Hochschule Harz
Hochschule für angewandte Wissenschaften

Dokumentation

Florian Swinareck
Matrikelnummer: 26784

Webprogrammierung

SoSe 2020

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

Meine Idee für die Bearbeitung der Aufgabenstellung kommt von einer bereits existierenden Webseite (myanimelist.net), welche dient eine Anime-Liste zu führen, sich zu informieren über vorhandene Anime, neue Anime, diese zu bewerten und so weiter. Auf dieser Seite gibt es jedoch auch eine Suchmaschine und Filter um nach bestimmte Anime zu suchen oder entsprechende Eigenschaften. Ich entschloss mich daher mich nicht nach einem Shop oder ähnliche Sachen zu orientieren und stattdessen mein Hobby zu nehmen und es als Thema für meine Webseite zu verwenden. Ich habe meine Webseite nicht responsive für mobile Geräte gemacht und daher an meinem eigenen Desktop optimiert (1080p). Dazu habe ich mich nicht wirklich an einem Problem orientiert oder versucht einen besonderen Grund zum Thema ersucht, sondern die Aufgabe diente für mich zum Lernen und besseren verstehen von Webprogrammieren. Bereits im ersten Semester sowohl auch im zweiten Semester haben wir zwei Webseiten erstellt und uns langsam an PHP gewagt. Mir hat damals schon die Arbeit an einer eigenen Webseite gefallen und habe diese Aufgabe daher mit gleicher Absicht gesehen. Die neuen erlernten Gebiete, wie Ajax oder SQL, sind daher relativ Neu und waren ziemlich ansprechend. Die Dokumentation soll meinen relativen Arbeitsablauf erklären und verdeutlichen.

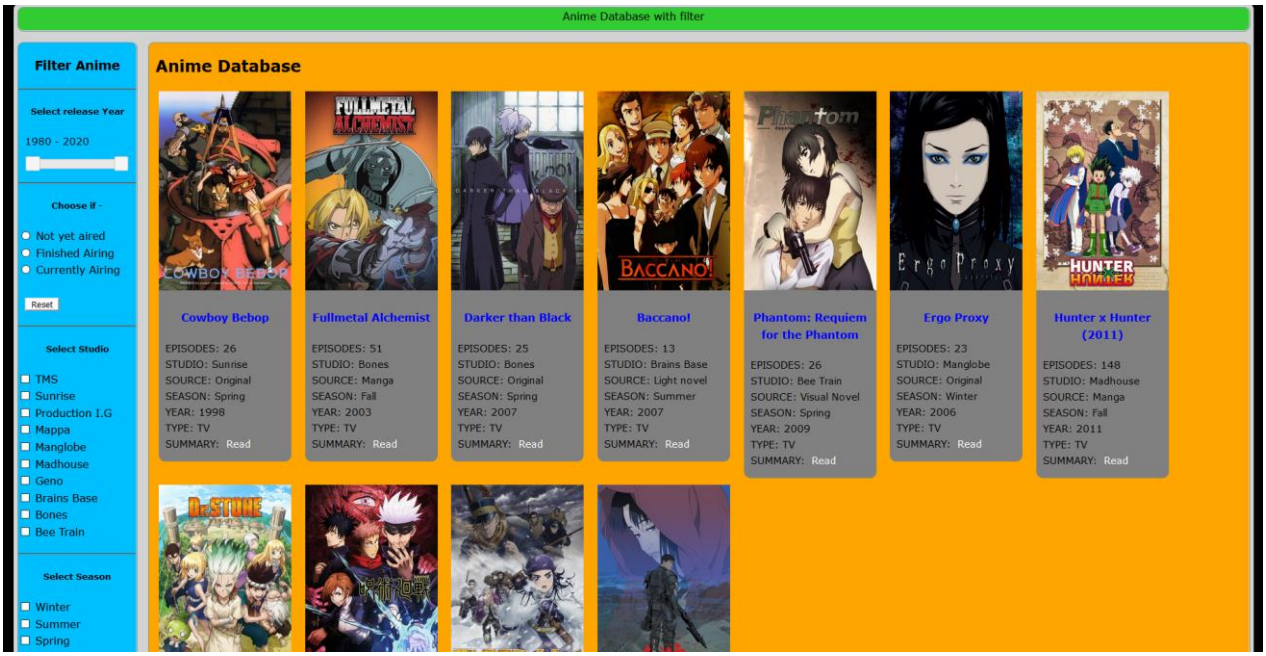

Abbildung 1 fertige Webseite

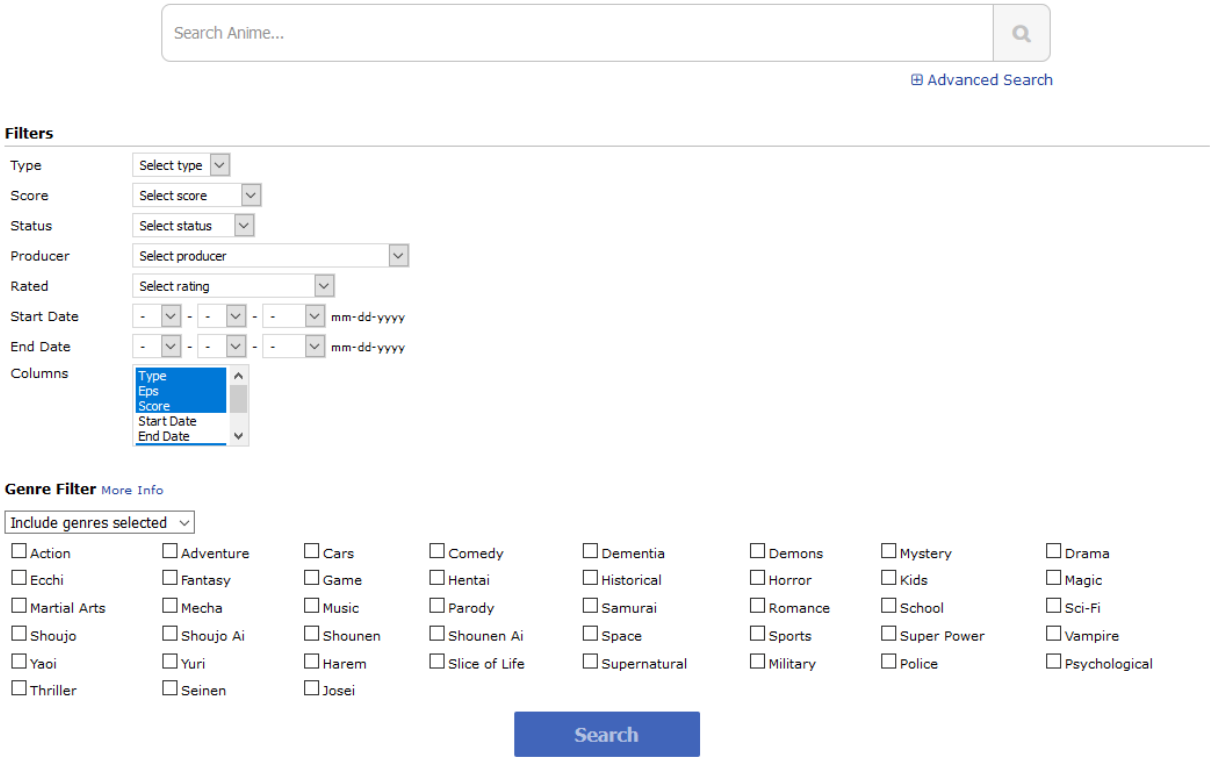

Abbildung 2 Inspiration - myanimelist.net

2. Grundlage

2.1 Aufbau

Für den Aufbau der Webseite habe ich die Vorlage aus der Aufgabenstellung verwendet. Den Header und Footer habe ich so belassen und nicht wirklich bearbeitet, auf der linken Seite ist die Navigation für die Filter-Checkboxen, drei Radio-Button und ein Slider für den numerischen Filter. Auf der rechten Seite werden alle Anime ausgegben entsprechend der Filter, ich habe alle in eine Box ähnliche Form gebracht anstatt alle als eine Tabelle ausgeben, zuerst hatte ich geplant Flexboxen beziehungsweise weitere Grids so anzulegen um sie visuell darzustellen, daran bin ich jedoch zum Teil gescheitert, trotzdem bin ich zufrieden, wie sie nun dargestellt werden.

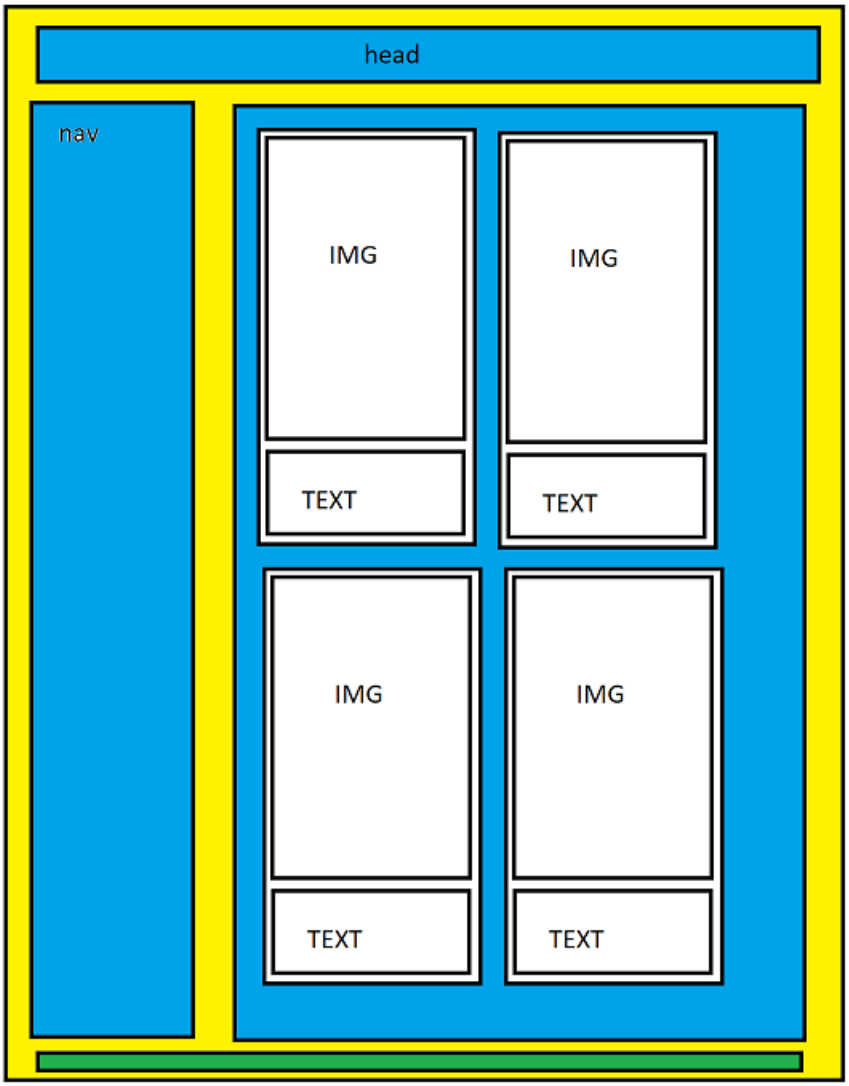

Abbildung 3 erste Skizze vom Layout

Die Struktur der Seite besteht aus der „index.php“, „connector.php“, „ajax.php“ und „index.css“. Im img Ordner sind alle Bilder hinterlegt.

2.2 Datenbank und SQL

	NUM	NAME	TYPE	EP	STUDIO	SEASON	YEAR	STATUS	SYN	IMG	SOURCE
Edit Copy Delete	1	Cowboy Bebop	TV	26	Sunrise	Spring	1998	Finished Airing	In the year 2071, humanity has colonized several o...	img/cowboyb.png	Original
Edit Copy Delete	2	Fullmetal Alchemist	TV	51	Bones	Fall	2003	Finished Airing	Edward Elric, a young, brilliant alchemist, has lo...	img/fullalch.png	Manga
Edit Copy Delete	3	Darker than Black	TV	25	Bones	Spring	2007	Finished Airing	It has been 10 years since Heaven's Gate appeared ...	img/dtb.png	Original
Edit Copy Delete	4	Baccano!	TV	13	Brains Base	Summer	2007	Finished Airing	During the early 1930s in Chicago, the transcontin...	img/baccano.png	Light novel
Edit Copy Delete	5	Phantom: Requiem for the Phantom	TV	26	Bee Train	Spring	2009	Finished Airing	Mafia is rife in America where assassinations are ...	img/phantom.png	Visual Novel
Edit Copy Delete	6	Ergo Proxy	TV	23	Manglobe	Winter	2006	Finished Airing	Within the domed city of Romdo lies one of the las...	img/ergo.png	Original
Edit Copy Delete	7	Hunter x Hunter (2011)	TV	148	Madhouse	Fall	2011	Finished Airing	Hunter x Hunter is set in a world where Hunters ex...	img/hunter.png	Manga
Edit Copy Delete	8	Dr. Stone	TV	24	TMS	Summer	2019	Finished Airing	After five years of harboring unspoken feelings, h...	img/stone.png	Manga
Edit Copy Delete	9	Jujutsu Kaisen	TV	25	Mappa	Fall	2020	Currently Airing	In a world where demons feed on unsuspecting human...	img/juju.png	Manga
Edit Copy Delete	10	Golden Kamuy 3rd Season	TV	12	Geno	Fall	2020	Not yet aired	Third season of Golden Kamuy.	img/kamuy3.png	Manga
Edit Copy Delete	11	Jin-Rou	Movie	1	Production I.G	Summer	2000	Finished Airing	After witnessing the suicide bombing of a terroris...	img/rou.png	Manga

Abbildung 4 Anime mit Spalten (Eigenschaften)

Alle Daten die auf der Seite angezeigt werden sind in einer Datenbank hinterlegt (alle Animes). Die Tabelle Anime enthält alle Informationen von einer normalen Nummerierung, Studio, Episoden und so weiter. Für die Datenbank gibt es im Allgemeinen nicht mehr zu erklären, im dritten Semester habe ich mich bereits in der Datenbank-Veranstaltung mit SQL-Befehlen und deren Aufbau beschäftigt. Hinzu kommt nun die Implementierung in PHP um die Webseite mit der Datenbank verbinden zu lassen. Dazu gibt es die „Connector.php“ welche auch verlinkt wird in „index.php“ und „ajax.php“ um dort SQL befehle zu schreiben. Ich habe mich um entschieden zu PDO anstatt mysqli Aufgrund von Tutorials und einer etwas besseren Dokumentation. SQL Befehle habe ich verwendet um die Informationen der Anime als Reihe auszugeben, sowohl für die Filter-Checkboxen als auch die eigentliche Ausgabe auf der rechten Seite der Webseite. Die Filter-Checkboxen stehen in dem PHP Code in der „index.php“. Ein SQL-Befehl wird in der Variable $query geschrieben. Durch die PDO-Befehle werden diese SQL-Anweisungen ausgeführt und in den Variablen $result gepackt. Wichtig ist das diese als Array angegeben sind. Dadurch werden alle Informationen aus der Tabelle in einer foreach-Schleife ausgegeben.

```php
<?php

$query = "
SELECT DISTINCT(season) FROM anime WHERE num != '0' ORDER BY season DESC
";
$statement = $connector->prepare($query);
$statement->execute();
$result = $statement->fetchAll();
foreach($result as $row)
{
?>
<div class="checkbox">
    <label><input type="checkbox" class="common_selector season" value="<?php echo $row['season']; ?>" > <?php echo $row['season']; ?></label>
</div>
<?php } ?>
```

Abbildung 5 Beispiel einer SQL Anweisung und Ausgabe in <NAV

2.3 Scripte: jQuery, Ajax

Funktionen werden verwendet um einen Filter zu erschaffen und Buttons ihre Funktion zu geben. Da diese Scripte hauptsächlich Neu für mich sind brauchte ich hier viel Hilfe vom Internet. Dokumentationen von Befehlen als auch Tutorials auf Webseiten und von Youtube gaben mir hier den Einblick, ohne wäre ich nicht so weit gekommen. Dazu entsteht hier eine eigene PHP Datei für die Filter der Daten auf der Webseite, diese Methoden und Funktionen waren für mich komplett neu, obwohl sie mich relativ an JavaScript erinnert haben. Zum ersten Mal habe ich mich in dieser Aufgabe intensiv mit Buttons und ihre Funktionen gekümmert und dazu kommt die mein wirklich erster Slider für den numerischen Filter auf der Webseite. Ich hatte relativ lange Problem

sogar mit diesem Slider, am Anfang habe ich ihn nicht mal visuell auf meiner Webseite gesehen bis ich gemerkt hatte, jQueryUI zu verlinken. Alle Scripte sind dazu verlinkt und nicht direkt in der Struktur der Seite gespeichert.

```
<script src="https://code.jquery.com/jquery-1.12.4.js"></script>
<script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
```

Abbildung 6 Verlinkung der Scripte

2.4 CSS und Aussehen

Das Aussehen der Seite kommt hauptsächlich aus der Vorlage der Grids, alles Restliche entstand aus meiner persönlichen Präferenz und vor allem aus meiner Skizze, jedoch basiert die Umsetzung letztens endlich auf etwas Anderes. Für CSS muss ich jedoch auch immer noch auf Tutorials zurückblicken um die Befehle richtig zu verwenden. Die Modal-Box basiert rein auf CSS, meine erste Umsetzung es mit JavaScript zu machen, schlug aus einen mir unbekannten Grund fehl. Viel mehr gibt es direkt zum Aussehen nicht, mir erschien das Aussehen als nebensächlich daher habe ich damit nicht mehr gemacht, hätte ich doch noch mehr Zeit gehabt, hätte ich auch noch einen schönen Hintergrund gemacht, als auch eine Startseite, welche dann einen zum Filter weiterleitet.

3. Funktionen und Filter

Die Idee der Filter ist dem User die Daten anzuzeigen, welche er nach deren Eigenschaften vorgibt und diese mit passenden Eigenschaften wieder zurück zu geben. Dafür habe ich erstmal mit Checkboxen angefangen, der User soll also eine oder mehrere Checkboxen anklicken um deren Eigenschaften zu übermitteln, nach diesen wird dann sortiert. Auf der zweiten Abbildung wird so eine Checkbox bereits dargestellt. Diese beinhaltet die Eigenschaft „Season“ worunter also nach der Datenbank: Winter, Spring, Summer und Fall zählen. Dementsprechend werden dem User also vier Checkboxen angezeigt und er hat dadurch die Möglichkeit eine oder mehrere Auszuwählen und deren Ergebnisse angezeigt zu bekommen. Nun muss

jedoch dieser Wert jedoch übergeben werden und durch einen SQL Befehl sortiert werden.

```
function filter_data()
{

    var action = 'ajax';
    var min_year = $('#min_year').val();
    var max_year = $('#max_year').val();
    var status = get_filter('status');
    var studio = get_filter('studio');
    var season = get_filter('season');
    var type = get_filter('type');
    $.ajax({
        url:"ajax.php",
        method:"POST",
        data:{action:action, min_year:min_year, max_year:max_year, studio:studio, season:season, status:status, type:type,},
        success:function(data){
            $('.filter_data').html(data);
        }
    });
}
```

Abbildung 7 Funktion filter_data

Die Funktion filter_data() enthält Variablen welche vor allem diese Eigenschaften sind um die Daten zu filtern. Action wird genutzt um die Daten weiter an die PHP Seite zu leiten. Die Funktion get_filter dient dafür die entsprechende Eigenschaft der Tabelle aus der Checkbox zu entnehmen und hier einzufügen.

```
function get_filter(class_name)
{
    var filter = [];
    $('.'+class_name+':checked').each(function(){
        filter.push($(this).val());
    });
    return filter;
}
```

Abbildung 8 Funktion get_filter

Class_name entspricht dadurch immer diejenige class, in der Checkbox (Studio, Season, ...). Dann wird es als Array zurückgegeben. Nun befinden wir uns auf einen neuen PHP Seite, welche in der URL im $ajax Code angegeben wurde. „ajax.php“

enthält den PHP Code mit den eigentlichen Filter.

```php
if(isset($_POST["action"]))
{
    $query = "
        SELECT * FROM anime WHERE num != '0'
    ";
    if(isset($_POST["min_year"], $_POST["max_year"]) && !empty($_POST["min_year"]) && !empty($_POST["max_year"]))
    {
        $query .= "
         AND year BETWEEN '".$_POST["min_year"]."' AND '".$_POST["max_year"]."'
        ";
    }
    if(isset($_POST["studio"]))
    {
        $studio = implode("','", $_POST["studio"]);
        $query .= "
         AND studio IN('".$studio."')
        ";
    }
```

Abbildung 9 ajax.php Beispiel

Hier in if(isset) wird nun geprüft für jede Eigenschaft (von Slider-Year, Season, …) ob ein Wert gesetzt ist, dann wird in der $query Variablen ein SQL-Befehl gesetzt. Der Befehl ist der eigentliche Filter, da er nur diese Werte ausgibt welche vorher in der Eigenschafts-Variablen ($studio Beispiel) gesetzt wurde und nur die passenden Daten, dann von der Tabelle Anime ausgibt. Das gilt so für jede Eigenschaft, bei Year wird jedoch eine Auswahl von zwei Zahlen ausgegeben, dementsprechend alle passenden Zahlen von der kleinsten Zahl (min_year) zur größten Zahl (max_year). Der Befehl implode transformiert Array in Strings um dadurch die angegebene Eigenschaft in normales Wort in dem SQL-Befehl anzugeben.

```php
$statement = $connector->prepare($query);
$statement->execute();
$result = $statement->fetchAll();
$total_row = $statement->rowCount();
$output = '';
if($total_row > 0)
{
    foreach($result as $row)
    {
        $output .= '
        <div class="item">
            <img src="'.$row['IMG'].'">
                    <h4>'. $row['NAME'] .'</h4>
                    <p>
                    EPISODES: '. $row['EP'] .'<br>
                    STUDIO: '. $row['STUDIO'] .'<br>
                    SOURCE: '. $row['SOURCE'] .'<br>
                    SEASON: '. $row['SEASON'] .'<br>
                    YEAR: '. $row['YEAR'] .'<br>
                    TYPE: '.$row['TYPE'] .'<br>
                    SUMMARY:

                    <a class="button" href="#popup'.$row['NUM'].'">Read</a>
                    <div id="popup'.$row['NUM'].'" class="overlay">
                        <div class="popup">
                            <a class="close" href="#">&times;</a>
                            <p class="content"> '. $row['SYN'] .' </p>
                        </div>
                    </div>
                    </p>
        </div>
        ';
    }
}
else
{
    $output = '<h3>No Anime fits the filter</h3>';
}
echo $output;
```

Abbildung 10 ajax.php Ausgabe

Die Variable $query wird nach den if-Anweisungen dann hier im $statement ausgeführt und alle Daten werden genommen (fetchAll) und in $result platziert, $total_row ergibt sich aus alle Reihen aus $statement durch rowCount() und solange es mehr Reihen gibt als 0 gibt es den $output für die class filter_data, welche alle Anime anzeigt, beziehungsweise nun auch gefiltert wiedergibt. Wenn es keine Reihen gibt an Daten (bsp. Filter entspricht keinem Anime) wird die else-Anweisung ausgegeben.

4. Zusammenfassung

Meine hier teils erklärte Webseite dient also hauptsächlich nach Aufgabenstellung, drei verschiedene Filter, einen Filter nach Jahr in einem Slider, mehrere Filter Checkboxen nach Studio, Season und so weiter und drei Radio-Button Filter welche jeweils drei bestimmte Eigenschaften wiedergeben aber nicht gleichzeitig alle wiedergeben. Zur Aufgabe selber hat mir eigentlich alles gefallen, Web-Design und eben alles was zur Programmierung im Web gehört, gefällt mir persönlich recht gut und vor allem viel mehr als die reine Java-Programmierung in den vorherigen Semestern. Trotzdem konnte ich nicht einfach mit dem programmieren beginnen, es braucht immer etwas Zeit und viel Hilfe aus dem Netz um bestimme Funktionen oder auch Befehle zu verstehen, sobald man jedoch eine Idee und Vorstellung im Kopf hat kann man recht gut danach arbeiten. Mein größeres Problem war die Zeit, während ich jetzt schon die Dokumentation schreibe, investiere ich hier am wenigsten Zeit, ich war mehr mit der Aufgabe selber beschäftigt und sowohl auch mit anderen Aufgaben aus diesem Semester, welche bis zum September verlegt wurden. Dazu fällt mir nun während ich diese Dokumentation schreibe neue Ideen für die Webseite ein, welche man hätte schon erledigen können, allerdings hab ich die ungefähre Länge der Dokumentation komplett unterschätzt. Dadurch fällt wieder als auch in anderen Abgaben auf, mehr Planung. Sich vor jeder Aufgabe bestimmte Skizzen zu zeichnen oder einfach zu Brainstormen kann mir sicherlich vieles erleichtern, jedoch immer wieder arbeite ich einfach nach keinen direkten Plan, sondern eben einfach nach Ideen die mir Stück für Stück einfallen. Dazu kommt auch die Zeit die ich gebraucht habe, nur um am Ende eine Idee nicht hinzubekommen wie ich es wollte, dadurch Zeit zu verlieren und dann wieder an einer neuen zu arbeiten, braucht sehr viel Zeitanspruch.

Beispiele für eigentliche Erweiterungen der Webseite welche ich nicht geschafft habe:

- Sortierung der ganzen Ergebnisse, Name, Episoden, usw.
- Die Einbindung meiner Genre Tabelle und damit den Filter nach Genre
- Einen richtigen Header mit eventuellen Verlinkungen nach Impressum usw.
- Das Gleiche für den Footer
- Aussehen der Webseite, Hintergrundbild, schönere Farben

- Mehrere Daten, dadurch mehrere Filter
- Top-Down-Menü Filter
- TextBox Filter um eventuell nach bestimmten Wörtern zu suchen (Beispiel nach Wörter im Titel) oder Zahlen
- Episoden-Slider
- Eine Startseite mit Verlinkungen
- Einen richtigen „reset“ Button um alle Filter zu deaktivieren und dann alle Daten wieder anzuzeigen

Dennoch bin ich persönlich mit meiner erreichten Leistung zufrieden und werde wohlmöglich auch unabhängig von der Aufgabe nochmal über diese Webseite gehen und einige Funktionen und Features einbauen. Ich erhoffe mir die Dokumentation ist ausreichend und ggf. enthält der Code zum Teil eine hoffentlich gute Beschreibung.