

communicated by:  
Lehr- und Forschungsgebiet Informatik 9

Prof Dr. Ulrik Schroeder



**RWTH**AACHEN  
UNIVERSITY

**Diese Arbeit wurde vorgelegt am Lehr- und Forschungsgebiet Informatik 9**

**The present work was submitted to Learning Technologies Research Group**

Kontextbasierte UI-Anpassung für zielgruppenspezifische Anforderungen an digitale Gamebooks

Context-based UI adaptation for target group-specific requirements of digital gamebooks

Bachelorarbeit

Bachelor-Thesis

von / presented by

Dietrichs, Jan

394439

Ulrik Schroeder

Simon Völker

Aachen, 21. März 2023



---

# Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>iii</b>
<b>Tabellenverzeichnis</b>	<b>v</b>
<b>Quellcodeverzeichnis</b>	<b>vii</b>
<b>I. Einleitung</b>	<b>3</b>
<b>1. Motivation</b>	<b>5</b>
1.1. Gamebooks für die digitale Lehre . . . . .	5
1.2. Zielgruppenspezifische Anforderungen . . . . .	5
1.3. Kontextbasierte UI-Anpassung . . . . .	6
<b>2. Ziel der Arbeit</b>	<b>7</b>
2.1. Verwandte Arbeiten . . . . .	7
2.2. Vorgehensweise . . . . .	8
<b>II. Anforderungen an E-Learning-Plattformen</b>	<b>9</b>
<b>3. Identifizierung von Nutzergruppen</b>	<b>11</b>
3.1. Akzeptanz durch Personalisierung . . . . .	11
3.2. Unterstützende UI-Elemente . . . . .	12
3.3. Barrierefreiheit . . . . .	14
<b>4. Technische Umsetzung</b>	<b>17</b>
4.1. Willkommensmenü . . . . .	17
4.2. Konfigurationen . . . . .	19
4.3. Bereitgestellte Browserfunktionen . . . . .	20
<b>III. Erweiterung der Gamebook-UI</b>	<b>21</b>
<b>5. Implementierung der Features</b>	<b>23</b>
5.1. Angular . . . . .	23
5.2. Backendkommunikation . . . . .	25
5.3. Die gewählten Erweiterungen . . . . .	26
<b>6. Implementierung des Willkommensmenüs</b>	<b>33</b>
6.1. Einstellungsmasken . . . . .	33

6.2. Kontextbasierte Anpassungen . . . . .	34
<b>IV. Ausblick und Zusammenfassung</b>	<b>37</b>
<b>7. Ausblick</b>	<b>39</b>
7.1. Neue UI-Anpassungen . . . . .	39
7.2. Neue Gamebook-Elemente . . . . .	40
7.3. Vorschläge für zukünftige Erweiterungen . . . . .	40
<b>8. Zusammenfassung</b>	<b>43</b>
<b>Anhänge</b>	<b>43</b>
<b>A. Literaturverzeichnis</b>	<b>45</b>

---

# Abbildungsverzeichnis

4.1. Flow-Chart die den Ablauf des Willkommensmenüs zeigt. (Eigene Darstellung) . . . . .	18
5.1. UI im Light-Theme mit erstem Farbschema . . . . .	26
5.2. UI im Light-Theme mit letztem Farbschema . . . . .	26
5.3. UI im Dark-Theme im ersten Farbschema . . . . .	27
5.4. Klassen des Headers . . . . .	27
5.5. UI mit kleiner Schriftgröße . . . . .	28
5.6. UI mit größerer Schriftgröße . . . . .	28
5.7. UI im Hochkontrastmodus . . . . .	30
5.8. Fokus auf Schaltfläche im Hochkontrastmodus . . . . .	30
6.1. Willkommensmenü mit eingestellter kleiner Schriftgröße . . . . .	33
6.2. Willkommensmenü mit eingestellter größerer Schriftgröße . . . . .	33
6.3. Willkommensmenü mit deaktiviertem Schalter . . . . .	34
6.4. Willkommensmenü mit aktiviertem Schalter . . . . .	34
6.5. Auswahl des Erscheinungsbildes im Light-Theme . . . . .	34
6.6. Auswahl des Erscheinungsbildes im Dark-Theme . . . . .	34



---

# Tabellenverzeichnis



---

# Quellcodeverzeichnis

5.1. Aus beispiel.component.scss . . . . .	24
5.2. Aus uiconfig.service.ts . . . . .	24
5.3. Aus beispiel.component.ts . . . . .	25
5.4. Aus beispiel.component.html . . . . .	25
5.5. Aus backend/models/user.js . . . . .	25
5.6. Aus backend/routes/user.js . . . . .	25
5.7. Aus backend/controllers/user.js . . . . .	26
5.8. Aus header.component.html . . . . .	27
5.9. Aus header.component.scss . . . . .	27
5.10. Aus borderbtn.component.ts . . . . .	29
5.11. Aus borderbtn.component.html . . . . .	29
5.12. Aus borderbtn.component.scss . . . . .	29
5.13. Aus header.component.scss . . . . .	30
5.14. Aus borderbtn.component.scss . . . . .	31
5.15. Aus borderbtn.component.html . . . . .	31



---

# Abstract

Diese Bachelorarbeit beschäftigt sich mit der UI-Erweiterung einer E-Learning-Plattform für digitale Gamebooks. Es wird gezeigt, welche unterschiedlichen Bedürfnisse verschiedene Zielgruppen an eine solche Plattform haben. Für diese Anforderungen werden UI-Anpassungen vorgeschlagen, die abhängig vom Nutzungskontext aktiviert werden können. Ziel dieser kontextbasierten Konfigurationen ist es, Optimierungskonflikte durch zielgruppenspezifische Anforderungen zu vermeiden und die Zielgruppe der Gamebook-Plattform somit zu vergrößern. Diese Erweiterungen werden so implementiert, dass sowohl neue Gamebook-Elemente leicht unterstützt, als auch neue UI-Anpassungen ergänzt werden können.

This bachelor's thesis deals with the UI extension of an e-learning platform for digital gamebooks. It is shown which different needs different target groups have for such a platform. For these requirements, UI adaptations are proposed that can be activated depending on the context of use. The goal of these context-based configurations is to avoid optimization conflicts due to target group specific requirements and thus to increase the target group of the gamebook platform. These extensions are implemented in such a way that new Gamebook elements can be easily supported as well as new UI customizations can be added.



---

# Teil I

## Einleitung



---

# Kapitel 1 Motivation

Um Verständnis für das Thema dieser Arbeit aufzubauen, sollen in diesem Kapitel zunächst die drei Hauptbegriffe des Titels erläutert werden. Dies geschieht in der umgekehrten Reihenfolge: Gamebooks, Anforderungen und UI-Anpassungen.

## 1.1. Gamebooks für die digitale Lehre

In der klassischen Lehrmethode des Frontalunterrichts wird Wissen in der Regel in einer linearen Abfolge vermittelt. Die Lehrkraft stellt die Lerninhalte mittels verbaler Darstellungen vor und stellt allenfalls gezielte Fragen an die Lernenden um den Ablauf des Unterrichts zu steuern.[8] Bei (Lehr-) Büchern ist dies ebenso der Fall. Die Lesenden werden vom Anfang zum Ende der Geschichte geleitet und haben keinen Einfluss auf den Ablauf, der einen alternativen, sinnvollen Ablauf ergeben würde. Diese Arbeit beschäftigt sich mit einer Sonderform von Büchern. Gamebooks sind Bücher, die einen individuellen Handlungsspielraum mit sich bringen. An verschiedenen Stellen der Geschichte wird den Lesenden eine Auswahl von Möglichkeiten geboten, wie die Handlung weiterverlaufen kann. Die Lernenden entscheiden sich für eine Variante und blättern an die entsprechend angegebene Stelle im Buch. Der Handlungsspielraum ist hier durch die Anzahl der Entscheidungspunkte beschränkt, die der Autor bereit stellt. Dieses Konzept lässt sich auch digital um- und in der Lehre einsetzen. Die digitale Umsetzung von Gamebooks erweitert die Interaktionsmöglichkeiten mit denen die Geschichte beeinflusst werden kann. Ein gezielter Einsatz der Entscheidungsknoten lässt den Autor verschiedene Funktionen, wie thematische Schwerpunkte, Fehlerwiederholung, oder Schwierigkeitsstufen, umsetzen. Die Partizipation der Lernenden wiederum ermöglicht nicht nur die Differenzierbarkeit der Lerninhalte auf verschiedene Lernniveaus, sie fördert auch die Akzeptanz.[15] Somit sind Gamebooks gut für das Selbststudium geeignet, welches neue Hürden wie mangelnde extrinsische Motivation oder fehlende Ansprechpartner bei Fragen mit sich bringt.[34]

## 1.2. Zielgruppenspezifische Anforderungen

Damit Bildung der breiten Gesellschaft verfügbar gemacht werden kann, gibt es einige Anforderungen, welche erfüllt werden müssen. Insbesondere Medien, die für das Selbststudium eingesetzt werden sollen, sollten so konzipiert sein, dass sie durch ihre Beschaffenheit keine zusätzlichen Hürden bilden.[25] Eine Ausgrenzung weniger Minderheiten kann nie vollends ausgeschlossen werden, Benutzungshilfen können einige Nachteile aber ausgleichen. Das Fachgebiet des User-Interface-Designs (UI-Designs) widmet

sich der Identifizierung unterschiedlicher Bedürfnisse von Anwender\*innen. Die zielgruppenspezifischen Anforderungen lassen sich in zwei Kategorien einteilen: Persönliche Vorlieben und Ausgleich von Nachteilen, insbesondere alters- und körperlichbedingt. Erfüllung persönlicher Vorlieben bei Gestaltung von UI-Elementen kann hierbei die Akzeptanz fördern, die für den Einsatz in der Lehre relevant ist.[21] Der Ausgleich von Nachteilen ist hingegen notwendig um manche Zielgruppen überhaupt erreichen zu können. Die Grenzen dieser Unterteilung können hierbei fließend verlaufen. So könnte beispielsweise diskutiert werden, ob eine bunte und verspielte Darstellung einfach nur den Geschmack jüngerer trifft oder, ob es gar notwendig ist um die Aufmerksamkeit zu erlangen. Dies ist aber nicht Gegenstand des Hauptteils dieser Arbeit und wird im Kapitel 7 - Ausblick nochmal aufgenommen.

### 1.3. Kontextbasierte UI-Anpassung

Wenn die individuellen Bedürfnisse der Nutzer\*innen identifiziert sind, geht es darum, Lösungen zu finden, die UI Bedarfsgerecht zu gestalten. Ziel sollte es sein, die generelle Gestaltung der Nutzeroberfläche so benutzerfreundlich wie möglich zu halten. Hierfür existieren Gestaltungsrichtlinien und Konzepte.[31] Einige zielgruppenspezifische Anforderungen können aber im Konflikt stehen. Zum Beispiel, wenn ein Gestaltungselement für eine Zielgruppe notwendig, für eine andere aber nicht praktikabel ist. Dafür können variierbare UI-Elemente eine individuelle Erfüllung der individuellen Bedürfnisse liefern. Durch kontextbasierte UI-Anpassungen kann eine Nutzeroberfläche für verschiedene Zielgruppen optimiert werden. Nutzende wählen ihre persönliche Konfiguration. So können Benutzungshilfen wie auch Gestaltungsvorlieben im Rahmen der gebotenen Möglichkeiten frei angepasst werden. Weiter ist es möglich gewisse Seiteneffekte an gewählte Konfigurationen zu binden. Durch den Kontext der Wahl wird eine Nutzergruppe identifiziert und weitere Anpassungen können werden vorgenommen.

---

# Kapitel 2 Ziel der Arbeit

Mit den erläuterten Grundbegriffen wird nun das Ziel der Arbeit dargestellt. Dafür wird zunächst das bestehende Gamebook-Projekt vorgestellt, auf dem die im folgenden vorgeschlagenen Funktionen aufbauen. Anschließend wird die dafür vorgesehene Arbeitsweise erläutert.

## 2.1. Verwandte Arbeiten

In seiner Bachelorarbeit *Entwicklung eines Generators für zugängliche Gamebook-UIs zur Anwendung in der digitalen Lehre* [20] hat Pulte (2022) eine erste eigene Nutzeroberfläche für die Darstellung von digitalen Gamebooks entwickelt und implementiert. Es handelt sich dabei um eine Weboberfläche, die aus einer XML-Repräsentation der Gamebooks dynamisch eine funktionale Anwendung generiert, welche in die überliegende Oberfläche eingebunden wird. Bei der Entwicklung waren gestalterische Aspekte der Barrierefreiheit bereits mit einbezogen, um eine nutzerfreundliche Anwendung anzubieten. Dabei beruft Pulte sich unter anderem auf die *Web Content Accessibility Guidelines*. [31] Diese Richtlinien schreiben gestalterischen Aspekte für eine zugängliche Benutzeroberfläche vor, die das Grundsystem standardweise liefern sollte. Auch für diese Arbeit werden die Funktionen danach ausgewählt und gestaltet. Eine genauere Erläuterung erfolgt im Abschnitt 3.3.

Im jetzigen Entwicklungszustands liefert der Generator von Pulte eine simple Benutzeroberfläche aus. Diese stellt sich kontrastreich in schwarz-weiß und mit einem blauen Hintergrund da, die die Inhalt- und Bedienelemente klar umreißen. Buttons sind mit Icons und Label versehen, was zum einen eine große Zielfläche liefert, zum anderen die Funktion schnell erfassen lässt. Die Webanwendung passt sich dynamisch an Größe des Browserfensters an und ist somit sowohl am Desktop, als auch auf Mobilgeräten bedienbar.

Als Bedienungshilfen werden ein weiteres Farbschema mit hohen Kontrasten, Textskalierung, Tastaturbedienung sowie ein Screenreader in der Arbeit beschrieben. Diese sollen überarbeitet und teilweise vollständig ersetzt werden. Die entsprechende Begründung findet sich im Teil III der Arbeit.

Im Ausblick seiner Arbeit gibt Pulte einige Punkte an, an die diese Arbeit anknüpfen wird. So präsentiert er hier die Idee eines LogIn-Dialogs, um Nutzenden von Beginn an eine passende Konfiguration der UI präsentieren zu können. Dieser Vorschlag wird in dieser Arbeit aufgegriffen und umgesetzt. Hier werden zum einen direkte Einstellungsmöglichkeiten über die angebotenen UI-Anpassungen zu finden sein, zum anderen werden weiterführende Informationen über die Nutzer\*innen gesammelt um einen Kontext herzustellen, der kontextspezifische Anpassungen aktiviert.

Die von Pulte entwickelte Benutzeroberfläche soll in dieser Arbeit weiter entwickelt werden, um den Gamebook-Generator für weitere Zielgruppen zugänglich zu machen und durch gesteigerte Akzeptanz den Lerneffekt zu erhöhen.

Des Weiteren greift diese Arbeit zu Teilen die Ergebnisse von Studien im Rahmen der Doktorarbeit *InfoBiTS: Informatische Bildung für Technikferne Seniorinnen und Senioren* [19] von Noichl (2020) auf. In Bezug auf die Zielgruppenerschließung der technikfernen Seniorinnen und Senioren können einige Erkenntnisse übernommen werden. So finden sich im Abschnitt 3.2 dieser Arbeit Ansätze zur Unterstützung dieser Personen. Insbesondere der Umgang mit fehlerhaften Toucheingaben beruht auf der Arbeit von Noichl.

## 2.2. Vorgehensweise

Einleitung

In der Einleitung wurden zunächst die drei Hauptbegriffe der Arbeit definiert. Mit dieser Motivation ist das Ziel der Arbeit eine Ausarbeitung von UI-Anforderungen für besser E-Learning-Inhalte sowie eine direkte Erweiterung einer bestehenden Webanwendung für digitale Gamebooks.

Anforderungen  
an E-Learning  
Plattformen

Die weitere Arbeit ist in drei Teile aufgeteilt. Zuerst werden Anforderungen an E-Learning Plattformen erarbeitet. Dazu findet eine genauere Beschreibung der Zielgruppen statt. Die Grundbedürfnisse sind hier als Basislinie all jene Gestaltungsaspekte, die allen Nutzenden den Zugriff und die Bedienung der Anwendung erleichtern. Hier wird die Annahme getroffen, dass eine gewisse Intuition in Bezug auf Webanwendungen vorhanden ist. Unter der Annahme besteht auch ein grundlegendes Interesse und Kenntnis über den Mehrwert der Gamebook-Anwendung. UI-Anpassungen gehen hier vor allem individuelle Vorlieben ein. Der Zweck dieses Angebots wird im Abschnitt 3.1 diskutiert. Davon abgehend werden insbesondere zwei Nutzergruppen spezifiziert. Senior\*innen sowie Kinder haben die nötige Intuition für digitale Anwendungen eventuell noch nicht entwickelt, da sie meist auf Erfahrungswerten beruht. Sie können aber durch eine geführte Oberfläche unterstützt werden. Auch die Darstellung kann auf diese Zielgruppen angepasst werden. Eine genauere Erklärung des Begriffs *Intuition* wird in diesem Abschnitt diskutiert.

Einige der Individualisierungsmöglichkeiten gleichen gezielt Nachteile aus, die im Alter auftreten können. Diese Punkte werden in den Abschnitten 3.2 und 3.3 ausgearbeitet. Weiterführend wird in Kapitel 4 die technische Seite der Anforderungen beschrieben.

Erweiterung  
der  
Gamebook-UI

Im darauf folgenden Teil wird die Implementierung der Features, sowie des Willkommensmenüs konkret beschrieben. Das Kapitel 5 befasst sich mit den einzelnen Features der UI-Anpassung. Auch werden hier Zusammenhänge zwischen wählbaren Optionen und versteckten Seiteneffekten hergestellt. Das Kapitel 6 behandelt das Willkommensmenü welches eine zielgruppengerechte Einführung in die UI-Konfigurationen bietet und Grundlage für die Kontextualisierung der Zielgruppe darstellt.

Ausblick und  
Zusammenfas-  
sungs

Im letzten Teil werden die Ergebnisse zusammengefasst und ein Ausblick auf aufkommende Fragen oder weitere Ideen gegeben. So wird auch ein Konzept zum evaluieren der UI-Anpassung bereitgestellt, da im Rahmen dieser Arbeit keine Studie durchgeführt wird.

---

# Teil II

## Anforderungen an E-Learning-Plattformen



---

# Kapitel 3 Identifizierung von Nutzergruppen

In diesem Teil der Arbeit sollen die individuellen Bedürfnisse von Nutzenden identifiziert werden, um daraus Anforderungen an eine E-Learning-Plattform zu formulieren. Dafür werden in diesem Kapitel zuerst die Nutzergruppen beschrieben, welche die Zielgruppe aller Lernwilligen ausmachen. Kontextbasierte UI-Anpassungen haben den Anspruch eine breite Zielgruppe anzusprechen.

## 3.1. Akzeptanz durch Personalisierung

*Intuition*

Im Forschungsbereich *Mensch-Maschine-Interaktion* gibt es wiederkehrend Untersuchungen zur Intuitivität technischer Systeme. Der Arbeitskreis *Intuitive Use of User Interfaces (IUII)* hat im Jahr 2006 [11] folgende Definition aufgestellt:

“Ein technisches System ist im Rahmen des spezifischen Kontextes einer Aufgabenstellung in dem Maße intuitiv benutzbar, in dem der jeweilige Benutzer mit ihm durch unbewusste Anwendung von Vorwissen effektiv interagieren kann.”

Ein genaues Modell zur Messbarkeit wurde aber nicht erarbeitet. Einer der Autoren, Jörn Hurienne, beschreibt *Intuition* zuletzt noch als “*notoriously vague concept*”. [22] Was aus der Definition hervorgeht, ist, dass die Intuition von Nutzer\*innen von technischen Systemen vorwiegend auf Erfahrung und Vorwissen der Nutzenden basiert. Dieses Vorwissen wird im Laufe des Lebens angesammelt. [1] Je früher die Interaktion mit digitalen Medien beginnt, desto mehr Intuition entsteht demnach. Eine ganze Generation wird daher auch als *digital natives* bezeichnet, da sie mit frühem Umgang mit technischen Systemen aufgewachsen sind und somit viel Übung hatten. [10] Aber auch Menschen, die vor dem Zeitalter der Heimcomputer geboren wurden, konnten und können diese Fähigkeiten noch erlernen und somit Intuition im Umgang mit technischen Systemen erlangen. [10] Generell besteht die Tendenz, dass bei Menschen ab 65 Jahren der Lernprozess länger dauert, weshalb einige die Grundkompetenzen im Umgang mit technischen Systemen nie erwerben. [3] Zum Erreichen dieser Zielgruppe könnten weitere unterstützende UI-Elemente für einen selbstsicheren Umgang nötig werden.

Im Folgenden wird die Nutzergruppe aus Menschen mit Vorerfahrung als Basiszielgruppe betrachtet, auf welches die E-Learning-Plattform ausgerichtet ist und die keine unterstützenden UI-Elemente benötigt. Dabei wird davon ausgegangen, dass die E-Learning-Plattform gewissen Gestaltungsrichtlinien und -mustern folgt, damit auf vorhandene Intuition zurückgegriffen werden kann.[29] In den folgenden Abschnitten werden Erweiterungen dieser Basiszielgruppe betrachtet und Lösungen für deren spezifische Anforderungen eingebracht.

User-  
Experience

Über die frustfreie Benutzung eines technischen Systems hinaus gibt es weitere Gestaltungsspielräume um die User Experience (UX) positiv zu beeinflussen. Als UX beschreibt die Gesellschaft für Informatik die *"ganzheitliche Sicht auf subjektiv erlebte Produktqualität"*. [9] Dies hat auch einen positiven Effekt auf den Lernerfolg, da die Interaktion mit der Plattform in den Hintergrund und die Lerninhalte in den Fokus der Aufmerksamkeit treten können. Dieser Effekt könnte noch weiter verstärkt werden. In der Werbeforschung wird seit einiger Zeit das Konzept *Akzeptanz durch Personalisierung* angewandt.[21] Durch persönlich zugeschnittene Inhalte steigt die Akzeptanz der Zielpersonen. Dieses Konzept soll im Folgenden auf E-Learning-Plattformen übertragen werden. Zugeschnittene Lerninhalte zur Differenzierung von Lernniveaus wurden in der Einleitung bereits als Vorteil der zur Nutzung von digitalen Gamebooks eingeführt. Personalisierung kann aber auch auf das User Interface (UI) übertragen werden, um potentiell die UX und somit den Lernerfolg zu steigern. Hierfür soll eine UI-Erweiterung implementiert werden, die dem Nutzer zunächst die Wahl zwischen einem hellen und einem dunklen Farbschema bietet. Für beide Schemata ist außerdem eine Hintergrundfarbe wählbar. Dafür stehen 8 gleichmäßig über das Tonspektrum verteilte Farben zur Auswahl, um ein breites Angebot zu schaffen. Als Akzentfarbe für Schaltflächen wird die jeweilige Komplementärfarbe verwendet. Die Lerninhalte bleiben zur besseren Lesbarkeit stets auf ungesättigtem Grund.

Inwiefern sich diese Akzeptanzförderung auf den Lernerfolg ausübt muss weiter untersucht werden. Im Abschnitt ?? wird ein Entwurf einer Evaluationsstudie vorgestellt um diese Auswirkung auszuwerten. In einem späteren Schritt lässt sich diese Erweiterung auch noch für zusätzliche Optionen zu verbesserten Barrierefreiheit einsetzen. Mehr dazu im Abschnitt 3.3.

Kompensation  
von fehlender  
Erfahrung

## 3.2. Unterstützende UI-Elemente

Wie zuvor beschrieben, beruht die Intuition im Umgang mit technischen Systemen vorwiegend auf Erfahrungen. Dies bedeutet, dass auch wenn die Bezeichnung es anderes vermuten lässt, niemand als *digital native* geboren wird. Ältere Personen benötigen eine gewisse Führung durch ein System. Beispielsweise wird eine Büroklammer in Mailanwendungen oft als Symbol für Anhänge verwendet, ohne dass es eine weitere Beschreibung gibt. Zumal auch nicht vorausgesetzt werden kann, dass Nutzende wissen, dass die technische Möglichkeit eines Mail-Anhangs existiert. Für vertraute Anwendende ist die Einsparung von Vorteil, da der gewonnene Platz die Übersichtlichkeit verbessert oder Raum für weitere Funktionen bietet. Ohne das nötige Vorwissen bedarf es aber einer Erklärung in Form eines Button-Labels. In einer Studie mit Seniorinnen und Senioren wurde die Aufgabe gestellt mehrere Icons ihrer Bedeutung zuzuordnen.[16] Auch wurde aus einer Auswahl verschiedener ähnlicher Symbole das, den Teilnehmenden nach, Aussagekräftigste ermittelt.[18]

Dies ist nur ein Beispiel für unterstützende UI-Elemente, die für eine Zielgruppenerweiterung auf ältere Personen eingeführt werden sollen. Kinder wiederum mangelt es in der Regel an gesammelter Erfahrung im Umgang mit technischen Systemen, da sie in ihrer bisherigen Lebenszeit nicht die Gelegenheiten hatten. Wie auch bei der älteren Zielgruppe können beispielsweise manche Symbole nicht zugeordnet werden, so dass Unterstützung von Nöten ist. Diese sollte altersgerecht sein. Beispielsweise könnte eine audiovisuelle Einführung in die Funktionen der Gamebook-UI die einzelnen Elemente erklären. Auch im Verlauf der Benutzung könnten wiederholt Hinweise auf Buttons oder ähnliches eingeblendet werden.

*Altersgerechte  
Unterstützung*

In dieser Arbeit soll die Führung durch die Funktionalitäten der Gamebook-UI zunächst in Textform geschehen. Als zusätzliche Unterstützung kann, durch einblenden von Symbolen wie Pfeilen oder Kreisen, der Blick geführt werden. Auffordernde Dialoge erklären wofür Buttons eingesetzt werden können. Da Gamebooks in der bisherigen Form hauptsächlich textbasiert sind, sollte von einer vorhandenen Lesekompetenz ausgegangen werden können.

Für eine Erschließung der Altersgruppe ohne Lesekompetenz sind zum einen andere Lerninhalte, aber auch weitere Gestaltungsprinzipien notwendig. Was bei einer solchen Erweiterung der Gamebook-UI beachtet werden müsste, wird im Abschnitt ?? beschrieben.

*Schaltflächen*

Die fertige E-Learning-Plattform soll als Web-Anwendung sowohl von Computern als auch von mobilen Geräten aus erreichbar sein. Um durch die Touch-Eingabe keine zusätzlichen Hürden beim benutzen der digitalen Gamebooks zu kreieren, gibt es ein paar Anforderungen zu beachten. Zum einen sollte die Anzahl an verschiedenen Gesten derartig begrenzt sein, dass Verwechslungen, beispielsweise durch unterschiedliche Dauer einer Berührung, nicht auftreten können. Zum anderen muss bei der Gestaltung von Buttons das Eingabeverhalten älterer Menschen beachtet werden. Im Paper *Adaptive Buttons für zielgruppengerechtes App Design* [17] stellt Noichl (2018) einen Ansatz für ein adaptives Design von Buttons in Apps vor. In ihrer Studie wurde beobachtet, dass die Zeile der Touch-Eingabe (Touchtargets) sich im unteren Drittel des Buttons befanden. Dabei landeten auch einige Touchtargets außerhalb der Umrandung des runden Buttons. Aus dieser Tatsache lässt sich die Gestaltungsrichtlinie folgern, dass immer genügend Abstand zwischen interaktiven UI-Elementen bedacht werden sollte. Noichl leitet aus der Tendenz zum unteren Drittel der Schaltfläche ab, dass die Zielfläche, sei es ein Label in Textform, oder ein Bild, innerhalb des Buttons nach oben verlagert werden sollte. Diese Vermutung wurde in einer kleinen Studie mit automatisch angepasster Textposition bestätigt. Hier wurde die Höhe des Textes im laufenden Verfahren korrigiert, wenn ein Touchtarget vom Mittelfeld abwich. Die genaue Anlage der Studie ist ebenfalls in diesem Paper beschrieben.[17] Es kann aber sein, dass eine Verlagerung des Textlabels nicht möglich ist. Daher soll in dieser Arbeit neben der adaptiven Anpassung der Texthöhe eine automatische Erweiterung des Bereichs um die Schaltfläche herum eingeführt werden. Über die Zeit könnte ein Verhaltensprofil der Nutzenden angelegt werden, welches die Größe des Bereichs beeinflusst in der Touchtargets einem Button zugeordnet werden. Dadurch kann sich die Trefferquote erhöhen.

## 3.3. Barrierefreiheit

Ein Ziel von E-Learning-Plattformen ist es mehr Menschen das Lernen ermöglichen zu können. Über das Internet können Lerninhalte von fast jedem Ort der Welt abgerufen werden. Dabei ist es notwendig die Barrieren zum Zugang so gering wie möglich zu halten, um den Zugang einer breiten Gruppe an Menschen möglich zu machen. Dies ist eine der Stärken digitaler Anwendungen, die das individuellen Gestalten von Inhalten oder auch kontextbasierte UI-Anpassungen erlaubt. Das Gremium *World Wide Web Consortium* (W3C) definiert diesbezüglich seit 1994 technische Standards, um eine Basis für eine geordnete Webentwicklung zu liefern. Der W3C-Direktor und Begründer des *World Wide Web* (WWW) Berners-Lee wurde 1997 in einer Presseerklärung so zitiert:

“The power of the Web is in its universality. Access by everyone regardless of disability is an essential aspect.”[30]

Um Web-Anwendungen nach diesem Prinzip zu gestalten, wurden von der W3C die in der Einleitung bereits vorgestellten *Web Content Accessibility Guidelines* [31] entwickelt. Sie umfassen Richtlinien, die sich vier Design-Prinzipien zuordnen lassen und nach drei Graden der Barrierefreiheit gemessen werden können. Webinhalte sollen den Kriterien *Wahrnehmbar*, *Bedienbar*, *Verständlich* und *Robust* unterliegen. In der Einleitung der WCAG [31] werden diese vier Prinzipien genauer erläutert. Im Folgenden werden nur die Prinzipien beschrieben, welche für die UI-Erweiterungen relevant sind, um einigen Menschen den Zugang zu den Lerninhalten ermöglichen zu können.

Das Prinzip der *Wahrnehmbarkeit* umfasst alle Anforderungen die sich mit der Sichtbarkeit von Information befassen. Für den höchsten Grad der Barrierefreiheit wird in der Richtlinie 1.4.6 [31] vor allem für Textinhalte ein verstärktes Kontrastverhältnis empfohlen. Menschen mit schlechter Sicht benötigen diese Unterstützung um die Inhalte besser lesen zu können. Insbesondere sollte beim eingesetzten Farbschema darauf geachtet werden, keine essenziellen Kontraste in Rot- und Grüntönen zu gestalten, um Personen mit einer Rot-Grün-Schwäche nicht auszuschließen. Dies sollte durch die einfach gehaltenen Farbräume bereits gegeben sein, da neben der gewählten Hauptfarbe für Hintergrundflächen Farben auf eine einzelne Akzentfarbe gesetzt wird, die lediglich auf ungesättigtem Grund erscheint.

Für den höchsten Grad der Barrierefreiheit wird in Ergänzung zu den in Abschnitt 3.1 vorgestellten Farbschemata ein Hochkontrast-Modus eingeführt. Dieser setzt auf einen schwarzen Hintergrund und kräftige Linien statt andersfarbige Flächen zur Abhebung von Elementen und eine helle, aber farbige Textfarbe um Überschriften abzuheben, die Inhalte von Steuerungselementen wie Buttonlabel unterscheidbar macht, welche in einer Signalfarbe gehalten werden. Dabei ist die Farbe nicht alleiniges Merkmal für ein Steuerungselement, sondern stellt, wie es die Richtlinie 1.4.1 der WCAG [31] vorschreibt nur eine Ergänzung da:

“Color is not used as the only visual means of conveying information, indicating an action, prompting a response, or distinguishing a visual element”

Aus diesem Grund werden beispielsweise Schaltflächen immer durch eine Umrandung erkennbar sein. Die genaue Gestaltung wird später im Kapitel 5 beschrieben.

Da die Plattform von verschiedenen Geräten aus aufrufbar sein soll ist ein fluides Design erforderlich. Dies bedeutet, dass Maße und Positionen nicht festgesetzt, sondern relativ angelegt werden. Dies trifft auch auf die Schriftgröße zu. Die Möglichkeit die Größe selbstständig zu verändern fällt ebenfalls unter das Prinzip der *Wahrnehmbarkeit*. Richtlinie 1.4.4 [31] beschreibt:

“Except for captions and images of text, text can be resized without assistive technology up to 200 percent without loss of content or functionality.”

Moderne Webbrowser erlauben es, Seiten zu vergrößern und somit auch den Text größer darzustellen. Zusätzlich soll eine Option erstellt werden, die das gezielte Ändern der Schriftgröße erlaubt. Dafür werden verschiedene Stufen der Schriftgröße angeboten, aus denen die Nutzenden wählen können. Die begrenzte Auswahl hat den Vorteil, dass die Seiteneffekte bei einer Änderung der Schriftgröße kontrollierbarer sind. Es kann sichergestellt werden, dass größerer Text andere Inhalte nicht verdrängt oder manche Steuerungselemente gänzlich aus dem Sichtfeld verschwinden. Auch die Funktionen des geführten Zugriffs kann so zuverlässig angepasst und getestet werden. Welche Seiteneffekte auftreten und wie mit diesen umgegangen wird, wird in Kapitel 5 ausführlich beschrieben.

Neben vielen visuellen Aspekten gibt es auch Richtlinien zur *Bedienbarkeit* und *Robustheit* die hier Beachtung finden sollen. Unter die Kategorie der *Robustheit* fällt die Nutzung auf verschiedenen Endgeräten. Mit der adaptiven Buttonanpassung wurde bereits eine Unterstützung der Toucheingabe eingeführt. Auch für Lernende am Computer soll es eine Unterstützung geben. Gerade in Kombination mit einem Screenreader ist es wichtig, dass Steuerungselemente so angelegt sind, dass sie als solche erkannt werden. Indem die Definition dieser Elemente korrekt geschieht, wird sichergestellt, dass zum einen Screenreader die Optionen vorlesen können, zum anderen aber auch eine Navigation mit der Tastatur alleine möglich ist. Die sogenannte Tab-Navigation lässt mit der Tab-Taste alle Schalt- oder Eingabeflächen nach einer festgelegten Reihenfolge durchwählen. Bei dem Design der Oberfläche sollte dementsprechend auch die Reihenfolge für eine sinnvolle Menüführung beachtet werden. Welche Reihenfolge genau sinnvoll ist eine Designentscheidung die es zu treffen gilt.



---

# Kapitel 4 Technische Umsetzung

Nachdem die UI-Anpassungen benannt und begründet wurden, welche die zielgruppenspezifischen Bedürfnisse verschiedener Personengruppen bedienen sollen, geht es im Folgenden um die technischen Anforderungen an die Gamebook-Plattform. Eine Hilfestellung muss nicht nur vorhanden, sondern auch zielgruppengerecht erreichbar sein. Als Lösung dafür wird im ersten Abschnitt ein Willkommensmenü vorgeschlagen, welches neben der Wahl verschiedener direkt sichtbarer Konfigurationen auch die Entscheidung über kontextbasierte Anpassungen vornimmt.

## 4.1. Willkommensmenü

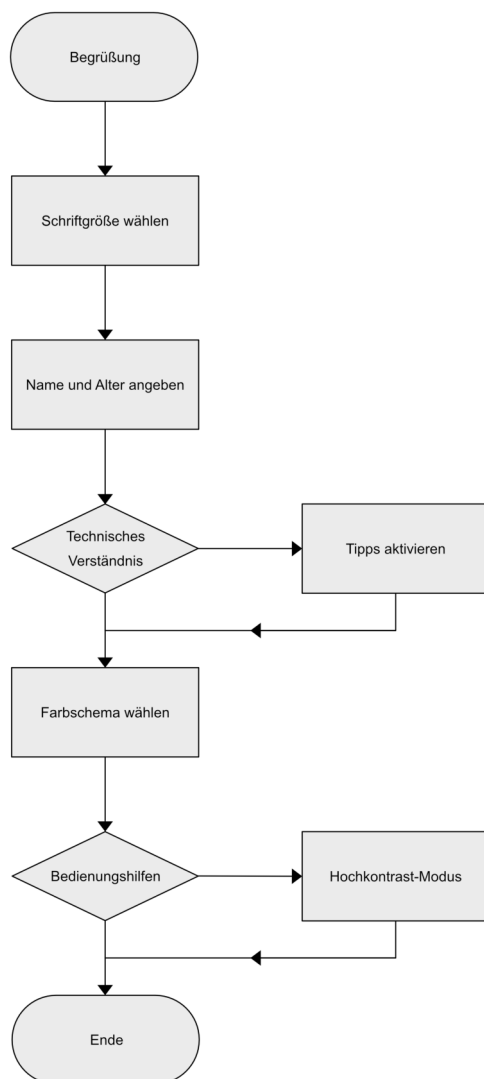
*Barrierefreiheit  
von Beginn an*

Ziel dieses Kapitels ist es die Anforderungen an eine E-Learning-Plattform zu definieren, um eine an die verschiedenen Zielgruppen angepasste, positive Nutzungserfahrung zu erreichen. Dafür genügt es nicht, die vorgeschlagenen Funktionen in die Plattform zu integrieren. Unter der Annahme, dass eine Person die Menüführung einer Plattform nicht intuitiv versteht, ist es schwer anzunehmen, dass die Person die Einstellungsmöglichkeit der Hilfestellung die zum Aktivieren notwendig sind, oder gar ihren Nutzen begreift. Allerdings ist es auch nicht von Vorteil und teilweise nicht möglich, so eine Funktion von Beginn an zu aktivieren. So sind manche Nutzer\*innen auf den Hochkontrastmodus angewiesen, andere könnte die Erscheinung aber abschrecken. Wie bereits zuvor erklärt, gibt es bei den Bedienungshilfen eine Art Optimierungskonflikt. Was der einen Gruppe hilft, kann die Erfahrung für eine andere Gruppe zum Nachteil werden. Daher kommt der Ansatz von kontextbasierter UI-Anpassung. Als Kontext werden alle Informationen über die einzelnen Nutzenden verstanden. Neben den eigenständig gewählten Einstellungen könnten auch weitere Dinge wie das Alter oder eine Selbsteinschätzung über die eigene Vorerfahrung abgefragt werden. Um diesen Kontext festzustellen, wird ein Willkommensmenü implementiert. In geführter Weise wird dieser Dialog den Nutzenden einige Fragen stellen. Aus den Antworten wird der Kontext definiert und eine Konfiguration von Funktionen ausgewählt. Eine zielgruppengerechte und barrierefreie Darstellung beschränkt sich nicht nur auf die Lerninhalte, sondern schließt den bereitgestellten Funktionsumfang mit ein.

Darüber hinaus bietet dieses Menü auch die Möglichkeit an, nicht nur das *Was?* der Antworten zu verarbeiten, sondern auch das *Wie?* zu analysieren. Das bedeutet, dass nicht nur Funktionen durch die Nutzenden aktiviert werden, sondern auch Anpassungen (beispielsweise auf Grund des Eingabeverhaltens) vorgenommen werden. Im vorherigen Kapitel wurde eine adaptive Anpassung der Schaltflächen mit Verschieben der Textlabel oder Erweiterung der Zielfläche vorgestellt. Der Grundstein dafür wird bereits im Willkommensmenü gelegt. Welche Informationen hier gesammelt und eingesetzt werden wird im Folgenden genauer beschrieben.

Wichtig ist die Reihenfolge, in der die Fragen präsentiert werden. Da es sich um eine textbasierte Anwendung handelt, sollte zuallererst die Erkennbarkeit des Textes sichergestellt werden. Zu Beginn wird die Schriftgröße direkt durch die Nutzenden festgelegt. Dies geschieht durch eine kurze Aufforderung und zwei Schalter zum vergrößern oder verkleinern des Textes. Änderungen werden dabei umgehend anhand der Textgröße eines Beispieltexts sichtbar. In Abbildung 4.1 lässt sich die Reihenfolge der Dialogfenster sehen. Eine grafische Darstellung der einzelnen Kacheln wird in Kapitel 6 zu sehen sein.

**Abbildung 4.1.:** Flow-Chart die den Ablauf des Willkommensmenüs zeigt. (Eigene Darstellung)



ren. Die im Rahmen dieser Arbeit implementierten Funktionen wurden unter anderem so ausgewählt, dass ein breites Spektrum an Menschen angesprochen werden und diese Struktur aufgebaut werden kann.

Im letzten Schritt werden die Bedienungshilfen angeboten. Auch diese können im gesamten Block übersprungen werden.

Um Informationen über die Interaktion mit den Schaltflächen zu sammeln, sollen die Nutzenden die eigene Altersgruppe aus einer Liste auswählen. Diese Information kann in Zukunft für weitere kontextbasierte Anpassungen genutzt werden.

Als nächstes sollen die Nutzenden eine Selbsteinschätzung über ihr technisches Verständnis abgeben. Bewerten sie sich selbst als nicht so vertraut mit technischen Systemen, wird ihnen ein Hilfestellung in Form des geführten Zugriffs angeboten. Sollten zu einem späteren Zeitpunkt weitere Hilfsangebote hinzugefügt werden, sollten diese ebenfalls an dieser Stelle Angeboten werden. Lernende die sich selbst als erfahren einschätzen überspringen diesen Schritt und fahren direkt mit dem Personalisierungsangebot fort.

Die Schritte der Personalisierung durchlaufen alle Anwender\*innen. Auch weitere Möglichkeiten zur Personalisierung der Lernplattform sollten an dieser Stelle vorgestellt werden. Die Aufteilung dieser Blöcke ermöglicht eine strukturierte Menüführung mit der Option vorgefertigte Konfigurationen zu wählen um sich nicht in Details zu verlie-

Über die gesamte Interaktion hinweg wird das Tippverhalten der Nutzenden analysiert. Wird wie in Noichels Studie von 2018[17] festgestellt, dass vermehrt Treffer im unteren Drittel oder gar am Rande der Schaltfläche landen, werden die Grenzen des Zielbereichs erweitert. Die adaptive Anpassung der Schaltflächen zeigt, wie Informationen aus dem Willkommensmenü genutzt werden können, um später eine reibungslose Benutzung zu ermöglichen. Durch die Selbsteinschätzung und die Zuordnung zu einer Altersgruppe sind die Faktoren gegeben, um auszuwerten, ob diese Bedienungshilfe nötig ist, ohne dass sich die Personen aktiv dazu entscheiden, da es eine Funktion ist, die nicht bewusst wahrnehmbar ist. Im Verlauf der Interaktion mit dem Willkommensmenü werden dann die Toucheingaben ausgewertet und die Anpassungen an den Schaltflächen vorgenommen.

Kontextualisierung

## 4.2. Konfigurationen

Sind die Einstellungen getroffen und Konfigurationen vorgenommen, müssen diese auf die UI übertragen werden. Hierfür wird eine Config-Datei benötigt, die nutzerspezifisch gespeichert wird.

Compile-Time und Run-Time

An dieser Stelle ist es wichtig, zwei Konzepte zu erwähnen. Bei der Softwareentwicklung lässt sich zwischen der *Compile-Time* und der *Run-Time* (auch Laufzeit) unterscheiden. Als *Compile-Time* werden die Operationen zusammengefasst die den lesbaren Quellcode in ein ausführbares Programm übersetzen.[2] Anschließend gibt es keine Möglichkeit mehr an den Grundstrukturen der Anwendung etwas zu verändern. Sobald eine Änderung übernommen werden soll, muss der Code neu kompiliert werden. Dies geschieht nicht während der laufenden Nutzung, wie der Name des Pendants bereits verrät. Lediglich dynamische Parameter die im Code als variabel angelegt sind, können auch verändert werden. Auch wenn Webanwendungen in der Regel nicht zu ausführbaren Programmen kompiliert, sondern ein sogenanntes Frontend vom Webserver an den Browser ausgeliefert und von diesem interpretiert werden, kommt ein Punkt auch hier zum tragen.[35] Mit Frontend werden Oberfläche und zugehörige Funktionen bezeichnet, die Anwender\*innen zu Gesicht bekommen. Änderungen zur *Run-Time* können nur an solchen Parametern vorgenommen werden, die zuvor so deklariert wurden. Dabei werden Manipulationen durch den Browser außenvor gelassen. Die *Run-Time*, umfasst das Verhalten eines Programmes in der Ausführung. Für den Fall einer Webanwendung für E-Learning ist damit der Teil des Lebenszyklus ab Start des Webserver und in erster Linie Aufruf der Homepage zu verstehen. Die geplanten UI-Anpassungen sollen zur Laufzeit vorgenommen werden. Das bedeutet, dass die entsprechenden Parameter im Frontend veränderbar deklariert sein müssen. Die gewählte Konfiguration, ob bestimmt durch das Willkommensmenü, oder nachträglich angepasst, wird vom Frontend referenziert und dementsprechend dargestellt. Daraus folgt, dass die Anpassungen jeder Zeit zur Laufzeit geändert werden können und der Code nicht abhängig von den Ergebnissen des Willkommensmenüs generiert wird.

CSS-Klassen

Die meisten Gestaltungsaspekte werden mit Hilfe von CSS-Attributen umgesetzt. Generell besteht die Möglichkeit mittels *JavaScript* CSS-Attribute zur Laufzeit für einzelne Elemente per ID zu verändern.[32] ID steht dabei für Identifikator. Der Nachteil dieses Vorgehen, der auch aus der Dokumentation hervorgeht, ist, dass nur das zuerst gefundene Element behandelt wird. Da eine Vergabe vieler eindeutigen IDs bei dynamischen Inhalten aufwendig und ineffizient ist fällt diese Lösung raus. Anstatt dessen werden

für die verschiedenen Konfigurationen CSS-Klassen definiert, denen die Elemente unter festgelegten Bedingungen zugewiesen werden. Dabei ist die Verwendung von globalen Variablen hilfreich. Da bestimmte Parameter an verschiedenen Stellen im Code eingesetzt werden, ist es effizienter sie zu Beginn zu initialisieren und an den entsprechenden Vorkommen zu referenzieren. Bei den Gestaltungsparametern handelt es sich um direkte Werte, wie beispielsweise die Schriftgröße, oder die gewählte Akzentfarbe, die direkt eingesetzt werden können. Die Konfigurationen werden auch durch globale Variablen repräsentiert. Ein Variable vom Typ *boolean* gibt an, ob die Oberfläche im Light- oder Dark-Theme erscheint. Diese Variable ist Kondition für die entsprechende CSS-Klasse.

#### Änderungen

Durch den Abruf der Konfiguration beim initialisieren der Seite sind die Einstellungen stets aktuell. Ändern Nutzende ihre Konfiguration wird dies in die Konfigurationsdatei geschrieben und die entsprechenden Konditionen für die CSS-Klassen werden neu gesetzt.

## 4.3. Bereitgestellte Browserfunktionen

#### Nutzung des User-Agents

Die moderne Webentwicklung ist auf eine Vielzahl von Endgeräten und Einsatzzwecken ausgelegt. Um den verschiedenen Varianten gerecht zu werden, geben Browser immer einen Satz an Informationen über die eigene Umgebung in Form des sogenannten *User-Agent* an den Webserver zurück. Basierend auf diesen Informationen, können einige UI-Anpassungen bereits ohne Eingriff der Nutzenden vorgenommen werden. Wenn diese ihre Präferenzen, wie beispielsweise die Wahl zwischen Light- und Dark-Theme, systemweit eingestellt haben, kann dies direkt übernommen werden.

#### Media queries

Damit dies geschieht, muss das Styling der Webapp entsprechend angelegt sein. Über *media queries* die mit der CSS Version 3 eingeführt wurden, können Fallunterscheidungen in der Gestaltung definiert werden.[33] Für die Farbschemata gibt es die Bedingung *prefers-color-Scheme* um zwischen Light- und Dark-Theme zu wählen. Auch der Hochkontrastmodus lässt sich über die Bedingung *prefers-contrast* voreinstellen. Wie genau die Bedingungen der *media queries* implementiert werden und wie eine Überlagerung der letztendlich gewählten Konfiguration umgesetzt soll im folgenden Teil III der Arbeit beschrieben werden.

#### Browser-Zoom

Für eine voreingestellte Schriftgröße existiert keine direkte *media query*.[33] Es empfiehlt sich, unabhängig von der geplanten Option zum verstellen der Schriftgröße diese responsiv anzulegen. Dies bedeutet, dass die Schriftgröße entweder kontinuierlich relativ zur Fenstergröße definiert ist oder dass mehrere Größenabschnitte definiert werden, welche mit einer jeweiligen fixen Schriftgröße einhergehen. Letzteres wird bei diesem Ansatz gewählt, da für das Feature zum Verstellen der Schriftgröße auch eine begrenzte Auswahl an Optionen angeboten wird. Darüber hinaus können Nutzende stets die vom Browser bereitgestellte Vergrößerung verwenden. Die Beschränkung wird gewählt, da ab einer gewissen Größe die Funktionalität der Seite sinkt und der Einsatz dieser Größen unwahrscheinlich ist.

#### Gesteuerter

#### Fokus-Wechsel

Um den Browser-Fokus mittels der Tab-Taste zu verschieben wird der html-tag *tabindex* verwendet. Somit können auch Elemente auswählbar gemacht werden, die an sich keine Interaktion anbieten.

---

## Teil III

# Erweiterung der Gamebook-UI



---

# Kapitel 5 Implementierung der Features

Die vorgeschlagenen Features wurden in die Webanwendung implementiert. Im Folgenden werden Besonderheiten bei diesem Vorgehen beschrieben.

## 5.1. Angular

*Komponenten*

Die bestehende Webanwendung wurde im *TypeScript*-Framework *Angular* entwickelt.[7] In *Angular* lassen sich verschiedene Elemente der Website in Komponenten gliedern, die wie Objektinstanzen über einen Selektor einsetzbar sind. Elemente können dynamisch erzeugt, wie auch zur *Runtime* über ein sogenanntes *Angular-Binding* `*ngIf` ein- und ausgeblendet werden.[6] Im Gamebook-Generator wird dies genutzt, um Gamebooks aus ihrer XML-Repräsentation in einen funktionalen Teil der Gamebook-UI zu übersetzen.

Die Gliederung in Elemente verringert die Menge an Code, welcher die UI beschreibt. Ein Baustein, wie eine Schaltfläche, wird in einer eigenen Komponente definiert, sodass Anpassungen an der Erscheinung nur in dieser Komponente vorgenommen werden müssen. Beim Kompilieren in das auslieferbare Front-End wird die Komponente an ihren Selektoren eingefügt und alle Änderungen sind an den entsprechenden Stellen, an denen die Komponente eingebunden wurde.

*Vorteile von SCSS*

Wie zuvor beschrieben, dienen unterschiedliche CSS-Klassen dazu, Gestaltungsparameter für die UI-Anpassungen zu sammeln. In den Standardeinstellung von *Angular* besitzt jede Komponente ihr eigenes Style-Sheet. So müssen auch gleichnamige Klassen in jeder Komponente neu definiert werden. Um die Arbeit zu vereinfachen, kann statt dem reinen CSS auch Skript Sprache SCSS genutzt werden, welche als Präprozessor dient und beim Kompilieren der *Angular*-Anwendung in reines CSS übersetzt wird. Mit SCSS wird das Styling um Variablen, Vererbungen, Verschachtelungen und anderen Funktionen erweitert.[26]

In einer globalen SCSS-Datei finden sich alle Variablen, die in mehreren Komponenten gebraucht werden. Bei den in dieser Arbeit implementierten Features sind das die auswählbaren Schriftgrößen, die Farbschemata, die zur Auswahl stehen und Abstände für adaptierte Schaltflächen. Variablen werden in SCSS mit `$` gekennzeichnet. In den einzelnen Komponenten wird das globale Styling über die folgende Funktion importiert:

```
1 @include <pfad>
```

[28]

Einige Features bestehen aus einer Auswahl verschiedener Modi. So passt sich die Hintergrundfarbe des Headers immer an die gewählte Primärfarbe an. Das Muster, in dem der Header gestylt wird, ist somit für alle Farbschemata gleich. Lediglich die Farbe, auf die der Parameter `background-color` gesetzt wird, unterscheidet sich. Um die redundante Klassendefinition zu vereinfachen, wird die SCSS-Regel `@each` verwendet:

**Quellcode 5.1:** Aus `beispiel.component.scss`

```
1  @include <pfad zu globaler scss-datei>

    {...}

5  $i = 0;
    @each $color in $color0, $color1, $color2, $color3 {
        .theme#{$1} {
            background-color: $color;
        }
10     $i = $i + 1;
    }
```

[27] Die Funktion ähnelt einer `for each`-Schleife. Zuerst wird eine Variable deklariert, und als Expression beispielsweise eine einfache Liste eingesetzt. Für jede Iteration nimmt die Variable nun die Form eines Listenelements an. Über eine Laufvariable lassen sich die Klassen eindeutig benennen. Beim Übersetzen im Präprozessor werden die einzelnen Klassen ausgeschrieben und die Funktion verschwindet.

Kommunikation  
zwischen  
den  
Komponenten

In *Angular* sind die Komponenten hierarchisch geordnet und Daten können direkt nur zwischen Eltern- und Kind-Komponenten übertragen werden. Um Daten zwischen allen Komponenten auszutauschen, wurde ein geteilter Dienst implementiert.[4] Daten, die geteilt werden sind die Identifikatoren für die aktuell gewählte UI-Konfiguration. Für jeden Identifikator wird ein *BehaviorSubject*-Objekt aus der *RxJS*-Bibliothek initialisiert.[24] Die *RxJS*-Bibliothek bietet eine Reihe von Funktionen für die reaktive Programmierung. So auch dieses Objekt, welches beobachtet werden kann und bei jeder Änderung die Beobachter über den neuen Wert informiert. Es grenzt sich außerdem dadurch ab, dass es einen Initialwert enthält, welches beim abonnieren direkt übermittelt wird. Dies sieht aus, wie im folgenden Beispiel:

**Quellcode 5.2:** Aus `uiconfig.service.ts`

```
1  private valueSource = new BehaviorSubject<string>("start");
    currentValue = this.valueSource.asObservable();

    changeValue(newValue: string) {
5      this.valueSource.next(newValue);
    }
```

Die Variable `valueSource` ist ein *BehaviorSubject*-Objekt vom Typ *string* mit dem Initialwert *start*. Die Variable `currentValue` wiederum verweist auf den Beobachter der `valueSource`. Alle Komponenten die, den Service einbinden können `currentValue` abonnieren und erhalten so den aktuellen und bei jeder Änderung umgehend den neuen Wert. Der Aufruf sollte in der Methode `ngOnInit()` geschehen, die beim Aufbauen der Seite ausgeführt wird. Auch kann aus jeder Komponente die Methode `changeValue(newValue: string)` aufgerufen werden, die den Wert von `valueSource` auf den übergebenen `newValue` setzt.

In einer Komponente sieht das ganze wie folgt aus:

**Quellcode 5.3:** Aus `beispiel.component.ts`

```

1   constructor(private uiconfig: UIconfigService) {}

   colorMode: string;

5   ngOnInit(): void {
       this.uiconfig.currentValue.subscribe(value => this.colorMode =
           value);
   }

```

Im *html*-Dokument wird ein Element der Klasse hinzugefügt, deren Klassenname gleich dem Inhalt der Variable `colorMode` ist. Dafür wird das *Angular-Binding* [`ngClass`] eingesetzt:[5]

**Quellcode 5.4:** Aus `beispiel.component.html`

```

1   <div class="Container" [ngClass]="colorMode">
       {...}
   </div>

```

## 5.2. Backendkommunikation

Die gewählte Konfiguration soll nach jeder Änderung im Profil gespeichert und beim Einloggen geladen werden. Dafür wird das Datenbank-Schema der User um die entsprechenden Variablen erweitert. Mehr zu den gewählten Variablentypen im nächsten Abschnitt.

**Quellcode 5.5:** Aus `backend/models/user.js`

```

1   const userSchema = mongoose.Schema({
       name: { type: String, required: true },
       email: { type: String, required: true, unique: true },
       password: { type: String, required: true },

5       // new variables
       ageID: { type: Number },
       darkMode: { type: Boolean },
       hcMode: { type: Boolean },
10      colorMode: { type: Number },
       fontSizeID: { type: Number },
       tipMode: { type: Boolean },
   });

```

Um auf das jeweilige User-Profil zugreifen zu können, muss die spezifische *UserID* aus der Datenbank im Token enthalten sein, welches beim LogIn gespeichert wird.

Frontend und Backend kommunizieren über eine sogenannte *REST-API*.<sup>[23]</sup> Im Backend existieren zwei Routen, um die Daten abzurufen und zu aktualisieren. Beim *GET-Request* ist die *UserID* Teil der Route, da *GET-Requests* keine Daten übermitteln. Im *PUT-Request* sind die zu aktualisierenden Daten Teil der Anfrage. Aber auch hier werden die entsprechenden Nutzerdaten definitionsgemäß über die *UserID* in der Route angesteuert.

**Quellcode 5.6:** Aus `backend/routes/user.js`

```

1   router.get("/theme/:userid", UserController.getPreferences);
   router.put("/preferences/:userid", UserController.setPreferences);

```

### 5.3. Die gewählten Erweiterungen

Die übermittelte *UserID* wird als Filter für die Datenbank-Abfragen verwendet. Diese ID ist eindeutig und wird beim Anlegen eines Users automatisch von der Datenbank generiert.

#### Quellcode 5.7: Aus backend/controllers/user.js

```
1   exports.getPreferences = (req, res, next) => {
    const filter = { _id: req.params.userid };
    User.findOne(filter)
      .then(...);
5   }

    export.setPreferences = (req, res, next) => {
    const filter = { _id: req.params.userid };
    const update = {...}
10   User.findOneAndUpdate(filter, update)
      .then(...);
    }
```

Die Funktionen `findOne()` [12] und `findOneAndUpdate()` [13] arbeiten auf dem ersten Eintrag der Datenbank, auf welchen der Filter zutrifft. Da die *UserID* eindeutig ist, existiert pro ID nur ein Eintrag, sodass immer das richtige Objekt gefunden wird. In der *Getter-Methode* werden der gefundene Eintrag im *Body* der *Response* zurückgegeben. In der *Setter-Methode* legt die Variable `update` fest, was am gefundenen Eintrag geändert wird. Die Hauptkomponente des Frontends ruft beim Initialisieren die *Getter-Methode* über den geteilten Service auf, sodass alle Parameter der Konfiguration dem gespeicherten Eintrag des Users entsprechen. Beim Änderungen an den Einstellungen wird an entsprechender Stelle über den geteilten Service die *Setter-Methode* aufgerufen, damit die gespeicherten Daten aktuell bleiben und beim nächsten Login wieder geladen werden können.

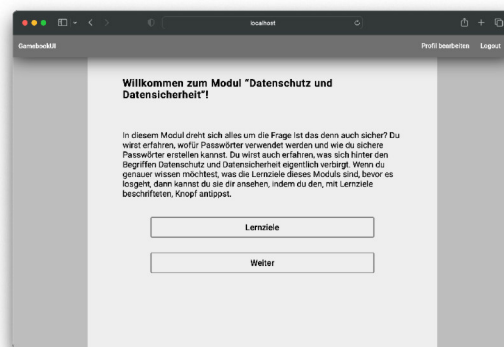
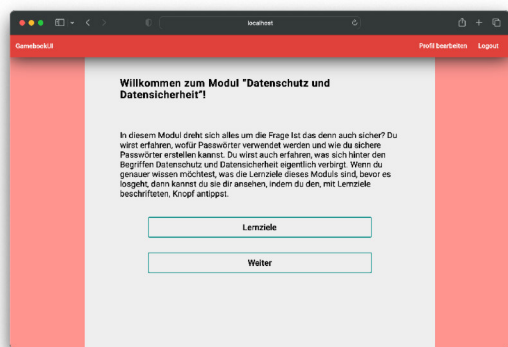
Erscheinungsbild

### 5.3. Die gewählten Erweiterungen

Für das Erscheinungsbild wurden zwei Konfigurations-Variablen eingeführt, die über den geteilten Service `uiconfig.service.ts` zwischen allen Komponenten geteilt werden. Ob die UI im Light- oder Dark-Theme erscheint, entscheidet ein *boolean*, da es nur zwei Zustände zu repräsentieren gibt. Steht die Variable `darkMode` auf `false` ist das Light-Theme aktiv, wie in den Abbildungen 5.1 und 5.2 zu sehen ist. In Abbildungen 5.3 ist die UI im Dark-Theme zu sehen. Die Variable `darkMode` steht auf `true`.

Abbildung 5.1.: UI im Light-Theme mit erstem Farbschema

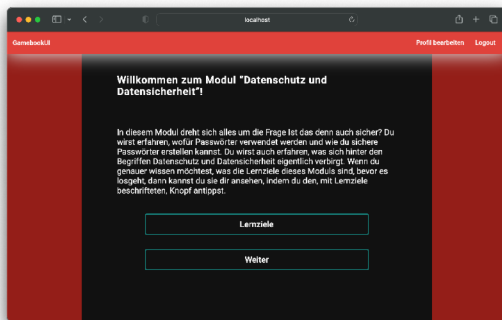
Abbildung 5.2.: UI im Light-Theme mit letztem Farbschema



Die Hauptfarbe wird über die Variable `colorMode` festgelegt. Es sind acht Farbmodi angelegt, die mit den Werten 0 bis 7 aktiviert werden können. Die Abbildungen 5.1 und 5.2 zeigen zwei Farbmodi im Light-Theme. Wie am Unterschied zwischen Abbildungen 5.1 und Abbildungen 5.3 zusehen ist, beeinflussen Light- und Dark-Theme auch die Hauptfarbe.

Für jeden Farbmodus sind vier Farben festgelegt. Eine Primärfarbe, die immer den Header färbt, eine Akzentfarbe, welche die Schaltflächen und andere Eingabeelemente hervorhebt, sowie eine helle und eine dunkle Variante der Primärfarbe, die im Light- beziehungsweise Dark-Theme den Anwendungshintergrund füllen. Schaltflächen unterscheiden sich aber nicht nur durch ihre Farbe, sondern fallen auch durch ihre Umrandung und ihr Verhalten beim Fokussieren auf. Sind sie im Fokus oder ist der Cursor im Zielbereich, füllt sich die Fläche der Schaltfläche mit der Akzentfarbe, um eine mögliche Interaktion zu signalisieren. Unabhängig vom Farbmodus sind der Hintergrund des Hauptinhalts und die darauf stehende Schrift immer schwarz und weiß. So wird stets ein hoher Kontrast garantiert.

**Abbildung 5.3.:** UI im Dark-Theme im ersten Farbschema



**Abbildung 5.4.:** Klassen des Headers

```
<!DOCTYPE html>
<html lang="en">
  <head>...</head>
  <body>
    <app-root_nghost-stt-c107 ng-version="14.0.6">
      <app-header_ngcontent-stt-c107_nghost-stt-c106>
        <mat-toolbar_ngcontent-stt-c106 class="mat-toolbar header mat-toolbar-single-row dark t8" ng-reflect-ng-class="{object Object}">...</mat-toolbar> flex = $0
      </app-header>
    </app-root>
    <main_ngcontent-stt-c107 ng-reflect-ng-class="{object Object}" class="dark t8">...</main>
  </body>
</html>
```

Abbildungen 5.4 zeigt die Klassen, die dem Header hinzugefügt wurden. Dies geschieht, wie in Abschnitt 5.1 beschrieben, über die `[ngClass]` Funktion. `[ngClass]` fügt immer nur die erste Klasse hinzu, deren Bedingung zutrifft. Daher sind die Konditionen für die verschiedenen Features aufgeteilt. Für die verschiedenen Farbklassen kommt die SCSS-Funktion `@each` zum Einsatz. Im Code sieht dies wie folgt aus:

**Quellcode 5.8:** Aus `header.component.html`

```
1 <mat-toolbar
  class="header"
  [class.dark] = "darkMode"
  [class.t8] = "hcMode"
5 [ngClass]="{'t0': colorMode == 0, 't1' : colorMode == 1, 't2':
  colorMode == 2, 't3' : colorMode == 3,'t4': colorMode == 4,
  't5' : colorMode == 5, 't6': colorMode == 6, 't7' :
  colorMode == 7}"
>
```

**Quellcode 5.9:** Aus `header.component.scss`

```
1 .header {
  width: 100%;
  position: fixed;
```

### 5.3. Die gewählten Erweiterungen

```
z-index: 10;
5 box-shadow: 0px 3px 50px $theme-dark10;

$i: 0;
@each $themeID in $t0-primary, $t1-primary, $t2-primary, $t3-
  primary, $t4-primary, $t5-primary, $t6-primary, $t7-primary
  {
10   &.t#{$i} {
      background-color: $themeID;
    }

    $i: $i + 1;
  }
15
&.dark {
  box-shadow: 0px 3px 50px $theme-bright10;
}
20
&.t8 {
  background-color: $theme-dark10;
  border: 2px solid $t8-link;
}
}
```

#### Schriftgröße

Die Schriftgröße ist in sieben Stufen verstellbar. Ähnlich wie bei der `colorMode` Variable, gibt es eine Zustands-Variable `fontSizeID`, die einen Index beinhaltet. Abhängig von diesem Inhalt werden die Elemente, die Text einer Klasse zugewiesen. Der Unterschied ist in den Abbildungen 5.5 und 5.6 zu sehen.

Abbildung 5.5.: UI mit kleiner Schriftgröße

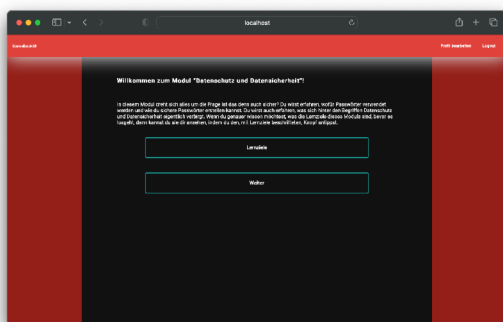
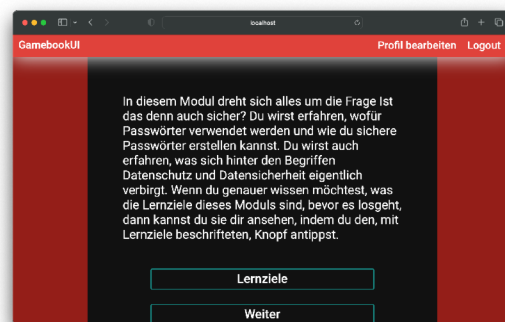


Abbildung 5.6.: UI mit größerer Schriftgröße



Da es Elemente mit verschiedener Schriftgröße gibt, sind in der Globalen sieben Basisschriftgrößen definiert, die für den normalen Text und die Schaltflächenbeschriftung gelten. Zusätzlich sind in der globalen SCSS-Datei zwei Faktoren für `h1`- und `h2`-Überschriften definiert. Diese werden entsprechend in den Klassen mit der Basisschriftgröße multipliziert. Für Navigations-Elemente existiert ebenfalls ein Offset-Wert, der mit der Basisschriftgröße verrechnet wird.

Die Bedienungshinweise stellen in dieser Reihe der UI-Anpassungen eine Ausnahme dar, da sie ein zusätzliches Element hinzufügen, welches ein- und ausgeblendet wird. Dies geschieht nicht beim Aufbau der Seite, sondern nach Ablauf einer voreingestellten Zeit. Dafür ist die Zustands-Variabel `showTip` vom Typ `boolean` definiert, die als Kondition für das *Angular-Binding* `*ngIf` eingesetzt wird. Zusätzlich existiert die Kontroll-Variabel `tipMode`, die die Einstellung speichert, ob die Hinweise aktiviert oder deaktiviert sind. Ist `tipMode` auf `true` gesetzt, startet nach dem Aufbau der Seite ein Timer, nach dessen Ablauf die Hinweise eingeblendet werden. So werden die Lernenden nicht vom eigentlichen Inhalt abgelenkt, sondern erhalten den Hinweis erst, wenn sie längere Zeit keine Eingabe gemacht haben.

Für das neue Hinweis-Element würde in der Komponente der Schaltflächen ein neuer Container der Klasse `overlay` eingefügt. Der Container kapselt das Hinweis-Element. Die Klasse positioniert den Container auf einer höheren Ebene als die restliche Anwendung und gibt ihr eine absolute Position. So beeinflusst das Element beim Erscheinen den restlichen Inhalt nicht.

**Quellcode 5.10:** Aus `borderbtn.component.ts`

```
1   ngOnInit(): void {
      {...}

      if (this.tipMode) {
5     setTimeout(() => this.showTip = true, 20000);
      }
    }
```

**Quellcode 5.11:** Aus `borderbtn.component.html`

```
1   <div class="overlay" *ngIf="showTip">
      {...}
    </div>
```

**Quellcode 5.12:** Aus `borderbtn.component.scss`

```
1   .overlay {
      z-index: 10;
      position: absolute;
    }
```

Der Hochkontrastmodus ist ein zusätzliches Erscheinungsbild, welches alle anderen Einstellungen zur Farbgestaltung überlagert. Es wird lediglich auf kontrastreiche Farben gesetzt, die alle eine feste Bedeutung haben, um Elemente verschiedener Funktion unterscheidbar zu machen. Diese Gestaltung ist in den Abbildungen

Abbildung 5.7.: UI im Hochkontrastmodus

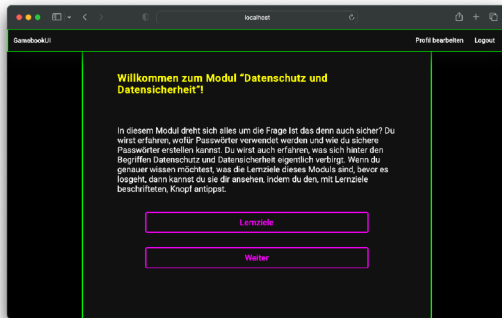
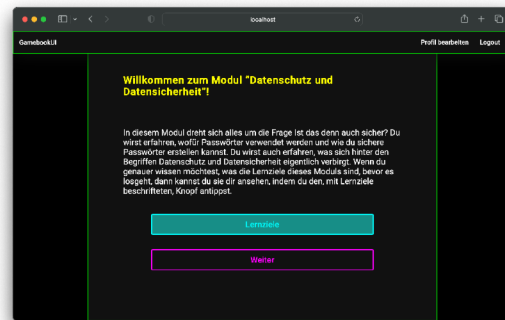


Abbildung 5.8.: Fokus auf Schaltfläche im Hochkontrastmodus



Der Hochkontrastmodus wird sich ähnlich zum restlichen angepassten Erscheinungsbild aktiviert. Die Besonderheit ist, dass Light- und Dark-Theme, wie auch der Farb-Modus überlagert werden und somit alle Elemente die eine Farbe haben, bei aktivem Hochkontrastmodus in eine entsprechende Klasse hinzugefügt werden. Die Überlagerung wird dadurch erreicht, dass die Eigenschaft `!important`, welche der CSS-Klasse hinzugefügt wird.

Quellcode 5.13: Aus header.component.scss

```
1  .header {
    {...}
    box-shadow: 0px 3px 50px $theme-dark10;

5     $i: 0;
    @each $themeID in $t0-primary, $t1-primary, $t2-primary, $t3-
      primary, $t4-primary, $t5-primary, $t6-primary, $t7-primary
      {
        &.t#{$i} {
          background-color: $themeID;
        }

10     $i: $i + 1;
      }

    &.dark {
15     box-shadow: 0px 3px 50px $theme-bright10;
    }

    &.t8 {
20     box-shadow: 0px 3px 50px $theme-dark10 !important;
        background-color: $theme-dark10 !important;
        border: 2px solid $t8-link !important;
    }
  }
```

### Tab Navigation

Um die interaktiven Elemente wie Schaltflächen per Tab-Taste auswählbar zu machen, wurde den *HTML*-Elementen ein `tabindex` von 0. Dies sorgt dafür, dass alle Elemente in der Reihenfolge ihres Vorkommens auswählbar sind.[14] Über verschiedene Zahlen ließe sich eine spezifische Reihenfolge festlegen. Da die Elemente der Gamebooks aber

automatisch generiert werden, wäre dies ein höherer Aufwand für den ein neuer Parameter beim Erstellen der Gamebooks vergeben werden müsste. Mit einem negativen Wert wären Elemente gezielt nicht auswählbar.

Um den aktuellen Fokus sichtbar zu machen wurden die CSS-Pseudoklasse `:focus`, mit der Pseudoklasse `:hover` gleichgesetzt. Die hervorgehobene Schaltfläche kann in Abbildung 5.8 gesehen werden.

**Quellcode 5.14:** Aus `borderbtn.component.scss`

```

1  .touchTarget {
    {...}
    width: 80%
    min-height: 60px;
5   cursor: pointer;

    &.inlargeTT {
        padding-bottom: $bt-offset-target;
    }
10  &:hover, &:focus {
        {...}
    }
}

15  .gb-button{
    width: 100%;
    height: 100%;
    align-items: center;

20  &.labelOffset {
        padding-bottom: $bt-offset-label;
    }
    {...}
}

```

Um die Zielfläche der Schaltflächen zu vergrößern, das Seitenlayout dabei aber nicht zu verändern, wurden zwei Container um den visuellen Button gesetzt. Die Vergrößerung der Fläche findet hier nur in der Höhe statt, da die Breite der Schaltflächen von der Breite des Inhalt-Containers vorgegeben ist. Dieser passt sich dynamisch an die Fensterbreite an, um eine responsive Nutzeroberfläche auszuliefern. Der äußere der beiden Container, die um den visuellen Button gelegt ist, hat eine feste Höhe. Dies stellt das gleichbleibende Layout sicher. Der innere Container `touchTarget` stellt die Zielfläche dar. Dieser Container löst das Klick-Event aus und hat ebenfalls feste Maße. Der visuelle Button, der innerhalb dieses Containers liegt, ist auf die maximale Größe gesetzt und somit immer gleich groß. Wenn die Schaltfläche vergrößert wird, geschieht dies, indem dem `touchTarget` ein `padding-bottom` hinzugefügt wird. So vergrößert sich der Zielbereich, der visuelle Button behält aber die gleichen Maße.

Für den Versatz des Text-Labels wird dem Text ebenfalls ein `padding-bottom` hinzugefügt. So wird das Label innerhalb des visuellen Buttons nach oben geschoben. Bis zu einer gewissen Größe des Versatzes verändert sich die Höhe des Buttons nicht.

**Quellcode 5.15:** Aus `borderbtn.component.html`

```

1  <div class="Container">
    <div

```

*Adaptive  
Schaltflächen*

### 5.3. Die gewählten Erweiterungen

---

```
class="touchTarget "  
  {...}  
5  (click)="storePage() "  
  >  
  <button  
    class="gb-button"  
    {...}  
10  >  
    {{data.content}}</button>  
  </div>  
</div>
```

---

# Kapitel 6 Implementierung des Willkommensmenüs

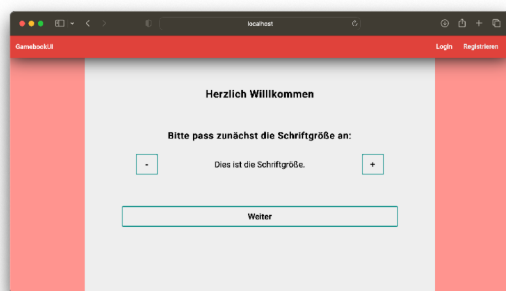
Der strukturelle Aufbau des Willkommensmenüs wurde bereits im Abschnitt 4.1 beschrieben. In diesem Kapitel soll es um die Implementierung gehen.

## 6.1. Einstellungsmasken

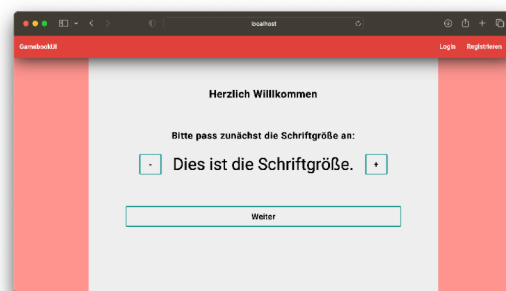
Einige der UI-Anpassungen werden direkt durch die Nutzer\*innen eingestellt. Hierfür sind Einstellungsmasken nötig, welche die Anpassung direkt sichtbar machen. Für die Schriftgröße passiert dies durch zwei Schaltflächen, die einen Text in der Mitte haben. Drückt der User auf die linke Fläche verkleinert, drückt er auf die rechte Fläche vergrößert sich der Text. So kann direkt ein Eindruck gewonnen werden, wie sich die Einstellung auf die UI, ohne dass sich zu viel verändert und Elemente aufgrund von Größenänderungen verschoben werden. Dafür wurde eine zusätzliche Zustands-Variable implementiert, welche die Klasse des Beispieltexsts bestimmt. Dies ist in den Abbildungen ?? und ?? zu sehen. Beim Bestätigen wird die Einstellung für die gesamte UI übernommen.

*Teileindruck  
bei Auswahl*

**Abbildung 6.1.:** Willkommensmenü mit eingestellter kleiner Schriftgröße



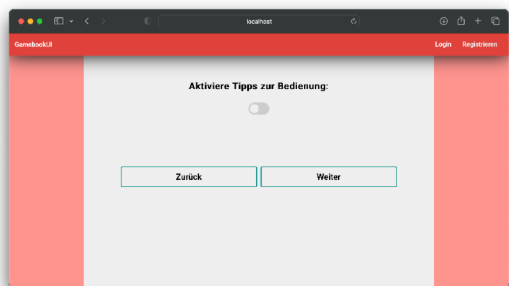
**Abbildung 6.2.:** Willkommensmenü mit eingestellter größerer Schriftgröße



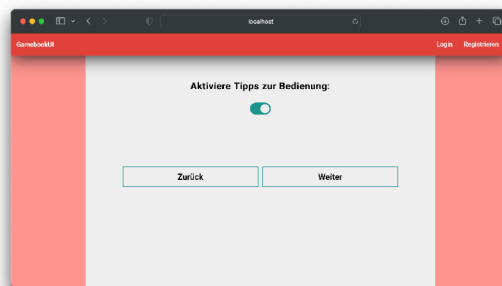
Einstellungen wie das Light- und Dark-Theme lassen sich am ganzen System am besten beurteilen. Für einfache Ja-Nein-Entscheidungen wurde ein Schalter implementiert, der wie ein Button ein Klick-Event auslöst, aber sein Erscheinen abhängig von der veränderten Zustands-Variable verändert. So kann der Status der Zustands-Variable stets anhand der Erscheinung des Schalters erkannt werden, auch wenn die Einstellung nicht wie bei Light- und Dark-Theme und Hochkontrastmodus direkt sichtbar ist, sondern wie bei den Bedienungshinweisen erst im eigentlichen Gamebook zu sehen ist. Der Unterschied ist in den Abbildungen 6.3 und 6.4 zu sehen.

*Schalter*

**Abbildung 6.3.:** Willkommensmenü mit deaktiviertem Schalter



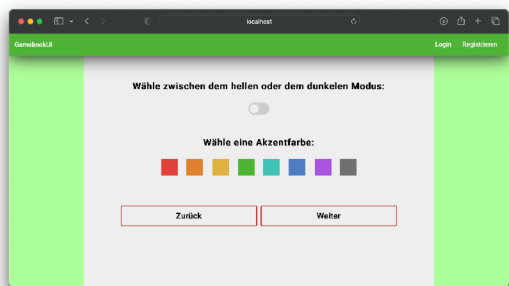
**Abbildung 6.4.:** Willkommensmenü mit aktiviertem Schalter



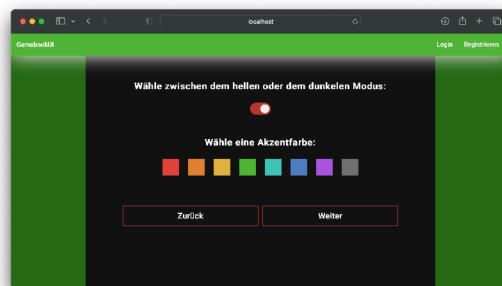
### Vollständiger Eindruck

Für die Akzentfarbe steht eine Auswahlliste zur Verfügung. Es gilt das Prinzip *“What you see is what you get.”* Die aufgelisteten Farben sind Beispiel für die wählbaren Primärfarben, in denen der Header bei Auswahl erscheint. Nach Auswahl einer Farbe wird diese auf das gesamte Erscheinungsbild übertragen, um einen Eindruck zu vermitteln. Um einen Gesamteindruck zu bekommen wird hier auch die Auswahl zwischen Light- und Dark-Theme angeboten, wie in den Abbildungen 6.5 und 6.6 zu sehen ist.

**Abbildung 6.5.:** Auswahl des Erscheinungsbildes im Light-Theme



**Abbildung 6.6.:** Auswahl des Erscheinungsbildes im Dark-Theme



## 6.2. Kontextbasierte Anpassungen

Unter den implementierten UI-Anpassungen ist eine, welche nicht im Willkommensmenü direkt von den Nutzer\*innen eingestellt wird. Die adaptive Schaltflächenanpassung wird abhängig von einem größeren Kontext aktiviert. Dafür werden drei Informationen berücksichtigt, die im Laufe des Willkommensmenüs gesammelt werden. Als Erstes werden Nutzer\*innen nach ihrer Altersgruppe gefragt. Dazu kommt eine Selbsteinschätzung, ob die Nutzer\*innen Erfahrung mit E-Learning Plattformen haben.

Im Willkommensmenü selber passen sich die Schaltflächen noch nicht an das Verhalten der Nutzenden an. Die vergrößerte Zielfläche um den visuellen Button herum wird dafür genutzt zu zählen, wie oft die Nutzerinnen den Button nach unten verfehlen. Dabei wird jedes Mal ein Zähler hochgezählt. Ab einer bestimmten Fehlerquote wird die Schaltflächenanpassung aktiviert. Es gibt dafür drei Schwellenwerte, welche von den zuvor gegebenen Informationen abhängt. Ordnen sich die Nutzer\*innen der höchsten Altersgruppe zu und geben an keine Vorerfahrung zu besitzen, genügt die geringste Anzahl Fehlern, um die Hilfe zu aktivieren. Der mittlere Schwellenwert gilt, wenn nur eins der beiden

Informationen gewählt wird. Sind die Nutzer\*innen in den jüngeren Altersgruppen und haben Vorerfahrung mit E-Learning-Plattformen, liegt der Schwellenwert sehr hoch.



---

# Teil IV

## Ausblick und Zusammenfassung



---

# Kapitel 7 Ausblick

Die vorgestellten Erweiterungen der Gamebook-UI bieten eine Grundlage für die Zielgruppenerweiterung der E-Learning-Plattform. Die Entwicklung ist damit aber noch nicht abgeschlossen. Es gibt einige weitere kontextbasierte Anpassungen, die das Potenzial haben, die Bedienung für einzelne Nutzer\*innen zu erleichtern. Eine weiterführende Studie, welche die Wirksamkeit der UI-Anpassungen auf Akzeptanz, wie Zugänglichkeit der genannten Personengruppen untersucht wäre nötig, damit diese und zukünftige Erweiterungen sinnvoll eingesetzt werden.

Auch das Gamebook an sich wird in Zukunft noch einige neue Elemente dazu bekommen. In diesem Kapitel wird ein kleiner Blick in diese Zukunft geworfen. Es werden Punkte in der bisherigen Implementation zusammengetragen, an die zum einen neue UI-Anpassungen, zum anderen neue Gamebook-Elemente anknüpfen müssen, um die bisher bestehenden Funktionen zu unterstützen. Zum Schluss folgen ein paar Vorschläge für zukünftige Erweiterungen.

## 7.1. Neue UI-Anpassungen

Die in dieser Arbeit vorgestellten UI-Anpassungen lassen sich technisch in drei Kategorien einteilen. Jede dieser Kategorien bringt eigene Dinge mit, auf die bei der Implementierung geachtet werden muss. Ein Einordnen in diese Kategorisierung, hilft den Überblick über die Anknüpfungspunkte im Code nicht zu verlieren.

Eine grundlegende Kategorie der UI-Anpassungen ist die der Anpassung bestehender UI-Elemente. Dies können allgemeine Elemente wie der Anwendungs-Header und -Hintergrund, oder aber Gamebook-spezifische Bausteine wie ein Button sein. Ein Großteil der in dieser Arbeit vorgestellten Anpassungen fällt unter diese Kategorie.

Neue Anpassungen an der bestehenden UI werden über CSS-Klassen realisiert, denen die Elemente unter definierten Bedingungen hinzugefügt werden. In den bisherigen Beispielen war dies immer eine Zustands-Variable, die wie ein Selektor fungiert. Somit sollten zuerst CSS-Klassen konzipiert werden, die die neuen Änderungen definieren. Dabei gilt es wiederkehrende Parameter in globale Variablen auszulagern, um den Zugriff aus allen Komponenten zu ermöglichen und so Änderungen zu erleichtern. Beim Definieren der Anpassungs-Klassen muss auf die Spezifität der Regeln geachtet werden, um ungewollte Überlagerungen des Standard-Stylings zu vermeiden. Dafür werden die Anpassungs-Klasse am besten so in der Element-Klasse verschachtelt, dass die definierten Regeln für Elemente gelten, die in beiden Klassen liegen.

Für die definierte Zustands-Variable gilt es, das Datenbank-Schema und die Backendkommunikation angepasst werden, damit diese auch im Nutzerprofil gespeichert werden kann.

Werden durch die geplante Funktion neue Elemente wie die Bedienungshinweise hinzugefügt, gibt es ein paar Überschneidungen in der Konzeption. Das Auslagern von Parametern in globalen Variablen sollte zum allgemeinen Stil der Anwendungsentwicklung gehören, um ein einheitliches Bild zu erhalten. Auch das Einfügen und der Umgang mit einer Zustands-Variable sollte genau so behandelt werden.

In der Konzeptphase sollte darüber entschieden werden, ob das neue Element an ein bestehendes Objekt gebunden ist, oder in einer eigenen Komponente erstellt werden soll, um an mehreren Stellen eingebunden werden zu können.

Unter die funktionalen Anpassungen fällt die adaptive Schaltflächenanpassung. Hierfür wurde in den Aufbau einer ganzen Komponente eingegriffen. Zu bedenken ist hier, ob und wie die Änderung, im aktivierten Zustand, Einfluss auf das Layout der Anwendung nehmen soll.

Zuletzt gilt es, für jegliche Art der UI-Anpassung, eine Aktivierungsoption in die im Abschnitt 4.1 vorgeschlagene Gliederung des Willkommensmenüs einzufügen.

## 7.2. Neue Gamebook-Elemente

Da die Entwicklung der Gamebook-Plattform nicht abgeschlossen ist, werden in Zukunft neue Gamebook-Elemente eingeführt. Um diese im Generator in die Anwendung einbinden zu können, sollte die neuen Elemente als neue Komponente angelegt werden. Damit neue Komponenten in die bestehenden UI-Anpassungen integriert werden können, müssen zuerst alle nötigen Zustands-Variablen über den geteilten Service *UIconfigService*, sowie die globale Styling-Datei in die Komponente eingebunden werden. Daraufhin können angepasste Erscheinungsbilder eingerichtet werden.

## 7.3. Vorschläge für zukünftige Erweiterungen

Eine große Hürde für viele Lernende ist die Sprachbarriere. Auch wenn die Unterstützung verschiedener Sprachen in erster Linie ein inhaltlicher Aufwand der Gamebook-Ersteller\*innen ist, muss auch die UI dafür angepasst werden. Neben mehrsprachigen Gamebooks würde alleine die Übersetzung der Anwendungs-Elemente, wie dem Willkommensmenü, eine Grundlage bieten, um Gamebooks in anderen Sprachen zu integrieren und somit die Zielgruppe zu vergrößern. Gamebook-Elemente mit Textinhalt könnten statt einer Text-Variable ein Datenkonstrukt beinhalten, welches eine Zuordnung von Sprache als Schlüssel und entsprechendem Text als Wert ermöglicht. Über eine globale Spracheinstellung wird nun jeweils der entsprechende Text in der Nutzeroberfläche angezeigt.

Die bisherige Plattform ist textbasiert. Dies schließt zum einen Personen mit Sehschwäche aus. Dafür bieten die meisten Endgeräte bereits Screenreader an, die die Inhalte vorlesen und die Navigation ermöglichen. So können diese beeinträchtigten Personen jegliche Inhalte ebenso nutzen. Gamebooks haben aber auch das Potenzial Kinder beim Lesen lernen zu unterstützen. Hierfür wäre ein klassischer Screenreader unangebracht, da das Lesen der Textinhalte des Gamebooks zur Übung gehört. Um Kindern die restliche Menüführung um die Gamebooks herum zu erleichtern, könnte eine Audioführung implementiert werden. Beim Berühren der Steuerelemente wird die Funktion vorgelesen,

oder ein anderes Signal wird gegeben, welches eine Verknüpfung zwischen Schrift und Inhalt herstellt.

Die bisherige Nutzeroberfläche ist sehr simpel gehalten. Von der Navigationsleiste im Header bis zu den Profileinstellungen sind die Schaltflächen klar strukturiert. Die Plattform bietet aber Potenzial, neue Funktionen neben den eigentlichen Gamebooks zu erhalten. Ein Beispiel ist die erweiterte Gamification, die das Lernen erleichtern und die Motivation erhöhen soll. Um bei wachsendem Funktionsumfang den Überblick über die Plattform nicht zu verlieren, kann ein Modus für einfachere Darstellung eingeführt werden, der bestimmte Elemente ausblendet und andere gestalterisch so anpasst, dass ein klares Bild entsteht, in denen gerade Lernende mit wenig technischer Erfahrung sich gut zurechtfinden.



---

## Kapitel 8 Zusammenfassung

Auf Basis verschiedener Grundsätze zur Verbesserung der User-Experience, wurden in dieser Arbeit kontextbasierte UI-Anpassungen integriert, um am Beispiel einer Webanwendung für digitale Gamebooks zu zeigen, wie E-Learning-Plattformen für verschiedene Zielgruppen erreichbar gemacht werden können. Um die zielgruppenspezifisch optimalen Grundsätze der Gestaltung zu wählen, wurden in Kapitel 3 zuerst die drei Nutzergruppen identifiziert, an die sich die verschiedenen Erweiterungen richten. Dabei wurde unter der Annahme gearbeitet, dass ein Großteil der Lernenden über die nötige Vorerfahrung und die Fähigkeiten verfügen, ohne Hilfsmittel mit technischen Systemen umgehen zu können. Für diese als *“Basiszielgruppe”* bezeichnete Menge an Nutzer\*innen wurden verschiedene Personalisierungs-Angebote integriert. Für alle denen die Vorerfahrung fehlt, wurden direkt leitende Hilfsmittel, wie auch Verhaltensanpassungen im Umgang mit den Bedienelementen eingefügt. Für Beeinträchtigte wurden verschiedene Hilfsmittel erschlossen, um die Darstellung der Inhalte zugänglicher zu machen. Grundlage hierfür waren die *Web Content Accessibility Guidelines*.

Um einen Optimierungskonflikt zu vermeiden, wurden all diese Erweiterungen der UI so angelegt, dass Nutzende selbst beeinflussen können, welche Anpassungen für sie aktiviert werden. Dafür wurde ein Willkommensmenü vorgestellt, welches über verschiedene Abfragen in gegliederter Abfolge einen Kontext über die Eigenschaften der Nutzenden erstellt.

Im letzten Teil der Arbeit wurde gezeigt, wie die zuvor beschriebenen UI-Anpassungen in die Gamebook-Plattform implementiert wurden. Der Fokus lag hier darauf, die einzelnen Parameter der Gestaltungsmöglichkeiten so zu implementieren, dass sie nachträglich noch effizient geändert werden können. Dabei wurde darauf geachtet, den modularen Ansatz, den das *Angular*-Framework bietet fortzuführen. Die dafür gewählten Methoden sind in diesem Teil ausführlich beschrieben. Somit können neue Erweiterungen leicht an die geschaffenen Strukturen anknüpfen und die Gamebook-Plattform stetig erweitert werden, um die Lerninhalte einer immer größeren Zielgruppe bereitstellen zu können.



---

# Anhang A Literaturverzeichnis

- [1] E. Chudnoff. *Intuition*. OUP Oxford, 2013.
- [2] Andrew Ferguson. A history of computer programming languages. [https://cs.brown.edu/~adf/programming\\_languages.html](https://cs.brown.edu/~adf/programming_languages.html), 2000. [Online; Abgerufen 06.03.2023].
- [3] Johanna Gebrande and Jens Friebe. Grundkompetenzen, bildungsverhalten und lernen im höheren lebensalter. ergebnisse der studie "competencies in later life"(cill); basic competencies, educational behavior, and learning in later life - results of the study "competencies in later life" (cill). *Zeitschrift für Pädagogik*, 61(2):192–204, 2015.
- [4] Google. Guidelines for creating ngmodules. <https://angular.io/guide/module-types#service>, 2023. [Online; Abgerufen 12.03.2023].
- [5] Google. Ngclass. <https://angular.io/api/common/NgClass>, 2023. [Online; Abgerufen 12.03.2023].
- [6] Google. Ngif. <https://angular.io/api/common/NgIf>, 2023. [Online; Abgerufen 12.03.2023].
- [7] Google. What is angular? <https://angular.io/guide/what-is-angular>, 2023. [Online; Abgerufen 12.03.2023].
- [8] H. Gudjons. *Frontalunterricht - neu entdeckt: Integration in offene Unterrichtsformen*. Uni-Taschenbücher. utb GmbH, 2021.
- [9] Marc Hassenzahl, Franz Koller, and Michael Burmester. Der user experience (ux) auf der spur: Zum einsatz von [www.attrakdiff.de](http://www.attrakdiff.de). In Henning Brau, Sarah Diefenbach, Marc Hassenzahl, Franz Koller, Matthias Peissner, and Kerstin Röse, editors, *Tagungsband UP08*, pages 78–82, Stuttgart, 2008. Fraunhofer Verlag.
- [10] Ellen Johanna Helsper and Rebecca Eynon. Digital natives: Where is the evidence? *British Educational Research Journal*, 36(3):503–520, 2010.
- [11] Carsten Mohs, Jörn Hurtienne, Martin C Kindsmüller, Johann Habakuk Israel, Herbert A Meyer, et al. Iuui-intuitive use of user interfaces: Auf dem weg zu einer wissenschaftlichen basis für das schlagwort „intuitivität“. *MMI-Interaktiv*, 11(11):75–84, 2006.
- [12] mongodb. `db.collection.findOne()` - mongodb manual. <https://www.mongodb.com/docs/manual/reference/method/db.collection.findOne/>, 2023. [Online; Abgerufen 12.03.2023].

- [13] mongodb. `db.collection.findOneAndUpdate()` - mongodb manual. <https://www.mongodb.com/docs/v5.2/reference/method/db.collection.findOneAndUpdate/>, 2023. [Online; Abgerufen 12.03.2023].
- [14] mozilla. `tabindex` - html: Hypertext markup language | mdn. [https://developer.mozilla.org/en-US/docs/Web/HTML/Global\\_attributes/tabindex?retiredLocale=de](https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes/tabindex?retiredLocale=de), 2023. [Online; Abgerufen 12.03.2023].
- [15] Fiona Fui-Hoon Nah, Qing Zeng, Venkata Rajasekhar Telaprolu, Abhishek Padmanabhuni Ayyappa, and Brenda Eschenbrenner. Gamification of education: A review of literature. In Fiona Fui-Hoon Nah, editor, *HCI in Business*, pages 401–409, Cham, 2014. Springer International Publishing.
- [16] Svenja Noichl, Nadine Bergner, and Ulrik Schroeder. Zielgruppengerechte App-Icons für Seniorinnen und Senioren. In *Bildungsräume : DeLFI 2017 - die 15. e-Learning Fachtagung Informatik der Gesellschaft für Informatik, 5. bis 8. September 2017, Chemnitz / Christoph Igel, Carsten Ullrich, Martin Wessner (Hrsg.)*, volume P273 of *GI-Edition : lecture notes in informatics*, pages 117–123, Bonn, Sep 2017. Die 15. e-Learning Fachtagung Informatik, Chemnitz (Germany), 5 Sep 2017 - 8 Sep 2017, Gesellschaft für Informatik e.V. (GI).
- [17] Svenja Noichl, René Röpke, and Ulrik Schroeder. Adaptive buttons für zielgruppen-gerechtes app design. In Raimund Dachselt and Gerhard Weber, editors, *Mensch und Computer 2018 - Workshopband*, Bonn, 2018. Gesellschaft für Informatik e.V.
- [18] Svenja Noichl and Ulrik Schroeder. Icons für seniorinnen und senioren – universell vs. adaptiv. In Raimund Dachselt and Gerhard Weber, editors, *Mensch und Computer 2018 - Workshopband*, Bonn, 2018. Gesellschaft für Informatik e.V.
- [19] Svenja Noichl and Ulrik Schroeder. InfoBiTS : Informatische Bildung für Technikferne Seniorinnen und Senioren. In *DELFI 2020 : die 18. Fachtagung Bildungstechnologien der Gesellschaft für Informatik e.V / Hrsg. Zender, Raphael; Ifenthaler, Dirk; Leonhardt, Thiemo; Schumacher, Clara; Gesellschaft für Informatik e.V.*, volume P-308 of *GI-Edition. Proceedings*, pages 385–386, Bonn, Sep 2020. 18. Fachtagung Bildungstechnologien der Gesellschaft für Informatik e.V., online, 14 Sep 2020 - 18 Sep 2020, Köllen.
- [20] Nils Pulte. Entwicklung eines generators für zugängliche gamebook-uis zur anwendung in der digitalen lehre. *Bachelorarbeit*, RWTH Aachen University, 2022.
- [21] I. Rathgeber. *Werbung von gestern bis morgen: Die Akzeptanz personalisierter Werbung*. Diplom.de, 2004.
- [22] Daniel Reinhardt and Jörn Hurtienne. Measuring intuitive use: Theoretical foundations. *International Journal of Human-Computer Interaction*, 0(0):1–31, 2023.
- [23] Julian Reschke Roy T. Fielding, Mark Nottingham. Http semantics, 9.3 method definitions. <https://www.rfc-editor.org/rfc/rfc9110.html#name-method-definitions1>, 2023. [Online; Abgerufen 12.03.2023].
- [24] RxJS. `Rxjs - behaviorsubject`. <https://rxjs.dev/api/index/class/BehaviorSubject>, 2023. [Online; Abgerufen 12.03.2023].

- [25] Dirk Schulze, Denise Prescher, Claudia Loitsch, Martin Spindler, and Gerhard Weber. Vorlesungsinhalte inklusive. barrierefreiheit in virtuellen lernumgebungen. In *Postgraduale Bildung mit digitalen Medien. Fallbeispiele aus den sächsischen Hochschulen*, pages 121–129. 2014.
- [26] Sass team. Sass: Documentation. <https://sass-lang.com/documentation/>, 2023. [Online; Abgerufen 12.03.2023].
- [27] Sass team. Sass: @each. <https://sass-lang.com/documentation/at-rules/control/each>, 2023. [Online; Abgerufen 12.03.2023].
- [28] Sass team. Sass: @import. <https://sass-lang.com/at-rules/import/>, 2023. [Online; Abgerufen 12.03.2023].
- [29] Martijn van Welie, Gerrit C. van der Veer, and Anton Eliëns. Patterns as tools for user interface design. In Jean Vanderdonckt and Christelle Farenc, editors, *Tools for Working with Guidelines*, pages 313–324, London, 2001. Springer London.
- [30] W3C. World wide web consortium launches international program office for web accessibility initiative. <https://www.w3.org/Press/IPO-announce>, 1997. [Online; Abgerufen 06.02.2023].
- [31] W3C. Web content accessibility guidelines 2.1. <https://www.w3.org/TR/WCAG21/>, May 2018. [Online; Abgerufen 03.11.2022].
- [32] W3School. Html dom document getelementbyid(). [https://www.w3schools.com/jsref/met\\_document\\_getelementbyid.asp](https://www.w3schools.com/jsref/met_document_getelementbyid.asp), 2023. [Online; Abgerufen 06.03.2023].
- [33] W3School. Responsive web design - media queries. [https://www.w3schools.com/css/css\\_rwd\\_mediaqueries.asp](https://www.w3schools.com/css/css_rwd_mediaqueries.asp), 2023. [Online; Abgerufen 06.03.2023].
- [34] Wolff-Dietrich Webler. «gebt den studierenden ihr studium zurück!» Über selbststudium, optimierende lernstrategien und autonomes lernen (in gruppen). *Beiträge zur Lehrerbildung*, 23(1):22–34, 2005.
- [35] N.J. Yeager and R.E. McGrath. *Web Server Technology*. Elsevier Science, 1996.



# Eidesstattliche Versicherung

Dietrichs, Jan

394439

\_\_\_\_\_  
Name, Vorname

\_\_\_\_\_  
Matrikelnummer (freiwillige Angabe)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/  
Masterarbeit\* mit dem Titel

Kontextbasierte UI-Anpassung für zielgruppenspezifische Anforderungen an digitale  
Gamebooks

selbstständig und ohne unzulässige fremde Hilfe (insbes. akademisches Ghostwriting) erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Aachen , 21. März 2023

\_\_\_\_\_  
Ort, Datum

\_\_\_\_\_  
Unterschrift

\*Nichtzutreffendes bitte streichen

## **Belehrung:**

### **§ 156 StGB: Falsche Versicherung an Eides Statt**

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

### **§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt**

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

Aachen , 21. März 2023

\_\_\_\_\_  
Ort, Datum

\_\_\_\_\_  
Unterschrift

