

Bananasplit – Mitfahrzentrale

Dokumentation

Erstellt von: Erblin Topalli, Thomas Hahn, Sergej Steinsiek, Benedikt Pankratz

Datum:

Sprint 1: 07.01.2025 - 10.01.2025

Sprint 2: 10.02.2025 - 14.02.2025

Sprint 3: 17.02.2025 - 21.02.2025

Sprint 4: 31.03.2025 - 04.04.2025

Inhaltsverzeichnis

Einleitung.....	02
Projektübersicht.....	02
Ziele & Motivation.....	03
Systemarchitektur.....	03-05
Benutzeroberfläche und Funktionalitäten.....	06-13
Interaktive Features und Toast's.....	14
Technische Dokumentation (Swagger UI).....	14
Implementierungsdetails.....	15-16
Test & Implementierung.....	17
Fazit	17
Anhang.....	18
Android App.....	19-23

1. Einleitung

Die Mitfahrzentrale **Bananasplit** zielt darauf ab, Fahrgemeinschaften so einfach und effizient wie möglich zu gestalten. Durch die Kombination von moderner Webtechnologie und einem durchdachten Benutzerkonzept wird es den Nutzern ermöglicht, nicht nur Spritkosten zu sparen, sondern auch einen Beitrag zur Umwelt zu leisten und neue Kontakte zu knüpfen.



2. Projektübersicht

Das Projekt **Mitfahrzentrale** stellt eine Webanwendung dar, die in mehreren Schichten aufgebaut ist:

- **Frontend:** Benutzerfreundliche Oberfläche mit responsivem Design
- **Backend:** API-gestützte Kommunikation, dokumentiert und getestet mittels Swagger UI
- **Datenbank:** Speicherung von Benutzerdaten, Fahrten Angeboten und Nachrichten



3. Ziele und Motivation

3.1 Motivation

- **Umweltschutz:** Reduzierung der CO₂-Emissionen durch gemeinsame Fahrten.
- **Kosteneinsparung:** Optimierung der Spritkosten durch Fahrgemeinschaften.
- **Soziale Interaktion:** Förderung des Community-Gefüls und Kennenlernen neuer Leute.

3.2 Projektziele

- **Benutzerfreundlichkeit:** Einfache Registrierung, Login und Bedienung der App.
 - **Effizienz:** Schnell verfügbare Informationen über Fahrten und Mitfahrgelegenheiten.
 - **Transparenz:** Durch Bewertungen und eine gut dokumentierte API wird Vertrauen aufgebaut.
-

4. Systemarchitektur und Technischer Hintergrund

Das System besteht aus mehreren Komponenten, die zusammenarbeiten, um ein reibungsloses Nutzungserlebnis zu gewährleisten.

4.1 Backend und REST-API

Im Backend-Bereich kommt eine API (nach REST-Architektur) zum Einsatz, die gemäß dem Open-API 3-Standard dokumentiert wurde. Die API-Dokumentation stellt alle Endpunkte, Datentransfer-Modelle sowie mögliche Fehlermeldungen in einer übersichtlichen Darstellung mittels Swagger dar.

Verbindungsdetails:

- **IP-Adresse:** <http://192.168.129.202/>
- **Open-API-Dokumentation:** <http://192.168.129.202:8080/docs>
- **Swagger UI:** <http://192.168.129.202:8080/swagger-ui/index.html#/>

Periodisch führt der Webservice Hintergrundprozess die automatische Bereinigung der Datenbank. Ein geplantes („scheduled“) Task überprüft im Fünf-Minuten-Takt den Status aktiver Fahrten und aktualisiert diesen bei Bedarf. Sämtliche Systemereignisse werden serverseitig detailliert protokolliert und für einen Zeitraum von 30 Tagen in der Konsole sowie auch als Dateien gespeichert, eine solche Log-Nachricht sieht in der Regel wie folgt aus:

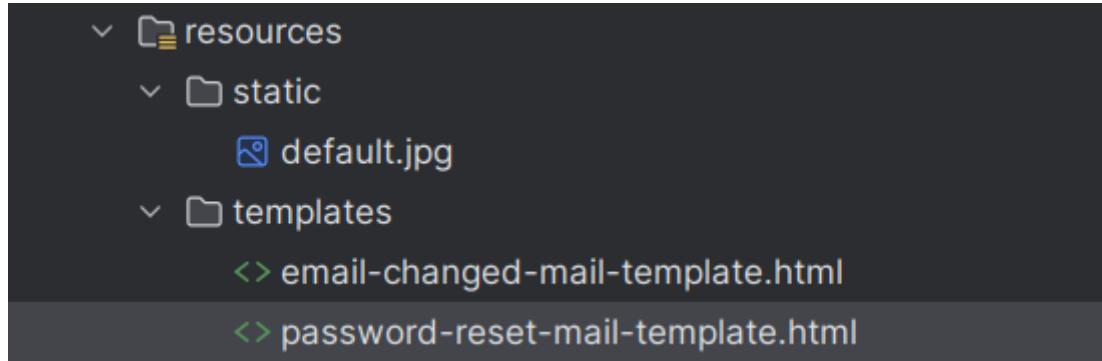
```
{"timestamp": "2025-04-04T11:10:21.626+02:00", "level": "ERROR", "thread": "http-nio-8080-exec-3", "requestId": "19ec30ef-6bc3-468a-8c73-cfc8c724f913", "logger": "c.v.middlewares.AuthenticationFilter", "target": "AUTH", "message": "Error on Auth occurred: JWT expired at 2025-04-03T09:45:57Z. Current time: 2025-04-04T09:10:21Z, a difference of 84264625 milliseconds. Allowed clock skew: 0 milliseconds."}
```

Im Log sind folgende Felder enthalten:

- **timestamp**: Der Zeitstempel des Log-Eintrags, der angibt, wann der Fehler aufgetreten ist.
- **level**: Das Schweregrad-Level der Meldung – in diesem Fall **ERROR**, was auf einen schwerwiegenden Fehler hinweist.
- **thread**: Der Name des Threads, der die Meldung erzeugt hat (hier: **http-nio-8080-exec-3**).
- **requestId**: Eine eindeutige ID zur Zuordnung der Anfrage im System. Pro Anfrage an den Server gibt es eine neue ID.
- **logger**: Die Klasse oder Komponente, die den Fehler protokolliert hat (**AuthenticationFilter**).
- **target**: Der betroffene Bereich oder Kontext – in diesem Fall **AUTH**, also die Authentifizierung.
- **message**: Die eigentliche Fehlermeldung mit genauer Beschreibung des Problems.

Die API bietet eine integrierte E-Mail-Funktion, die unter anderem beim Zurücksetzen eines Passworts verwendet wird. Zudem wird bei einer Änderung der E-Mail-Adresse in den Benutzereinstellungen eine Benachrichtigung an die bisher hinterlegte Adresse versendet.

Die E-Mail-Templates sind vollständig dynamisch und können programmatisch angepasst werden. Inhalte und Layouts lassen sich je nach Anwendungsfall flexibel konfigurieren.



Banana-Split-API 0 OAS 3.0

/docs

A ride-sharing API for connecting drivers and passengers.

[Terms of service](#)

[Support - Website](#)

Servers Authorize

User Endpoint Endpoints for user data. ^

Method	Endpoint	Description	Lock	Authorizer
GET	/bananasplit/api/v0/users/{id}	Get user by ID [USER, ADMIN]	🔒	∨
DELETE	/bananasplit/api/v0/users/{id}	Delete user by ID [ADMIN]	🔒	∨
PATCH	/bananasplit/api/v0/users/{id}	Change user information's [ADMIN]	🔒	∨
GET	/bananasplit/api/v0/users/me	Get authenticated user [USER, ADMIN]	🔒	∨
DELETE	/bananasplit/api/v0/users/me	Delete current user [USER, ADMIN]	🔒	∨
PATCH	/bananasplit/api/v0/users/me	Change authenticated user information's [USER, ADMIN]	🔒	∨
GET	/bananasplit/api/v0/users	Get all users [USER, ADMIN]	🔒	∨

Notification Endpoint Endpoints for user notifications. ^

Method	Endpoint	Description	Lock	Authorizer
POST	/bananasplit/api/v0/notifications/{id}/read	Mark a notification as read [USER, ADMIN]	🔒	∨
GET	/bananasplit/api/v0/notifications/{id}	Retrieve a specific notification by ID [USER, ADMIN]	🔒	∨
GET	/bananasplit/api/v0/notifications/me	Get all notifications for the authenticated user [USER, ADMIN]	🔒	∨

Picture Endpoint Endpoints for pictures. ^

Method	Endpoint	Description	Lock	Authorizer
GET	/bananasplit/api/v0/pictures/{id}	Get picture by ID [USER, ADMIN]	🔒	∨
PUT	/bananasplit/api/v0/pictures/{id}	Update picture by ID [USER, ADMIN]	🔒	∨

Rides Endpoint		
Endpoints for rides.		
GET	/bananasplit/api/v0/rides	Get all Rides [USER, ADMIN]
POST	/bananasplit/api/v0/rides	Create a Ride [USER, ADMIN]
POST	/bananasplit/api/v0/rides/{id}/join	Join a ride [USER, ADMIN]
GET	/bananasplit/api/v0/rides/{id}	Get a ride by ID [USER, ADMIN]
DELETE	/bananasplit/api/v0/rides/{id}	Delete a ride by ID [ADMIN]
GET	/bananasplit/api/v0/rides/{id}/participants	Get all Participants for a ride by ID [USER, ADMIN]
GET	/bananasplit/api/v0/rides/me	Get rides created by authenticated user [USER, ADMIN]
DELETE	/bananasplit/api/v0/rides/{id}/leave	Leave a ride [USER, ADMIN]
DELETE	/bananasplit/api/v0/rides/me/{id}	Delete a ride created by authenticated user [USER, ADMIN]
Authentication Endpoint		
Endpoints for user registration and authentication.		
POST	/bananasplit/api/v0/auth/signup	User Registration [NONE, USER, ADMIN]
POST	/bananasplit/api/v0/auth/password-reset/request	Password Reset Request [NONE, USER, ADMIN]
POST	/bananasplit/api/v0/auth/password-reset/confirm	Password Reset Confirm [NONE, USER, ADMIN]
POST	/bananasplit/api/v0/auth/logout	User Logout [USER, ADMIN]
POST	/bananasplit/api/v0/auth/login	User Login [NONE, USER, ADMIN]

4.2 Frontend

- **Design:** Ein konsistentes Layout mit einer klar strukturierten Navbar, Content-Bereichen und Footer.

4.3 Datenbank und Speicher

Die Datenbankstruktur wird beim ersten Start der API automatisch angelegt.

- **Struktur:** Benutzer-, Fahrten- und Nachrichtenverwaltung
- **Sicherheit:** Passwort-Verschlüsselung und gesicherte Authentifizierungsprozesse

5. Benutzeroberfläche und Funktionalitäten

5.1 Allgemeine Navigation

Jede Seite innerhalb der Anwendung verfügt über folgende Elemente:

- **Navbar:**

- Logo (verlinkt zum Dashboard)
- Home-Button
- Profil-Button (führt zur Profilverwaltung)
- Messages-Button (zeigt Benachrichtigungen)
- Rides-Button (Fahrtenübersicht)
- Logout-Button

- **Footer:**

- Copyright
- Impressum
- Datenschutzinformationen



5.2 Dashboard

Nach dem Login wird der Nutzer auf das Dashboard weitergeleitet. Hier erhält er eine Übersicht über:

- Angebote für Fahrten
- Direktzugriff auf alle wichtigen Funktionen (Fahrten anbieten, suchen, Nachrichten, Profilverwaltung)

The screenshot displays the BananaSplit Ride dashboard with the following sections:

- Teile deine Fahrt**: A section to share rides, with the sub-instruction "Finde Mitfahrmöglichkeiten, spare Kosten und lerne neue Leute kennen!". It features a yellow "Deine Fahrt anbieten" button.
- Fahrt suchen**: A search section to find rides, with the sub-instruction "Suche nach passenden Mitfahrmöglichkeiten in deiner Nähe.". It includes a search bar with the placeholder "Nach Stadt oder Ort suchen...".
- Nachrichten**: A messages section, with the sub-instruction "Hier findest du alle Neuigkeiten, Anfragen und Mitteilungen.". It features a dark blue "Zu den Nachrichten" button.
- Profil verwalten**: A profile management section, with the sub-instruction "Halte deine persönlichen Daten aktuell, um mehr Vertrauen bei anderen Nutzern zu schaffen.". It features a dark blue "Profil ansehen" button.

At the bottom of the dashboard, there is a dark brown footer bar containing the following links: "BananaSplit Ride", "© 2012 Mitfahrzentrale - Alle Rechte vorbehalten", "Impressum", "Datenschutz", and "Logout".

5.3 Registrierung und Login + Loading Screen im Login

Registrierung

- **Schritte:**

- Auf der Startseite registrieren
- Datenschutz akzeptieren

- **Eingabefelder:**

- E-Mail
- Benutzername
- Passwort

Login

- **Schritte:**

1. Login mittels E-Mail und Passwort
2. Bei vergessenem Passwort: Über den "Passwort vergessen"-Link eine E-Mail (via Google SMTP) erhalten, um das Passwort zurückzusetzen.

Loading Screen um Benutzerfreundliche UI bei Ladezeiten zu bieten

Login	Registration
E-Mail: E-Mail-Adresse	E-Mail: E-Mail-Adresse
Passwort: Passwort	Benutzername: Benutzername
Anmelden	← Registration
Passwort vergessen?	Ich habe die Datenschutzerklärung gelesen und stimme Ihnen zu.
Du hast noch kein Konto? Registrieren	Registrieren



5.4 Profilverwaltung

Auf der Profilseite können Nutzer ihre persönlichen Daten verwalten:

- **Funktionen:**

- Profilbild hochladen (Unterstützung gängiger Bildformate wie png, jpeg, gif, etc.)
- Passwort ändern
- Benutzerkonto löschen
- Daten erweitern und bearbeiten (Benutzername, E-Mail, Vorname, Nachname, Telefonnummer, Adresse)


Profil

Username:

Email:

Vorname:

Nachname:

Telefonnummer:

Straße:

Nr.:

PLZ:

Ort:

Ändern

5.5 Fahrtenverwaltung

Fahrten erstellen

- **Eingabefelder:**
 - **Startadresse:** Straße, Hausnummer, PLZ, Stadt, Land
 - **Zieladresse:** Straße, Hausnummer, PLZ, Stadt, Land
 - **Zusätzliche Informationen:**
 - Pflicht: Datum und Uhrzeit des Fahrtbeginns, Anzahl freier Plätze
 - Optional: Bemerkungen
- **Aktionen:**
 - Fahrt erstellen
 - Eigene Fahrten löschen
 - Mitfahrer aus der Fahrt entfernen

Fahrten suchen und beitreten

- **Funktionen:**
 - Filter- und Sortierungsmöglichkeiten in der Übersicht
 - Details zu Fahrten:
 - Ersteller
 - Von – Bis
 - Datum und Uhrzeit
 - Verfügbare Plätze
 - Darstellung einer Leaflet Map mit Start- und Zielpunkt (Marker in Grün und Rot)
 - Beitreits Funktion zur gewünschten Fahrt

Mitfahrzentrale

Finde Fahrten, die zu deinem Ziel führen oder erstelle deine eigene Fahrt.

Startort:	Zielort:	Fahrer:
Startort eingeben	Zielort eingeben	Fahrername eingeben
Sortieren nach:	Richtung:	Status:
Erstellt	Absteigend	Aktiv
Filter anwenden		
Fahrt erstellen		

Prutting → Rosenheim
Ersteller: test
Freie Plätze: 1
Datum: 8.4.2025
Uhrzeit: 10:55



Waldkraiburg → Traunstein
Ersteller: Thomas_
Freie Plätze: 3
Datum: 8.4.2025
Uhrzeit: 10:02



Fahrtdetails

Ersteller: sergej-stk1

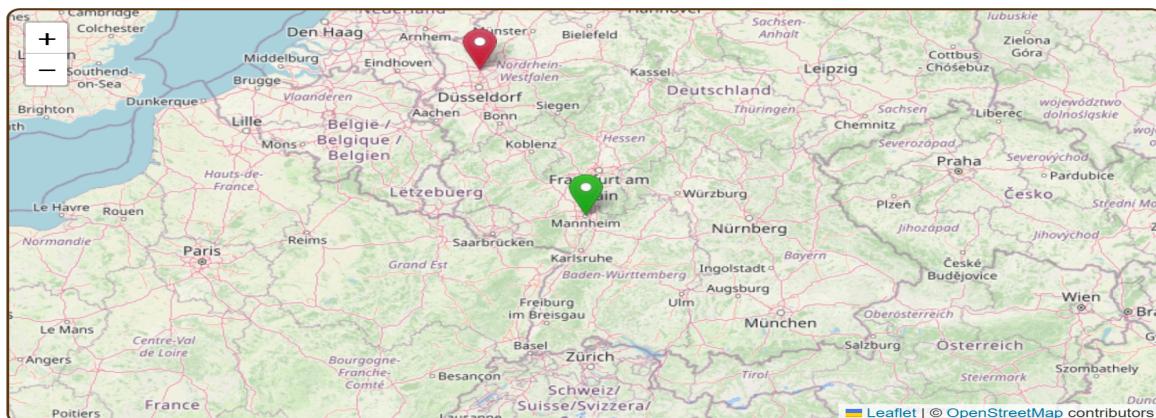
Von: Mannheim

Nach: Duisburg

Datum: 29.12.2025

Uhrzeit: 19:00

Freie Plätze: 2



Zurück

Beitreten

Verlassen

Teilnehmer



Thomas_ wirklich entfernen?

Entfernen

Abbrechen

5.6 Nachrichten und Benachrichtigungen

- **Funktionen:**

- Einsicht und Verwaltung eingehender Nachrichten
- "Alles als gelesen markieren"-Funktion, die alle Nachrichten als gelesen markiert und nach Neuladen der Seite entfernt

- **Automatisierung:**

- Nachrichten werden automatisch generiert (z. B. bei Erstellung einer Fahrt oder bei Beitritt zu einer Fahrt).
- Nutzer können keine eigenen Nachrichten erstellen oder versenden.

Nachrichten

Alle als gelesen markieren

Es wurde die Fahrt von 'Prutting' bis 'Rosenheim' erfolgreich angelegt.

Mitteilung

Es wurde die Fahrt von 'Prutting' bis 'Rosenheim' erfolgreich angelegt.

Als gelesen markieren

6. Interaktive Features und Toast-Benachrichtigungen

Im gesamten Projekt sind individuelle Toasts implementiert. Diese stellen sicher, dass der Nutzer zu jeder wichtigen Aktion ein Feedback erhält – sei es eine Erfolgsmeldung oder eine Fehlermeldung.

- **Implementierung:**

- Die Toasts wurden als wiederverwendbarer Hook entwickelt.
- Verwendung: `showToast('Nachricht', success/error)`

- **Vorteil:**

- Einheitliches Feedback an den Nutzer an jeder Stelle der Anwendung.



Erfolgreich der Fahrt beigetreten!

7. Technische Dokumentation via Swagger UI

Die technische Dokumentation der API erfolgt über Swagger UI. Dies ermöglicht:

- **Transparenz:** Jeder API-Endpunkt, seine Eingaben, Ausgaben und mögliche Fehler werden detailliert dokumentiert.
- **Testbarkeit:** Entwickler können direkt über die Swagger-Oberfläche API-Aufrufe testen.
- **Verfügbarkeit:**
 - **Swagger UI URL:** <http://192.168.129.202:8080/swagger-ui/index.html#/>

8. Implementierungsdetails und Best Practices

8.1 Code-Struktur und Modularität

- **Frontend:**
 - Klare Trennung von Komponenten (Navbar, Footer, Seiteninhalte)
 - Wiederverwendbare Hooks und Komponenten (z. B. Toast-Benachrichtigungen)
- **Backend:**
 - REST-API mit sauber dokumentierten Endpunkten
 - Authentifizierungsprozesse (Registrierung, Login, Passwortzurücksetzung)
- **Datenvalidierung:**
 - Überprüfung aller Eingabefelder auf Serverseite, um Sicherheitslücken zu vermeiden.

8.2 Sicherheit und Datenschutz

- **Datenschutz:**
 - Einhaltung gängiger Datenschutzbestimmungen
 - Fiktive Daten im Impressum und in Datenschutzrichtlinien (anpassbar für reale Einsatzszenarien)
- **Sicherheitsmaßnahmen:**
 - Verschlüsselte Passwörter
 - Sichere Übertragung der Daten (HTTPS empfohlen)

8.3 Testing und Qualitätssicherung

- **Unit-Tests und Integrationstests:** Regelmäßige Überprüfung der Funktionalitäten
 - **Feedback-Schleifen:** Einbau von Toast-Benachrichtigungen, um dem Nutzer direkt Rückmeldung zu geben
 - **Dokumentation:** Laufende Aktualisierung der technischen Dokumentation über Swagger UI
-

9. Test und Qualitätssicherung

9.1 Teststrategien

- **Manuelle Tests:**
 - Überprüfung aller Funktionalitäten (Registrierung, Login, Profilverwaltung, Fahrtenverwaltung, Nachrichten)
 - **Automatisierte Tests:**
 - Integration von Unit-Tests im Backend
 - End-to-End-Tests zur Sicherstellung der Benutzerführung
 - **Feedback:**
 - Einholen von Nutzerfeedback zur kontinuierlichen Verbesserung
-

10. Fazit und Ausblick

Die Mitfahrzentrale **Bananasplit** bietet den Nutzern eine intuitive Plattform zur Organisation von Fahrgemeinschaften. Durch die Einbindung moderner Technologien, klar strukturierte Benutzeroberflächen und eine umfassende API-Dokumentation über Swagger wird ein hoher Standard in Sachen Usability und technischer Qualität erreicht.

Ausblick:

- Erweiterungen der Funktionalitäten (z. B. dynamische Routenberechnung)
 - Weitere Sicherheitsfeatures und Optimierung der Performance
 - Integration von zusätzlichen Feedback-Mechanismen für Nutzerbewertungen
-

11. Anhang

11.1 Glossar

- **Dashboard:** Zentrale Übersichtsseite nach dem Login
- **Navbar:** Navigationsleiste, die auf jeder Seite verfügbar ist
- **Toast:** Kleine Benachrichtigung, die kurzzeitig angezeigt wird
- **Swagger UI:** Tool zur Dokumentation und Interaktion mit der API

11.2 Abkürzungen

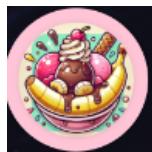
- **API:** Application Programming Interface
- **CO₂:** Kohlendioxid
- **HTTP:** Hypertext Transfer Protocol

11.3 Links und Ressourcen

- **Projekt-IP:** <http://192.168.129.202/>
- **Swagger UI:** <http://192.168.129.202:8080/swagger-ui/index.html#/>

12. Android App in Kotlin

Die Android-Applikation wurde in Kotlin entwickelt und bildet das Konzept der Webanwendung in einer mobilen Umgebung ab. Die App bietet eine benutzerfreundliche Oberfläche, die es dem Nutzer ermöglicht, alle Funktionen der Mitfahrzentrale intuitiv zu durchklicken – ganz analog zur Web-App. Es wurden alle Bereiche implementiert, die auch in der Webanwendung vorhanden sind, wobei der Fokus auf einer übersichtlichen und reaktionsschnellen Navigation liegt. Dabei wurde bewusst darauf verzichtet, echte Funktionalitäten zu integrieren, sodass die App als Prototyp und Demonstrationsmodell fungiert.



Benutzeroberfläche und Navigation

Die Benutzeroberfläche wurde mit Blick auf ein modernes, cleanes Design gestaltet, das sich an den gängigen Material Design-Richtlinien orientiert. Die App beinhaltet folgende Hauptbereiche:

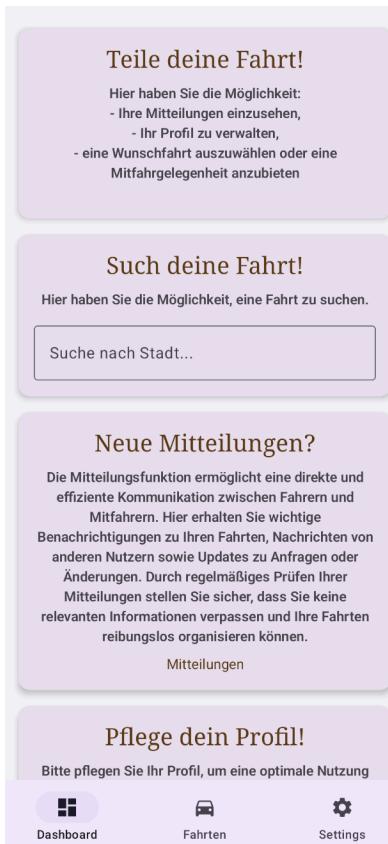
- **Start- und Login-Screen:**

Beim ersten Start der App werden die Nutzer mit einem Willkommensbildschirm begrüßt. Hier können sie sich registrieren oder sich direkt anmelden. Das Design ist minimalistisch, um den Einstieg einfach zu halten.

Login	Zurück	Zurück
E-Mail	Passwort ändern	Registrierung
Passwort	Neues Passwort	E-Mail-Adresse
Login	Neues Passwort wiederholen	Benutzername
Passwort vergessen?	Passwort ändern	Passwort
Registrieren		Passwort wiederholen
		Registrieren

- **Dashboard:**

Nach dem Login gelangt der Nutzer auf ein Dashboard, das als zentrale Übersicht dient. Hier sind alle wesentlichen Navigationsoptionen vorhanden, wie z. B. das Erstellen und Suchen von Fahrten, der Zugriff auf Nachrichten und die Profilverwaltung. Das Dashboard ist so strukturiert, dass die wichtigsten Aktionen direkt erkennbar und leicht zugänglich sind.



- **Navigationsleiste (Navbar):**

Eine fest integrierte Navigationsleiste ermöglicht es dem Nutzer, jederzeit zwischen den verschiedenen Bereichen der App zu wechseln. Diese Leiste enthält Icons und Texte, die zu den Bereichen „Dashboard“, „Fahrten“ und „Settings“ führen.

Funktionsbereiche und Simulation

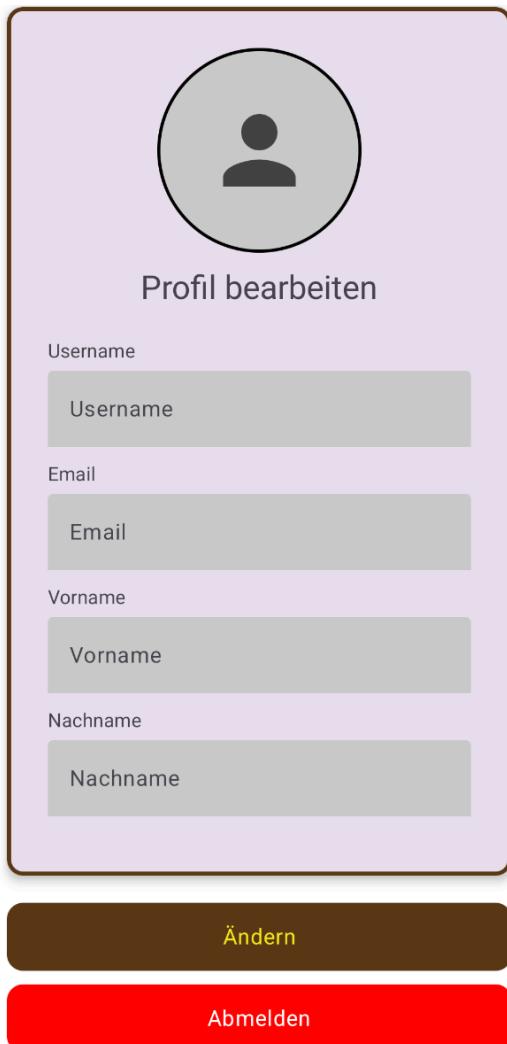
Obwohl keine echten Funktionalitäten implementiert wurden, wurde besonderer Wert darauf gelegt, dass alle Bereiche der Web-App auch in der Android-App simuliert werden. Dies beinhaltet:

- **Registrierung und Login:**

Die Registrierungs- und Login-Bildschirme bieten alle erforderlichen Eingabefelder (E-Mail, Benutzername, Passwort) und stellen einen nahtlosen Übergang zur Hauptanwendung sicher. Fehlermeldungen und Bestätigungstexte werden durch Dummy-Texte simuliert, um den Ablauf realistisch wirken zu lassen.

- **Profilverwaltung:**

Der Profilbereich erlaubt dem Nutzer, ein Profilbild hochzuladen und persönliche Daten wie Vorname, Nachname, E-Mail, Telefonnummer und Adresse anzusehen und zu bearbeiten. Die UI simuliert hierbei die Interaktion, ohne dass die Daten tatsächlich verarbeitet werden.



- **Fahrtenverwaltung:**

In diesem Bereich können Nutzer „Fahrten erstellen“ und „Fahrten beitreten“. Alle Eingabefelder (Start- und Zieladresse, Datum, Uhrzeit, freie Plätze) sind vorhanden. Zusätzlich wird eine Karte angezeigt, die mit Dummy-Markern die Start- und Zielpunkte visualisiert. Der gesamte Ablauf wurde so gestaltet, dass er den realen Funktionsfluss der Web-App widerspiegelt, jedoch ohne echte Backend-Anbindung.

The screenshot shows the 'Mitfahrzentrale' app interface. On the left, there's a main header 'Mitfahrzentrale' with a sub-instruction 'Finde Fahrten, die zu deinem Ziel führen oder erstelle deine eigene Fahrt.' Below it is a large brown button labeled 'Fahrt erstellen'. To the right, there are three cards, each representing a trip: 'Fahrt 1: Berlin nach Hamburg', 'Fahrt 2: München nach Stuttgart', and 'Fahrt 3: Köln nach Düsseldorf'. Each card has a blue 'Fahrt beitreten' button. At the bottom of the main screen, there are navigation arrows and the text 'Seite 1 von 1'. On the right side, there's a separate panel titled '← Fahrtdetails' containing a 'Fahrt Detail' card with the text 'Detailbeschreibung der Fahrt' and another blue 'Fahrt beitreten' button.

- **Nachrichten und Benachrichtigungen:**

Die Nachrichtenfunktion wurde ebenfalls simuliert. Nutzer können in einer Nachrichtenübersicht Dummy-Nachrichten einsehen, die wie in der Web-App automatisch generiert werden. Dabei wird auch die Funktion „Alles als gelesen markieren“ angezeigt, um den Nutzerfluss zu verdeutlichen.

Fahrt beigetreten!