INF202 : Autohaus Service Projekt

Architektur Design

Türkisch-Deutsche Universität
Dipl.- Ing.Ömer KARACAN

Java Benders

Melih Eyüboğlu (190503020) : e190503020@stud.tau.edu.tr
Berkan Türkoğlu (190501030) : e190501030@stud.tau.edu.tr

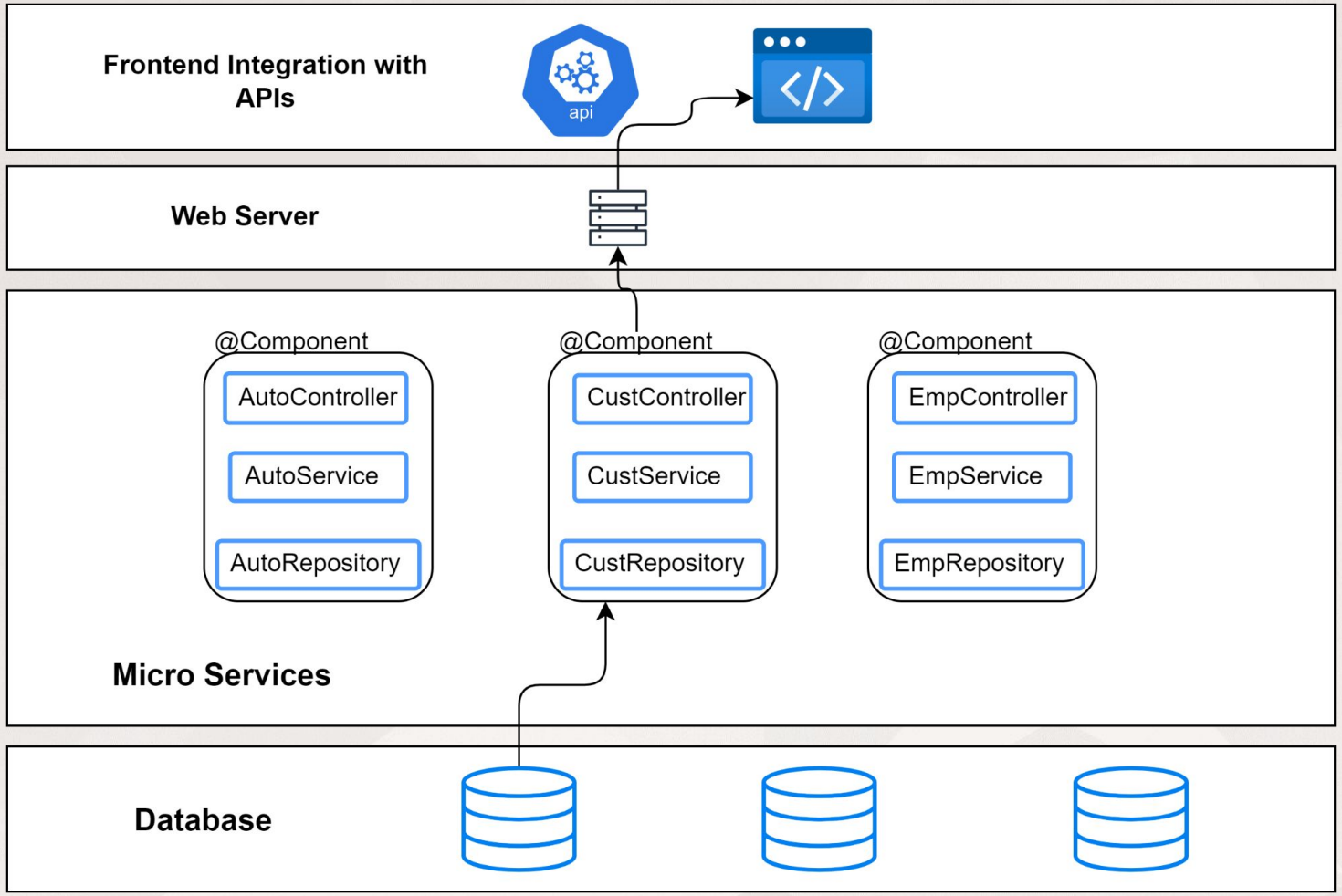
Gliederung

- Architekturüberblick
- Controller Klassen
- Rückverfolgbarkeit der Anforderungen
- Daten-Modelle

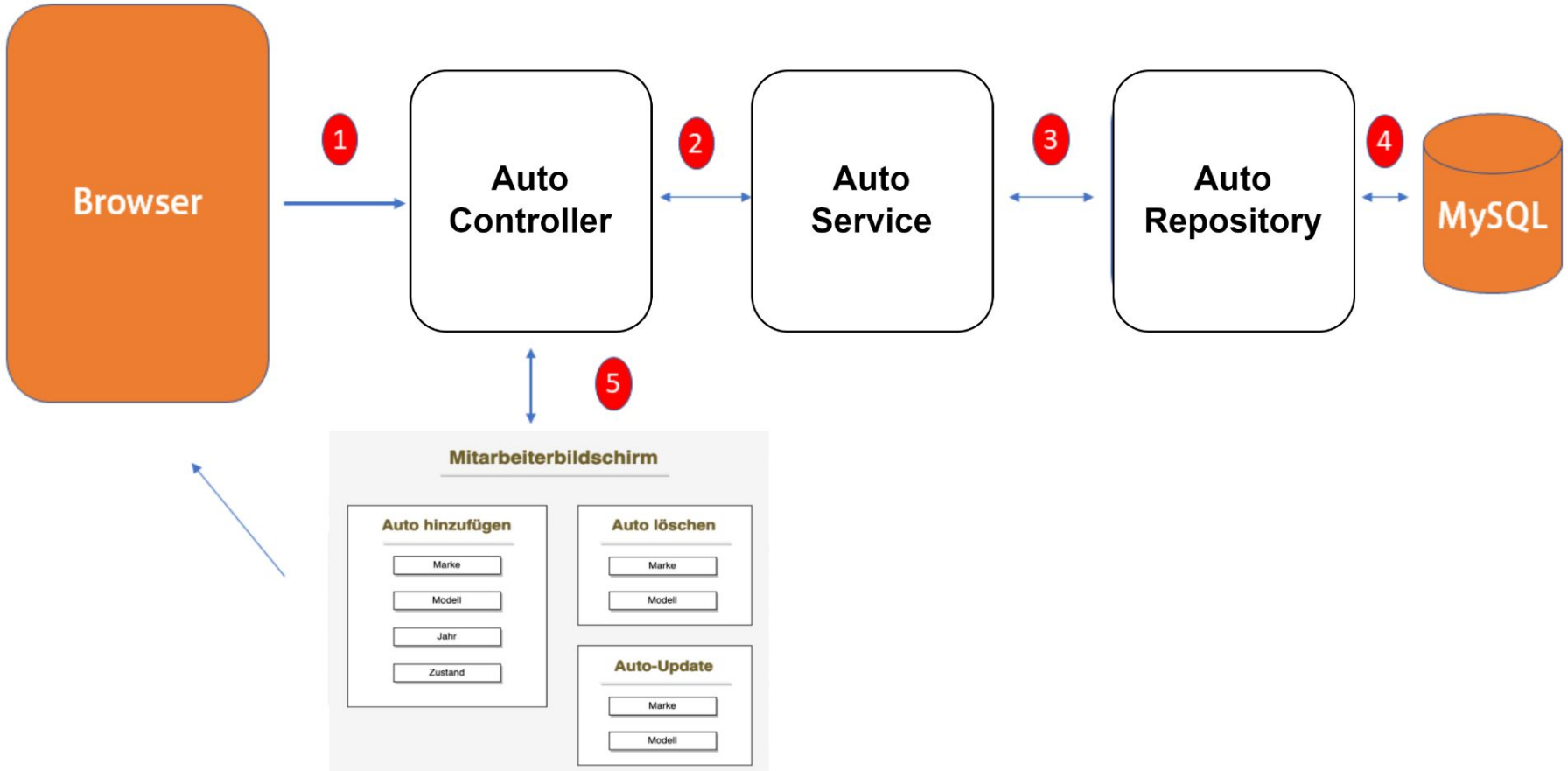
Architekturüberblick

Die allgemeine Architektur der Komponenten in der Anwendung, nämlich das Auto, die Mitarbeiter und die Kundenoperationen in der Datenbank, ist in der Abbildung dargestellt.

Jede Komponente hat einen eigenen Microservice, was das System robuster gegen Fehler macht.



Architekturüberblick



Die Arbeitsarchitektur einer einzelnen Komponente ist in der Abbildung als Diagramm zusammengefasst.

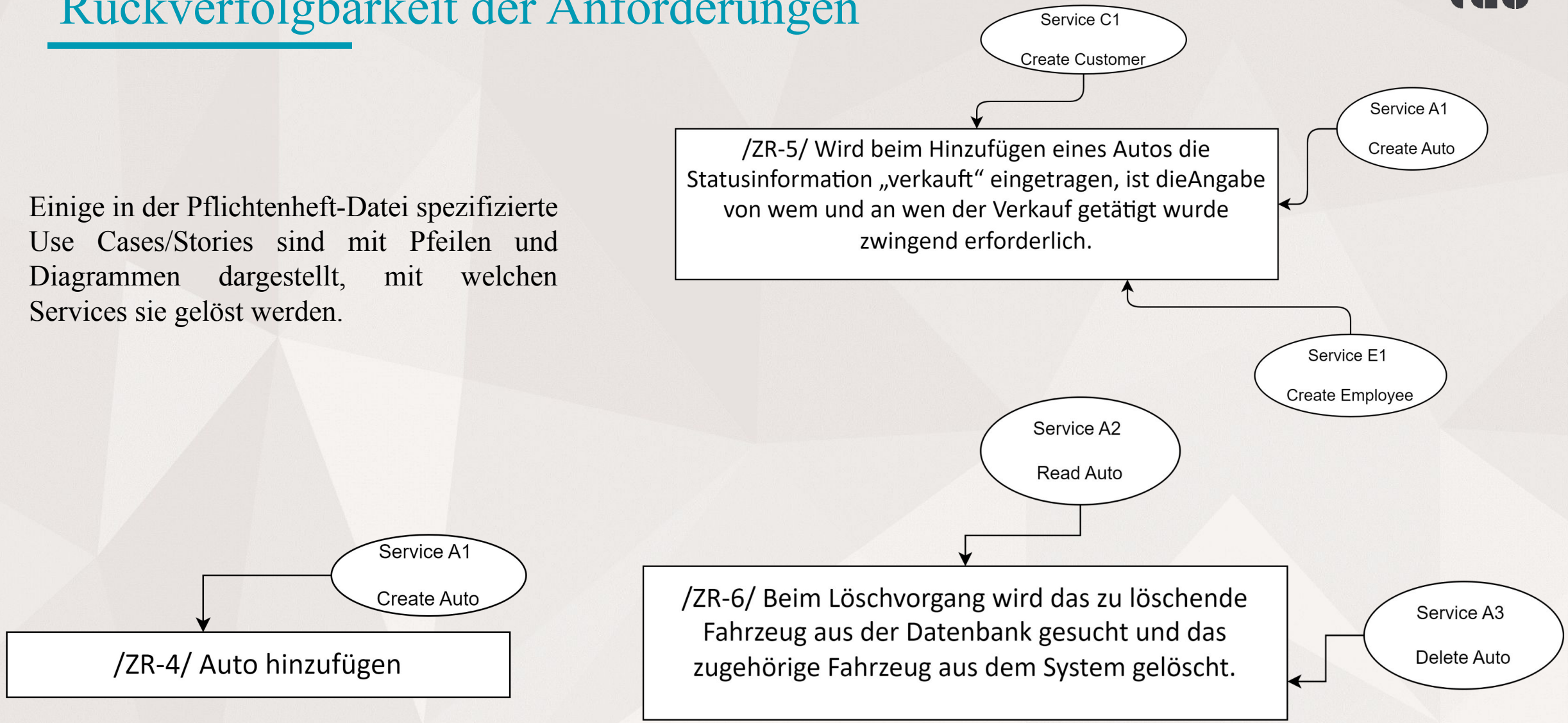
Wenn wir an Anwendungsfälle denken, wird nicht jeder Dienst von Autokomponenten für jeden Benutzer offen sein. Kunde und Mitarbeiter haben unterschiedliche Leistungen.

Controller Klassen

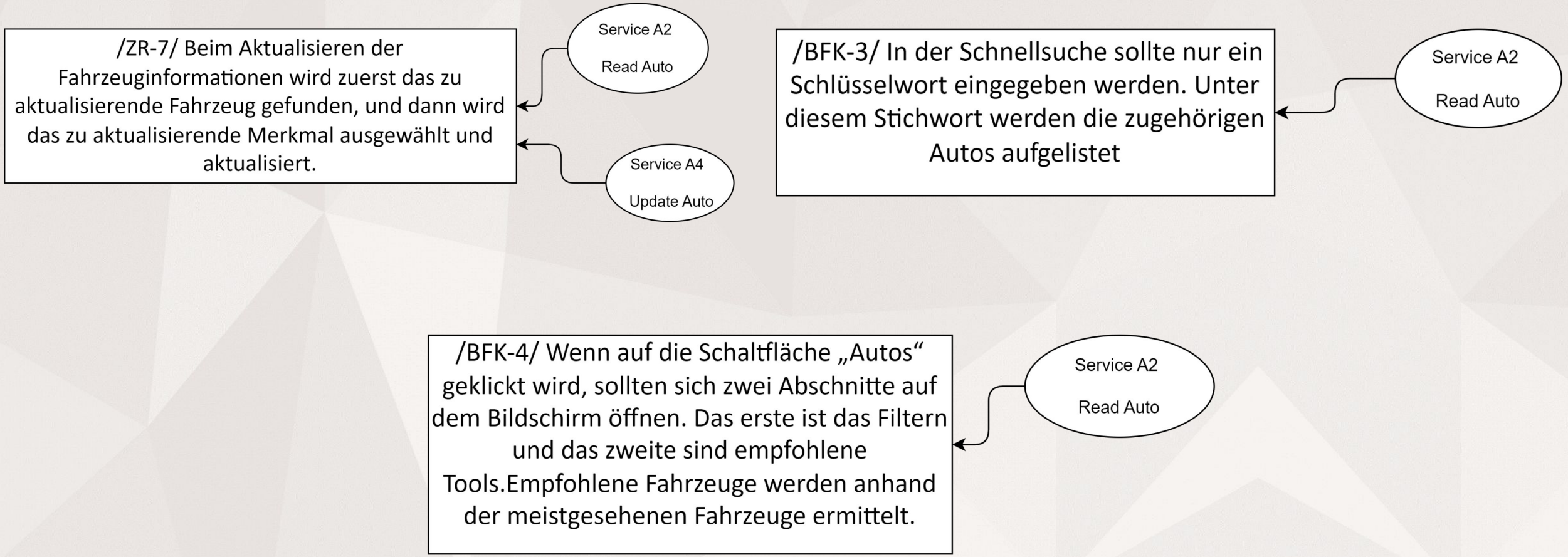
Für jeder Service Klasse gibt es eine Control Klasse, die verbünden ist. Die Control Klasse ruft dann die entsprechenden Methoden der Service Class auf, um die Anfrage zu verarbeiten und eine Antwort zurückzugeben. Die Service Class ist dabei unabhängig von der Control Class und kann in verschiedenen Teilen der Anwendung wiederverwendet werden. Eine Methode könnte in der Control Class mit dem Mapping @GetMapping("/autos/{Marke}") versehen sein. In dieser Methode kann dann die entsprechende Service Class aufgerufen werden, um die Details des Autos abzurufen und als Antwort zurückzugeben. Das Mapping kann auch weitere Optionen enthalten, wie z.B. die Angabe von Request-Parametern, Pfadvariablen oder Anfrage-Headern. Das Mapping gibt also an, welche Parameter von der Control Class verarbeitet werden sollen.

Rückverfolgbarkeit der Anforderungen

Einige in der Pflichtenheft-Datei spezifizierte Use Cases/Stories sind mit Pfeilen und Diagrammen dargestellt, mit welchen Services sie gelöst werden.

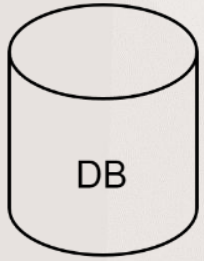


Rückverfolgbarkeit der Anforderungen



Daten-Modelle

Hier ist dargestellt, wie die Daten gespeichert werden.



AUTOS				
Autos	Marke	Modell	Zustand	Jahr
Auto1	Marke1	Modell1	Zustand1	Jahr1
Auto2	Marke2	Modell2	Zustand2	Jahr2
Auto3	Marke3	Modell3	Zustand3	Jahr3

KUNDEN			
Kunden	Name	Adresse	Tel. Nummer
Kunde1	Name1	Adresse1	Tel1
Kunde2	Name2	Adresse2	Tel2
Kunde3	Name3	Adresse3	Tel3

MITARBEITER			
Mitarbeiter	Name	Position	Letzte Verkauf
Mitarb.1	Name1	Pos1	LV1
Mitarb.2	Name2	Pos2	LV2
Mitarb.3	Name3	Pos3	LV3

Daten-Modelle

Hier sind die Pseudocode Beispiele für die Repository-Klassen:

AutosRepository:

```
@Query("SELECT a FROM Autos a WHERE a.Marke = ?1")
List<Autos> findByMarke(String Marke);

@Query("SELECT a FROM Autos a WHERE a.Modell = ?1")
List<Autos> findByModell(String Modell);

@Query("SELECT a FROM Autos a WHERE a.Zustand = ?1")
List<Autos> findByZustand(String Zustand);

@Query("SELECT a FROM Autos a WHERE a.Jahr = ?1")
List<Autos> findByJahr(String Jahr);
```

Daten-Modelle

KundenRepository:

```
@Query("SELECT k FROM Kunden k WHERE k.TelNummer = ?1")
Kunden findByTelNummer(String TelNummer);
```

MitarbeiterRepository:

```
@Query("SELECT m FROM Mitarbeiter m WHERE m.Name = ?1")
Mitarbeiter findByName("String Name");
```

Danke für ihre Aufmerksamkeit