

# **Next-Level Relaunch der Schulhomepage: neuestes Design und aktuellste Animationen**

## **DIPLOMARBEIT**

verfasst im Rahmen der

**Reife- und Diplomprüfung**

an der

**Höheren Abteilung für Informationstechnologie-Medientechnik**

Eingereicht von:

Mona Angerer  
Peter Klose

Betreuer:

Prof. Mag. Martin Huemer

Projektpartner:

HTL Leonding

Leonding, April 2024

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Die Arbeit wurde bisher in gleicher oder ähnlicher Weise keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Die vorliegende Diplomarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

Leonding, April 2024

M.Angerer & P.Klose

# Abstract

The present diploma thesis deals with the redesign of the school homepage of HTL Leonding, aiming to bring the design up to date and present the most important information logically and clearly.

The current version of the HTL Leonding homepage serves as the starting point for this work. An examination of the best designs available at the time of creation subsequently led to a rough draft of the website. This draft was then grouped into blocks that could be implemented in Storyblok. For each block, a corresponding representation was developed in Next.js.

This thesis illustrates the development of the website in the process as it was developed and provides insights into the technical background.



# Zusammenfassung

Die vorliegende Diplomarbeit beschäftigt sich mit der Neugestaltung der Schulhomepage der HTL Leonding, mit dem Ziel, das Design auf den neuesten Stand zu bringen und die wichtigsten Informationen logisch und übersichtlich darzustellen.

Die aktuelle Version der HTL Leonding Homepage bildet den Ausgangspunkt dieser Arbeit. Eine Auseinandersetzung mit den besten Designs zum Zeitpunkt der Erstellung führte anschließend zu einem Grobentwurf der Webseite. Dieser Entwurf wurde dann in Blöcke gruppiert, die anschließend in Storyblok implementiert werden konnten. Für alle Blöcke wurde dann eine entsprechende Darstellung in Next.js entwickelt.

Diese Arbeit zeigt die Entstehung der Webseite in dem Ablauf, wie sie entwickelt wurde, und bietet Einblicke in den technischen Hintergrund.



# Inhaltsverzeichnis

<b>1 Einleitung</b>	<b>1</b>
1.1 Abkürzungen . . . . .	1
1.2 Ausgangssituation . . . . .	1
1.3 Problemstellung . . . . .	1
1.4 Aufgabenstellung und Ziele . . . . .	2
<b>2 Design</b>	<b>4</b>
2.1 Recherche . . . . .	4
2.2 Content-Strukturierung . . . . .	5
2.3 Workshop . . . . .	5
2.4 Entwurf und Usability-Tests . . . . .	8
2.5 Finales Design . . . . .	9
<b>3 Technologien</b>	<b>12</b>
3.1 Storyblok . . . . .	12
3.2 Next.Js 13 . . . . .	12
3.3 Vercel . . . . .	13
3.4 TailwindCss . . . . .	13
3.5 Framer Motion . . . . .	14
3.6 Lottie . . . . .	14
<b>4 Umsetzung</b>	<b>15</b>
4.1 Storyblok . . . . .	15
4.2 Next . . . . .	35
4.3 Animationen . . . . .	43
<b>5 Zusammenfassung</b>	<b>44</b>
<b>Glossar</b>	<b>v</b>

<b>Literaturverzeichnis</b>	<b>VI</b>
<b>Abbildungsverzeichnis</b>	<b>VII</b>
<b>Tabellenverzeichnis</b>	<b>VIII</b>
<b>Quellcodeverzeichnis</b>	<b>IX</b>
<b>Anhang</b>	<b>X</b>

# 1 Einleitung

## 1.1 Abkürzungen

Um zu Kennzeichnen, welche Abschnitte der Diplomarbeit von wem verfasst wurden, werden in den Überschriften der Kapitel die Buchstaben M für Mona Angerer und P für Peter Klose hinzugefügt. Wenn kein Autor angeführt wird, ist der folgende Abschnitt von Beiden geschrieben.

## 1.2 Ausgangssituation

Seit dem Jahr 2019 verfügt die HTL Leonding über eine Website, die damals ebenfalls von SchülerInnen entwickelt wurde und auf Wordpress basiert.

## 1.3 Problemstellung

Die HTL Leonding-Website, die 2019 von Schülern auf Wordpress-Basis entwickelt wurde, zeigt eine durchaus den Anforderungen einer Schulhomepage entsprechenden, jedoch in die Jahre gekommene Plattform. Die technischen Möglichkeiten und Designstandards haben sich in den letzten Jahren erheblich weiterentwickelt, was dazu führt, dass die aktuelle Website nicht mehr den zeitgemäßen Ansprüchen entspricht.

Die bestehende Website weist Unstimmigkeiten in der Benutzerführung auf, insbesondere im Hinblick auf die Menüstruktur, die als unübersichtlich wahrgenommen wird. UserInnen fällt es schwer, sich auf der Oberfläche zurechtzufinden und den gesuchten Inhalt auf Anhieb zu finden. Während mancher Content nur schwer zu finden ist, weist die Homepage auch über Inhalte auf, die an mehreren Stellen und unterschiedlichen Unterseiten zu finden ist. Dies verstärkt zusätzlich die Verwirrung und schlechte intuitive Handhabung. Zudem verfügt sie über lange Ladezeiten, was die Benutzererfahrung deutlich beeinträchtigt. Durch die langen Unterbrechungen, die in der Bedienung entstehen könnten und die immer kürzer werdende Aufmerksamkeitsspanne und Ungeduld der Menschen, könnte es nicht nur zu einer getrübten Stimmung, sondern sogar zum

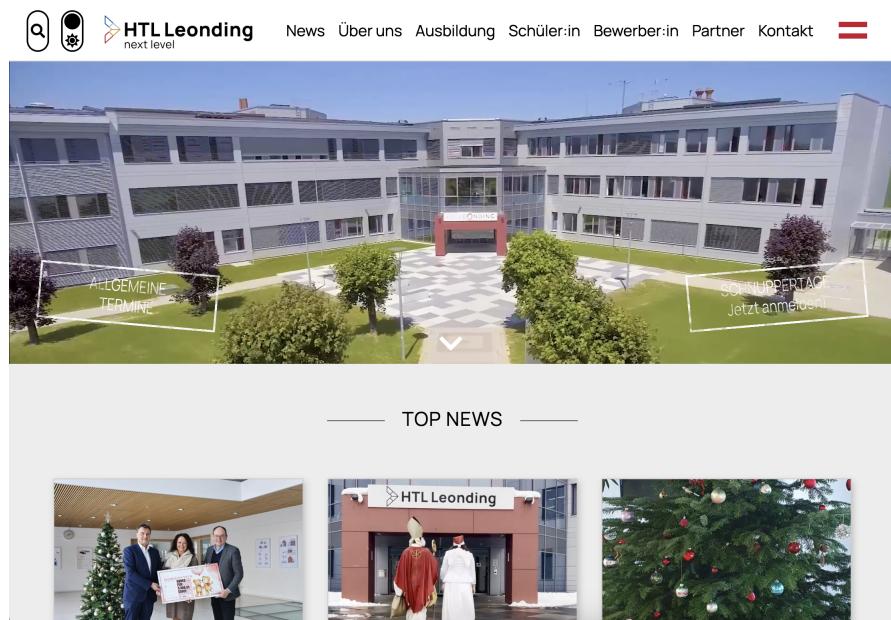


Abbildung 1: Startseite Ausgangssituation

Verlassen der Website kommen. Das Design erscheint zu bunt und durch die vielen Bilder Videos, die oftmals einen Großteil der Seite einnehmen, wirkt der Webauftritt der HTL Leonding überladen und nicht mehr zeitgemäß. Des Weiteren folgt die Webanwendung einem strikten Box-Design, was fehlende Dynamik zur Folge hat und eintönig wirkt. Auch im Mobile-Modus gibt es zudem Herausforderungen, wie schwierige Bedienung des Menüs und Schwierigkeiten beim Zurechtfinden und Navigieren. Da immer mehr Menschen eher auf ihren Mobilgeräten und nicht nur ihren PCs und Laptops Webseiten aufrufen, gewinnt insbesondere dies progressiv an Bedeutung.

Es hat sich gezeigt, dass die Website möglicherweise nicht mehr effektiv die Bedürfnisse und Ziele der verschiedenen Nutzergruppen erfüllt. Die Schulleitung hat angesichts dieser Erkenntnisse den Vorschlag gemacht, im Rahmen einer Diplomarbeit einen neuen Webauftritt zu gestalten. Dies bietet die Chance, die bestehenden Herausforderungen zu adressieren, die Website zu modernisieren und eine verbesserte Benutzererfahrung für alle Zielgruppen zu schaffen.

## 1.4 Aufgabenstellung und Ziele

Um ein breites Spektrum an Technologien abzudecken und unterschiedliche Ansätze zu fördern, werden zwei Teams mit dieser Diplomarbeit betraut. Diese Herangehensweise ermöglicht es, verschiedene Ideen und Lösungsansätze zu erforschen und so eine

fundierte Grundlage für die Gestaltung des neuen Webauftritts der HTL Leonding zu schaffen.

Die Vorgabe besteht darin, als Backend das headless CMS Storyblok zu verwenden. Die Motivation hinter dieser Entscheidung liegt in der aktuellen Relevanz und Flexibilität dieses Content Management Systems. Die SchülerInnen werden dazu ermutigt, sich eingehend über verschiedene Frontend-Varianten zu informieren und dazu zu recherchieren, um diejenige zu finden, die sich am besten mit Storyblok kombinieren lässt und den Anforderungen am besten gerecht wird. Die neue Website soll eine mit entsprechende Backend-Struktur aufgebaut sein, so dass sie für die Zukunft eine leichte Wartung ermöglicht. Um die Benutzerfreundlichkeit zu erhöhen wird gefordert, den Content besser zu strukturieren, abzuwägen, welche Inhalte ins LeoWiki ausgelagert werden sollen und die Inhalte so zu platzieren, dass eine intuitive Navigation für alle Benutzergruppen ermöglicht wird. Das Design der Website soll an die derzeitigen Standards und Trends angepasst werden und mithilfe von Animationen und spannendem Design einladend und ansprechend gestaltet werden.

Das endgültige Ziel ist eine technisch sauber gelöste, gut strukturierte, ansprechend gestaltete Weboberfläche, die eine einfache und intuitive Navigation ermöglicht. Alle relevanten Inhalte, die die HTL Leonding repräsentieren, sind auf der Homepage zu finden und bilden ein einladendes Bild der HTL Leonding.

## 2 Design

Eines der Hauptthemen der Diplomarbeit ist die Überarbeitung des bisherigen Designs. Um dies zu erreichen, werden nicht nur über die aktuellen Standards und Trends analysiert, der aktuelle Content der Website restrukturiert und Usability-Tests durchgeführt, sondern auch ein mehrstündiger Workshop zum Thema UI/UX Design an der JKU absolviert.

### 2.1 Recherche

Um ein umfassendes Verständnis für die derzeit gängigen Designmethoden erhalten zu können, wurden zahlreiche bekannte große Websites wie die von Apple (...) und anderen analysiert. Dabei werden die überschneidenden Merkmale identifiziert und herausgearbeitet und diejenigen, die für die HTL-Website besonders relevant sind, anschließend genauer untersucht. Auffällig ist dabei die übergeordnete Präferenz für das Motto „Weniger ist mehr“. Durchgehend dominiert minimalistisches Design mit vielen weißen Flächen, wodurch ein edler, strukturierter Eindruck erweckt wird.

Im Kontrast dazu werden jedoch auch viele Vektor- oder svg-Animationen, oftmals auch im „handgezeichneten“ Stil, verwendet, die das saubere Layout auflockern und Bewegung in die Benutzeroberfläche bringen.

Darüber hinaus gewinnen Scrollingeffekte zunehmend an Bedeutung und werden von immer mehr Unternehmen als wichtiges und vielseitig einsetzbares Designelement betrachtet. Ob Parallax-Effekte, Scrollitelling, Immersive oder Horizontal Scrolling – das Weiterscrollen wird nicht mehr nur als „Bildschirminhalt verschieben“ gesehen, sondern wird zu einem immer wichtigeren, vielseitig eingesetzten Element.

Ein weiterer interessanter Trend ist die verstärkte Aufmerksamkeit auf individuell gestaltete Error-Seiten. Diese werden zunehmend in das Gesamtdesign der Website integriert und mit kleinen Spielereien wie Animationen oder Mini-Games ausgeschmückt.

Geometrische Ästhetik, insbesondere abstrakte Formen wie Dreiecke, Kreise, Vierecke oder eine Kombination davon, gewinnen im Web-Bereich deutlich sichtbar an Popularität.

Auf vielen hochwertigen Websites tragen diese geometrischen Gestaltungselemente dazu bei, eine strukturierte und interessante Oberfläche zu schaffen.

## 2.2 Content-Strukturierung

Durch Gespräche mit SchülerInnen, Lehrkräften und InteressentInnen stellt sich hinaus, dass einer der am häufigsten bemängelten Aspekte der bisherigen HTL-Website der unübersichtliche Aufbau mit etlichen Unterseiten ist. Es fällt den Usern teilweise schwer, sich auf der Benutzeroberfläche zu orientieren und den gesuchten Inhalt auf Anhieb zu finden. Um dieses Problem zu beheben, wird zunächst eine eingehende Analyse des Menüs und seiner Unterpunkte durchgeführt, um einen umfassenden Überblick über den gesamten Website-Inhalt zu erhalten. Nach intensiver Prüfung wird anschließend festgestellt, dass durch eine Neustrukturierung von 7 Menüpunkten auf nur noch 5 Hauptseiten übergegangen werden kann.

Des Weiteren wird ein erheblicher Teil des Inhalts von der Website ins LeoWiki, das interne Wiki der HTL Leonding, ausgelagert. Dies verhindert, dass Content, der nur für Personen, die bereits an der HTL Leonding lernen oder lehren, relevant ist, für Außenstehende sichtbar ist und ermöglicht, dass die meisten Seiten keine weiteren Unterseiten besitzen und somit als One Pager fungieren, auf denen der Inhalt durch einfaches Scrollen zugänglich ist.

## 2.3 Workshop

An dem Workshop auf dem Campus der JKU, der von ... von der Firma KBC abgehalten wurde, nahmen beiden Diplomarbeitsteams, der Betreuungslehrer Herr Professor Huemer und die beiden Professorinnen Frau Engleitner und Frau Rammelmüller teil.

Um einen Ausgangspunkt für die Entwicklung eines Designkonzepts zu schaffen, führt man zunächst eine umfassende Problemanalyse durch (Siehe Abbildung 2). In Teams wird die bewährte Post-It-Methode angewendet, bei der jeder/jede TeilnehmerIn unterschiedlich farbige Zettel erhält, um Mängel und Verbesserungsvorschläge auf gemeinsame Plakate zu kleben. Dieser kollaborative Ansatz ermöglicht die Erstellung einer Art Mindmap, auf der die Schwächen der aktuellen HTL-Website deutlich herausgearbeitet werden. Dabei werden Herausforderungen wie die Unübersichtlichkeit des Menüs, eine zu bunte Gestaltung und Probleme im Mobile-Modus hervorgehoben. Diese Methode

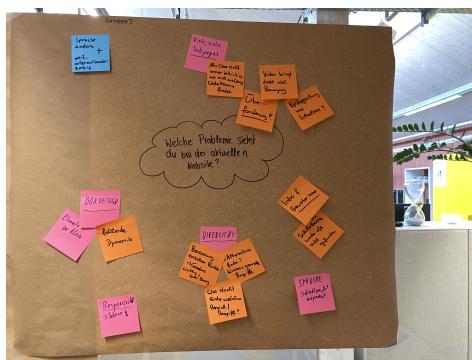


Abbildung 2: Problemanalyse



Abbildung 3: Zielgruppenanalyse

lenkt den Fokus von Anfang an auf die Lösungsfindung in Beachtung der bereits existierenden Probleme, um somit nicht nur eine Neuimplementierung, sondern eine konkrete Verbesserung der Website zu erreichen.

Der kreative kollaborative Prozess setzt sich fort und mündet in einer detaillierten Zielgruppenanalyse (Siehe Abbildung 3). Dieser Schritt ist von besonderer Bedeutung in der Designentwicklung, da eine Benutzeroberfläche erst dann als gelungen betrachtet werden kann, wenn sie von den BenutzerInnen intuitiv genutzt werden kann und ihren individuellen Anforderungen gerecht wird. Bei der HTL-Website werden verschiedene Benutzergruppen identifiziert, darunter SchülerInnen, LehrerInnen, InteressentInnen, Firmen und Eltern. Zusätzlich berücksichtigt man deren spezifische Intentionen. Beispielsweise ist für Unternehmen von großem Interesse, welche Projekte an der Schule verfolgt werden und welche Technologien dafür verwendet werden. Eltern und InteressentInnen wiederum möchten vorrangig Informationen zu den angebotenen Zweigen und Fächern an der HTL erhalten, während Lehrkräfte und SchülerInnen insbesondere anstehende Events und Aktivitäten im Blick haben möchten. Diese präzise Zielgruppenanalyse bildet die Grundlage für ein benutzerzentriertes Designkonzept, das den Bedürfnissen aller Zielgruppen gerecht wird.

Um die Benutzerperspektive noch intensiver zu erfassen, geht man im weiteren Verlauf darauf ein, welche Gedanken, Wünsche, Handlungen und Emotionen die NutzerInnen während der Verwendung der Website durchlaufen (Siehe Abbildung 4). Dabei werden nicht nur positive Gefühle und Gedanken, wie Vorfreude und Neugierde, herausgearbeitet, sondern auch potenzielle Ängste oder Unsicherheiten. Hierzu gehören beispielsweise Fragen wie: „Bin ich gut genug für die HTL?“ oder „Habe ich überhaupt Chancen, aufgenommen zu werden?“.

Die Berücksichtigung dieser vielschichtigen Nutzererfahrungen ermöglicht eine empathische Gestaltung der Benutzeroberfläche, die nicht nur informativ ist, sondern auch dazu



Abbildung 4: UserInnen Gefühle



Abbildung 5: UserInnen Missionen

beiträgt, positive Emotionen zu fördern und mögliche Ängste zu mildern. Durch diese eingehende Analyse der Nutzerperspektive wird die HTL-Website nicht nur funktional, sondern auch emotional ansprechend und unterstützend gestaltet.

Um eine intuitive Benutzererfahrung auf der Website sicherzustellen, werden darüber hinaus potenzielle Missionen, Gedankengänge und Vorgehensweisen der NutzerInnen berücksichtigt, die sie bei ihrem Besuch auf dem HTL-Webauftritt haben könnten (Siehe Abbildung 5). Es wurde dabei analysiert, welche konkreten Ziele sie verfolgen, welche Informationen sie suchen und welche Schritte sie wahrscheinlich unternehmen möchten.

Diese detaillierte Betrachtung der Nutzerinteraktion ermöglicht es, die Benutzeroberfläche so zu gestalten, dass sie den natürlichen Denk- und Handlungsmustern der BenutzerInnen entspricht. Durch das Verstehen der potenziellen Missionen und Gedankengänge wird sichergestellt, dass die Website nicht nur informativ ist, sondern auch nahtlos in den individuellen Ablauf der NutzerInnen integriert wird. Dieser Ansatz fördert eine reibungslose und effektive Nutzung der Website.

Mit dem erlangten Wissen über die zu behebenden Probleme und die unterschiedlichen Usergruppen, deren individuelle Anforderungen an die HTL-Website, die Ziele, die sie mit dem Besuch der Website verfolgen möchten und die Emotionen und Eindrücke, die man in den Usern beim benutzen der Oberfläche erwecken will, wird nun der Startpunkt für die Erstellung eines Designentwurfs erleichtert. Dazu wurde unter den TeilnehmerInnen des Workshops Zettel und Stifte ausgeteilt, um Skizzen und mögliche Layouts für die Weboberfläche zu gestalten. (Siehe Abbildung 6) Durch diese Methode werden eine Vielzahl an Ideen erbracht, die in der gesamten Gruppe geteilt und diskutiert werden. Dieser kreative Ansatz ermöglicht es, die gewonnenen Erkenntnisse unmittelbar in konkrete visuelle Konzepte umzusetzen und so den Grundstein für eine optimierte HTL-Website zu legen.

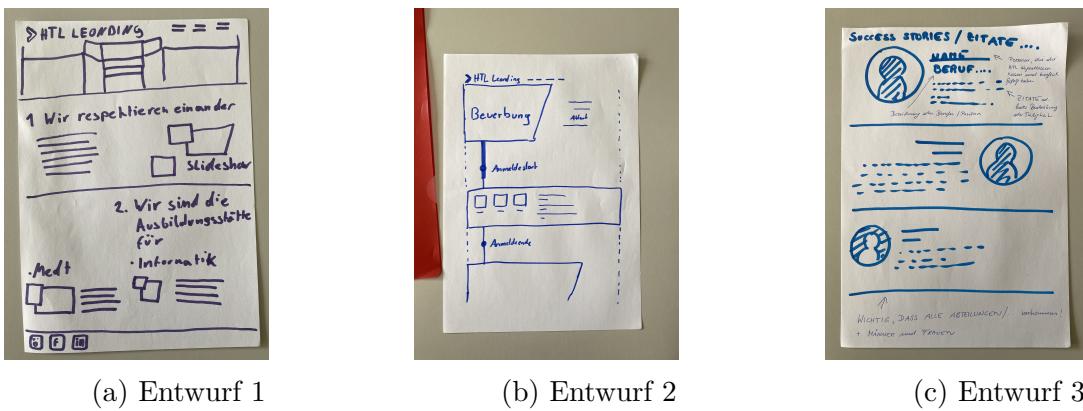


Abbildung 6: erste Entwürfe

## 2.4 Entwurf und Usability-Tests

Nach umfassenden Analysen und Untersuchungen der geeigneten Gestaltungselemente, Effekte und Animationen für die HTL-Website wird unter Anwendung des im Workshop erworbenen Wissens und der Designmethoden ein erster Entwurf erstellt. Die Gestaltung wird mithilfe der Plattform Figma in Form eines Click-Dummies skizziert. (Siehe Abbildung 7) Anschließend präsentiert man diesen Entwurf SchülerInnen der ersten und zweiten Klasse der HTL sowie Lehrkräften und externen Personen. Diese Form der Testung ermöglicht es, direktes Feedback und Bewertungen zu sammeln, um den Entwurf weiter zu verfeinern und optimal an die Bedürfnisse der BenutzerInnen anzupassen. Der Prozess integriert somit gezielt die Perspektiven der verschiedenen Zielgruppen, um eine benutzerfreundliche Website zu gewährleisten. Die erhaltenen Rückmeldungen bekräftigen die verbesserte Struktur und die aufgewertete Anordnung durch die Reduzierung der Menüpunkte. Zudem wird bestätigt, dass die Navigation auf der Benutzeroberfläche merklich vereinfacht wurde. Allerdings werden auch einige Anregungen und Kritiken geäußert, darunter die Empfehlung, vermehrt Schülerfotos einzubinden, um die Website persönlicher zu gestalten. Diese Maßnahme soll dazu beitragen, ein einladenderes Bild der HTL Leonding zu vermitteln. Auch werden einige Verschiebungen von Inhalt in andere Menüpunkte vorgeschlagen, um eine logischere Anordnung sicherzustellen. Nach der Implementierung dieser Verbesserungsvorschläge wird der Prozess mehrfach wiederholt, um das Design weiter zu perfektionieren und eine optimale Zufriedenheit aller BenutzerInnen sowie eine intuitive Steuerung der Website zu erreichen.

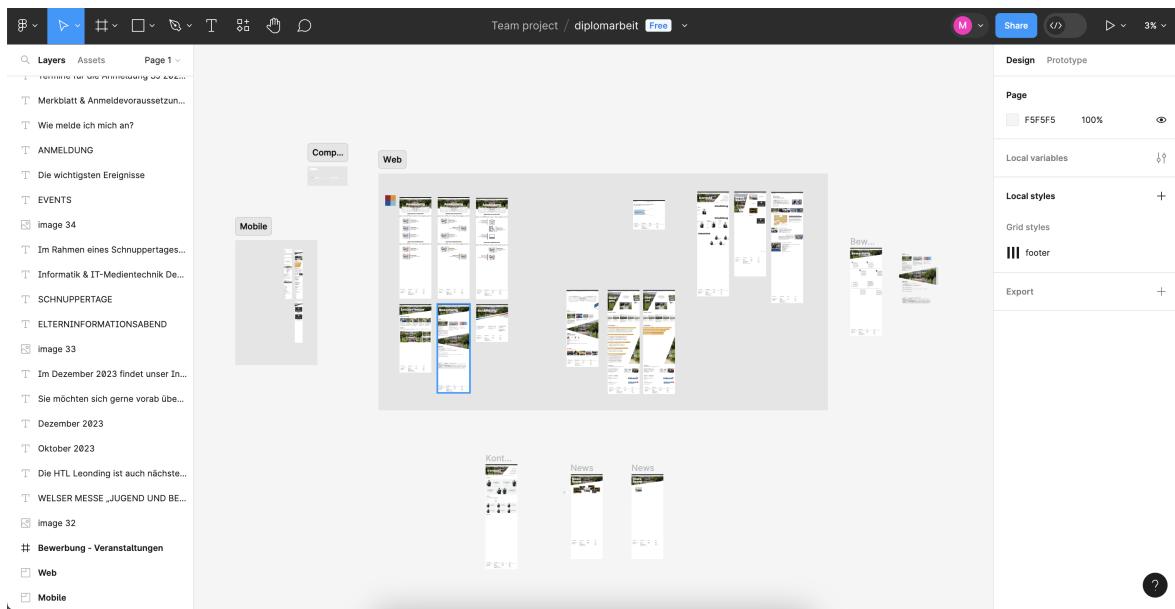


Abbildung 7: Entwurf mit Figma

## 2.5 Finales Design

Nachdem der Prozess des Usability-Testings und der Überarbeitungen einige Male durchlaufen wird, entsteht am Ende ein optimiertes und finales Design.

Dieses beinhaltet folgende Designelemente und unterscheidet sich in diesen Punkten zu der Gestaltung der bisherigen HTL-Website:

### 2.5.1 Menu und Inhalt

Im Vergleich zum alten Menu-Header (Siehe Abbildung 8) hat der Neue (Siehe Abbildung 9) statt 7 nur mehr 5 Elemente, um eine sauberere Ansicht zu schaffen und Verwirrung in der Navigation zu vermeiden. Die Inhalte aus dem Reiter „News“ können nun durch einen Link auf der Seite „Über uns“ erreicht werden, die Partnerfirmen stehen nun auf der Startseite. Der Menüpunkt „Schüler:innen“ ist nun am rechten Bildschirmrand platziert, um die Hauptelemente des Headers erneut zu vermindern und um eine Art „Profil“ oder „Einloggen“ zu suggestieren. Dort finden sich ebenfalls keine nur für bereits an der HTL Leonding lernende Personen, denn dieser Inhalt wurde umfassend in das LeoWiki ausgelagert. Stattdessen findet man unter dem Reiter jetzt die an der Schule angebotenen Programme und SchülerInnenfotos, um einen persönlichen Einblick zu bieten und den Wünschen und Vorschlägen der bei den Usability-Tests befragten Personen nachzugehen.



Abbildung 8: alter Header



Abbildung 9: neuer Header

## 2.5.2 Farben und Formen

Da die bisherige Website sehr viele Farben beinhaltete, erschien sie überladen und überfordernd. Nun wird bei der Gestaltung auf viel weiße Fläche gesetzt, die abgesehen von dem Bildmaterial lediglich von kleinen Akzenten in den vier Farben der Abteilungen, die auch im Logo vorkommen, und einer fünften Farbe, die eine Mischung der vier "HTL-Leonding-Farben" ist, unterbrochen wird. Der Trend zu viel "Whitespace" ist, wie im Kapitel Recherche bereits behandelt wurde, ein häufig aufkommendes, modernes Designkonzept.

Auch wurde statt auf das starre Block-Design auf mehr Abwechslung und Bewegung in den Formen gesetzt. So ist das Dreieck eine der wichtigsten Elemente des neuen Webauftritts. Der Grund, weshalb die Wahl auf ausgerechnet diese Grundform gefallen ist, liegt in dem Logo der Schule. Dieses besteht aus einem Pfeil, in dem sich die bereits angesprochenen vier Abteilungsfarben wiederspiegeln. Mittels der verschiedenen Implementierung des Dreiecks auf den unterschiedlichen Seiten der Website spiegeln sich so die Schrägen, die man bereits im Logo findet, wieder. Auch erscheint der Gesamteindruck dynamischer und spannender, was zu einer verbesserten User-Experience führt.

## 2.5.3 Dynamik und Bewegung

Nicht nur die wiederholte Einbringung des nicht-typischen, wandelbaren Elements des Dreiecks bringt mehr Spannung und Dynamik in die Weboberfläche, auch sind viele sich bewegende Objekte auf der Website eingebunden. Um allerdings keine Unruhe und Stress in den Benutzern auszulösen, ist dieser bewegte Content subtil und nicht zu oft angewendet und langsam in seiner Bewegung. Beispiele dafür finden sich auf der Seite "Abteilungen", wo sich eine Übersicht der Fächernamen von links nach rechts und umgekehrt über den Bildschirm bewegt, oder auf den Seiten der einzelnen Abteilungen, wie beispielsweise "Informatik - SSE", wo sich eine Reihe von Bildern, die die Fachrichtung beschreiben, sich vertikal über das Display bewegen.

## 2.5.4 Animationen

Um das durch den vielen Whitespace erzeugte schlichte Design etwas aufzubrechen, wurde eine Startanimation, sowie Animationen für jede Abteilung eingefügt. Die Startanimation, die beim Aufrufen der Website abgespielt wird, beginnt mit dem HTL Leonding-Logo mit dem Schriftzug "HTL Leonding next level", geht dazu über dass der Schriftzug verschwindet, sich das Logo in die Mitte des Bildschirms bewegt und in eine vereinfachte Grafik der HTL von oben verwandelt. In der Schlussposition befindet sich die Grafik der HTL am rechten Bildschirmrand und die wichtigsten Events scrollen aus dem Gebäude.

# 3 Technologien

## 3.1 Storyblok

Storyblok ist ein Headless Context Management System (CMS), welches von einem Absolventen unserer Schule mitentwickelt worden ist. Dessen Verwendung ist auch die einzige Vorgabe, die seitens der Schule für die Umsetzung der Dipolarbeit gestellt wurde. Ein reguläres CMS, unter anderem Wordpress, mit dem die bisherige HTL-Website umgesetzt wurde, ist eine Softwareprodukt, welches den Benutzern ermöglicht, digital die Daten deren Webseite zu erstellen, gemeinsam zu bearbeiten, zu speichern und anschließend zu veröffentlichen. Storyblok hingegen ist kein herkömmliches CMS, sondern ein headless, wortwörtlich aus dem englischen übersetzt "kopfloses" CMS, was bedeutet, dass Storyblok nur die Daten und Datenspeicherung betrifft (der Body), nicht aber die Anwendung selbst, die wir als User sehen, in unseren Fall die Webseite (der Head). Durch diese Struktur und der daraus resultierenden Abgrenzung haben die Entwickler sehr viel Freiheit. Diejenigen, die den Content bearbeiten und erstellen müssen sich keine Gedanken über die Endgeräte machen. Weiters sind die Frontend-Entwickler nicht limitiert auf eine bestimmte Technologie. Somit kann eine Marke oder ein Produkt per iOS App, Android App und Web representiert werden, ohne, dass man sich auf eine bestimmte Technologie fokussieren muss. Jeder kann nativ beziehungsweise mit der Software, die ihm am meisten zusagt, programmieren.

## 3.2 Next.Js 13

Als Frontend-Technologie stützen wir uns auf Next.js 13, ein React Framework für full-stack Web-Applikationen. Die hauptsächlichen Features von Next sind dabei folgende:

### 3.2.1 Routing

Next liefert direkt einen Filesystem basierten Router mit. Dieser unterstützt unter anderem Layouts, verschachtelte Routen, den Ladestatus und viele weitere. Für die

HTML-Website wurde dabei der neue App Router Ansatz gewählt, welcher mit Next 13 veröffentlicht wurde. Durch ihn werden die Neuheiten von React, wie Server Components und Streaming in Next eingebaut.

### 3.2.2 Optimierung

In der Webbranche ist Pagespeed mitunter das Wichtigste. Dies hat auch Next verstanden und bietet daher einige Optimierungsoptionen bei Bildern, Links oder auch bei den Metadaten und Scripts. Bei den Bildern wurde das `<img>` Element dahingehend überarbeitet, dass Bilder "lazy"geladen werden und dynamisch auf die Größe des Bildschirms angepasst werden. Das `<a>` Tag beziehungsweise der Link wurde dahingehend angepasst, dass die Webseite, wenn sie vollständig geladen ist, die möglichen Ziele vorlädt um somit eine schnelleren und weicheren Wechsel der Seiten zu ermöglichen.

### 3.2.3 Rendern

Next bietet auch die Möglichkeit der Client oder Server-Components. Client-Components sind die Teile der Webseite, die im Client, also im Browser, geladen und gezeigt werden. Server-Components hingegen müssen zunächst auf dem Server zusammengebaut werden und werden dann erst zum Client geschickt.

## 3.3 Vercel

Beim Deployment Vercel hauptsächlich verwendet, weil Next aus dem selben Hause stammt und das Deployment somit keine Problem darstellt. Sobald 'next build' funktioniert, kann das GitHub Repository einfach mit einem Vercel Account verbunden werden und der main-Branch ist nach 2 Minuten deployed.

## 3.4 TailwindCss

Als CSS Framework wird TailwindCSS als PostCSS Plugin verwendet. Um trotzdem eine übersichtliche Struktur in die Klassennamen beizubehalten wurde die VS-Code Erweiterung Prettier verwendet. Dadurch werden die Klassennamen gemäß der vorgeschlagenen Sortierung von Tailwind geordnet.

## 3.5 Framer Motion

Als Animationsframework wird Framer Motion verwendet. Dadurch wurden alle Frontend-seitigen Animationen, die den HTML- und CSS-Code betreffen, erstellt. Dieses Framework behandelt jedoch nicht nur einfache CSS-Animationen; die Fähigkeiten des Frameworks gehen weit darüber hinaus. Beispiele hierfür sind Layout-Animationen, die seitenübergreifende Transformationen ermöglichen, sowie Load- und Unload-Animationen, die das Laden und Entladen des DOM animieren können, um Seitenübergänge zu ermöglichen.

## 3.6 Lottie

Als SVG-Animationsframework wird Lottie verwendet. Einerseits erfolgt dies bereits beim Export aus After Effects als Lottie JSON und andererseits im Code zur Einbindung dieser JSONs in den Lottie Player. Dadurch werden alle übrigen Animationen, die nicht durch Framer Motion gesteuert werden können, durch Lottie gesteuert.

# 4 Umsetzung

## 4.1 Storyblok

In Storyblok muss zuallererst die gewollte Struktur der Webseite wiedergespiegelt werden. Dies wird im Bereich 'Content' gemacht. Für Pfade mit Unterseiten werden Ordner erstellt und die restlichen Seiten als 'Page', 'Branch' oder 'Article' definiert. Die Content-Gestaltung und Festlegung findet dann in diesen 3 Typen statt.

### 4.1.1 Storyblok API

Da Storyblok ein headless CMS ist, ist die einzige Möglichkeit auf die Daten zuzugreifen deren API-Schnittstelle im REST Standard. Die von Storyblok bereitgestellte API verwendet auch in Error Handling bekannte HTTP Error Meldungen ansonsten, wenn es keinen Fehler gibt werden Daten im JSON Format weitergegeben. Die Basis URL des Endpoint lautet dabei wie folgt: <https://api.storyblok.com/v2>. Um jedoch die richtigen Daten zu erhalten müssen noch die 2 Wichtigsten Parameter gesetzt werden nämlich **version** entweder als **draft** oder **published** und der Token der zur Authentifizierung dient. Sonstige Parameter beziehungsweise Einstellungen die verwendet wurden lauten:

#### Pagination

Die Pagination ermöglicht es Listen von Elementen zu begrenzen um sie dann mit mehreren Unterseiten verfügbar zu machen. Ein Beispiel wäre die News Seite, sie lädt die neusten 25 News und wenn man auf 'weitere News' am Ende der Seite klickt erhält man die nächsten 25. Hierfür werden die Parameter `per_page` und `page` verwendet. Der standardmäßigen eingestellte Wert für beide lautet 25 bei `per_page` und 1 bei `page`.

#### Stories

Stories stellen gesamte Seiten dar, wenn man also die `/ausbildung/it-medientechnik` URL aufrufen will macht man um alle Daten zu erhalten einen Request an die

<https://api.storyblok.com/v2/cdn/stories/ausbildung/it-medientechnik> URL. Hiermit würde man einen Großteil der Daten bekommen jedoch nicht alle. Verlinkte News, die eine eigene URL besitzen würde man nur als verlinke ID sehen. Um dieses Problem zu lösen bietet Storyblok den sogenannten **resolve\_relations** Parameter. Hinter diesen Parameter muss allerdings noch der richtige Verweis angegeben werden in folgenden Format: **component\_name.field\_name** somit werden dann wirklich alle Daten geladen die benötigt werden um die Seite darzustellen.

### 4.1.2 Internationalisierung

Storyblok bietet für die Internationalisierung 3 verschiedene Ansätze: Field Level translation, Folder Level translation und Space Level translation. In dieser Arbeit wurde der **Field Level Translation** Ansatz gewählt. Das Setup dafür schaut wie folgt aus:

Als Erstes müssen in den Einstellungen des Spaces unter dem Reiter 'Internationalization' die gewünschten Sprachen eingestellt werden. In diesen Fall war dies 'de' für Deutsch und 'en' für Englisch. Im Editor kann dann die Sprache auf Englisch gestellt werden und alle Felder, bei denen die Translation Checkbox angehakt ist können übersetzt werden.

Um diese Sprache dann auch auf der Webseite angezeigt zu bekommen muss bei dem Request an die API am Ende einfach der Parameter **language=** angeben werden. In diesem Fall wäre dies 'de' oder 'en'.

### 4.1.3 Bausteine der Webseite

Wie der Name schon andeutet, handelt vieles in diesem headless CMS von Blöcken. Diese haben Variablen, welche dann im Editor mit Content befüllt werden können. Einen Überblick über alle verfügbaren Blöcke hat man in der Block library. Dort kann man auch neue Blöcke erstellen. In der finalen Version der Webseite sind 41 Blöcke zu finden.

All unsere Blöcke entsprechen der selben Namensschreibweise. Dabei wird die Snake case Schreibweise verwendet, diese beginnen immer klein und mehrere Worte trennt ein \_ als Beispiel 'default\_component'.

In einem Block gibt es dann eine Ansammlung von Feldern, dabei gibt es mehrere zur Auswahl:

Tabelle 1: Field Types, die von Storyblok bereitgestellt werden

Feldname	Beschreibung
Blocks	Weitere Blöcke können hinzugefügt werden.
Text	Einfaches einzeiliges Textfeld
Textarea	Mehrzeiliges Textfeld ohne Formatierung
Richtext	Mehrzeiliges Textfeld mit Formatierungsmöglichkeiten (JSON Format)
Markdown	Mehrzeiliges Textfeld mit Formatierungsmöglichkeiten (Markdown Format)
Number	Nummernfeld ohne Formatierung
Date/Time	Datum und Uhrzeit - Picker
Boolean	Checkbox - true/false
Multi-Options	Liste von Schlüssel-Wert-Paaren, die konfigurierbar sind. Entweder hartcodiert, als externes JSON oder als Datenbank. Mehrere Werte sind auswählbar
Single-Option	Liste von Schlüssel-Wert-Paaren, die konfigurierbar sind. Entweder hartcodiert, als externes JSON oder als Datenbank. Kein oder ein Werte sind auswählbar
Asset	Fileauswahl, die in den Typen limitiert werden kann (nur Images, Videos, Audio, Textdokumente) aber ansonsten alle Typen annimmt.
Multi-Assets	Fileauswahl, die in den Typen limitiert werden kann (nur Images, Videos, Audio, Textdokumente) aber ansonsten alle Typen annimmt. Mehrere Files können hier gleichzeitig ausgewählt werden.
Link	Link auf eine Unterseite aus Storyblok oder als externe URL
Table	In der größte Formatierbare Tabelle, keine Textformatierung möglich

Tabelle 1: Field Types, die von Storyblok bereitgestellt werden

Feldname	Beschreibung
Group	Gruppierungs Folder für die Attribute
Image(old)	Veraltete Option Bilder einzubinden - Asset sollte nun verwendet werden
File(old)	Veraltete Option Files einzubinden - Asset sollte nun verwendet werden
Plugin	Zugriff auf externe Storyblok-Pugins - UI UX differenziert mit jedem Plugin

All diese Attribute haben ein Bearbeitungsfenster, welches von oben nach unten mindestens den Typen, Darstellungsnamen, Technischen Namen, Checkbox - Notwendig, Checkbox - Übersetzbare und Beschreibungsfeld bietet.

Die 2 Wichtigsten sind aber die beiden Checkboxen. Sie ermöglichen uns sicherzustellen, dass der Wert gesetzt werden muss (Checkbox - Notwendig) und dass man ihn falls Notwendig übersetzen kann (Checkbox - Übersetzbare).

Nachstehen werden diese beiden Checkboxen mit den Spalten T für translateable und R für required dargestellt.

## Page

Der Page-Block war schon vorgefertigt von Storyblok und wurde genau so übernommen. Dieser ist sehr simpel aufgebaut. Das einzige Attribut, welches er beinhaltet ist body, ein Blocks Element.

## Article

Article ist der Blok, welcher für News, Projekte, Clubs bzw. Events benutzt wird. Wie Page ist er ein Root Level Block, er besitzt jedoch keinen dynamischen body sondern nur folgende Attribute

Tabelle 2: Attribute des Article Blocks

Attribute	R	T	Beschreibung
headline	✓		Überschrift des Newsbeitrags oder Events
subline		✓	Unterüberschrift
type	✓		Typ des Artikels: News, Event, Projekt, Club
allocate			Zugeordnete Abteilung, mitunter auch Allgemein, Sport oder Reisen
date	✓		Erstellungsdatum, Datum des Events
content	✓		Formatierter Text
image	✓		Titelbild für diesen Beitrag
assets			Weitere Medien
subpage_enabled			Soll ein Link zur vollen Seite dieses Beitrags angezeigt werden

## Branch

Branch spiegelt unsere Abteilungen wieder, er bietet wie Page einen body an Blocks und weitere wichtige Attribute zur richtigen Darstellung der Abteilungen sind:

Tabelle 3: Attribute des Branch Blocks

Attribute	R	T	Beschreibung
headline	✓	✓	Abteilungsname
subline		✓	Kurzbeschreibung der Abteilung
allocate			Zugeordnete Abteilung
imagevideo			Link zum YouTube-Video der Abteilung

Tabelle 3: Attribute des Branch Blocks

Attribute	R	T	Beschreibung
folder			Hinterlegter PDF-Folder der Abteilung
description			Beschreibung der Abteilung

## All Articles

All Articles ist in Storyblok ein relativ simpler Block, er dient dazu alle Articles die zu einer bestimmten Kategorie gehören anzuzeigen.

Tabelle 4: Attribute des All Articles Blocks

Attribute	R	T	Beschreibung
headline	✓	✓	Überschrift
type	✓	Article	Kategorie Auswahl (News,Events,Projekte,Clubs)
filter			Filtermöglichkeit der Articles (default: false)

## Basic Slider

Basic Slider ist wie der Name schon sagt ein einfacher Slider, welcher mit einem Mouse drag bedient wird. Da dieser Effekt aber nicht mehr in unser UI,UX-Design passt könnte es bei der Finalen Version dazu kommen, dass er durch den Scroll Slider ersetzt wird.

Tabelle 5: Attribute des Basic Slider Blocks

Attribute	R	T	Beschreibung
headline	✓	✓	Überschrift

Tabelle 5: Attribute des Basic Slider Blocks

Attribute	R	T	Beschreibung
content	✓		Mindestens 3 Iframe Container sind notwendig, andere Blocks sind nicht erlaubt

## Classes

Classes stellt die Struktur für die Darstellung der Klassen an der HTL dar. Es handelt sich um eine Abgeschlossene Sektion in seinem Content sind nur Class Entry Blocks erlaubt.

Tabelle 6: Attribute des Classes Blocks

Attribute	R	T	Beschreibung
headline	✓	✓	Überschrift
content	✓		Beliebige Anzahl von Class Entry Blocks

## Classes Entry

Classes Entry stellt wirklich genau eine Klasse dar. Gedacht wäre das Schul Klassenfoto als Repräsentation.

Tabelle 7: Attribute des Classes Entry Blocks

Attribute	R	T	Beschreibung
classname	✓		Name der Klasse
img	✓		Klassenfoto der Klasse als Datei (nur Bilder erlaubt)
headofclass			Klassenvorstand der Klasse

## Custom Image

Custom Image dient als Grundlage, wenn im Frontend spezielle Anforderungen an ein Bild gestellt sind. Als Beispiel die Broschüre in Über Uns, diese musste gedreht werden.

Tabelle 8: Attribute des Custom Image Blocks

Attribute	R	T	Beschreibung
image	✓		Darzustellende Bilddatei
type			Single-Select Liste der speziellen Anforderungen, wie es dargestellt werden soll

## Custom Link

Custom Link dient dazu alle Links die existieren gleich zu gestalten. Obwohl ein Link-Attribut in Storyblok existiert haben wir uns nicht dafür ausgesprochen, weil es keine Möglichkeit gibt Symbole oder andere Spezielle Styles zu hinterlegen.

Tabelle 9: Attribute des Custom Link Blocks

Attribute	R	T	Beschreibung
link	✓		Link zu internen oder externen Medien bzw. Seiten
symbol			Auswählbare Symbole, die vor dem Link angezeigt werden
display_name	✓	✓	Anzeigename des Links

## Designable Table

Storyblok bietet ein Plugin, welches es ermöglicht Tabellen darzustellen. Bei diesem Plugin ist es jedoch nicht möglich einen designten Body (fetter und farbiger Text) einzufügen. Aus diesem Grund wurde ein eigener Blok erstellt, welcher diese Probleme beseitigt.

Tabelle 10: Attribute des Designable Table Blocks

Attribute	R	T	Beschreibung
headline		✓	Überschrift
columns		✓	Anzahl der gewünschten Spalten
header		✓	Der 'Table Row' Block wird vorgesehen, die Menge ist dabei auf 1 limitiert
body		✓	Mindestens ein 'Table Row' Block wird vorgesehen

### Table Row

Der Table Row Block, stellt eine Zeile der Tabelle dar. Aus diesem Grund wurde bei den Attributen des Blocks nur ein value hinterlegt, welches wieder 'Blocks' Element ist. Bei diesem ist die Blockauswahl auf 'Table Value' Limitiert.

### Table Value

Der 'Table Value' Block wurde so konfiguriert, sodass die einzige Eingabe ein Richtext-Element ist. Durch dieses wird eine komplette Formatierung gewährleistet.

### Faq Collection

Um die häufig gestellten Fragen und Antworten darstellen zu können wurde eine FAQ Ansammlung erstellt.

Tabelle 11: Attribute des Faq Collection Blocks

Attribute	R	T	Beschreibung
headline		✓	Überschrift
description			Zusätzliche Beschreibung als Richtext
faqs		✓	Beliebige Anzahl von FAQ-Element Blocks

## Faq Element

Faq Element dient wiederum nur zur Befüllung der FAQ Collection.

Tabelle 12: Attribute des Faq Element Blocks

Attribute	R	T	Beschreibung
question	✓	✓	Einfacher Text für die Frage
answer	✓	✓	Richtext für eine formatierte Antwort
video			Link zur Antwort-Video von Direktor DI Richard Kainerstorfer
show_video			Boolescher Wert ob das Video angezeigt werden soll oder nicht (default: false)

## Footer

Footer ist der Block, wie der Name schon sagt, der den Footer der Webseite wiederspiegelt. Seine Attribute beinhalten nur footer\_bg - ein Bild welches zu späziellen Anlässen als Hintergrund verwendet werden kann und columns eine Ansammlung von Footer Col Blöcken.

## Footer Col

Der Footer Col Block ist eine Spalte des Footers - hierbei wurde eine Überschift der spalte (headline) und eine Ansammlung an Weiterführenden Links (links) für die Umsetzung eingerichtet.

## Grid

Der Grid Block wurde so eingerichtet, sodass alle gängigen Grids Konfiguriert werden können. Dafür wurden einige Attribute festgelegt.

Tabelle 13: Attribute des Grid Blocks

Attribute	R	T	Beschreibung
content	✓		Ansammlung von Blöcken, die einen definierten oder konfigurierbaren Column Span haben
columns	✓		Minimale Anzahl der Spalten
mediumcolumns			Spaltenanzahl ab einer Breite von 768px
largecolumns			Spaltenanzahl ab einer Breite von 1024px
max_w			Maximale Breite des Grids
width			Definiert als Column Span (nur notwendig, wenn ein Grid in einem anderen Grid verwendet wird)

## Grid Item

Grid Item bietet den Content für das Grid um vielseitig einsetzbar zu sein sind viele Konfigurationen möglich.

Tabelle 14: Attribute des Grid Item Blocks

Attribute	R	T	Beschreibung
headline	✓		Überschrift
subline	✓		Unterüberschrift
content	✓		Formatierbarer Text als Richtext
width			Festlegung des Column Span des Elements
type			Definiert den Stil des Elements (Ausbildung, Hochformat, Querformat)

Tabelle 14: Attribute des Grid Item Blocks

Attribute	R	T	Beschreibung
content_type			Definiert, ob Main_Image oder eine Animation angezeigt wird
animation			Auswahl aus den verschiedenen Animationen der Abteilungen - nur verfügbar, wenn content_type = animation
main_image			Hauptbild für dieses Grid-Item
image_right			Boolescher Wert (default: false)
link			Link, auf den das gesamte Element verweist
sub_images			Ansammlung von mehreren zusätzlichen Bildern
allocate			Zuordnung zu den Abteilungen

## Headline

Headline bietet die Möglichkeit Blöcken ohne Überschrift oder Abschnitten eine passende Überschrift zu geben. Sie beinhaltet nur die Attribute headline und no\_spacing\_y.

## Hero

Hero definiert die Oberste Sektion in all unseren Seiten ausgenommen der Startseite. Die Besonderheit ist das Schräge bild und der zusätzlich mögliche Content.

Tabelle 15: Attribute des Hero Blocks

Attribute	R	T	Beschreibung
headline	✓	✓	Große Überschrift
background_image	✓		Bild, welches hinter der Überschrift liegt

Tabelle 15: Attribute des Hero Blocks

Attribute	R	T	Beschreibung
type	✓		Definiert das Aussehen des Hero Elements
height	✓		Setzt die Höhe der Sektion
fixed			Boolescher Wert, der die Position auf absolut setzt (default: false)
hero_features	✓		Blocks Attribut, das mit den Hero Feature Blöcken befüllt werden kann
additional_content			Wird nur bei News, Events und weiteren verwendet, der gesamte Inhalt befindet sich somit in der Hero Component

## Hero Feature

Zusätzliche Information kann hiermit neben dem Hero Block angezeigt werden. Dieser ist wieder konfigurierbar um verschiedene Styles und Funktionen zu ermöglichen.

Tabelle 16: Attribute des Hero Feature Blocks

Attribute	R	T	Beschreibung
symbol			Bietet eine Auswahl an Symbolen an, die davor angezeigt werden können
type	✓		Definiert die Darstellung entweder groß oder klein
text			Dargestellter Text
link			Falls notwendig, kann ein weiterführender Link angegeben werden

## IFrame Container

Um IFRAMES per Storyblok einbauen zu können wurde der IFrame Container Block erstellt, dieser bietet durch den Link die Möglichkeit beliebige IFRAMES wie Videos, Maps oder Ähnliches einzubinden.

Tabelle 17: Attribute des IFrame Container Blocks

Attribute	R	T	Beschreibung
headline	✓	✓	Überschrift
iframe_content	✓		Link zu dem einzubindenden IFrame
links			Ansammlung von Custom Link Blocks
additional_info		✓	Formatierbarer Text als Richtext

## Link Collection

Ansammlung von Custom Link Blocks falls notwenig auch mit Überschrift

## Marquee

Marquee Block bietet die Möglichkeit Content Horizontal automatisch scrollen zu lassen, so wie in einer Slideshow.

Tabelle 18: Attribute des Marquee Blocks

Attribute	R	T	Beschreibung
content	≥ 4		Ansammlung von Blöcken, die bewegt werden
reverse_direction			Dreht die Richtung der Marquee um (default: false)
spacing_top			Abstand oben (default: false)
spacing_bottom			Abstand unten (default: false)

Tabelle 18: Attribute des Marquee Blocks

Attribute	R	T	Beschreibung
display_all_mobile			Wird der Marquee oder eine vertikale Liste angezeigt (Nur auf Mobilgeräten, default: false)
speed			Definiert die Geschwindigkeit der Marquee

## Navbar

So wie der Footer Block ist auch der Navbar Block die Implementierung der Navbar unserer Webseite.

Tabelle 19: Attribute des Navbar Blocks

Attribute	R	T	Beschreibung
logo	✓		Logo, welches angezeigt wird
logo_dark	✓		Logo, welches im Darkmode angezeigt wird
middle_nav			Ansammlung von den hauptsächlichen Links
side_nav			Ansammlung von Links auf der rechten Seite

## Scroll Slider

Der Scroll Slider Block ist einer unserer Hauptfeatures. Durch ihn wird der horizontale Scroll in einen vertikalen Scroll umgewandelt.

Tabelle 20: Attribute des Scroll Slider Blocks

Attribute	R	T	Beschreibung
title	✓	✓	Überschrift

Tabelle 20: Attribute des Scroll Slider Blocks

Attribute	R	T	Beschreibung
slider_table			Tabelle, die zusätzliche Informationen anzeigen kann
scroll_start_right			Festlegung, ob der Content der Start ist oder nichts
slider			Blocks-Element zur Darstellung der Slider-Blöcke (nur Slider Item)
alternating			Boolescher Wert, der festlegt, ob die Blöcke zentriert sind oder abwechselnd auf der Y-Höhe verschoben sind (default: false)
scroll_speed			Legt die virtuelle Höhe des Elements fest und somit die Schnelligkeit der seitlichen Verschiebung während des Scrollens

### Scroll Slider Select

Scroll Slider Select bietet die selbe Funktionalität wie der Scroll Slider, der Content wird aber nicht durch Blocks sondern durch das auswählen beliebiger 'Article' Blocks festgelegt.

Tabelle 21: Attribute des Scroll Slider Select Blocks

Attribute	R	T	Beschreibung
title	✓	✓	Überschrift
show_title_animation			Soll die Startanimation (Logo-Schultransformation) abgespielt werden (default: false)
scroll_start_right			Festlegung, ob der Content der Start ist oder nichts

Tabelle 21: Attribute des Scroll Slider Select Blocks

Attribute	R	T	Beschreibung
slider			Multi-Option zum Wählen der gewünschten Stories. Nur Stories mit dem Typen "Article" werden akzeptiert
scroll_speed			Legt die virtuelle Höhe des Elements fest und somit die Schnelligkeit der seitlichen Verschiebung während des Scrollens

## Site Jump

Site Jump bietet die Möglichkeit zu verschiedenen Überschiften auf der selben Seite zu springen. Diese müssen als Jumplink in den Blocks Attribut 'headlines' angegeben werden und mit den zugeordneten headlines gleich sein.

## Slider Item

Slider Item realisiert die einzelnen Contentbausteine von Scroll Slider.

Tabelle 22: Attribute des Slider Item Blocks

Attribute	R	T	Beschreibung
headline	✓	✓	Überschrift
subline		✓	Unterüberschrift
content		✓	Formatierbarer Text als Richtext
image			Hauptbild für dieses Element
allocate			Zuordnung zu den Abteilungen
type	✓		Definiert den Style des Elements (Event, Kontakt groß/klein, Bewerbung)

## Spacer

Der Spacer Block bietet die Möglichkeit falls notwendig mehr platz zwischen 2 Anderen Blöcken auf Y-Ebene einzufügen. Hierfür stehen folgende rem-Werte (2,4,6,8) im Size Attribut bereit.

## Sponsor

Der Sponsor Block wurde entwickelt, da die Sponsoren ein anderes Design benötigen als es mit einem einfachen GridItem Block möglich ist. Die Besonderheit liegt im Ausblenden des zusätzlichen Kontents.

Tabelle 23: Attribute des Sponsor Blocks

Attribute	R	T	Beschreibung
headline		✓	Name der Firma / des Sponsors
subline		✓	Zusätzliche Info zum Sponsor
image		✓	Logo des Sponsors
link			Link zur gewünschten Adresse
only_image			Boolesche Auswahl, ob nur das Logo oder auch die headline und subline angezeigt werden sollen (default: false)

## Table

Wenn man einen relativ simplen Table benötigt ohne nennenswerte Textformatierung, dann bietet sich dieser Table Block an. Er beinhaltet nur den standardmäßigen Tabel bereitgestellt von einem Storyblok Plugin.

Tabelle 24: Attribute des Table Blocks

Attribute	R	T	Beschreibung
headline		✓	Überschrift über der Tabelle

Tabelle 24: Attribute des Table Blocks

Attribute	R	T	Beschreibung
table	✓		Ein Table als Plugin wird verwendet, dieser kann dann nach Belieben bearbeitet und befüllt werden

## Text

Um textuellen Content wie in Datenschutz und auf dem Impressum mit Storyblok editierbar zu machen wurde der Text Block implementiert.

Tabelle 25: Attribute des Text Blocks

Attribute	R	T	Beschreibung
headline	✓		Überschrift
divider			Boolescher Wert, ob ein Divider zwischen headline und content angezeigt werden soll (default: false)
content	✓		Formatierbarer Text als Richtext
max_w	✓		Legt die maximale Breite des Elements fest

## Timetable

Der Timetable Block bietet die Möglichkeit einen Stundenplan der HTL digital nachzustellen. Die Attribute sind headline und eine Ansammlung von Timetable Cols Blocks (columns).

### Timetable Cols

Die Timetable Cols spiegeln einen Tag im Stundenplan wieder. Sie beinhalten daher nur einen Text für den Tag (day) und eine Sammlung von Timetable Subject Blocks (subjects).

## Timetable Subject

Timetable Subject ist das wichtigste und am meisten konfigurierbar am Timetable.

Tabelle 26: Attribute des Timetable Subject Blocks

Attribute	R	T	Beschreibung
name	✓		Name des Faches
short_name	✓		Kurzbezeichnung des Faches
description		✓	Beschreibung des Faches
is_specialized_subject			Legt fest, ob dieses Fach abteilungsspezifisch ist
main_image	✓		Bild, welches das Fach beschreibt
color			Farbe des Faches, wie in Untis
jahrgang1-5			Nummerischer Wert, der die Stundenzahl in den einzelnen Jahren beschreibt

## Value Document

Der Value Document Block ist für die Representation des Wertedokuments an der HTL-Leonding. Es wurde eine eigener Block erstellt um ein spezielles Design mit einer Schrägen umsetzen zu können.

Tabelle 27: Attribute des Value Document Blocks

Attribute	R	T	Beschreibung
headline	✓		Überschrift
subline	✓		Unterüberschrift
content			Genau 5 Value_document_entry Blöcke werden vorausgesetzt
background_image			Hintergrundbild für das Dreieck auf der Seite

## Value Document Entry

Value Document Entry ermöglicht einen Eintrag in das Wertedokument. Die Attribute belaufen sich nur auf die Kernaussage des Wertes 'headline' und auf eine detaillierte Beschreibung im Richtext-Format (description)

# 4.2 Next

Am Start dieses Projekts war Next.js 13 die neueste Version und somit wurde dieses Projekt auch mit Next.js 13 in Verbindung mit dem neuen App Router erstellt. Dies geschah durch den Befehl `npx create-next-app@latest` dadurch war auch die gesamte Ordnerstruktur des Projekts fertig und die Template Seite konnte mit `npm run dev` gestartet und auf **localhost:3000** aufgerufen werden.

## 4.2.1 Struktur

### Folder

Die von Storyblok generierten Folder sind **public** und **app**. Der public Ordner ist dabei für statische Assets verantwortlich. In unserer Implementierung wurde er nur für das Schullogo und eine Illustration der Schule verwendet. Der app Ordner ist aber der wichtigste in dem ganzen Projekt er ist wie schon vorher erwähnt zuständig für das Routing der gesamten Webseite.

Speziell von uns erstellt wurden die Ordner **animations**, **components**, **types** und **util** um eine bessere Übersicht über das Projekt zu erhalten. In den Ordner befinden sich dann die dazugehörigen Dateien.

Unter **animations** sind dies alle exportierten Animation die auf der Webseite angezeigt werden sollen im JSON-Format.

Unter **components** sind alle erstellen Components, wiederum aufgeteilt in Unterordner um eine Struktur zu bewahren. Das wichtigste File in components ist aber der **StoryblokProvider.tsx**, dieser ermöglicht die Verbindung zwischen Storyblok und Next.

Unter **types** ist das File **interface.tsx** zu finden in diesem sind alle TS-Typen für das ganze Projekt festgehalten.

Unter **util** befinden sich dann noch 2 Files, welche rein Technische TS Funktionen beinhalten.

## Files

Files im Root-Level des Projekts dienen hauptsächlich zur Konfiguration des Projekts. Dabei handelt es von den Files **.env.local**, **middleware.js**, **next-env.d.ts**, **next.config.js**, **package.json** und **tsconfig.json**.

Bei **.env.local** handelt es sich um das File, welches im lokalen betrieb (npm run dev), die ganzen Konfigurationen beinhaltet in unseren konkreten fall ist dies nur der **STORYBLOK\_API\_KEY** und die **STORYBLOK\_API\_VERSION**.

In **middleware.js** wird eine Middleware ausgeführt, welche den Standort und im Zusammenhang damit die Sprache der Seite einstellt. Dafür wird am start der URL ein /de oder ein /en eingefügt falls nichts genannt und auch keine besonderen Optionen bekannt sind wird standardmäßigen /de verwendet.

**next-env.d.ts** ist die TS Deklaration für Next.js

In **next.config.js** findet die Konfiguration des ganzen Projekt statt.

In **package.json** befinden sich die Abhängigkeiten und Skripte des Projekts.

**tsconfig.json** ist das Konfigurationsfile für TypeScript.

### 4.2.2 Routing

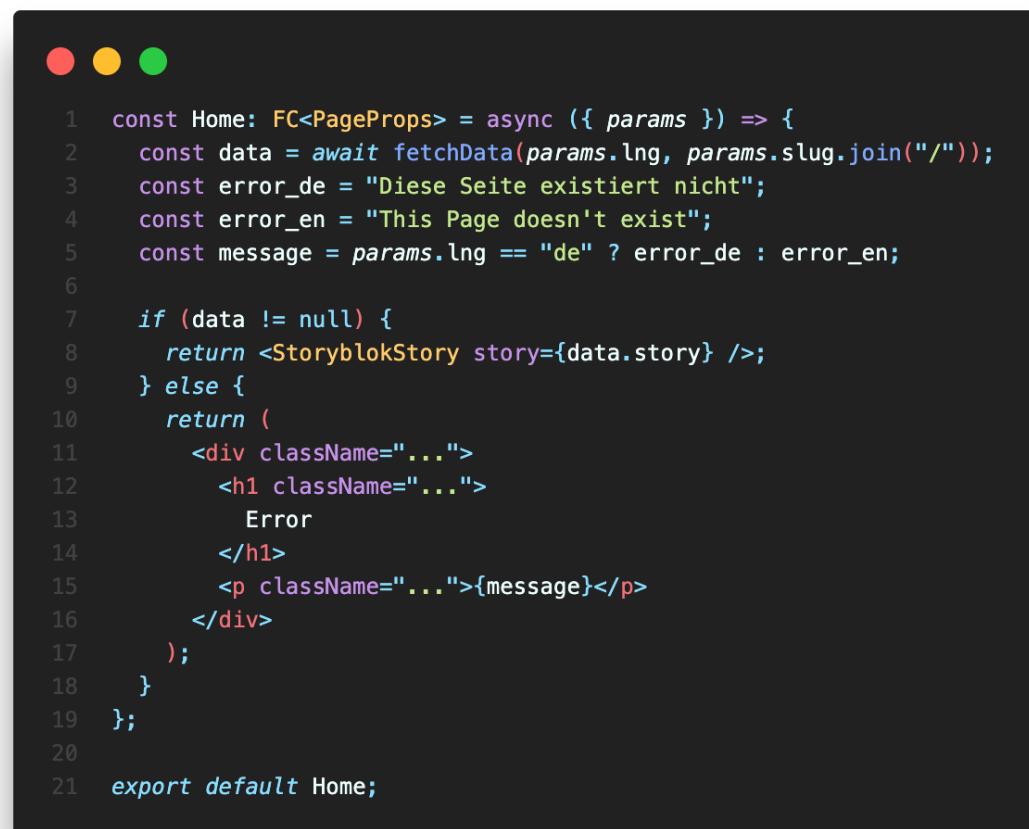
Um die Rute mywebsite.com/products/shoes/newest zu realisieren müsste man im App Router (Ordner app) 3 Unterordner erstellen sodass folgenden Ordnerstruktur entsteht app/products/shoes/newest. Bei Statischen Seiten, wo der Content als HTML gepflegt wird macht dies auch Sinn. All das was man nacher auf der Webseite sehen will muss in das 'page.tsx' File an der Richtigen Stelle. Da es sich aber bei Storyblok um dynamisch generierte Seiten handelt, bei denen man auch zusätzlich schon die Pfadstruktur anhand der Ordner und Seiten in Storyblok festlegt wurde dieser Ansatz nicht geweht. Bei Änderungen der Struktur in Storyblok könnte es sein, dass die Seite gar nichtmehr funktioniert.

Die Lösung dafür waren dynamische Routen realisierbar durch Ordner, bei denen der name in Eckigen Klammern stand ein sogenannter [Dynamic Segment] Ordner. Bei

diesem Ordner handelt es sich um einen Teilweise Dynamischen Pfad, er macht nämlich nur diese eine Stelle im Pfad dynamisch. Ein Beispiel dafür wäre app/blog/[slug]/page.tsx Pfade wie /blog/a /blog/b blog/b funktionieren alle, wenn man aber /blog/a/b eingibt ist dies nicht mehr möglich.

Um dies zu ermöglichen wurde ein Catch-all Statement verwendet dies ermöglicht ein Match aller weiteren Elemente. Dies wurde durch einen Ordner mit dem Namen [...slug] realisiert.

Die Ordnerstruktur des Projekt sieht also am Ende so aus /app/[lng]/[...slug]. Der [lng] Ordner wurde dabei absichtlich nicht in [...slug] inkludiert, damit alle Subpages einen eindeutigen Zugriff auf den lng Parameter haben. Um nun alle Seiten darstellen zu können muss nur mehr die Page.tsx erstellt werden, welche im finalen Aufbau sehr simpel erscheint:



```
1  const Home: FC<PageProps> = async ({ params }) => {
2    const data = await fetchData(params.lng, params.slug.join("/"));
3    const error_de = "Diese Seite existiert nicht";
4    const error_en = "This Page doesn't exist";
5    const message = params.lng === "de" ? error_de : error_en;
6
7    if (data != null) {
8      return <StoryblokStory story={data.story} />;
9    } else {
10      return (
11        <div className="...">
12          <h1 className="...">
13            Error
14            </h1>
15            <p className="...">{message}</p>
16          </div>
17        );
18    }
19  };
20
21  export default Home;
```

Abbildung 10: Finales page.tsx

### 4.2.3 Storyblok

Nachdem die gerade erstelle Next13 Seite erfolgreich funktionierte wurde das Storyblok Tutorial durchgearbeitet. Im späteren verlauf der Arbeit stellten sich aber immer mehr Probleme im Bezug zu der am Anfang erstellten Verbindung, sodass in der Mitte der Arbeit die Verbindung verändert wurde und am Schluss die komplette API-Schnittstelle selbst implementiert wurde. Nicht aber nur das 5 Minuten Start Tutorial war noch nicht auf dem neuesten Stand sondern auch der Großteil der restlichen Tutorails, welche das Zusammenspiel von Next13 und Storyblok erklären sollten waren nicht auf den Aktuellsten Stand beziehungsweise nicht auf den App Router Ansatz ausgelegt.

#### Verheiraten mit Next

Nach mehreren Versuchen wurde dann der Storyblok Provider Ansatz gewählt. Dabei handelt es sich um ein Zusammenspiel einer Client Slide Component namens Storyblok-Provider und dem Root Level layout.tsx file. Beide Files rufen dabei die storyblokInit() Funktion auf.

Der StoryblokProvider übernimmt dabei die Aufgabe des Linken der Components zu den Blocks von Storyblok.



```
1 import Page from "@/components/Page";
2 import Branch from "@/components/Branch";
3 import Hero from "@/components/Hero";
4 import Navbar from "@/components/Navbar";
5 import Footer from "@/components/Footer/Footer";
6 import FooterCol from "@/components/Footer/FooterColumn";
7 import Link from "@/components/Basic/Link";
8 ...
```

Abbildung 11: Ausschnitt StoryblokProvider.tsx

Alle Componenten werden importiert und mit einem passenden Namen versehen. In einer Variable in diesem Fall const components werden dann alle Storyblok Blöcke den Componenten zugeordnet.

```
1  const components = {
2    ...
3    page: Page,
4    branch: Branch,
5    custom_link: Link,
6    hero: Hero,
7    navbar: Navbar,
8    footer: Footer,
9    footer_col: FooterCol,
10   ...
11};
```

Abbildung 12: Ausschnitt StoryblokProvider.tsx

Diese Ansammlung von Componenten kann dann im der storyblokInit Funktion mitgegeben werden um Storyblok mit den Componenten zu verbinden.

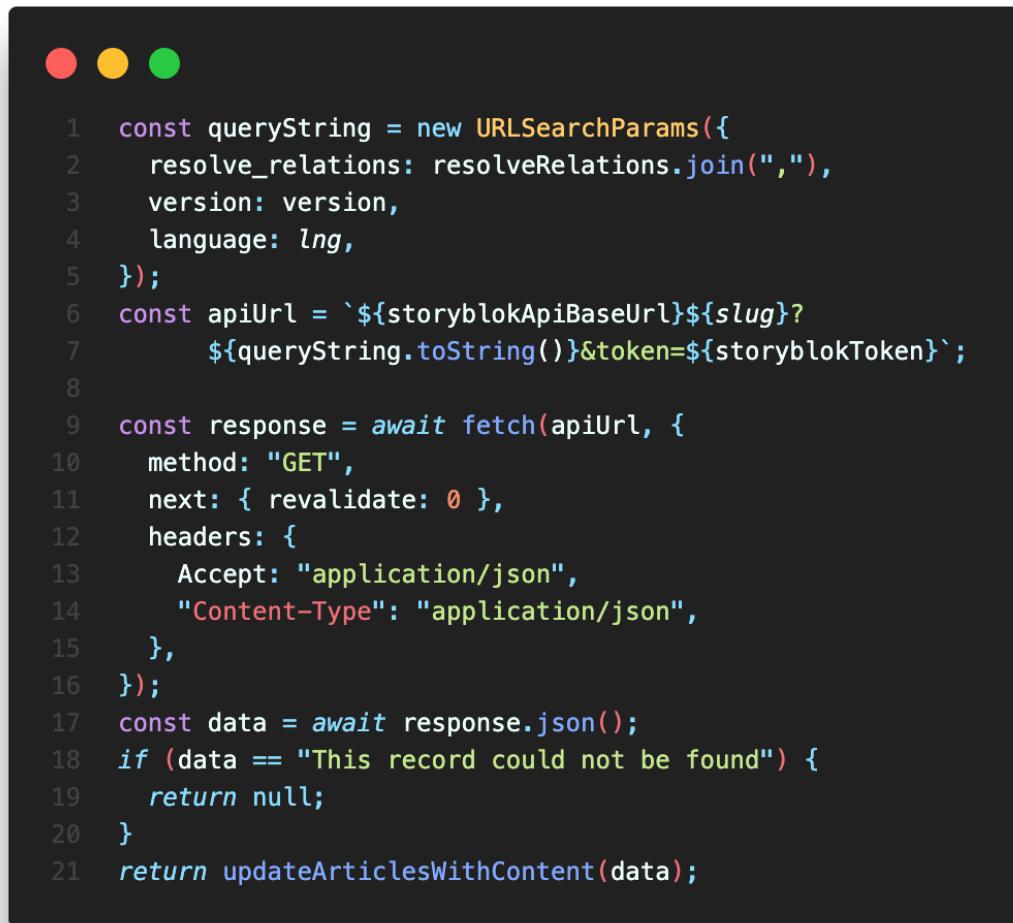
```
1  storyblokInit({
2    accessToken: process.env.storyblokApiToken,
3    use: [apiPlugin],
4    components,
5  });
6
7  export default function StoryblokProvider({
8    children,
9  }: {
10    children: React.ReactNode;
11  }) {
12    return children;
13}
```

Abbildung 13: Ausschnitt StoryblokProvider.tsx

## Datenabfrage

Da die von Stroyblok/react/rsc bereitgestellte storyblokApi.get() Funktion keine Next revalidation unterstützt, wurde der GET Request mittels JS fetch ausprogrammiert. Dabei trat das größte Problem bei den Sub-Stories auf. Sonst können diese durch die einfache Angabe der richtigen resolve\_relations Parameter gelöst werden. Wenn aber der API zugriff manuell erfolgt, befinden sich die 'Aufgelösten Beziehungen' ganz am Ende des JSONs und müssen manuell an die richtige Stelle transformiert werden.

Dies wurde durch 2 Funktionen in einem Util File namens storyblok.tsx gelöst. Wobei die fetchData im Grunde genommen nur einen Fetch-Request an die API schickt und dass Ergebnis, dann im return an die weitere Funktion weitergibt.



```

1  const queryString = new URLSearchParams({
2    resolve_relations: resolveRelations.join(","),
3    version: version,
4    language: lng,
5  });
6  const apiUrl = `${storyblokApiBaseUrl}${slug}?
7    ${queryString.toString()}&token=${storyblokToken}`;
8
9  const response = await fetch(apiUrl, {
10    method: "GET",
11    next: { revalidate: 0 },
12    headers: {
13      Accept: "application/json",
14      "Content-Type": "application/json",
15    },
16  });
17  const data = await response.json();
18  if (data == "This record could not be found") {
19    return null;
20  }
21  return updateArticlesWithContent(data);

```

Abbildung 14: Fetch an Stroyblok-API

Die andere Funktion, welche man am Ende dieses Codeausschnitts sehen kann ist aber von weitaus größerer Bedeutung. Sie löst das vorhin angesprochene Problem der 'resolve

Relations'. Sie speichert alle Verweise und die dazugehörigen originalen Daten. Folgend werden alle Elemente in der Story überprüft ob sie den passenden UUID beinhalten, wenn ja wird der Content ausgetauscht. Diese Erklärung entspricht nicht ganz der Realität, da in der Realität zwischenabfragen die Performance erhöhen somit wird nur die UUID verglichen wenn es sich um das Attribut atricles oder slider handelt.



```

1  function updateArticlesWithContent(data: any): any {
2      const relsMap: Record<string, any> = {};
3
4      data.rels?.forEach((rel: any) => {
5          relsMap[rel.uuid] = rel;
6      });
7
8      data.story?.content.body?.forEach((section: any) => {
9          if (section.articles) {
10              section.articles = section.articles.map(
11                  (articleUUID: string) => relsMap[articleUUID],
12              );
13          }
14          if (section.slider && typeof section.slider[0] == "string") {
15              section.slider = section.slider.map(
16                  (sliderUUID: string) => relsMap[sliderUUID],
17              );
18          }
19      });
20
21      return data;
22  }

```

Abbildung 15: Datentransformation - Storyblok Fetch

#### 4.2.4 Packages

Da es sich bei diesem Projekt um eine Node basierte Architektur handelt wurden einige Packages als Erweiterungen installiert. Packages, welche in Kombination mit Storyblok stehen werden hier nicht angeführt und werden in den Unterkapitel Storyblok näher beschrieben werden. Auch Packages wie react und typescript werden hier nicht behandelt, da sie nicht explizit hinzugefügt worden sind und somit die Basis des Projektes bilden.

## Framer Motion

Framer Motion ist eine Animationslibery für React. In dieser Diplomarbeit übernimmt Framer Motion alle Animationen mit ausnahme von den SVG Animationen. Diese werden von React-Lottie abgespielt.

Bevor man Framer Motion verwenden kann muss man es installieren, da es ein Node Package ist, ist dies nicht schwierig: *npm install framer-motion*.

Die Verwendung von Framer Motion ist simpel überall dort wo man es verwenden möchte muss man es importieren: *import { motion } from "framer-motion"*. Wenn man also nun aus einem html Tag ein Motion html Tag machen will muss man vor den Tag ein **motion**. schreiben als Beispiel ein <div> wird zu einem <motion.div>.

## TailwindCss

In Kombination mit postess autoprefixer ...

## React Lottie

... Lottie einbinden

## React Icons

... Icons einbinden

## React Fast Marquee

... Libery, die eine Marquee kinfigurierbare Marquee bereitstellt.

### 4.2.5 Komponenten

Für fast jeden Block im Storyblok-Backend wurde in Next eine Komponente erstellt. Die Daten werden dann mithilfe von TypeScript typisiert übergeben und können so richtig dargestellt werden. Da die meisten Komponenten sehr logisch Strukturiert und aufgebaut sind wird hier nicht mehr weiter darauf eingegangen. Die folgenden Komponenten die dennoch beschrieben werden weichen von den vorher angemerkt Eigenschaften ab, da sie spezielle Berechnungen beinhalten, eine andere Datenstruktur aufweisen und die Hauptaugenmerke unserer Webseite darbieten.

## **ScrollSlider und ScrollSliderSelect**

Die wichtigste und die auffälligste Komponente ist dabei der ScrollSlider bzw. der ScrollSlider mit Select beziehungsweise Latest Funktion. Diese Kombination bieten die Möglichkeit den Scroll von der Y-Achse in die X-Achse zu transformiert. Man scrollt also nicht mehr wie gewohnt von Oben nach Unten sondern von Links nach Rechts. Dies bietet einerseits die Möglichkeit den Benutzer auf den angezeigten Content zu fokussieren indem der Flow der Webseite unterbrochen wird.

### **All Articles**

#### **Hero**

#### **ValueDokument ValueDokumentEntry**

#### **Timetable**

### **4.2.6 Internationalisierung**

### **4.2.7 Deployment**

## **4.3 Animationen**

### **4.3.1 Erstellung**

### **4.3.2 Einbindung**

# 5 Zusammenfassung

Aufzählungen:

- Itemize Level 1
  - Itemize Level 2
    - Itemize Level 3 (vermeiden)
- 1. Enumerate Level 1
  - a. Enumerate Level 2
    - i. Enumerate Level 3 (vermeiden)

**Desc** Level 1

**Desc** Level 2 (vermeiden)

**Desc** Level 3 (vermeiden)

# Glossar

**GUID** Globally Unique Identifier

# **Literaturverzeichnis**

# Abbildungsverzeichnis

1	Startseite Ausgangssituation	2
2	Problemanalyse	6
3	Zielgruppenanalyse	6
4	UserInnen Gefühle	7
5	UserInnen Missionen	7
6	erste Entwürfe	8
7	Entwurf mit Figma	9
8	alter Header	10
9	neuer Header	10
10	Finales page.tsx	37
11	Ausschnitt StoryblokProvider.tsx	38
12	Ausschnitt StoryblokProvider.tsx	39
13	Ausschnitt StoryblokProvider.tsx	39
14	Fetch an Stroyblok-API	40
15	Datentransformation - Storyblok Fetch	41

# Tabellenverzeichnis

1	Field Types, die von Storyblok bereitgestellt werden . . . . .	17
1	Field Types, die von Storyblok bereitgestellt werden . . . . .	18
2	Attribute des Article Blocks . . . . .	19
3	Attribute des Branch Blocks . . . . .	19
3	Attribute des Branch Blocks . . . . .	20
4	Attribute des All Articles Blocks . . . . .	20
5	Attribute des Basic Slider Blocks . . . . .	20
5	Attribute des Basic Slider Blocks . . . . .	21
6	Attribute des Classes Blocks . . . . .	21
7	Attribute des Classes Entry Blocks . . . . .	21
8	Attribute des Custom Image Blocks . . . . .	22
9	Attribute des Custom Link Blocks . . . . .	22
10	Attribute des Designable Table Blocks . . . . .	23
11	Attribute des Faq Collection Blocks . . . . .	23
12	Attribute des Faq Element Blocks . . . . .	24
13	Attribute des Grid Blocks . . . . .	25
14	Attribute des Grid Item Blocks . . . . .	25
14	Attribute des Grid Item Blocks . . . . .	26
15	Attribute des Hero Blocks . . . . .	26
15	Attribute des Hero Blocks . . . . .	27
16	Attribute des Hero Feature Blocks . . . . .	27
17	Attribute des IFrame Container Blocks . . . . .	28
18	Attribute des Marquee Blocks . . . . .	28
18	Attribute des Marquee Blocks . . . . .	29
19	Attribute des Navbar Blocks . . . . .	29
20	Attribute des Scroll Slider Blocks . . . . .	29
20	Attribute des Scroll Slider Blocks . . . . .	30
21	Attribute des Scroll Slider Select Blocks . . . . .	30
21	Attribute des Scroll Slider Select Blocks . . . . .	31
22	Attribute des Slider Item Blocks . . . . .	31
23	Attribute des Sponsor Blocks . . . . .	32
24	Attribute des Table Blocks . . . . .	32
24	Attribute des Table Blocks . . . . .	33
25	Attribute des Text Blocks . . . . .	33
26	Attribute des Timetable Subject Blocks . . . . .	34
27	Attribute des Value Document Blocks . . . . .	34

# **Quellcodeverzeichnis**

# **Anhang**