**Projekt Java API Nationalities**

Am Anfang haben wir eine API über Star Wars Filme und deren Charaktere gewählt, an welcher wir in 2 Lektionen gearbeitet haben. Wir haben unser GitHub Repository erstellt, in welchem wir alle Dateien, welche wir in Visual Studio Code geschrieben haben, speichern. Man kann in VS-Code mit dem Befehl git pull (link zu dem Repository) die Daten auf den lokalen Rechner klonen. Mit dem Befehl git push kann man die Daten auf das GitHub Repository hochladen. Wir haben vor jeder Lektion einen kurzen Input zum Projekt bekommen, mit welchem wir eine kleine Hilfestellung hatten. Wir haben diese Aufgabe zuerst sehr stark unterschätzt. Wir haben zu lange nach einer guten API gesucht und somit hatten wir weniger Zeit in der Schule an dem Code zu arbeiten.

Wir haben uns den Code von Forkify angesehen und unseren zuerst danach aufgebaut. Das Projekt sah anfangs nicht sehr gross aus, jedoch hatten wir sehr viele Probleme und wir haben bemerkt, wie schwierig es war diese vielen Funktionen zum Laufen zu bringen. Besonders schwierig war es die Filme mit ihren Schauspielern in die API einzubinden. Die Star Wars API hat die Daten in linkt gespeichert, welche dann zum nächsten Datensatz verbindet. Dies wurde uns jedoch zu schwierig, da wir in diesem Bereich noch keine Idee hatten, wie wir dies zum Laufen bringen könnten.

Wir haben ein wenig herumprobiert, was uns jedoch nicht viel weitergebracht hatte. Somit habe wir uns beschlossen eine andere API auszuwählen, mit welcher wir die verlangten Funktionen mit unserem Fachwissen erfüllen konnten. Wir haben uns danach dazu entschlossen das Projekt ohne View, Controller und Model zu schreiben, sondern alles in ein Java File hineinzuschreiben. Auch haben wir uns entschieden kein SCSS zu benutzen, sondern normales CSS, weil wir uns damit besser auskennen.

Wir haben bei unserem GitHub Repository den Namen geändert und bei uns in Visual Studio Code das Repository aktualisiert. Danach haben wir uns einen Plan überlegt, wie wir das am besten angehen könnten. Bei der Letzten API war unsere Herangehensweise nicht die beste und somit war dies ein guter Schritt zum Erfolg.

Als erstes haben wir in dem Projekt die Dateien erstellt und den anfangs HTML-Code geschrieben. Wir haben bereits die meisten Lektionen in der Schule für die Vorbereitung und die richtige Auswahl der API verbraucht. Wir hatten also nur noch zwei Wochen in der Schule, in der wir noch daran Programmieren konnten.

Wir haben in der index.html Datei die Skripts eingebaut und die verschiedenen Elemente benannt. Den Platzhalter für die Such-Box haben wir sheet1 genannt. Wir haben in diesen die Suchfunktion eingebaut, mit welcher man die Namen suchen kann. Die Suche setzt sich aus einem Eingabefeld und einem Knopf zusammen. Der Suchknopf ist mit einem Bild ausgestattet. In dem Suchfeld steht „Write your name“.

Im Feld new\_name ist der Text „Neuer Eintrag“ bei welchem man durch den Knopf namens save\_button den neuen Eintrag speichern kann.

Die Herkunft der Namen sollte dann in dem Div Origin angezeigt werden.

Im dem nameslist\_container stehen die Namen, welche durch das Suchen aufgerufen werden. Hier kann man einen Namen als Favoriten speichern oder ihn löschen oder bearbeiten.

Die API [nationalize.io](https://nationalize.io) war nicht die allerbeste API, weil sie die Namen nicht als Array gibt. Somit müsste man jeden Namen selber aufschreiben und zu einem Array umformen. Deswegen haben wir die Namen in eine JSON-Datei geschrieben und danach vom JSON-File zum Array umgewandelt. Wir konnten auch nicht alle der 260'000 Namen verwenden, da sonst die Webseite nicht richtig funktioniert. Wir mussten somit die Liste der Namen auf etwa 1000 reduzieren. Nun hat es viel weniger Namen, jedoch funktioniert die Webseite mit einer kurzen Ladezeit, welche der Browser braucht, um alle Namen einzulesen.

Im CSS-File haben wir den Hintergrund gefärbt und die Ränder des Suchfeldes und der Namen abgerundet und ihnen Farbe gegeben. Auch haben wir einen CSS-Code dafür geschrieben, wenn man den Mauszeiger über den Knopf bewegt er die Farbe ändert. Die neu eingegebenen Namen mussten wir dort auch noch färben.

Im JavaScript-File haben wir zuerst drei const definiert, einen für die Liste der Namen, einen const für die Namenseingabe und einen für die neuen Namen.

Unsere erste Funktion wandelt die Daten der JSON-Datei in einen Array um. Dies wird mit einer asynchronen Funktion gemacht.

Danach werden in der Konsole alle Namen ausgegeben, mit welchem man sehen kann, ob alles funktioniert oder nicht.

Im const loadNames ruft man die Namen ungefiltert auf und werden angezeigt.

Der nächste const zeigt die Namen an und berechnet + 1 womit man zum nächsten Namen kommt. Danach werden die nächsten Namen geladen.

Die Suchfunktion wird durch einen EventListener ausgelöst. Wenn der Benutzer die Taste loslässt wird dies der Suchfunktion weitergegeben und es wird unter den Arrays nach den Namen mit den Buchstaben gesucht. Die Funktion wandelt die Arrays automatisch in kleine Buchstaben um, damit besser gesucht werden kann.

Wenn man auf den Knopf klickt wird die nächste Funktion ausgelöst, da ein EventListener darauf schaut, ob der Knopf gedrückt wurde oder nicht. In dieser Funktion wird ein neues Feld generiert, in welches der neue Name gespeichert wird.

Mit der nächsten Funktion kann man einen Namen löschen, indem man auf den Löschknopf klickt, welcher mit einem EventListener verbunden ist, welcher die Funktion nach dem Klicken ausführt.

Die nächste Funktion ist mit einem EventListener verbunden, welcher nach einem Klick auf den Editierknopf den Namen editieren lässt. Es öffnet sich ein Fenster, in welchem man den Namen ändern kann. Wenn man bestätigt ändert sich der Name und wenn man abbricht bleibt der Name gleich wie vorher.

Man kann mit der nächsten Funktion, bei welcher der EventListener auch durch einen Klick auf den Knopf reagiert, auslösen. Mit dieser Funktion kann man einen Namen als Favorit hinzufügen. Es dient dem Benutzer als Lesezeichen und man kann dadurch schneller auf die Namen zugreifen, die man braucht.

In der nächsten Funktion wird durch einen Klick auf den Knopf mit einem EventListener der angeklickte Name identifiziert.

Mit der nächsten Funktion wird die API aufgerufen. Es wird der Link geöffnet mit dem Namen, welchen man sucht hinter dem Link. Danach werden die Daten wieder ausgegeben.

Mit der Funktion loadOrigin wird die Herkunft der Person geladen und an dem const daten weitergegeben.

Mit der nächsten Funktion werden die Namen mit deren Herkunft ausgegeben und angezeigt.