

**HÖHERE TECHNISCHE BUNDES - LEHR- UND   
VERSUCHSANSTALT MÖDLING**

**Höhere Lehranstalt für  
Elektronik und Technische Informatik**

HTL_Logo

**DIPLOMARBEIT**

Gesamtprojekt

**W2W-Where/What to Watch**

**<JavaScript, TypeScript>**

<Abdelrahaman> <Shehata>5BHEL Betreuer/in: Dipl.-Ing Niklas Hack

**<CSS>**

<Marco> <Rustemi> 5BHEL Betreuer/in: Dipl.-Ing Niklas Hack

**<HTML>**

<Robert> <Ardelean> 5BHEL Betreuer/in: Dipl.-Ing Niklas Hack

ausgeführt im Schuljahr 2019/20

Abgabevermerk:

Datum: übernommen von:

# Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Diplomarbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch bei keiner anderen Prüferin/ keinem anderen Prüfer als Prüfungsleistung eingereicht. Mir ist bekannt, dass Zuwiderhandeln mit der Note „nicht genügend“ (ohne Möglichkeit einer Nachbesserung oder Wiederholung) geahndet wird und weitere rechtliche Schritte nach sich ziehen kann.

Diese Arbeit wurde neben der gedruckten Version auch auf CD-ROM zur Prüfung der o.g. Erklärung bei der zuständigen Prüferin/dem zuständigen Prüfer hinterlegt.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ort und Datum)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Unterschrift <Abdelrahaman Shehata>)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Unterschrift <Marco Rustemi>)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Unterschrift <Robert Ardelean>)

**DIPLOMARBEIT**

**DOKUMENTATION**

|  |  |
| --- | --- |
| Namen der Verfasser/innen | Abdelrahaman Shehata, Marco Rustemi, Robert Ardelean |
| Jahrgang  Schuljahr | 5BHEL  2020/2021 |
| Thema der Diplomarbeit | W2W- Where/What to Watch |
| Kooperationspartner | - |

|  |  |
| --- | --- |
| Aufgabenstellung | Die Aufgabe unserer App/Website ist es dem Benutzer Zeit zu ersparen. Die Website soll dem Benutzer zeigen, welche Filme/Serien bald auf den beliebtesten Streaming-Diensten hochgeladen bzw. von den Streaming-Diensten entfernt werden. |

|  |  |
| --- | --- |
| Realisierung | * Angular * TypeScript * VisualCode * Bulma * SwiperJS * GitHub * uNoGS   Wir haben uns dafür entschieden unsere Diplomarbeit mit Angular zu realisieren. |

|  |  |
| --- | --- |
| Ergebnisse |  |

|  |  |
| --- | --- |
| Typische Grafik, Foto etc.  (mit Erläuterung) |  |

|  |  |
| --- | --- |
| Teilnahme an Wettbewerben, Auszeichnungen | Noch nicht aber kommt! |

|  |  |
| --- | --- |
| Möglichkeiten der Einsichtnahme in die Arbeit | Im Archiv der Abteilung Elektronik und Technische Informatik der HTL Mödling |

|  |  |  |
| --- | --- | --- |
| Approbation  (Datum / Unterschrift) | Prüfer/Prüferin | Direktor/Direktorin  Abteilungsvorstand/Abteilungsvorständin |

**DIPLOMA THESIS**

**Documentation**

|  |  |
| --- | --- |
| Author(s) | Abdelrahaman Shehata, Marco Rustemi, Robert Ardelean |
| Form  Academic year | 5BHEL  2020/2021 |
| Topic | W2W- Where/What to Watch |
| Co-operation partners | - |

|  |  |
| --- | --- |
| Assignment of tasks | The main task of our App/Website is to save time for the users. The website will show the user which movies/series will get on the streaming service or will be taken down. |

|  |  |
| --- | --- |
| Realisation |  |

|  |  |
| --- | --- |
| Results |  |

|  |  |
| --- | --- |
| Illustrative graph, photo  (incl. explanation) |  |

|  |  |
| --- | --- |
| Participation in competitions Awards | Not now but in the future |

|  |  |
| --- | --- |
| Accessibility of  Diploma Thesis | Stowed in the archive of the secondary technical college of Moedling, department of electronics and computer engineering |

|  |  |  |
| --- | --- | --- |
| Approval  (Date / Signature) | Examiner | Head of College / Department |

# Inhaltsverzeichnis

[1 Eidesstattliche Erklärung 2](#_Toc34983750)

[2 Inhaltsverzeichnis 7](#_Toc34983751)

[3 Pflichtenheft 10](#_Toc34983752)

[3.1 Zielsetzung 10](#_Toc34983753)

[3.2 Wunschkriterien 10](#_Toc34983754)

[3.3 Abgrenzungskriterien 10](#_Toc34983755)

[3.4 Projektumfeldanalyse 10](#_Toc34983756)

[3.4.1 Plattformabhängigkeit 11](#_Toc34983757)

[3.4.1.1 Smartphone/Tablets 11](#_Toc34983758)

[3.4.1.2 Computer 11](#_Toc34983759)

[3.4.2 Technische Produktumgebung 11](#_Toc34983760)

[3.5 Sonstiges 11](#_Toc34983761)

[3.5.1 Kooperationspartner 11](#_Toc34983762)

[4 Projektplan 12](#_Toc34983763)

[4.1 Gesamtprojektplan 12](#_Toc34983764)

[4.2 Planung 12](#_Toc34983765)

[4.3 Realisierung 12](#_Toc34983766)

[5 Arbeitsplan 13](#_Toc34983767)

[5.1 Planung 13](#_Toc34983768)

[5.1.1 Pflichtenheft 13](#_Toc34983769)

[5.1.2 Systemspezifikation 13](#_Toc34983770)

[5.1.3 Projekt – und Arbeitsplan 13](#_Toc34983771)

[5.2 Realisierung 13](#_Toc34983772)

[6 Umfeldanalyse 14](#_Toc34983773)

[6.1 Entwicklungsumgebung 14](#_Toc34983774)

[6.1.1 Notepad++ 14](#_Toc34983775)

[6.1.1.1 Vorteile 14](#_Toc34983776)

[6.1.1.2 Nachteile 14](#_Toc34983777)

[6.1.1.3 Kosten 14](#_Toc34983778)

[6.1.2 Visual Studio 14](#_Toc34983779)

[6.1.2.1 Vorteile 14](#_Toc34983780)

[6.1.2.2 Nachteile 14](#_Toc34983781)

[6.1.2.3 Kosten 14](#_Toc34983782)

[6.1.3 Visual Studio Code 15](#_Toc34983783)

[6.1.3.1 Vorteile 15](#_Toc34983784)

[6.1.3.2 Nachteile 15](#_Toc34983785)

[6.1.3.3 Kosten 15](#_Toc34983786)

[6.1.4 Entscheidung 15](#_Toc34983787)

[6.2 Frameworks 15](#_Toc34983788)

[6.2.1 Angular 15](#_Toc34983789)

[6.2.1.1 Vorteile 15](#_Toc34983790)

[6.2.1.2 Nachteile 16](#_Toc34983791)

[6.2.2 React 16](#_Toc34983792)

[6.2.2.1 Vorteile 16](#_Toc34983793)

[6.2.2.2 Nachteile 16](#_Toc34983794)

[6.2.3 Entscheidung 16](#_Toc34983795)

[6.3 Anwendungsschnittstelle (API) 16](#_Toc34983796)

[6.3.1 uNoGS 16](#_Toc34983797)

[6.3.1.1 Vorteile 16](#_Toc34983798)

[6.3.1.2 Nachteile 16](#_Toc34983799)

[6.3.2 Streamzui 17](#_Toc34983800)

[6.3.2.1 Vorteile 17](#_Toc34983801)

[6.3.2.2 Nachteile 17](#_Toc34983802)

[6.3.3 Netflix 17](#_Toc34983803)

[6.3.3.1 Vorteile 17](#_Toc34983804)

[6.3.3.2 Nachteile 17](#_Toc34983805)

[6.3.4 Amazon Prime 17](#_Toc34983806)

[6.3.4.1 Vorteile 17](#_Toc34983807)

[6.3.4.2 Nachteile 17](#_Toc34983808)

[6.3.5 Entscheidung 17](#_Toc34983809)

[7 Systemspezifikation 18](#_Toc34983810)

[7.1 Blockschaltbild 18](#_Toc34983811)

[7.1.1 Allgemeines Blockschaltbild 18](#_Toc34983812)

[7.2 Systemüberblick 18](#_Toc34983813)

[7.2.1 Funktionalität der Baugruppen 18](#_Toc34983814)

[7.2.1.1 Baugruppe I 18](#_Toc34983815)

[7.2.1.2 Baugruppe II 18](#_Toc34983816)

[7.3 Externe Schnittstellen 19](#_Toc34983817)

[7.3.1 <HTTP> 19](#_Toc34983818)

[7.3.2 <API> 19](#_Toc34983819)

[8 Use Cases (Testfälle) 20](#_Toc34983820)

[8.1 Use Case „Aufruf ohne Internetverbindung“ 20](#_Toc34983821)

[8.1.1 Randbedingung 20](#_Toc34983822)

[8.1.2 Testablauf 20](#_Toc34983823)

[8.1.3 Erwartetes Ergebnis 20](#_Toc34983824)

[8.2 Use Case „Aufruf mit Internetverbindung” 20](#_Toc34983825)

[9 Detailspezifikation [OPTIONAL] 21](#_Toc34983826)

[9.1 Detailspezifikation <Abc> 21](#_Toc34983827)

[9.1.1 <Detail 1> 21](#_Toc34983828)

[9.1.2 <Detail 2> 21](#_Toc34983829)

[9.2 Detailspezifikation <Xyz> 21](#_Toc34983830)

[9.2.1 <Detail 998> 21](#_Toc34983831)

[9.2.2 <Detail 999> 21](#_Toc34983832)

[9.3 Interne Schnittstellen 21](#_Toc34983833)

[9.3.1 <Schnittstelle S> 21](#_Toc34983834)

[9.3.2 <Schnittstelle T> 21](#_Toc34983835)

[10 Benutzerhandbuch 22](#_Toc34983836)

[10.1 Benutzerhandbuch <Teil Abc> 22](#_Toc34983837)

[10.2 Benutzerhandbuch <Teil Xyz> 22](#_Toc34983838)

[11 Testfallspezifikation 23](#_Toc34983839)

[11.1 Testgruppe (Betriebsbereitschaft) 23](#_Toc34983840)

[11.1.1 Testfall <A> 23](#_Toc34983841)

[11.1.2 Testfall <B> 23](#_Toc34983842)

[12 Webseitendesign/ -programmierung 24](#_Toc34983843)

[12.1 Startseite 24](#_Toc34983844)

[Diese Seite kann unter dem Link https://W2W.azurewebsite.com gefunden werden. Dieser Link ist eine externe Zugriffsmöglichkeit auf die Hauptseite unserer Website. Die Startseite unserer Website wird ausfolgenden Dateien gebildet: index.html, bulma.css und bulma.js. 24](#_Toc34983845)

[12.1.1 Einleitung 24](#_Toc34983846)

[12.1.2 Head 24](#_Toc34983847)

[12.1.2.1 HTML 24](#_Toc34983848)

[12.1.3 Body 24](#_Toc34983849)

[12.1.3.1 MainGrid 24](#_Toc34983850)

[12.1.3.1.1 HTML 24](#_Toc34983851)

[12.1.3.1.2 CSS 24](#_Toc34983852)

[13 Abbildungsverzeichnis 25](#_Toc34983853)

[14 Begleitprotokolle 25](#_Toc34983854)

[14.1 Begleitprotokoll <Schüler 1> 25](#_Toc34983855)

[14.2 Begleitprotokoll <Schüler 2> 26](#_Toc34983856)

[14.3 Begleitprotokoll <Schüler 3> 26](#_Toc34983857)

[15 Anhang 26](#_Toc34983858)

[15.1 Istbestand 26](#_Toc34983859)

[15.2 Angebote 26](#_Toc34983860)

[15.3 Lieferscheine 26](#_Toc34983861)

[15.4 Dimensionierung 26](#_Toc34983862)

[15.5 Messprotokolle 26](#_Toc34983863)

[15.6 Testprotokolle 26](#_Toc34983864)

# Pflichtenheft

## Zielsetzung

Ziel ist es auf unsere Website ,,W2W Where/What to Watch“ eine funktionierende Kommunikation zwischen Website und dem Server über die API herzustellen. Der User soll von einem benutzerfreundlichen und innovativen Design überzeugt werden, die Website öfters zu benutzen. Wenn der User die Website aufruft soll ihm angezeigt werden welche Serien/Filme in nächster Zeit verschwinden werden oder neu zum Streamingdienst hinzugefügt werden.

## Wunschkriterien

Es gibt noch einige Verbesserungs-/Erweiterungsmöglichkeiten, die man hinzufügen könnte zu unserer Website, jedoch sind diese nur dazu da, um den User ein noch besseres Erlebnis beim Benutzen der Website zu bekommen. Die folgenden Änderungen, die wir noch gerne hinzufügen würden, haben keine Auswirkung auf die Funktionalität unserer Website!

* Der User soll die Möglichkeit haben zwischen verschieden Streamingdiensten zu wählen, da zurzeit nur ein Streamingdienst ausgewählt werden kann.
* Der User soll die Möglichkeit haben sich auf unserer Website einzuloggen und dadurch verschieden Extras freizuschalten. Diese Extras wären zum Beispiel Benachrichtigungen über seine Lieblingsserien zu bekommen oder sein Lieblingsgenre auszuwählen, um auf der Startseite Vorschläge vom gewählten Genre zu bekommen.
* Wir wollen auch noch eine Datenbank implementieren, die es uns ermöglicht die Daten, die wir über die API vom Server bekommen abzuspeichern. Dadurch ersparen wir uns Kosten beim wiederholten Aufrufen über die API des Streamingdienstes, da wir dann nur noch drei Mal am Tag einen API-Call machen müssen.

## Abgrenzungskriterien

Es ist nicht Teil des Projekts, mehrere Streamingdienste zur Auswahl zu geben beziehungsweise diese Funktion in die Website einzubauen !!!!!!!!!!!!!!!!!

## Projektumfeldanalyse

Das Projekt hat keine Vorlage und setzt auf keinem alten Projekt beziehungsweise auf keiner Idee eines anderen Projekts auf. Wir haben uns im Internet als Hilfestellung verschiedene Designs von Websites angeschaut, um unser Wunschdesign zu kreieren. Die Darstellung der Website wurde neu erfunden und nicht von einer anderen Website kopiert.

### Plattformabhängigkeit

#### Smartphone/Tablets

Durch CSS3 ist es möglich die Website in einer optimierten Ansicht auf Smartphones zu benutzen. Bei der Ansicht am Mobiltelefon werden die Steuerelement und das Design angepasst an die Größe des Geräts. Die eigentlichen Funktionen bleiben trotz der Anpassung erhalten. Die Funktion der Website wird bei allen Mobilen Geräten, die einen Browser haben der regelmäßig auf den neusten Stand gebracht wird, gewährleistet. Das oben geschriebene gilt auch für das Tablet.

#### Computer

Die Darstellung ist auch für den Computer optimiert und die Funktionalitäten sind die gleichen wie bei der mobilen Ansicht. Zu beachten ist, dass das Design anders dargestellt wird als bei Smartphones/Tablets. Die Benutzerfreundlichkeit bleibt wie bei den mobilen Geräten erhalten.

### Technische Produktumgebung

Die Website wurde mit dem Framework „Angular.io“ geschrieben. Angular bietet die Möglichkeit Single-Page-Applikationen zu programmieren und wird heutzutage auch oft von größeren Unternehmen verwendet. VisualCode verwenden wir als unseren Editor bzw. Entwicklungsumgebung. Die Website „Azure.microsoft.com“ ist die Hostwebsite unserer Website.

## Sonstiges

### Kooperationspartner

Wir haben für unser Projekt keine Kooperationspartner. Wir haben keine Hilfe von anderen Programmierern bzw. Designern in Anspruch genommen. Dieses Projekt entstand nur durch die 3 oben angegebenen Teilnehmer.

# Projektplan

## Gesamtprojektplan

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  | **Projekt** | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  | Planung | |  |  |  |  |  |  |  |
|  |  |  | Realisierung | |  |  |  |  |  |
|  |  |  |  |  | Test | |  |  |  |
|  |  |  |  |  |  |  | Abgabe | |  |
|  |  |  |  |  |  |  |  |  |  |
| Projektstart  1.9.2019 | | Planung ENDE  12.10.2019 | | Realisierung ENDE  15.1.2020 | | Testen ENDE  15.2.2020 | | Projekt FERTIG  7.4.2020 | |
|  | |  | |  | |  | |  | |

## Planung

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  | **Planung** | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  | Projektanalyse | |  |  |  |  |  |  |  |
|  |  |  | Pflichtenheft | |  |  |  |  |  |
|  |  |  |  |  | Systemspez. | |  |  |  |
|  |  |  |  |  |  |  | Detailspez. | |  |
|  |  |  |  |  |  |  |  |  |  |
| Projektstart  1.9.2019 | | Analyse ENDE 15.9.2019 | | Pflichtenheft FERTIG 25.9.2019 | | Systemspez. FERTIG 1.10.2019 | | Planung FERTIG 12.10.2019 | |
|  | |  | |  | |  | |  | |
|  | |  | |  | |  | |  | |

## Realisierung

<Termine für alle Phasen aus dem Gesamtprojektplan …>

# Arbeitsplan

## Planung

### Pflichtenheft

|  |  |
| --- | --- |
| **Name** | **Tätigkeit / Verantwortung** |
| Shehata | Autor, Hauptverantwortlich |
| Rustemi | Autor |
| Ardelean | Autor |

### Systemspezifikation

|  |  |
| --- | --- |
| **Name** | **Tätigkeit / Verantwortung** |
| Rustemi | Autor |
| Ardelean | Autor |
| Shehata | Autor, Hauptverantwortlich |

### Projekt – und Arbeitsplan

|  |  |
| --- | --- |
| **Name** | **Tätigkeit / Verantwortung** |
| Shehata | Autor, Hauptverantwortlich |
| Rustemi | Autor |
| Ardelean | Autor |

## Realisierung

<Arbeitsplan für alle Phasen aus dem Gesamtprojektplan …>

# Umfeldanalyse

## Entwicklungsumgebung

Die Auswahlkriterien:

1. Syntax-Highlighting
2. Übersichtlicher Dateien-Manager
3. Angemessener CPU-Verbrauch

### Notepad++

Version: 7.5.6 64-Bit (19. März 2019)

Lizenz: Freeware

#### Vorteile

* Plug-In Erweiterbar
* Automatische Codevervollständigung per Tastendruck
* Syntax-Highlighting
* Geringer CPU-Verbrauch

#### Nachteile

* Nicht übersichtlicher Dateien-Manager
* Split-Screening der Dateien nur begrenzt möglich (maximal 2 Dateien möglich)

#### Kosten

Da es sich hier um eine Freeware handelt, würden keine Kosten entstehen.

### Visual Studio

Version: Enterprise 2019

Lizenz: Kostenlos durch Schullizenz (ansonsten 7707€/einmalig)

#### Vorteile

* Benutzte Umgebung während der Schulzeit
* Syntax-Highlighting
* Übersichtlicher Dateien-Manager
* Automatischer Codevervollständigung per Tastendruck
* Git Extension inkludiert

#### Nachteile

* Sehr hoher CPU-Verbrauch, durch viele irrelevante Funktionen
* Start-Ladezeiten

#### Kosten

Da die HTL Mödling uns eine Schülerlizenz zur Verfügung stellt, würden keine Kosten entstehen

### Visual Studio Code

Version: 1.39.2 64-Bit (15. Oktober 2019)

Lizenz: Freeware

#### Vorteile

* Sehr ähnliche Umgebung wie Visual Studio
* Syntax-Highlighting
* Mit Typescript entwickelt worden
* Übersichtlicher Dateien-Manager
* Automatischer Codevervollständigung per Tastendruck
* Split-Screening mehrfach nutzbar
* Erweiterbar durch Plug-Ins
* Geringer CPU-Verbrauch
* Git Extension inkludiert

#### Nachteile

* Keine nennenswerten Nachteile

#### Kosten

Da es sich hier um ein OpenSource-Programm handelt, würden keine Kosten entstehen.

### Entscheidung

Am Anfang war die Entscheidung schwierig, da wir uns bis dahin nur mit Visual Studio auseinandergesetzt haben, aber nach einigen Gegenüberstellungen der genannten Entwicklungsumgebungen haben wir einen klaren Favoriten deklariert. Am Ende fiel die Wahl auf Visual Studio Code.

## Frameworks

Die Auswahlkriterien:

1. Einfache Syntax
2. Keine Kosten
3. Komponenten Responsive

### Angular

Version: 8.2.14

Lizenz: Open Source Software

#### Vorteile

* Responsive und Mobile-First Design
* Single-Page-Application
* Kann sowohl auf dem Client als auch auf dem Server gerendert werden
* Keine Kosten, da Open Source
* Kurze Ladezeiten
* Hat ein Grundgerüst
* Vertraut durch Praxisunterricht
* Erlaubt es Progressive Web-Apps zu entwickeln

#### Nachteile

* Einfache Syntax, jedoch mit einer steilen Lernkurve
* Lückenhafte Kommandozeilen-Dokumentation

### React

Version: 16.12.0

Lizenz: Open Source

#### Vorteile

* Serverseitiges Rendering
* Komponentenbasierend
* Einseitige Data-Binding
* Keine Kosten, da Open Source
* Responsive und Mobile-First Design

#### Nachteile

* Fast schon eine Library
* Erst vor kurzem veröffentlicht worden

### Entscheidung

Da wir mit Angular schon im Praxisunterricht in Berührung gekommen sind, haben wir uns entschieden Angular zu verwenden.

## Anwendungsschnittstelle (API)

Die Auswahlkriterien:

* Übersichtliche Informationen
* Keine Lizenz
* Viele Informationen
* Keine Kosten

### uNoGS

#### Vorteile

* Keine Kosten
* Keine Lizenz nötig
* Viele Informationen
* Übersichtliche Informationen

#### Nachteile

* Ist nicht die originale API des Streaming-Anbieters

### Streamzui

#### Vorteile

* Viele Informationen
* Übersichtliche Informationen

#### Nachteile

* Kosten (9.00€ /Monat für 3000 Aufrufe/Monat)
* Nur Filme, keine Serien

### Netflix

#### Vorteile

* Unabhängig von Drittanbieter, da offizieller Anbieter
* Übersichtliche Informationen
* Alle Informationen (bzgl. Netflix)
* Idealer Ansprechpartner bei Fragen, Problemen etc.

#### Nachteile

* Lizenz nötig, die Netflix nur nach gewissen Kriterien verteilt

### Amazon Prime

#### Vorteile

* Unabhängig von Drittanbieter, da offizieller Anbieter
* Übersichtliche Informationen
* Alle Informationen (bzgl. Amazon Prime)
* Idealer Ansprechpartner bei Fragen, Problemen etc.

#### Nachteile

* Lizenz nötig, die Amazon Prime nur nach gewissen Kriterien verteilt

### Entscheidung

Eigentlich war geplant, dass wir die API direkt von den Streaming Diensten implementieren, jedoch hatten wir große und unerwartete Schwierigkeiten die dazugehörige Lizenz zu erhalten. Da bei Drittanbietern meisten Kosten anfallen, wie zum Beispiel Streamzui, haben wir uns wir den kostenlosen Anbieter uNoGS entschieden.

# Geschichte

## GitHub

## Was ist GitHub?

Der Onlinedienst GitHub, der seit 2018 zu Microsoft gehört, hat ihren Sitz in San Francisco und stellt für Software-Entwicklungsprojekte seine Server bereit. GitHub ist einer der größten Filehosting Webdienste der Welt.

## Wozu benutzen wir GitHub?

Wir benutzen GitHub, um unser Projekt-Ordner und die darin liegende Dateien miteinander zu teilen. Dabei können wir die Dateien unseres Partners runterladen, verändern und wiederhochladen, während ein anderer diese Datei noch immer modifiziert. Dabei protokolliert GitHub die jeweiligen Up- und Downloads, so kann jeder der auf das Repository zugreifen kann, schauen wann welche Veränderungen durchgeführt wurden.

Beim Hochladen einer modifizierten Datei kann es jedoch zu Problemen führen, wenn zwei Personen auf derselben Stelle die Datei modifiziert (Merge-Fehler). Falls so ein Fehler auftritt muss die jeweilige Person bestimmen, ob etwas gelöscht, überschrieben oder nur ein Teil verschoben werden muss, damit der Merge-Fehler nicht mehr auftritt.

## Wichtige Begriffe

### Repository

Ein „Repository“ ist eine zentrale Ablage, bei den Objekten (Dateien, Programme, Ordner etc.)

### Commit

Ein „Commit“ ist ein Schreibzugriff, der die jeweiligen Dateien verändert bzw. überschreibt.

### Push

Ein „Push“ ist ein Upload, der die veränderte („commited“) Datei auf die Server hochladet und danach den anderen Nutzern zur Verfügung steht.

### Pull

Ein „Pull“ ist ein Download, der das Repository, Datei oder Ordner runterladet und dabei das Original über die

### Branch

Ein „Branch“ ist eine Abspaltung von einer anderen Version, dabei können an unterschiedliche Versionen weiterentwickelt werden.

### Merging

Das Verschmelzen zwischen zwei Branches nennt man Merging. Beim Merging können Fehler auftreten, wenn jeweilige Zeilen

## Funktionsweise

Damit man mit dem Repository arbeiten kann, muss man den aktuellen (oder einen älteren) Stand des Projekts herunterzuladen, sofern man die nötigen Berechtigungen hat. (=Pull Request).

## Angular JS

Angular JS ist die erste Version von Angular und erschien im Jahr 2009. Angular Js wird verwendet, um Single Page Webanwendungen nach einem Model View Muster zu erstellen. Angular Js ist ein Open Source Framework vom US-amerikanischen Unternehmen Google.

# Das Konzept

Angular JS wurde so entwickelt, dass es auf der clientseitigen Generierung und den Erweiterungen des Vokabulars von HTML basiert. Die Funktionalität im Rahmen der View wird abgebildet ohne auf DOM-Manipulation via jQuery. DOM-Manipulation (Document Object Model) ist eine Schnittstelle zwischen HTML und dynamischen JavaScript. Elemente werden als Objekte dargestellt und werden dynamisch abgerufen, verändert, hinzugefügt und gelöscht. JQuery stellt Funktionen zur Verfügung für DOM-Navigation beziehungsweise DOM-Manipulation. JQuery ist eine freie JavaScript-Bibliothek und auch die meistverwendete (70% der 10.000 meistgesuchten Websites). Die Gültigkeit der Daten wird in Angular JS über den Rahmen von Eingabeformularen als Funktionalität der View behandelt. Es wird and die HTML5-Form der Validation gedacht und durch Angular JS in einen Webbrowser unabhängige JavaScript Version implementiert. Angular JS Webclients sind so aufgebaut, dass die Strukturierung über die Basis von Modulen, View-Templates, Controllern, Scopes, Filtern und Providern erfolgt. Der Dependency-Container von Angular JS ist für die Zusammenführung der Elemente zuständig. Die Anwendung, besteht nur aus lose gekoppelte wiederverwendbaren Teilkomponenten. Durch die Berücksichtigung von Iso/IEC25010 können wartbare Anwendungen programmiert werden.

# Struktur

## Controller:

Controller in Angular JS werden zu einem Model zusammengefasst. Mit Hilfe des Dependency-Containers werden die Modelle in die Applikation eingebunden. Die View und das Model sind verbunden, es herrscht eine bidirektionale Datenverbindung, das heißt, Benutzereingaben wirken sich auf das Model aus. Wenn sich das Model programmatisch ändert wirkt sich das auch auf die Benutzeransicht aus.

### Direktiven:

In Angular Js ist es möglich Direktiven zu erstellen, das heißt, es ist möglich benutzerdefinierte HTML Elemente und Attribute zu erstellen. Vordefinierte Direktiven sind erkennbar am ng-Namensraum im Präfix. Die Art des benutzenden Präfixes ist vom Validator abhängig.



Elemente werden über jQuery ausgewählt.

Interpolation:  
JavaScript Ausdrücke werden über doppelte geschweifte Klammern im HTML Code eingebettet. Es werden keine Sprunganweisungen unterstützt. Filter werden über den Pipe Operator hinzugefügt, sie wirken sich auf das angezeigt Resultat aus. Alternativ kann auch ein ng-Binding verwendet werden, dadurch wird jedoch verhindert, dass der Browser beim erstmaligen Laden die Vorlage anzeigt, falls Angular JS die Daten nicht schnell genug geladen und damit die Vorlage ersetzt hat.

## Services:

Die Geschäftslogik ist in Services enthalten, sie binden externe Ressourcen ein und werden als Singleton instanziiert. Services können selbst erstellt werden oder von Drittanbietern übernommen werden. Viele Services werden bereits von Angular JS bereitgestellt, sie sind erkennbar am $-Präfix.

## Kommunikation zwischen Scopes:

Scope-Objekte kapseln die Funktionen und Daten von Controllern. Das Scope-Objekt ermöglicht dem Controller mit anderen Controllern oder Services kommunizieren kann. Mit $emit werden Nachrichten an alle übergeordneten Scopes gesendet, während mit $broadcast Nachrichten an alle untergeordneten Scopes gesendet werden. Um auf Nachrichten zu reagieren müssen Controller oder Services mit $on Methode registriert werden.

## Routen

Unter Routen versteht man, dass festlegen der Zuordnung von URLs zu spezifischen Ansichten. Angular JS stellt hierfür das ngRoute-Modul zu Verfügung, hiermit können innerhalb der globalen HTML Vorlage verschiedene Ansichten dynamisch in ein Element mit einer ngView Direktive nachgeladen werden.

Angular 8.2

# Unterschied zwischen Angular 8.2 und Angular JS:

Angular wurde von Grund auf neu geschrieben und hat dementsprechend viele Unterschiede. Hier einige Beispiele:

* Die neueste Version von Angular 8.2 kennt keine Scopes oder Controller, es wird eine Hierarchie von Komponenten als zentrales Architekturkonzept verwendet.
* In Angular 8.2 wird die Syntax vereinfacht, mit [] Klammern werden Bindings für Eigenschaften und mit () Klammern werden Binding für Events erzeugt.
* In der neuen Version haben die mobilen Plattformen eine besondere Priorität (Mobile-First Ansatz).
* Es werden Module verwendet, um die Kernfunktionalität leichter und schneller zu machen.
* Es werden nur moderne Browser unterstützt.
* In der neuen Version wird TypeScript benutzt, dadurch hat man folgende Möglichkeiten:
  + Klassenbasierte objektorientierte Programmierung
  + Statische Typisierung
  + Generics
* TypeScript ist rückwärtskompatibel mit JavaScript.

# Node.Js

Node Js dient der Softwareentwicklung zum Betrieb von Netzwerkanwendungen über eine serverseitige Plattform. Webserver werden gut damit realisiert. Node Js wird in der JavaScript Laufzeitumgebung ausgeführt, es bietet eine ressourcensparende Architektur, dadurch wird eine große gleichzeitige Anzahl an Netzwerkverbindungen ermöglicht. Das bietet den Vorteil das pro bestehende Verbindung weniger Arbeitsspeicher verwendet wird. Die Performance wurde bei Node Js mit nonblocking I/O verbessert dadurch wird der Prozessor nicht mit langsam ablaufenden Operationen verlangsamt.

# Typescript

Typescript ist eine von Microsoft entwickelte Programmiersprache, die auf auf den ECMAScript6-Standard basiert. In Typescript übernahm Klassen, Vererbungen, Module und anonyme Funktionen von Ecmascript6. JavaScript wird als gültiger Typescript code erkannt, dadurch kann man JavaScript Bibliotheken wie jQuery oder Angular JS verwenden. In Typescript wird die Kapselung von Klassen, Interfaces, Funktionen und Variablen unterstützt. Mit Hilfe von Plug-ins kann Typescript in verschiedene Management-Tools integriert werden.

# Bulma

Bulma ist ein freies CSS-Framework, das auf Flexbox basiert. Es hat auf HTML und CSS basierende Button, Formulare, Tabellen, Grid-Systeme, Navigations- und andere Oberflächengestaltungselemente.

# MongoDB

Die Programmierung von MongoDB wurde im Oktober 2007 begonnen von dem Unternehmen 10gen, welches 2013 zu MongoDb.INC umbenannt wurde. Das erste Mal Online war das Programm im Februar 2009. MongoDB ist die meist verbreitete NoSQL Datenbank.

MongoDB ist eine noSQL Datenbank, die mit der Programmiersprache C++ realisiert wurde. Die Datenbank ist dokumentenorientiert geschrieben und kann dadurch JSON Dokumente verwalten.

# Azure

Azure eine Cloud-Plattform von Microsoft. Die Cloud wurde 2008 vorgestellt und 2010 veröffentlicht. Die Plattform funktioniert sowohl auf Linux als auch auf Windows. Die Plattform bietet über 600 Services wie zum Beispiel App-Service, Speicher, Datenbanken, Virtuelle Computer und noch vieles mehr.

Azure ist ein Layer, welcher Windows Server benutzt. Diese laufen mit Windows Server 2008 aufwärts und Hyper-V. Hyper-V ist eine Virtualisierungstechnik für 32(x86)- und 64(x64)-Bit Systeme, die auf Hypervisor basiert. Hypervisor(VMM) ist eine Klasse von Systemen, die es erlaubt eine virtuelle Umgebung zu erstellen, welche Komponenten haben kann, welche sich von der tatsächlichen Hardware unterscheiden kann. Dies ermöglicht Prozesse, welche auf der tatsächlichen Hardware nicht funktionieren würden.

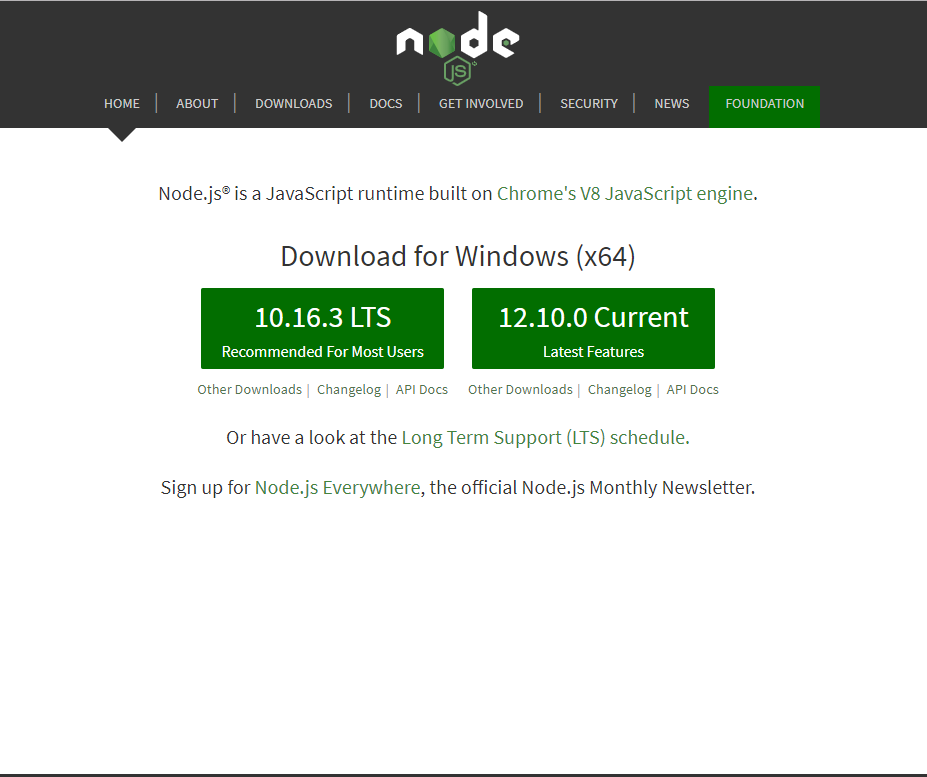
Der Service ist in 54 Regionen verfügbar und Erweiterungen sind noch geplant. Weiteres hat Microsoft auf allen Kontinenten Server und ist damit der erste große Cloud-Provider der in Afrika auch stationiert ist. Für Europa wird ein Server in Nordeuropa(Dublin) und Westeuropa(Amsterdam) angeboten.

# Azure App-Service

Der Azure App-Service ermöglicht es Web-Apps, mobile Back-Ends und REST-APIs in mehreren Programmiersprachen zu erstellen. Der Service erlaubt auch das Hosten von den zuvor angeführten Anwendungen. Es wird sowohl Windows als auch Linux unterstützt. Bereitstellen kann man den Code über GitHub(Git-Repositories), Azure-DevOps, Bitbucket und weiteren anderen Quellen. Die Anwendung kann in .NET, Node.js, PHP, Ruby, Java, Python(nur Linux) oder HTML programmiert werden.

How to install Angular

1. Die Angular Setup Seite öffnen unter folgendem Link: <https://angular.io/guide/setup-local>
2. Für Angular benötigt man Node JS. Um Node JS herunterzuladen folgenden link benutzen und die Empfohlene Version runterladen.



1. Installierung der Angular CLI: Hierfür öffnet man die Konsole(CMD) und tippt folgenden Befehl ein:

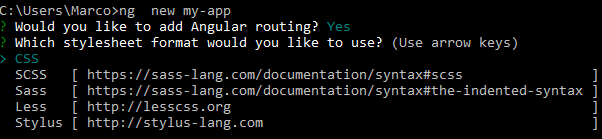


1. Um sich einen Arbeitsplatz zu schaffen und mit der Initialisierung der App zu starten folgenden Befehl in die Konsole tippen:



Bei der Frage ob man ein Routing hinzufügen will Yes (y) antworten.

Bei Stylesheet Format CSS auswählen.



Jetzt ist die App erstellt und bereit gestartet zu werden.

1. Um die App zu starten folgende Befehle eintippen:



1. Die App ist unter  <http://localhost:4200/> zu finden.

# Systemspezifikation

## Blockschaltbild

### Allgemeines Blockschaltbild



Abbildung 1 - Blockschaltbild

Bei diesem Blockschaltbild werden die verschiedenen Interaktionen zwischen Servern und Clients über das Internet beschrieben. Beim Aufruf der Website wird eine Anfrage (Http Request) an den Server geschickt. Diese Anfrage erfolgt asynchron. Die API holt die Daten von der Datenbank des Servers und schickt sie zurück an den Client. Diese Daten kommen meist im JSON Format und werden anschließend clientseitig serialisiert.

Anschließend werden die Daten auf der Website abgebildet. Es gibt nur einmal beim ersten Aufrufen der Website einen Datentausch zwischen Client und Webserver. Nachdem die Website dargestellt wurde, ist es dem Nutzer erlaubt, die Funktionen der Website zu nutzen.

## Systemüberblick

### Funktionalität der Baugruppen

#### Baugruppe I

<Beschreibung der Baugruppe I>

Azure Appservice erklären

#### Baugruppe II

<Beschreibung der Baugruppe II>

HTML,CSS,JS

## Externe Schnittstellen

### <HTTP>

HTTP oder auch Hypertext Transfer Protocol ist ein Kommunikationsprotokoll im Internet. Die Aufgabe dieses Protokolls ist, Dateien vom Webserver anzufordern und in den Browser zu laden. Der Browser verarbeitet die empfangenen Dateien und interpretiert den Inhalt.

HTTP sendet mit der ,,http://´´ einen Request an den Server. Der Server schickt eine Rückmeldung, diese beinhaltet einen Statuscode in der ersten Zeile. Dieser Code ist dreistellig und enthält Informationen über die Verfügbarkeit der Daten.

Beispiel: Nachdem man die URL im Browser eingegeben hat, wird die Anfrage an den Server geschickt. Darauf bekommt man eine Rückmeldung zurück. Beinhaltet sie in der ersten Zeile den Code 404, würde dies bedeuten, dass die Dateien nicht gefunden werden können.

### <API>

Application Programming Interface oder kurzgesagt API ist eine Programmierschnittstelle, die von verschiedenen Unternehmen zur Verfügung gestellt wird, um sich an ihr System Anbinden zu können. Eine Schnittstelle ist eine fix definierte Datenstruktur, über die man eine Anwendung ansprechen kann. Die Anwendung stellt ein Set von Funktionen (Methoden) zur Verfügung und gibt exakt vor, welche Eingaben sie erwartet und welche Ausgabe sie liefert. Diese Definition von Ein- und Ausgabe nennt man auch Spezifikation. Was genau die Anwendung im Hintergrund tut ist in einer Spezifikation nicht beschrieben und ist für den Nutzer der Schnittstelle (Consumer) irrelevant. Bei unserer Website benutzen wir die API, um die Daten vom Server des Streamingdienstes zu bekommen.

# Use Cases (Testfälle)

In diesem Kapitel werden verschiedene Use Cases als Testfälle der Webseite www.w2w.azurewebsites.net beschrieben und analysiert. Funktionen, die der Anwender verwenden bzw. ausführen kann, werden hier beschrieben und durch einen Testvorgang getestet. In dieser Dokumentation werden den Funktionen verschiedene Ausführungsnamen zugeordnet um sich von anderen Testfällen unterscheiden zu können. Die Ergebnisse werden bewertet und dokumentiert.

## Use Case „Aufruf ohne Internetverbindung“

### Randbedingung

Der Anwender darf nicht mit einer funktionierenden Internetverbindung verbunden sein, zusätzlich wird ein vollständig installierter Webbrowser benötigt.

### Testablauf

Die Website [www.w2w.azurewebsites.net](http://www.w2w.azurewebsites.net) wird ohne Internetverbindung geöffnet.

### Erwartetes Ergebnis

Das Laden der Website schlägt fehl. Dies wird durch eine Fehlermeldung des Browsers gekennzeichnet.





## Use Case „Aufruf mit Internetverbindung”

Erklärung was der zweite Anwendungsfall für eine Funktionalität bietet.

# Benutzerhandbuch

Benutzerhandbuch

# Benutzerhandbuch Benutzeroberfläche

## Zugang zur Website

Alle Benutzer, die diese Website ([www.w2w.azurewebsites.net](http://www.w2w.azurewebsites.net)) aufrufen, sollen uneingeschränkten Zugang zu allen Informationen haben, die unser Projekt zu Verfügung stellt. Deswegen ist auf der Startseite eine Übersicht aller Streaming-Dienst zu sehen, von denen wir die notwendigen Zugriffsberechtigungen haben, um dem Benutzer selbst entscheiden zu lassen auf welchem Streaming-Dienst er gerne eine/n Serie/Filme suchen will.

Es ist entscheidend auf welchem Gerät die Website aufgerufen wird. Die Mobile Ansicht ist viel kompakter, da bestimmte Funktionen nur auf der Desktop Ansicht verfügbar ist.

# Desktop Ansicht

## Navigations-Bar

Durch Klicken auf die jeweiligen Labels wechselt der Benutzer zwischen den einzelnen Streaming-Anbieter. Dadurch soll die Suche für die Benutzer vereinfacht werden, zudem wird dadurch die Chance erhöht, dass der Benutzer auf seiner Suche fündig wird.

## Serien/Filme Suche

Durch Klicken auf die jeweiligen Serien/Filme -Banner soll der Benutzer direkt auf die jeweilige Streaming-Dienst-Seiter weitergeleitet werden. Dort angekommen wird er aufgefordert sich für die jeweilige Streaming anzumelden, sofern er noch nicht eingeloggt ist. Durch das Weiterleiten soll dem Benutzer wieder einmal die Suche vereinfacht werden.

## Bestimmte Suche

Der Button „Search“ neben den Streaming Diensten soll dem Anwender helfen nach einer/einem bestimmter/bestimmten Film zu suchen. Dabei gibt der Benutzer den Namen des/der gewünschten Filmes/Serie ein. Nach einem gefundenen Treffer kann der Benutzer dann selbst entscheiden, ob er zum gewünschten Titel weitergeleitet werden soll.

# Mobile Ansicht

## Navigations-Bar

Durchs Drücken auf den Menü-Button öffnet der Benutzer ein Burger-Menü, das die einzelnen Streaming-Anbieter anzeigt. Dadurch soll die Suche für den Benutzer vereinfacht werden, zudem wird dadurch die Chance erhöht, dass der Benutzer bei der Suche fündig wird. Im Burger-Menü werden die jeweiligen Streaming-Anbieter aufgelistet.

## Serien/Filme Suche

Durch Drücken auf die jeweiligen Serien/Filme -Banner soll der Benutzer direkt auf die jeweilige Streaming-Dienst-Apps weitergeleitet werden. Wenn die jeweilige Streaming-App noch nicht runtergeladen ist, wird er auf den App-Store des installierten Betriebssystems weitergeleitet. Wenn die Applikation dann heruntergeladen ist, wird er aufgefordert sich für die jeweilige Streaming-Seite anzumelden. Durch das Weiterleiten soll dem Benutzer wieder einmal die Suche vereinfacht werden.

## Bestimmte Suche

Der Button „Search“ unter den Streaming Diensten soll dem Anwender helfen nach einer/einem bestimmter/bestimmten Film zu suchen. Dabei gibt der Benutzer den Namen des/der gewünschten Filmes/Serie ein. Nach einem gefundenen Treffer kann der Benutzer dann selbst entscheiden, ob er zum gewünschten Titel weitergeleitet werden soll.

# Webseitendesign/ -programmierung

## Startseite

## Diese Seite kann unter dem Link <https://W2W.azurewebsite.com> gefunden werden. Dieser Link ist eine externe Zugriffsmöglichkeit auf die Hauptseite unserer Website. Die Startseite unserer Website wird ausfolgenden Dateien gebildet: index.html, bulma.css und bulma.js.

### Einleitung

Sobald ein User den obigen Link im World Wide Web aufruft, erscheint die Startseite unserer Diplomarbeit. Der User kann über die Startseite auf die Unterseiten zugreifen. Die Unterseiten sind die vom User ausgewählten Streamingdienst Seiten. Auf diesen Seiten kommen nur die Informationen von der API für den gewählte Streamingdienst.

### Head

Der Titel ist auf „W2W“ gesetzt. Das CSS-Stylesheet

1. Navigationsleiste

Window-Width-min: 1024px  
Scroll-Position: 0px



Window-Width-min: 1024px  
Scroll-Position: über 100px

Window-Width-max-: 1023px Window-Width-max-: 1023px   
Scroll-Position: 0px Scroll-Position: über 100px



<!--Navbar-->

<nav id="nav-moving" class="navbar" role="navigation"   
aria-label="main navigation">

    <div class="navbar-brand">

        <a  class="navbar item"

routerLink="" (click)="openHome()">

           <img id="w2w-logo"

src="assets/W2W\_Logo\_V2.png"

width="80" height="30"/>

        </a>

        <a  role="button" class="navbar-burger burger"

aria-label="menu" aria-expanded="false"

data-target="top-nav" style="color: white">

            <span aria-hidden="true"></span>

            <span aria-hidden="true"></span>

            <span aria-hidden="true"></span>

        </a>

    </div>

    <div id="top-nav" class="navbar-menu">

        <div class="navbar-start">

            <a id="home" class="navbar-item item active-link"

routerLink="" (click)="openPage($event)">

                Home

                <div class="underline"></div>

            </a>

            <a  id="netflix" class="navbar-item item"

routerLink="netflix"

(click)="openPage($event)">

                Netflix

               <div class="underline"></div>

            </a>

            <a  id="prime" class="navbar-item item"

routerLink="prime"

(click)="openPage($event)">

                Prime

               <div class="underline"></div>

            </a>

            <a  id="disney" class="navbar-item item"

routerLink="disney+"

(click)="openPage($event)">

                Disney+

              <div class="underline"></div>

            </a>

            <a  id="skyx" class="navbar-item item"

routerLink="skyx"

(click)="openPage($event)">

                Sky X

              <div class="underline"></div>

            </a>

            <div class="navbar-item has-dropdown1 is-

hoverable">

                <a class="navbar-link">

                    More

                </a>

                <div class="navbar-dropdown">

                    <a class="navbar-item item"

(click)="openPage($event)">

                        About

                    </a>

                    <a class="navbar-item item"

(click)="openPage($event)">

                        Jobs

                    </a>

                    <a class="navbar-item item"

(click)="openPage($event)">

                        Contact

                    </a>

                    <hr class="navbar-divider">

                    <a class="navbar-item item"

routerLink="contact"

(click)="openPage($event);">

                        Report an issue

                    </a>

                </div><!--Ende More Dropdown-->

            </div><!--Ende More-->

        </div><!--Ende linke Seite-->

        <div class="navbar-end">

            <div class="navbar-item">

                <form id="search">

                    <input type="search" placeholder="Search>

                </form>

            </div>

        </div><!--Ende rechte Seite-->

    </div><!--Ende Nav-Menü-->

</nav>

HTML

|  |  |
| --- | --- |
| role="navigation" | Das Role Attribut dient als Orientierungspunkt für den Browser. Dadurch weiß der Browser beim Laden der Website wo der Navigationsbereich ist. |
| aria-label= "main navigation" | Das Aria-Label definiert einen String zum Benennen des Navigationsbereichs in der Mobilen Ansicht. |
| routerLink="" | RouterLink dient zum internen Verbinden der Komponenten/Unterseiten. In HTML auch als „Href“ bekannt. |
| role="button" | Zeigt dem Browser beim Laden der Website wo der Button ist. |

CSS

|  |  |
| --- | --- |
| .navbar | Der Navigationsbereich wird erstellt und an einer Position fixiert. |
| .is-fixed-top | Macht die Position des jeweiligen Objekts Absolut |
| #nav-moving | Erlaubt uns die Farbe des Navigationsbereichs zu bestimmen. |
| .navbar-brand | Dient zur Einrichtung der Eigenschaften des Logos. |
| .navbar-item | Dient zur Anpassung der Navigationsleiste. Hiermit wird definiert was in den einzelnen Elementen des Navigationsbereichs steht. |
| #w2w Logo | Lässt uns die Breite des Logos definieren. |
| .navbar-burger burger | Klasse zum Definieren der Eigenschaften des Burger-menüs. |
| #navi | Dient für die Eigenschaften, die den Unterstrich bei den einzelnen Wörtern im Navigationsmenü definieren. |
| .underline | Der eigentliche Unterstrich unter den Wörtern. |
| #top-nav | Wird zum toggeln verwendet, dient zum Öffnen und Schließen des Burger-menüs. |
| .navbar-start | Bestimmt an welcher Stelle der Navigationsleiste ein Element steht. |
| .navbar-menu | Klasse zum Verstecken der Navigationsleiste in der mobilen Ansicht. Wird diese Klasse „is active“ hinzugefügt erscheint die Navigationsleiste. |
| .has-dropdown1 | Diese Klasse wird dazu verwendet, um aus einem normalen Element der Navigationsleiste ein Element mit einem Dropdown Menü zu machen. |
| .is-hoverable | Hoverable zeigt ,dass sich das Menü öffnet, wenn man nur darüberstreicht. |
| .navbar-dropdown | Diese Klasse ist die Umgebung der Items im Dropdown Menü. |
| .navbar-divider | Durch diese Klasse wird ein Strich zum Separieren der Items im Dropdown Menü erstellt. |
| .navbar-end | Lässt die Items der Navigationsleiste am Ende beginnen. |

Javascript/Typescript

|  |  |
| --- | --- |
| (click)="()" | Ist ein DOM-Event mit dem beim Anklicken eine Funktion aufgerufen wird. |
| openHome() | Die Funktion sorgt dafür, dass beim Anklicken des Logos „Home“ unterstrichen wird. |
| openPage($event) | Die Funktion, die dafür sorgt, dass das aktive Element gewechselt wird und mit „underline“ unterstrichen wird.  Als Übergabeparameter haben wir das das Element, welches angeklickt wurde. |
| changeNavbar() | Diese Funktion fügt der Klasse „navbar“, zu unterschiedlichen Zeitpunkten abhängig von der Größe des Fensters, die Klasse „is-fixed-top“ hinzu.  Die Farbe wird auch abhängig von der Größe und der Position geändert. |
| closeNavbarMenu() | Diese Funktion ermöglicht das Schließen des Menüs in der mobilen Ansicht bzw. des Dropdown Menüs |
| toggleNavbarMenu() | Mit dieser Funktion werden das Öffnen und Schließen des Menüs ermöglicht. Weiters wird auch der Burger damit zu einem „X“ beim Öffnen des Menüs. |
| toggleDropDown() | Mit dieser Funktion werden das Öffnen und Schließen des Dropdown-Menüs ermöglicht. |

1. Billboard



<!--Swiper-Billboard-->

<div class="billboard swiper-container">

    <div class="swiper-wrapper">

        <div class="swiper-slide">

<img class="parallax-bg"

src="assets/recommend/4ee860ca-3b62-

43fd-b47d-1f45d2649dd8.\_UR3000,

600\_SX3000\_FMwebp\_.jpg">

<div class="center-text">

<div class="title"

data-swiper-parallax="-300">

<h1>

W2W

</h1>

</div>

<div class="subtitle"

data-swiper-parallax="-200">

Who we are

</div>

<div class="text"

data-swiper-parallax="-100">

<p>Want to know what’s new or catch up on

thing before they leave your Streaming

Service, then welcome to <br>

<strong>W2W</strong>

</p>

</div>

</div>

        </div>

        <div class="swiper-slide">

                <img src="assets/recommend/099fa944-e12f-

4432-b0a8-0fca78ca9be8.\_UR3000,

600\_SX3000\_FMwebp\_.jpg">

        </div>

        <div class="swiper-slide">

                <img src="assets/recommend/9704c6fc-5ec7-

4aad-bfc6-df34922fbb73.\_UR3000,

600\_SX3000\_FMwebp\_.jpg">

        </div>

        <div class="swiper-slide">

                <img src="assets/recommend/837a3cdd-ac71-

4bb0-81b3-b1f70ac4ddaa.\_UR3000,

600\_SX3000\_FMwebp\_.jpg">

        </div>

    </div>

<!--Ende von Swiper-->

    <!-- Add Pagination -->

    <div class="swiper-pagination swiper-pagination-white">

</div>

    <!-- Add Navigation -->

    <div id="start-bt" class="swiper-button-prev

swiper-button-white">

</div>

<div id="start-bt" class="swiper-button-next

swiper-button-white">

</div>

</div>

<!--Ende von Swiper-Billboard-->

HTML

|  |  |
| --- | --- |
| data-swiper-parallax |  |

CSS

|  |  |
| --- | --- |
| .billboard | Bestimmt den Container und dessen Eigenschaften |
| .swiper-container | Swiper-Container gibt an wie groß der Bereich ist, in dem alle Teile der Swiper Anwendung drinnen liegen. |
| .swiper-wrapper | Swiper-Wrapper bestimmt den Bereich, in dem alle Bilder drinnen sind. |
| .swiper-slide | Swiper-Slide bestimmt den Bereich, in dem ein einzelnes Bild drinnen ist. |
| .parallax-bg | Parallax-bg bestimmt die Eigenschaften der Bilder. |
| .center-text | Der Text bekommt um ihn vertikal zu zentrieren. |
| .swiper-pagination | Diese Klasse bestimmt die Position und die Eigenschaften der Bullets. Hier wird auch definiert wie viele Slides vorhanden sind. |
| .swiper-pagination-white | Swiper-Pagination-White macht die Bullets für die Slides weiß. |
| #start-bt | Bestimmt die Transparenz der Buttons die wie Pfeile ausschauen und zum swipen vorhanden sind. |
| .swiper-button-prev | Mit dieser Klasse werden die Eigenschaften und die Position für den „vorheriger“ Pfeil bestimmt. |
| .swiper-button-next | Mit dieser Klasse werden die Eigenschaften und die Position für den „nächster“ Pfeil bestimmt |
| .swiper-button-white | Die Klasse wird verwendet, um den „Next Pfeil/Prev“ weiß zu machen. |

Javascript/Typescript

|  |  |
| --- | --- |
| swiperMobile() | Bestimmt die Details des Swipers bis zur Fensterbreite 1027px. |
| swiperDesktop() | Bestimmt die Details des Swipers zwischen der Fensterbreite 1027px und 1600px. |
| swiperTV() | Bestimmt die Details des Swipers ab der Fensterbreite 1600px. |
| swiperInit() | Initialisiert den Swiper |

**Home**

<app-billboard></app-billboard>

<div (window:resize)="initSwiperOnResize($event)"

style="color: black;">

    <div class="tile is-ancestor">

        <div class="tile is-12 is-vertical is-parent">

            <app-netflix-home></app-netflix-home>

            <app-prime-home></app-prime-home>

            <app-disney-home></app-disney-home>

            <app-sky-home></app-sky-home>

        </div>

    </div>

</div>

Javascript/Typescript

|  |  |
| --- | --- |
| (window:resize) | Ist ein Event, welches getriggert wird, wenn man den Browser verkleinert bzw. vergrößert. |
| initSwiperOnResize($event) | Die Funktion, die aufgerufen wird. Sie soll die Swiper neu initialisieren und die Größe der Container verändern. |
| mobileSwiper() | Bestimmt die Details der Swiper von „Coming Soon“ und „Leaving Soon“ bis zur Fensterbreite 1027px. |
| tabletSwiper() | Bestimmt die Details der Swiper von „Coming Soon“ und „Leaving Soon“ in der vertikalen Orientierung zwischen der Fensterbreite 1027px und 1440px . |
| desktopMonitorSwiper() | Bestimmt die Details der Swiper von „Coming Soon“ und „Leaving Soon“ in der horizontalen Orientierung zwischen der Fensterbreite 1027px und 2201px. |
| tvSwiper() | Bestimmt die Details der Swiper von „Coming Soon“ und „Leaving Soon“ bis ab der Fensterbreite 2201px. |
| swiperInit() | Initialisiert die Swiper |
| setSwiperContainerHeight() | Bestimmt die Höhe aller auftauchenden Swiper-Container auf der Homepage |

1. NETFLIX-HOME

<!--Beginn von Netflix-->

<div class="tile is-child">

    <div class="columns box"

style="background-color:#141414;">

        <!--Beginn von Netflix Logo-->

<div class="column full-half-third">

            <div class="center-pic">

                <img id="netflix-logo"

src="./assets/

640px-Netflix\_2015\_logo.svg.png"/>

            </div>

        </div>

        <!--Ende von Netflix Logo Column-->

<div class="column full-half-2third">

            <div id="flex-col" class="tabs is-centered

is-boxed">

                <ul>

                    <li  class="netflixtab is-active"

(click)="openTab($event,

'netflix-cs','netflix-tab',

'netflixtab')">

<a>Recently Released</a>

</li>

                    <li  class="netflixtab"

(click)="openTab($event,

'netflix-leaving','netflix-tab',

'netflixtab')">

<a>Leaving Soon</a>

</li>

                </ul>

            </div>

            <div class="container">

                <!--Beginn von Coming Netflix-->

                <div  id="netflix-cs" class="netflix-tab">

COMING-SOON SWIPER

</div>

<!--Ende von Coming Netflix-->

<!--Beginn von Leaving Netflix-->

<div  id="netflix-leaving"

class="netflix-tab"

style="display: none;">

LEAVING-SOON SWIPER

</div>

<!--Ende von Leaving Netflix-->

</div>

<!--Ende vom Container-->

</div>

<!—Ende von Swiper Column🡪

</div>

<!--Ende von Gapless column-->

</div>

<!--Ende von Netflix-->

|  |  |
| --- | --- |
| .tile | Tile wird verwendet um ein 2-dimensionales Bulma Layout zu kreieren. |
| .is-ancestor | Die Klasse is-ancestor wird verwendet, um alle anderen Parent Tiles aneinander zu binden. |
| .is-12 | Mit is-12 wird die Breite vom jeweiligen Tile definiert. |
| .is-vertical | Is-vertical wird verwendet, um die Tiles Vertikal zu setzen. |
| .is-parent | Tile is-parent definiert die zweite Schicht des Layouts. |
| .is-child | Diese Klasse ist die dritte Schicht des Layouts und wird verwendet, um den Code darzustellen. |
| .columns box | Mit dieser Klasse werden die Abstände der Boxen entfernt. |
| . column |  |
| # netflix-logo |  |
| # flex-col |  |
| . tabs is-centered is-boxed |  |
| .netflixtab is-active |  |
| # netflix-cs |  |
| . netflix-tab |  |
| # direction-row-col |  |
| . swiper-slide TV |  |
| .verti-pos |  |
| . swiper-pagination down-slide |  |
| # btnext-verti |  |
| . netflix-leaving |  |
| style="display: none;" |  |
| .leaving-verti-pos |  |

Javascript/Typescript

|  |  |
| --- | --- |
| (click)="()" | Ist ein DOM-Event, mit dem beim Anklicken eine Funktion aufgerufen wird. |
| openTab() | Die Funktion sorgt dafür, dass beim Anklicken eines der 2 Tabs, der jeweilige Swiper angezeigt wird und der andere versteckt wird. |
| $event | Ist ein Parameter, mit dem man Informationen über das angeklickte Element bekommt.  Damit wird dem angeklickten Element die Klasse „is-active“ hinzugefügt. |
| netflix-cs | Ist dafür zuständig, dass der „Coming-Soon“-Tab angezeigt wird. Der Parameter hebt das „display: none;“ von „netflix-tab“ auf. |
| netflix-leaving | Ist dafür zuständig, dass der „Leaving-Soon“-Tab angezeigt wird. Der Parameter hebt das „display: none;“ von „netflix-tab“ auf. |
| netflix-tab | Sorgt dafür, dass alle Tabs nicht angezeigt werden in dem er deren Style auf „display: none;“ setzt. |
| netflixtab | Der Parameter, der dafür sorgt, das alle Tabs die Klasse „is-active“ verlieren. |

## Disney-Home

<!--Beginn von Disney+-->

<div class="tile is-child">

    <div class="columns box"

style="background-color: #131313;">

        <!--Beginn von Disney Logo-->

        <div class="column full-half-third">

            <div class="center-pic">

                <img id="disney-logo"

src="./assets/

640px-Disney+\_logo.svg.png"/>

            </div>

        </div>

        <!--Ende von Disney Logo-->

        <div class="column full-half-2third">

            <div id="flex-col" class="tabs is-centered

is-boxed">

                <ul>

                    <li class="disneytab is-active"

(click)="openTab($event,'disney-cs',

'disney-tab','disneytab')">

<a>Coming Soon</a>

</li>

                    <li class="disneytab"

(click)="openTab($event,

'disney-leaving','disney-tab',

'disneytab')">

<a>Leaving Soon</a>

</li>

                </ul>

            </div>

            <div class="container">

<!--Beginn von Coming Disney-->

<div  id="disney-cs" class="disney-tab">

COMING-SOON SWIPER

</div>

<!--Ende von Coming Disney-->

<!--Beginn von Leaving Disney-->

<div  id="disney-leaving" class="disney-tab"

style="display: none;">

LEAVING-SOON SWIPER

</div>

<!--Ende von Leaving Disney-->

</div>

            <!--Ende vom Container-->

</div>

<!--Ende von Swiper Column-->

</div>

<!--Ende von Gapless column-->

</div>

<!--Ende von Disney+-->

Javascript/Typescript

|  |  |
| --- | --- |
| (click)="()" | Ist ein DOM-Event, mit dem beim Anklicken eine Funktion aufgerufen wird. |
| openTab() | Die Funktion sorgt dafür, dass beim Anklicken eines der 2 Tabs, der jeweilige Swiper angezeigt wird und der andere versteckt wird. |
| $event | Ist ein Parameter, mit dem man Informationen über das angeklickte Element bekommt.  Damit wird dem angeklickten Element die Klasse „is-active“ hinzugefügt. |
| disney-cs | Ist dafür zuständig, dass der „Coming-Soon“-Tab angezeigt wird. Der Parameter hebt das „display: none;“ von „disney-tab“ auf. |
| disney-leaving | Ist dafür zuständig, dass der „Leaving-Soon“-Tab angezeigt wird. Der Parameter hebt das „display: none;“ von „disney-tab“ auf. |
| disney-tab | Sorgt dafür, dass alle Tabs nicht angezeigt werden in dem er deren Style auf „display: none;“ setzt. |
| disneytab | Der Parameter, der dafür sorgt, das alle Tabs die Klasse „is-active“ verlieren. |

## PRIME-HOME

<!--Beginn von Prime-->

<div class="tile is-child">

    <div class="columns box"

style="background-color: #0f171e;">

        <!--Beginn von Mobile Logo-->

        <div class="column full-half-third

is-hidden-desktop-monitor">

            <div class="center-pic">

                <img id="prime-logo"

src="./assets/Prime\_Video1.png"/>

            </div>

        </div>

        <!--Ende von Mobile Logo Column-->

        <div class="column full-half-2third">

            <div id="flex-col" class="tabs is-centered

is-boxed">

                <ul>

                    <li class="primetab is-active"

(click)="openTab($event,'prime-cs',

'prime-tab','primetab')">

<a>Coming Soon</a>

</li>

<li class="primetab"

(click)="openTab($event,

'prime-leaving','prime-tab',

'primetab')">

<a>Leaving Soon</a>

</li>

                </ul>

            </div>

            <div class="container">

                <!--Beginn von Coming Prime-->

                <div id="prime-cs" class="prime-tab">

COMING-SOON SWIPER

                </div>

                <!--Ende von Comning Prime-->

                <!--Beginn von Leaving Prime-->

                <div id="prime-leaving" class="prime-tab"

style="display: none;">

LEAVING-SOON SWIPER

                </div>

                <!--Ende von Leaving Prime-->

            </div>

            <!--Ende von Container-->

        </div>

        <!--Ende von Swiper Column-->

        <!--Beginn von Desktop Prime Logo-->

        <div class="column full-half-third

is-hidden-touch-fernseher

is-hidden-touch-landscape">

            <div class="center-pic">

                <img id="prime-logo"

src="./assets/Prime\_Video1.png"/>

            </div>

        </div>

        <!--Ende von Desktop Prime Logo Column-->

    </div>

    <!--Ende von Gapless column-->

</div>

<!--Ende von Prime-->

Javascript/Typescript

|  |  |
| --- | --- |
| (click)="()" | Ist ein DOM-Event, mit dem beim Anklicken eine Funktion aufgerufen wird. |
| openTab() | Die Funktion sorgt dafür, dass beim Anklicken eines der 2 Tabs, der jeweilige Swiper angezeigt wird und der andere versteckt wird. |
| $event | Ist ein Parameter, mit dem man Informationen über das angeklickte Element bekommt.  Damit wird dem angeklickten Element die Klasse „is-active“ hinzugefügt. |
| prime-cs | Ist dafür zuständig, dass der „Coming-Soon“-Tab angezeigt wird. Der Parameter hebt das „display: none;“ von „prime-tab“ auf. |
| prime-leaving | Ist dafür zuständig, dass der „Leaving-Soon“-Tab angezeigt wird. Der Parameter hebt das „display: none;“ von „prime-tab“ auf. |
| prime-tab | Sorgt dafür, dass alle Tabs nicht angezeigt werden in dem er deren Style auf „display: none;“ setzt. |
| primetab | Der Parameter, der dafür sorgt, das alle Tabs die Klasse „is-active“ verlieren. |

## SKY-HOME

<!--Beginn von Sky-->

<div class="tile is-child">

    <div class="columns box"

style="background-color: #10599E;">

        <!--Beginn von Mobile Logo-->

        <div class="column full-half-third

is-hidden-desktop-monitor">

            <div class="center-pic">

                <img id="sky-logo"

src="./assets/skyq-nav.svg"/>

            </div>

        </div>

        <!--Ende von Mobile Logo Column-->

        <div class="column full-half-2third">

            <div  id="flex-col"

class="tabs is-centered is-boxed">

                <ul>

                    <li class="skytab is-active"  (click)="openTab($event,'sky-cs',

'sky-tab','skytab')">

<a>Coming Soon</a>

</li>

<li class="skytab"

(click)="openTab($event,

'sky-leaving','sky-tab','skytab')">

<a>Leaving Soon</a></li>

                </ul>

            </div>

            <div class="container">

                <!--Beginn von Coming Sky-->

                <div id="sky-cs" class="sky-tab">

COMING-SOON SWIPER

                </div>

                <!--Ende von Coming Sky-->

                <!--Beginn von Leaving Sky-->

                <div id="sky-leaving" class="sky-tab"

style="display: none;">

LEAVING-SOON SWIPER

                </div>

                <!--Ende von Leaving Sky-->

            </div>

            <!--Ende vom Container-->

        </div>

        <!--Ende von Swiper Sky-->

        <!--Beginn von Desktop Sky Logo-->

        <div class="column full-half-third

is-hidden-touch-fernseher

is-hidden-touch-landscape">

            <div class="center-pic">

                <img id="sky-logo"

src="./assets\skyq-nav.svg"/>

            </div>

        </div>

        <!--Ende von Desktop Sky Logo-->

    </div>

    <!--Ende von Gapless column-->

</div>

<!--Ende von Sky X-->

Javascript/Typescript

|  |  |
| --- | --- |
| (click)="()" | Ist ein DOM-Event, mit dem beim Anklicken eine Funktion aufgerufen wird. |
| openTab() | Die Funktion sorgt dafür, dass beim Anklicken eines der 2 Tabs, der jeweilige Swiper angezeigt wird und der andere versteckt wird. |
| $event | Ist ein Parameter, mit dem man Informationen über das angeklickte Element bekommt.  Damit wird dem angeklickten Element die Klasse „is-active“ hinzugefügt. |
| sky-cs | Ist dafür zuständig, dass der „Coming-Soon“-Tab angezeigt wird. Der Parameter hebt das „display: none;“ von „sky-tab“ auf. |
| sky-leaving | Ist dafür zuständig, dass der „Leaving-Soon“-Tab angezeigt wird. Der Parameter hebt das „display: none;“ von „sky-tab“ auf. |
| sky-tab | Sorgt dafür, dass alle Tabs nicht angezeigt werden in dem er deren Style auf „display: none;“ setzt. |
| skytab | Der Parameter, der dafür sorgt, das alle Tabs die Klasse „is-active“ verlieren. |

**COMING-SOON Swiper**

<div class="home swiper-container">

<div class="home swiper-wrapper">

<ng-container \*ngFor="let cs of dataValue.ITEMS">

<div class="swiper-slide">

<div class="flip-card-inner">

<div class="flip-card-front">

<img  class="border-picture"

src="{{cs?.image}}" alt="{{cs?.title}}"

style=" height: 100%; width: 100%;"/>

</div>

<div class="flip-card-back">

<h1>Title: <br>{{cs?.title}}</h1>

<p>Watch:

<a href="https://www.netflix.com/title/

{{cs?.netflixid}}">

<p>

<fa-icon [icon]="faPlayCircle">

</fa-icon>

</p>

</a>

</p>

<p>Type: {{cs?.type}}</p>

<p>Rating: <br>{{cs?.rating}}</p>

<p>Added: <br>{{cs?.unogsdate}}</p>

</div>

</div>

</div>

</ng-container>

<!-- Add Pagination -->

<div class="swiper-pagination down-slide swiper-

pagination-white">

</div>

<!-- Add Arrows -->

<div id="btnext-verti" class="swiper-button-next">

</div>

<div id="btprev-verti" class="swiper-button-prev">

</div>

</div>

</div>

Javascript/Typescript

|  |  |
| --- | --- |
| \*ngFor | Ist eine strukturelle Direktive von Angular mit der man innerhalb von HTML die DOM manipulieren kann. Diese Syntax steht für eine For-Schleife.“\*“ Präfix wandelt intern die Attribute in HTML-Tags um. |
| let cs of dataValue.ITEMS | „let cs” deklariert eine Input-Variable, welche nur innerhalb von „ng-container“ gültig ist. “dataValue“ ist das Objekt, welches die Daten der API enthält. “ITEMS“ ist ein Array von dem Objekt Item, welches die einzelnen Informationen der neuen Filme/Serien enthält. |
| {{…}} | Interpolation referenziert einen Ausdruck und dieser wird in HTML-Text umgewandelt. |
| ? | Hindert Angular daran eine Fehlermeldung anzuzeigen, wenn ein Element null oder undefined ist. Dies ist bei asynchronen Aufrufen hilfreich, wenn Daten über eine API kommen. |
| {{cs?.image}} | Zeigt das Bild des Films/der Serie an |
| {{cs?.title}} | Zeigt den Titel des Films/der Serie an |
| {{cs?.netflixid}} | Zeigt die Netflix-ID des Films/der Serie an. Durch das Anklicken des Icons wird man sofort auf die Netflix Seite des Titels weitergeführt. |
| {{cs?.type}} | Zeigt an ob der Titel ein Film oder eine Serie ist |
| {{cs?.rating}} | Zeigt die IMDB-Wertung der Serie/des Films an |
| {{cs?.unogsdate}} | Zeigt das Datum, ab welches die Serie/der Film hinzugefügt wurde, an. |

**Leaving Soon Swiper-Slider**

<div class="home swiper-container">

<div class="home swiper-wrapper">

<ng-container \*ngFor="let ls of expire.ITEMS">

<div class="swiper-slide">

<div class="flip-card-inner">

<div class="flip-card-front">

<img  class="border-picture"

src="{{ls?.image}}"

alt="{{ls?.title}}"

style="height: 100%; width: 100%;"/>

</div>

<div class="flip-card-back">

<h1>Title: <br>{{ls?.title}}</h1>

<p>Watch:

<a href="https://www.netflix.com/title/

{{ls?.netflixid}}">

<p>

<fa-icon [icon]="faPlayCircle">

</fa-icon>

</p>

</a>

</p>

<p>Type: {{ls?.type}}</p>

<p>Rating: <br>{{ls?.rating}}</p>

<p>Leaves: <br>{{ls?.unogsdate}}</p>

</div>

</div>

</div>

</ng-container>

<!-- Add Pagination -->

<div class="swiper-pagination down-slide swiper-

pagination-white">

</div>

<!-- Add Arrows -->

<div id="btnext-verti" class="swiper-button-next">

</div>

<div id="btprev-verti" class="swiper-button-prev">

</div>

</div>

</div>

Javascript/Typescript

|  |  |
| --- | --- |
| \*ngFor | Ist eine strukturelle Direktive von Angular mit der man innerhalb von HTML die DOM manipulieren kann. Diese Syntax steht für eine For-Schleife.“\*“ Präfix wandelt intern die Attribute in HTML-Tags um. |
| let ls of expire.ITEMS | „let ls” deklariert eine Input-Variable, welche nur innerhalb von „ng-container“ gültig ist. “expire“ ist das Objekt, welches die Daten der API enthält. “ITEMS“ ist ein Array von dem Objekt Item, welches die einzelnen Informationen der neuen Filme/Serien enthält. |
| {{…}} | Interpolation referenziert einen Ausdruck und dieser wird in HTML-Text umgewandelt. |
| ? | Hindert Angular daran eine Fehlermeldung anzuzeigen, wenn ein Element null oder undefined ist. Dies ist bei asynchronen Aufrufen hilfreich, wenn Daten über eine API kommen. |
| {{ls?.image}} | Zeigt das Bild des Films/der Serie an |
| {{ls?.title}} | Zeigt den Titel des Films/der Serie an |
| {{ls?.netflixid}} | Zeigt die Netflix-ID des Films/der Serie an. Durch das Anklicken des Icons wird man sofort auf die Netflix Seite des Titels weitergeführt. |
| {{ls?.type}} | Zeigt an ob der Titel ein Film oder eine Serie ist |
| {{ls?.rating}} | Zeigt die IMDB-Wertung der Serie/des Films an |
| {{ls?.unogsdate}} | Zeigt das Datum, ab welches die Serie/der Film hinzugefügt wurde, an. |

**REPORT AN ISSUE**

<section class="hero is-primary is-bold">

<div class="hero-body">

<div class="container">

<h1 class="title">Contact Us!</h1>

</div>

</div>

</section>

<section class="section">

<div class="container">

<!-- form goes here -->

<form (ngSubmit)="processForm()" #contactForm="ngForm">

<div class="field">

<label class="label">Name</label>

<input  type="text" name="name" class="input"

placeholder="Your Name" [(ngModel)]="name"

required minlength="3" #nameInput="ngModel">

<div class="help is-error"

\*ngIf="nameInput.invalid && nameInput.touched">

Name is required and needs to be

at least 3 characters long.

</div>

</div>

<div class="field">

<label class="label">E-mail</label>

<input  type="email" name="email" class="input"

placeholder="Your Email" [(ngModel)]="email"

required email #emailInput="ngModel">

<div class="help is-error"

\*ngIf="emailInput.invalid &&

emailInput.touched">

Needs to be a valid email.

</div>

</div>

<div class="field">

<label class="label">Your Massage</label> <textarea  class="textarea" name="message"

placeholder="What's on your mind?"

[(ngModel)]="message"

required #messageInput="ngModel">

</textarea>

<div class="help is-error"

\*ngIf="messageInput.invalid &&

messageInput.touched">

Your message is required!

</div>

</div>

<button [disabled]="contactForm.invalid"

type="submit" class="button is-danger is-large">

Submit

</button>

</form>

</div>

</section>

Javascript/Typescript

|  |  |
| --- | --- |
| (ngSubmit)="" | Ist ein DOM-Event, mit dem beim Anklicken eines Submit-Buttons eine Funktion aufgerufen wird. |
| processForm() | Diese Funktion soll in Zukunft die Informationen an einen Backend-Server weiterleiten. Derzeit gibt er eine Meldung aus dass die Nachricht erhalten wurde. |
| [(ngModel)]="VARIABLE" | Erlaubt das „Two-Way data binding“ in Angular für Forms und bietet auch weitere Funktionen wie an, wie Fehlererkennung und Erkennung invalider Eingaben. Dadurch kann man Daten zwischen der HTML- und der TS-Datei auszutauschen. Zwischen den Rufzeichen kommt die Variable. Man kann es dem Value in Standard HTML gleichsetzen. |
| email,minlength="Zahl", required | Sind Funktionen der Klasse „Validators“. Dienen der Überprüfung von Form-Elemente. |
| required | Die Elemente, die markiert sind, müssen ausgefüllt sein, ansonsten bleibt der Submit-Knopf deaktiviert. |
| minlength="3", | Der Wert des Elements muss in unserem Code mindestens eine Länge von 3 Zeichen haben. |
| email | Das Element muss eine valide E-Mail Schreibweise enthalten: abcd@gmail.com |
| \*ngIf | Ist eine strukturelle Direktive von Angular mit der man innerhalb von HTML die DOM manipulieren kann. Diese Syntax steht für eine IF-Schleife.“\*“ Präfix wandelt intern die Attribute in HTML-Tags um. |
| #NAME="ngModel" | „#NAME“erlaubt es uns dem HTML-Tag eine Angular Referenzvariable zu zuweisen. Um auf die „Form“-Kontrolle(FormControl) von Angular zu nutzen, wird „ngModel“ als Key übergeben. |
| invalid,touched | Dienen der Überprüfung von Eingaben und sind Funktionen der von Angular zur Verfügung gestellten Klasse „FormControl“ und sind Erweiterungen der Validators. |
| \*ngIf="nameInput.invalid  && nameInput.touched" | Wird die Mindestlänge nicht eingehalten und nicht mehr auf das Element fokussiert, wird eine Fehlermeldung angezeigt. |
| \*ngIf="emailInput.invalid  && emailInput.touched" | Wird das E-Mail-Format nicht eingehalten und nicht mehr auf das Element fokussiert, wird eine Fehlermeldung angezeigt. |
| \*ngIf="messageInput.invalid  && messageInput.touched" | Ist die Box leer und es steht kein Fokus mehr auf das Element, wird eine Fehlermeldung angezeigt. |

**FOOTER**

<footer  class="footer-index is-hidden-touch"

style="border-top:groove white 0.1px;">

<div style="display:table-row;">

<div class="navFooterLinkCol">

<div class="navFooterColHead">

Über W2W

</div>

<ul>

<li style="word-break:break-all;">

Karriere bei uns

</li>

<li>

Über uns

</li>

</ul>

</div>

<div class="foot-space" style="display:table-cell;">

</div>

<div class="navFooterLinkCol">

<div class="navFooterColHead">

Über W2W

</div>

<ul>

<li>

Karriere bei uns

</li>

<li>

Über uns

</li>

</ul>

</div>

<div class="foot-space" style="display:table-cell;">

</div>

<div class="navFooterLinkCol">

<div class="navFooterColHead">

Über W2W

</div>

<ul>

<li>

Karriere bei uns

</li>

<li>

Über uns

</li>

</ul>

</div>

<div class="foot-space" style="display:table-cell;">

</div>

<div class="navFooterLinkCol">

<div class="navFooterColHead">

Über W2W

</div>

<ul>

<li>

Karriere bei uns

</li>

<li>

Über uns

</li>

</ul>

</div>

</div>

</footer>

# Testfallspezifikation

## Testgruppe (Betriebsbereitschaft)

### Testfall <A>

**Randbedingung:**

<Randbedingungen>

**Testablauf:**

<Eingabe(n) / Aktionen>

**Erwartetes Ergebnis:**

<Welche Ausgabe / Aktion / Zustand soll erreicht werden>

### Testfall <B>

**Randbedingung:**

<Randbedingungen>

**Testablauf:**

<Eingabe(n) / Aktionen>

**Erwartetes Ergebnis:**

<Welche Ausgabe / Aktion / Zustand soll erreicht werden>

# Webseitendesign/ -programmierung

## Startseite

Diese Seite kann unter dem Link <https://W2W.azurewebsite.com> gefunden werden. Dieser Link ist eine externe Zugriffsmöglichkeit auf die Hauptseite unserer Website. Die Startseite unserer Website wird ausfolgenden Dateien gebildet: index.html, bulma.css und bulma.js.

### Einleitung

Sobald ein User den obigen Link im World Wide Web aufruft, erscheint die Startseite unserer Diplomarbeit. Der User kann über die Startseite auf die Unterseiten zugreifen. Die Unterseiten sind die vom User ausgewählten Streamingdienst Seiten. Auf diesen Seiten kommen nur die Informationen von der API für den gewählte Streamingdienst.

### Head

#### HTML

Der Titel ist auf „Where 2 Watch“ gesetzt. Das CSS-Stylesheet

### Body

#### MainGrid

##### HTML

##### CSS

***Verwendung von Literaturlisten***

Quelle 28.1.2013:

<http://www.bildungswissenschaft.uni-wuerzburg.de/fileadmin/06030200/_temp_/Hinweise_fuer_die_Verwendung_von_Literatur_20.03.09_01.pdf>

Alle Quellen, die im Text zitiert werden, müssen auch in der Literaturliste zu finden sein. Das Literaturverzeichnis befindet sich am Ende einer wissenschaftlichen Arbeit und ermöglicht dem Leser die genutzten Quellen zu recherchieren. Besonders zu beachten ist die Einheitlichkeit des Literaturverzeichnisses! Für die unterschiedlichen Quellenarten (Monographien/ Aufsätze aus Sammelwerken; Beiträge aus Zeitschriften) muss – trotz der spezifischen Unterschiede – eine einheitliche Grundstruktur gewählt werden.

***Wissenschaftliches Schreiben – Quellenangaben und Zitate***

Quelle 3.12.2012:

<http://www.lai.fu-berlin.de/studium/studienberatung/magister/wiss_arb/Checkliste_zur_Formatierung_wissenschaftlicher_Texte.pdf>

Zu allem was Ihr schreibt, wenn es nicht etwas ist, das wirklich auf eurem eigenen Mist gewachsen ist, muss klar sein, wo es herkommt. Normalerweise wird das *Autor-Datum-*Schema verwendet, d.h. im Text in runden Klammern erscheinen Nachname des Autors und Erscheinungsdatum plus, wenn es sich um ein Zitat handelt, oder sich auf eine bestimmte Seite bezogen wird, Seitenzahl(en). Auf dieses Schema basieren folgende Hinweise. Alternativen zu diesem Schema finden sich im Anschluss. Beispiele:

* (Appadurai 1996) – auf das ganze Buch wird Bezug genommen,
* (Bhabha 1990: 54-55) – eine Idee oder ein Zitat von Seiten 54-55,
* (vgl. Chiappini 2001) – vergleiche mit dem Buch von Chiappini,
* (Anderson 1983 nach Appadurai 1996: 16) – Appadurai hat Anderson auf Seite 16 zitiert und es wird das Zitat von Anderson hier zitiert.

Des Weiteren gilt Folgendes: Einzelnen Wörtern/ Konzepten, die Ihr von anderen Autoren verwendet wird die Referenz direkt angefügt auch wenn der Satz danach weitergeht, so etwa Checkliste zur Formatierung wissenschaftlicher Texte Seite 5 von 9 wenn ihr von „vorgestellten Gemeinschaften“ (Anderson 1983) etwas schreibt, bezieht sich das auf Benedict Andersons Konzept – im Buch *Imagined Communities* aus dem Jahre 1983 zu finden.

Ideen oder Gedankengänge, die von anderen stammen werden am Ende des betreffenden Satzes *vor* dem Punkt referiert (Chiappini 2001: 89). Wenn sich der ganze Absatz an einem (oder mehreren) Autoren orientiert dann steht die Referenz am Ende des Absatzes *nach* dem letzten Punkt. (Bhabha 1990; vgl. Bhabha 1993)

❑ Referenzen überprüfen.

Originalzitate werden „als solche durch Anführungszeichen kenntlich gemacht. Dort wo sie enden muss die Referenz stehen“ (Müller 2004), auch wenn der Satz noch nicht zu Ende ist.

Originalzitate, die länger als drei Zeilen sind, müssen in einer kleineren Schriftgröße und *ohne* Anführungszeichen kenntlich gemacht werden. Das heißt, in diesem Fall bleiben auch alle originalen Anführungszeichen erhalten. Wenn Ihr in Zitaten etwas verändert, dann muss das deutlich werden: Zusätzliche Wörter, etwa grammatikalische Veränderungen, werden durch eckige Klammern kenntlich [gemacht], wenn ihr Wörter auslasst, dann stehen einfache Auslassungspunkte … und wenn ganze Sätze ausgelassen werden, dann Auslassungspunkte in runden Klammern. (…)

Noch mal im Einzelnen: ... drei Punkte für ein oder mehrere ausgelassene Wörter, (...) drei Punkte in runden Klammern, wenn mehr als ein Satz ausgelassen wurde,

(...)

Wenn ganze Absätze ausgelassen werden stehen die Auslassungszeichen in einem eigenen Abschnitt,

[„eingefügtes Wort“ oder „veränderter Buchstabe“] wenn vom Verfasser Veränderungen vorgenommen werden, so stehen diese in einer eckigen Klammer,

[sic] so gibt man an, dass eine Textstelle originalgetreu übernommen wurde, der Verfasser der Arbeit aber von einem Druckfehler der Textstelle ausgeht oder sich von einem verwendeten Wort distanzieren will.

Am Ende des mehrzeiligen Zitates steht die Referenz nach dem Punkt. (Schmidt 1979: 77-78)

Zitate aus fremdsprachiger Literatur sollten, wenn sie im Fließtext stehen, der Verständlichkeit halber übersetzt sein (vor allem keine halben Sätze in einer anderen Sprache). Bisweilen kann es aber notwendig sein ein Zitat im Original stehen zu lassen. Mehrzeilige Zitate müssen nicht übersetzt werden. Allerdings muss in letzteren beiden Fällen eine Übersetzung des Verfassers / der Verfasserin der Arbeit als Fußnote beigefügt und als solche sein. Ist das Zitat bereits im Text übersetzt ist es sinnvoll das Original in einer Fußnote beizufügen.

❑ Zitate überprüfen.

Zu Fußnoten allgemein: Bitte keine Extraformatierungen. Die Fußnotenreferenzen sollten direkt hinter dem Wort[[1]](#footnote-1) (ohne Leerzeichen) oder hinter dem Satz stehen.[[2]](#footnote-2) Bei Anführungszeichen „nach den Anführungszeichen.“[[3]](#footnote-3)

❑ Fußnoten überprüfen.

❑ Informationen, die aus einer eigenen Feldforschung hervorgehen, müssen entsprechend kenntlich sein. Ihr könnt allerdings auch am Anfang des betreffenden Abschnittes eine Checkliste zur Formatierung wissenschaftlicher Texte Seite 6 von 9 Fußnote anfügen wo ihr kurz erwähnt, dass alle Infos, die nicht extra gekennzeichnet sind, von Eurer Forschung im August 2006 etc. stammen.

❑ Interviews müssen mit Namen, Datum und Ort entweder separat in der Bibliographie aufgeführt werden (+ entsprechende Referenzen im Text) oder als Fußnote. Wurde der Name geändert muss dies kenntlich gemacht werden. (Achtung – generell gilt immer: Informantenschutz!)

Wem das *Autor-Datum-*Schema nicht gefällt, der kann alternativ auch die kompletten Referenzen in die Fußnoten packen. Dann steht in der Regel in der Fußnote, in der ein Buch zum ersten Mal verwendet wird, die komplette Referenz, in allen weiteren wird mit *op. cit*. darauf hingewiesen, dass die Referenz weiter vorne zu suchen ist. Auch sind die Abkürzungen ebd. (*ibidem*) und ders./dies. (*idem*) hilfreich wenn ein Werk/ Autor mehrmals hintereinander zitiert wird.

# Abbildungsverzeichnis

[Abbildung 1 - Blockschaltbild 13](#_Toc2763959)

# Begleitprotokolle

In einem Begleitprotokoll sind der Arbeitsablauf (zeitliche Auflistung, wann und wie lange an der abschließenden Arbeit gearbeitet wurde) sowie die verwendeten Hilfsmittel und Hilfestellungen zu dokumentieren. **Jedes Teammitglied** ist verpflichtet, selbstständig sein **eigenes Begleitprotokoll** zu führen. Das Begleitprotokoll ist der schriftlichen Arbeit beizulegen (§ 9 Abs. 2 Prüfungsordnung BMHS).

In der Rubrik Erstellung finden Sie eine Begleitprotokoll-Vorlage sowie Erläuterungen zum Begleitprotokoll. Sprechen Sie aber mit Ihrem Betreuer/Ihrer Betreuerin, ob Sie dieses Begleitprotokoll als Vorlage verwenden können.

Quelle: <http://www.diplomarbeiten-bbs.at/faq/faq-schuelerinnen>

Im Begleitprotokoll, das als Nachweis von Tätigkeiten, Meetings und Entscheidungen während der Diplomarbeit gilt, sind laufend Aufzeichnungen von den Schülerinnen bzw. von den Schülern zu führen.

Dazu gibt es mehrere Möglichkeiten:

* das auf der DA-Webseite (www.dipolmarbeiten-bbs.at) vorgeschlagene Formular „Begleitprotokoll“ oder
* die Projektmanagement Tools (mit Taskverwaltung, Zeittracking und Meeting-Protokollen) oder
* die digitale Ablage in einem Dokumentenverwaltungssystem (z. B. Dropbox usw.)

Die gewählte Form ist mit der Betreuerin bzw. dem Betreuer abzuklären und beinhalte folgende Aufzeichnungen:

* Dokumentation wichtiger Entscheidungen und Ereignisse,
* Teambesprechungen deren Inhalte und Beschlüsse,
* Besprechungen mit Betreuerinnen und Betreuern,
* Dokumentation des individuellen Zeitaufwandes,
* Kontakt zu Sponsoren, Investoren und Partnern.

Alle Inhalte müssen korrekt und vollständig dokumentiert sein. Auf Wunsch der Betreuerin bzw. des Betreuers sind die Aufzeichnungen jederzeit vorzulegen.

Diese Aufzeichnungen dienen als:

* Nachweis von Tätigkeiten und Besprechungen,
* Nachweis der Betreuungstätigkeit,
* Überblick und Nachvollziehbarkeit von wichtigen Entscheidungen,
* Nachvollziehbarkeit des Informationsflusses.

Quelle: <http://www.diplomarbeiten-bbs.at/erstellung>

**Vorschlag** 🡺 Monatliche Zeit-Übersicht auf Basis der Wochenberichte

## Begleitprotokoll <Schüler 1>

|  |  |  |
| --- | --- | --- |
| **Zeitraum** | **Arbeiten / Tätigkeiten / Meetings / …** | **Stunden** |
| 2019/08 |  |  |
| 2019/09 |  |  |
| 2019/10 |  |  |
| 2019/11 |  |  |
| 2019/12 |  |  |
| 2020/01 |  |  |
| 2020/02 |  |  |
| 2020/03 |  |  |
| 2020/04 |  |  |

## Begleitprotokoll <Schüler 2>

## Begleitprotokoll <Schüler 3>

# Anhang

## Istbestand

## Angebote

## Lieferscheine

## Dimensionierung

## Messprotokolle

## Testprotokolle

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dokument: | **Testprotokoll** | | | | |
| Projekt: | **<DA-Name>** | | | | |
| Version: |  | | | Datum: |  |
|  |  |  |  |  |  |
| Gültig für: | | | | | |
| Testfallbeschreibung: | **Diplomarbeit Kapitel 10** Testfallspezifikation | | | | |
| Test-Objekt: | **<DA-Name> Prototyp** | | | | |
|  |  |  |  |  |  |
|  | Name: | | | Abteilung: | |
| Test-Leiter: |  | | | Elektronik und Technische Informatik | |
| Tester: |  | | | Elektronik und Technische Informatik | |
|  |  |  |  |  |  |
| Testfall | Datum / Zeit | Status OK / not OK | Bemerkung: | | |
| Testfall <A> |  | OK |  | | |
| Testfall <B> |  | OK |  | | |

1. Ein Wort ist eine Folge von mehreren Buchstaben ohne " " (Leerzeichen) [↑](#footnote-ref-1)
2. Ein Satz ist eine Folge von Wörtern, abgeschlossen mit dem Satzzeichen "." (Punkt) [↑](#footnote-ref-2)
3. Eine Fußnote einfügen durch Cursor positionieren [STRG] + [ALT] + [F] drücken. Durch Löschen der Fußnote im Text (oben) wird auch die Fußnote entfernt (nachfolgende rücken automatisch auf) [↑](#footnote-ref-3)