Sensorik** sowohl Fahrten aufzeichnen als auch eine **qualitative Fahrstilanalyse** durchführen und dabei **konsequent nutzerzentriert** (statt versicherungszentriert) ausgestaltet sind. Genau in diesem Spannungsfeld positioniert sich die in dieser Studienarbeit entwickelte Fahrstilanalyse-App.

9.3 Entwicklung

9.3.1 Prototyping mit Figma

Vor der Implementierung der Fahrstilanalyse-App wurde ein vollständiger UX-Prototyp in Figma erstellt. Prototyping ermöglicht es, Aufbau, Navigation und Interaktionsabläufe einer Anwendung bereits in einer frühen Phase zu planen und zu evaluieren, ohne dafür Entwicklungsressourcen in Form von Code zu binden ([32]; [33]). Durch klickbare Prototypen können typische Nutzungsszenarien – etwa Login, Start und Stopp einer Fahrt, Einsicht in Fahrtdetails oder das Anzeigen von Auswertungen und Gamification-Elementen – realitätsnah durchgespielt und iterativ verbessert werden. Auf diese Weise lassen sich Inkonsistenzen im Layout, fehlende Funktionen oder umständliche Nutzerwege frühzeitig erkennen und beheben, was Entwicklungsaufwand in späteren Phasen reduziert ([32]).

Bei der Konzeption des Prototyps wurde darauf geachtet, dass alle zuvor definierten fachlichen Mindestanforderungen der App im UX-Design berücksichtigt sind. Dazu zählen insbesondere das Tracken von Fahrten, die Anzeige einer Fahrtenhistorie, detaillierte Fahrtauswertungen mit Kennzahlen und Fahrfehlern, Live-Feedback zur aktuellen Fahrt sowie grundlegende Konto- und Profileinstellungen. Diese Anforderungen wurden in Figma modelliert, sodass überprüft werden konnte, ob alle notwendigen Informationen und Aktionen über die Oberfläche erreichbar sind, ohne den Nutzer zu überfordern.

todo{Abschnitt 2.6 referenzieren, wo Ablenkung thematisiert wird}

Ein zentraler Aspekt des Prototypings war die Gestaltung einer nutzerfreundlichen und zugleich fahrtauglichen App. Wie in Abschnitt 2.6 dargestellt, stellen mobile Endgeräte während der Fahrt eine relevante Ablenkungsquelle dar. Daher wurde das UX-Konzept so ausgelegt, dass der Großteil der Interaktion vor oder nach der Fahrt stattfindet. Während der Fahrt soll die App möglichst passiv im Hintergrund laufen: Die Aufzeichnung wird über wenige, klar gestaltete Bedienelemente gestartet, oder wird automatisch gestartet beziehungsweise beendet, detaillierte Analysen, Diagramme und Texte sind hingegen der Nachbetrachtung vorbehalten. Im Prototyp wurde geprüft, ob diese Trennung konsequent eingehalten wird und ob potenziell ablenkende Elemente (z. B. komplexe Menüs, lange Texte, kleine Bedienelemente) während der Fahrt vermieden werden.

todo{Namen der App DriveWise noch irgendwo erwähnen oder wenigstens hier erläutern, nicht das er so random hier auftaucht}

Die Startseite der App (Abbildung 2) bildet den Einstieg in den Prototyp. Sie zeigt den aktuellen DriveWise-Score, einen prominent platzierten Button zum Starten des Trackings sowie eine kompakte Übersicht zur aktuellen Fahrt (Fahrfehler, Distanz, Dauer). Damit wird der Nutzer direkt zu den zentralen Funktionen der App geführt, ohne dass eine aufwändige Navigation erforderlich ist.

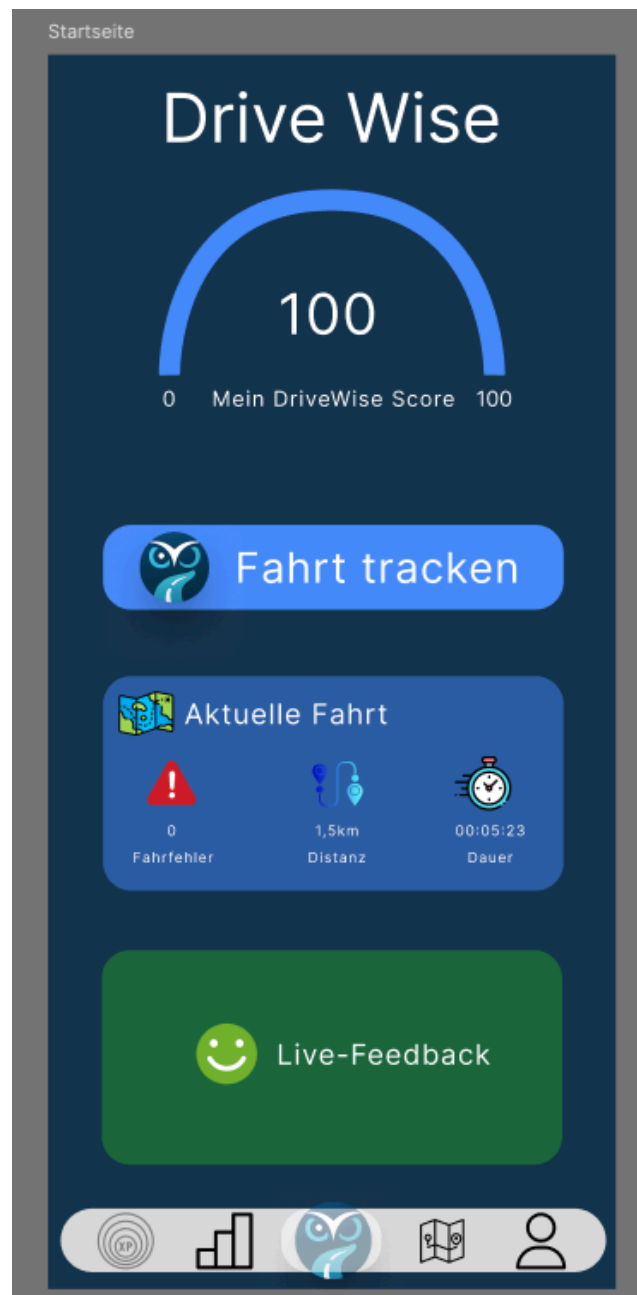


Abbildung 2 – Startseite des Figma-Prototyps mit DriveWise-Score, Einstieg in das Fahrtracking und Übersicht zur aktuellen Fahrt.

Die Fahrtenübersicht (Abbildung 3) stellt alle aufgezeichneten Fahrten chronologisch dar. Für jede Fahrt werden Start- und Zielort, Datum, Uhrzeit, Distanz und Dauer angezeigt. Die Gestaltung ist bewusst schlicht gehalten, um einen schnellen Überblick zu ermöglichen und die Auswahl einer Fahrt für die Detailansicht zu erleichtern.



Abbildung 3 – Fahrtenübersicht mit chronologischer Liste aller aufgezeichneten Fahrten.

In der Detailansicht (Abbildung 4) werden eine Kartenansicht der gefahrenen Strecke und die wichtigsten Kennzahlen zur Fahrt kombiniert. Neben Durchschnitts- und Höchstgeschwindigkeit werden unter anderem die maximale Beschleunigung in G-Kräften sowie die Anzahl der erkannten Fahrfehler (Bremsen, Lenken, Beschleunigen) angezeigt. Dadurch kann der Nutzer sein Fahrverhalten für eine konkrete Fahrt im Nachgang nachvollziehen und gezielt Verbesserungspotenziale identifizieren.

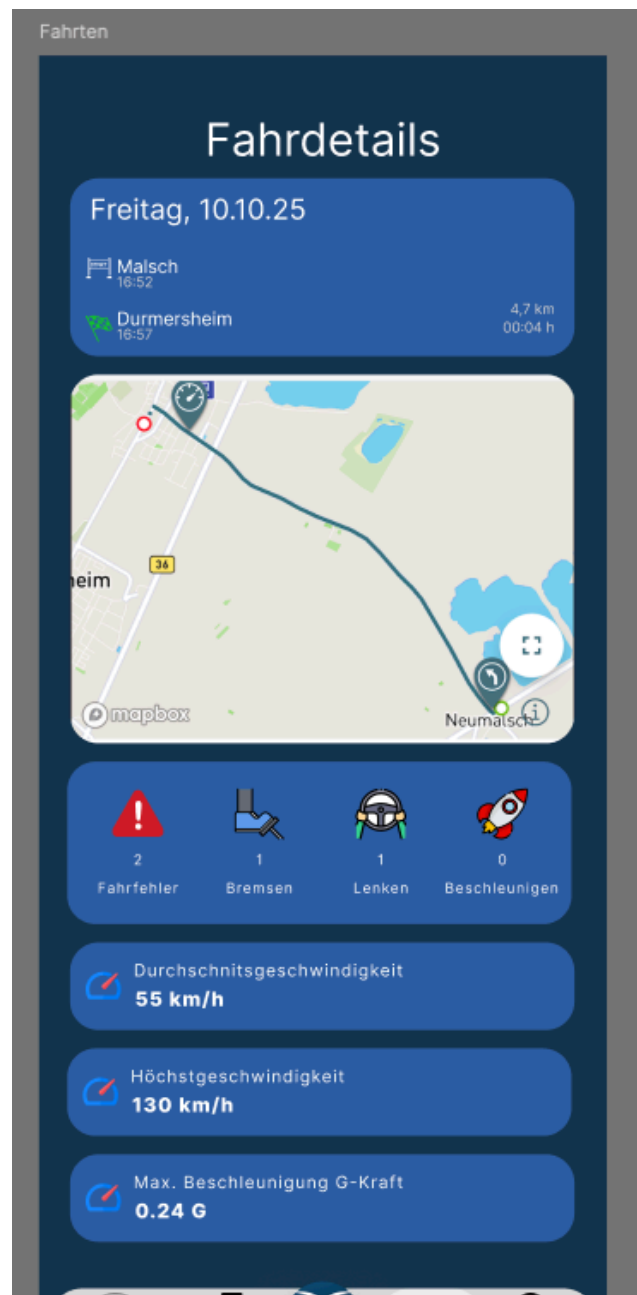


Abbildung 4 – Detailansicht einer Fahrt mit Kartenansicht, Fahrfehlern und zentralen Kennzahlen.

Zusätzlich wurde eine Achievements-Ansicht mit Level- und Quest-System (Abbildung 5) entworfen, die den Gamification-Ansatz der App abbildet. Nutzer erhalten Erfahrungspunkte (XP) für gefahrene Kilometer und können durch das Erreichen bestimmter Ziele (z. B. Gesamtdistanz oder fehlerfreie Fahrten) Quests abschließen. Diese Elemente sollen die langfristige Nutzung der App unterstützen und ein positives, motivationsförderndes Nutzungserlebnis schaffen.



Abbildung 5 – Achievements-Ansicht mit Level-System und Quests zur langfristigen Motivation eines defensiven Fahrstils.

Der in [Figma](#) erstellte Prototyp diente somit nicht nur als visuelle Grundlage für die spätere Implementierung in [SwiftUI](#), sondern auch als Werkzeug zur Überprüfung, ob alle Mindestanforderungen funktional abgedeckt sind, die App in ihrer Struktur nachvollziehbar bleibt und gleichzeitig die Ablenkung des Fahrers während der Fahrt auf ein Minimum reduziert wird.

9.4 Umsetzung des UX-Prototyps in SwiftUI

Nachdem das visuelle Design der Fahrstilanalyse-App in [Figma](#) ausgearbeitet worden waren, wurde das dort definierte User Interface im nächsten Schritt mit [SwiftUI](#) umgesetzt. Ziel dieser Phase war zunächst, das in [Figma](#) entworfene UX-Design so genau wie möglich „1:1“ nachzubilden – bewusst zunächst ohne vollständige fachliche Funktionalität. Auf diese Weise konnte die Navigationsstruktur, das visuelle Erscheinungsbild und die Informationshierarchie der App in einer lauffähigen [iOS](#)-App überprüft werden, bevor Messlogik und Datenanbindung integriert wurden ([14]; [21]).

9.4.1 Navigationsstruktur mit TabView

Die globale Navigation der App entspricht der in [Figma](#) definierten Tab-Bar mit fünf Hauptbereichen: **Erfolge**, **Statistiken**, **Startseite**, **Fahrten** und **Profil**. In [SwiftUI](#) wurde diese Struktur mit einer `TabView` umgesetzt, die über eine interne `Tab-Enum` gesteuert wird:

```
enum Tab: Hashable {  
    case achievements, statistics, startseite, fahrten, profil  
}  
  
@State private var selection: Tab = .startseite  
  
var body: some View {  
    TabView(selection: $selection) {  
        // ...  
    }  
}
```

Jeder Tab ist in einen eigenen `NavigationStack` eingebettet. Dies ermöglicht es, innerhalb eines Bereichs (z. B. von der Fahrtenliste in die Fahrtdetails) eigene Navigationshierarchien aufzubauen, ohne die globale Tab-Navigation zu beeinflussen. Die gemeinsame Applogik, etwa der aktuelle Fahrstatus und die Liste aller aufgezeichneten Fahrten, wird über ein zentrales `DriveManager`-Objekt als `@StateObject` in `ContentView` erzeugt und mittels `.environmentObject(driveManager)` an alle Unter-Views weitergereicht. Dadurch bleibt die Zustandsverwaltung klar getrennt von der reinen UI-Definition.

Für den mittleren Tab (Startseite) wurde – analog zum [Figma](#)-Entwurf – anstelle eines Standard-SF-Symbols ein eigenes App-Icon verwendet. In [SwiftUI](#) wird

hierzu geprüft, ob das Asset vorhanden ist; falls nicht, fällt die Implementierung auf ein System-Icon zurück. Dies erhöht die visuelle Wiedererkennbarkeit der zentralen Startansicht.

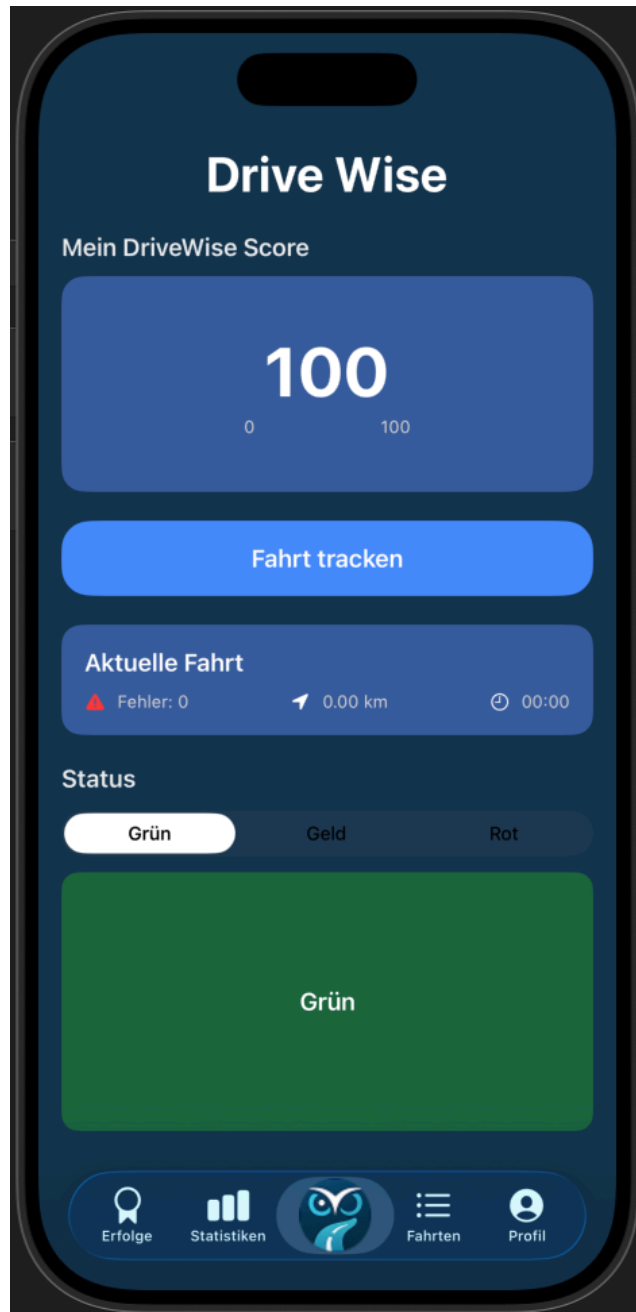


Abbildung 6 – Startseite in der [SwiftUI](#)-Implementierung mit DriveWise-Score, Tracking-Button und Übersicht zur aktuellen Fahrt.

9.4.2 Startseite: Mapping des Figma-Layouts nach SwiftUI

Die Startseite wurde in SwiftUI so umgesetzt, dass Aufbau und Gestaltungslogik dem Figma-Prototyp möglichst genau entsprechen. Die View StartseiteView verwendet einen ZStack für den farbigen Hintergrund und einen ScrollView für den vertikal scrollbaren Inhalt. Die in Figma definierten Karten und Bereiche (Score-Karte, Track-Button, aktuelle Fahrt, Status-Karte) sind jeweils als VStack-Blöcke mit abgerundeten Rechtecken (RoundedRectangle) und projektspezifischen Farbkonstanten realisiert.

Der Titel „Drive Wise“ und der „Mein DriveWise Score“-Block bilden die visuelle Einstiegsfläche. Darunter folgt ein großer, vollflächiger Button zum Starten bzw. Beenden der Fahrt. In der ersten Iteration wurde dieser Button lediglich als UI-Element umgesetzt; in einem späteren Schritt wurde die Tap-Action an den DriveManager angebunden, um tatsächlich Messdaten zu erfassen. Die Statuskarte am unteren Bildschirmrand wurde über eine Picker-Komponente realisiert, die zwischen verschiedenen farbigen Zuständen (z. B. Grün/Gelb/Rot) umschalten kann. Diese Struktur orientiert sich eng an den im Prototyp definierten UX-Prinzipien: zentrale Funktion im Fokus, klare Lesbarkeit und minimale visuelle Ablenkung während der Fahrt.

9.4.3 Fahrtenübersicht und Fahrtdetails

Die Fahrtenübersicht wurde in der Implementierung als eigene View (FahrtenListView) umgesetzt und in die TabView integriert. Inhaltlich und visuell entspricht sie der in Figma modellierten Liste: Pro Fahrt werden Startort, Zielort, Distanz, Uhrzeit und Datum angezeigt. Die Einträge werden gruppiert nach Datum dargestellt, sodass der Nutzer vergangene Fahrten schnell einordnen kann.

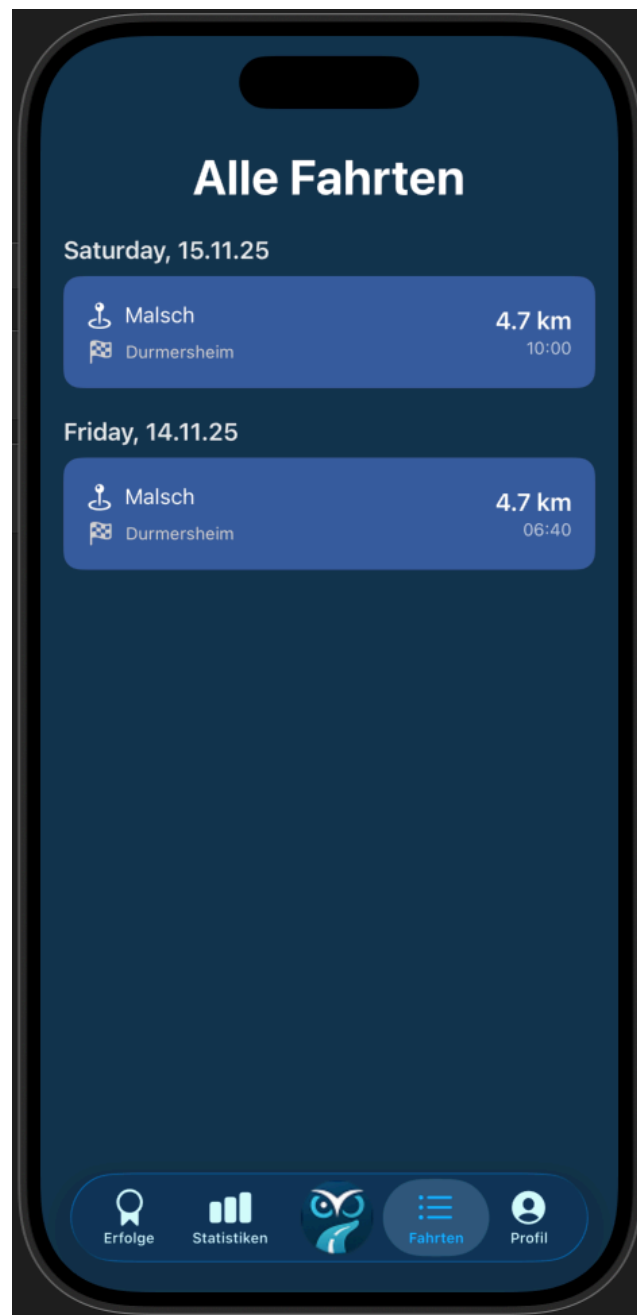


Abbildung 7 – Fahrtenübersicht in [SwiftUI](#) mit gruppierter Darstellung aller aufgezeichneten Fahrten.

Wählt der Nutzer eine Fahrt aus der Liste aus, navigiert die App in die `FahrtDetailView`. Diese View wurde bewusst so gestaltet, dass sie die im [Figma](#)-Prototyp angelegte Struktur übernimmt: Eine Kopfkarte mit Datum, Start- und Zielort sowie Distanz und Dauer, gefolgt von einem Bereich für die Kartenansicht der Route (zunächst als Platzhalter) und einer Fehlerübersicht. Die Fehlerübersicht besteht aus einer Karte mit vier Spalten (gesamt, Bremsen, Lenken, Beschleu-

nigen) und nutzt SF-Symbole (z. B. steeringwheel oder rocket.fill) zur visuellen Unterstützung.

Darunter folgen Kennzahl-Karten für Durchschnittsgeschwindigkeit, Höchstgeschwindigkeit und maximale Beschleunigung. Jede dieser Karten ist als eigene Zeile (metricRow) umgesetzt, bestehend aus Icon, Titel und Wert. Damit orientiert sich die Implementierung eng an der im [Figma](#)-Prototyp definierten Informationshierarchie: Zunächst kontextuelle Metadaten zur Fahrt, dann sicherheitsrelevante Ereignisse, anschließend aggregierte Kennzahlen.



Abbildung 8 – Fahrtdetailansicht mit Kopfkarte, Kartenplatzhalter, Fehlerübersicht und Kennzahlkarten.

9.4.4 Statistik- und Erfolgsansichten

Die im [Figma](#)-Prototyp vorgesehenen Auswertungsbereiche wurden in [SwiftUI](#) als separate Tabs umgesetzt. Die Statistik-Ansicht (Statistiken) fasst die Fahrten einer Woche oder eines Monats zusammen und zeigt unter anderem die mittlere

Geschwindigkeit, die Anzahl der Fahrten und Fahrfehler sowie die Gesamtdistanz. Im aktuellen Implementierungsstand sind Balkendiagramme als Platzhalter vorgesehen, die später durch konkrete Diagrammkomponenten ersetzt werden sollen. Struktur und Layout der Statistik-Screen entsprechen bereits dem [Figma-Design](#), sodass die spätere Integration realer Daten hauptsächlich die Inhalte betrifft.



Abbildung 9 – Statistik-Ansicht mit Wochen-/Monatsumschaltung und Kennzahlenübersicht.

Die Erfolge-Ansicht (Erfolge) visualisiert das Gamification-Konzept der App. Sie zeigt das aktuelle Level, eine Fortschrittsleiste zum nächsten Level und eine Liste von Quests, bei denen der Nutzer für das Erreichen bestimmter Distanzen Erfahrungspunkte erhält. In [SwiftUI](#) wurde dieser Bereich als scrollbare Liste von Karten umgesetzt, die in Aufbau und Farbgebung an die [Figma](#)-Vorlage angelehnt sind. Die Inhalte sind zunächst statisch; die dynamische Berechnung von XP und Leveln kann in einer späteren Ausbaustufe über das zentrale Zustandsobjekt ergänzt werden.



Abbildung 10 – Erfolge-Ansicht mit Levelanzeige, Fortschrittsbalken und Quest-Liste.

9.4.5 Zwischenfazit zur UI-Implementierung

Die Umsetzung des [Figma-UX-Prototyps](#) in [SwiftUI](#) diene als wichtiger Zwischenschritt zwischen konzeptionellem Design und funktionsfähiger Anwen-

dung. Durch die Konzentration auf Layout, Navigationsstruktur und Informationsdarstellung konnte sichergestellt werden, dass

- die im Grundlagenkapitel definierten **UX**-Ziele (Nutzerfreundlichkeit, Verständlichkeit, Reduktion von Ablenkung während der Fahrt) eingehalten werden,
- alle fachlichen Mindestanforderungen bereits in der Oberfläche sichtbar angelegt sind,
- und spätere Implementierungsschritte (Anbindung der Sensorik, Berechnung der Fahrfehler, Darstellung realer Statistiken) in eine konsistente **UI** integriert werden können.

Damit bildet die **SwiftUI**-Implementierung des **UX** ein stabiles Fundament, auf dem die weitere technische Umsetzung der Fahrstilanalyse aufbauen kann.

10

Code

10.1 Features einbauen

- CoreMotion für Bewegungssensoren
- MapKit für Kartenanzeige
- CoreData für lokale Speicherung
- HealthKit für Health Daten (Geschwindigkeit)
- Algorithmen zur Fahrverhaltensanalyse
- CoreData für lokale Speicherung (Lokal oder iCloud)



Anlysen (Joscha, Max)

- Algorithmen zur Fahrverhaltensanalyse
 - Beschleunigung
 - Bremsverhalten
 - Kurvenverhalten
 - Geschwindigkeit

12

Gamification (Joscha)

- Punkte System
- Levels
- Belohnungen
- Benachrichtigungen

13

Testen

14

Schlussbetrachtung

14.1 Learnings

14.2 Fazit

14.3 Ausblick

A Literatur

- [1] YellowFox GmbH, „Fahrstilanalyse & Fahrerbewertung“. [Online]. Verfügbar unter: <https://www.yellowfox.de/loesungen/fahrerbewertung-fahrstilanalyse/>
- [2] C. Saiprasert, T. Pholprasit, und W. Pattara-atikom, „Detecting Driving Events Using Smartphone“. [Online]. Verfügbar unter: https://www.researchgate.net/publication/279949922_Detecting_Driving_Events_Using_Smartphone
- [3] S. Science, „Analyse von Fahrverhalten für sicherere Straßen“. [Online]. Verfügbar unter: <https://scisimple.com/de/articles/2025-06-29-analyse-von-fahrverhalten-fuer-sicherere-strassen%E2%80%93ak58j7v>
- [4] S. Moosavi, A. Nandi, und R. Ramnath, „Discovery of Driving Patterns by Trajectory Segmentation“, in *Proceedings of the 3rd ACM SIGSPATIAL PhD Workshop*, 2016. [Online]. Verfügbar unter: <https://arxiv.org/abs/1804.08748>
- [5] Technische Hochschule Ingolstadt, „Fahrerverhaltens-Schätzer – Projektübersicht“. [Online]. Verfügbar unter: <https://www.thi.de/forschung/institut-fuer-innovative-mobilitaet-iimo/els/projekte/fahrerverhaltensschaetzer/>
- [6] World Health Organization, *Mobile Phone Use: A Growing Problem of Driver Distraction*. Geneva, Switzerland: World Health Organization, 2011. [Online]. Verfügbar unter: https://www.who.int/violence_injury_prevention/publications/road_traffic/en/index.html
- [7] D. L. Strayer, J. M. Cooper, J. Turrill, J. Coleman, N. Medeiros-Ward, und F. Biondi, „Measuring Cognitive Distraction in the Automobile“. [Online]. Verfügbar unter: <https://aaaafoundation.org/measuring-cognitive-distraction-in-the-automobile/>
- [8] A. A. Bangash, D. Tiganov, K. Ali, und A. Hindle, „Energy Efficient Guidelines for iOS Core Location Framework“, *Proc. IEEE ICSME*, S. 1–12, 2021.
- [9] T. Liu, H. Wang, X. Zhang, X. Chen, J. Xu, und X. Zhu, „Smartphone Positioning and Accuracy Analysis Based on GNSS Raw Observations“, *Sensors*, Bd. 21, Nr. 12, S. 4148, 2021, doi: [10.3390/s21124148](https://doi.org/10.3390/s21124148).

-
- [10] N. Klara und others, „Driving Style Recognition Using Smartphone Sensors Based on Fuzzy Logic“, *ResearchGate*, Bd. 12, S. 1967–1978, 2021.
 - [11] A. Allan, *Basic Sensors in iOS: Programming the Accelerometer, Gyroscope, and More*. O'Reilly Media, 2011, S. 37–83.
 - [12] R. Faragher und R. Harle, „Location Fingerprinting With Bluetooth Low Energy Beacons“, *IEEE Journal on Selected Areas in Communications*, Bd. 33, Nr. 11, S. 2418–2428, 2015.
 - [13] A. Inc., *The Swift Programming Language (Swift 1.0)*. Apple Books, 2014. [Online]. Verfügbar unter: <https://developer.apple.com/swift/>
 - [14] A. Inc., „Swift Language Guide“. [Online]. Verfügbar unter: <https://docs.swift.org/swift-book/documentation/the-swift-programming-language/>
 - [15] A. Inc., „Xcode Overview – Apple Developer“. [Online]. Verfügbar unter: <https://developer.apple.com/xcode/>
 - [16] A. Inc., „Core Location Framework“. [Online]. Verfügbar unter: <https://developer.apple.com/documentation/corelocation>
 - [17] A. Inc., „Core Motion Framework“. [Online]. Verfügbar unter: <https://developer.apple.com/documentation/coremotion>
 - [18] A. Inc., „CLLocation Class Reference“. [Online]. Verfügbar unter: <https://developer.apple.com/documentation/corelocation/cllocation>
 - [19] A. Inc., „FileManager Class Reference“. [Online]. Verfügbar unter: <https://developer.apple.com/documentation/foundation/filemanager>
 - [20] N. Nurseitov, M. Paulson, R. Reynolds, und C. Izurieta, „Comparison of JSON and XML Data Interchange Formats: A Case Study“, in *CAINE Conference*, 2009, S. 157–162.
 - [21] A. Inc., „SwiftUI – Declarative Interfaces for Any Apple Device“. [Online]. Verfügbar unter: <https://developer.apple.com/documentation/swiftui/>
 - [22] Google Inc., „Firebase Documentation“. [Online]. Verfügbar unter: <https://firebase.google.com/docs>
 - [23] Google Inc., „Firebase Authentication Documentation“. [Online]. Verfügbar unter: <https://firebase.google.com/docs/auth>

-
- [24] Google Inc., „Firebase Authentication for iOS – FirebaseUI Setup Guide“. [Online]. Verfügbar unter: https://firebase.google.com/docs/auth/ios/firebaseui?hl=de#set_up_sign-in_methods
- [25] Google Inc., „Firebase Pricing Overview“. [Online]. Verfügbar unter: <https://firebase.google.com/pricing>
- [26] M. Sailer, J. U. Hense, S. K. Mayr, und H. Mandl, „How Gamification Motivates: An Experimental Study of the Effects of Specific Game Design Elements on Psychological Need Satisfaction“, *Computers in Human Behavior*, Bd. 69, S. 371–380, 2017, doi: [10.1016/j.chb.2016.12.033](https://doi.org/10.1016/j.chb.2016.12.033).
- [27] P. Bitrián, I. Buil, und S. Catalán, „Enhancing User Engagement: The Role of Gamification in Mobile Apps“, *Journal of Business Research*, Bd. 132, S. 170–185, 2021, doi: [10.1016/j.jbusres.2021.04.028](https://doi.org/10.1016/j.jbusres.2021.04.028).
- [28] R. M. Ryan und E. L. Deci, „Self-Determination Theory and the Facilitation of Intrinsic Motivation, Social Development, and Well-Being“, *American Psychologist*, Bd. 55, Nr. 1, S. 68–78, 2000, doi: [10.1037/0003-066X.55.1.68](https://doi.org/10.1037/0003-066X.55.1.68).
- [29] I. Petersen, *User Experience verstehen. In: Hoffmann, S. (eds) Digitales Produktmanagement*. Springer Gabler, 2023.
- [30] D. A. Norman, *The Design of Everyday Things*. Basic Books, 2013.
- [31] J. J. Garrett, *The Elements of User Experience*. New Riders, 2010.
- [32] T. Green und K. Brandon, „Figma and the UX Process“, *UX Design with Figma*. Springer, 2024. [Online]. Verfügbar unter: https://link.springer.com/chapter/10.1007/979-8-8688-0324-6_3
- [33] Ambient Digital GmbH, „Figma – das kollaborative Design-Tool“. [Online]. Verfügbar unter: <https://ambient.digital/wissen/ambipedia/figma/>
- [34] I. Sommerville, *Software Engineering*, 10. Aufl. Harlow, United Kingdom: Pearson, 2016.
- [35] K. Pohl und C. Rupp, *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam, Foundation Level, IREB Compliant*, 2nd Aufl. Santa Barbara, CA, USA: Rocky Nook Inc., 2015.

-
- [36] IEEE, ISO, und IEC, „ISO/IEC/IEEE 29148:2018 — Systems and software engineering — Life cycle processes — Requirements engineering“. IEEE, ISO/IEC, New York, USA, 2018. doi: [10.1109/IEEESTD.2018.8559686](https://doi.org/10.1109/IEEESTD.2018.8559686).
- [37] Geotab Team, „What is telematics?“. [Online]. Verfügbar unter: <https://www.geotab.com/blog/what-is-telematics/>
- [38] National Association of Insurance Commissioners, „Telematics“. [Online]. Verfügbar unter: <https://content.naic.org/insurance-topics/telematics>
- [39] National Association of Insurance Commissioners, „Understanding Usage-Based Insurance“. [Online]. Verfügbar unter: <https://content.naic.org/article/consumer-insight-understanding-usage-based-insurance>
- [40] D. Karapiperis u. a., „Usage-Based Insurance and Vehicle Telematics: Insurance Market and Regulatory Implications“, technical report, 2015. [Online]. Verfügbar unter: https://content.naic.org/sites/default/files/inline-files/cipr_study_150324_usage_based_insurance_and_vehicle_telematics_study_series_1.pdf
- [41] O. Ghaffarpasand, M. Burke, L. K. Osei, H. Ursell, S. Chapman, und F. D. Pope, „Vehicle Telematics for Safer, Cleaner and More Sustainable Urban Transport: A Review“, *Sustainability*, Bd. 14, Nr. 24, S. 16386, 2022, doi: [10.3390/su142416386](https://doi.org/10.3390/su142416386).
- [42] A. Cevolini, E. Morotti, E. Esposito, L. Romanelli, R. Tisseur, und C. Misani, „Can Telematics Improve Driving Style? The Use of Behavioral Data in Motor Insurance“, *Big Data and Cognitive Computing*, Bd. 9, Nr. 9, S. 225, 2025, doi: [10.3390/bdcc9090225](https://doi.org/10.3390/bdcc9090225).
- [43] HUK24 AG, „Autoversicherung mit Telematik Plus“. [Online]. Verfügbar unter: <https://www.huk24.de/telematikplus>
- [44] Geo Tracker - GPS Tracker, „Geo Tracker – GPS Tracker (Webseite)“. [Online]. Verfügbar unter: <https://geo-tracker.org/de>
- [45] Geo Tracker - GPS Tracker, „Geo Tracker – GPS Tracker (App Store Beschreibung)“. [Online]. Verfügbar unter: <https://apps.apple.com/de/app/geo-tracker-gps-tracker/id1610586137>
- [46] Driversnote, „Fahrten einfach tracken~| Driversnote GPS-Meilentracker“. [Online]. Verfügbar unter: <https://www.driversnote.de/tracking>

-
- [47] Galaxy studio apps, „Speedometer with G-FORCE meter“. [Online]. Verfügbar unter: <https://play.google.com/store/apps/details?id=oops.speedometerwithgforce>

B Glossar

API – Application Programming Interface: Programmierschnittstelle zum Austausch von Daten und Funktionen zwischen Softwarekomponenten. [20](#)

CAN – Controller Area Network: Fahrzeug-Bussystem zur Kommunikation zwischen Steuergeräten. Wird häufig genutzt, um Fahrzeugsignale und Zustände zu übertragen. [3](#)

Core Location: Apple-Framework für Standortdienste unter iOS, z. B. Zugriff auf GPS-basierte Positions- und Bewegungsinformationen. [1](#), [13](#), [13](#), [20](#)

Core Motion: Apple-Framework für Bewegungs- und Sensorikdaten unter iOS, z. B. Beschleunigungs- und Gyroskopdaten. [1](#), [13](#), [13](#), [20](#)

DHBW – Duale Hochschule Baden-Württemberg: Duale Hochschule Baden-Württemberg; Hochschule mit dualem Studienmodell (Wechsel aus Theorie- und Praxisphasen). [1](#), [20](#), [20](#)

Datensparsamkeit: Prinzip, nur die Daten zu erheben und zu verarbeiten, die für den Zweck wirklich erforderlich sind (Privacy by Design). [1](#), [2](#), [13](#)

Figma: Design- und Prototyping-Tool zur Erstellung von UI-/UX-Layouts und Interaktionsflüssen. [1](#), [2](#), [2](#), [3](#), [12](#), [19](#), [19](#), [19](#), [19](#), [19](#), [25](#), [31](#), [31](#), [31](#), [33](#), [36](#), [37](#), [37](#), [37](#), [39](#), [39](#), [39](#), [39](#), [40](#), [41](#), [42](#), [43](#), [44](#), [45](#)

Firebase: Entwicklungsplattform von Google mit Tools für Backend-Services, Authentifizierung, Datenbanken und Analytics. [1](#), [14](#), [14](#), [14](#), [14](#)

G-Sensor: Beschleunigungssensor (Accelerometer), der lineare Beschleunigungen entlang verschiedener Achsen misst. [1](#)

GPS – Global Positioning System: Satellitengestütztes System zur Positions- und Zeitbestimmung. Wird u. a. zur Ermittlung von Strecke und Geschwindigkeit genutzt. [1](#), [1](#), [2](#), [10](#), [10](#), [10](#), [21](#), [25](#), [26](#), [27](#), [29](#), [29](#), [29](#), [29](#), [30](#)

Gyroskop: Sensor zur Messung von Rotationsbewegungen bzw. Winkelgeschwindigkeiten (Drehbewegungen) eines Geräts. [1](#)

Heuristik: Pragmatische Entscheidungsregel, die in der Praxis gut funktioniert (z. B. Schwellwerte), ohne ein vollständiges Modell der Realität zu benötigen. [3](#)

IDE – Integrierte Entwicklungsumgebung: Software zur Softwareentwicklung mit Editor, Compiler, Debugger und weiteren Werkzeugen. [12](#), [20](#)

JSON – JavaScript Object Notation: Textbasiertes Datenformat zum Austausch strukturierter Daten. Kompakt und gut lesbar. [13](#), [20](#)

KI – Künstliche Intelligenz: Simulation intelligenten Verhaltens durch Computersysteme (englisch: AI – Artificial Intelligence). [4](#)

Logging: Kontinuierliches Protokollieren und Speichern von Mess- oder Ereignisdaten über einen Zeitraum. [2](#)

ML – Machine Learning: Methoden, bei denen Modelle aus Daten Muster lernen, um Vorhersagen oder Klassifikationen zu treffen. [3](#)

OBD – On-Board Diagnostics: Fahrzeugdiagnose-Schnittstelle zur Abfrage von Fehlercodes und Sensordaten im Fahrzeug. [2](#), [3](#), [24](#), [30](#)

Scoring: Bewertungssystem, das aus Messdaten einen zusammenfassenden Punktwert ableitet (z. B. Risiko- oder Fahrstil-Score). [3](#)

Sensorfusion: Kombination mehrerer Sensordatenquellen (z. B. GPS und Bewegungssensoren), um robustere und genauere Schätzwerte zu erhalten. [11](#)

Swift: Von Apple entwickelte Programmiersprache für iOS, macOS, watchOS und tvOS. Moderne, typsichere Sprache mit klarer Syntax. [1](#), [12](#), [12](#), [12](#), [12](#), [20](#)

SwiftUI: Deklaratives UI-Framework von Apple zur Gestaltung von Benutzeroberflächen für alle Apple-Plattformen. [1](#), [2](#), [2](#), [13](#), [13](#), [20](#), [36](#), [37](#), [37](#), [37](#), [37](#), [38](#), [39](#), [39](#), [40](#), [42](#), [44](#), [45](#), [46](#)

UBI – Usage-Based Insurance: Nutzungsbasierte Versicherung: Tarifmodell, bei dem die Prämie vom tatsächlichen Fahrverhalten abhängt. [27](#)

UI – User Interface: Benutzerschnittstelle: Die sichtbaren und interaktiven Elemente einer Software, über die Nutzer mit dem System interagieren. [2](#), [17](#), [19](#), [25](#), [37](#), [39](#), [45](#), [46](#)

UX – User Experience: Nutzererlebnis: Wahrnehmung und Erfahrung eines Nutzers bei der Interaktion mit einem Produkt (z. B. Bedienbarkeit, Verständlichkeit, Zufriedenheit). [1](#), [1](#), [2](#), [1](#), [13](#), [17](#), [17](#), [17](#), [17](#), [17](#), [17](#), [17](#), [17](#), [17](#), [17](#), [18](#), [18](#), [18](#), [18](#), [18](#), [19](#), [19](#), [19](#), [19](#), [21](#), [21](#), [25](#), [31](#), [31](#), [31](#), [37](#), [37](#), [39](#), [45](#), [46](#), [46](#)

WHO – World Health Organization: Weltgesundheitsorganisation: Sonderorganisation der Vereinten Nationen für Gesundheit. [8](#)

Xcode: Integrierte Entwicklungsumgebung (IDE) von Apple für die Entwicklung von Apps für iOS, macOS, watchOS und tvOS. [12](#), [20](#), [25](#)

iOS: Mobiles Betriebssystem von Apple für iPhone (und historisch auch iPod touch). Es bildet die Plattform, auf der iOS-Apps ausgeführt werden. [1](#), [1](#), [1](#), [2](#), [12](#), [12](#), [12](#), [12](#), [12](#), [13](#), [13](#), [14](#), [17](#), [20](#), [21](#), [22](#), [26](#), [37](#)

iPadOS: Betriebssystem von Apple für iPad-Geräte, basierend auf iOS. [12](#)

macOS: Betriebssystem von Apple für Mac-Computer. [12](#)

watchOS: Betriebssystem von Apple für die Apple Watch. [12](#)

C Anhang

C.1 UX-Prototyp in Figma

Dieser Anhang dokumentiert den vollständigen UX-Prototyp der Fahrstilanalyse-App, der in Figma erstellt wurde.

todo{Hier noch das richtige Kapitel verlinken}}

Während im Designkapitel nur die zentralen Ansichten (Startseite, Fahrtenübersicht, Fahrtdetails und Achievements) dargestellt werden, werden hier ergänzend die übrigen Views gezeigt. Damit wird sichtbar, dass die gesamte Nutzerreise – vom ersten Einstieg über die Authentifizierung bis zur Profilverwaltung – durchgängig im Prototyp modelliert wurde.

C.2 A.1 Einstiegs- und Authentifizierungsansichten

Die folgenden Abbildungen zeigen die Einstiegs- und Anmeldebildschirme der App. Sie bilden den ersten Kontaktpunkt für neue und bestehende Nutzer und sind daher für einen verständlichen und reibungslosen Einstieg wesentlich.

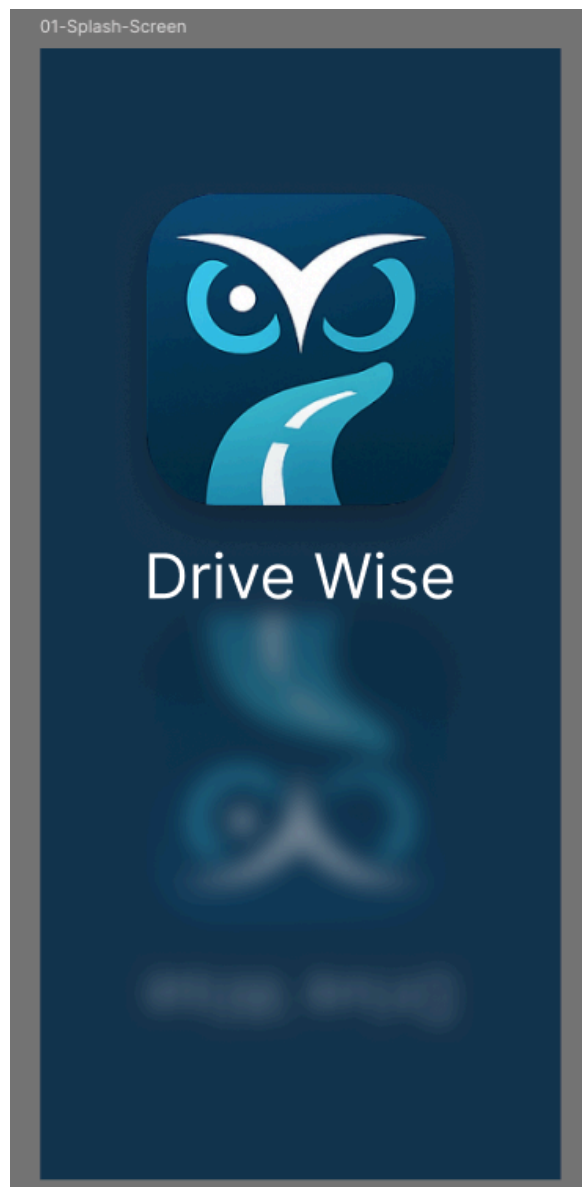



Abbildung 11 — Splash-Screen der App mit App-Icon und Titel, der beim Start der Anwendung kurz angezeigt wird.

02-Login-1



Hallo Driver!

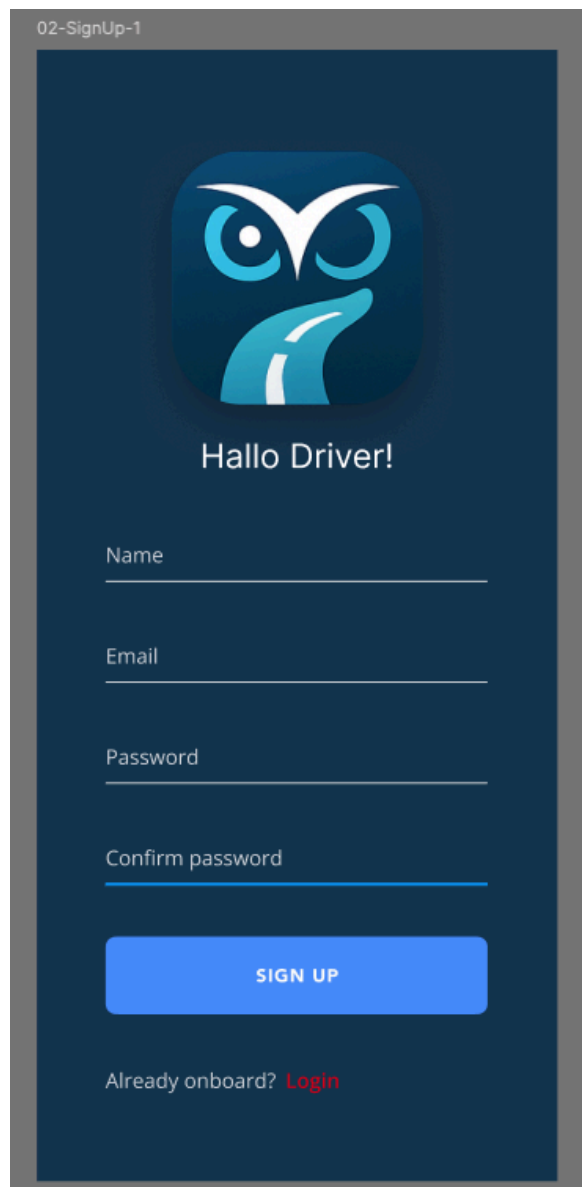
Email

Password


LOGIN

Need an account? [Sign up](#)

Abbildung 12 — Login-Ansicht mit Eingabefeldern für E-Mail und Passwort sowie der Möglichkeit, zur Registrierung zu wechseln.



02-SignUp-1



Hallo Driver!

Name

Email

Password

Confirm password

SIGN UP

Already onboard? [Login](#)

Abbildung 13 — Registrierungsansicht für neue Nutzer mit Feldern für Name, E-Mail und Passwort.

C.3 A.2 Profil- und Kontoverwaltung

Die Profil- und Einstellungsansichten ermöglichen es dem Nutzer, seine Kontoinformationen anzupassen. Dazu gehören insbesondere das Ändern von E-Mail-Adresse und Passwort. Diese Screens ergänzen die im Hauptteil vorgestellten fachlichen Funktionen um die notwendige Kontenverwaltung.

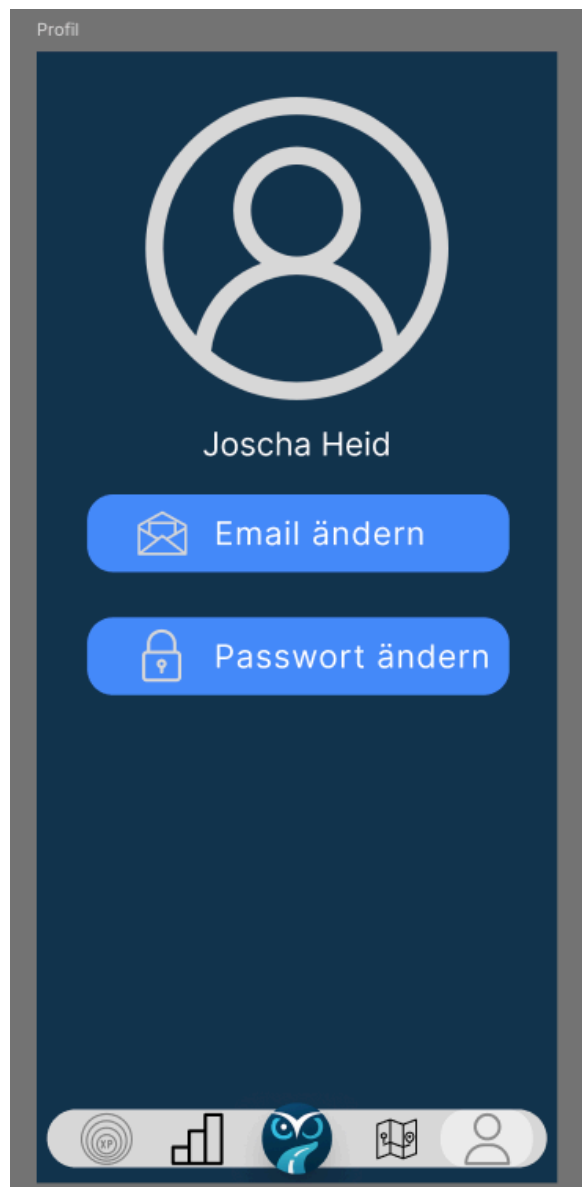


Abbildung 14 — Profil- und Einstellungsübersicht mit Anzeige des Benutzernamens und Zugriff auf Kontoaktionen.

The screenshot shows a mobile application interface with a dark blue background. At the top, the word "Profil" is displayed in a light gray font. Below it is a white back arrow icon. The main content area contains two white text input fields: the first is labeled "Aktuelle Email" and the second is labeled "Neue Email". Below these fields is a prominent blue button with rounded corners, featuring a white envelope icon and the text "Email ändern". At the bottom of the screen is a navigation bar with five icons: a fingerprint scanner, a bar chart, a blue owl logo, a document with a magnifying glass, and a person silhouette.

Abbildung 15 — Ansicht zum Ändern der E-Mail-Adresse mit Eingabefeldern für aktuelle und neue E-Mail-Adresse.

The image shows a mobile application screen titled "Profil" (Profile) at the top. Below the title is a back arrow icon. The main content area has a dark blue background and contains three input fields for password management: "Aktuelles Passwort" (Current Password), "Neues Passwort" (New Password), and "Neues Passwort bestätigen" (Confirm New Password). Each field is represented by a horizontal line. Below these fields is a large green button with a white padlock icon and the text "Passwort ändern" (Change Password). At the bottom of the screen is a navigation bar with five icons: a circular icon with "XP", a bar chart, a blue owl logo, a map icon, and a person icon.

Abbildung 16 — Ansicht zum Ändern des Passworts mit Eingabefeldern für aktuelles und neues Passwort sowie Bestätigung.

Selbstständigkeitserklärung

Gemäß Ziffer 1.1.13 der Anlage 1 zu §§ 3, 4 und 5 der Studien- und Prüfungsordnung für die Bachelorstudiengänge im Studienbereich Technik der Dualen Hochschule Baden- Württemberg vom 29.09.2017. Wir versichern hiermit, dass wir unsere Arbeit mit dem Thema:

Entwicklung einer App zur Analyse des Fahrverhaltens von PKW-Fahrern

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben. Wir versichern zudem, dass alle eingereichten Fassungen übereinstimmen.

Karlsruhe, 13.01.2026

Joscha Heid

Max Joch