DOKUMENTATION (MATRIKELNUMMERN: 3929850, 4557700)

1. Benutzte Technologien

Allgemein wurde das Web-Projekt auf der Beer-API aufgebaut da sich diese als guten Startpunkt nehmen ließ. Deswegen wurde Node als Frontend genommen und Angular da wir darin schon ein wenig Erfahrung hatten.

2. Starten

In Beiden Ordnern npm i

```
PS C:\Users\cpubo\Music\express-beer-api copy\Node> node .\server.js
```

Starten des Backend

```
PS C:\Users\cpubo\Music\express-beer-api copy\Anngu\routing-app> ng serve
```

Starten des Frontend

POSTMAN IST IM ZIP ORDNER

3. AUFGABEN

Aufgabe 1

Über Ihre Schnittstelle können neue Wohnungen, Bauplätze oder Häuser angelegt werden.

Postman: Create Haus
Implementierung: middleware.js Zeile:88

Über Ihre Schnittstelle können alle verfügbaren Objekte (Wohnungen, Bauplätze und Häuser) abgefragt werden, welche das System kennt.

Postman: Get all Häuser

Implementierung: middleware.js Zeile: 22

Über Ihre Schnittstelle kann nach Wohnung, Bauplätzen oder Häusern gefiltert werden.

Postman: Filtern nach Haustyp

Implementierung: middleware.js Zeile: 77

Über Ihre Schnittstelle kann nach einem Objekt über die Adresse gefiltert werden (Sucheingabe)

Postman: Filtern nach Adresse

Implementierung: middleware.js Zeile: 67

Anhand einer eindeutigen Kennzeichnung (ID) kann nach einem einzelnen Objekt gefiltert werden

Postman: Get Haus ID

Implementierung: middleware.js Zeile: 31

Über Ihre Schnittstelle kann ein bestehendes Objekt bearbeitet werden.

Postman: Update Haus

Implementierung: middleware.js Zeile: 137

Über Ihre Schnittstelle kann ein bestehendes Objekt gelöscht werden.

Postman: Delete Haus

Implementierung: middleware.js Zeile: 120

Über Ihre Schnittstelle kann man sich als Interessent an einem Objekt eintragen. Die Anzahl der Interessenten sollten Sie dabei speichern und später ausgeben.

Postman: Interessenten + 1 (Zusatz Interessenten -1 um Interessentenzahl zu mindern)

Implementierung: middleware.js Zeile: 170 (188)

Für ein einzelnes Objekt können Sie sich nur die Anzahl der Interessenten ausgeben

Postman: Get Interessenten

Implementierung: middleware.js Zeile: 56

Für jedes Objekt kann ein Bild hinterlegt werden. Sie können die Bilder lokal auf dem Filesystem speichern. Diese müssen bei einem Neustart nicht gelöscht werden.

Implementierung: middleware.js Zeile: 42

Bieten Sie einen Endpunkt an, welcher für die Übermittlung von externen Daten verwendet werden kann. Das übermittelte JSON-Format ist weiter unten angehängt. Dieses wird im Body von einem externen Softwareclient übermittelt (vgl. Anforderung mit Bauunternehmen)

Postman: Create Haus

Implementierung: middleware.js Zeile:88

Bieten Sie einen Endpunkt an, welcher Testdaten für ihr System erstellt. Dieser Endpunkt soll selbständig Datensätze für ihr System anlegen. (Hintergrund: Damit können Sie Daten erstellen, nachdem Sie Ihren Server z.B. neugestartet habe

Postman: Add Random Häuser (Manuell mit Eingabe der Menge)

http://localhost:8080/addRandomHaus/16 (/menge optional)

Implementierung: middleware.js Zeile: 205, 212

Aufgabe 2
Kommunizieren Sie mit Ihrer Schnittstelle aus Aufgabe 1 über das http Protokoll

D.h. alle Operationen die Sie in der Applikation ausführen können sollten über die Server Schnittstelle passieren und nicht Clientseitig (Ein neues Objekt wird nicht im Client erstellt und gespeichert sondern über Ihre Schnittstelle)

Nach dem starten der Anwendung soll eine Übersicht von allen Objekten (Wohnungen, Häusern und Bauplätzen) angezeigt werden

Implementierung: start.component

Nach verschiedenen Objekten soll gefiltert werden können

Implementierung: start.component

Nach Objekten soll gesucht werden können (Mindestens nach Adresse)

Implementierung: start.component

Neue Objekte können über die Anwendung erstellt werden

Implementierung: start.component → Haus kreieren Button create-haus.component

Bestehende Objekte können in einer Detailansicht bearbeitet werden

Implementierung: start.component → Klicken aufs Bild haus-detail.component

Bestehende Objekte können gelöscht werden

Implementierung: start.component → Klicken aufs Bild haus-detail.component

Auf ein Objekt kann sich ein <Benutzer> als Interessent eintragen

Implementierung: start.component

Die Anwendung verwendet einen Router für die Navigation

Implementierung: app.component

Die Anwendung sollte <Benutzer> über fehlende Eingaben und andere Fehler informieren (z.B. API konnte einen bestimmten Datensatz nicht finden)

Implementierung: Alerts überall

Aufgabe 3
Erweitern Sie Ihre bestehenden Lösungen aus Aufgabe 1 und 2 um die folgenden Funktionalitäten:

Neue Objekte sollen automatisch auf der Übersichtsseite angezeigt werden. Überlegen Sie sich hierfür einen Mechanismus mit den vorgestellten Konzepten aus der Vorlesung

- Ich habe den WebSocket-Server im Backend implementiert, um eingehende WebSocket-Verbindungen zu verwalten.
- Ich habe den Client-Code im Frontend implementiert, um eine WebSocket-Verbindung mit dem Server herzustellen und auf eingehende Nachrichten zu reagieren.
- Ich habe eine Event-Listener-Methode im Frontend implementiert, die auf eingehende WebSocket-Nachrichten reagiert und die neue Immobilie zur Liste der vorhandenen Immobilien hinzufügt.
- Ich habe die API-Route zum Erstellen von Immobilien im Backend aktualisiert, um eine WebSocket-Nachricht an alle verbundenen Clients zu senden, wenn eine neue Immobilie erstellt wird.

→ Es hat aber nicht geklappt

Objekte mit mindestens drei Interessenten können nicht mehr gelöscht oder bearbeitet werden. Dies gilt für die Schnittstelle und Single Page Anwendung

Implementierung: middleware.js Zeile 51,130,159