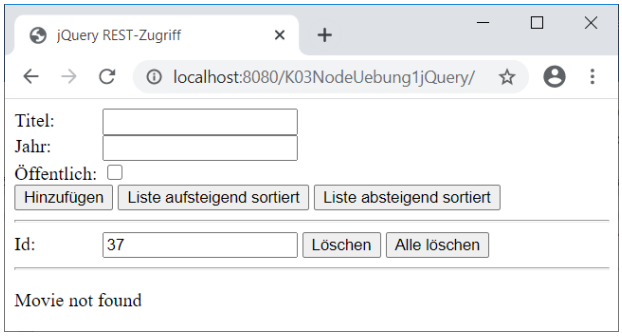
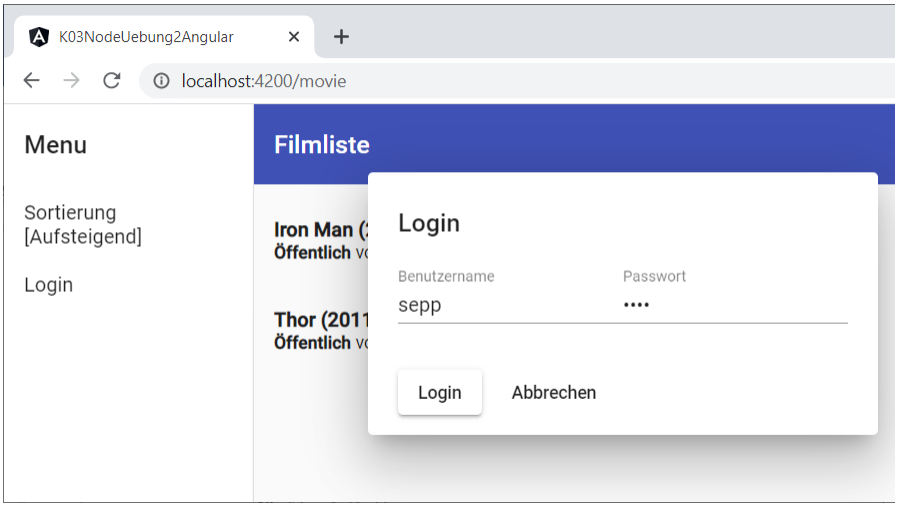
1. Bringen Sie das vorgestellte Beispiel in welchem über *jQuery* auf den Web-Service der Filmverwaltung zugegriffen wird zum Laufen.

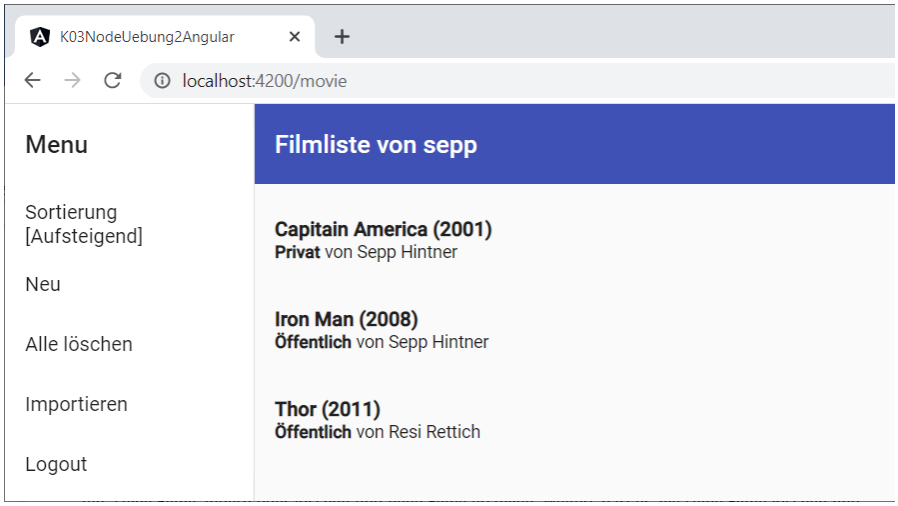


Erstellen Sie dazu zuerst den Web-Service laut Ausführungen im Unterricht. Dabei soll die Authentifizierung des Benutzers noch nicht realisiert werden. Gehen Sie vielmehr davon aus, dass immer nur derselbe Benutzer mit dem Web-Service arbeitet. Dabei wird der Benutzer nicht vom Client an den Server geschickt, sondern der Web-Service geht davon aus, dass immer nur dieser eine Benutzer zugreift. Entsprechend werden nur die Daten dieses Benutzers geliefert bzw. Operationen dieses Benutzers zugelassen.

Erweitern Sie dann das jQuery-Web-Projekt um die Möglichkeit einen *Film über seine Id* und *alle Filme zu löschen*.

2. Erstellen Sie eine Web-Anwendung in *Angular* zur Verwaltung der Filme. Diese muss sowohl für nicht authentifizierte Benutzer als auch für jene die sich angemeldet haben funktionieren. Ein nicht angemeldeter Benutzer darf nur die veröffentlichten Filme in auf- oder absteigender Reihenfolge betrachten. Ein angemeldeter Benutzer darf seine und alle öffentlichen Filme in auf- oder absteigender Reihenfolge betrachten, seine Filme ändern oder löschen und neue Filme erstellen. Weiters darf er alle seine Filme löschen und Filme importieren.





Filme sollen in einer List-Komponente mehrzeilig ausgegeben werden. Die Sortierreihenfolge muss über das Menü eingestellt werden können.

Die Authentifizierung der Benutzer soll über *JSON Web Tokens* erfolgen. Der *„Exkurs: Authentifizierung in Node.js durch JSON Web Tokens (JWT)"* zeigt Ihnen wie Sie einerseits Ihre Angular-Anwendung für den Login-Vorgang anpassen und den Web-Service dazu veranlassen JWTs zu erstellen und zu verwalten.

Verwenden Sie zur Eingabe von Benutzernamen und Passwort den `MatDialog`-Service von Angular Material.

Dazu sollen Sie einen neuen Web-Service – der mit JWTs arbeitet – bereitstellen, der folgender Möglichkeiten bietet (**ACHTUNG:** Den Web-Service den Sie für Aufgabe 1 erstellt haben, sollen Sie nicht antasten damit Aufgabe 1 mit dem alten Web-Service nach wie funktioniert und getestet werden kann):

Methode	Route	Statuscode/Error
GET	/movie /movie?sort=asc /movie?sort=desc	200 liefert Liste der öffentlichen Filme und jene des Benutzers wenn der Benutzer angemeldet ist 400 No movies found, 500 Database error
GET	/movie/:id	200 liefert den gefundenen Film dessen Besitzer der Benutzer ist 400 Movie not found, 500 Database error
POST	/movie	200 liefert den eingetragenen Film mit neuer id und fullname. Dabei muss der owner des Films mit dem Benutzer übereinstimmen 400 User not found, Title exists, 500 Database error
PUT	/movie/:id	200 liefert geänderten Film. Dabei muss der owner des Films mit dem Benutzer übereinstimmen 400 User not found, Title exists, Movie exists, Movie not found, 500 Database error
DELETE	/movie/clear	200 Leerer Body bei Erfolg. Es werden nur die Filme des übergebenen Benutzers gelöscht 400 Movies not found, 500 Database error
DELETE	/movie/:id	200 Leerer Body bei Erfolg. Dabei darf der Film nur gelöscht werden, wenn der owner mit dem Benutzer übereinstimmt 400 Movie not found, 500 Database error
POST	/movie/login	200 liefert JWT mit Benutzername ur, und Benutzer-Id id in Payload, 401 Unauthorized
GET	/movie/published /movie/published?sort=asc /movie/published?sort=desc	200 liefert Liste der öffentlichen Filme, Benutzer ist dabei nicht angemeldet, 400 No movies found, 500 Database error

Der letzte Endpunkt soll dazu dienen nicht angemeldeten Benutzern nur die öffentlichen Filme zu liefern (**ACHTUNG:** Bei den Routen des Web-Services muss obiger GET-Endpunkt vor dem GET-Endpunkt für '/' definiert werden, weil ansonsten bei einem Request auf '/published' zuerst der '/'-Endpunkt kontrolliert wird und dieser eine Authentifizierung verlangt).

Beim Hinzufügen und Ändern von Filmen müssen Titel und Jahr eingegeben werden. Auch muss ein Fehler angezeigt werden, falls der eingegebenen Titel bereits existiert.

Im Ändern-Formular muss zudem die Möglichkeit gegeben werden, den Film zu löschen.

Beim Löschen aller Filme eines Benutzers muss über MatDialog eine Meldung angezeigt werden, welche dem Benutzer entscheiden lässt, ob er den Vorgang fortsetzen möchte.

3. Realisieren Sie dann in Ihrer Web-Anwendung einen *Datei-Upload* zum Importieren von Filmen. Das Dateiformat soll jenes des letzten Kapitels sein. Im *„Exkurs: Angular-Komponente für den Datei-Upload“* wird Ihnen gezeigt, wie Sie einen Upload in Angular realisieren können.

Binden Sie die Upload-Komponente in über `MatDialog` ein. Die Rückmeldungen des Web-Services sollen über eine Angular Material `MatSnackbar`-Komponente ausgegeben werden. Passen Sie den Web-Service entsprechend an.