Abschlussdokumentation Caring

Mautendorfer, Stich, Steinberger, Wirnsberger

5AHIF, SYP-BSD **LIN, MUH, REL**

# **Inhaltsverzeichnis**

[**Inhaltsverzeichnis** 2](#_Toc38821486)

[**1.** **einleitung** 3](#_Toc38821487)

[**2.** **Lastenheft** 3](#_Toc38821488)

[Allgemeines 4](#_Toc38821489)

[Verteiler und Freigabe 4](#_Toc38821490)

[Benutzer / Zielgruppe 4](#_Toc38821491)

[Ziele des Anbieters 4](#_Toc38821492)

[Ziele und Nutzen des Anwenders 4](#_Toc38821493)

[Systemvoraussetzungen 4](#_Toc38821494)

[Ressourcen 4](#_Toc38821495)

[Technische Anforderung 5](#_Toc38821496)

[Funktionelle Anforderung 5](#_Toc38821497)

[**3.** **Technischer Ansatz** 5](#_Toc38821498)

[**4.** **Modellierung** 7](#_Toc38821499)

[**5.** **User Stories** 9](#_Toc38821500)

[Activity Diagramm 15](#_Toc38821501)

[**6.** **Umsetzung** 15](#_Toc38821502)

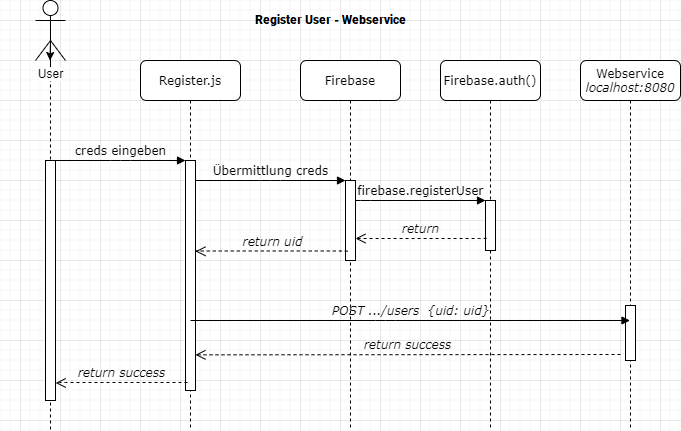
[Datenmodell 16](#_Toc38821503)

[Webservice Schnittstellendokumentation 16](#_Toc38821504)

[SQL-Statements 17](#_Toc38821505)

[Trigger 18](#_Toc38821506)

[Sequenzdiagram 19](#_Toc38821507)

[ 19](#_Toc38821508)

[**7.** **Planung des Prozesses** 20](#_Toc38821509)

[Sprintplanung (Iteration 3) 20](#_Toc38821510)

[**7.** **RESÜMEE** 22](#_Toc38821511)

[Konfliktmanagement 22](#_Toc38821512)

[**8.** **Anhang** 23](#_Toc38821513)

[Bildverzeichniss 23](#_Toc38821514)

[Code Snippetverzeichniss 23](#_Toc38821515)

[Graphverzeichniss 23](#_Toc38821516)

# **einleitung**

Dieses Dokument dient als abschliedene Dokumentation unseres BSD-Projektes. Wir werden mithilfe von verschiedenen Dokumenten, Bildern und Diagrammen festhalten wie der Arbeitsablauf und die Entwicklung vorrangegangen sind.

Zur genauen Einsicht findet man alle Dateien auf GitHub: <https://github.com/HTL-Villach-Steinbej/Caring>

# **Lastenheft**

Das Lastenheft setzt grundsetzliche Dinge wie z.B. Programmiersprache fest, und klärt Ziele und Ausmaß des Projekts ab.

## Allgemeines

### Ziel und Zweck dieses Dokuments

Dieses Lastenheft beschreibt die genauen Abläufe und Anforderungen an das Programm Caring. Es soll jegliche Fragen abklären und die Anforderungen detailiert beschreiben.

### Abkürzungen

Caring

### Ablage, Gültigkeit und Bezüge zu anderen Dokumenten

Keine weiteren Dokumente

## Verteiler und Freigabe

### Verteiler für dieses Lastenheft

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rolle | Name | Telefon | E-Mail | Bemerkungen |
| Projektleiter | Steinberger |  |  |  |
| Dev-Team | Stich |  |  |  |
| Dev-Team | Wirnsberger |  |  |  |
| Dev-Team | Mautendorfer |  |  |  |

## Benutzer / Zielgruppe

Alle Personen, die auf Mietwägen angewiesen sind bzw. das Angebot nutzen wollen

## Ziele des Anbieters

Ziel ist es, eine Infrastruktur zu errichten, die es Nutzern ermöglicht, einfach und bequem jederzeit ein Auto auszuleihen und dann bei Bedarf wieder abzustellen.

## Ziele und Nutzen des Anwenders

Hilfe beim Ausbau des Autonetzes

## Systemvoraussetzungen

Android Smartphone 6.0 bzw. Windows 7, 8, 10 oder später

## Ressourcen

Autos, Mitarbeiter, Infrastruktur

**Muss-Kriterien:** Es soll eine Anwendung erstellt werden, die es ermöglicht, sich als User bei Caring anzumelden und dann in weiterer Folge frei herumstehende KFZs über die App ausborgen zu können. Die Abrechnung erfolgt über Kreditkarte oder PayPal und die Kosten werden nach verstrichener Zeit berechnet. Falls durch einen vorherigen Benutzer ein Schaden verursacht wurde, kann man diesen per App melden, um sich selbst zu entlasten.

**Erweiterung**: Später ist es auch angedacht, dass Privatpersonen durch ein „Anmeldungsverfahren“ befugt werden, Autos anzubieten.

## Technische Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** |  | **Nichttechnischer Titel** | |  | | |
| **Quelle** |  | | **Verweise** |  | **Priorität** |  |

### Technische Umsetzung

Das Projekt wird in 3 Phasen eingeteilt:

* Standard Desktop-Anwendung (C#, JS, SQL)
* Integration von Webservice (Webservice)
* Hinzufügen von Android-App (Android Studio)

Als Datenbank werden wir SQL verwenden.

### Wechselwirkungen

Die Wechselwirkung zwischen Kunden und uns soll anfangs nur über die Applikation erfolgen, soll später aber noch auf Website ausgeweitet werden.

### Risiken

Betrug durch Kunden, Versicherungspolitik, nicht vorhandener Markt für das System, Schwierigkeiten beim Aufbau des Netzes

### Vergleich mit bestehenden Lösungen

Tier ist das einzig vergleichbare Unternehmen. Unterschied ist das Fortbewegungsmittel

### Schätzung des Aufwands

Sehr hoch

## Funktionelle Anforderung

Folgende Funktionen werden in der Grundversion implementiert:

* User registrieren (inkl. Datenbank)
* User anmelden
* Zahlungsmethode hinzufügen
* Auto ausleihen (Anfrage stellen)
* Schaden am Auto melden

Analysetools

# **Technischer Ansatz**

In Bild 1 können Sie erkennen die technische Umsetzung vom Frontend und Backend und deren Kommunikation.

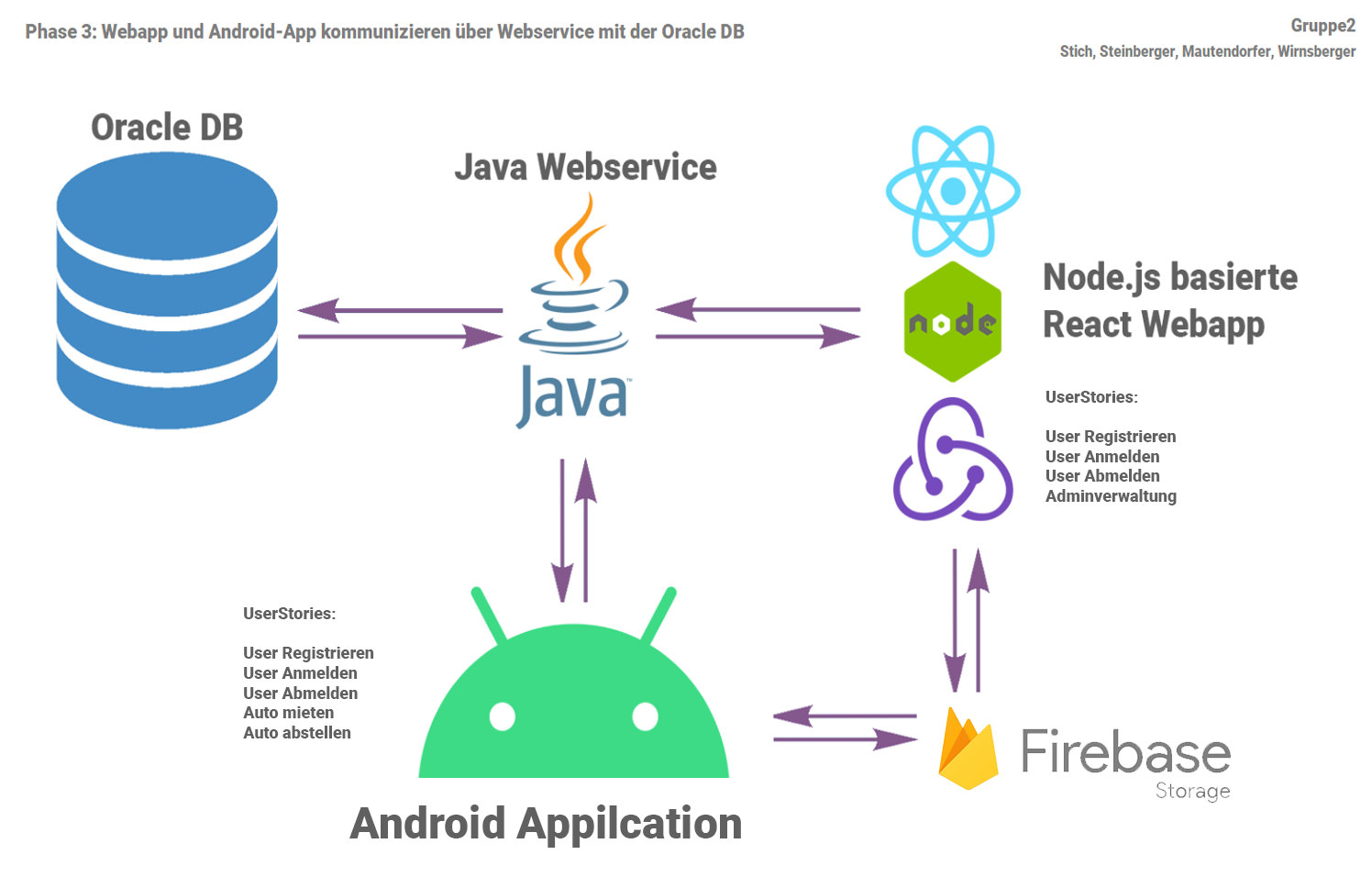


Bild - Systemarchitektur

# **Modellierung**

Eine Vorgabe des Projekts war die Verwendung des Prozessmodells Scrum. Dementsprechend haben wir User Stories formuliert und diese in einem Product Backlog zusammengefasst.

Um ein besseres Verständnis für das Projekt zu erzeugen, verwenden wir Diagramme und graphische Übersichten.

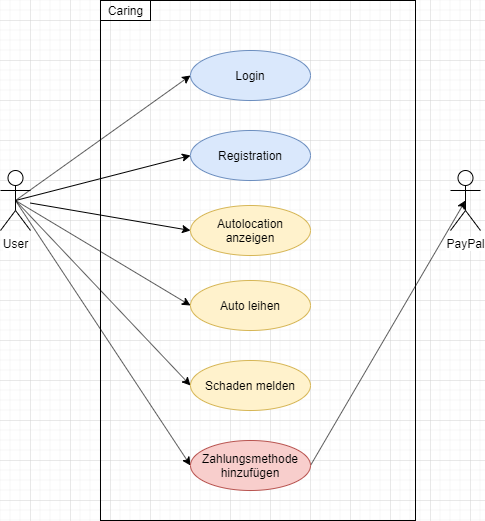


Bild II – Use case diagram

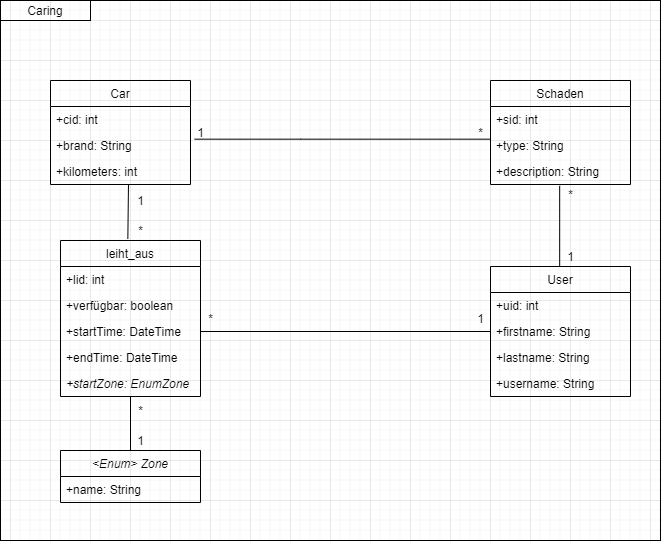


Bild III – Domainmodell

Das oben gezeigte Bild 6 zeigt die statische Sicht des Projekts mithilfe eines UML Klassendiagramms.

# **User Stories**

Abgeleitet aus den oben gezeigten Diagrammen ergeben sich folgende User Stories



Bild – Übersicht User-Stories



Bild V – User Story Map Teil 1

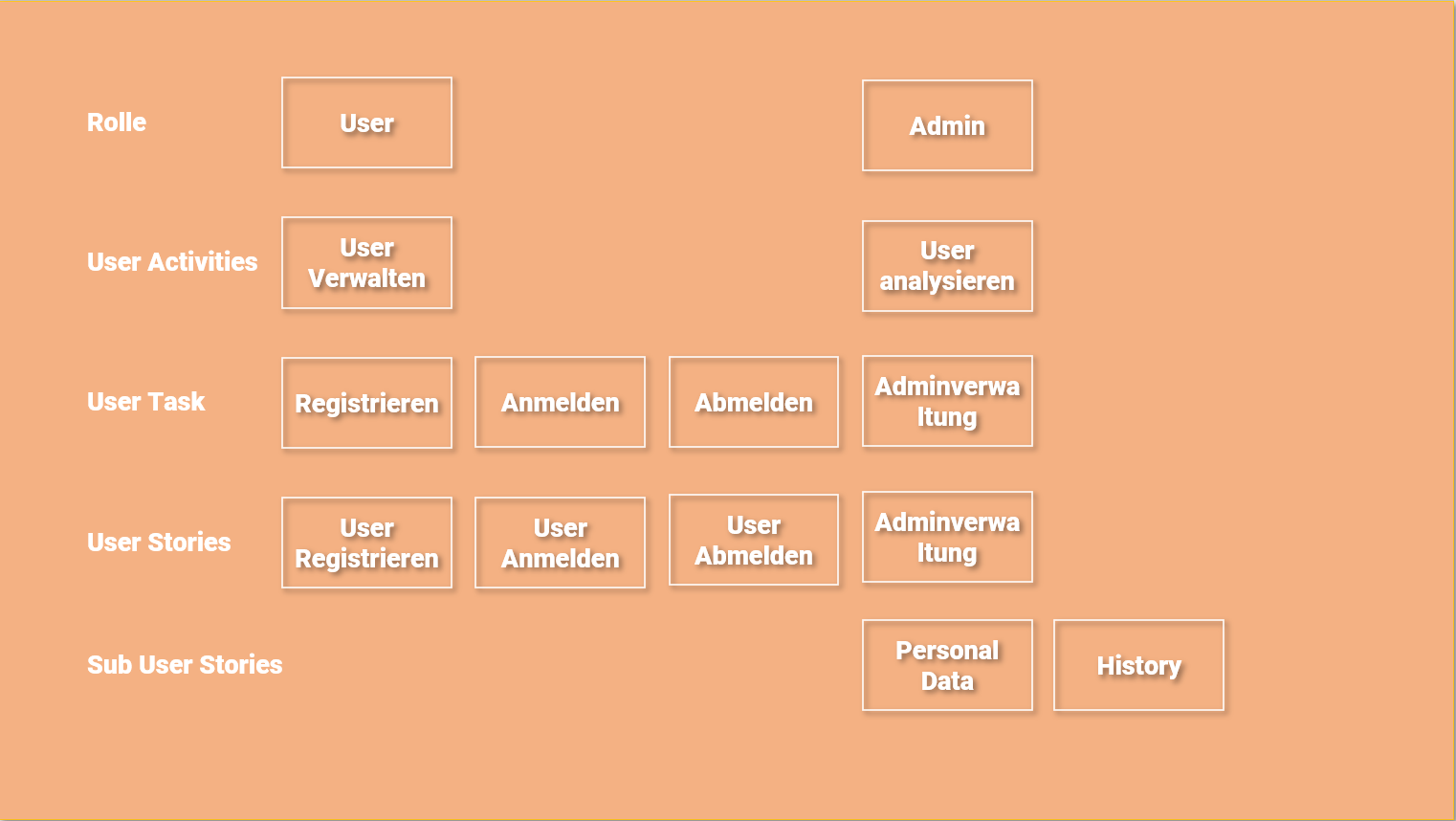


Bild VI - User Story Map Teil 2

### Autolocation anzeigen

|  |  |  |
| --- | --- | --- |
| **Rolle** | **User Story** | **Grund** |
| Als Benutzer | In der Karte angezeigt bekommen wo sich das aktuelle Auto befindet | Auffinden der Autos |

**Akzeptanzkriterien:**

* Standort des Autos muss abrufbar sein
* Auto muss vorhanden sein
* Benutzer muss ein Auto auswählen
* Benutzer muss angemeldet sein

|  |  |  |  |
| --- | --- | --- | --- |
| **Testcase** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer hat GPS aktiviert | Sucht Auto | Benutzer bekommt Auto auf der Map angezeigt |
| 2 | Benutzer hat kein GPS aktiviert | Sucht Auto | Fehler, da man nicht weiß wo sich der Benutzer befindet |

### Schaden melden

|  |  |  |
| --- | --- | --- |
| **Rolle** | **User Story** | **Grund** |
| Als Benutzer | möchte ich einen Schaden melden |  |

**Akzeptanzkriterien:**

* Benutzer muss angemeldet sein
* Schaden muss existieren

|  |  |  |  |
| --- | --- | --- | --- |
| **Testcase** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer ist angemeldet | Auto hat einen Schaden, Benutzer meldet Schaden | Schaden wurde gemeldet |
| 2 | Benutzer nicht angemeldet | Auto hat Schaden | Schaden kann nicht gemeldet werden |

### Zahlungsmittel hinzufügen

|  |  |  |
| --- | --- | --- |
| **Rolle** | **Userstory** | **Grund** |
| Als Benutzer möchte ich | Bezahlmittel verbinden | Um mit dem Auto fahren zu können |

**Akzeptanzkriterium**:

* Auswahl der Zahlungsmethode PayPal oder Kreditkarte
* Verknüpfung erfolgreich, wenn PayPal Login funktioniert hat oder Kreditkarten Daten richtig eingegeben
* Bei Kreditkarten Verknüpfung müssen alle Felder ausgefüllt werden
* Bei PayPal muss Login beim PayPal account funktionieren

|  |  |  |  |
| --- | --- | --- | --- |
| **Testfälle** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer hat noch kein Zahlungsmittel verbunden | Benutzer verknüpft Account mit PayPal und gibt die Richtigen PayPal Anmeldedaten ein | PayPal Account wird verknüpft |
| 2 | Benutzer hat noch kein Zahlungsmittel verbunden | Benutzer wählt Kreditkarte als Zahlungsmittel aus und gibt die Richtige Kreditkartennummer, Ablaufdatum ein | Kreditkarte wird verknüpft |
| 3 | Benutzer hat noch kein Zahlungsmittel verbunden | Benutzer wählt PayPal aus und gibt falschen Anmeldedaten ein | Verknüpfung nicht erfolgreich |
| 4 | Benutzer hat noch kein Zahlungsmittel verbunden | Benutzer wählt Kreditkarte aus und gibt falsche Daten ein | Verknüpfung nicht erfolgreich |

### Registrieren

|  |  |  |
| --- | --- | --- |
| **Rolle** | **Userstory** | **Grund** |
| Als Benutzer möchte ich | mich registrieren |  |

**Akzeptanzkriterium**:

* Pflichtfelder: Name, Email, Passwort.
* Passwort: mindestens 6 Zeichen lang und muss mindestens eine Zahl enthalten
* Email muss eindeutig sein

|  |  |  |  |
| --- | --- | --- | --- |
| **Testfälle** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer mit dem der EMAIL  [test@test.com](mailto:test@test.com) gibt es nicht | Er gibt alle Pflichtfelder der Akzeptanzkriterien richtig ein unter dem Namen *Tester01* | Der Account wird erstellt |
| 2 | Benutzer der EMAIL  [test@test.com](mailto:test@test.com) gibt es nicht | Es wird mindestens eines der Pflichtfelder nicht der Akzeptanzkriterien entsprechend ausgefüllt unter dem Namen *Tester01*. | Der Account wird nicht erstellt |
| 3 | Benutzer mit der EMAIL  [test@test.com](mailto:test@test.com) gibt es bereits. | Er gibt alle Pflichtfelder, die den Akzeptanzkriterien entsprechen richtig ein unter der EMAIL  [test@test.com](mailto:test@test.com) | Es wird darauf hingewiesen, dass es diese EMAIL schon gibt |

### Anmelden

|  |  |  |
| --- | --- | --- |
| **Rolle** | **User Story** | **Grund** |
| Als Benutzer | möchte ich mich mit meinem vorhandenen Account anmelden | Benutzung der Plattform |

**Akzeptanzkriterien:**

* email und Passwort Feld müssen ausgefüllt sein. (Pflichtfelder)
* Eingegebene EMAIL muss vorhanden sein.
* Passwort muss zur EMAIL passen.

|  |  |  |  |
| --- | --- | --- | --- |
| **Testcase** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer mit der EMAIL  [test@test.com](mailto:test@test.com)hat bereits ein Konto | Er gibt die richtige email und das passende Passwort dazu ein. | Benutzer ist eingeloggt |
| 2 | Benutzer mit der EMAIL  [test@test.com](mailto:test@test.com) hat noch kein Konto | Er gibt den noch nicht registrierte Email und ein Passwort ein. | Login ist fehlgeschlagen |
| 3 | Benutzer mit der EMAIL  [test@test.com](mailto:test@test.com) hat bereits ein Konto | Er gibt einen richtige Email, aber das Falsche Passwort ein. | Login fehlgeschlagen |
| 4 | Benutzer mit der EMAIL  [test@test.com](mailto:test@test.com) hat bereits ein Konto | Es wird mindestens eines der Pflichtfelder nicht der Akzeptanzkriterien entsprechend ausgefüllt unter der EMAIL  [test@test.com](mailto:test@test.com) | Login fehlgeschlagen |

### Auto anbieten

|  |  |  |
| --- | --- | --- |
| **Rolle** | **User Story** | **Grund** |
| Als Benutzer | möchte ich mein Auto anbieten | Bereitstellung des Autos |

**Akzeptanzkriterien:**

* User muss Eingeloggt sein
* User muss gültigen Account besitzen
* Pflichtfelder Location, Marke, Laufleistung, Bezeichnung müssen richtig ausgefüllt werden

|  |  |  |  |
| --- | --- | --- | --- |
| **Testcase** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer ist eingeloggt | Er füllt die Pflichtfelder richtig aus und bietet das Auto an | Auto kann nun gemietet werden |
| 2 | Benutzer ist eingeloggt | Er füllt die Pflichtfelder nicht richtig aus und bietet das Auto an | Auto anbieten fehlgeschlagen |

### Auto mieten

|  |  |  |
| --- | --- | --- |
| **Rolle** | **Userstory** | **Grund** |
| Als Benutzer möchte ich | Ein Auto meiten | Um mit dem Auto fahren zu können |

**Akzeptanzkriterium**:

* Gültige Zahlungsmethode hinzugefügt
* Auto ist nicht an einen anderen laufenden Mietprozess gebunden

|  |  |  |  |
| --- | --- | --- | --- |
| **Testfälle** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer will Auto mieten | Er wählt verfügbares Auto aus und klickt auf mieten | Auto wird gemietet |
| 2 | Benutzer will Auto mieten | Auto wird bereits vermietet | Auto wird gar nicht angezeigt, er kann es nicht mieten |

### Auto abstellen

|  |  |  |
| --- | --- | --- |
| **Rolle** | **User Story** | **Grund** |
| Als Benutzer | Möchte ich einen Mietvorgang abbrechen und beenden | Auto wird nicht mehr benötigt |

**Akzeptanzkriterien:**

* Benutzer muss sich in einem gültigen Mietvorgang befinden
* User muss Zahlungsmethode verknüpft haben

|  |  |  |  |
| --- | --- | --- | --- |
| **Testcase** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer beendet den Prozess | Klickt auf Beenden und eine neue Location muss eingegeben werden | Pauschale wird berechnet und User muss den Abstellplatz angeben |
| 2 | Benutzer beendet den Prozess ohne Angabe neuer Location | Klickt auf Beenden | User kann Vorgang erst beenden und ins HM zurückkehren wenn er einen gültigen Abstellplatz angegeben hat |

### 

### Admin-/Userverwaltung

|  |  |  |
| --- | --- | --- |
| **Rolle** | **Userstory** | **Grund** |
| Als Admin möchte ich | User verwalten und Daten analysieren | Um bessere Übersicht der User zu erhalten |

**Akzeptanzkriterium**:

* Angemeldeter Admin
* Zu verwaltende User sind vorhanden
* Daten zur Analyse müssen vorhanden sein

|  |  |  |  |
| --- | --- | --- | --- |
| **Testfälle** | **Vorbedingung** | **Aktion** | **Ergebnis** |
| 1 | Benutzer ist nicht Admin | Will auf das Admin Dashboard zugreifen | Kein Zugriff erlaubt |
| 2 | Benutzer ist Admin | Will auf das Admin Dashboard zugreifen | Zugriff wird erlaubt |

## Activity Diagramm

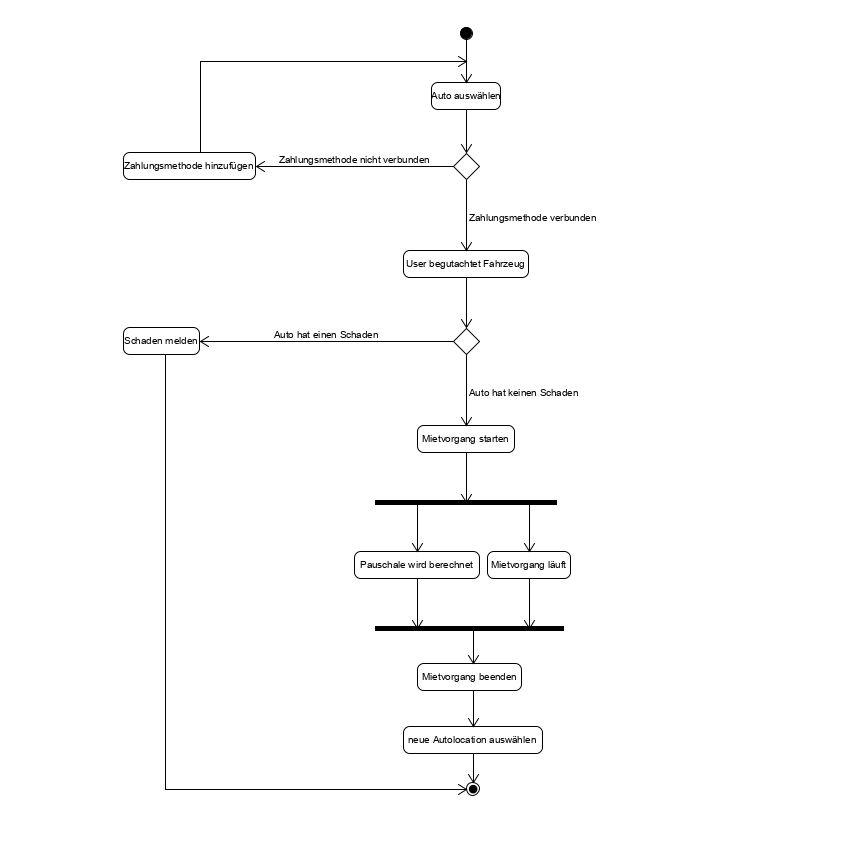


Bild VII – Activity Diagramm

Das ActivityDiagram klärt Fragen bezüglich des zeitlich-, funktionellen Ablaufs. Es beschreibt die User-Story „Auto ausleihen“. Bild 5.

# **Umsetzung**

Als Datenbank haben wir Firebase und Oracle verwendet. Diese werden per Webservice mit den Applikationen verbunden.

## Datenmodell

Das Datenmodell im Bild 9 zeigt die Struktur der Oracle Datenbank.

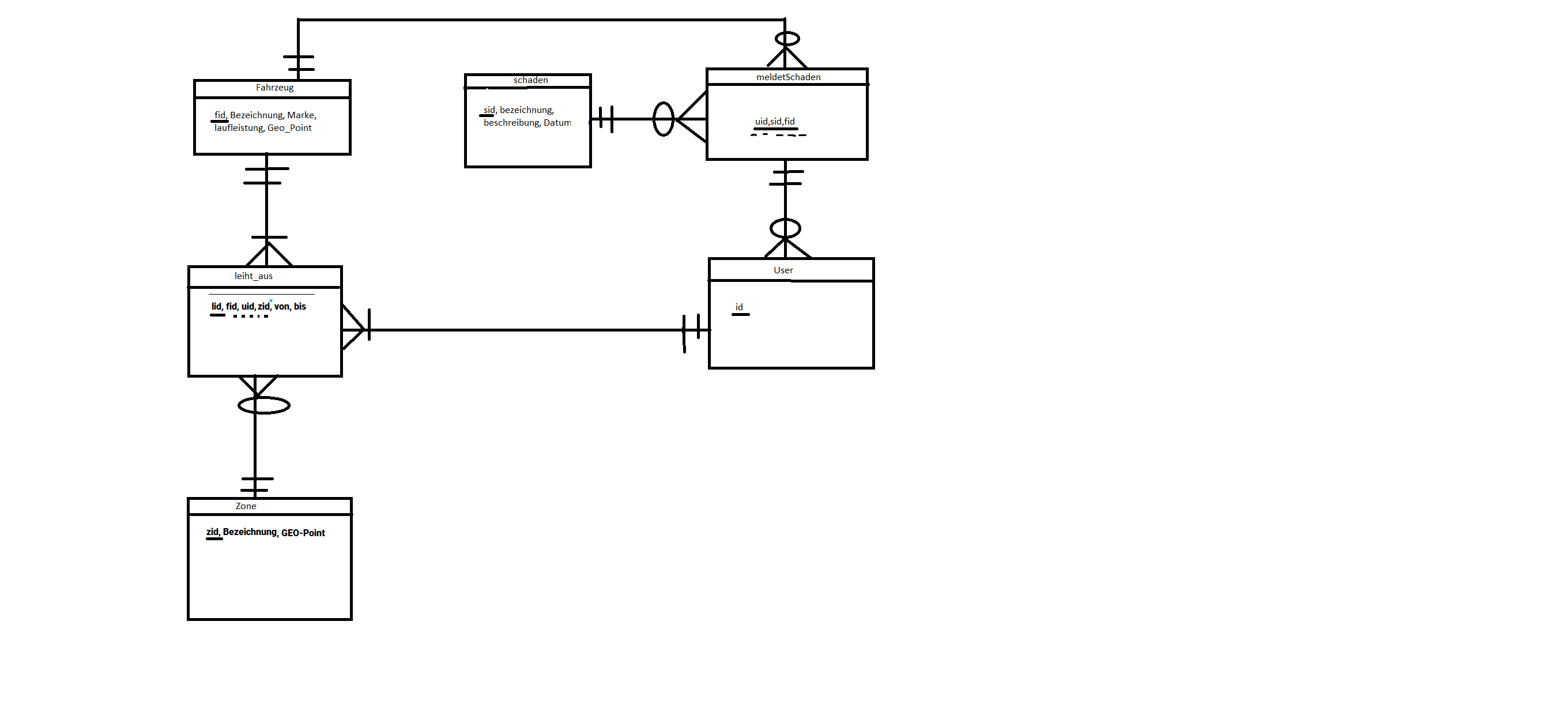


Bild VIII – ERD

## Webservice Schnittstellendokumentation

Die Schnittstellendefinition (Bild 10) bezieht sich ausschließlich auf Oracle. Alle unten gezeigten Routen sind verfügbar und können von den verschiedenen Applikationen aufgerufen werden.

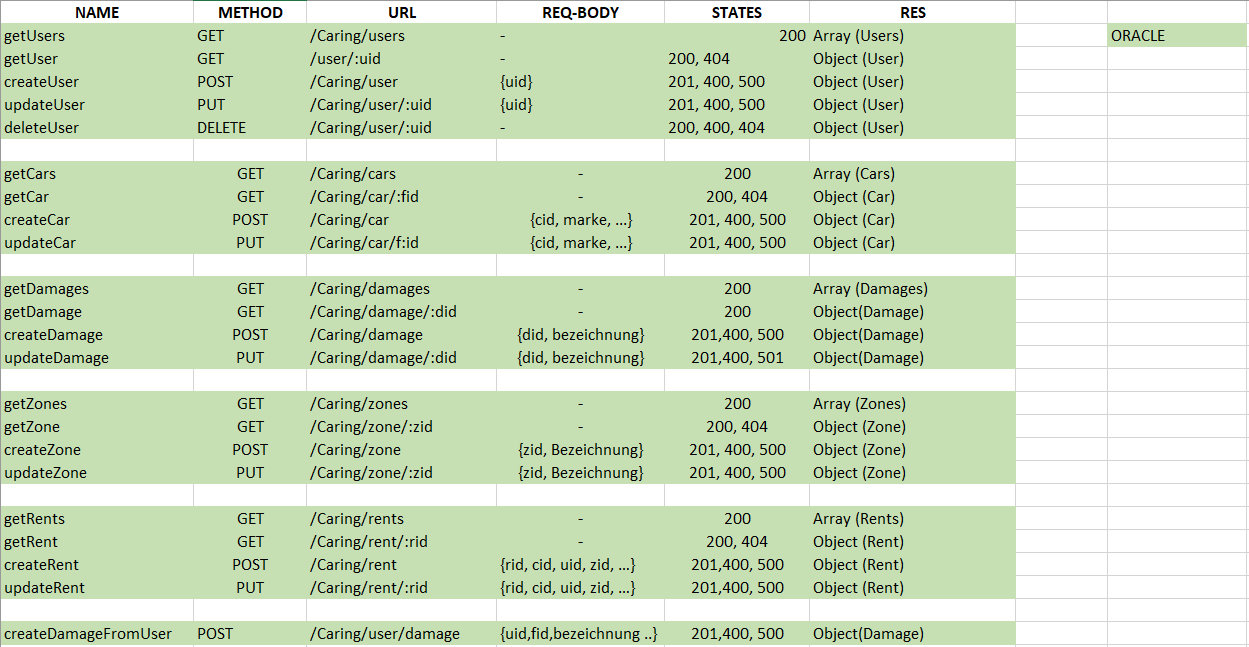


Bild IX – Webservice Schnittstellen

## SQL-Statements

**DROP** **TABLE** fahrzeug **cascade** **constraints**;

**DROP** **TABLE** **zone** **cascade** **constraints**;

**DROP** **TABLE** schaden **cascade** **constraints**;

**DROP** **TABLE** leiht\_aus **cascade** **constraints**;

**DROP** **TABLE** users **cascade** **constraints**;

**DROP** **TABLE** meldetSchaden **cascade** **constraints**;

**CREATE** **TABLE** fahrzeug(

fid int **primary** **key**,

bezeichnung VARCHAR2(100),

marke VARCHAR2(100),

laufleistung INT,

geo\_point SDO\_GEOMETRY,

ausgeliehen VARCHAR2(15)

);

**CREATE** **TABLE** users(

u\_id VARCHAR2(30),

**PRIMARY** **KEY**(u\_id)

);

**CREATE** **TABLE** **zone**(

zid int **primary** **key** ,

bezeichnung VARCHAR2(100),

kostensatz int,

radius SDO\_GEOMETRY

);

**CREATE** **TABLE** schaden(

sid int **primary** **key**,

bezeichnung VARCHAR2(100),

beschreibung VARCHAR2(200),

datum **Date**

);

**CREATE** **TABLE** leiht\_aus (

lid int **PRIMARY** **KEY**,

la\_fid int,

la\_uid VARCHAR2(30),

la\_zid int,

von **TIMESTAMP**,

bis **TIMESTAMP**,

**FOREIGN** **KEY**(la\_fid) REFERENCES fahrzeug(fid) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY**(la\_uid) REFERENCES users(u\_id) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY**(la\_zid) REFERENCES **zone**(zid) **ON** **DELETE** **CASCADE**

);

**Create** **Table** meldetSchaden (

ms\_uid VARCHAR2(30),

ms\_sid int,

ms\_fid int,

**PRIMARY** **KEY**(ms\_uid,ms\_sid,ms\_fid),

**FOREIGN** **KEY**(ms\_uid) REFERENCES users(u\_id) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY**(ms\_sid) REFERENCES schaden(sid) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY**(ms\_fid) REFERENCES fahrzeug(fid) **ON** **DELETE** **CASCADE**

);

**insert** **into** users values('DkTycfHAhYfYAgzOkUXiopMqcfn1');

**insert** **into** users values('EcAUWSlaZChBpK7DuzoKKE3Pq8f1');

**insert** **into** users values('JM8MyOjSTZVL6xI3U5Xj4IrXRj82');

**insert** **into** users values('j6kUD6XYc5WO9p92YcvtDGnh9o43');

**insert** **into** users values('vfq4pAEVRPQQSk7JfxVG4WcudUJ3');

**insert** **into** fahrzeug values(1,'A5 TFSI','AUDI', 106243, SDO\_GEOMETRY(2001, **NULL**, SDO\_POINT\_TYPE(12, 14, **NULL**),**NULL**,**NULL**),'ja');

**insert** **into** fahrzeug values(2,'A6 TDI','AUDI', 56712, SDO\_GEOMETRY(2001, **NULL**, SDO\_POINT\_TYPE(12, 14, **NULL**),**NULL**,**NULL**),'ja');

**insert** **into** fahrzeug values(3,'A4 TDI','AUDI', 123, SDO\_GEOMETRY(2001, **NULL**, SDO\_POINT\_TYPE(12, 14, **NULL**),**NULL**,**NULL**),'ja');

**insert** **into** fahrzeug values(4,'A3 TDI','AUDI', 8832, SDO\_GEOMETRY(2001, **NULL**, SDO\_POINT\_TYPE(12, 14, **NULL**),**NULL**,**NULL**),'ja');

**insert** **into** **zone** values(1,'Villach LIND',100, SDO\_GEOMETRY(2003, **NULL**, **NULL**, SDO\_ELEM\_INFO\_ARRAY(1,1003,4), SDO\_ORDINATE\_ARRAY(8,7, 10,9, 8,11)));

**insert** **into** **zone** values(2,'Untere Fellach',10, SDO\_GEOMETRY(2003, **NULL**, **NULL**, SDO\_ELEM\_INFO\_ARRAY(1,1003,4), SDO\_ORDINATE\_ARRAY(8,7, 10,9, 8,11)));

**insert** **into** **zone** values(3,'Velden',10, SDO\_GEOMETRY(2003, **NULL**, **NULL**, SDO\_ELEM\_INFO\_ARRAY(1,1003,4), SDO\_ORDINATE\_ARRAY(8,7, 10,9, 8,11)));

**insert** **into** **zone** values(5,'Kötschach',1, SDO\_GEOMETRY(2003, **NULL**, **NULL**, SDO\_ELEM\_INFO\_ARRAY(1,1003,4), SDO\_ORDINATE\_ARRAY(8,7, 10,9, 8,11)));

**insert** **into** **zone** values(4,'Mauthen',1, SDO\_GEOMETRY(2003, **NULL**, **NULL**, SDO\_ELEM\_INFO\_ARRAY(1,1003,4), SDO\_ORDINATE\_ARRAY(8,7, 10,9, 8,11)));

**insert** **into** schaden values(1,'Bordstein','test',TO\_DATE('17/12/2015', 'DD/MM/YYYY'));

**insert** **into** schaden values(2,'Fussgeher','test',TO\_DATE('17/12/2015', 'DD/MM/YYYY'));

**insert** **into** schaden values(3,'Motorschaden','test',TO\_DATE('17/12/2015', 'DD/MM/YYYY'));

**insert** **into** schaden values(4,'Steinschlag','test',TO\_DATE('17/12/2015', 'DD/MM/YYYY'));

**commit**;

Code Snippet 1 – SQL Statements

## Trigger

create or replace TRIGGER TRIGGER1

BEFORE UPDATE ON LEIHT\_AUS

FOR EACH ROW

BEGIN

Update fahrzeug Set ausgeliehen='nein' WHERE fid= :NEW.la\_fid;

END;

Code Snippet 2 – Trigger 1

create or replace TRIGGER TRIGGER2

BEFORE INSERT ON LEIHT\_AUS

FOR EACH ROW

DECLARE

isausgeliehen fahrzeug.ausgeliehen%TYPE := NULL;

BEGIN

select ausgeliehen into isausgeliehen from fahrzeug Where fid = :NEW.la\_fid;

IF isausgeliehen like 'ja' THEN

RAISE\_APPLICATION\_ERROR(-20000, 'Auto ist bereits ausgeliehen');

End if;

IF isausgeliehen like 'nein' THEN

Update fahrzeug Set ausgeliehen='ja' WHERE fid= :NEW.la\_fid;

END IF;

END;

Code Snippet 3 – Trigger 2

## Sequenzdiagram

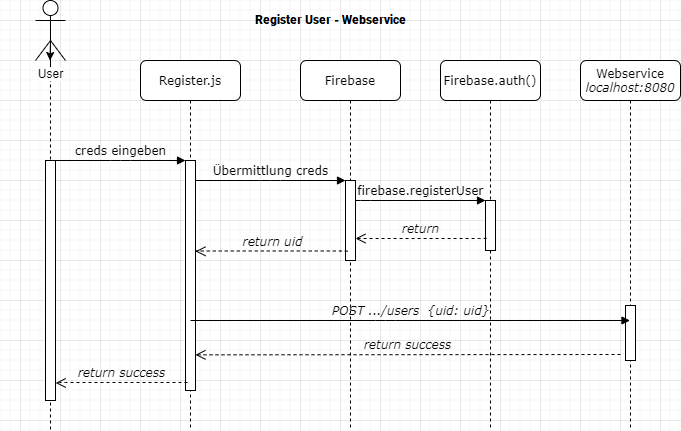


Bild - Sequencdiagramm

Das Sequenzdiagramm (Bild 4) zeigt den funktionellen Ablauf der User-Story „User Registrieren“.

# **Planung des Prozesses**

Vorgabe durch die PO (Lehrer) war es mit Scrum zu arbeiten.

## Sprintplanung (Iteration 3)

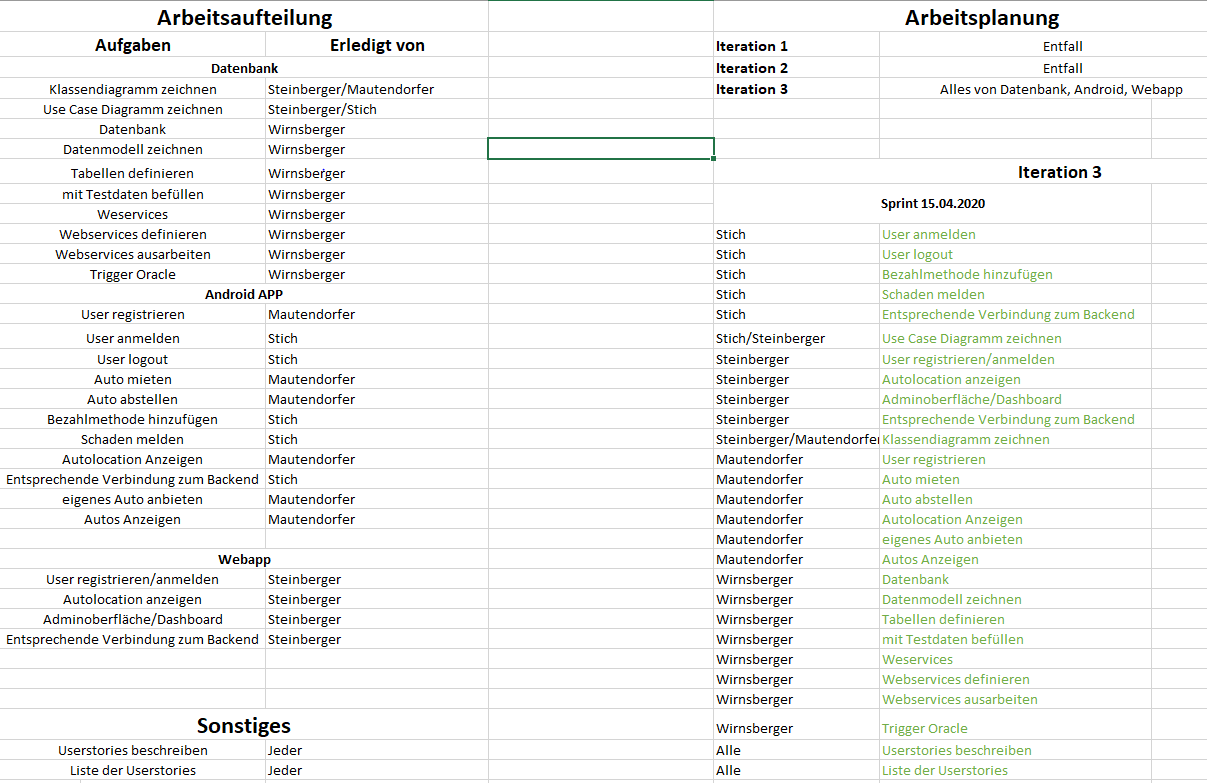


Bild XI - Sprintplanung

Die Im Excel-Sheet grün markierten User-Stories wurden vollständig implementiert sowie getestet. Aufgrund von Konflikten ist der PBL gleich der Sprintplan (siehe Konfliktmanagement).

Bild 14 zeigt den Arbeitsaufwand pro User-Story (geschätzt)

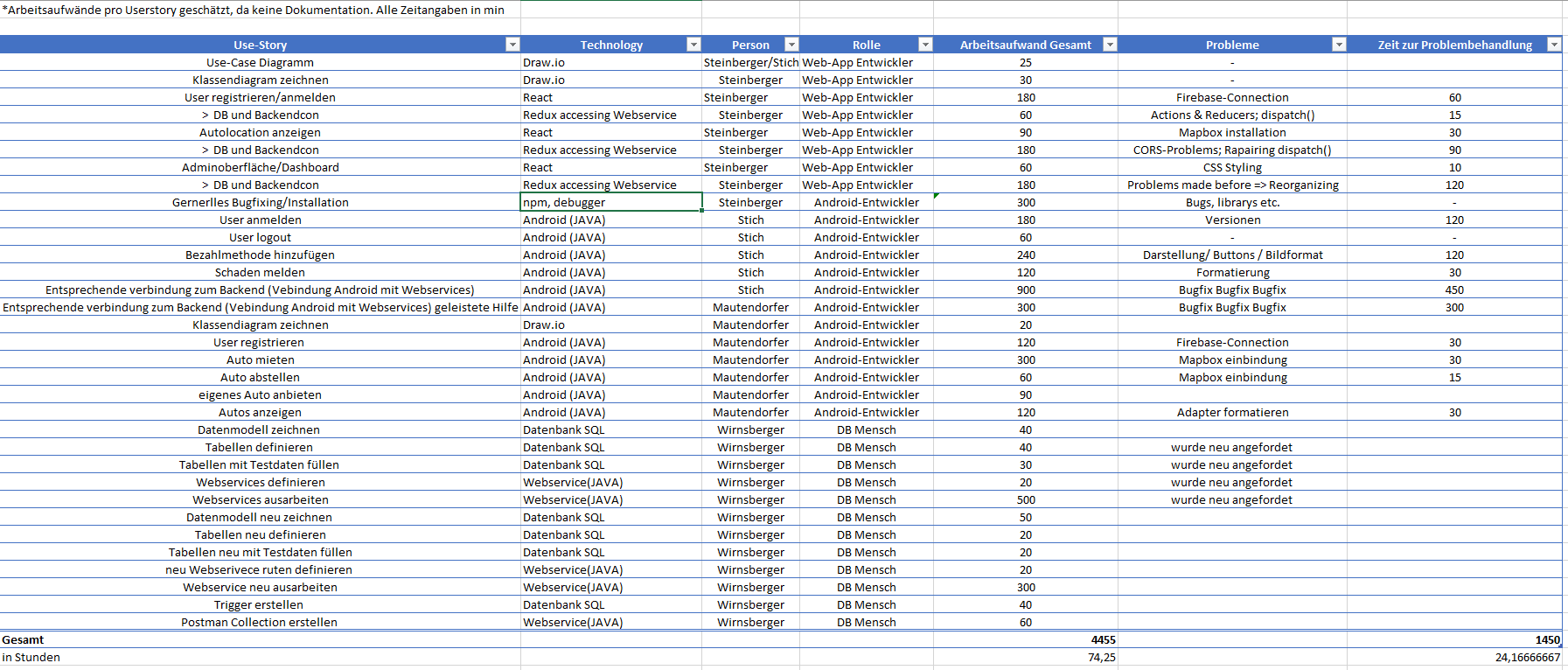


Bild - Arbeitsaufteilung

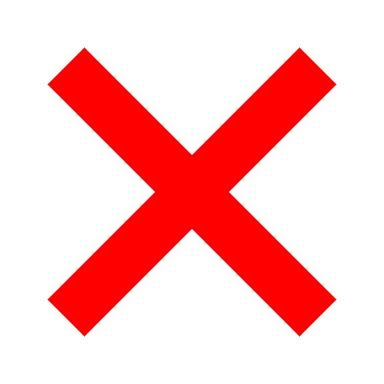
Graph - Arbeitsaufteilung

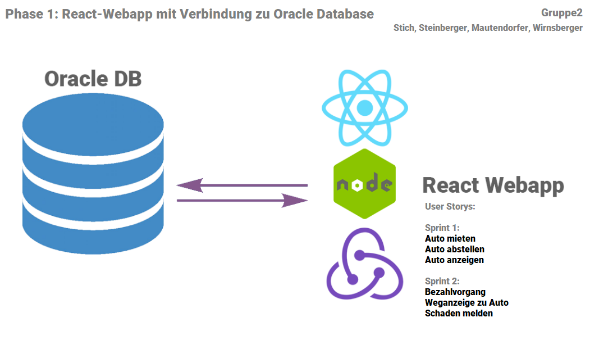
# **RESÜMEE**

Hier werden wir einen Soll-Ist-Vergleich anstellen und Fehler und Probleme klären.

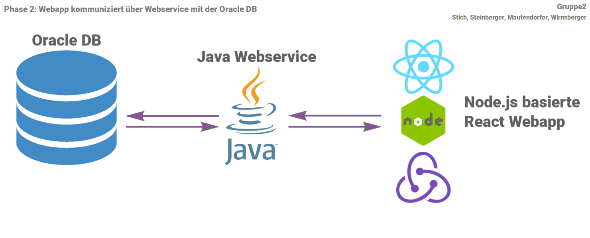
## Konfliktmanagement

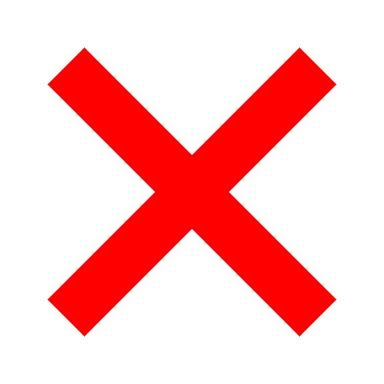
Aufgrund unzureichender Planung und Faulheit des Teams, hat es bei der Realisierung Schwierigkeiten gegeben. Die ersten zwei Iteration entfallen, da in dieser Zeit keine tatsächliche Lösung entstanden sind.

**Iteration 1:**



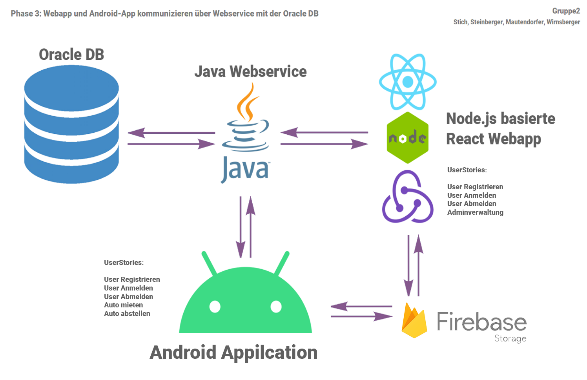
Alle User-Stories der Webapp entfallen im obrigen Bild. Die Datenbank wurde angefangen zu modellieren und implementieren.

**Iteration 2:**



Das im Bild oben erkennbare Webservice und die React-App konnte in Iteration nicht realisiert werden.

**Iteration 3:**



In Iteration konnten alle vom Team geplanten Features odnungsgemäß eingebaut werden.

# **Anhang**

## Bildverzeichniss

[Bild I - Systemarchitektur 6](#_Toc38821597)

[Bild II – Use case diagram 7](#_Toc38821598)

[Bild III – Domainmodell 8](#_Toc38821599)

[Bild IV – Übersicht User-Stories 9](#_Toc38821600)

[Bild V – User Story Map Teil 1 9](#_Toc38821601)

[Bild VI - User Story Map Teil 2 10](#_Toc38821602)

[Bild VII – Activity Diagramm 15](#_Toc38821603)

[Bild VIII – ERD 16](#_Toc38821604)

[Bild IX – Webservice Schnittstellen 16](#_Toc38821605)

[Bild X - Sequencdiagramm 19](#_Toc38821606)

[Bild XI - Sprintplanung 20](#_Toc38821607)

[Bild XII - Arbeitsaufteilung 20](#_Toc38821608)

## Code Snippetverzeichniss

[Code Snippet 1 – SQL Statements 18](#_Toc38821610)

[Code Snippet 2 – Trigger 1 18](#_Toc38821611)

[Code Snippet 3 – Trigger 2 19](#_Toc38821612)

## Graphverzeichniss

[Graph 1 - Arbeitsaufteilung 21](#_Toc38821609)