REST-Spezifikation zu "Interne Mitfahrgelegenheit"

Moritz Basel

23. Januar 2018

1 Einführung

1.1 Zweck des Dokuments

Dieses Dokument dient als Handbuch für Frontenddeveloper zur Software Ïnterne Mitfahrgelegenheit.̈ Es wird versucht, die REST-API vollständig aufzuzählen und alle möglichen Anwendungsfälle zu erfassen. Dennoch besteht die Möglichkeit, dass einzelne Spezialfälle (noch) nicht abgedeckt werden. Bei Auffinden undokumentierten Verhaltens, bei Featureanfragen und Sonstigen bitten wir um Rückmeldung unter <incomplete>.

1.2 Terminologie

Endpoints bezeichnen eine spezifische (z.b. `/api/check_api` oder gruppierte (z.B. `/api/users/<uid>`) URIs

1.3 Allgemeines

Alle Endpunkte lesen und schreiben ausschließlich JSON. Der Content-Type in jedem Http-Request-Header muss als 'application/json' vorliegen.
Für Endpunkte, bei denen Autorisierung nötig ist, wird diese erbracht, indem der Http-Header 'Authorization' gesetzt ist und den Access-Token der von `/api/auth` erhalten wurde, in folgender Form enthält
'Authorization' : 'Bearer ey.....', z.B.:
'Authorization' : 'Bearer eyJhbGciOiJI.3u4zudusua.asudfuu23'

Inhaltsverzeichnis

2 JSON-Objekte

2.1 User

Die JSON-Repräsentation eines Users enthält folgende Felder:

1. username
2. email
3. phoneNumber
4. globalAdminStatus
5. id

2.2 Appointment

Die JSON-Repräsentation eines Appointments enthält folgende Felder:

1. id
2. startLocation
3. startTime
4. repeatTime
5. distance
6. status
7. startTimeTimestamp

Zur Semantik des Felds "repeatTime":
Folgende Werte sind erlaubt

"none"	Das Appointment wird nicht wiederholt
"daily"	Am nächsten Tag findet das Appointment erneut statt
"weekly"	Nach einer Woche findet das Appointment erneut statt

3 User-Management

3.1 Überblick

POST /api/users	Erstellt einen neuen Benutzer
GET /api/users/<UID>	User-Profil
GET /api/users/<Username>	User-ID
POST /api/auth	Login-Token Erstellung
PUT, PATCH /api/users/<UID>	User-Update
GET /api/users/<UID>/distance	Kilometerzähler
GET /api/users/<UID>/appointments	Appointments des Users
GET /api/users	Alle User

3.2 POST /api/users

Erwartet eine JSON-Repräsentation eines Benutzers und Zusätzlich das 'password'-Feld und erstellt diesen nach Validierung der Daten. HTTP-Status 201 bei Erfolg.

3.3 PUT,PATCH /api/users/<UID>

Erwartet eine JSON-Repräsentation eines Benutzers, sowie optional ein neues Passwort im Feld 'password' und überschreibt die alten Nutzerdaten.

3.4 GET /api/users/<UID>

Agiert wie erwartet - gibt die JSON-Repräsentation des Nutzers zurück. Zugriffsrechte hat jeder Nutzer.

3.5 GET /api/users/<Username>

Gibt lediglich die User-ID in der Form 'id' : 0 zurück. Diese kann dann weiterverwendet werden für spätere API-Zugriffe. Dies ist (im Moment 0.2.0) noch offen für anonyme Zugriffe.

3.6 POST /api/auth

Gibt den Login-Token im JSON-Eintrag 'access_token' zurück. Dieser muss bei Zugriffen auf geschützte http-Endpunkte im Header-Feld 'Authorization' in allen Requests mitgegeben werden.
Außerdem werden alle Nutzerdaten in den Feldern 'username', 'email', 'phoneNumber', 'globalAdminStatus' versendet. Dies wird nur zur Unterstützung getan, da im JWT Body alle Felder in Base64 vorhanden sind.

3.7 GET /api/users/<UID>/distance

Gibt die tatsächlich als Fahrer gefahrene Distanz eines Users zurück.

3.8 GET /api/users/<UID>/appointments

Gibt die Appointments des Nutzers zurück. (a.k.a. alle Appointments, an denen der User teilnimmt)

3.9 GET /api/users

Gibt die Alle Nutzerdaten in einer Liste zurück. Arbeitet mit vollen User-JSON-Repräsentationen.

4 Appointment-Funktionalitäten

4.1 Überblick

GET /api/appointments/<appointmentID>	Daten zum gewähltem Appointment
GET /api/users/<userID>/appointments	Liste der Appointments zum Nutzer
GET /api/appointments	Liste aller Appointments
POST /api/appointments	Erstellt ein neues Appointment
GET /api/appointments/<appointmentID>/users	Liste alle teilnehmenden user

4.2 GET /api/appointments/<appointmentID>

Gibt eine JSON-Repräsentation des Appointments zurück.

4.3 GET /api/users/<userID>/appointments

Gibt die Appointments eines Users in einer Liste zurück. Nutzt die volle JSON-Represäntation von Appointments.
Es wird ein GET-URL-Parameter unterstützt:
finished=true/false (default: false).
Bei finished=true werden auch Appointments im Status APPOINTMENT_RETIRED zurückgegeben.

4.4 GET /api/appointments

Gibt die volle Liste aller Appointments in einer Liste zurück. Nutzt die volle JSON-Represäntation von Appointments.

4.5 POST /api/appointments

Erstellt ein neues Appointment. Verlangt eine vollständige JSON-Represäntation im Request-Body.

4.6 GET /api/appointments/<appointmentID>/users

Liste aller am Appointment teilnehmenden User. Dieser Endpunkt verhält sich besonders und gibt Daten in besonderer Form zurück. Für weitere Details, die Datei zum Appointment Lifecycle konsultieren.

5 Entwicklungs-API

5.1 Überblick

DELETE /api/dev/removeUser/<uname>	Löscht bestimmten Nutzer
GET /api/dev/check_token	User-ID des Nutzers
GET /api/dev/check-api	Simpler Test
GET /api/dev/log	Log

5.2 DELETE /api/dev/removeUser/<username>

Löscht den gewählten Nutzer vollständig. Kann nur vom Nutzer selbst und von globalen Administratoren durchgeführt werden.