

Projektbericht
Studiengang : Informatik

Eine simple Lernplattform als responsive Webanwendung

von

Samuel Stein

78106

Betreuender Professor: Dr. Marc Hermann

Einreichungsdatum : 15. August 2021

Eidesstattliche Erklärung

Hiermit erkläre ich, **Samuel Stein**, dass ich die vorliegenden Angaben in dieser Arbeit wahrheitsgetreu und selbständig verfasst habe.

Weiterhin versichere ich, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben, dass alle Ausführungen, die anderen Schriften wörtlich oder sinngemäß entnommen wurden, kenntlich gemacht sind und dass die Arbeit in gleicher oder ähnlicher Fassung noch nicht Bestandteil einer Studien- oder Prüfungsleistung war.

15.08.2021

Ort, Datum

A handwritten signature in black ink, appearing to read 'Samuel Stein', written in a cursive style.

Unterschrift (Student)

Kurzfassung

In diesem Projekt geht es um die Erstellung einer einfachen responsiven Lernplattform in Form einer Webanwendung. Es behandelt die Grundlagen im Frontend und Backend sowie die Integration von User Experience und User Interface Design. Des Weiteren werden Entwürfe für eine mögliche Webanwendung visualisiert und die Umsetzung wichtiger Funktionen gezeigt.

Inhaltsverzeichnis

Eidesstattliche Erklärung	i
Kurzfassung	ii
Inhaltsverzeichnis	iii
Abbildungsverzeichnis	vi
Quelltextverzeichnis	vii
Abkürzungsverzeichnis	viii
1. Einleitung	1
1.1. Motivation	1
1.2. Ziel der Arbeit	1
1.3. Vorgehen	1
2. Theoretische Grundlagen	3
2.1. E-Learning auf Lernplattform	3
2.2. Backend	4
2.2.1. MySQL	4
2.2.2. Node.js	5
2.2.3. REST API/HTTP Requests	5
2.2.4. Express.js	7

2.2.5. Docker	8
2.3. Frontend	9
2.3.1. HyperText Markup Language	9
2.3.2. Cascading Style Sheets	9
2.3.3. JavaScript	10
2.3.4. React	10
2.3.5. Bibliotheken	12
2.4. User Experience und User Interface Design	14
2.4.1. User Interface Design (UI)	14
2.4.2. User Experience (UX)	16
3. Entwurf	18
3.1. Ziel der Webanwendung	18
3.2. Aufbau der Webanwendung	18
3.3. Backend	18
3.4. Webserver zum Testen	19
3.5. Planung des User Interfaces	20
3.5.1. Entwurf der Seiten	21
3.6. Benutzergruppen	23
3.7. Funktionen	26
3.8. Anwendungsfälle	28
4. Implementierung der Webanwendung	30
4.1. Installation und Konfiguration	30
4.2. Implementierung	33
4.2.1. JSON Web Token zum Authentifizieren	33
4.2.2. Responsive Designs	34

5. Zusammenfassung und Ausblick	37
5.1. Zusammenfassung	37
5.2. Erreichte Ergebnisse	37
5.3. Denkanstoß	37
Literatur	38
A. Anhang A	40
B. Anhang B	41

Abbildungsverzeichnis

2.1. CSS Syntax [3, S. 2]	9
3.1. Farbpalette der Webanwendung	20
3.2. Entwurf der Homepage	22
3.3. Entwurf der Loginpage	23
3.4. Entwurf des Dashboards	24
3.5. Anwendungsfall 1	28
3.6. Anwendungsfall 1 - Tabelle	28
3.7. Anwendungsfall 2	29
3.8. Anwendungsfall 2 - Tabelle	29
4.1. Ordnerstruktur der Homepage	31
4.2. Responsive Ansicht der Homepage	35
4.3. Responsive Ansicht des Dashboards	36

Listings

2.1. Usertabelle mit Sequelize erstellen	13
4.1. AuthMiddleware.js	33
4.2. CSS Grid über 736 Pixel	34
4.3. CSS Grid unter 736 Pixel	34

Abkürzungsverzeichnis

1. Einleitung

Dieses Kapitel befasst sich mit der Motivation der Arbeit. Darauf folgen das Ziel und die Gliederung.

1.1. Motivation

Um einen Einblick in verschiedene Bereiche zu bekommen, bieten Lernplattformen im Internet eine große Bühne. Das Lernen über Webanwendungen kann dabei auf jede Person angepasst werden.

Doch wie funktionieren diese Lernplattformen? Wie werden sie programmiert und welche Kenntnisse sind erforderlich? Es stellt sich auch die Frage, wie responsive Webanwendungen gestaltet werden.

Um detaillierte Antworten auf diese Frage zu erhalten, kam die Motivation auf, das Thema Eine simple Lernplattform als responsive Webanwendung in dem Modul IN-Projekt von Dr. Hermann zu wählen.

1.2. Ziel der Arbeit

Ziel der Arbeit ist es, zu zeigen, wie responsive Webanwendungen im Rahmen des Lernens erstellt werden können. Sie soll aber auch zeigen, wie diese im Allgemeinen erstellt werden und welche Schwierigkeiten dabei aufgezeigt werden können. Es werden die Grundlagen zum Programmieren der Webanwendung und deren Funktionen gezeigt. Darüber hinaus werden Methoden und Merkmale moderner Designs und responsiver Webanwendungen vorgestellt.

1.3. Vorgehen

Zu Beginn der Arbeit werden Definitionen zum Thema Webanwendungsprogrammierung erläutert.

Anschließend wird aufgezeigt, wie die Nutzerwahrnehmung im Zusammenhang mit der Nutzererfahrung und der Gestaltung von Benutzeroberflächen gezielt beeinflusst werden kann.

Im Folgenden werden Entwürfe und Anwendungsfälle der Lernplattform als Webanwendung vorgestellt. Zusätzlich werden Nutzergruppen und Funktionen näher beschrieben.

Daraufhin wird auf die Implementierung und Konfiguration der Webanwendung eingegangen.

Abschließend wird eine Zusammenfassung des gesamten Themas verfasst. Es werden Überlegungen zum zukünftigen Potenzial von Lernplattformen als Webanwendungen diskutiert.

2. Theoretische Grundlagen

2.1. E-Learning auf Lernplattform

Eine Lernplattform kann auch als E-Learning-Plattform bezeichnet werden. „E-Learning ist eine Alternative zum traditionellen Lernen und beschreibt das Lernen mit Unterstützung von digitalen elektronischen Werkzeugen und Medien.“[14, S. 55] . Auch schon im Jahr 2007 war E-Learning von großer Bedeutung für die Weiterentwicklung des Bildungswesens: „E-Learning gibt uns endlich die bisher nie da gewesene Change zur Öffnung neuer Wege für die Bildung, zur Steigerung der Bildungsqualität und zugleich zur deutlichen und dauerhaften Rationalisierung des Ressourceneinsatzes in den westlichen Bereichen des Bildungswesen.“[22] . Im Endeffekt ist eine webbasierte Lernplattform eine Ressource zur Verbesserung der Qualität der Bildung. Aber nicht nur für Schulen und Universitäten, sondern auch für Unternehmen können E-Learning-Plattformen und die damit verbundenen Methoden eine große Rolle spielen, insbesondere in Krisenzeiten. Der Ausbruch des Corona-Virus im Jahr 2019 hat besonders deutlich gemacht, wie wichtig es ist, flexibel zu sein, um die Gesundheit der Menschen zu schützen. Doch diese Lernplattformen bieten neben der Verhinderung der Ausbreitung von schweren Krankheiten noch weitere Vorteile. Der größte Vorteil ist jedoch, dass das Lernen perfekt auf den Nutzer zugeschnitten werden kann. Zum einen hat der Nutzer die Möglichkeit, Zeit und Ort für das Lernen zu wählen, wenn er an bestimmten Themen oder Aufgaben arbeitet. Das macht es vor allem für Menschen, die nicht die Möglichkeit haben, regelmäßig am Unterricht teilzunehmen, viel zugänglicher. Auf Lernplattformen haben die Nutzer auch die Möglichkeit des individualisierten Lernens. Dabei stehen das Lerntempo und das eigene Interesse im Vordergrund. Es sei an dieser Stelle erwähnt, dass dies insbesondere beim Thema „Selbststudium von Interesse ist.

Um eine responsive Lernplattform Fullstack selbst programmieren zu können, muss eine Vielzahl von verschiedenen Grundlagen verstanden werden. Unter Fullstack versteht man die Gesamtheit einer Anwendung, die sowohl das Frontend als auch das Backend umfasst. Um das Ganze jedoch möglichst modern zu gestalten, müssen auch die später beschriebenen Theorien und Methoden der User Experience und des User Interface Designs berücksichtigt werden.

2.2. Backend

Eine der grundlegenden Fragen, die gestellt werden müssen, ist die folgende. Wie können wichtige Daten dauerhaft und für alle verfügbar gespeichert und verarbeitet werden? Um diese Frage zu lösen, trägt das Backend bei.

Mit dem Backend ist der serverseitige Teil gemeint, der für den Benutzer nicht direkt sichtbar ist. Hier machen der Server, auf dem zum Beispiel die Webanwendung gehostet wird, sowie die zugehörige Datenbank den Großteil aus. Aber auch Anfragen, die vom Frontend kommen, werden dort verarbeitet. Je mehr Daten verarbeitet werden, desto leistungsfähiger muss das Backend sein, um einen regelmäßigen Datenabgleich zu gewährleisten. [19, vlg.]

Folgend werden die Komponenten erläutert, welche für die Programmierung der Lernplattform und deren Funktionen relevant sind.

2.2.1. MySQL

MySQL ist eine kostenlose relationale Datenbank, die es ermöglicht, Daten zu speichern, welche aufbewahrt und verarbeitet werden sollen. Insbesondere ist dies für die Lernplattform als Webanwendung wichtig, um Daten der Nutzer zu speichern, damit zum Beispiel das Anmelde- und Registrierungssystem realisierbar ist. Aber auch Informationen über die Module und Quizze, werden in der relationalen Datenbank gespeichert. Relationale Datenbanken sind Datenbanken, die in Tabellen organisiert sind. [vgl. 1, S. 11ff.] Die Zeilen der Tabellen sind die jeweiligen Objekte und die Spalten bestimmen die Attribute mit einem bestimmten Datentyp, wie zum Beispiel Integer oder Char, dieser. Um Beziehungen zwischen verschiedenen Tabellen herzustellen, werden Schlüssel benötigt. Diese Schlüssel sind sowohl der Primärschlüssel als auch der Fremdschlüssel, die mindestens eine Spalte der Tabelle belegen. Der Primärschlüssel ist in jeder Zeile eindeutig, so dass jedes Objekt einer Tabelle genau bestimmt werden kann. Ein Fremdschlüssel ist ebenfalls eine Spalte einer Tabelle, die ein Primärschlüssel in der gegenüberliegenden Tabelle ist, um Tabellen zu verbinden.

Es gibt verschiedene Arten von Beziehungen zwischen Tabellen. Diese Beziehungen oder Verknüpfungen können 1:1, 1:N oder N:M sein.

Die 1:1-Beziehung beschreibt eine Beziehung zwischen Tabelle A und Tabelle B, dass es für jedes Objekt aus Tabelle A höchstens eine Zeile aus Tabelle B gibt.

Bei einer 1:N-Beziehung kann ein Objekt aus Tabelle A mehreren Objekten aus Tabelle B zugeordnet werden, aber nicht umgekehrt.

In einer N:M-Beziehung kann eine beliebige Anzahl von Objekten aus Tabelle A mit einer beliebigen Anzahl von Objekten der Tabelle B in Verbindung stehen, und umgekehrt. [vgl. 1, S. 12ff.]

Das Wichtigste an MySQL für die Umsetzung des Projekts ist jedoch die Durchführung von Grundoperationen der permanenten Speicherung. Diese werden auch CRUD genannt. Die einzelnen Buchstaben stehen für die Operationen create, read, update und delete. Diese werden verwendet, um Objekte in Tabellen hinzuzufügen, zu lesen, zu ändern oder zu löschen.

2.2.2. Node.js

Node.js ist ein Open-Source-JavaScript-Framework, das asynchrone Ereignisse nutzt, um mehrere Verbindungen gleichzeitig zu verarbeiten. Node.js gilt auch als attraktive Entwicklungsplattform für Webanwendungen und alle anderen Arten von Netzwerkservern. Dabei bietet es die Möglichkeit, serverseitigen JavaScript-Code auszuführen, um die Kommunikation zwischen Client und Server zu gewährleisten. [vgl. 6, S. 7] Node.js weist dabei mehrere Vorteile auf. Diese sind hohe Leistung und Skalierbarkeit. Außerdem ist es in der Lage, große Datenbankabfragen mit hoher Performance zu implementieren. [vgl. 25, S. 3]. Dabei genießt Node.js einen hohen Bekanntheitsgrad. So nutzen kleinere, aber auch große Unternehmen wie Paypal, Netflix oder Microsoft Node für Projekte unterschiedlicher Größe. [vgl. 6, S. 1]

Eine weitere Besonderheit an Node.js ist der Node.js Package Manager (NPM). NPM ermöglicht es, einfache Module von Drittanbietern einzubinden, indem es das Herunterladen und Einbinden dieser Module vereinfacht. [vgl. 6, S. 31] Um Module von Drittanbietern einzubinden, muss lediglich der Befehl *npm install modulname* verwendet werden. Dieser Node.js-Paketmanager ist für die spätere Implementierung der Lernplattform wichtig.

2.2.3. REST API/HTTP Requests

Das Hypertext Transfer Protocol (HTTP) wird verwendet, um Daten über Anfragen zu empfangen oder zu senden. HTTP ist ein Protokoll, das die Kommunikation zwischen Servern und Clients vorschreibt. Wenn der Client Informationen oder eine Datei vom Server erhalten möchte, sendet er eine sogenannte HTTP-Request (dt. Anfrage) an den Server. Im Rahmen des Projekts werden hauptsächlich die GET-, POST- und DELETE-Anfragen verwendet.

Die GET-Request fordert eine bestimmte Ressource vom Server zum Client an.

Die POST-Request wird verwendet um Ressourcen vom Client zum Server zu übermitteln.

Die DELETE-Request löscht bestimmte Ressourcen beim Server.

Nachdem der Server eine Anfrage vom Client erhalten hat, sendet er eine Antwort an den Client zurück. Die relevantesten Antworten für einen erfolgreichen Austausch sind diejenigen im Bereich 200 bis 299. Antworten zwischen 300 und 399 sind ein Zeichen dafür, dass die Anfrage weitergeleitet wird. Zwischen 400 und 599 werden Fehler gekennzeichnet. Antworten zwischen 400 und 499 sind Fehler, welche vom Client stammen und 500 bis 599 Fehler, welche vom Server ausgehen.

API steht für Anwendungsprogrammierschnittstelle. Sie wird von einem Softwaresystem bereitgestellt, damit externe Anwendungen mit einem bestimmten Teil des Softwaresystems arbeiten beziehungsweise kommunizieren können. Der Prozess einer API sieht wie folgt aus.

Als erstes sendet eine clientseitige Anwendung eine Anfrage an die API über den Uniform Resource Identifier (URI). Wenn eine gültige Anfrage an die API gestellt wird, wird sie an den Webserver weitergeleitet. Dieser Webserver sendet eine Antwort zurück. Abschließend überträgt die API die angeforderten Daten. [13, vgl.]

REST steht für Representational State Transfer und ist ein Architekturstil, der für die Erstellung und Organisation verteilter Systeme definiert wurde. Grundsätzlich wird eine REST-API verwendet, um ein verteiltes System in mehreren Bereichen zu verbessern. Dabei soll der Einsatz von REST und die damit verbundene Einfachheit und Effizienz zu einer Leistungssteigerung, Skalierbarkeit der Komponenten und einfacheren Interaktion zwischen den Systemen führen. [vgl. 9, S. 3] Laut dem bekannten amerikanischen Informatiker Roy T. Fieldings, einem der Hauptautoren der HTTP-Spezifikationen und dem Begründer des REST-Architekturstils, gibt es zwei Arten, den Prozess der Architekturgestaltung zu betrachten. Die erste konzentriert sich auf Kreativität und unbegrenzte Visionen, bei der ein Designer die Möglichkeit hat, etwas aus dem Nichts zu schaffen. Die zweite Sichtweise besteht darin, mit den Systemanforderungen zu beginnen, ohne konkrete Vorgaben zu machen. Das REST-Konzept bezieht sich auf die zweite Sichtweise und bietet Flexibilität. [vgl. 24, S. 77]

REST besteht aus definierten Gestaltungsprinzipien, welche die reibungslose Kommunikation zwischen dem Client, der auf die API zugreifen möchte, und dem Server, der die Ressourcen der API anbietet, sicherstellen sollen. Es gibt sechs Prinzipien.

Einheitliche Schnittstelle: Dieselben Daten sollten mit einem einheitlichen Ressourcenbezeichner (URI) verknüpft werden. Zum Beispiel sollte alles, was zu Benutzern

gehört, in den URI `/users` eingebunden werden. Die Gestaltung der URIs sollte denselben Mustern folgen.

Entkopplung von Client und Server: Client und Server sollten voneinander unabhängig sein. Clients sollten nur dann auf bestimmte Ressourcen zugreifen können, wenn ihnen der URI bekannt ist. Andererseits sollten Serveranwendungen die Clientanwendungen nur ändern, wenn die angeforderten Daten übergeben werden, und nicht anders.

Zustandslosigkeit: Jede Anfrage an die REST-API sollte alle Informationen enthalten, um zustandslos zu sein.

Cache-Fähigkeit: Daten-Caching führt in diesem Zusammenhang zu einer verbesserten Leistung und Skalierbarkeit auf der Client-Seite und beschreibt die Zwischenspeicherung von Ressourcen.

Mehrschichtige Systemarchitektur: Aus Sicherheitsgründen sollten die Anfragen nicht direkt erkennen lassen, ob sie mit einer Endanwendung oder mehreren Zwischenschnittstellen kommunizieren.

Code auf Anfrage: Auch hier sollte der Code aus Sicherheitsgründen optional nur dort ausführbar sein, wo er erwünscht ist, damit kein bössartiger Code integriert werden kann. [13, vgl.]

Schließlich können die angeforderten Ressourcen in verschiedenen Formaten bereitgestellt werden. In der späteren Implementierung wird die JavaScript Object Notation (JSON) verwendet. Das Konzept von JSON ist recht einfach gehalten und bietet die Möglichkeit, dass die Ein- und Ausgabe auch von Menschen leicht gelesen werden kann. Das Zusammenspiel von REST API und JSON wird auch bei der Implementierung der Authentifizierung von Benutzern im Laufe der Ausarbeitung deutlich.

2.2.4. Express.js

„Express ist ein minimalistisches und flexibles Node.js-Framework für Webanwendungen, das eine Reihe von robusten Funktionen für Web- und mobile Anwendungen bietet.“ [11] Express.js enthält viele Verbindungskomponenten und löst zahlreiche Aufgaben, wie z. B. HTTP-Anfragen, und bietet die Möglichkeit, sich mit verschiedenen Arten von Datenbanken zu verbinden. [vgl. 5, S. 2] Darüber hinaus interpretiert Express die HTTP-Methoden, URIs zu Routen und sowohl Eingabe- als auch Ausgabedaten. [vgl. 5, S. 54] Express bildet somit das Backend mit und wickelt alle Interaktionen zwischen der Datenbank und dem Frontend ab, um eine reibungslose

Übertragung zu gewährleisten.

Routen sind ein Teil des Express-Codes und bestehen jeweils aus einer HTTP-Methode wie GET, POST, PUT oder DELETE, einem Endpunkt als URI-PFAD und einer Funktion, die diese verbindet. [15, vgl.] Dabei wird festgelegt, wie die Client-Anforderungen an einen bestimmten Endpunkt reagieren. Express hat dabei meist eine Hauptdatei, welche im folgenden Projekt als `server.js` benannt wird. Anschließend können in der Express.js-App Einstellungen vorgenommen werden, um Erweiterungen vorzunehmen.

2.2.5. Docker

Docker sind als kostengünstige und schnelle Option für die Lernplattform wichtig. „Docker ist eine offene Plattform für die Entwicklung, Bereitstellung und Ausführung von Anwendungen [...] und ermöglicht es, Anwendungen von der Infrastruktur zu trennen, so dass Software schnell bereitgestellt werden kann.“[8] Dies alles ist möglich, indem die Anwendung in einer isolierten Umgebung, einem sogenannten Container, ausgeführt wird. In diesen Containern können sogenannte Images installiert werden, welche zum Beispiel Datenbanksysteme oder Betriebssysteme sind. Von den Containern können viele Images gleichzeitig mit wenigen Befehlen auf jedem Betriebssystem gestartet und verwaltet werden. [8, vgl.] In erster Linie wird im Rahmen des Projekts Docker verwendet, um die MySQL-Datenbank als Image einzurichten und grundlegend zu verwalten. Darüber hinaus können HTTP-Anfragen über dieses Docker-Image gestellt werden, was den Zugriff auf die Struktur der Datenbank von außen ermöglicht, um Daten hinzuzufügen, zu löschen, zu ändern oder zu lesen.

2.3. Frontend

Das Frontend ist der Teil des Anwendungsprogramms, mit dem der Benutzer direkt interagiert. Hier wird versucht, die Prozesse des Backends als grafische Oberfläche darzustellen und dem Benutzer sowohl die Eingabe des Benutzers als auch die Ausgabe des Benutzers zu zeigen.

2.3.1. HyperText Markup Language

HTML steht für HyperText Markup Language und ist die standardmäßige Sprache für den Browser. Sie ermöglicht es dem Browser, Farben, Schriftarten, Links oder auch Grafiken darzustellen. Letztlich beschreibt HTML die Struktur von Webseiten. Um die Struktur darzustellen, werden sogenannte HTML-Tags verwendet. Einleitende HTML-Tags werden immer in spitze Klammern gesetzt. Daran schließt sich der gewünschte Inhalt an. Abschließende HTML-Tags enthalten innerhalb der Klammern einen Backslash.

Ein Beispiel ist hierbei `<div>Hello World!</div>`. Der Tag `div` wird als eine Art Container für HTML-Elemente verwendet, die später mit CSS gestaltet oder mit JavaScript manipuliert werden können. [23, vgl.] Dies ist für die spätere Gestaltung der Lernplattform von großer Bedeutung. Um das Erscheinungsbild der HTML-Elemente genauer zu beeinflussen, kann es mit Hilfe von CSS unterstützt werden.

2.3.2. Cascading Style Sheets

CSS steht für Cascading Style Sheets und ist eine Sprache, mit der festgelegt wird, wie HTML-Dokumente auf einer Website angezeigt werden sollen. CSS spielt eine wichtige Rolle bei der Gestaltung der Lernplattform. Mit CSS lassen sich Hintergründe, Textstile, Positionen von Elementen und Transformationen oder Animationen ändern und anzeigen. Die Syntax sieht wie folgt aus:

```
Selector
  ↓
.header {
  background-color: red;
  border: 1px solid blue;
}
```

The diagram illustrates the components of a CSS rule. An arrow labeled 'Selector' points to the `.header` part of the rule. Another arrow labeled 'Property' points to the `border` property. A third arrow labeled 'Value' points to the `1px solid blue` value.

Abbildung 2.1.: CSS Syntax [3, S. 2]

Die Selectoren sind in folgender Lernplattform die Klassennamen innerhalb der div-Tags. Um einen genaueren Einblick in die möglichen Eigenschaften und Werte zu erhalten, empfiehlt sich ein Blick in die CSS-Dokumentation von zum Beispiel Mozilla.

Um das Layout der Webanwendung zu erstellen, werden später zwei grundlegende Layoutmodule oder Layoutmethoden verwendet. Diese sind Flexbox und CSS Grid.

Flexbox ist ein eindimensionales Layout, in dem Elemente horizontal oder vertikal positioniert werden können. Das Element befindet sich in einem sogenannten Flexcontainer und wird als Flexelement bezeichnet.

CSS Grid hingegen ist zweidimensional und unterteilt das Layout in ein aus Zeilen und Spalten bestehendes Raster und wird als das leistungsfähigste Layoutsystem bezeichnet. [3, S. 255] Das äußerste Element ist hierbei der Grid Container und die inneren sind die Grid Elemente.

2.3.3. JavaScript

Wenn es darum geht, Websites dynamisch zu gestalten, wird JavaScript als Skript- oder Programmiersprache verwendet, um komplexe Merkmale und Funktionen auf der Website zu programmieren. [16, vgl.] JavaScript bietet die Möglichkeit, mit der bereits erwähnten API zu arbeiten und die in ihr enthaltenen Daten zu verarbeiten. So ist es auch eine gängige Anwendung, dass JavaScript das HTML und CSS der Website dynamisch verändert. Wenn zum Beispiel ein Button gedrückt wird, wird eine JavaScript-Funktion ausgeführt, die den Kontext der Webanwendung ändert. JavaScript findet seine Anwendung sowohl im Front-End als auch im Back-End verwendet.

2.3.4. React

React ist eine Open-Source-JavaScript-Bibliothek, die von Ingenieuren des Unternehmens Facebook erfunden wurde, um komplexe Benutzeroberflächen darzustellen. Die Stärke liegt dabei in den Komponenten, die individuell erstellt und wiederverwendbar sind. Dies erleichtert Webentwicklern die Gestaltung umfangreicher Benutzeroberflächen und gehört zu den am häufigsten verwendeten Frameworks in diesem Bereich. [vgl. 12, S. 1ff.]

Hooks

React bietet Funktionen, die in den Zustand von React und seiner Komponente integriert werden können, ohne dass dafür Klassen erstellt werden müssen. Diese Funktionen werden Hooks genannt. Die folgenden drei Hooks werden häufig in der Webanwendung des Projekts verwendet.

useState: Dieser Hook wird verwendet, um zustandsbezogene Werte zu speichern und verfügt über eine Funktion, um sie zu aktualisieren. Er kann sowohl Datentypen wie Integer und Strings als auch ganze Arrays voller Objekte speichern.

Die Struktur hierfür ist wie folgt: `const [state, setState] = useState();`
`setState` kann dann im späteren Verlauf als Funktion aufgerufen werden

useEffect: `useEffect` kann für Seiteneffekte, wie zum Beispiel für das Loginsystem verwendet werden. Diese Effekte werden innerhalb des Hooks nach jedem Rendering ausgeführt. Es kann aber auch festgelegt werden, ob die `useEffects` nur unter bestimmten Bedingungen ausgeführt werden oder nicht.

useContext: Dieser Hook wird verwendet, um Daten innerhalb der gesamten Komponenten auszutauschen und sie global anzuwenden.

Es können jedoch auch selbst geschriebene Hooks verwendet werden, indem sie in einer Klasse definiert werden.

Eine weitere wichtige Komponente von React sind die sogenannten Props, die für Properties stehen. Sie werden verwendet, um Eigenschaften von einzelnen Komponenten an andere Komponenten zu übergeben und können mit der Java-Vererbung verglichen werden.

React-Router-DOM

React-Router-DOM ist eine grundlegende Bibliothek, die für Webanwendungen verwendet werden kann, um Komponenten miteinander zu verbinden. Sie besteht aus drei Hauptkategorien von React-Router-Komponenten. Dies sind Router, Route Matcher und Navigationen. [20, vgl.]

Router bilden die Grundlage einer React-Router-Anwendung, insbesondere wenn es um Webprojekte geht. `BrowseRouter` verwendet die bekannten normalen URL-Pfade.

Route Matchers bestehen aus zwei Komponenten, um URL-Änderungen zu ermögli-

chen und Unterseiten aufzurufen. Diese Komponenten sind `switch` und `route`. Wenn ein `<Switch>` gerendert wird, wird das untergeordnete `<Route>`-Element durchsucht, um den Pfad der URL sicherzustellen. Wird eine übereinstimmende Route-URL gefunden, so wird diese gerendert und alle anderen Pfade werden ignoriert.

Navigationen: Die Komponente `<link>` erzeugt einen Anker zu einer anderen Webseite. Dieser entspricht dem Anker aus dem HTML-Dokument.

React-Router DOM bietet auch zwei praktische Hooks, `useHistory` und `useParams`.

Über den `useHistory`-Hook wird der Zugriff auf die History-Instanz gewährt, die zur Navigation zu den Komponenten verwendet werden kann. Zum Beispiel kann die Funktion `push(„URL“)` verwendet werden, um den Benutzer zur gewünschten URL umzuleiten.

Der Hook `useParams` gibt ein Objekt mit einem Schlüssel/Wert-Paar spezifischer URL-Parameter zurück. Damit ist es zum Beispiel möglich, auf bestimmte IDs innerhalb der URL zuzugreifen und den Kontext auf Basis der ID zu ändern.

2.3.5. Bibliotheken

Um die Implementierung der Webanwendung zu vereinfachen, werden einige Bibliotheken verwendet, die über hilfreiche Funktionen verfügen.

Axios ist zum Beispiel ein HTTP-Client für `node.js` und den Browser. Auf der Client- bzw. Browserseite verwendet er das `XMLHttpRequest` und auf der Serverseite das native `node.js` `http`-Modul. Über *Axios* können die Anfragen an diese gestellt werden. Darüber hinaus bietet *Axios* einen Schutz gegen Cross-Site-Request-Forgery, der verhindert, dass unautorisierte Befehle vom Benutzer ausgeführt werden können. Wie *Axios* verwendet wird, wird im Kapitel über die Implementierung näher erläutert. Dieser Client wird primär zum Senden von Informationen in der Webanwendung verwendet.

Die Bibliothek *BCrypt* wird zum Hashing von Benutzerpasswörtern verwendet. Dies erfolgte, weil das Hashing von Passwörtern heutzutage ein grundlegender Baustein der IT-Sicherheit für Anwendungen, insbesondere im Web, ist.

Sequelize ORM vereinfacht die Transaktionsunterstützung bei der Verwendung von MySQL. [21]. Sequelize kann auch verwendet werden, um Tabellen für die Datenbank zu erstellen, sobald das Projekt mit der Datenbank verbunden ist.

```
1 module.exports = (sequelize, DataTypes) => {
2   username: {
3     type: DataTypes.STRING,
4     allowNull: false,
5   },
6   password: {
7     type: DataTypes.STRING,
8     allowNull: false,
9   },
10  });
11  return Users;
12 };
```

Quelltext 2.1: Usertabelle mit Sequelize erstellen

YUP ist ein JavaScript-Schema-Builder für die Validierung von Werten. Er kann zum Beispiel verwendet werden, um bei der Registrierung Mindestanforderungen wie die Länge des Passworts festzulegen.

Das *JSON Web Token (JWT)* ist für die Autorisierung und den Informationsaustausch auf der Lernplattform zuständig. JWT ist ein Standard (RFC 7519), der Methoden für die sichere Übertragung von Informationen in Form eines JSON-Objekts definiert. Die in diesem Objekt enthaltenen Informationen können vom Server überprüft und als vertrauenswürdig eingestuft werden. [4, vgl.]

Sobald der Benutzer angemeldet ist, enthält jede Anfrage das JSON-Web-Token für den Zugriff auf Routen, Dienste oder Ressourcen. Außerdem können so Informationen sicher zwischen Parteien oder Komponenten ausgetauscht und übertragen werden.

2.4. User Experience und User Interface Design

Damit eine Webanwendung bei den Nutzern gut ankommt, kommt es nicht allein auf gute Funktionalitäten, sondern auch auf das richtige Aussehen der Website an. Die Bewertung und Wahrnehmung der Website durch die Nutzer besteht im Wesentlichen aus dem Inhalt, der Benutzerfreundlichkeit und der Ästhetik. [vgl. 17, S. 1]

Untersuchungen der Monash University in Australien haben gezeigt, dass die Gestaltung der Website eines Unternehmens besonders wichtig für dessen Image sein kann. So spielten beispielsweise eine einfache Navigation und nützliche Informationen auf einer Website eine signifikante Bedeutung. [7]

Bei der Gestaltung der modernen Webanwendungen spielen User Experience (UX) und User Interface Design (UI) eine entscheidende Rolle.

2.4.1. User Interface Design (UI)

Bei der Gestaltung von Benutzeroberflächen geht es in erster Linie darum, wie sich der Benutzer beim Betrachten des Entwurfs verhält. Es wird versucht, eine Art Hierarchie der Aufmerksamkeit zu bilden, um die wichtigsten Informationen für den Benutzer auf der Grundlage seines Verhaltens hervorzuheben. Der Ansatz wird durch die Verwendung von Farben, Schriftarten und Schriftgrößen untermauert. Wenn alle diese Attribute beispielsweise für einen Text gleich gewählt werden, ist es relativ schwierig, den Benutzer und sein Leseverhalten zu beeinflussen. Deshalb müssen innerhalb eines Textes unterschiedliche Helligkeiten von Farben, Schriftgrößen und Schriftarten verwendet werden. Aber auch die Größen und Farben von Buttons oder Eingabefeldern lenken die Aufmerksamkeit des Benutzers auf diese. [10, vgl.]

Auch das Layout der Website sollte berücksichtigt werden. Ein Layout mit verschiedenen Formen und Bildern ist für die Nutzer viel interessanter als ein reiner Blocktext. Auch die Formulare sollten weder zu lang noch zu kurz gewählt werden, um das Lesen angenehmer zu gestalten.

Die Ausrichtung des Texts zwischen dem Titel und dem Haupttext ist ein weiterer wichtiger Aspekt. Um mehr Design und Kohärenz mit der Ausrichtung zu schaffen, sollten der Titel und der Haupttext die gleiche Ausrichtung haben. Eine Verringerung der Anzahl der Ausrichtungen wirkt sich positiv auf das Layout des Entwurfs aus.

Ein weiterer Punkt ist der White Space (dt. Weißraum), das heißt die Leerräume um Texte oder Elemente herum. Dieser wird genutzt, um einen hochwertigeren

Eindruck des Produkts zu vermitteln.

Die Zielgruppe hingegen ist einer der wichtigsten Punkte bei der Auswahl des Designs. Es ist wichtig, herauszufinden, welche Art von Bildern, Farben und Formen bestimmte Zielgruppen ansprechen. [10, vgl.]

Farbtheorie

Farben sind ein besonders wichtiger Bestandteil des täglichen Lebens. In diesem Zusammenhang haben Farben einen starken Einfluss auf Emotionen und Gefühle. Sie können auch in Bezug auf die Temperatur beschrieben werden, zum Beispiel als warm oder kühl. Warme Farben sind zum Beispiel gelb, orange oder rot und gelten als aktiv und anregend. Kühle Farben hingegen sind grün, blau oder violett und gehören zu den entspannenden und ruhigen Farben. Laut einer Studie wurden Hauptfarben wie Rot, Gelb und Grün von den Teilnehmern zu 79,6 Prozent als positiv empfunden. Im Vergleich dazu wurden Mischfarben wie Gelb-Rot, Grün-Gelb zu 64,5 Prozent positiv wahrgenommen, und achromatische Farben wie Weiß, Grau und Schwarz wurden nur von 29,2 Prozent der Teilnehmer als positiv empfunden. [18, S. 5] Die Farbtheorie beschäftigt sich mit den richtigen Farbkombinationen. Um zu bestimmen, welche Farbkombinationen für das Design zu wählen sind, müssen die Zielgruppe und die Emotionen, die beim Benutzer ausgelöst werden sollen, berücksichtigt werden.

Im Folgenden wurde die Lernplattform in der Farbe Blau gestaltet, weil diese auch eine hohe Anzahl positiver Reaktionen hervorrufen kann. Diese sind Glück, Gelassenheit, Ruhe und Hoffnung oder auch das Gefühl von Geborgenheit. Die Gründe dafür sind die Assoziationen, die mit Wasser, Himmel, Meer und Strand verbunden sind. [18]

Um geeignete Farben zum Kombinieren zu finden, gibt es mehrere wissenschaftliche Methoden. - Analogische Farben: Geeignet für die Navigationsleiste und den Hauptteil der Website oder das Logo und seinen Hintergrund, um die Harmonie zu betonen. - Komplementärfarben werden verwendet, wenn Farben hervorgehoben werden sollen. Sie sollten jedoch nicht für die Farbe der Schrift und des Hintergrunds verwendet werden, da dies für den Leser verwirrend wirken kann.

Es können Farbpaletten verwendet werden, wie zum Beispiel die triadische Farbpalette, die aus drei verschiedenen besteht, oder das perfekte Quadrat, das aus vier verschiedenen Farben besteht. Adobe Color bietet beispielweise diese Funktionen, um die richtigen Farben auszuwählen. [2]

Typography

Auch die Schriftarten und Schriftgrößen können beim Leser einen fundamentalen Eindruck hinterlassen. Im Bereich der Typografie gibt es zwei große Schriftarten. Eine davon ist die Serif-Familie, die von historischen Stein- und Marmorschnitzereien inspiriert wurde und am unteren Ende eine Art Fuß hat. Diese Füße wurden geschaffen, weil ein 90-Grad-Winkel damals am Ende nicht möglich war. Diese Schriftfamilie wird verwendet, um Designs seriöser, autoritärer oder etwas älter aussehen zu lassen. Diese Familie ist jedoch auch in mehrere kleinere Untergruppen unterteilt, die als Schriftarten bezeichnet werden und die ebenfalls moderner aussehen oder einen Übergangscharakter haben usw. Verschiedene Schriften können, wie auch Farben, verschiedene Stimmungen unterstreichen. [10, vgl.]

Auf der anderen Seite gibt es die Familie der San-Serifs, die als vertrauter, freundlicher und zugänglicher gilt. Der Grund dafür sind die perfekten rechten Winkel innerhalb der Buchstaben. Auch hier gibt es mehrere Unterfamilien. Außerdem geht es darum, wie leicht und angenehm die Wörter zu lesen sind.

Auch die Kombination von Schriftarten kann für das Design nützlich sein, aber es ist auch wichtig zu berücksichtigen, wie sehr sich die gewünschte Emotion zwischen ihnen unterscheidet.

2.4.2. User Experience (UX)

Das Ziel der User Experience ist es, eine gute Erfahrung mit der Website zu machen. Es wird versucht, ein mögliches Gefühl der Einfachheit und Mühelosigkeit zu schaffen und Dinge zu vermeiden, die für den Benutzer aufdringlich sind. Um dies zu erreichen, können die folgenden vier Dinge berücksichtigt werden.

Einfachheit: Der Grundgedanke ist, die Informationen nicht verwirrend und komplex zu gestalten, da dies dazu führen kann, dass die Nutzer überfordert sind.

Konsistenz: Sowohl das Design als auch die Funktionalitäten sollten identisch aussehen, damit sich der Nutzer an die verschiedenen Unterseiten gewöhnt.

Lesemuster berücksichtigen. Es gibt viele verschiedene Lesemuster, die berücksichtigt werden können, um das natürliche Leseverhalten der Nutzer zu verstehen.

Keine falschen Versprechungen: Die Nutzer sollten zum Beispiel nicht das Gefühl haben, beim Kauf von Produkten betrogen worden zu sein.

Plattformdesign/Reaktionsfähigkeit: Responsive Design bezieht sich darauf, wie der Inhalt der Webanwendung auf verschiedenen Geräten angezeigt wird. In der Vergangenheit wurden Designs nur im Format des Desktops erstellt. Heutzutage

werden jedoch viele Websites und Webanwendungen von mobilen Geräten besucht, auf denen ein normales Desktop-Design unübersichtlich aussehen kann und keine Benutzerfreundlichkeit in Bezug auf mobile Geräte bietet. Wie bereits erwähnt, bietet CSS mit CSS Grid und Flexbox die Möglichkeit, ein klares und zielgerichtetes responsives Design zu erstellen. [\[10, vgl.\]](#)

3. Entwurf

Dieses Kapitel definiert sowohl das Backend als auch das Frontend und deren Funktionen im Bezug zu programmierten Lernplattform.

3.1. Ziel der Webanwendung

Ziel des Projekts war es, eine beispielhafte funktionale Lernplattform zu entwickeln, bei der die Nutzer die Möglichkeit haben, zwischen verschiedenen Modulen zu wählen. Bei diesen Modulen handelt es sich sozusagen um Themen, die gelernt werden können und an deren Ende ein Quiz steht, bei dem der Nutzer Punkte sammeln kann. Um eine Wettbewerbsatmosphäre zu schaffen, sollte es eine Rangliste mit den Nutzern geben, die die meisten Punkte gesammelt haben. Das grundlegende Ziel ist also, Wissen auf vereinfachte Weise zu vermitteln.

3.2. Aufbau der Webanwendung

Um die Webanwendung zu realisieren, werden im Backend MySQL für die Datenbank, Node.js und Express.js für die REST-API verwendet. Im Frontend werden Teile von HTML zur Architektur der Webanwendung, CSS zur Gestaltung der Webanwendung und React zur Kombination von JavaScript-Funktionen und Backend verwendet.

3.3. Backend

Wie bereits erwähnt, soll für das Backend eine MySQL-Datenbank erstellt werden. Damit das Frontend mit dem Backend kommunizieren und Daten speichern kann, wird ein Server benötigt. Da es sich bei der zu konzipierenden Lernplattform um eine nicht allzu umfangreiche Webanwendung handelt, die keine große Datenspeicherung erfordert, reicht ein Virtual Private Server aus. Dies ist ein virtueller Webserver, der auf einem physischen Server basiert. Dieser wird bei der Contabo GmbH mit 4 CPU-Kernen, 8 GigaByte Ram und 200 GigaByte SSD bestellt. Um die

MySQL-Datenbank einzurichten, wird ein Docker-Container mit dem MySQL-Image erstellt.

Um die Daten angemessen zu speichern, enthält die Datenbank mehrere Tabellen. Diese Tabellen sind Users, Modules, Quizzes und UserQuizScores.

Users hat als Primärschlüssel eine ID, die für jeden Benutzer inkrementiert wird. Außerdem sind in dieser Tabelle der Benutzername, das Passwort als Hash, die Punktzahl und eine Autorisierungsrolle enthalten.

Module hat lediglich eine ID als Primärschlüssel, einen Titel und eine Beschreibung für das Modul.

Die Tabelle für die Quizze verwendet ebenfalls eine ID als Primärschlüssel. Außerdem wird dort eine Frage mit vier möglichen Antworten, die richtige Antwort und die Punktzahl gespeichert. Die Beziehung zwischen Modulen und Quizfragen ist 1:N. Ein Modul hat also mehr als eine Frage, welche im Quiz dargestellt werden kann. Da für ein Modul mehrere Quiz möglich sein sollen, wird die ModulID als Fremdschlüssel übergeben.

3.4. Webserver zum Testen

Um auf den Virtual Private Server einen Webserver zu involvieren, wurde NGINX installiert. Nach der Installation und Einrichtung des Webservers ist die Webanwendung unter der Adresse **http://62.171.138.202/** erreichbar.

Zum Testen der Benutzergruppen kombiniert mit deren Berechtigungen werden folgende Nutzer angelegt:

Student:

Benutzername: student

Passwort: student

Lehrer:

Benutzername: lehrer

Passwort: lehrer

Admin:

Benutzername: admin

Passwort: admin

3.5. Planung des User Interfaces

Bei der Konzeption des Projekts war die grundlegende Frage, wie die Informationen präsentiert werden sollten. Der Fokus lag dabei auf den Inhalten und Navigationsleisten je nach Desktop- und Mobilgerät. So ist diese für die mobile Ansicht durchgängig an ein einspaltiges Layout angepasst, während es sich für die Desktop-Ansicht ebenfalls um ein zweispaltiges Layout handelt.

Farbenpalette

Wie bereits im Kapitel User Interface Design erwähnt, spielen Farben eine große Rolle bei der Wahrnehmung des Benutzers. Die Webanwendung verwendet grundsätzlich vier Farben zur Gestaltung des Layouts und der Elemente. Diese sind in der folgenden Abbildung dargestellt.

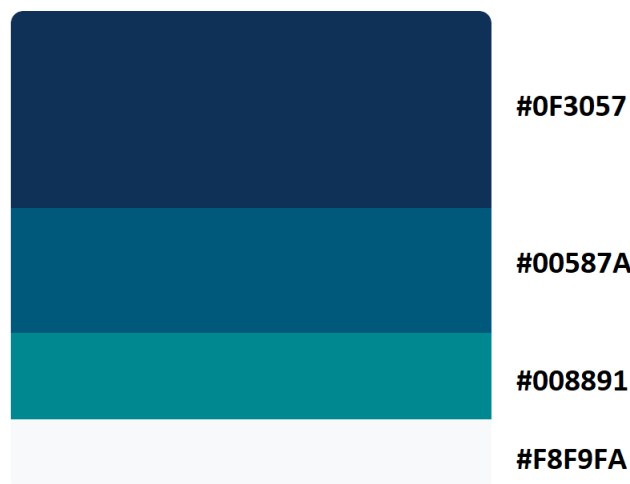


Abbildung 3.1.: Farbpalette der Webanwendung

Die Farben in der Grafik wurden gewählt, da sie grundsätzlich eine positive Emotion mit sich bringen. Blaue Farbtöne können dabei an das Meer, den Ozean und den damit verbundenen Strand erinnern. Die Farbe Blau kann dabei mit den Gefühlen der Sicherheit, des Glücks, Gelassenheit, Frieden oder Hoffnung einhergehen. Das Ziel war es, eine ruhige und nicht zu auffällige Farbe zu wählen, welche den Benutzer zu sehr ablenken könnte.

Typographie

Um beim Benutzer keinen zu autoritären Eindruck zu erwecken und das Ganze modern wirken zu lassen, wurden Schriftarten aus der San-Serif-Familie verwendet. Außerdem haben wichtige Überschriften eine größere Schrift als der Haupttext. Die Schriftfarbe wird hingegen bei Textarten mit dunklem Hintergrund in der Farbe Weiß und bei hellen Hintergründen in der Farbe Schwarz dargestellt.

3.5.1. Entwurf der Seiten

Das Design der Webanwendung beschränkt sich auf drei verschiedene Arten von Designs, um den Benutzer mit den Seitentypen vertraut zu machen. Zunächst gibt es die Hauptseite, die Anmelde- und Registrierungsseite und schließlich das Dashboard mit verschiedenen Unterseiten.

Hauptseite

Die Hauptseite sollte dem Nutzer einen ersten Überblick über die Möglichkeiten der autorisierten Inhalte geben. Der primäre erste Eindruck sollte mit der Registrierung und den damit positiv verbundenen Inhalten verbunden sein.

Grundsätzlich unterscheidet sich die Desktop-Ansicht der Hauptseite nicht von der mobilen Ansicht. Allerdings muss das Bild der Hauptseite für die mobile Ansicht responsive angepasst werden.

Es gibt zwei Arten der Hauptseite. Einmal als nicht autorisierter Benutzer und einmal als autorisierter Benutzer. Die beiden Typen unterscheiden sich jedoch nicht im Layout, sondern elementar nur in den Buttons und den damit verbundenen Funktionen. Wenn ein nicht-autorisierter Benutzer die Webanwendung zum ersten Mal betritt, gelangt er auf die Hauptseite, die auch Landing Page genannt wird. Ist der Nutzer jedoch angemeldet, hat er die Möglichkeit zum Dashboard zu gelangen oder sich auszuloggen.

Das Design der Hauptseite ist wie folgt gestaltet:

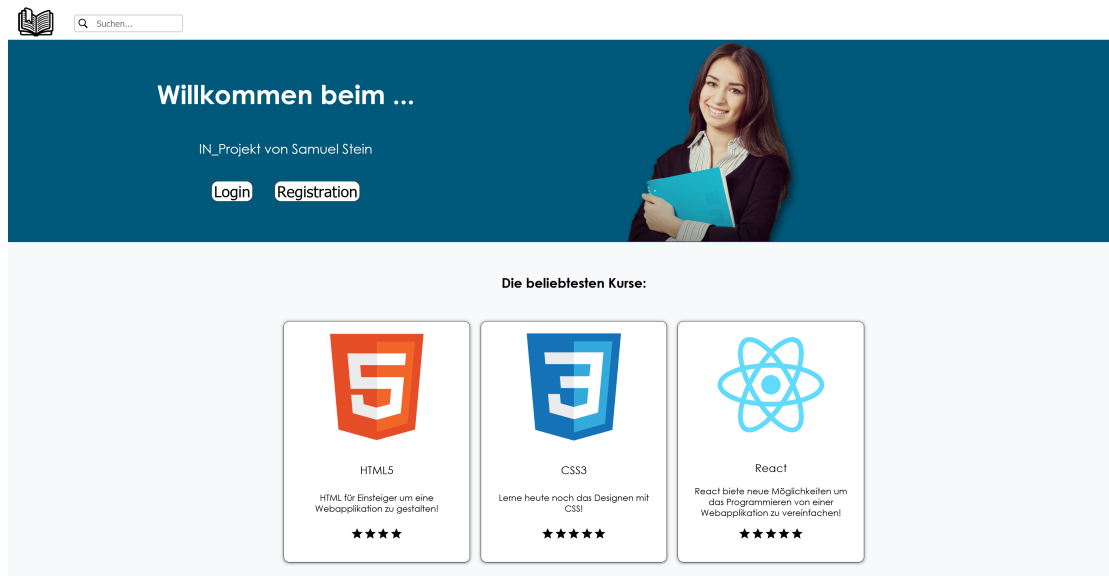


Abbildung 3.2.: Entwurf der Homepage

Login-/Registrierung

Die Anmelde- und Registrierungsseite sollte einfach sein. In der Mitte der Webseite sollte ein Feld für den Benutzernamen und das Passwort sein. Um die Seite dynamischer zu gestalten, werden während des Logins kleine Animationen angezeigt. Alternativ kann der Nutzer sich zur Registrierungsseite umleiten lassen. Nach der Registrierung wird der Benutzer zur Anmeldeseite und nach der Anmeldung und Autorisierung wird der Benutzer zum Dashboard weitergeleitet. Im Falle einer falschen Autorisierung mit falschen Anmeldedaten wird eine Fehlermeldung mit dem entsprechenden Fehler auf der Client-Seite angezeigt.

Die Anmeldeseite sieht wie folgt aus:

Dashboard

Das Dashboard bildet den Mittelpunkt für die zugelassenen Inhalte. Das Grundlayout ist ein zweiseitiges Layout und besteht aus einer Kopfleiste, einer Seitenleiste und dem Inhalt. Jeder Nutzer kann zu weiteren Unterseiten navigieren, ohne, dass das Grundlayout verändert wird. Die Unterseiten des Dashboards sind über die Sei-

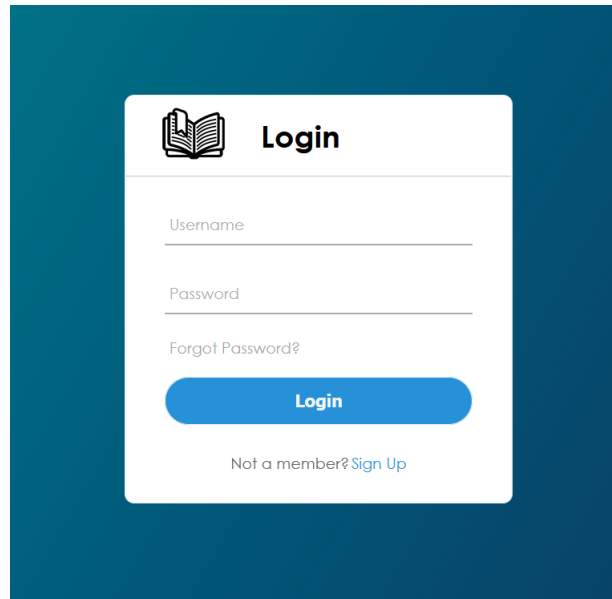
The image shows a login page design on a dark blue background. A white card contains a book icon and the title 'Login'. Below the title are input fields for 'Username' and 'Password', followed by a 'Forgot Password?' link. A blue 'Login' button is centered below these fields. At the bottom of the card, there is a link that says 'Not a member? Sign Up'.

Abbildung 3.3.: Entwurf der Loginpage

tenleiste zu erreichen, deren Funktionen im weiteren Verlauf der Ausarbeitung noch näher erläutert werden. In der mobilen Ansicht ändert sich das zweispaltige Layout jedoch in ein einspaltiges Layout. Hier sollte die Seitenleiste zu einem bekannten Burger-Menü werden, damit der Benutzer sie ein- und ausblenden kann.

3.6. Benutzergruppen

Auf der Lernplattform gibt es im Grunde drei Arten von Nutzern. Diese sind Schüler, Lehrer und Administrator und können sich alle sowohl anmelden als auch abmelden. Bei der Anmeldung wird die Berechtigung des Benutzers bestimmt. Je nach Berechtigung werden dem Benutzer in der Seitenleiste des Dashboards verschiedene Registerkarten angezeigt. Bei einer Registrierung bekommt der Nutzer die Studentenberechtigung zugeteilt.

Das Dashboard sieht wie folgt aus:



Abbildung 3.4.: Entwurf des Dashboards

Student

Der Student ist der Teil des Benutzers, der die Zielgruppe der gesamten Webanwendung ist. Nachdem der Student sich authentifiziert hat, wird er auf das Dashboard weitergeleitet. Dort ist es möglich, Module auszuwählen, die am Ende der Website ein Quiz enthalten, das absolviert werden kann. Dieses Quiz und die darin enthaltenen Fragen sind spezifisch für das jeweilige Modul.

Außerdem hat der Student eine Reiter namens Profil in der Seitenleiste, unter der er sein Passwort ändern oder sein Profil löschen kann. In der Seitenleiste befindet sich auch ein Button mit der Bezeichnung "Logout". Wenn der Schüler auf diesen Button drückt, wird er wieder zur Startseite zurückgeleitet.

Lehrer

Lehrer sollten in erster Linie in der Lage sein, Module und die dazugehörigen Quizze zu erstellen. Um ein Quiz zu erstellen kann der Lehrer im Reiter „Quiz erstellen“ mehrere Textfelder ausfüllen, welche aus einer Frage und vier Antwortmöglichkeiten besteht. Allerdings stehen dieser Benutzergruppe auch alle Funktionalitäten der Gruppe Schüler zur Verfügung.

Admin

Der Admin ist die höchste Benutzergruppe in Bezug auf den Benutzerberechtigung. Sie ist in der Lage, die Funktionen der beiden anderen Gruppen zu nutzen und hat im Grunde nur eine zusätzliche Funktion. Das ist die Verwaltung der einzelnen Nutzer. So kann ein normaler Student zu einem Lehrer ernannt werden und umgekehrt. Des Weiteren ist der Admin in der Lage Nutzer zu entfernen.

3.7. Funktionen

Auch wenn einige Funktionen der Webanwendung bereits unter dem Punkt Benutzergruppen erwähnt wurden, werden in diesem Abschnitt alle Hauptfunktionen ausführlicher erläutert.

Login und Registrierung

Das *Login- und Registrierungssystem* zur Authentisierung und zum Erstellen von Nutzern bilden hierbei eine der wichtigsten Funktionen für die Webanwendung. Die Requests und Responses des Registrierungssystems sind für den Benutzer im Frontend nur teilweise sichtbar. Grundsätzlich soll der Benutzer in der Lage sein, ein Konto mit Benutzernamen und Passwort anzulegen. Wenn der Benutzer bereits existiert, wird kein weiterer Benutzer angelegt. Darüber hinaus gibt es, wie in der Bibliothek YUP beschrieben, ein Schema, das die Anforderungen an die Registrierung bildet. So sollte der Benutzername aus mindestens drei und maximal 15 Buchstaben bestehen. Das Passwort des Benutzers sollte ebenfalls aus mindestens vier und maximal 20 Zeichen bestehen. Wenn die Registrierung erfolgreich ist, wird der Benutzer zunächst als Student angelegt.

Nachdem sich der Nutzer erfolgreich registriert hat, wird er auf die Anmeldeseite weitergeleitet. Wenn der Benutzer sich erfolgreich anmeldet, wird er zu seinem Dashboard weitergeleitet. Andernfalls erhält er Fehlermeldungen. Diese weisen gezielt auf den Fehler hin, der aufgetreten ist. Die Fehler können entweder ein nicht vorhandener Benutzer oder ein falsches Passwort sein.

Sobald der Benutzer erfolgreich eingeloggt ist, sieht er im Dashboard die für seine Benutzergruppe bestimmten Inhalte.

Module und Quizze

Der Hauptnutzen für die Nutzer besteht darin, dass es Module geben wird, welche Quizfragen enthalten. So wird jeder registrierte Nutzer in der Lage sein, Module im Dashboard zu bearbeiten, die jede Art von Themen enthalten können. Diese Module sollen von den Lehrern der Benutzergruppe erstellt werden. Darüber hinaus können die Lehrer Quizfragen zu den einzelnen Modulen hinzufügen. Dabei handelt es sich um Quizfragen, die dem Multiple-Choice-Format entsprechen. Die Studenten haben also vier Antwortmöglichkeiten, von denen eine richtig und die anderen falsch sind. Ob der Nutzer die richtige Antwort gewählt hat, sieht er am Ende an der Anzahl der richtigen Antworten im Vergleich zu der Anzahl der Quizfragen.

Punktesystem

Für jede richtig beantwortete Frage sollte der Schüler Punkte erhalten. Gehört der Schüler zu den zehn Nutzern mit den meisten Punkten, wird der Student in einer Rangliste angezeigt. Die Punktzahl der Quizfragen wird von den Lehrern und dem entsprechenden Schwierigkeitsgrad der Frage bestimmt.

Fortschritt der Module

Der Nutzer sollte in der Lage sein, den Fortschritt seiner Module zu sehen, in denen er Fragen abgeschlossen hat. Dieser Fortschritt wird auch im Dashboard des Nutzers angezeigt.

Profileinstellungen

Die Nutzer sollten einige Einstellungen an ihrem eigenen Profil vornehmen können. Dazu gehören das Ändern des Passworts und das Löschen des eigenen Kontos. Sobald das Profil gelöscht wird, wird der Kunde zur Homepage zurückgeleitet und das Profil wird aus der Datenbank gelöscht.

Modulübersicht

Auch nicht registrierte Nutzer sollen eine Übersicht über alle verfügbaren Module mit einer kurzen Beschreibung erhalten. Dies wird bereits auf der Startseite sichtbar sein.

3.8. Anwendungsfälle

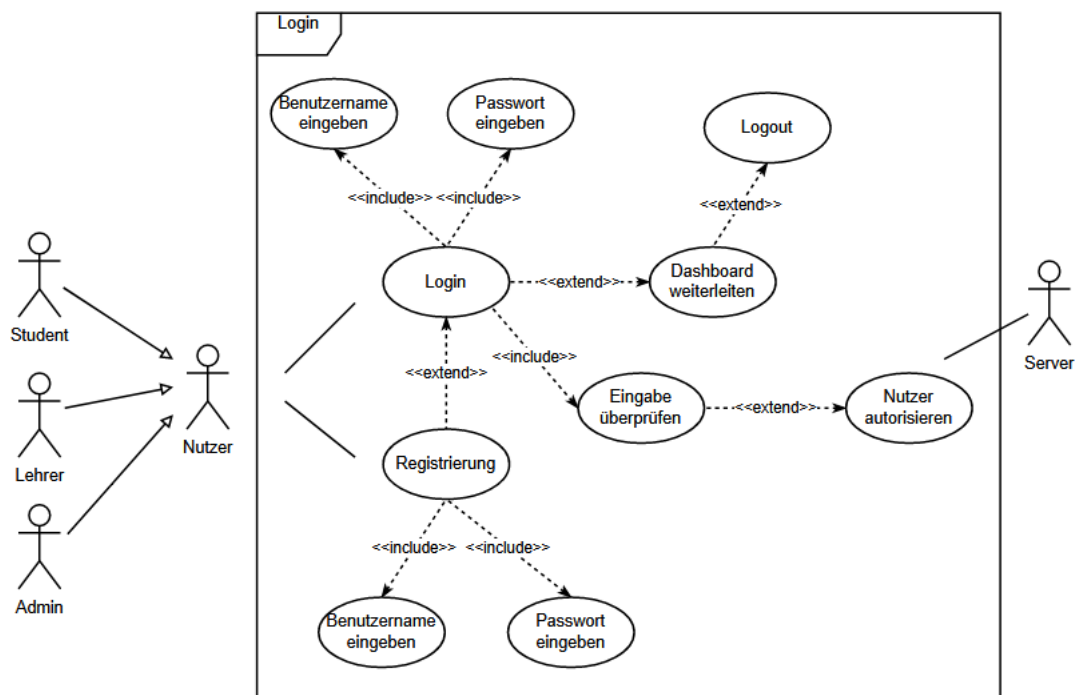


Abbildung 3.5.: Anwendungsfall 1

Element	Beispiel
Anwendungsfall	Login und Registrierung
Kurzbeschreibung	Ein Nutzer registriert sich und meldet sich an.
Akteure	Nutzer
Normaler Ablauf	1. Der Akteur gibt seinen Benutzernamen und ein Passwort zum Registrieren ein. 2. Der Akteur meldet sich mit seinen Benutzerdaten an. 3. Der Akteur wird an das Dashboard weitergeleitet.
Alternativer Ablauf	1 a) Der Akteur gibt entweder zu viele oder zu wenige Zeichen an und der Account wird nicht registriert. -> System gibt Fehlermeldung aus! 2 a) Der Akteur gibt falsche Benutzerdaten ein. -> System gibt Fehlermeldung aus!
Vorbedingung	Es gibt keine Vorbedingung.
Nachbedingung	Der Benutzer wird zum Dashboard weitergeleitet.

Abbildung 3.6.: Anwendungsfall 1 - Tabelle

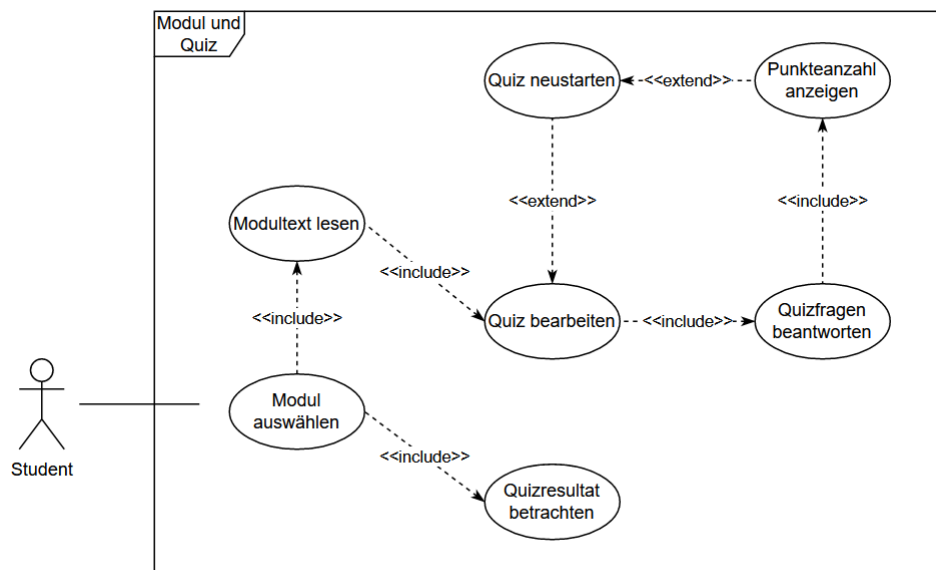


Abbildung 3.7.: Anwendungsfall 2

Element	Beispiel
Anwendungsfall	Modul und Quiz bearbeiten
Kurzbeschreibung	Ein Student bearbeitet ein Modul und das dazugehörige Quiz.
Akteure	Student
Normaler Ablauf	<ol style="list-style-type: none"> 1. Der Akteur wählt ein Modul aus, welches er bearbeiten möchte. 2. Der Student liest den Modultext durch. 3. Der Student bearbeitet das dazugehörige Quiz. 4. Der Akteur wählt eine Antwortmöglichkeit aus. 5. Das System zeigt die erreichte Punkteanzahl an.
Alternativer Ablauf	6. Der Student hat die Möglichkeit das Quiz neuzustarten
Vorbedingung	Der Nutzer muss angemeldet und im Dashboard sein.
Nachbedingung	Der Benutzer wird zum Dashboard weitergeleitet

Abbildung 3.8.: Anwendungsfall 2 - Tabelle

4. Implementierung der Webanwendung

In diesem Kapitel wird die konkrete Umsetzung des Lösungskonzepts, das in Kapitel 3 beschrieben wird.

4.1. Installation und Konfiguration

Für die Programmierung der Webanwendung wurde der Code-Editor Visual Studio Code (VSC) verwendet. Nach der Erstellung des Projekts sind einige Vorbereitungen zu treffen. Zunächst wird die Webanwendung in zwei Ordner unterteilt. Diese sind zum einen das Frontend und zum anderen das Backend. Das VSC-Terminal bietet hier die Verwendung von Linux-Befehlen an, so dass diese Ordner mit dem Befehl *mkdir* erstellt wurden.

Anschließend müssen die bereits erwähnten Bibliotheken mit dem Node Package Manager installiert werden. Die folgende Liste zeigt, welche Befehle in welchem Ordner verwendet werden müssen.

Frontend

```
1 npx create-react-app .
2 npm install axios
3 npm install react-router-dom
4 npm install formik
5 npm install yup
6 npm install @material-ui/core
7 npm install @material-ui/icons
```

Wie bereits im Kapitel der theoretischen Grundlagen erwähnt, sind *axios*, *react-router-dom*, *formik* und *yup* hilfreiche Bibliotheken. Der Befehl *npx create-react-app .* stellt die notwendigen Voraussetzungen für die erstellte React-Anwendung sicher und installiert sie im aktuellen Verzeichnis. Material-UI bietet eine große Anzahl von Icons für das Frontend-Design.

Backend

```
1 npm init
2 npm install mysql2
3 npm install express
4 npm install cors
5 npm install nodemon
6 npm install sequelize sequelize-cli
7 npx sequelize init
8 npm install bcrypt
9 npm install jsonwebtoken
```

Nach der Installation und Einbindung der Bibliotheken ist eine übersichtliche Anordnung aller Komponenten in der Webanwendung durch die folgende Verzeichnishierarchie realisiert.

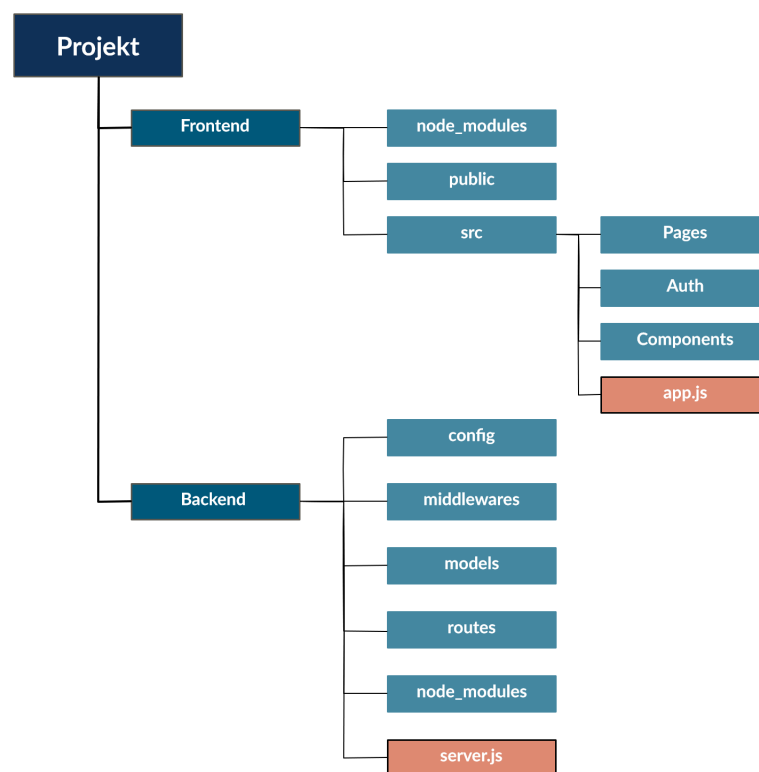


Abbildung 4.1.: Ordnerstruktur der Homepage

In den Ordnern namens `node_modules` befinden sich die Funktionen, die Kom-

munikation der Komponenten gewährleisten.

Frontend-Ordner

Im src-Ordner des Frontends befinden sich die eigentlichen Ordner und Funktionen, die den selbstgeschriebenen Code des Frontends ausmachen.

Im Ordner Pages befinden sich alle Webseiten der Webanwendung mit ihrer CSS-Datei.

Auth enthält die Funktion und deren Zustand zur Authentifizierung von Benutzern mit ihren wichtigen Informationen, die verarbeitet werden müssen.

Alle Komponenten, wie das Quiz und die Navigationsleisten, werden im Ordner Components abgelegt.

Die app.js ist sozusagen der Knotenpunkt aller Seiten und Unterseiten der Webanwendung, damit alle Seiten und deren URL miteinander kommunizieren können. Sie prüft auch den Status, ob der Benutzer eingeloggt ist oder nicht und zeigt darauf basierend die spezifische Homepage an.

Backend-Ordner

Im config-Ordner befindet sich eine config.json-Datei. In dieser werden die Anmelde-daten, sowie die IP-Adresse und die Sprache der Datenbank gespeichert, um sich mit ihr zu verbinden und die MySQL-Anweisungen im späteren Kurs erfolgreich zu nutzen.

Im Middlewares-Ordner werden Dienste gespeichert, die im Projekt verwendet werden, um Abfragen zum Status durchzuführen. Hier wird zum Beispiel geprüft, ob der Client ein accessToken hat.

Durch die Einbindung der Sequelize Bibliothek wurde der Ordner models erstellt. In diesem Ordner ist es möglich, Tabellen für die Datenbank im JSON-Format zu erstellen. So werden die Tabellen Modules, Quizzes, UserQuizScores und Users als JavaScript-Dateien erstellt.

Der Ordner Routes enthält alle URL-Routen und Anfragen in der Skriptsprache JavaScript. Hier werden drei Routen benötigt. Module, Quizzes und Users. Hier

werden die Ressourcen verarbeitet.

4.2. Implementierung

4.2.1. JSON Web Token zum Authentifizieren

Um sicherzustellen, dass ein Benutzer authentifiziert ist, überprüft dieses Projekt den LocalStorage im Webbrowser. Wenn der Benutzer authentifiziert ist, dann hat er einen accessToken, das JSON Web Token im LocalStorage. In diesem Token sind der Benutzername und die zugehörige UserID gespeichert.

Zur Überprüfung des Tokens wird der JavaScript-Befehl *localStorage.getItem(token)* verwendet. Wenn kein Token vorhanden ist, gibt diese Funktion *null* zurück.

Die Bibliothek von jsonwebtoken bietet sowohl eine Funktion namens sign, als auch eine namens verify. Um zu verifizieren, ob ein Webtoken existiert wird folgende Funktion als Middleware verwendet.

```
1 const {verify} = require('jsonwebtoken');
2 const validateToken = (req, res, next) => {
3   const accessToken = req.header("accessToken");
4   if (!accessToken) return res.json({error: "Benutzer ist nicht
5     eingeloggt!"});
6   try {
7     const validToken = verify(accessToken, "importantsecret");
8     req.user = validToken;
9     if (validToken) {
10       return next();
11     }
12   } catch (err) {
13     return res.json({error: err});
14   }
15 }
16 module.exports = {validateToken};
```

Quelltext 4.1: AuthMiddleware.js

4.2.2. Responsive Designs

Da das Responsive Design mit einer der wichtigsten Bestandteile der Ausarbeitung ist, wird dies im Anschluss aufgezeigt und erläutert. Hierbei werden die Vorteile von CSS Grid verwendet. Grundsätzlich können verschiedene Ansichten bei verschiedener Auflösung in der CSS datei mit dem Befehl: *@media screen and (min-width: 736px)* angepasst werden. Ab einer Mindestweite des Gerätes von 736 Pixel wird folgendes CSS Grid-layout beim Dashboard verwendet:

```
1  .page__grid {  
2      display: grid;  
3      grid-template-columns: 180px 1fr;  
4      grid-template-rows: 70px 30px 1fr 0px;  
5      grid-template-areas:  
6          "page__header page__header"  
7          "page__title page__title"  
8          "page__sidebar page__content"  
9          "page__sidebar page__content"  
10         "page__footer page__footer";  
11 }
```

Quelltext 4.2: CSS Grid über 736 Pixel

Bei kleineren Geräten mit einer Weite bis maximal 736 Pixel hingegen wird folgendes Verwendet:

```
1  .page__grid {  
2      display: grid;  
3      background-color: white;  
4      grid-template-columns: 1fr 1fr;  
5      grid-template-rows: 70px 0px 60px 1fr 1fr;  
6      grid-template-areas:  
7          "page__header page__header"  
8          "page__title page__title"  
9          "page__sidebar page__sidebar"  
10         "page__content page__content"  
11         "page__footer page__footer";  
12 }
```

Quelltext 4.3: CSS Grid unter 736 Pixel

Die Hauptunterschiede dabei liegen in der Zeile 8 und der Zeile 9 in den Abbildungen 4.2 und 4.3. Diese bestimmen die Anordnung der Rasterflächen. Aber auch innerhalb der einzelnen Grid-Template-Areas werden Anpassungen vorgenommen.

Aber auch innerhalb der einzelnen Grid-Template-Areas werden Anpassungen vorgenommen. So werden zum Beispiel mit Flexbox Elemente vertikal dargestellt, die vorher horizontal angezeigt wurden. Dies wird in den folgenden Unterpunkten im Vergleich zum Entwurf deutlich.

Während das Design für den Login und die Registrierung bereits für mobile Geräte und die damit verbundenen kleineren Auflösungen gleich bleibt, müssen die Hauptseite und das Dashboard für das responsive Design angepasst werden.

Hauptseite

In der mobilen Ansicht der Hauptseite ist das geteilte Banner untereinander angeordnet. Aber auch die einzelnen Module der Übersicht sind untereinander positioniert.

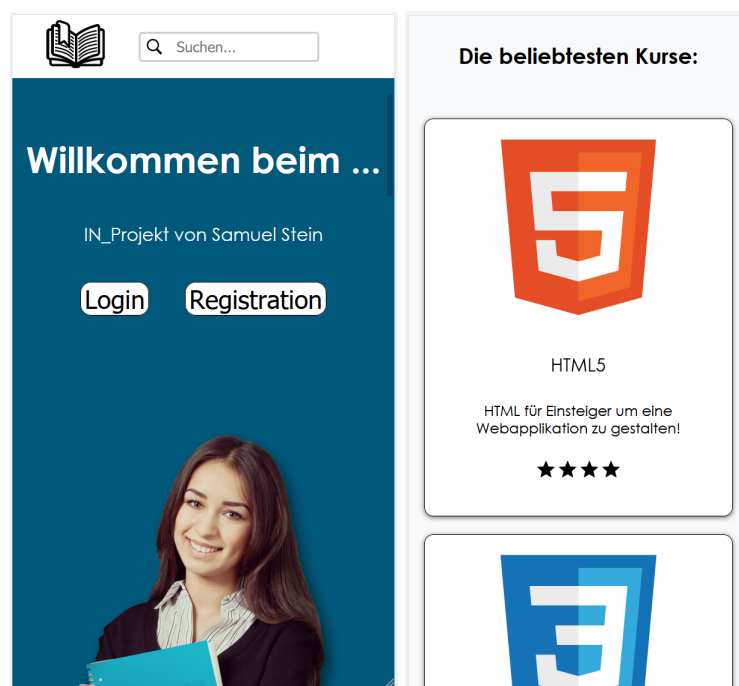


Abbildung 4.2.: Responsive Ansicht der Homepage

Dashboard

Mit dem Burgermenu wird die Seitenleiste zur Navigation ein und ausgeklappt, damit der Nutzer mehr Platz für die wesentliche Unterseite auf seinem Mobilgerät hat.

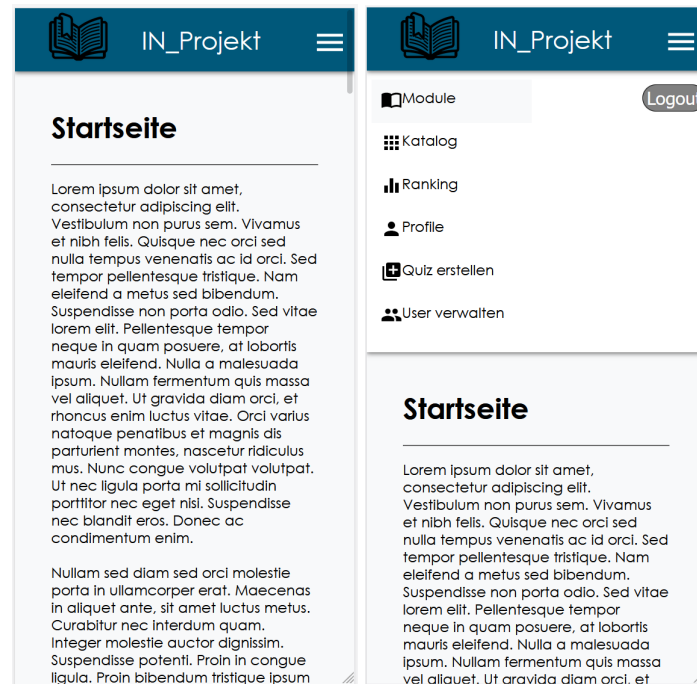


Abbildung 4.3.: Responsive Ansicht des Dashboards

5. Zusammenfassung und Ausblick

5.1. Zusammenfassung

Um eine Lernplattform als responsive Webanwendung zu erstellen, ist eine Menge Grundwissen über das Backend und das Frontend erforderlich. Für das Backend muss das Wissen über Datenbanken und die Kommunikation von Client zu Server vertraut sein. Für das Frontend hingegen müssen HTML, CSS und JavaScript für den Entwickler kenntnisreich sein, um einfache, responsive oder komplexe Designs und Animationen darstellen zu können. Wenn die Webanwendung zusätzlich noch modern und benutzerfreundlich gestaltet sein soll, dann müssen Verhaltensweisen von Menschen berücksichtigt werden.

5.2. Erreichte Ergebnisse

Von den gewünschten Zielen wurden das meiste erreicht. Das, was nicht umgesetzt wurde ist, dass die Berechtigungsgruppe Lehrer Module über das Dashboard erstellen kann. Dies wurde aus dem Grund nicht umgesetzt, da die Gestaltung des Modultextes nicht so problemlos möglich war. Die Lösung hierfür war, dass der Text des Modules im Frontend gespeichert wird, um anschließend mit CSS den Stil des Textes anzupassen. Ansonsten wurde das responsive Design und weitere Funktionen der einzelnen Nutzergruppen umgesetzt.

5.3. Denkanstoß

Es wird zunehmend darüber diskutiert, dass Lernen individuell erfolgen muss. Lernplattformen sind dafür sehr gut geeignet, da die Nutzer nicht physisch anwesend sein müssen und ihr Lerntempo selbst bestimmen können. Lernplattformen eignen sich ebenfalls hervorragend für das Selbststudium in Verbindung mit bestimmten Aufgaben. Aber auch das Design und die Funktionen sollten im Vordergrund stehen, um die Attraktivität des E-Learnings zu erhöhen.

Literatur

- [1] Ralf Adams. *SQL: Der Grundkurs für Ausbildung und Praxis : mit Beispielen in MySQL/MariaDB, PostgreSQL und T-SQL*. 3., aktualisierte Auflage. München: Hanser, 2020. ISBN: 978-3-446-46110-9.
- [2] Adobe Color. *Color Wheel*. URL: <https://color.adobe.com/de/create/color-wheel>.
- [3] Joe Attardi. *Modern CSS*. Berkeley, CA: Apress, 2020. ISBN: 978-1-4842-6293-1. DOI: [10.1007/978-1-4842-6294-8](https://doi.org/10.1007/978-1-4842-6294-8).
- [4] Auth0. *Introduction to JSON Web Tokens*. URL: <https://jwt.io/introduction>.
- [5] Azat Mardan. *Express.js Guide*. 2014. ISBN: 978-1494269272.
- [6] David Herron. *Node.js Web Development : Server-side Development with Node 10 Made Easy, 4th Edition*. 4, 2018. ISBN: 9781788626859.
- [7] N. Di Ferrante u. a. „Ehlers-Danlos type V (X-linked form): a lysyl oxidase deficiency“. In: *Birth defects original article series* 11.6 (1975), S. 31–37. ISSN: 0547-6844.
- [8] Docker. *Docker Overview*. URL: <https://docs.docker.com/get-started/overview/>.
- [9] Fernando Doglio. *REST API Development with Node.js*. Berkeley, CA: Apress, 2018. ISBN: 978-1-4842-3714-4. DOI: [10.1007/978-1-4842-3715-1](https://doi.org/10.1007/978-1-4842-3715-1).
- [10] Dr. Angela Yu. *The Complete 2021 Web Development Bootcamp*. 2021. URL: <https://www.udemy.com/course/the-complete-web-development-bootcamp/learn/lecture/19655710?start=45#questions>.
- [11] Express. *Express- Webanwendungen*. URL: <https://expressjs.com/de/>.
- [12] Cory Gackenheimer. *Introduction to React*. The expert's voice in web development. Berkeley, CA: Apress, 2015. ISBN: 978-1-4842-1245-5. URL: <http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&AN=1064176>.
- [13] IBM Cloud Education. *REST APIs*. 2021. URL: https://www.ibm.com/cloud/learn/rest-apis?utm_medium=OSocial&utm_source=Youtube&utm_content=000023UA&utm_term=10010608&utm_id=YTDDescription-101-What-is-a-REST-API-LH-REST-APIs-Guide&cm_mmc=OSocial_Youtube-_-Cloud+and+Data+Platform_SFT+Cloud+Platform+Digital-_-WW_WW-_-YTDDescription-101-What-is-a-REST-API-LH-REST-APIs-Guide&cm_mmca1=000023UA&cm_mmca2=10010608.

- [14] Sujit Kumar Basak, Marguerite Wotto und Paul Bélanger. „E-learning, M-learning and D-learning: Conceptual definition and comparative analysis“. In: *E-Learning and Digital Media* 15.4 (2018), S. 191–216. ISSN: 2042-7530. DOI: [10.1177/2042753018785180](https://doi.org/10.1177/2042753018785180).
- [15] MDN contributors. *Express Tutorial Part 4: Routes and controllers*. 2021. URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/routes.
- [16] MDN contributors. *What is JavaScript?* 18. Juni 2021. URL: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript.
- [17] Meinald Thielsch, Iris Blotenberg, Rafael Jaron. *User evaluation of websites: From first impression to recommendation*. 2014. URL: https://www.researchgate.net/publication/259149186_User_evaluation_of_websites_From_first_impression_to_recommendation.
- [18] Helen Epps Naz Kaya. *Relationship between color and emotion: a study of college students*. 2004. URL: http://irtel.uni-mannheim.de/lehre/expra/artikel/Kaya_Epps_2004b.pdf.
- [19] Otto Geißler. *Was ist ein Frontend und ein Backend?* 2018. URL: <https://www.datacenter-insider.de/was-ist-ein-frontend-und-ein-backend-a-714429/>.
- [20] React Router. *React - Primary Components*. URL: <https://reactrouter.com/web/guides/primary-components>.
- [21] Sequelize. *Sequelize Documentation v6*. URL: <https://sequelize.org/master/>.
- [22] VirtusD Virtuelle Universität Deutschland -. *VirtusD Virtuelle Universität Deutschland - E-Learning für eine bessere Bildung an den Hochschulen*. 2007.
- [23] W3schools. *HTML <div> Tag*. URL: https://www.w3schools.com/tags/tag_div.asp.
- [24] R. F. Wachter, G. P. Briggs und C. E. Pedersen. „Precipitation of phase I antigen of Coxiella burnetii by sodium sulfite“. In: *Acta virologica* 19.6 (1975), S. 500. ISSN: 0001-723X.
- [25] Frank Zammetti. *Modern Full-Stack Development*. Berkeley, CA: Apress, 2020. ISBN: 978-1-4842-5737-1. DOI: [10.1007/978-1-4842-5738-8](https://doi.org/10.1007/978-1-4842-5738-8).

A. Anhang A

B. Anhang B