Vehicle Marketplace Dokumentation

1. Einführung

Projektname: Vehicle Marketplace
Kurzbeschreibung:
Eine Webanwendung für den Kauf und Verkauf von Fahrzeugen. Benutzer können sich registrieren, Fahrzeuge einstellen, filtern und durchsuchen, mit Verkäufern kommunizieren und ihre Fahrzeuge verwalten.

2. Funktionalitäten

2.1 Benutzerrollen

- Käufer: Fahrzeuge durchsuchen, Fahrzeugdetails einsehen, Verkäufer kontaktieren
- Verkäufer: Fahrzeuge hinzufügen, bearbeiten und löschen, Verkaufte Fahrzeuge verwalten, Nachrichten von Käufern empfangen und beantworten

2.2 Hauptfunktionen

- Registrierung und Anmeldung von Benutzern
- Fahrzeuge hinzufügen, bearbeiten und löschen
- Filterung und Suche nach Fahrzeugen
- Kommunikation zwischen Käufern und Verkäufern
- Verwaltung des Fahrzeugstatus (verfügbar, verkauft)

3. Systemarchitektur

Frontend: Angular (TypeScript, HTML, CSS)
Backend: Node.js mit Express.js
Datenbank: PostgreSQL
Authentifizierung: JWT (JSON Web Token)

4. Technische Umsetzung

4.1 Backend

Das Backend basiert auf Node.js und Express und stellt verschiedene API-Endpunkte zur
API-Endpunkte:

- **Benutzerverwaltung:**
  - POST /register – Registrierung eines Benutzers
  - POST /login – Benutzer-Login
- **Fahrzeugverwaltung:**
  - GET /api/vehicles/user/:userId – Fahrzeuge eines Nutzers abrufen
  - GET /vehicles – Liste aller Fahrzeuge abrufen
  - GET /vehicles/:id – Ein spezifisches Fahrzeug abrufen
  - POST /vehicles – Ein Fahrzeug hinzufügen
  - PATCH /vehicles/:id – Fahrzeug bearbeiten

    - DELETE /vehicles/:id – Fahrzeug löschen
- **Marken und Modelle:**
    - GET /api/brands – Fahrzeugmarken abrufen
    - GET /api/models – Alle Modelle abrufen
    - GET /api/models/:brand_id – Modelle einer Marke abrufen
- **Nachrichtenverwaltung:**
    - POST /messages – Nachricht senden
    - GET /messages/:userId – Nachrichten eines Benutzers abrufen

4.2 Frontend

Das Frontend wurde mit Angular als SPA (Single Page Application) entwickelt und kommuniziert mit dem Backend über REST-APIs.

4.2.1 Komponenten

- VehicleComponent: Lädt Fahrzeugliste und ermöglicht Filter- und Suchfunktionen
- VehicleDetailsComponent: Zeigt die Detailansicht eines Fahrzeugs und ermöglicht Nachrichten an Verkäufer
- AddVehicleComponent: Ermöglicht das Hinzufügen neuer Fahrzeuge, Bearbeiten und Löschen von Fahrzeugen
- MessageComponent: Zeigt Nachrichten zwischen Käufern und Verkäufern

4.3 Services

- AuthserviceService: Verwaltet Benutzer-Authentifizierung
- VehicleService: Verwaltet Fahrzeugdaten
- VehicleFactoryService: Erstellt spezifische Fahrzeuginstanzen basierend auf VehicleData
- MessageService: Verwaltet Nachrichten zwischen Benutzern
- SanitizationService: Filtert Benutzereingaben, um SQL-Injection zu verhindern

5. Datenbankstruktur

- users: (id, username, password) – Speichert Benutzerdaten
- vehicles: (id, name, price, description, category, brand, model, sellr_id, is_solld, image) – Speichert Fahrzeugdaten
- messages: (id, sender_id, receiver_id, content, created_at) – Speichert Nachrichten
-models: (id, name, brand_id)- Speichert die Modelle
-brands(id, name, category) – Speichert die Marken

6. Sicherheitsmaßnahmen

- SanitizationService: Verhindert SQL-Injections durch Bereinigung von Benutzereingaben.
- JWT-Authentifizierung: Sicheres Login-System mit Token-basiertem Zugriff.
- Role-Based Access Control (RBAC): Verhindert unberechtigten Zugriff auf Ressourcen.

7. Problemstellung

Beim Absenden eines POST-Requests an den Server (`/api/vehicles`) tritt ein `500 Internal Server Error` mit der Fehlermeldung auf:

```
TypeError: Response body object should not be disturbed or locked
    at extractBody (node:internal/deps/undici/undici:5518:17)
```

Untersuchungsschritte:

- Die Datenbankverbindung zu PostgreSQL funktioniert und andere Abfragen laufen problemlos.
- `GET /api/vehicles/:id` funktioniert einwandfrei.
- `console.log()` zeigt, dass die Daten korrekt im Request-Body ankommen.
- SQL-Queries wurden manuell getestet und funktionieren.
  Auch mit Hilfe des Tutors David Jaming konnte das Problem nicht behoben werden.

8.Demonstration

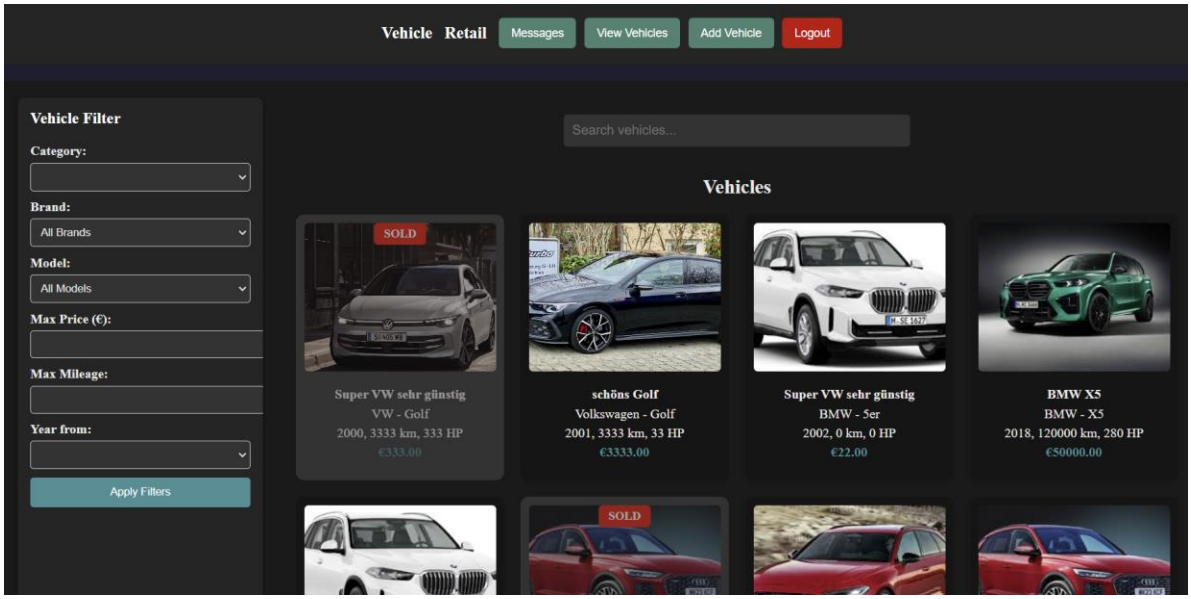

Figure 1 Hauptseite

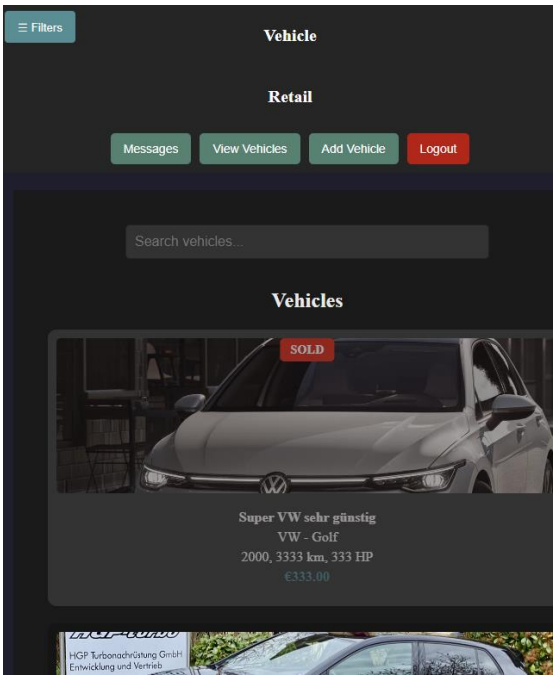

Figure 2 Responsive Hauptseite

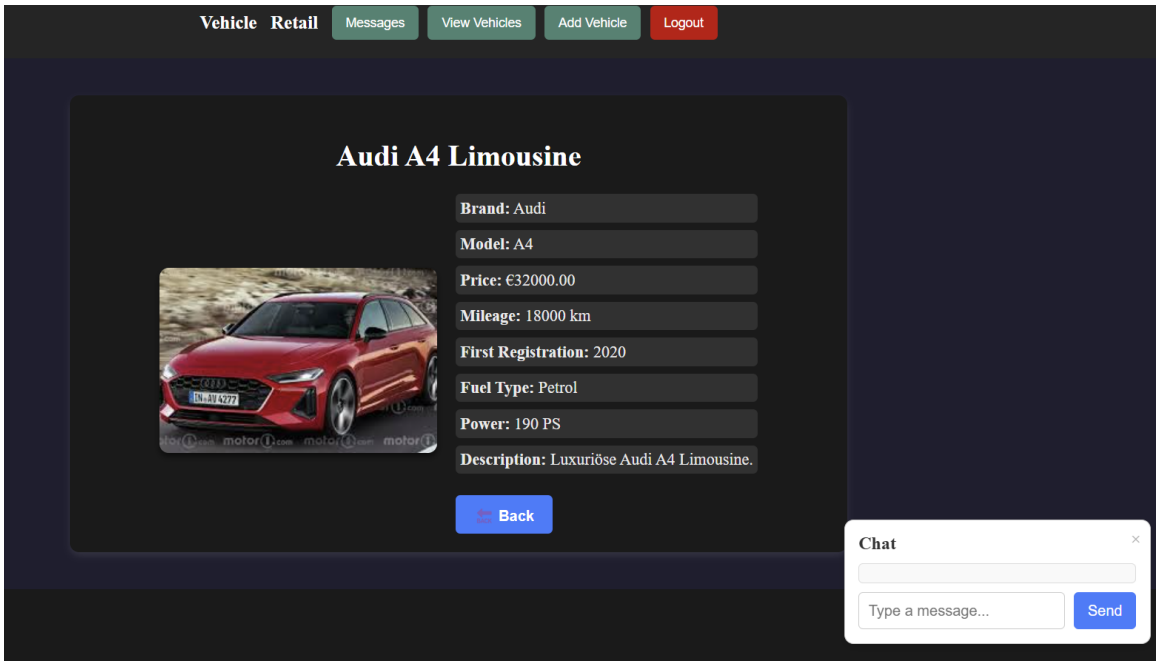

Figure 3 Vehicle Detail

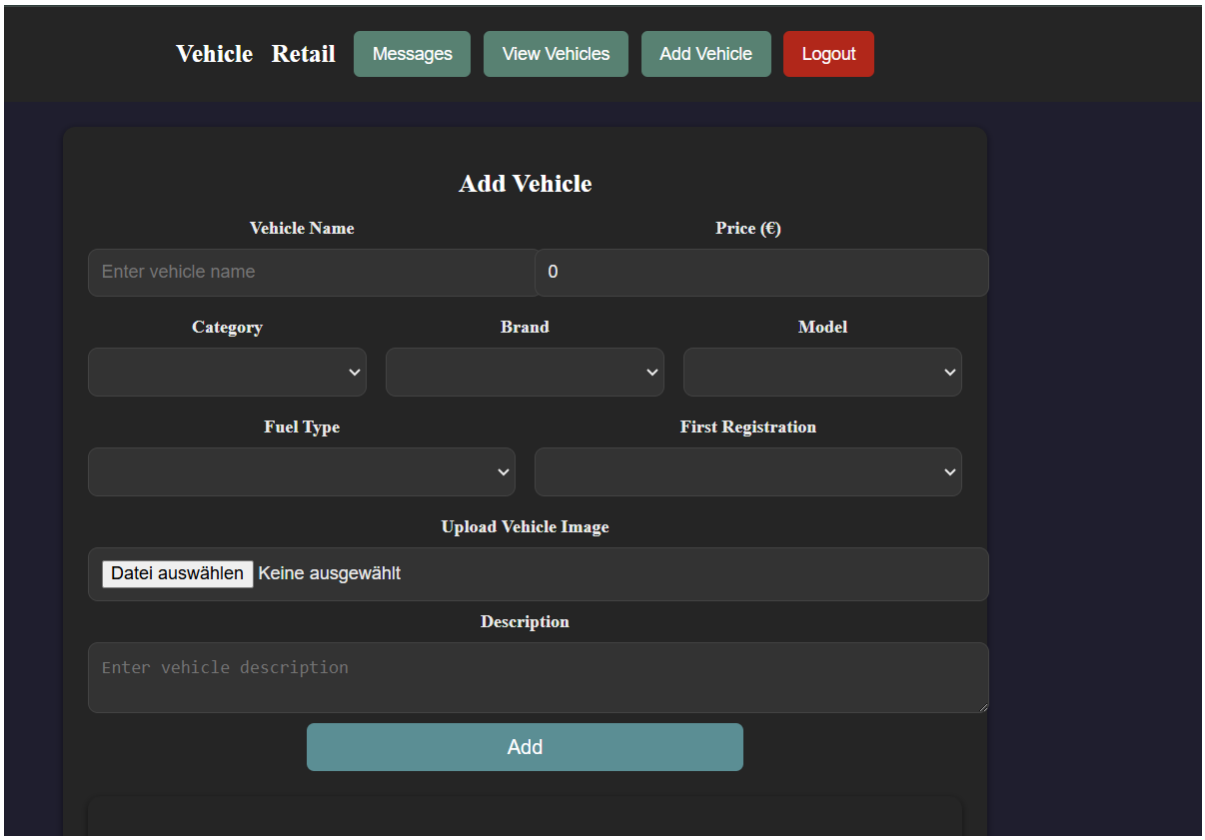

Figure 4 Add Vehicle

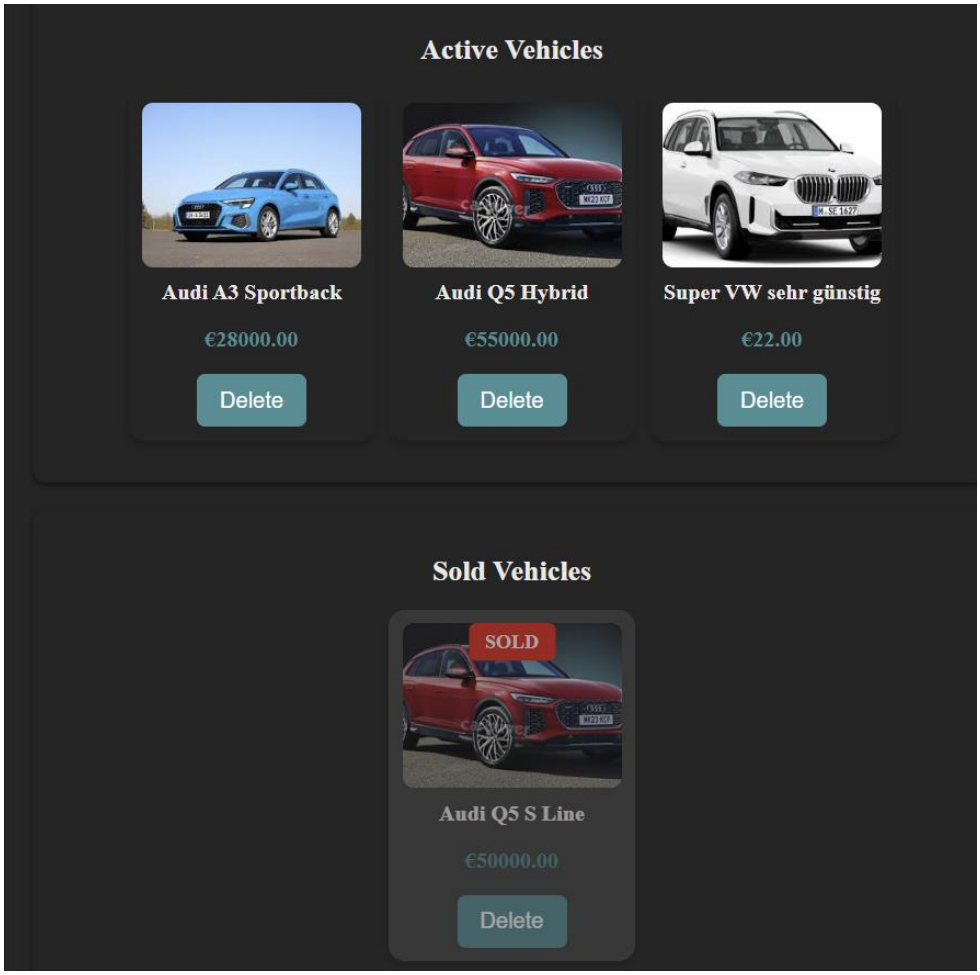

Figure 5 Aktuelle Anzeigen

9. Fazit

Der Vehicle Marketplace bietet eine intuitive Möglichkeit, Fahrzeuge online zu kaufen und zu verkaufen. Die Plattform wurde mit modernen Technologien wie Angular, Node.js und PostgreSQL entwickelt und legt großen Wert auf Sicherheit und Benutzerfreundlichkeit.