

Studiengang

**Wirtschaftsinformatik  
Schwerpunkt: IT Engineer**

PO Version 2020

**Bachelorarbeit**

**Entwicklung einer sozialen Netzwerkplattform für  
Studenten: Konzeption, Implementierung und  
Evaluation „Full-Stack-Anwendung mit Laravel und  
React.“**

Development of a social networking platform for students: Design,  
implementation and evaluation “Full-stack application with Laravel and  
React.”

vorgelegt von

**Billy Max Tsane Tsafack**

09. Dezember 2024

Betreuung: Prof. Dipl.-Ing. Klaus Knopper  
Zweitkorrektor: Dipl.-Handelslehrer Andreas Heß

## **Abstract**

In an increasingly digitalised world, exchanges between students are gaining in importance. The StudentCommunity WebApp, a platform designed specifically for the needs of students, is the subject of this study. The aim is to develop a new type of solution that enables improved networking and cooperation.

The app's functions include customised group communication, a calendar of events, a job exchange and numerous other tools that simplify students' everyday lives. Modern technologies such as Laravel in the backend and React in the frontend are used in this context. The development process is documented in this thesis, from conception to design and implementation.

This thesis shows how the StudentCommunity WebApp fills an important gap by providing security, usability and targeted support for academic and social interaction by comparing it to existing platforms and identifying its shortcomings.

## **Kurzfassung**

In einer zunehmend digitalisierten Welt gewinnt der Austausch zwischen Studierenden an Bedeutung. Die StudentCommunity-WebApp, eine Plattform, die speziell für die Bedürfnisse von Studierenden konzipiert wurde, ist das Thema dieser Studie. Das Ziel ist die Entwicklung einer neuartigen Lösung, die eine verbesserte Vernetzung und Kooperation ermöglicht.

Zu den Funktionen der App gehören eine individuell angepasste Gruppenkommunikation, ein Veranstaltungskalender, eine Jobbörse und zahlreiche andere Hilfsmittel, die den Alltag der Studierenden vereinfachen. In diesem Zusammenhang werden moderne Technologien wie Laravel im Backend und React im Frontend verwendet. Der Entwicklungsprozess wird in dieser Arbeit festgehalten, von der Konzeption über das Design bis zur Umsetzung.

Diese Arbeit zeigt, wie die StudentCommunity-WebApp eine wichtige Lücke füllt, indem sie Sicherheit, Benutzerfreundlichkeit und eine gezielte Unterstützung der akademischen und sozialen Interaktion bietet, indem sie mit bestehenden Plattformen verglichen und ihre Mängel identifiziert werden.

## **Vorwort**

Aus der Feststellung, dass Studierende häufig auf allgemeinen Plattformen kommunizieren, die nur teilweise auf ihre Bedürfnisse zugeschnitten sind, entstand die Idee, die StudentCommunity-WebApp zu entwickeln. Es wurde offensichtlich, dass eine Lösung fehlt, die akademische und soziale Anforderungen gleichermaßen berücksichtigt.

Diese Arbeit zielt darauf ab, nicht nur eine technische Lösung zu präsentieren, sondern auch zu verdeutlichen, wie digitale Hilfsmittel zur Förderung der wissenschaftlichen Kooperation und zur Stärkung von sozialen Netzwerken in Studiengemeinschaften beitragen können.

Ich wünsche mir, dass die vorgestellten Ideen und Resultate nicht nur als technisches Vorhaben betrachtet werden, sondern auch als Beitrag zur Förderung der digitalen Kommunikation unter den Studierenden.

## Danksagung

Mit dem Abschluss dieser Arbeit möchte ich die Gelegenheit nutzen, all Jenen zu danken, die mich auf meinem Weg begleitet und unterstützt haben. Mein besonderer Dank gilt meinem Betreuer, **Prof. Dipl.-Ing. Klaus Knopper**, für seine fachkundige Unterstützung, seine Geduld und die wertvollen Anregungen, die wesentlich zum Gelingen dieser Arbeit beigetragen haben. Seine Fachkenntnisse und sein konstruktives Feedback haben mir geholfen, zahlreiche Herausforderungen zu bewältigen und mein Wissen zu vertiefen.

Diese Arbeit war nicht frei von Schwierigkeiten. Insbesondere die Anforderungen an ein effektives Zeitmanagement sowie die intensive Recherche- und Korrekturphase haben mich immer wieder vor große Herausforderungen gestellt. Doch genau diese Herausforderungen haben meinen Lernprozess und meine Entwicklung während dieser Zeit beeinflusst.

Mein aufrichtiger Dank gilt auch meiner Familie und meinen Freunden. Insbesondere danke ich meiner Mutter **Frau Tsafack Bibiane** und meinen **Geschwistern**, die mich mit unerschütterlicher moralischer Unterstützung begleitet haben. Der Glaube an mich von Ihnen hat mich dazu motiviert, auch in herausfordernden Situationen nicht aufzugeben.

Diese Arbeit widme ich im Gedenken an meinen verstorbenen Vater, **Herrn Tsafack Pierre Marie**. Sein Vorbild, seine Werte und seine Ermutigung haben mich stets begleitet und inspiriert, mein Bestes zu geben. Seine Präsenz in meinen Gedanken war ein Anker, der mich auch in den herausforderndsten Zeiten gestärkt hat.

Ohne all diese Menschen wäre der Abschluss dieses Projekts nicht möglich gewesen, und ich bin zutiefst dankbar für Ihre Unterstützung und Ihren Beitrag zu diesem Erfolg.

# Inhaltsverzeichnis

<b>Abstract.....</b>	<b>II</b>
<b>Kurzfassung.....</b>	<b>III</b>
<b>Vorwort.....</b>	<b>IV</b>
<b>Danksagung.....</b>	<b>V</b>
<b>Inhaltsverzeichnis .....</b>	<b>VI</b>
<b>Abbildungsverzeichnis .....</b>	<b>VIII</b>
<b>Tabellenverzeichnis .....</b>	<b>IX</b>
<b>Abkürzungsverzeichnis .....</b>	<b>X</b>
<b>1. Einleitung.....</b>	<b>1</b>
1.1. Zielsetzung der Arbeit.....	2
1.2. Aufbau der Arbeit.....	2
1.3. Vorgehensweis.....	3
<b>2. Theoretische Grundlagen und Stand der Forschung .....</b>	<b>5</b>
2.1. Soziale Netzwerke: Definition und Bedeutung.....	5
2.2. Technologische Grundlagen: Laravel und React.....	6
2.3. User Experience (UX) und Design-Prinzipien.....	8
<b>3. Anforderungsanalyse .....</b>	<b>11</b>
3.1. Funktionale Anforderungen.....	11
3.2. Nicht-funktionale Anforderungen.....	14
<b>4. Systemarchitektur (High-Level-Architektur) .....</b>	<b>17</b>
4.1. Datenbankdesign (Grundlegende Tabellen).....	20
4.2. User Interface Design (Hauptseiten und Navigation).....	25
<b>5. Implementierung .....</b>	<b>33</b>
5.1. Entwicklungsumgebung und Werkzeuge.....	34
5.2. Backend-Entwicklung mit Laravel.....	38
5.2.1. API-Design und Implementierung.....	40
5.2.2. Authentifizierung und Autorisierung.....	47
5.3. Frontend-Entwicklung mit React .....	50
5.3.1. Komponentenstruktur .....	50
5.3.2. Integration mit dem Backend.....	55
5.4. Wichtige Funktionalitäten .....	59
5.4.1. Benutzerregistrierung und Login .....	59
5.4.2. Profilerstellung und -verwaltung.....	63
5.4.3. Chat Funktionalitäten.....	67

<b>6.</b>	<b>Test und Evaluation .....</b>	<b>70</b>
6.1.	<i>Feature- Tests.....</i>	<i>70</i>
6.2.	<i>Usability Tests.....</i>	<i>75</i>
<b>7.</b>	<b>Fazit und Ausblick .....</b>	<b>78</b>
7.1.	<i>Zusammenfassung der Ergebnisse .....</i>	<i>78</i>
7.2.	<i>Persönliches Fazit .....</i>	<i>78</i>
7.3.	<i>Zukünftige Arbeiten und Erweiterungsmöglichkeiten.....</i>	<i>79</i>
	<b>Literaturverzeichnis .....</b>	<b>XI</b>
	<b>Anhang .....</b>	<b>XIV</b>
1.	<b>Gitlab Ticketübersichten .....</b>	<b>XIV</b>
2.	<b>Swagger Dokumentation .....</b>	<b>XV</b>
	<b>Eidesstattliche Erklärung .....</b>	<b>XVII</b>

## Abbildungsverzeichnis

Abb. 1: Wasserfallmodell .....	4
Abb. 2: Funktionale Anforderungen (eigene Erstellung) .....	12
Abb. 3: Nicht-funktionale Anforderungen (eigene Erstellung) .....	15
Abb. 4: StudentCommunity System Architektur (eigene Erstellung) .....	18
Abb. 5: StudentCommunity Datenbank Struktur (eigene Erstellung) .....	21
Abb. 6: Register Page (Screenshot).....	26
Abb. 7: Login Page (Screenshot).....	27
Abb. 8: Chat Page (Screenshot) .....	28
Abb. 9: Event Page (Screenshot) .....	28
Abb. 10: JobPage (Screenshot).....	29
Abb. 11: Profile Settings Page (Screenshot) .....	30
Abb. 12: Sidebar (Screenshot).....	31
Abb. 13: Logo (Screenshot).....	31
Abb. 14: docker-compose.yml Datei (Screenshot) .....	35
Abb. 15: Dockerfile Datei (Screenshot) .....	36
Abb. 16: Klassendiagramm der StudentCommunityWebApp (eigene Erstellung) .....	39
Abb. 17: Verbindung zur Datenbank (Screenshot) .....	41
Abb. 18: User Model (Screenshot) .....	42
Abb. 19: User Migration (Screenshot).....	43
Abb. 20: UserTabelle in der Datenbank (Screenshot).....	44
Abb. 21: UserController (Screenshot) .....	45
Abb. 22: Route Login API (Screenshot).....	46
Abb. 23: Middleware der Login Route (Screenshot).....	46
Abb. 24: Guards (Screenshot)      Abb. 25: ProvidersLaravel (Screenshot).....	47
Abb. 26: Login Methode (Screenshot) .....	48
Abb. 27: Komponentenstruktur (Screenshot).....	51
Abb. 28: ApplicationLogo Komponente (Screenshot).....	52
Abb. 29: AuthenticatedLayout (Screenshot) .....	53
Abb. 30: Einsetzung des GuestLayout in der Login Seite.....	54
Abb. 31: Sanctum Datei Konfiguration (Screenshot) .....	58
Abb. 32: Validierungslogik für die Registrierung (Screenshot).....	60
Abb. 33: UserRole Logik (Screenshot) .....	60
Abb. 34: Login Logik (Screenshot) .....	61
Abb. 35: ProfileController (Screenshot) .....	64
Abb. 36: Edit Datei für die Profile Seite (Screenshot).....	66
Abb. 37: WebSockets Connections.....	67
Abb. 38: WebSocket Server .....	68
Abb. 39: WebSocket Verbindung.....	69
Abb. 40: phpunit.xml Datei (Screenshot).....	71
Abb. 41: Registration Test Datei (Screenshot) .....	73
Abb. 42: Feature Test 1 (Screenshot).....	74
Abb. 43: Feature Test 2 (Screenshot).....	74
Abb. 44: Taskübersichten (Screenshot) .....	XIV
Abb. 45: Swagger Dokumentierung (Screenshot).....	XV



## Tabellenverzeichnis

Tabelle 1: Dokumentation der Nutzung von KI-Tools .....	XVI
---	-----

## Abkürzungsverzeichnis

API.....	<i>Application Programming Interface</i>
CORS.....	<i>Cross-Origin Resource Sharing</i>
DOM.....	<i>Document Object Model</i>
ERM .....	<i>Entity Relationship Model</i>
JSON.....	<i>JavaScript Object Notation</i>
JWT .....	<i>JSON Web Token</i>
MVC .....	<i>Model-View-Controllers</i>
NPM .....	<i>Node Package Manager</i>
ORM.....	<i>objektrelationale Mapping</i>
SPA .....	<i>Single-Page Application</i>
SQL .....	<i>Structured Query Language</i>
SSR .....	<i>Server-Side Rendering</i>
TCP .....	<i>Transmission Control Protocol</i>
UI .....	<i>User Interface</i>
UX.....	<i>User Experience</i>
XSS .....	<i>Cross-Site Scripting</i>

## 1. Einleitung

Heutzutage spielt die Kommunikation zwischen Studenten eine entscheidende Rolle für den studentischen Erfolg und die persönliche Entwicklung. Mit der zunehmenden Verbreitung digitaler Technologien und sozialer Medien hat sich die Art und Weise, wie Studenten miteinander interagieren, erheblich verändert. Die digitale Transformation hat die Kommunikation zwischen Studierenden und Hochschulen grundlegend verändert. Soziale Medien und Online-Plattformen spielen eine zunehmend wichtige Rolle in der Hochschulkommunikation, wobei einige Institutionen diese Kanäle intensiv nutzen, während andere zurückhaltender sind (Metag & Schäfer, 2017).

Diese Entwicklung bietet zahlreiche Vorteile, wie die Möglichkeit des schnellen und unkomplizierten Austauschs von Informationen, die Vernetzung über geografische Grenzen hinweg und die Förderung der Zusammenarbeit in virtuellen Lernumgebungen. In diesem Kontext erscheint die Idee, eine dezidierte Plattform zu schaffen, die speziell auf die Bedürfnisse von Studenten zugeschnitten ist und deren Interaktion durch Nachrichten, Gruppen oder Posts und Kommentare erleichtert, als äußerst relevant und zeitgemäß.

Die StudentCommunity-WebApp unterscheidet sich von herkömmlichen Plattformen, weil sie gezielt auf die studentischen und sozialen Bedürfnisse der Studierenden abgestimmt ist. Ihr Angebot umfasst integrierte Funktionen wie eine Jobbörse, einen Veranstaltungskalender und Kooperationstools sowie eine klar strukturierte Gruppenkommunikation für Kurse und Projekte. Im Unterschied zu herkömmlichen Plattformen wie WhatsApp oder Facebook stehen Datenschutz, werbefreier Betrieb und die Überwachung personenbezogener Daten im Mittelpunkt. Außerdem unterstützt die Anwendung den Austausch innerhalb der Hochschule oder Studiengruppe und trägt somit zur Entstehung eines starken Community-Gefühls bei. Ihre benutzerfreundliche und anpassbare Benutzeroberfläche stellt eine wichtige Lösung dar, die es den Studierenden ermöglicht, sich effizient zu organisieren und miteinander zu interagieren.

## **1.1. Zielsetzung der Arbeit**

Das Ziel dieser Bachelorarbeit ist es, eine Plattform für soziale Netzwerke für Studierende zu entwickeln. Diese soll eine sichere und benutzerfreundliche Umgebung für Kommunikation, Kooperation und Austausch von Informationen schaffen. Die Plattform zielt darauf ab, Studierenden den Austausch über Nachrichten, Gruppen, Beiträge und Kommentare zu ermöglichen. Dabei werden moderne Webtechnologien wie Laravel im Backend und React im Frontend verwendet. Ziel der Arbeit ist es, durch die Implementierung und Evaluation der Plattform zu demonstrieren, wie digitale Technologien die studentische Interaktion und Zusammenarbeit verbessern können. Gleichzeitig wird untersucht, wie Laravel und React effektiv zusammenarbeiten, um eine skalierbare und performante Plattform zu schaffen. Darüber hinaus werden sowohl funktionale als auch nicht-funktionale Anforderungen wie Benutzerfreundlichkeit, Sicherheit und Skalierbarkeit berücksichtigt und in die Entwicklung integriert.

## **1.2. Aufbau der Arbeit**

Die Arbeit beginnt mit einer Einführung in den theoretischen Hintergrund und der Analyse der Anforderungen, die an eine soziale Netzwerkplattform im Hochschulkontext gestellt werden. Dabei werden sowohl die funktionalen als auch die nicht-funktionalen Anforderungen definiert, die für die Entwicklung einer solchen Plattform relevant sind.

Anschließend werden die technische Konzeption und Implementierung der Plattform beschrieben. Dies umfasst eine detaillierte Darstellung der eingesetzten Technologien, insbesondere Laravel und React, sowie die Art und Weise, wie diese Technologien zur Erfüllung der Plattformanforderungen beitragen. Der Softwareentwurf und die Softwarearchitektur werden dabei ebenfalls berücksichtigt, um sicherzustellen, dass die Architektur der Plattform skalierbar, wartbar und effizient ist.

Daraufhin folgt die Evaluation der entwickelten Plattform, in der die Benutzerfreundlichkeit, Sicherheit, Performance und Skalierbarkeit getestet und bewertet werden. Der Prozess der Evaluation basiert auf Nutzertests und Messungen der Systemleistung.

Ein zentraler Aspekt der Arbeit ist auch das User Interface Design (UI), dass die Benutzerfreundlichkeit der Anwendung maßgeblich beeinflusst. Für das Design der Benutzeroberfläche wurde Figma verwendet, um ein ansprechendes und funktionales Layout zu erstellen. Weitere Details zum UI-Design, insbesondere die Hauptseiten und die Navigation, werden im Abschnitt **4.2. User Interface Design (Hauptseiten und Navigation)** erläutert. Dieser Teil wird durch eigene Screenshots aus der Figma Plattform unterstützt, um die Designentscheidungen anschaulich zu erklären.

Zum Abschluss der Arbeit werden die zentralen Ergebnisse zusammengefasst, ein Fazit gezogen und ein Ausblick auf mögliche Weiterentwicklungen der Plattform gegeben.

### **1.3. Vorgehensweis**

Um sicherzustellen, dass ein Softwareentwicklungsprojekt strukturiert und erfolgreich abläuft, werden die wesentlichen Prozesse der Softwareentwicklung berücksichtigt: Planung, Analyse, Entwurf, Programmierung und Tests. Eine systematische und zielgerichtete Entwicklung basiert auf diesen Phasen.

Die Phasen sind im Wasserfallmodell festgelegt. Für den Projekterfolg werden im nächsten Abschnitt das Wasserfallmodell und seine Phasen erläutert.

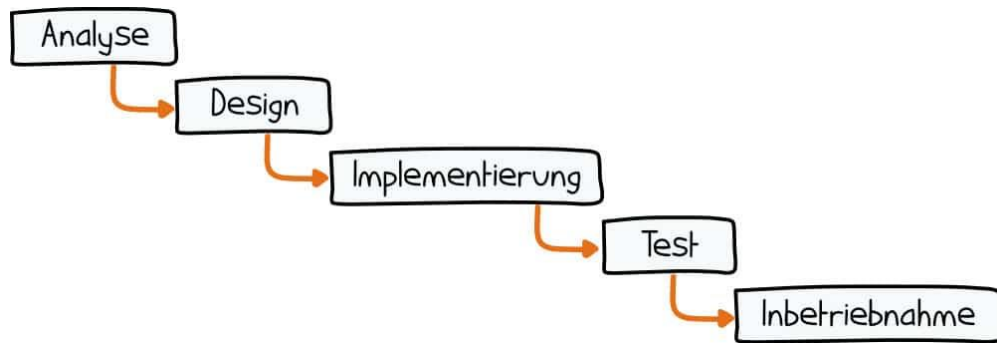


Abb. 1: Wasserfallmodell

In der Software-Entwicklung besteht das Wasserfallmodell aus fünf gängigen Phasen, die eine eindeutig strukturierte Herangehensweise bieten. Zuerst werden die Projektziele und Anforderungen in der Anforderungsanalyse genau festgelegt, um die Grundlage für den Erfolg zu schaffen. Diese Erfordernisse werden während der Entwurfsphase in technische Spezifikationen umgesetzt, die als Grundlage für die Realisierung dienen. Anschließend wird die Umsetzung durchgeführt, bei der die geplanten Anforderungen umgesetzt und ein funktionierendes Produkt entwickelt wird. Es wird während der Testphase geprüft, ob das erstellte Produkt den vorgegebenen Anforderungen gerecht wird. Zum Schluss werden die Inbetriebnahme und Wartung durchgeführt. Dabei wird das Produkt in einer effizienten Umgebung verwendet, es wird fortlaufend verbessert und Fehler werden behoben.<sup>1</sup>

---

<sup>1</sup> Andrea. (2023, 17. April). Das Wasserfallmodell einfach erklärt. Projekte Leicht Gemacht. <https://projekte-leicht-gemacht.de/blog/projektmanagement/klassisch/wasserfallmodell/>

## **2. Theoretische Grundlagen und Stand der Forschung**

Dieses Kapitel beschäftigt sich mit den theoretischen und technologischen Prinzipien der Arbeit, die bei der Erstellung der Plattform Student-Community entscheidend sind. Es wird erklärt, wie sich soziale Netzwerke verhalten, welche Technologien in dem Projekt verwendet werden und welche Grundsätze der Benutzererfahrung (User Experience, UX) in diesem Zusammenhang wichtig sind. Diese theoretischen Konzepte bilden die Grundlage für die Planung, Umsetzung und Bewertung der Plattform.

### **2.1. Soziale Netzwerke: Definition und Bedeutung**

Ein soziales Netzwerk ist eine Form digitaler Beziehungen, die es Personen erlaubt, sich über das Internet verbunden zu haben. Dies sind Plattformen, die auf Web-2.0-Anwendungen technischer Art beruhen. Benutzer erstellen ihre individuellen Profile, um Informationen über sich auszutauschen, wie zum Beispiel ihre Hobbys, Interessen, ihren Beruf oder ihren Wohnort. Diese Netzwerke bieten nicht nur die Möglichkeit, Freundschaften oder geschäftliche Beziehungen aufzubauen, sondern auch den Austausch von Nachrichten, Kommentaren und Medien wie Fotos und Videos.

Es gibt Plattformen für berufliche Netzwerke wie „Xing“ und für Studierende Netzwerke wie „StudiVZ“. Soziale Netzwerke können je nach Zielgruppe unterschieden werden. Andere Plattformen wie „MySpace“ oder „Facebook“ sind weit verbreitet und haben eine breite Nutzerbasis. Diese Netzwerke ermöglichen es den Nutzern insgesamt, miteinander zu kommunizieren, Ideen auszutauschen und Inhalte auszutauschen.<sup>2</sup>

---

<sup>2</sup> Kuphal, A. (2010, S. 2). Soziale Netzwerke und ihre Vor- und Nachteile: Speziell: Cybermobbing. GRIN Verlag

## 2.2. Technologische Grundlagen: Laravel und React

### React

Die JavaScript-Bibliothek React.js, auch bekannt als React, wird häufig genutzt, um dynamische Benutzeroberflächen (User Interfaces, UIs) zu erstellen. Die Grundlage dafür sind wiederverwendbare Komponenten. Dies ermöglicht eine modulare und effiziente Gestaltung verschiedener Teile einer Anwendung. Entwickler haben die Möglichkeit, eine Komponente wie eine Navigationsleiste oder einen Hauptinhalt einmal zu erstellen und sie überall zu nutzen, wo sie gebraucht wird, anstatt redundanten Code zu wiederholen.

Ein wichtiges Kennzeichen von React ist, dass es eine Single-Page-Anwendung (SPA) darstellt. Bei React wird der Seiteninhalt direkt über die React-Komponenten gerendert, im Gegensatz zu konventionellen Webanwendungen, die bei jeder neuen Seite eine Anfrage an den Server senden. Dies hat zur Folge, dass das Benutzererlebnis schneller und reibungsloser verläuft, da die Seite nicht wieder geladen werden muss.

Darüber hinaus nutzt React häufig JSX (JavaScript XML), eine JavaScript-Syntaxerweiterung, die es Programmierern erlaubt, Benutzeroberfläche und Logik auf innovative Weise zu verknüpfen. Die Interaktion mit dem DOM wird durch JSX vereinfacht und die Notwendigkeit expliziter DOM-Manipulationen wird reduziert.<sup>3</sup>

Da Benutzeroberflächenelemente effizient in verschiedene Anwendungsbereiche integriert und wiederverwendet werden können, fördert React komponentenbasierte Architektur Modularität und Wiederverwendbarkeit. Dies führt zu einer signifikanten Steigerung der Entwicklungsproduktivität. Außerdem ist die Verwendung des virtuellen DOMs und der Single-Page-Application (SPA)-Architektur ein wichtiger Vorteil für soziale Netzwerke mit oft wechselnden Inhalten, da sie eine schnelle Leistung und ein reibungsloses Benutzererlebnis

---

<sup>3</sup> Kinsta. (2023b, Oktober 9). Was ist React.js? Ein Blick auf die beliebte JavaScript-Bibliothek. Kinsta®. <https://kinsta.com/de/wissensdatenbank/was-ist-react-js/>



gewährleistet. Außerdem lässt sich React dank seiner Erweiterbarkeit problemlos mit anderen Technologien integrieren. Dies macht es besonders geeignet für Webanwendungen mit Skalierbarkeit und zeitgemäßer Gestaltung.<sup>4</sup>

## Laravel

Laravel dient der Erstellung stabiler und skalierbarer Webanwendungen und ist ein zeitgemäßes PHP-Framework. Die Entwicklungszeit wird deutlich verkürzt, da es viele vordefinierte Funktionen wie Authentifizierung, Routing und die Einbindung von HTML-Vorlagen bereitstellt. Laravel stützt sich auf das Architekturmodell des Model-View-Controllers (MVC), welches eine deutliche Abgrenzung von Daten, Logik und Benutzeroberfläche sicherstellt.

Das objektrelationale Mapping (ORM) ist ein weiteres bedeutendes Merkmal von Laravel, das eine unkomplizierte Verwaltung von Datenbankoperationen erlaubt. Dies macht den Zugang zu Daten und deren Manipulation deutlich einfacher. Laravel bietet auch Unterstützung für Abhängigkeitsmanagement und modulare Paketsysteme, was es Entwicklern ermöglicht, die Anwendung problemlos, um weitere Features zu erweitern.<sup>5</sup>

Laravel ist perfekt geeignet, um das Backend dieser Plattform zu entwickeln, da es eine hohe Skalierbarkeit aufweist und problemlos mit Millionen von Anfragen pro Monat umgehen kann – ein wichtiger Vorteil für wachsende soziale Netzwerke. Eine ausdrucksstarke Kommandozeilenschnittstelle und vorgefertigte Module, die Funktionen wie Authentifizierung und Datenbankmigrationen effizient umsetzen, unterstützen die schnelle Entwicklung. Darüber hinaus schützt Laravel Benutzerdaten mithilfe zuverlässiger Sicherheitsfunktionen und bietet einen wirksamen Schutz vor gängigen Sicherheitsrisiken wie SQL-Injections.<sup>6</sup>

---

<sup>4</sup> Kinsta. (2023b, Oktober 9). Was ist React.js? Ein Blick auf die beliebte JavaScript-Bibliothek. Kinsta®. <https://kinsta.com/de/wissensdatenbank/warum-react/>

<sup>5</sup> Kinsta. (2023, 6. Oktober). Das Laravel PHP-Framework - Web-App-Konstruktion für jedermann. Kinsta®. <https://kinsta.com/de/wissensdatenbank/was-ist-laravel/>

<sup>6</sup> Kinsta. (2023, 6. Oktober). Das Laravel PHP-Framework - Web-App-Konstruktion für jedermann. Kinsta®. <https://kinsta.com/de/wissensdatenbank/warum-solltest-du-laravel-verwenden/>

## 2.3. User Experience (UX) und Design-Prinzipien

*Die User Experience (UX) ist von entscheidender Bedeutung für die Entwicklung zeitgemäßer Webanwendungen, vor allem in sozialen Netzwerken, in denen die Interaktionen der Nutzer von zentraler Bedeutung sind. Die User Experience wird als das Zusammenspiel von Nutzer und Produkt definiert und wie diese Interaktionen positiv oder negativ wahrgenommen werden. Ein erfolgreiches UX-Design ermöglicht es den Nutzern, die Anwendung intuitiv und leicht zu bedienen. Dies führt zu einer Steigerung ihrer Zufriedenheit und Bindung an das Produkt.*

### **Grundlegende UX-Design-Prinzipien:**

- **Benutzerzentriertes Design:** Ein gutes UX-Design konzentriert sich stets auf den Benutzer. Unger und Chandler heben hervor, dass es für die Entwicklung eines Produkts von entscheidender Bedeutung ist, die Bedürfnisse, Verhaltensweisen und Ziele der Benutzer genau zu verstehen. Um ein Verständnis für die Schwierigkeiten und Bedürfnisse der Zielgruppe zu erlangen, ist es häufig notwendig, Benutzerforschung durchzuführen, zum Beispiel durch Umfragen, Interviews oder Beobachtungen.
- **Klarheit und Einfachheit:** Die Benutzeroberflächen von sozialen Netzwerken wie Facebook und Instagram sind so konzipiert, dass sie übersichtlich und klar bleiben, obwohl sie eine Vielzahl von Funktionen bieten. Unger und Chandler legen großen Wert darauf, einfache und klare Schnittstellen zu entwickeln, die leicht verständlich sind, ohne viel Erklärung. Das heißt, dass die Schaltflächen deutlich markiert und die Navigation einfach sein sollte, damit die Nutzer ein schnelles und effizientes Erlebnis haben.
- **Konsistenz:** Im UX-Design ist Konsistenz ein weiteres wichtiges Grundsatzprinzip. Benutzer erwarten, dass sich die Interaktion in allen Abschnitten der Anwendung gleichmäßig gestaltet. Beispielsweise sollte dies in der gesamten Anwendung gleich sein, wenn bestimmte Aktionen wie das „Teilen“ oder „Kommentieren“ an einer Stelle auf eine bestimmte Weise durchgeführt

werden. Dadurch wird die kognitive Belastung verringert und die Nutzerfreundlichkeit verbessert.

- **Feedback und Reaktionsfähigkeit:** Gemäß Unger und Chandler ist es von großer Bedeutung, dass die Benutzer regelmäßig Rückmeldungen von der Anwendung bekommen. Dies kann mittels Bestätigungsmeldungen, Ladeindikatoren oder Warnungen erfolgen, welche dem Nutzer mitteilen, dass seine Handlung registriert wurde. Die Fähigkeit zur Reaktion ist ebenfalls von großer Bedeutung, vor allem in sozialen Netzwerken, da die Nutzer davon ausgehen, dass ihre Beiträge sofortige Resultate liefern.
- **Zugänglichkeit:** Ein effektives UX-Design muss für eine Vielzahl von Nutzern zugänglich sein, auch für solche mit Behinderungen. Das heißt, dass die Anwendung sowohl für Personen mit motorischen Einschränkungen als auch für Personen mit Seh- oder Hörstörungen verwendbar sein muss. Unger und Chandler heben hervor, dass ein Design ohne Barrieren nicht nur den Anwendungsbereich erweitert, sondern auch das Benutzererlebnis für alle verbessert.

Die Bedeutung von UX für dieses Projekt liegt darin begründet, dass die User Experience beziehungsweise die Erfahrung der Studenten in sozialen Netzwerken entscheidend ist, um die Bindung von Nutzern und die Förderung von Interaktionen sicherzustellen. Ein sorgfältig ausgearbeitetes UX-Design ermöglicht es den Nutzern, Beiträge problemlos zu erstellen, zu kommentieren oder zu teilen, ohne frustriert zu werden. Die Schnittstellen in sozialen Netzwerken müssen flexibel, aber gleichzeitig klar strukturiert sein, da sie dynamisch sind und häufig von nutzergeneriertem Inhalt handeln.<sup>7</sup>

---

<sup>7</sup> Unger, R. & Chandler, C. (2012, Kapitel 1). A Project Guide to UX-Design: For user experience designers in the field or in the making. New Riders.

### **Grundsätze des Designs dieses Projekts:**

- **Intuitive Navigation:** Die Plattform soll den Benutzern die Möglichkeit bieten, schnell auf die gewünschten Funktionen zuzugreifen, ob es darum geht, einen neuen Beitrag zu schreiben oder mit anderen Mitgliedern in Kontakt zu treten.
- **Konsistentes visuelles Design:** Ein einheitliches visuelles Design gewährleistet ein vertrautes und wiedererkennbares Erlebnis, indem die verschiedenen Interface-Komponenten wie Buttons und Menüs einheitlich gestaltet werden.
- **Responsiveness:** Da soziale Netzwerke von vielen Nutzern auf unterschiedlichen Geräten (Desktop, Tablet, Smartphone) genutzt werden können, ist das Design anpassungsfähig und responsive.
- **User-Feedback:** Es ist wichtig, den Nutzern fortlaufend mitzuteilen, ob ihre Handlungen (z. B. das Teilen eines Beitrags oder das Liken eines Kommentars) erfolgreich waren.

### **3. Anforderungsanalyse**

Im Softwareentwicklungsprozess ist die Anforderungsanalyse ein wichtiger Schritt, um sicherzustellen, dass sämtliche Erwartungen und Bedürfnisse der Nutzer sowie anderer Interessengruppen deutlich festgelegt und begriffen werden. Sie beinhaltet die systematische Erfassung, Dokumentation und Überprüfung von Anforderungen, um sicherzustellen, dass sie in der späteren Entwicklungsphase ohne Probleme umgesetzt werden können. Dabei werden nicht-funktionale Anforderungen, die Qualitätsaspekte wie Leistung, Sicherheit und Benutzerfreundlichkeit betreffen, sowie funktionale Anforderungen berücksichtigt, welche die Funktionalität der Software beschreiben. Eine genaue Auswertung der Anforderungen reduziert die Wahrscheinlichkeit von Missverständnissen und verhindert mögliche Verzögerungen oder teure Änderungen im Verlauf des Projekts.<sup>8</sup>

Für die Erstellung der funktionalen und nicht-funktionalen Anforderungen, wurde die Software Miro als Hilfsmittel verwendet. Diese werden zunächst in dem folgenden Unterpunkt dargestellt.

#### **3.1. Funktionale Anforderungen**

Funktionale Anforderungen beschreiben die spezifischen Funktionen und Fähigkeiten einer Software, die notwendig sind, um die geschäftlichen Bedürfnisse zu erfüllen. Diese Anforderungen definieren, was das System tun soll, und dienen als Grundlage für die Entwicklung und Implementierung der Funktionen.<sup>9</sup>

---

<sup>8</sup> Steinweg, C. (2013, S. 92). Projektkompass Softwareentwicklung: Geschäftsorientierte Entwicklung von IT-Systemen. Springer-Verlag.

<sup>9</sup> Vogel, O., Arnold, I., Chughtai, A., Ihler, E., Kehrler, T., Mehlig, U. & Zdun, U. (2009, S. 476). Software-Architektur: Grundlagen - Konzepte - Praxis. Springer-Verlag.

Nachfolgend werden die funktionalen Anforderungen als Bildkatalog dargestellt.

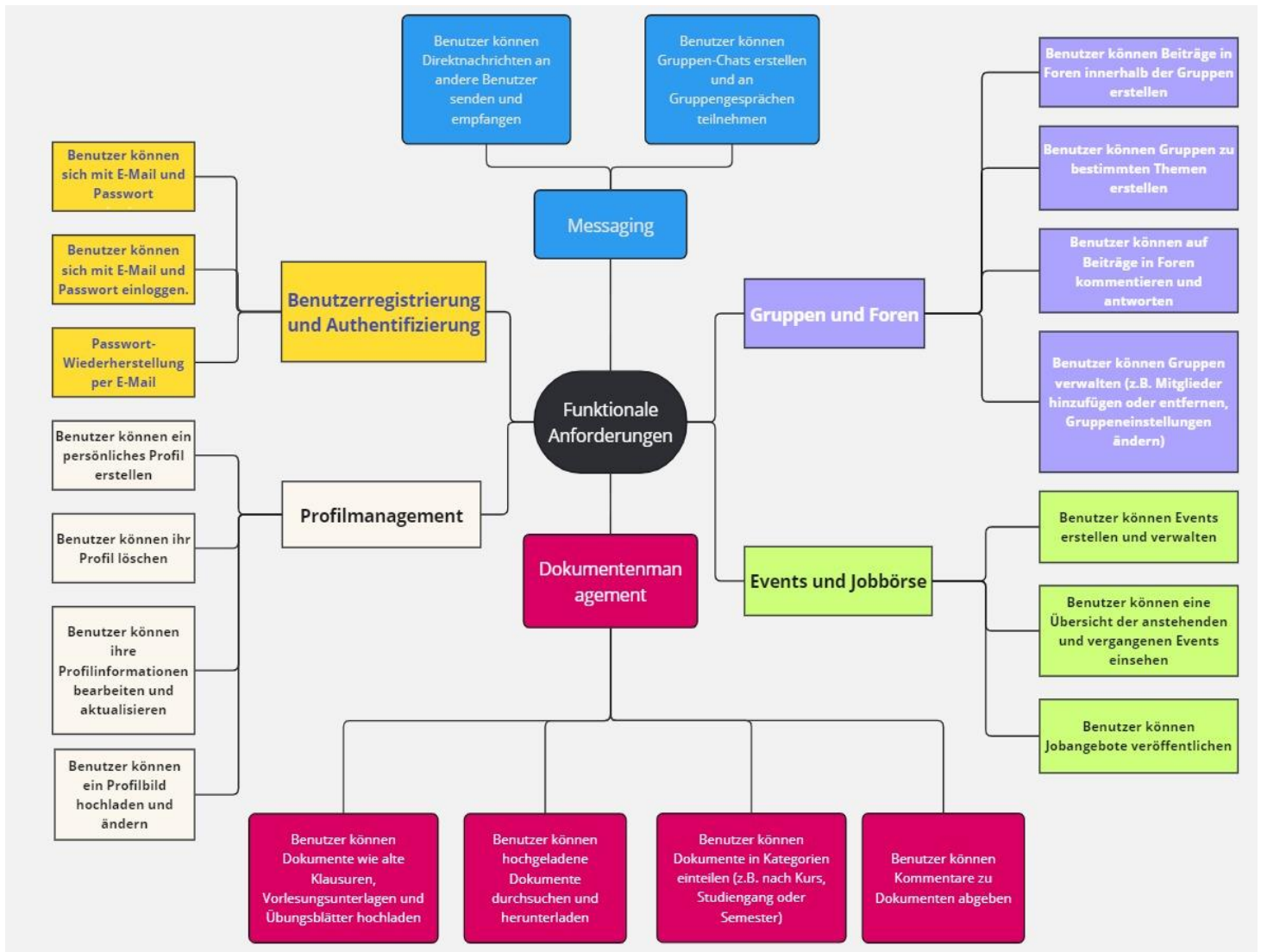


Abb. 2: Funktionale Anforderungen (eigene Erstellung)

Das Bild zeigt die verschiedenen funktionalen Anforderungen für die Web-App-StudentCommunity an. Es ist in sechs Haupt Anforderungen nämlich Messaging, „Benutzerregistrierung und Authentifizierung“, „Gruppen und Foren“, Profilmanagement, Dokumentenmanagement und „Events und Jobbörse“ strukturiert.

➤ **Messaging**

- Benutzer können Direktnachrichten an andere Benutzer senden und empfangen.
- Benutzer können Gruppenchats erstellen und an Gruppengesprächen teilnehmen.

➤ **Benutzerregistrierung und Authentifizierung**

- Benutzer können sich mit E-Mail und Passwort einloggen.
- Passwort Wiedererstellung per E-Mail

➤ **Gruppen und Foren**

- Benutzer können Beiträge in Foren innerhalb der Gruppen erstellen.
- Benutzer können Gruppen zu bestimmten Themen erstellen.
- Benutzer können auf Beiträge in Foren kommentieren und antworten.
- Benutzer können Gruppen verwalten (z. B. Mitglieder hinzufügen oder entfernen, Gruppeneinstellungen ändern).

➤ **Profilmanagement**

- Benutzer können ein persönliches Profil erstellen.
- Benutzer können Ihr Profil löschen.
- Benutzer können Ihre Profilinformationen bearbeiten und aktualisieren.
- Benutzer können ein Profilbild hochladen und ändern.

➤ **Dokumentenmanagement**

- Benutzer können Dokumente wie alte Klausuren, Vorlesungsunterlagen und Übungsblätter hochladen.
- Benutzer können hochgeladene Dokumente durchsuchen und herunterladen.
- Benutzer können Dokumente in Kategorien einstellen (z. B. nach Kurs, Studiengang oder Semester).

- Benutzer können Kommentare zu Dokumenten abgeben.

### ➤ **Events und Jobbörse**

- Benutzer können Events erstellen und verwalten.
- Benutzer können eine Übersicht der anstehenden und vergangenen Events einsehen.
- Benutzer können Jobangebote veröffentlichen.

## **3.2. Nicht-funktionale Anforderungen**

Nicht-funktionale Anforderungen spezifizieren die Qualitätsmerkmale eines Systems und definieren, wie es die beschriebenen Funktionen erbringen soll. Diese Anforderungen sind entscheidend für die Benutzerfreundlichkeit, Wartbarkeit und Leistung der Software.<sup>10</sup>

Wie für die funktionalen Anforderungen werden die nicht funktionalen Anforderungen in der Folge angezeigt.

---

<sup>10</sup> Vogel, O., Arnold, I., Chughtai, A., Ihler, E., Kehrer, T., Mehlig, U. & Zdun, U. (2009, S. 476). Software-Architektur: Grundlagen - Konzepte - Praxis. Springer-Verlag.



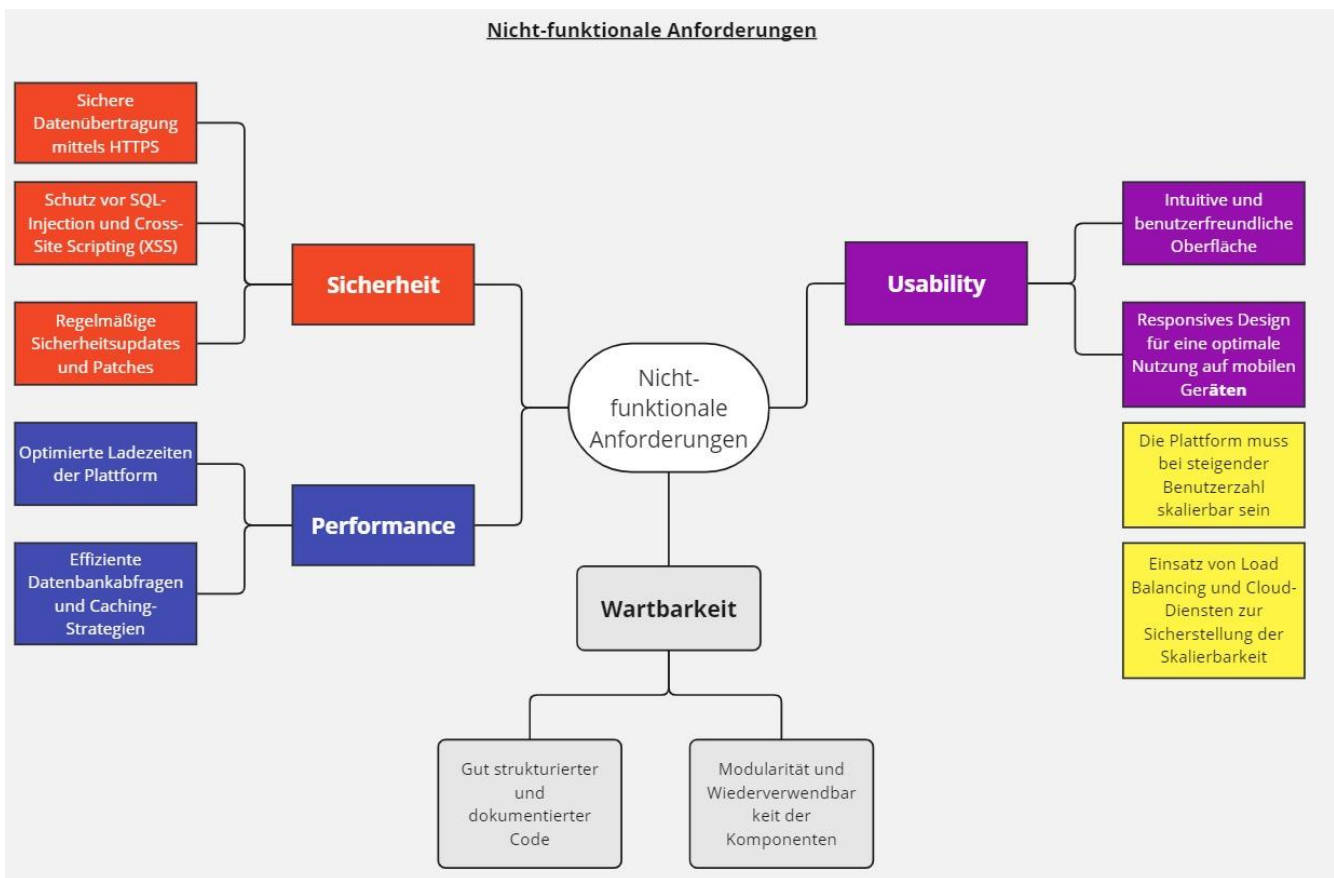


Abb. 3: Nicht-funktionale Anforderungen (eigene Erstellung)

Es gibt aus dem Bild vier Haupt nichtfunktionale Anforderungen. Als Erstes gibt es die Usability, die Sicherheit, die Performance und die Wartbarkeit.

### ➤ Usability

- Intuitive und benutzerfreundliche Oberfläche.
- Responsives Design für eine optimale Nutzung auf mobilen Geräten.

### ➤ Sicherheit

- Sichere Datenübertragung mittels HTTPS.
- Schutz vor SQL-Injection und Cross-Site Scripting (XSS).
- Regelmäßige Sicherheitsupdates und Patches.

➤ **Performance**

- Optimierte Ladezeiten der Plattform.
- Effiziente Datenbankabfragen und Caching-Strategien.

➤ **Wartbarkeit**

- Gut strukturierter und dokumentierter Code.
- Modularität und wieder Wiederverwendbarkeit der Komponenten.

Zusammengefasst sind es diese Anforderungen, die die Merkmale und Funktionen der Anwendung StudentCommunity-WebApp definieren.

## 4. Systemarchitektur (High-Level-Architektur)

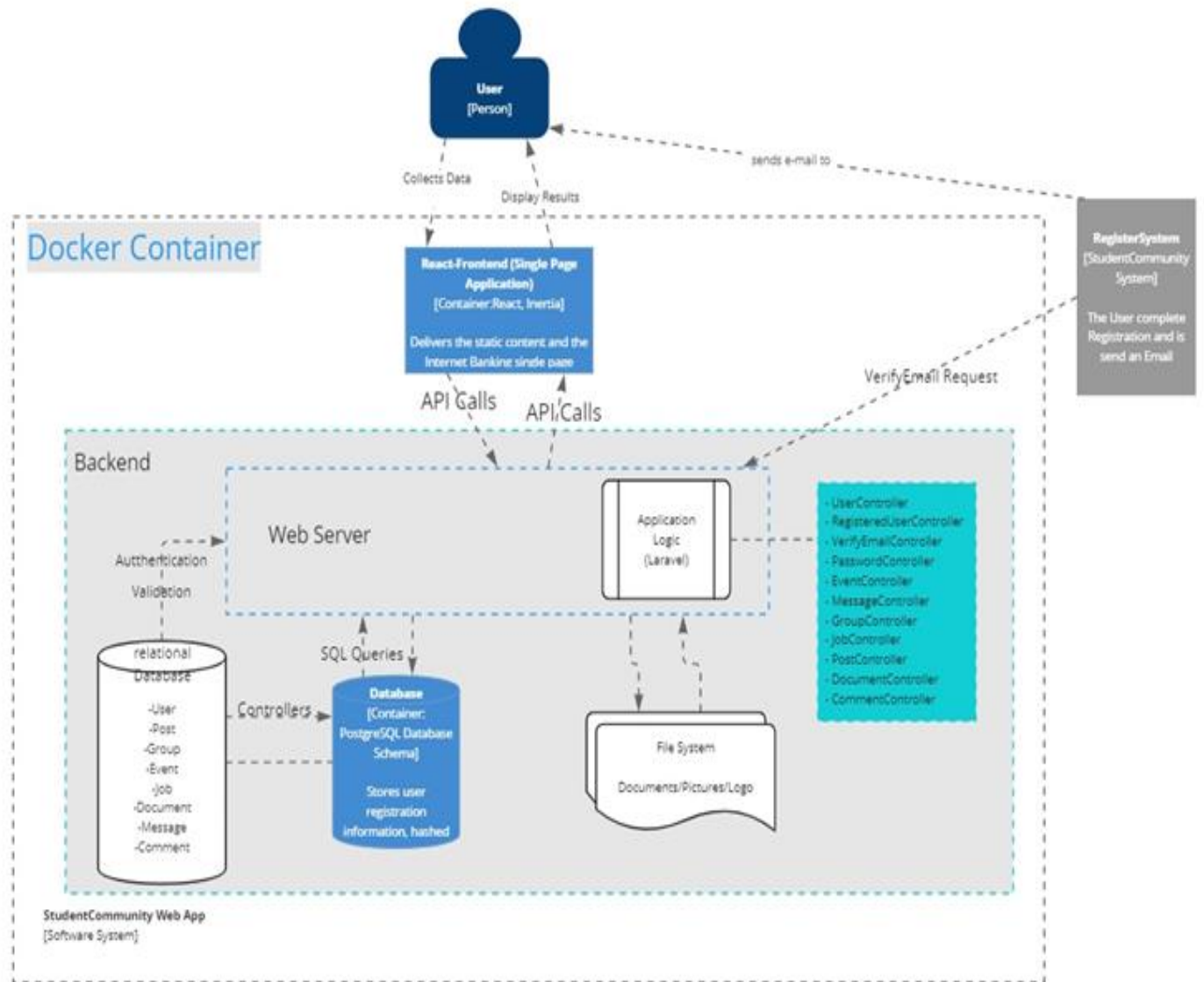
Das oberste strukturelle Konzept eines Softwaresystems ist die Softwarearchitektur. Sie fungiert als „Blaupause“. Die Elemente des Systems, ihre Beziehungen und Interaktionen werden von ihr beschrieben. Dabei bleibt sie auf einer abstrakten Ebene, ohne sich in Details der Implementierung zu vertiefen. Eine gute Bauweise sorgt dafür, dass das gesamte System verstanden werden kann und gewährleistet, dass sämtliche funktionalen und nichtfunktionalen Anforderungen erfüllt sind.

Eine leistungsfähige Software-Architektur weist verschiedene wesentliche Eigenschaften auf. Durch Abstraktion kann das System in seiner Gesamtheit auf einer höheren Ebene betrachtet werden. Die Struktur beinhaltet die erforderlichen Funktionen und das sich ständig verändernde Verhalten des Systems. In diesem Zusammenhang werden die Anforderungen an das System wie Leistung, Sicherheit, Flexibilität und Wartungsfreundlichkeit beachtet. Die Abstraktion von Details führt dazu, dass Designentscheidungen auf einem niedrigen Niveau vernachlässigt werden, wodurch die Architektur auf das Wesentliche fokussiert wird.

Sowohl technische als auch organisatorische Ziele werden von einer guten Softwarearchitektur unterstützt. Sie fungiert als Mittel zur Kommunikation mit Interessengruppen, da sie das High-Level-Design deutlich präsentiert. Der „rote Faden“ gibt Entwicklern einen Überblick über das System und hilft ihnen, sich zu orientieren. Durch die Eingliederung in Arbeitsblöcke ist es möglich, die Entwicklung parallel durch separate Module voranzutreiben. Außerdem trägt die Architektur dazu bei, die Systemanforderungen zu erfüllen, sei es funktional oder nicht-funktional. Schließlich können Anpassungen und Erweiterungen im Lebenszyklus kostengünstig umgesetzt werden, wodurch die Struktur Wartungs- und Entwicklungskosten verringert.<sup>11</sup>

---

<sup>11</sup> Software-Architektur | PFP Software GmbH. (o. D.).  
<https://pfp.gmbh/beratung/architektur/>



Container Diagram for StudentCommunit Web App

Abb. 4: StudentCommunity System Architektur (eigene Erstellung)

Dieses Diagramm stellt die verschiedenen Komponenten und ihre Interaktionen dar, die zusammenarbeiten, um die Hauptfunktionalitäten der Plattform zu ermöglichen. Die Architektur umfasst das Frontend, das Backend, die Datenbank und das Dateisystem, und zeigt die Kommunikationswege und Abhängigkeiten zwischen diesen Teilen.

## Aufbau und Architekturkonzepte des Student Community Web-App Systems

- **Frontend (React/Inertia)** Das Frontend ist eine React-basierte Single-Page-Anwendung, die innerhalb eines eigenen Docker-Containers läuft. Es stellt die Benutzeroberfläche bereit und kommuniziert über API-Calls mit dem Backend, um Daten anzuzeigen und zu aktualisieren. Diese Trennung ermöglicht eine klare Unterscheidung zwischen Präsentation und Logik, was die **Flexibilität** und **Wiederverwendbarkeit** fördert.
- **Backend (Laravel)** Das Backend, ebenfalls in einem Docker-Container, besteht aus einer Webserver-Komponente, einer Anwendungslogik (Laravel) und einem relationalen Datenbanksystem (PostgreSQL). Die Architektur enthält eine Reihe von **Controllern** (z. B. UserController, EventController), die die Funktionalitäten des Systems kapseln und Schnittstellen für das Frontend bereitstellen. Diese Struktur trägt zur **Kapselung der Funktionalität** und zur **Entkopplung** der Komponenten bei, was die **Erweiterbarkeit** und **Wartbarkeit** verbesserten.
- **Datenbank (PostgreSQL)** Die Datenbank speichert Benutzer-, Post-, Gruppen-, Event- und Jobdaten und stellt SQL-Schnittstellen zur Verfügung, über die die Anwendung Daten speichert und abrufen. Diese Schicht ermöglicht eine saubere Trennung zwischen **Datenpersistenz** und **Anwendungslogik**, was die **Stabilität** und **Performanz** verbesserten.
- **Registersystem** Das Registrierungssystem verwaltet die Benutzerregistrierung und versendet Bestätigungs-E-Mails, um neue Benutzer zu authentifizieren. Diese Komponente sorgt für den **Schutz vor unbefugtem Zugriff** und erleichtert die **Verwaltung der Benutzeridentität**.

**Filesystem** Das Dateisystem dient zur Speicherung hochgeladener Dokumente und anderer Dateien. Dies entlastet die Datenbank und vereinfacht die Verwaltung von Dokumenten und großen Dateien. Die dargestellte Architektur fördert eine modulare und skalierbare Struktur, die es ermöglicht, das System leicht zu erweitern und an zukünftige Anforderungen anzupassen.

## 4.1. Datenbankdesign (Grundlegende Tabellen)

Die Basis für ein effizientes Datenmanagement und eine optimale Leistung in IT-Systemen ist ein sorgfältig geplantes und gut strukturiertes Datenbankdesign. Eine Datenbank setzt sich aus Tabellen zusammen, die miteinander verknüpft und mithilfe von SQL-Abfragen zugänglich gemacht werden können. Es existieren verschiedene Formen von Datenbanksystemen: SQL-Datenbanken, die auf relationalen Strukturen beruhen und durch ihre Stabilität und eindeutigen Standards gekennzeichnet sind, sowie NoSQL-Datenbanken, die sich durch ihre Möglichkeit zur Speicherung unstrukturierter Daten auszeichnen.

Das Ziel eines guten Datenbankdesigns ist es, Informationen auf eine Weise zu strukturieren, die sie effizient abrufen, ändern und speichern kann. Die Minimierung von überflüssigen Daten, die Gewährleistung der Datenintegrität und die Skalierbarkeit sind hierbei von entscheidender Bedeutung. Eine gut konzipierte Datenbank ist für künftiges Wachstum gerüstet und hilft nicht nur bei geplanten Abfragen, sondern auch bei flexiblen Ad-hoc-Anfragen. Ein gutes Design stellt sicher, dass das System trotz des zunehmenden Datenvolumens schnell und verlässlich bleibt.<sup>12</sup>

Im nächsten Teil wird das Datenbankdesign des StudentCommunity-Systems präsentiert. Das Design gewährleistet eine strukturierte und effiziente Speicherung sämtlicher relevanter Daten, die für die Verwaltung von Studierenden, Kursen, Veranstaltungen und Interaktionen in der Gemeinschaft relevant sind. Die Basistabellen und Beziehungen wurden entwickelt, um eine hohe Datenintegrität sowie flexible Erweiterungsmöglichkeiten sicherzustellen. Mit dieser Struktur der Datenbank kann für die StudentCommunity eine benutzerfreundliche und leistungsfähige Plattform bereitgestellt werden, die den einfachen Abruf und die Verwaltung von Informationen ermöglicht.

---

<sup>12</sup> Garden, T. (2020, 2. Oktober). Grundlagen Datenbankdesign: So funktioniert der Einstieg. talentgarden. <https://blog.talentgarden.com/de/blog/data/grundlagen-datenbankdesign-so-funktioniert-der-einstieg>

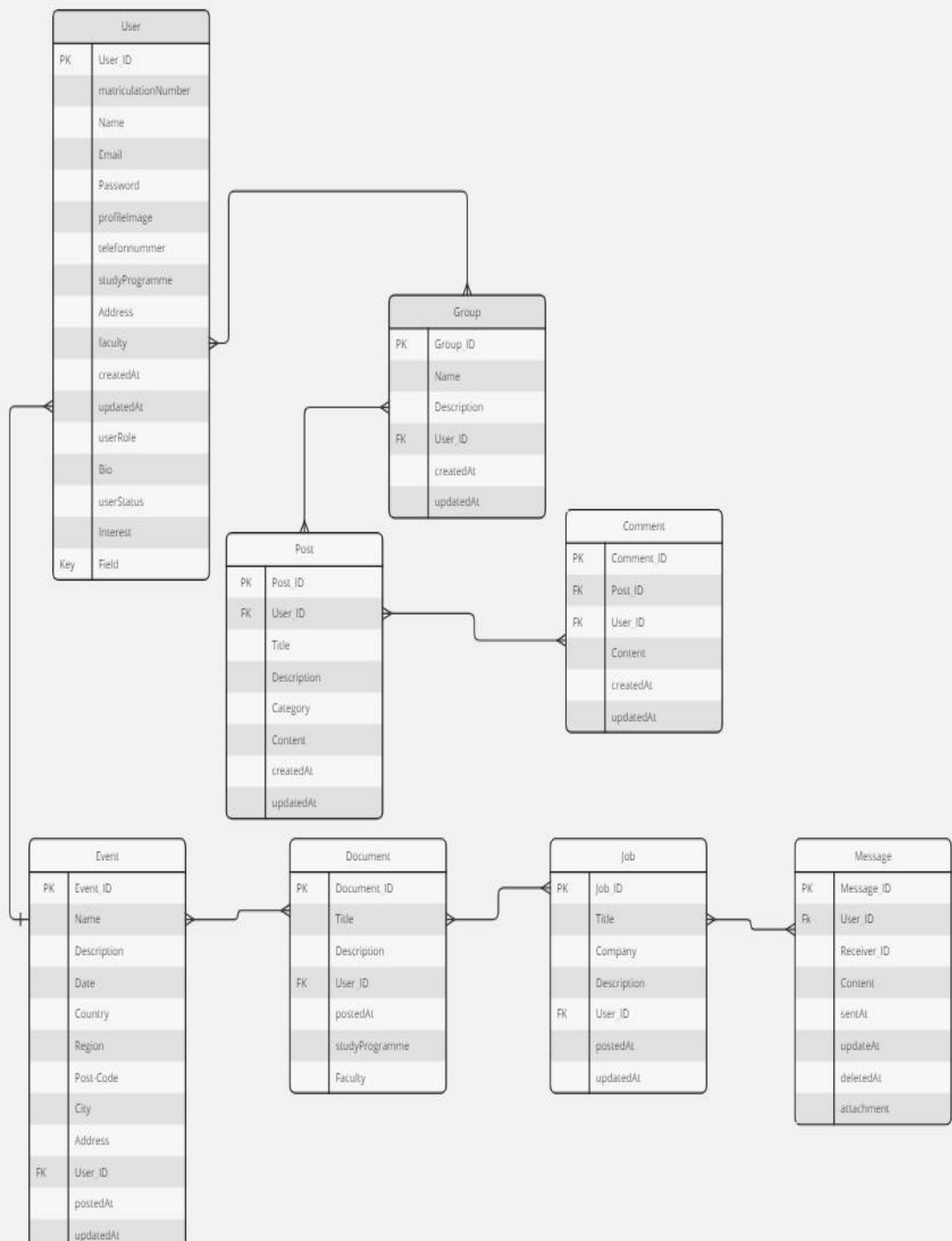


Abb. 5: StudentCommunity Datenbank Struktur (eigene Erstellung)

Das ERM-Diagramm zeigt die grundlegenden Tabellen und deren Beziehungen, die das Datenmodell des

Systems formen. Im Folgenden werden die zentralen Tabellen des Systems und ihre wesentlichen Attribute beschrieben.

### **Zentrale Tabellen des Datenmodells:**

#### ➤ **User (Nutzer)**

- **Zweck:** Speichert Informationen über die Benutzer der Plattform.
- **Wichtige Felder:** User\_ID (Primärschlüssel), Name, E-Mail, Passwort, Rolle.
- **Beschreibung:** Diese Tabelle enthält die Grundinformationen der Benutzer und unterstützt die Verwaltung von verschiedenen Benutzerrollen sowie Authentifizierung und Autorisierung.

#### ➤ **Group (Gruppe)**

- **Zweck:** Erfasst Gruppen, in denen Benutzer zusammenarbeiten oder Informationen teilen können.
- **Wichtige Felder:** Group\_ID (Primärschlüssel), Name, Beschreibung, User\_ID (Verweis auf Gruppenadministrator).
- **Beschreibung:** Ermöglicht die Bildung und Verwaltung von Gruppen innerhalb der Community. Über die Beziehung zur User-Tabelle kann jede Gruppe einen Administrator zugewiesen bekommen.

#### ➤ **Post (Beitrag)**

- **Zweck:** Speichert Beiträge, die von Benutzern erstellt werden, um Informationen zu teilen.
- **Wichtige Felder:** Post\_ID (Primärschlüssel), Title, Description, Category, User\_ID (Verweis auf Ersteller).
- **Beschreibung:** Diese Tabelle dient zur Speicherung der von Benutzern erstellten Beiträge und umfasst Felder wie Titel und Beschreibung. Jeder Beitrag wird einem Benutzer zugeordnet.



➤ **Comment (Kommentar)**

- **Zweck:** Verfolgt Kommentare zu den Beiträgen, um eine Interaktion zwischen den Benutzern zu ermöglichen.
- **Wichtige Felder:** Comment\_ID (Primärschlüssel), Content, Post\_ID (Verweis auf zugehörigen Beitrag), User\_ID (Verweis auf Ersteller).
- **Beschreibung:** Die Tabelle stellt eine Verbindung zwischen Benutzern und Beiträgen her, um Benutzerkommentare zu verwalten.

➤ **Event (Veranstaltung)**

- **Zweck:** Speichert Informationen zu Veranstaltungen, die innerhalb der Community organisiert werden.
- **Wichtige Felder:** Event\_ID (Primärschlüssel), Title, Description, Date, Location.
- **Beschreibung:** Ermöglicht die Planung und Ankündigung von Events. Die Event-Tabelle enthält Informationen über Titel, Beschreibung, Ort und Datum.

➤ **Document (Dokument)**

- **Zweck:** Verwaltung von Dokumenten, die Benutzer hochladen und teilen können.
- **Wichtige Felder:** Document\_ID (Primärschlüssel), Title, Path, User\_ID (Verweis auf Ersteller).
- **Beschreibung:** Dokumente werden in dieser Tabelle erfasst, einschließlich des Dateipfads und eines Titels. Die User\_ID verknüpft das Dokument mit dem jeweiligen Ersteller.

➤ **Job (Stellenanzeige)**

- **Zweck:** Speichert Stellenangebote, die von Unternehmen oder Benutzern ausgeschrieben werden können.
- **Wichtige Felder:** Job\_ID (Primärschlüssel), Title, Description, Company.

- **Beschreibung:** Diese Tabelle enthält Informationen zu offenen Stellen, die von Benutzern oder Unternehmen ausgeschrieben werden können, und umfasst Titel, Beschreibung und Unternehmensname.

➤ **Message (Nachricht)**

- **Zweck:** Ermöglicht private Nachrichten zwischen Benutzern der Community.
- **Wichtige Felder:** Message\_ID (Primärschlüssel), Content, Sender\_ID (Absender), Receiver\_ID (Empfänger).
- **Beschreibung:** Die Nachrichtentabelle ermöglicht die Kommunikation zwischen Benutzern und speichert den Inhalt und die beteiligten Benutzer (Sender und Empfänger).

**Beziehungen zwischen Tabellen**

- **Benutzer und Gruppen:** Jeder Benutzer kann Mitglied in mehreren Gruppen sein, und jede Gruppe hat einen Administrator, der über die User\_ID in der Group\_Tabelle definiert ist.
- **Beitrag und Kommentar:** Ein Beitrag kann mehrere Kommentare haben. Jeder Kommentar ist einem spezifischen Beitrag zugeordnet, was durch die Referenz zur Post-ID in der Kommentar-Tabelle sichergestellt wird.
- **Benutzer und Veranstaltungen:** Benutzer können Veranstaltungen organisieren. Die Event-Tabelle speichert die Einzelheiten jeder Veranstaltung.
- **Dokumente und Benutzer:** Jeder Benutzer kann mehrere Dokumente hochladen. Die Document-Tabelle referenziert den Ersteller jedes Dokuments über User\_ID.
- **Stellenangebote:** Die Job-Tabelle ermöglicht das Speichern und Veröffentlichen von Job-Angeboten durch verschiedene Benutzer oder Unternehmen.

- **Private Nachrichten:** Die Message-Tabelle speichert direkte Nachrichten zwischen Benutzern, ermöglicht durch die Referenzen auf den Absender und Empfänger (Sender\_ID und Receiver\_ID).

## 4.2. User Interface Design (Hauptseiten und Navigation)

Die StudentCommunity-WebApp verwendete das Design-Tool Figma, das für seine Effizienz und Vielseitigkeit bekannt ist, um das User Interface (UI) Design zu entwickeln. Das Design zielte darauf ab, eine benutzerfreundliche, moderne und intuitiv bedienbare Benutzeroberfläche zu entwickeln, die den Bedürfnissen einer sich ständig verändernden Studentenschaft entspricht. Um den Benutzern ein angenehmes und reibungsloses Erlebnis zu ermöglichen, ist das Design auf klare Navigation, Ästhetik und Funktionalität ausgerichtet.

Die wichtigsten Seiten und Elemente, die im Rahmen des UI-Designs erstellt wurden, sind:

**Registerpage:** Für die Registrierung neuer Nutzer.

**Loginpage:** Für den sicheren Zugang zur Plattform.

**Chatpage:** Für die Kommunikation zwischen Nutzern.

**Eventpage:** Für die Organisation von Veranstaltungen.

**Jobpage:** Für die Suche und Veröffentlichung von Jobangeboten.

**Profile Settings Page:** Zur Verwaltung persönlicher Informationen und Einstellungen.

**Sidebar:** Eine zentrale Navigationseinheit, die schnellen Zugriff auf die wichtigsten Funktionen der App bietet.

**Logo**

Im Folgenden werden diese Seiten und deren Hauptmerkmale detailliert vorgestellt, um das Design und die Funktionalität der StudentCommunity-WebApp näher zu erläutern.

➤ **Registerpage:**

Sc StudentCommunity

Bitte geben Sie Ihre Anmeldeinformationen ein

Vorname  
force@adresseemail.com

Nachname  
force@adresseemail.com

Matriko-Nr  
force@adresseemail.com

Adress  
force@adresseemail.com

Studiengang  
force@adresseemail.com

Fachbereich  
force@adresseemail.com

E-Mail-Adresse  
force@adresseemail.com

Telefonnummer  
(+237) 696 88 77 55

Passwort  
.....

Ihr Passwort bestätigen  
.....

☐ Ich stimme zu die Nutzungsbedingungen.

Registrieren

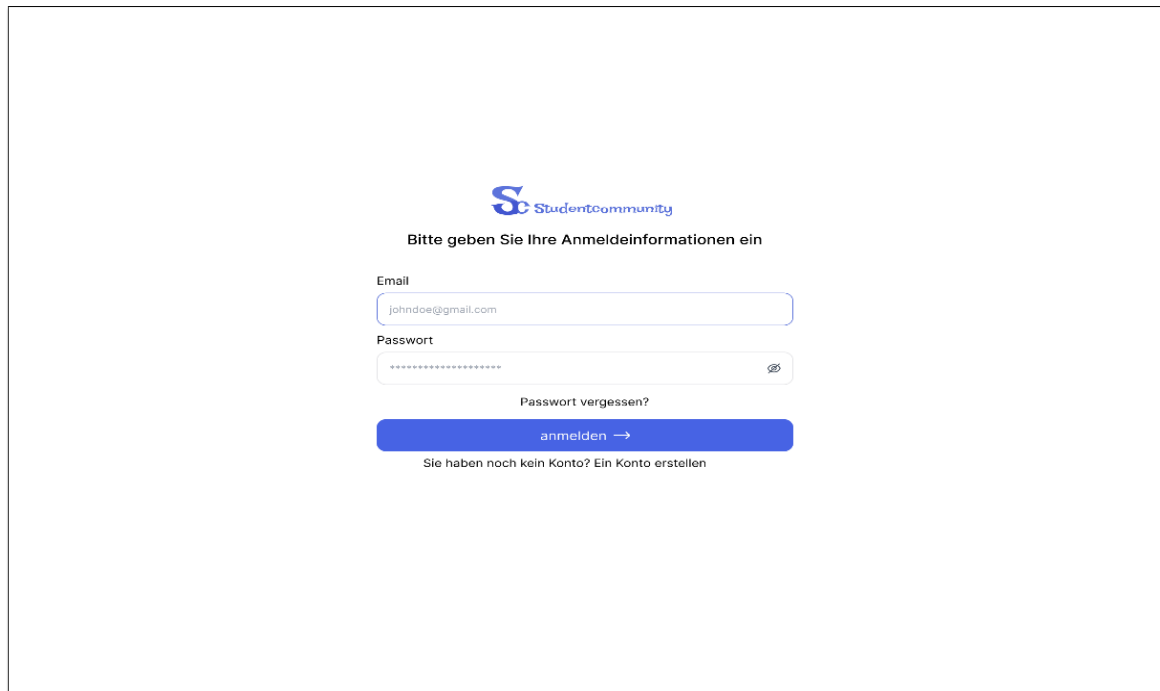
Sie haben bereits ein Konto? [Anmelden](#)

Abb. 6: Register Page (Screenshot)

Die Registrierungsseite ist der erste Berührungspunkt für neue Nutzer der App.

- **Zweck:** Nutzer registrieren sich mit ihren persönlichen Daten, wie Name, E-Mail-Adresse und Passwort. Außerdem können auch Interessen oder Studienrichtungen angegeben werden, um ein personalisiertes Erlebnis zu ermöglichen.
- **Design-Elemente:**
  - Eingabefelder mit klaren Labels und Validierung.
  - Ein Fortschrittsindikator, falls die Registrierung mehrere Schritte umfasst.
  - Ein ansprechender Call-to-Action-Button („Registrieren“).

➤ **Loginpage:**



*Abb. 7: Login Page (Screenshot)*

Diese Seite erlaubt den Nutzern, sich sicher und schnell anzumelden.

- **Zweck:** Authentifizierung bestehender Nutzer durch Eingabe Ihrer Anmeldedaten (E-Mail/Benutzername und Passwort).
- **Design-Elemente:**
  - Minimalistisches Layout für eine schnelle Anmeldung.
  - „Passwort vergessen?“ Option für einfache Kontowiederherstellung.
  - Optionale „Bleiben Sie eingeloggt“-Funktion.
  - Optionale „registrieren“- Leitet zum Registrierung Page.

## ➤ Chatpage:

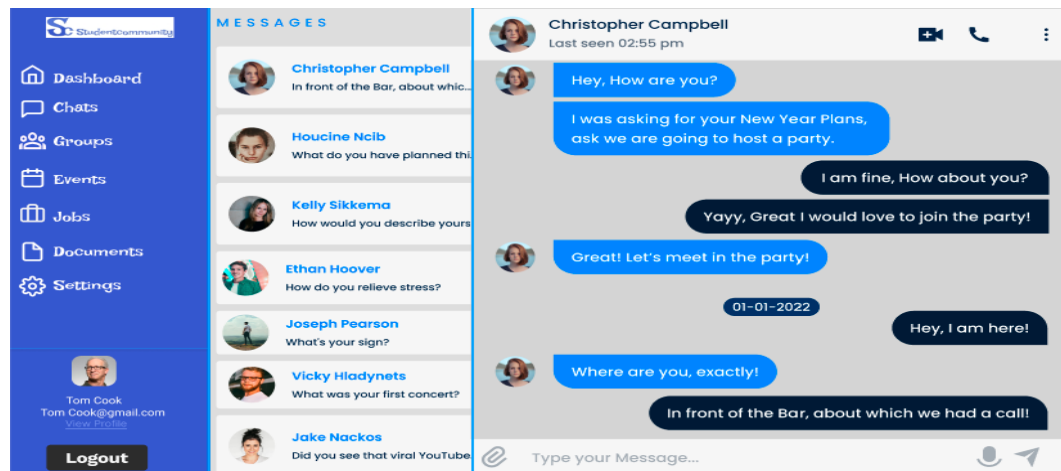


Abb. 8: Chat Page (Screenshot)

Die Chat-Seite ermöglicht die Kommunikation zwischen den Nutzern der Plattform.

- **Zweck:** Echtzeit-Nachrichtenversand zwischen Individuen oder Gruppen.
- **Design-Elemente:**
  - Eine Liste mit Konversationen (linke Seite).
  - Der Haupt-Chat-Bereich mit Nachrichten-Thread (rechte Seite).
  - Funktionen wie Emojis, Anhänge und „Lesen“-Status.

## ➤ Eventpage:

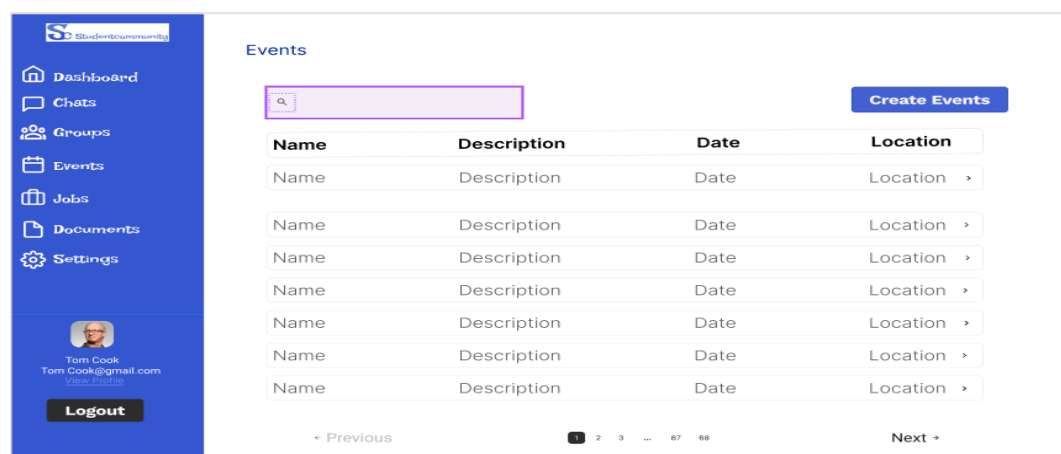


Abb. 9: Event Page (Screenshot)

Die Event-Seite dient der Organisation und Teilnahme an Veranstaltungen.

- **Zweck:** Nutzer können sich über bevorstehende Veranstaltungen informieren und eigene Events erstellen.
- **Design-Elemente:**
  - Kalender- und Listenansicht für anstehende Events.
  - Eventdetails wie Titel, Datum, Ort und Beschreibung.

➤ **Jobpage:**

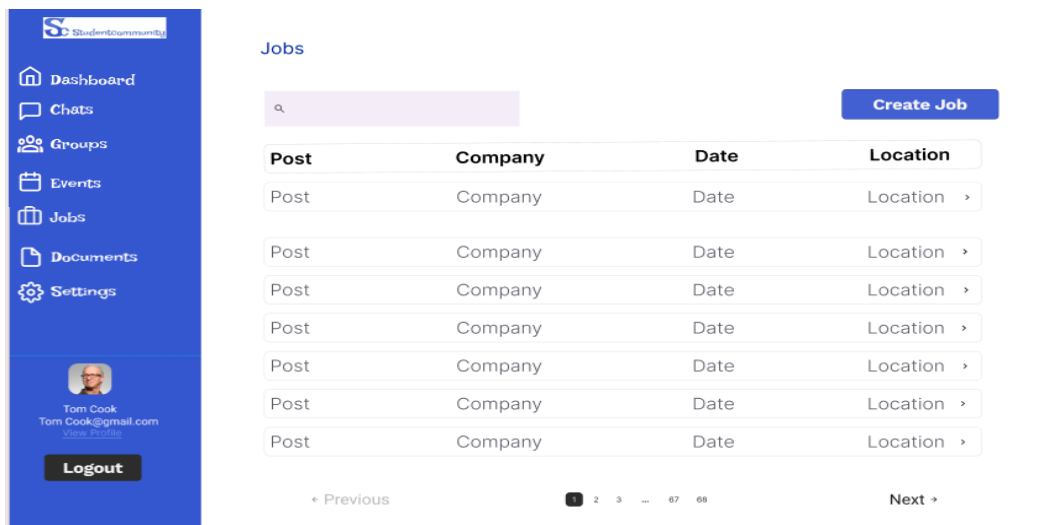


Abb. 10: JobPage (Screenshot)

Diese Seite bietet Zugang zu Jobangeboten und Karrierechancen.

- **Zweck:** Veröffentlichung und Suche nach Stellenangeboten, Praktika oder Projekten.
- **Design-Elemente:**
  - Such- und Filteroptionen (nach Ort, Unternehmen, Datum Anfang, Datum, Stelle Name).
  - Detaillierte Jobanzeigen mit Kontaktinformationen und Bewerbungslinks.

Möglichkeit, eigene Angebote zu veröffentlichen.

➤ **Profile Settings Page:**

StudentCommunity

Dashboard

Chats

Groups

Events

Jobs

Documents

Settings

Profile Settings

Account Settings

Tom Cook  
Tom Cook@gmail.com  
[View Profile](#)

Logout

Profile Settings

Profile Information

Update your account's profile information and email address.

Name

Value

Email

Value

Bio

Value

Interest

Value

Profile\_Picture

Choose File No Chosen File

Save

Abb. 11: Profile Settings Page (Screenshot)

Hier können Nutzer ihre persönlichen Informationen anpassen.

- **Zweck:** Verwaltung des Nutzerprofils, einschließlich Profilbild und Bio.
- **Design-Elemente:**
  - Klare Abschnitte für persönliche Informationen, App-Benachrichtigungen und Sicherheit.
  - Änderungsbuttons mit sofortiger Bestätigung oder Speichermöglichkeit.



➤ **Sidebar:**

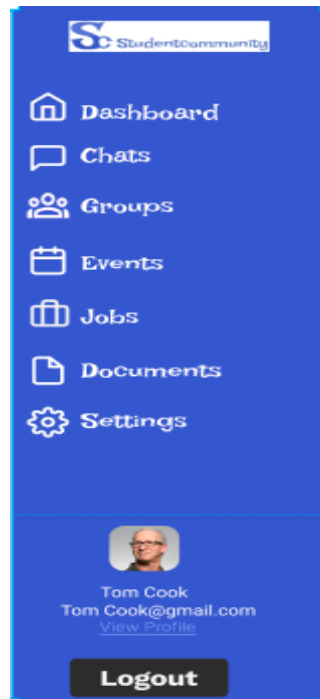


Abb. 12: Sidebar (Screenshot)

Die Sidebar ist das zentrale Navigationselement der App.

- **Zweck:** Ermöglicht schnellen Zugriff auf die Hauptfunktionen der App.
- **Design-Elemente:**
  - Klare Symbole und Labels für Seiten wie „Dashboard“, „Chat“, „Groups“, „Events“, „Jobs“, „Documents“ und „Settings“.
  - Ein responsives Design, das sich für Desktop und mobile Geräte eignet.
  - Anpassbare Shortcuts für häufig verwendete Seiten.

➤ **Logo:**



Abb. 13: Logo (Screenshot)

Das Logo ist das, was das Symbol der Anwendung repräsentiert.

Zusammenfassend lässt sich sagen, dass das User Interface Design der StudentCommunity-WebApp sorgfältig entwickelt wurde, um die Bedürfnisse der Nutzer zu erfüllen und eine intuitive Navigation zu ermöglichen. Jede Seite ist darauf ausgelegt, ihre jeweilige Funktion effizient zu erfüllen und dabei ästhetisch ansprechend zu sein. Im Anhang folgen die detaillierten Visualisierungen der Seiten, wie sie in Figma entworfen wurden.

## 5. Implementierung

Bei der Erstellung der StudentCommunity-WebApp ist die Umsetzung der entscheidende Schritt. In diesem Stadium erfolgt die Implementierung der zuvor konzipierten Konzepte, wie etwa der Datenbankstruktur und des User-Interface-Designs, in einsatzfähigen Code. Das Ziel besteht darin, die vorgesehenen Funktionen ohne Fehler zu implementieren, damit die Anwendung den Benutzern eine zuverlässige und benutzerfreundliche Plattform bietet.

Im Rahmen der Implementierung der StudentCommunity-WebApp sind verschiedene Kernbereiche von Bedeutung, um die Funktionalität und Benutzerfreundlichkeit der Anwendung zu gewährleisten. Die wichtigsten Aspekte, die in diesem Kapitel vorgestellt werden, sind:

- **Entwicklungsumgebung und Werkzeuge:** Verwendete Tools und Technologien für eine effiziente und reibungslose Entwicklung.
- **API-Design und Implementierung:** Entwicklung von APIs für eine nahtlose Kommunikation zwischen Frontend und Backend.
- **Authentifizierung und Autorisierung:** Implementierung sicherer Mechanismen zur Benutzerverwaltung und Zugriffskontrolle.
- **Komponentenstruktur:** Modulare Gestaltung der Benutzeroberfläche für Wartungsfreundlichkeit und Erweiterbarkeit.
- **Integration mit dem Backend:** Verbindung der React-Komponenten mit den Backend-APIs für dynamische Inhalte.

## 5.1. Entwicklungsumgebung und Werkzeuge

Die Softwareentwicklung basiert auf der Entwicklungsumgebung und den verwendeten Tools. Um die StudentCommunity-WebApp zu entwickeln, wurden sorgfältig Technologien und Tools verwendet, die die Entwicklung des Backends und des Frontends gleichermaßen unterstützen.

### Hauptkomponenten der Entwicklungsumgebung

#### ➤ Docker:

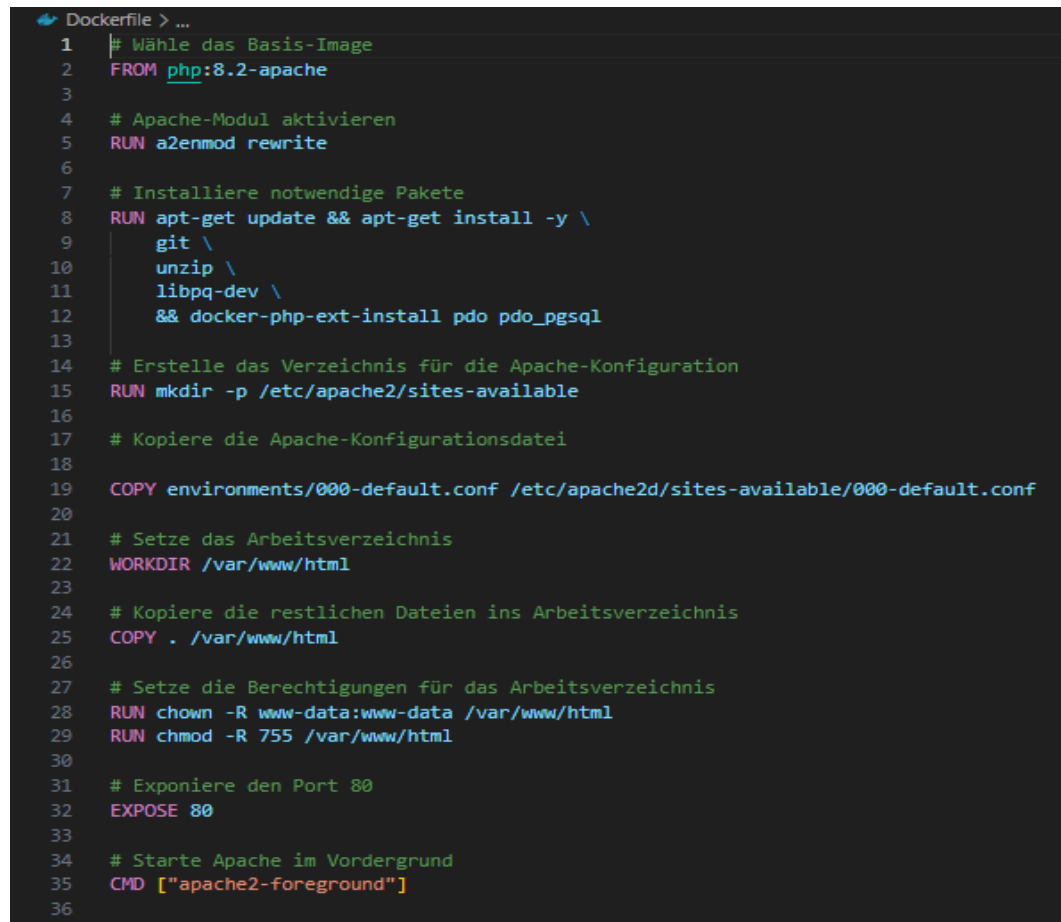
Um die Anwendung zu containerisieren und eine einheitliche Umgebung für Entwicklung, Test und Deployment zu gewährleisten, wurde Docker eingesetzt. Dadurch konnten das Backend, Frontend und Datenbank parallel ausgeführt werden. Es ist auch wichtig zu wissen wie Docker in dem Projekt definiert und eingesetzt wurde. Docker wurde in diesem Projekt durch die Verwendung einer `docker-compose.yml` und einer `Dockerfile` Datei, die als nächste mithilfe des Aussehens der Strukturen erläutert werden.

Zuerst ist die `docker-compose.yml` Datei, die in der nächsten Seite als folgendem aussieht:

```
docker-compose.yml
1 | version: '4.36'
2 | services:
3 |   db:
4 |     image: postgres
5 |     container_name: postgres
6 |     volumes:
7 |       - ./postgresql/data:/var/lib/postgresql
8 |     ports:
9 |       - "5432:5432"
10 |     environment:
11 |       POSTGRES_DB: student_community_db
12 |       POSTGRES_USER: studentcommunity
13 |       POSTGRES_PASSWORD: studentcommunity
14 |     networks:
15 |       - student_community
16 |   pgadmin:
17 |     image: dpage/pgadmin4
18 |     container_name: pgAdmin
19 |     ports:
20 |       - "5050:80"
21 |     depends_on:
22 |       - db
23 |     environment:
24 |       PGADMIN_DEFAULT_EMAIL: mendumomaxwellryan@yahoo.com
25 |       PGADMIN_DEFAULT_PASSWORD: studentcommunity
26 |     networks:
27 |       - student_community
28 |   apache:
29 |     build:
30 |       context: .
31 |       dockerfile: Dockerfile
32 |     container_name: student_community
33 |     ports:
34 |       - "8000:80"
35 |       - "443:443"
36 |     volumes:
37 |       - ./var/www/html/
38 |       - ./environments:/etc/apache2/sites-available
39 |     depends_on:
40 |       - db
41 |     restart: always
42 |     networks:
43 |       - student_community
44 |   networks:
45 |     student_community:
46 |       driver: bridge
47 |   volumes:
48 |     db_data:
```

Abb. 14: docker-compose.yml Datei (Screenshot)

Die Datei docker-compose.yml steuert die unterschiedlichen Projektcontainer. Der Datenbankcontainer mit PostgreSQL, der Webserver-Container mit Laravel und ein Container für phpPgAdmin, der die PostgreSQL-Datenbank verwaltet, sind unter anderem in ihr festgelegt. Die Datei gewährleistet die Kommunikation zwischen den Diensten über ein gemeinsames Netzwerk und reguliert die Abhängigkeiten, etwa den Start des Webserver erst nach Fertigstellung der Datenbank. Volumes persistent speichert die PostgreSQL-Daten, sodass sie auch beim Neustart des Containers erhalten bleiben.



```
Dockerfile > ...
1  # Wähle das Basis-Image
2  FROM php:8.2-apache
3
4  # Apache-Modul aktivieren
5  RUN a2enmod rewrite
6
7  # Installiere notwendige Pakete
8  RUN apt-get update && apt-get install -y \
9      git \
10     unzip \
11     libpq-dev \
12     && docker-php-ext-install pdo pdo_pgsql
13
14 # Erstelle das Verzeichnis für die Apache-Konfiguration
15 RUN mkdir -p /etc/apache2/sites-available
16
17 # Kopiere die Apache-Konfigurationsdatei
18
19 COPY environments/000-default.conf /etc/apache2d/sites-available/000-default.conf
20
21 # Setze das Arbeitsverzeichnis
22 WORKDIR /var/www/html
23
24 # Kopiere die restlichen Dateien ins Arbeitsverzeichnis
25 COPY . /var/www/html
26
27 # Setze die Berechtigungen für das Arbeitsverzeichnis
28 RUN chown -R www-data:www-data /var/www/html
29 RUN chmod -R 755 /var/www/html
30
31 # Exponiere den Port 80
32 EXPOSE 80
33
34 # Starte Apache im Vordergrund
35 CMD ["apache2-foreground"]
36
```

Abb. 15: Dockerfile Datei (Screenshot)

Die Apache-Webserverkonfiguration mit PHP 8.2, die für die Ausführung der Laravel-Anwendung erforderlich ist, ist in der Dockerfile festgelegt. Um sicherzustellen, dass die Interaktion mit der PostgreSQL-Datenbank funktioniert, werden darin wichtige PHP-Module wie `pdo_pgsql` eingebaut. Außerdem erfolgt die Festlegung von Arbeitsverzeichnissen, die Festlegung von Berechtigungen und die Vorbereitung der Umgebung für die Anwendung Laravel. Zur Verarbeitung von HTTP-Anfragen wird der Apache-Server eingerichtet. Zum Schluss wird der Server im Vordergrund gestartet.

Um sicherzustellen, dass die Funktionalität der Anwendung gewährleistet ist, werden in diesem Laravel-Projekt neben grundlegenden Docker-Befehlen wie `docker-compose up -d`, dass die Container im Hintergrund auf der Grundlage der Konfiguration in der `docker-compose.yml`-Datei starten, `docker-compose ps` die aktiven Container überprüft und `docker-compose down` sie stoppt. Durch die Verwendung von `docker-compose exec apache php artisan` ist es möglich, Laravel-Befehle direkt im Apache-Container auszuführen, was die Verwaltung der Anwendung erleichtert. Um sicherzustellen, dass neue Änderungen an der Konfiguration oder am Code korrekt übernommen werden, wird der Befehl `docker-compose exec apache php artisan cache:clear` zum Beispiel verwendet. Testdaten oder Stammdaten können mit `docker-compose exec apache php artisan db:seed` in die Datenbank eingetragen werden. Dies ist insbesondere in der Entwicklungs- und Testphase hilfreich. Um die Leistung zu optimieren, erzeugt der `docker-compose exec apache php artisan config:cache` Befehl einen Cache für die Konfigurationsdateien der Software. Außerdem ist es möglich, mithilfe von Frontend-Tools wie Laravel Mix Assets zu erstellen, zum Beispiel mithilfe des im Container ausgeführten Befehls `npm run dev`. Indem diese Befehle in der Docker-Umgebung verwendet werden, entsteht eine zuverlässige und wiederholbare Entwicklungsumgebung, die Versionskonflikte vermeidet und Entwicklern eine effiziente Projektarbeit ermöglicht.<sup>13</sup>

➤ **LaravelFramework:**

Laravel wurde als leistungsfähiges PHP-Framework genutzt, um das Backend und die API zu entwickeln. Eingebaute Lösungen für Routing, Authentifizierung und Datenbankoperationen sind vorhanden.

➤ **React:**

Für die Entwicklung einer dynamischen und benutzerfreundlichen Benutzeroberfläche wurde React als Frontend-Framework verwendet. Unterstützung durch den Einsatz wiederverwendbarer Bauteile und staatlicher Verwaltung.

---

<sup>13</sup> Helbert, D. (2021, 3. Juni). Laravel und Docker: Die perfekte Entwicklungsumgebung. Laravel Tutorials. <https://laravel.dirk-helbert.de/laravel-docker/>

➤ **PostgreSQL:**

Die relationale Datenbank wurde genutzt, um die Anwendungsdaten zu speichern und zu verwalten.

➤ **VisualStudioCode:**

Die wichtigste Umgebung für die Entwicklung von Frontend- und Backend-Anwendungen. Plugins für PHP, React, Docker und Datenbankintegration sind enthalten.

➤ **Figma:**

wird für die Gestaltung der Benutzeroberfläche und die Erstellung von Prototypen verwendet. Half bei der Visualisierung von Designentscheidungen und der effizienten Kommunikation.

➤ **GitLab:**

Um die Version zu kontrollieren und Task zu erstellen.

Die Aktionen von GitLab wurden in CI/CD-Pipelines eingesetzt. Die Task Tabelle wird als Screenshot im Anhang hinzugefügt.

➤ **Swagger:**

Es diene als Tool, um die APIs von der StudentCommunity-WebApp zu testen und zu dokumentieren. Das Aussehen des Swagger Dokumentierung wird im Anhang hinzugefügt.

## 5.2. Backend-Entwicklung mit Laravel

Die Entwicklung des Backends der StudentCommunity-WebApp wurde unter Verwendung des PHP-Frameworks Laravel durchgeführt. Laravel wurde aufgrund seiner Benutzerfreundlichkeit, der umfangreichen Bibliotheken und der Unterstützung moderner Entwicklungsmethoden ausgewählt. Der Schwerpunkt lag auf der Bereitstellung eines robusten, sicheren und gut organisierten Backends, das als Basis für die gesamte Anwendung dient. Bevor die Entwicklung des Backends der Webapp begann, wurde ein Klassendiagramm erstellt mit den verschiedenen Methoden und Attributen, die für die Implementierung besonders waren. Folgend zeigt sich das erstellte Klassendiagramm der StudentCommunity-Web-App.



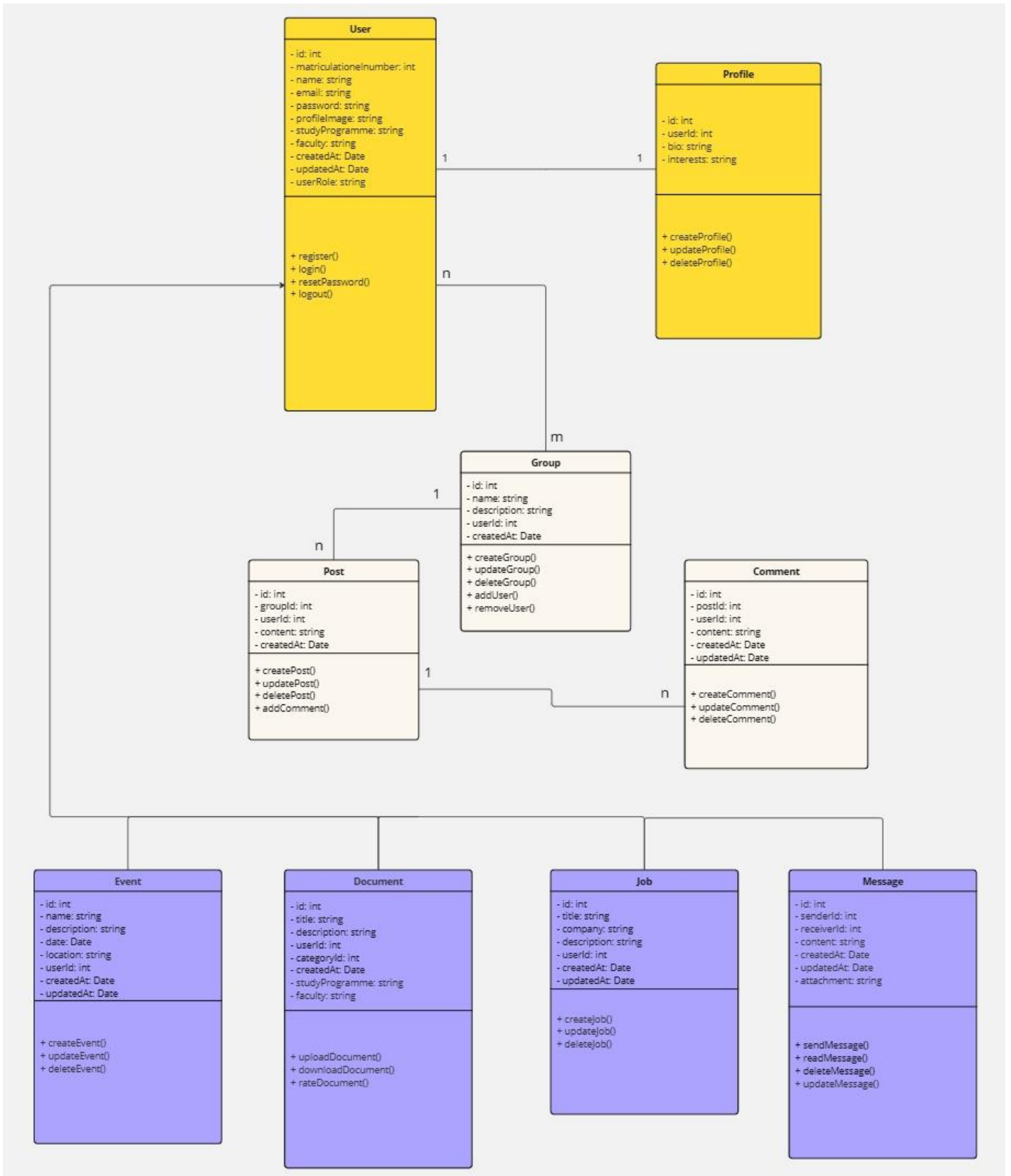


Abb. 16: Klassendiagramm der StudentCommunityWebApp (eigene Erstellung)

Dieses Klassendiagramm erläutert die Beziehungen zwischen den verschiedenen Klassen und auch die benötigten Methoden für die Implementierung der Software.

Es ist aber wichtig zu wissen, wie man ein Projekt in Laravel erstellt und was man dafür braucht. Um ein Laravel Projekt zu erstellen, sollen PHP, Composer und Laravel Installer schon im lokalen Computer installiert sein. Außerdem müssen auch NPM und Node installiert, um die Web App-Frontend-Assets auszuführen. Durch den Composer: `composer global require laravel/installer`, kann der Laravel installer installiert werden. Wenn das erfolgreich gemacht ist, könnte eine neue Laravel App erstellt werden. Das Befehl dafür ist: `laravel new example-app` und danach ist die Laravel's lokale Entwicklungsserver, Queue worker, und Vite Entwicklungsserver durch den `dev`. Composer Script :

```
cd example-app
```

```
npm install && npm run build
```

```
composer run dev
```

ausgeführt.

Nach der Ausführung ist die App im Web Browser <http://localhost:8000> verfügbar.<sup>14</sup>

Die nächsten Schritte werden dann die Konfiguration der Datenbank, Implementierung von den APIs und weitere Kernfunktion von Laravel. Diese werden im folgenden Punkten erklärt.

### 5.2.1. API-Design und Implementierung

Bei der Entwicklung der StudentCommunity-WebApp waren das API-Design und die Implementierung entscheidende Bestandteile der Backendentwicklung. Um eine problemlose Interaktion zwischen Frontend und Backend zu gewährleisten, wurden die APIs konzipiert. Eine deutliche Struktur, und Sicherheit und eine effiziente Datenübertragung standen dabei im Mittelpunkt. Die integrierten Funktionen und Bibliotheken von Laravel machen es zu einer idealen Basis für die Entwicklung zeitgemäßer RESTFUL-APIs.

---

<sup>14</sup> LaRavel - the PHP framework for web artisans. (o. D.). <https://laravel.com/docs/11.x/installation>

Zur Erstellung von APIs, bietet Laravel eine bestimmte Reihenfolge an. Es gilt als:

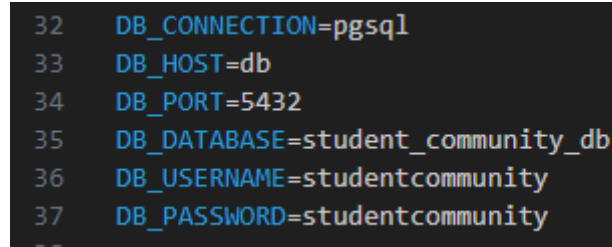
### Reihenfolge zur Erstellung von APIs der StudentCommunity-WebApp in Laravel:

#### ➤ Datenbankkonfiguration:

Anpassung der .env-Datei, um die Verbindung zur Datenbank herzustellen.

- Sicherstellen, dass die Zugangsdaten (Datenbankname, Benutzer, Passwort) korrekt sind.<sup>15</sup>

Die Verbindung zur Datenbank in der ENV-Datei sieht so aus:



```
32 DB_CONNECTION=pgsql
33 DB_HOST=db
34 DB_PORT=5432
35 DB_DATABASE=student_community_db
36 DB_USERNAME=studentcommunity
37 DB_PASSWORD=studentcommunity
```

Abb. 17: Verbindung zur Datenbank (Screenshot)

#### ➤ Erstellung der Modelle:

- **Definition von Modellen mit dem Befehl:**

```
php artisan make:model ModelName
```

- Modelle repräsentieren die Datenbanktabellen und deren Beziehungen.<sup>16</sup>

---

<sup>15</sup> LaRavel - the PHP framework for web artisans. (o. D.).

<https://laravel.com/docs/11.x/installation>

<sup>16</sup> LaRavel - the PHP framework for web artisans. (o. D.).

<https://laravel.com/docs/11.x/eloquent#generating-model-classes>

Beispiel von Model ist das User Model:

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;
use Tymon\JWTAuth\Contracts\JWTSubject;
use Illuminate\Contracts\Auth\MustVerifyEmail;

class User extends Authenticatable implements JWTSubject, MustVerifyEmail
{
    use HasFactory, Notifiable;

    protected $table = 'users';

    protected $fillable = [
        'matriculationNumber',
        'name',
        'email',
        'password',
        'phoneNumber',
        'address',
        'profile_picture',
        'studyProgramme',
        '...',
    ];
}
```

Abb. 18: User Model (Screenshot)

Beim Benutzermodell werden die verschiedenen Attribute und Beziehungen einfach mit den anderen Daten erstellt.

### ➤ Migrationen für die Datenbank

- Erstellung von Migrationsdateien mit dem Befehl:  
`php artisan make:migration create_user_table`
- In den Migrationsdateien werden die Tabellenstruktur und deren Spalten definiert.<sup>17</sup>

<sup>17</sup> LaRavel - the PHP framework for web artisans. (o. D.).  
<https://laravel.com/docs/11.x/migration>

Die Migrationsdatei der User wird als folgende angezeigt.

```
database > migrations > 0001_01_01_000000_create_users_table.php > ...
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10       * Run the migrations.
11       */
12     public function up(): void
13     {
14         Schema::create('users', function (Blueprint $table) {
15             $table->id();
16             $table->string('name');
17             $table->string('email')->unique();
18             $table->timestamp('email_verified_at')->nullable();
19             $table->string('password');
20             $table->rememberToken();
21             $table->timestamps();
22         });
23     }
```

Abb. 19: User Migration (Screenshot)

Die Migration User Tabelle zeigt alle Attribute an, die die App in der Datenbank benötigt. Z. B. user\_id, user\_name, user\_email.

Außer Tabellen bei der Migration zu erstellen, gibt es auch die Möglichkeiten Tabellen zu aktualisieren mit dem Befehl: `php artisan make:migration update_user_table`, um andere Attribute zu ändern oder zu modifizieren. Dann ist es möglich Attributen aus den Tabellen zu entfernen mit dem Befehl: `php artisan make:migration drop_user_table`. Durch die Drop-Methode könnten Attributen aus der Tabelle gelöscht werden. Um Attribute in der Tabelle hinzuzufügen, gibt man den Befehl: `php artisan make:migration add_user_table`.

Nachdem alle Tabellen fertig erstellt wurden, werden diese Tabellen in die Datenbank migriert mit dem Befehl: `docker-compose exec apache php artisan migrate`.

Unter ist das Aussehen der User\_Table in PostgreSQL dargestellt.

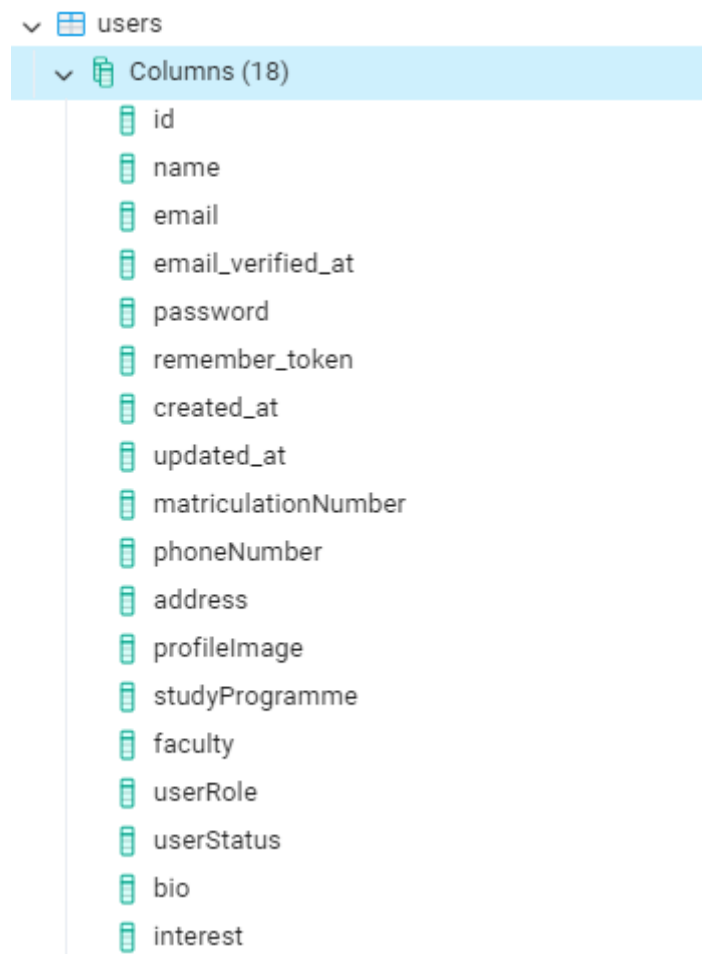


Abb. 20: UserTabelle in der Datenbank (Screenshot)

Das Bild zeigt alle Attribute der User, die in die Datenbank migriert wurden.

### ➤ Erstellung von Controllern

- Generieren von Controllern, die die API-Logik enthalten: `php artisan make:controller UserController`
- Implementierung der Methoden (z. B. index, store, update, delete) im Controller, um Datenbankoperationen durchzuführen.<sup>18</sup>

---

<sup>18</sup> LaRavel - the PHP framework for web artisans. (o. D.).  
<https://laravel.com/docs/11.x/controllers>

Der Logik für den Login API der StudentCommunity-WebApp wurde folgende implementiert:

```
app > Http > Controllers > UserController.php > ...
10 class UserController extends Controller
11 {
12     //
13     public $token = true;
14
15     public function login(Request $request)
16     {
17
18         $input = $request->only('email', 'password');
19         $jwt_token = null;
20
21         if (!$jwt_token = JWTAuth::attempt($input)) {
22             return response()->json([
23                 'success' => false,
24                 'message' => 'Invalid Email or Password',
25             ], Response::HTTP_UNAUTHORIZED);
26         }
27
28         $user = auth('api')->user();
29         if (!$request->user()->hasVerifiedEmail()) {
30             return response()->json([
31                 'success' => false,
32                 'message' => 'Please verify your email before logging in.',
33             ], Response::HTTP_FORBIDDEN);
34         }
35
36         return response()->json([
37             'success' => true,
```

Abb. 21: UserController (Screenshot)

Die API-Login-Logik authentifiziert einen Nutzer, indem man die übergebenen E-mail- und Password-Daten überprüft. Zunächst wird versucht, die Anmeldung mit `JWTAuth::attempt()` zu validieren; bei fehlerhaften Anmeldedaten wird eine 401 Unauthorized-Antwort zurückgegeben. Nach erfolgreicher Authentifizierung wird der eingeloggte Nutzer über `auth('api')->user()` abgerufen.

Anschließend wird geprüft, ob der Nutzer seine E-Mail-Adresse verifiziert hat (`hasVerifiedEmail()`); falls nicht, folgt eine 403 Forbidden-Antwort. Wenn alle Prüfungen erfolgreich sind, wird eine JSON-Antwort mit einem Bestätigungsstatus und optional dem Token zurückgegeben, um sicherzustellen, dass nur korrekt authentifizierte und verifizierte Nutzer Zugriff erhalten.

Außer der Login Logik gibt's noch weitere Logiken wie Logout, getUser und searchUser, und sie sind alle API-Methoden, die für die Verwendung der StudentCommunity-WebApp aufrufbar sind.

➤ **API-Routen definieren**

- Eintragen der API-Endpunkte in der Datei `routes/api.php`.
- Jeder Endpunkt wird einer Controller-Methode zugeordnet, z. B.:

```
Route::get('login', [UserController::class, 'login'])
    ->name('login');
```

Abb. 22: Route Login API (Screenshot)

Die Route ist entweder in Swagger für das Testen des APIs oder bei dem Frontend aufrufbar.<sup>19</sup>

➤ **Middleware einrichten**

- Hinzufügen von Middleware für Sicherheits- und Zugriffssteuerungsmechanismen.
- Beispiele:
  - Authentifizierung mit JWT oder Passport.
  - Validierung von Eingaben oder Cross-Origin Resource Sharing (CORS).
- Middleware wird den Routen zugeordnet<sup>20</sup>, z. B.:

```
19
20 Route::middleware('guest')->group(function () {
21     Route::get('login', [UserController::class, 'login'])
22         ->name('login');
23 });
```

Abb. 23: Middleware der Login Route (Screenshot)

Die Implementierung von Controllern und Routen, die Implementierung von Middleware, die Konfiguration der Datenbank, die Definition von Modellen und Migrationen sowie die Erstellung von APIs in Laravel sind klare und etablierte

---

<sup>19</sup> LaRavel - the PHP framework for web artisans. (o. D.).  
<https://laravel.com/docs/11.x/Routing>

<sup>20</sup> LaRavel - the PHP framework for web artisans. (o. D.).  
<https://laravel.com/docs/11.x/middleware>



Schritte. Diese strukturierte Herangehensweise sorgt dafür, dass das Backend der StudentCommunity-WebApp robust und wartbar ist.

### 5.2.2. Authentifizierung und Autorisierung

Bei der Erstellung zeitgemäßer Webanwendungen ist die Einführung von Authentifizierungs- und Autorisierungsfunktionen von zentraler Bedeutung. Laravel stellt ein ausführliches, nutzerfreundliches Framework zur Verfügung, mit dem Entwickler diese wichtigen Sicherheitsmechanismen rasch, sicher und effizient integrieren können.

Das Hauptprinzip des Authentifizierungssystems von Laravel sind Guards und Providers, die sich in der `config/auth.php` Datei befinden. Die Authentifizierung von Benutzern für jede Anfrage wird von Guards festgelegt – zum Beispiel durch Sitzungsdaten oder API-Token. Im Gegensatz dazu legen Providers fest, wie Benutzer aus anderen Speichern oder der Datenbank abgerufen werden. Laravel ermöglicht aufgrund dieser deutlichen Abgrenzung von Authentifizierung und Datenverwaltung eine maximale Flexibilität für verschiedene Anwendungsbereiche.<sup>21</sup>

```
'guards' => [
    'web' => [
        'driver' => 'session',
        'provider' => 'users',
    ],

    'api' => [
        'driver' => 'jwt',
        'provider' => 'users',
    ],
],
```

Abb. 24: Guards (Screenshot)

```
'providers' => [
    'users' => [
        'driver' => 'eloquent',
        'model' => env('AUTH_MODEL', App\Models\User::class),
    ],

    // 'users' => [
    //     'driver' => 'database',
    //     'table' => 'users',
    // ],
],
```

Abb. 25: ProvidersLaravel (Screenshot)

Laravel bietet verschiedene Starter-Kits, darunter Laravel Breeze, Laravel Jetstream und Laravel Fortify, um die Entwicklungszeit zu reduzieren. Diese stellen bereits entwickelte Authentifizierungslösungen zur Verfügung, die als Basis für eigene Anwendungen verwendet oder modifiziert werden können.<sup>22</sup>

<sup>21</sup> LaRavel - the PHP framework for web artisans. (o. D.-b).

<https://laravel.com/docs/11.x/authentication#introduction>

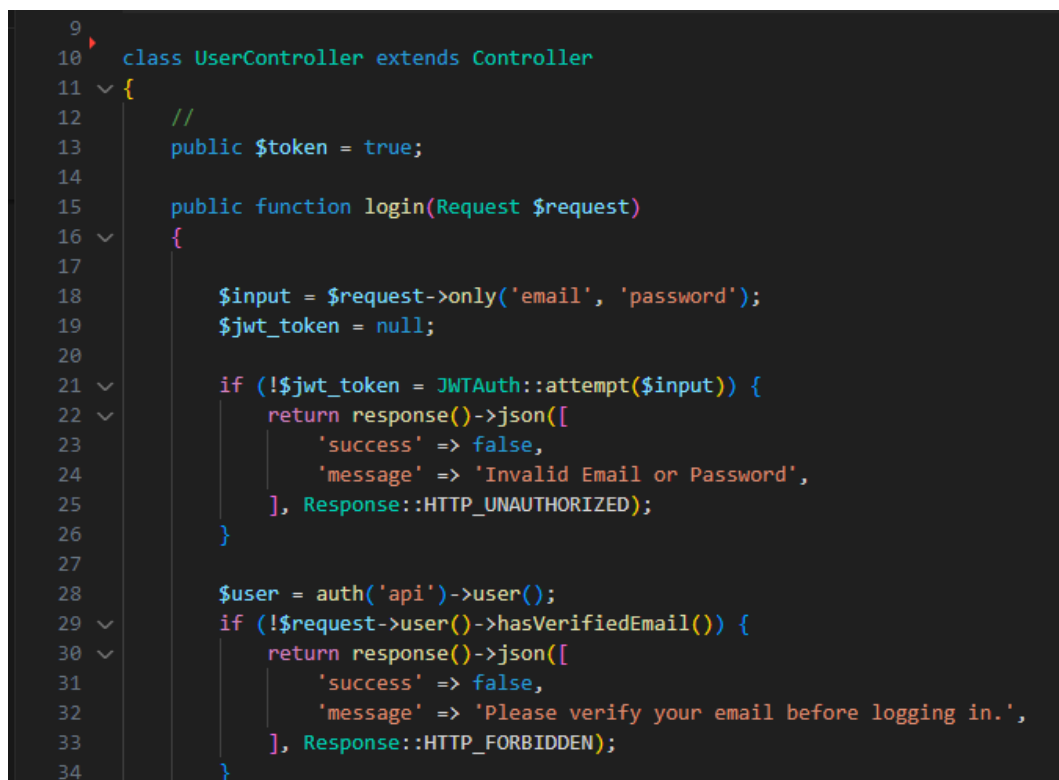
<sup>22</sup> LaRavel - the PHP framework for web artisans. (o. D.-c).

<https://laravel.com/docs/11.x/authentication#ecosystem-overview>

Für die Implementierung der StudentCommunity-WebApp, wurde Laravel Breeze verwendet.

Laravel nutzt eingebaute Dienste wie Session und Auth für Anwendungen mit Browser-basierten Sitzungen, die eine Cookie-basierte Authentifizierung ermöglichen. Laravel stellt jedoch leistungsstarke Pakete wie Sanctum und Passport für API-Anfragen zur Verfügung. Sanctum ist dabei häufig die bevorzugte Option, da es eine schlanke und vielseitige Lösung für Single-Page-Anwendungen (SPAs), mobile Apps und herkömmliche Webanwendungen ist.<sup>23</sup>

In der StudentCommunity-WebApp gibt's eine Authentifizierungsphase während des Einloggens des Benutzers. Mithilfe des folgenden Bildes wird erklärt, wie ein User beim Einloggen authentifiziert wird.



```
9
10 class UserController extends Controller
11 {
12     //
13     public $token = true;
14
15     public function login(Request $request)
16     {
17
18         $input = $request->only('email', 'password');
19         $jwt_token = null;
20
21         if (!$jwt_token = JWTAuth::attempt($input)) {
22             return response()->json([
23                 'success' => false,
24                 'message' => 'Invalid Email or Password',
25             ], Response::HTTP_UNAUTHORIZED);
26         }
27
28         $user = auth('api')->user();
29         if (!$request->user()->hasVerifiedEmail()) {
30             return response()->json([
31                 'success' => false,
32                 'message' => 'Please verify your email before logging in.',
33             ], Response::HTTP_FORBIDDEN);
34         }
35     }
36 }
```

Abb. 26: Login Methode (Screenshot)

Das Bild zeigt den Code einer Login-Funktion, die im User-Controller implementiert ist. Die Funktion ist für die Authentifizierungsphase in der

<sup>23</sup> LaRavel - the PHP framework for web artisans. (o. D.-d).  
<https://laravel.com/docs/11.x/authentication#laravels-api-authentication-services>

StudentCommunity-WebApp verantwortlich. Es lassen sich folgende Schritte und Sicherheitsüberprüfungen erkennen:

➤ **Empfangen von Eingaben**

Die Funktion empfängt über das `$request-Objekt` die Eingaben der Benutzer, nämlich E-Mail und PASSWORD, und speichert diese in der Variablen `$input`. Dies stellt sicher, dass nur die relevanten Anmeldeinformationen verarbeitet werden.

➤ **JWT-basierte Authentifizierung**

Mithilfe von `JWTAuth::attempt($input)` wird überprüft, ob die eingegebenen E-Mail- und Passwort-Kombinationen mit den Daten in der Datenbank übereinstimmen. Falls die Authentifizierung fehlschlägt, wird eine JSON-Antwort zurückgegeben, die angibt, dass die Anmeldedaten ungültig sind (HTTP-Statuscode 401 Unauthorized).

➤ **Benutzerüberprüfung**

Nachdem der Benutzer erfolgreich authentifiziert wurde, wird sein Datensatz über `auth('api')->user()` abgerufen. Anschließend wird überprüft, ob der Benutzer seine E-Mail-Adresse verifiziert hat. Diese Sicherheitsmaßnahme ist entscheidend, um sicherzustellen, dass nur verifizierte Benutzer auf die Plattform zugreifen können.

➤ **Fehlende E-Mail-Verifizierung**

Falls die E-Mail-Adresse des Benutzers nicht verifiziert ist (`!$request->user()->hasVerifiedEmail()`), wird eine weitere JSON-Antwort zurückgegeben, die den Benutzer auffordert, seine E-Mail zu verifizieren. In diesem Fall wird der HTTP-Statuscode 403 Forbidden zurückgegeben.

Laravel bietet neben diesen Authentifizierungsdiensten auch eine anpassungsfähige Autorisierungslogik mithilfe von Gates und Policies. Diese ermöglichen eine detaillierte Verwaltung von Benutzerrechten und Rollen. Auf diese Weise wird gewährleistet, dass die Nutzer nicht nur authentifiziert, sondern

auch befugt sind, in der Anwendung bestimmte Aktionen oder Ressourcen auszuführen.

Laravel stellt aufgrund dieser umfangreichen, aber leicht umzusetzenden Mechanismen eine zuverlässige Basis für den Aufbau von sicheren und zeitgemäßen Webanwendungen bereit.

### **5.3. Frontend-Entwicklung mit React**

Die Umsetzung dieser Anwendung im Frontend erfolgte mit dem React-Framework, welches durch seine Effizienz, Modularität und Wiederverwendbarkeit von Komponenten gekennzeichnet ist. Durch die Bereitstellung einer komponentenbasierten Architektur, die die Wartbarkeit und Erweiterbarkeit des Codes unterstützt, kann React interaktive Benutzeroberflächen entwickeln.

Dieses Kapitel beschreibt die Struktur und Organisation der Interface-Elemente der Anwendung, um eine intuitive und benutzerfreundliche Interaktion sicherzustellen. Zuerst wird die Struktur der Komponenten erläutert, die die Hierarchie und den Aufbau der verschiedenen React-Komponenten in der Anwendung repräsentieren. Es folgt eine Erklärung zur Integration in das Backend, wobei der Schwerpunkt auf der Interaktion zwischen dem Frontend und der Laravel-gestützten API liegt. Dies beinhaltet die Nutzung von HTTP-Anfragen für die Übertragung von Daten sowie die Steuerung des Zustands der Anwendung.

#### **5.3.1. Komponentenstruktur**

Die Struktur des Frontends der Anwendung wurde so gestaltet, dass eine deutliche Abgrenzung der Verantwortlichkeiten und die Wiederverwendbarkeit der Bestandteile möglich sind. Der Schwerpunkt liegt darauf, dass die Komponenten modular sind, was die Entwicklung sowie die Wartung erleichtern wird. Nachfolgend wird die Struktur der Anwendung über einen Screenshot dargestellt.

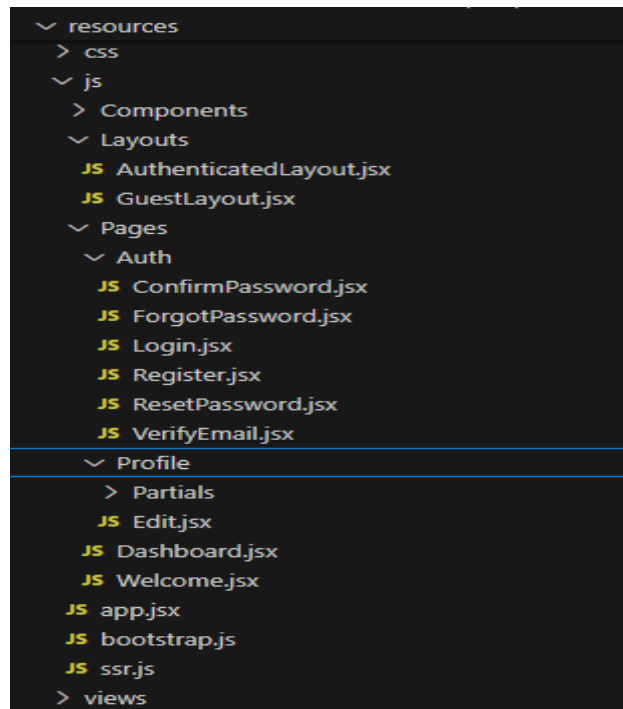


Abb. 27: Komponentenstruktur (Screenshot)

Das dargestellte Bild zeigt, wie die Dateien und Ordner strukturiert wurden. Unter anderem wird diese Strukturierung im Detail analysiert und erklärt.

### Analyse der Struktur:

#### ➤ Components:

- Dieser Ordner enthält generische, wiederverwendbare UI-Komponenten, die in verschiedene Seiten eingebunden werden. Beispiele sind PrimaryButton, ApplicationLogo, NavLink oder TextInput. Diese Komponenten fördern Konsistenz und Wiederverwendbarkeit. Für das Beispiel ApplicationLogo, sieht die Implementierung als folgende aus:

```
1  export default function ApplicationLogo(props) {  
2      return (  
3            
9      );  
10 }  
11
```

Abb. 28: ApplicationLogo Komponente (Screenshot)

Diese Komponente wird später im Layout verwendet als eine der wichtigen Komponenten für verschiedene Seiten der Anwendung. Z. B. sind die Register Page, die Login Page und das Dashboard, wo das Application-Logo aufgerufen wird. In den weiteren Punkten ist dieses Verfahren besser angezeigt.

➤ **Styles:**

- Im Ordner **CSS\_** befinden sich die Stile für die Anwendung. Diese können entweder als globale CSS-Dateien oder in Kombination mit CSS-in-JS-Bibliotheken verwendet werden.

➤ **Layout:**

- Die Layouts enthalten die grundlegenden Rahmenwerke der Seiten. Es gibt zwei Hauptlayouts:
  - **Authenticated-Layout.jsx:** wird für Seiten verwendet, die nur für angemeldete Benutzer zugänglich sind (z. B. *Dashboard*, *Profile*).
  - **Guest-Layout.jsx:** wird für Seiten genutzt, die für Gäste (nicht angemeldete Benutzer) zugänglich sind, wie z. B. *Login* und *Register*.
- Diese Layouts gliedern das Grundgerüst jeder Seite, einschließlich Kopfzeilen, Navigation oder Footer.

- Diese Layouts enthalten zahlreiche verschiedene Elemente wie das Anwendungslogo und werden aus dem Ordner Komponenten importiert.

```
resources > js > Layouts > JS AuthenticatedLayout.jsx > Authenticated
1 import ApplicationLogo from '@Components/ApplicationLogo';
2 import Dropdown from '@Components/Dropdown';
3 import NavLink from '@Components/NavLink';
4 import ResponsiveNavLink from '@Components/ResponsiveNavLink';
5 import { Link, usePage } from '@inertiajs/react';
6 import { useState } from 'react';
7
8 export default function Authenticated({ header, children }) {
9   const user = usePage().props.auth.user;
10
11   const [showingNavigationDropdown, setShowingNavigationDropdown] =
12     useState(false);
13
14   return (
15     <div className="min-h-screen bg-gray-100">
16       <nav className="border-b border-gray-100 bg-white">
17         <div className="mx-auto max-w-7xl px-4 sm:px-6 lg:px-8">
18           <div className="flex h-16 justify-between">
19             <div className="flex">
20               <div className="flex shrink-0 items-center">
21                 <Link href="/" />
22                 <ApplicationLogo className="block h-9 w-auto fill-current text-gray-800" />
23               </div>
24             </div>

```

Abb. 29: AuthenticatedLayout (Screenshot)

#### ➤ Pages:

- **Auth:** Der Ordner enthält alle Authentifizierungsseiten, wie:
  - *Login.jsx* und *Register.jsx* für die Anmeldung und Registrierung.
  - *VerifyEmail.jsx*, *ForgotPassword.jsx*, *ResetPassword.jsx* und *ConfirmPassword.jsx* für Funktionen wie Passwort-Wiederherstellung und E-Mail-Verifizierung.
  - Alle Seiten im Auth-Ordner werden immer im Guest-Layout eingesetzt. Es wird im nächsten Bild die Login Seite angezeigt.

```

1 import Checkbox from '@Components/Checkbox';
2 import InputError from '@Components/InputError';
3 import InputLabel from '@Components/InputLabel';
4 import PrimaryButton from '@Components/PrimaryButton';
5 import TextInput from '@Components/TextInput';
6 import GuestLayout from '@Layouts/GuestLayout';
7 import { Head, Link, useForm } from '@inertiajs/react';
8
9 export default function Login({ status, canResetPassword }) {
10 >   const { data, setData, post, processing, errors, reset } = useForm({ ...
14     });
15
16 >   const submit = (e) => { ...
21     };
22
23   return (
24 >     <GuestLayout> ...
114     </GuestLayout>
115   );
116 }
117

```

Abb. 30: Einsetzung des GuestLayout in der Login Seite

- **Profile:**

- Enthält Seiten wie *Edit.jsx*, die es den Benutzern ermöglichen, ihre Profildaten zu bearbeiten.
- Der Unterordner **Partials** ist für kleinere profilebezogene Sub-Komponenten reserviert, die wiederverwendet werden können wie *DeleteUserForm.jsx*, *UpdatePasswordForm.jsx* und *UpdateProfileInformationForm.jsx*.
- Zusätzliche Seiten wie *Dashboard.jsx* und *Welcome.jsx* repräsentieren zentrale Bereiche der Anwendung.

Die Dashboard Seite enthält auch ein Layout, aber das Authenticated-Layout, weil es geht, um Benutzer, die schon authentisiert sind.

- **Globale Dateien:**

- **app.jsx:** Diese Datei dient als Einstiegspunkt für die React-Anwendung und enthält die Hauptkonfiguration, wie Routen.
- **Bootstrap.js:** Dient zur Initialisierung und Konfiguration von globalen Ressourcen oder Bibliotheken.



- **ssr.js:** Kann für serverseitiges Rendern (SSR) oder spezifische Backend-Integrationen genutzt werden.

➤ **Komponentenorganisation:**

Die Strukturierung ermöglicht eine klare Trennung nach Verantwortlichkeiten:

- **Layouts:** Wiederverwendbare Rahmengerüste für verschiedene Benutzergruppen.
- **Pages:** Spezifische Seitenansichten für verschiedene Funktionen.
- **Components:** Generische Bausteine für konsistente UI-Elemente.
- **Auth & Profile:** Funktionale Gruppierung, die thematisch zusammenhängende Seiten und Sub-Komponenten organisiert.

➤ **Vorteile dieser Struktur:**

- **Klarheit:** Jede Komponente hat eine spezifische Aufgabe, was die Codebasis verständlicher macht.
- **Wiederverwendbarkeit:** Durch die Trennung in **Components** können UI-Bausteine mehrfach genutzt werden.
- **Erweiterbarkeit:** Neue Features können leicht hinzugefügt werden, ohne die bestehende Struktur zu beeinträchtigen.

### 5.3.2. Integration mit dem Backend

Die Einbindung von Laravel, React und Inertia in diese Anwendung erfolgte durch die Verknüpfung von Frontend und Backend. Diese Technologien sorgen für eine reibungslose Interaktion und eine zeitgemäße Single-Page-Anwendungsstruktur. Das Laravel-Breeze-Starterkit wurde genutzt, um die grundlegenden Funktionen rasch und effizient umzusetzen.

➤ **Laravel Breeze und Inertia**

- **Inertia** fungiert als Verbindungsstück zwischen React und Laravel (Backend). Es erlaubt die direkte Übermittlung von serverseitig generierten Daten an

- React-Komponenten, ohne auf herkömmliche API-Endpunkte und JSON-Responses angewiesen zu sein.<sup>24</sup>
- **Laravel Breeze** stellt eine unkomplizierte und kompakte Option dar, um die elementaren Funktionen des Benutzermanagements wie E-Mail-Verifizierung, Passwort-Reset und Registrierung zu integrieren. Der react-Stack mit Inertia wurde während der Installation ausgewählt, um React-Komponenten in einer Inertia-Architektur einzusetzen.<sup>25</sup>

### ➤ Installationsschritte für Laravel Breeze mit React

Nach der Erstellung eines neuen Laravel-Projekts wurden die folgenden Befehle verwendet, um Laravel Breeze und React mit Inertia zu installieren:

- **Neues Laravel-Projekt erstellen:**

```
Composer create-project laravel/laravel studentcommunity  
cd studentcommunity
```

- **Laravel Breeze installieren:**

```
composer require laravel/breeze --dev
```

- **Breeze-Installationsbefehle ausführen :**

```
php artisan breeze:install react
```

Hier wird das React-Stack mit Inertia konfiguriert. Dies installiert die grundlegenden Dateien und Konfigurationen für React und Laravel.

---

<sup>24</sup> Inertia.js - the modern monolith. (o. D.). <https://inertiajs.com/>

<sup>25</sup> Stanković, D. (2023, 23. November). Laravel Breeze: The Ultimate Installation Guide for Laravel Starter Kit's Authentication System. Medium.  
<https://medium.com/@stdejan/laravel-breeze-the-ultimate-installation-guide-for-laravel-starter-kits-authentication-system-a59ecf40dcf6>

- **Frontend-Abhängigkeiten installieren:**

```
npm install
npm run dev
```

- **Migrationen ausführen:**

```
php artisan migrate
```

Nach der erfolgreichen Ausführung dieser Befehle werden die Basisintegration zwischen Laravel und React eingerichtet, und die Benutzerverwaltung könnte direkt genutzt werden.

Laravel Sanctum wurde zur Authentifizierung genutzt und wurde speziell für SPAs konzipiert. Sanctum ermöglicht eine unkomplizierte Verknüpfung von API- und Cookie-basierter Authentifizierung. Die Laravel Sanctum ist automatisch bei der Installation eines Laravel Projekt erstellt. Es befindet sich in der Path des Projektes `config/sanctum.php`.<sup>26</sup>

➤ **Ständige Authentifizierung:** Sanctum nutzt Cookies zur Authentifizierung von Benutzern zwischen Front- und Backend. Dabei dürfen die Cookies ausschließlich auf zuverlässigen Domains (z. B. localhost:8000 für das React-Frontend) gesetzt werden.<sup>27</sup>

➤ **CSRF-Schutz:** Sanctum stellt den Cross-Site Request Forgery (CSRF)-Schutz zur Verfügung. Dies ist insbesondere bei der Arbeit mit Cookie-basierten Sitzungen von Bedeutung.<sup>28</sup>

➤ **Sanctum Guards:**

Die **Sanctum Guards** bestimmen, welche Authentifizierungsmechanismen Laravel Sanctum verwenden soll, um Benutzeranfragen zu authentifizieren. Für diese Anwendung wurde ‚web‘ als Guard eingesetzt.

---

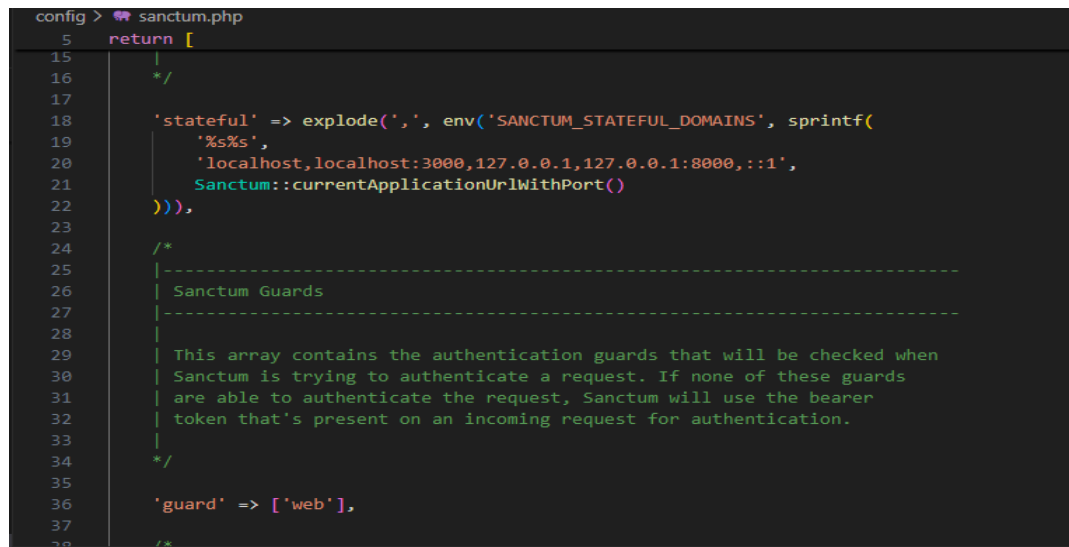
<sup>26</sup> LaRavel - the PHP framework for web artisans. (o. D.-b).  
<https://laravel.com/docs/11.x/sanctum#main-content>

<sup>27</sup> LaRavel - the PHP framework for web artisans. (o. D.-b).  
<https://laravel.com/docs/11.x/sanctum#spa-authentication>

<sup>28</sup> LaRavel - the PHP framework for web artisans. (o. D.-b).  
<https://laravel.com/docs/11.x/sanctum#csrf-protection>

- **Guard 'web':**

- Wird für **sessionbasierte Authentifizierung** verwendet. Ideal für SPAs (Single-Page Applications) wie React, die Cookies nutzen, um die Benutzeranmeldung und -sitzung zu verwalten.
- Sanctum überprüft, ob eine gültige **Session** existiert. Falls nicht, wird automatisch ein Bearer-Token für die Authentifizierung herangezogen.



```

config > sanctum.php
5      return [
15     |
16     | */
17     |
18     | 'stateful' => explode(',', env('SANCTUM_STATEFUL_DOMAINS', sprintf(
19     | '%s%s',
20     | 'localhost,localhost:3000,127.0.0.1,127.0.0.1:8000,::1',
21     | Sanctum::currentApplicationUrlWithPort()
22     | )),
23     |
24     | /*
25     | |-----
26     | | Sanctum Guards
27     | |-----
28     | |
29     | | This array contains the authentication guards that will be checked when
30     | | Sanctum is trying to authenticate a request. If none of these guards
31     | | are able to authenticate the request, Sanctum will use the bearer
32     | | token that's present on an incoming request for authentication.
33     | |
34     | | */
35     |
36     | 'guard' => ['web'],
37     |
38     | /*

```

Abb. 31: Sanctum Datei Konfiguration (Screenshot)

➤ **Anwendung bei der Integration:**

- **Frontend (React):** Sendet Anfragen, die automatisch Cookies (z. B. für CSRF-Schutz) beinhalten.

**Backend (Laravel):** Verwendet den Web-Guard, um Benutzer über Sessions zu identifizieren und Anfragen zu verarbeiten. Falls erforderlich, greift Sanctum auf tokenbasierte Authentifizierung zurück. Die Integration konnte mithilfe der Verknüpfung von Laravel Breeze, React und Inertia rasch und effektiv realisiert werden. Sanctum gewährleistete die Sicherheit und Benutzerfreundlichkeit der Authentifizierung, während Inertia die Interaktion zwischen Backend und Frontend vereinfachte. Diese Interaktion verleiht der gesamten Architektur Flexibilität und die Möglichkeit zur Erweiterung.

## 5.4. Wichtige Funktionalitäten

### 5.4.1. Benutzerregistrierung und Login

Zentrale Funktionen der Anwendung sind die Registrierung der Benutzer und die Anmeldung, die mithilfe von Laravel-Controllern und React-Komponenten realisiert wurden. Diese Features sorgen dafür, dass sich Nutzer registrieren und anmelden können, während die Authentifizierung über Laravel Sanctum sicher durchgeführt wird.

#### ➤ Benutzerregistrierung

Registrierung des Benutzers Die Registrierung wird durch den `RegisteredUserController` Controller vorgenommen. Die E-Mail-Adresse wird verwendet, um Benutzerdaten zu validieren, zu speichern und automatisch Rollen (z. B. Student oder Dozent) zuzuweisen. Der Registrierungsprozess wird durch eine automatische Anmeldung und die Übermittlung einer E-Mail-Verifikation abgeschlossen.

Weiter folgt wie die Logik der Registrierung sich im Backend als Code präsentiert.

- **Validierung:** Um sicherzustellen, dass nur genaue und vollständige Daten gespeichert werden, werden sämtliche Eingaben gründlich überprüft.

```

38 public function store(Request $request)
39 {
40     $validator = Validator::make($request->all(), [
41         'matriculationNumber' => 'required|max:6|unique:users,matriculationNumber',
42         'name' => 'required|string|max:255',
43         'email' => 'required|email|unique:users,email',
44         'password' => 'required|string|min:8|confirmed',
45         'phoneNumber' => 'nullable|numeric',
46         'profileImage' => 'nullable|image',
47         'studyProgramme' => 'required|string',
48         'faculty' => 'required|string',
49     ]);
50
51     if ($validator->fails()) {
52         return redirect()->back()->withErrors($validator)->withInput();
53     }
54

```

Abb. 32: Validierungslogik für die Registrierung (Screenshot)

- **Rollenlogik:** Eine Überprüfung der E-Mail-Adresse bestimmt, welche Rolle der Benutzer hat. Nutzer mit einer @stud-Adresse werden als Student bezeichnet, während andere als Dozent bezeichnet werden.

```

if ($validator->fails()) {
    return redirect()->back()->withErrors($validator)->withInput();
}

$user = new User();
$user->matriculationNumber = $request->input('matriculationNumber');
$user->name = $request->input('name');
$user->email = $request->input('email');
$user->password = Hash::make($request->password);
$user->phoneNumber = $request->input('phoneNumber');
$user->studyProgramme = $request->input('studyProgramme');
$user->faculty = $request->input('faculty');

if (strpos($user->email, '@stud') !== false) {
    $user->userRole = 'student';
} else {
    $user->userRole = 'dozent';
}

$user->save();

Auth::login($user);

```

Abb. 33: UserRole Logik (Screenshot)

- **E-Mail-Verifikation:** Nachdem die Registrierung erfolgreich abgeschlossen ist, wird an die angegebene E-Mail-Adresse eine Verifikationsnachricht gesendet, durch die Funktion:

```
$user->sendEmailVerificationNotification();
```

Die zugehörige React-Komponente `Register.jsx` bietet eine benutzerfreundliche Oberfläche zur Eingabe der Registrierungsdaten. Sie validiert die Eingaben direkt im Browser und sendet die Daten bei erfolgreicher Validierung an die Store-Methode des Controllers.

### ➤ Benutzerlogin

Der `AuthenticatedSessionController` führt die Umsetzung der Benutzerlogins aus. Die Store-Methode prüft die Daten des Benutzers, führt eine neue Sitzung ein und führt den Benutzer auf die Dashboard-Seite.

#### Codeausschnitt – Login:

```
27      /**
28       * Handle an incoming authentication request.
29       */
30      public function store(LoginRequest $request): RedirectResponse
31      {
32          $request->authenticate();
33
34          $request->session()->regenerate();
35
36          return redirect()->intended(route('dashboard', absolute: false));
37      }
38  }
```

Abb. 34: Login Logik (Screenshot)

- **Validierung:** Zur Validierung werden die Eingabedaten auf ihre Genauigkeit überprüft.
- **Session-Management:** Es erfolgen die Erstellung einer neuen Session und die Ungültigkeit vorhandener Sessions.
- **CSRF-Schutz:** Laravel Sanctum bietet eine automatische Verwaltung von CSRF-Tokens, um Anfragen zu schützen.

Außer den Logiken in den `RegisteredUserController` und `AuthenticatedSessionController`, gibt's auch wichtige Methoden in dem Register und Login Seiten mit React und diese werden auch als nächstes erklärt.

➤ **Anmeldeformular (Login.jsx):**

- **HandleSubmit:** Wenn der Anwender das Formular absendet, tritt es auf. Überprüft die Eingabedaten und schickt eine Anfrage an den Server (in der Regel über die Authentifizierungsstrecke).
- **SetData:** Die Formfelder werden auf der Grundlage der Benutzereingaben aktualisiert.
- **Fehlerbehandlung:** Durch den Einsatz von Errors aus dem Formularzustand werden Fehler wie ungültige Passwörter oder fehlende Felder angezeigt.

➤ **Registrierungsformular (register.jsx):**

- Bitte übermitteln Sie die Benutzerdaten an die Serverroute, um ein neues Konto zu erstellen.
- **Password\_confirmation:** Ein besonderes Feld, um sicherzustellen, dass das Passwort des Benutzers ordnungsgemäß wiederholt wird.

➤ **TextInput-Komponente:**

- Zur Erstellung konsistenter Eingabefelder wird sie eingesetzt.
- **Eigenschaften:**
  - **Wert:** Ermöglicht eine Datenbindung.
  - **OnChange:** Reaktion auf die Eingabe von Benutzern.
  - **isFocused:** Stellt das Feld zum Beispiel für das erste Eingabefeld automatisch in den Fokus.
  - **Vorteile:**
    - Modular aufgebaut und in unterschiedlichen Formen wiederverwendbar.
    - Unterstützt Fehlermeldungen und spezielle Layoutwünsche.



➤ **Layoutbezogene Aspekte:**

- **Einsatz von Layoutkomponenten:** Die Layoutkomponente (z. B. `AuthenticatedLayout`) gewährleistet eine gleichbleibende Struktur, die Kopfzeilen, Fußzeilen und Abstände umfasst.
- **Design:** Moderne Benutzeroberfläche wird durch responsive Elemente mit Klassen aus Tailwind CSS sichergestellt.
- **Modal-Komponenten:** Diese werden genutzt, um den Prozess benutzerfreundlicher zu machen, etwa bei Aktionen wie der Benutzerlöschung.

Alle genannten und erklärten Punkte sind wichtige Punkte, die für den Aufbau des User Interface der StudentCommunity-WebApp verwendet und eingesetzt wurden.

#### **5.4.2. Profilerstellung und -verwaltung**

Die Profilverwaltung ist ein zentraler Bestandteil der Anwendung der StudentCommunity-WebApp und ermöglicht es Benutzern, ihre persönlichen Daten zu aktualisieren, ihr Passwort zu ändern oder ihr Konto zu löschen. In diesem System wird eine klare Trennung zwischen Backend-Logik und Frontend-Komponenten genutzt, um eine flexible und sichere Benutzererfahrung zu gewährleisten.

Das Backend, implementiert in Laravel, stellt die Funktionen bereit, um Benutzerprofile zu bearbeiten, zu aktualisieren oder zu löschen. Diese Funktionen sind über die Profile-Controller-Methoden `edit`, `update` und `destroy` realisiert. Gleichzeitig bietet das Frontend, basierend auf React und Inertia.js, benutzerfreundliche und reaktionsfähige Komponenten, die den Profilverwaltungsprozess intuitiv gestalten

Die Edit-Funktion wird verwendet, um das Profilbearbeitungsformular zu laden und die Benutzerdaten zu initialisieren. Dabei werden die E-Mail-Verifizierungsanforderungen mithilfe von Inertia-Props überprüft. Nutzerdaten wie Name und E-Mail werden von der Update-Funktion aktualisiert. Sie werden in der Datenbank gespeichert und die Verifizierung wird bei Änderungen der E-Mail zurückgesetzt. Das Passwort wird von der Destroy-Funktion validiert, das

Benutzerkonto wird gelöscht, der Benutzer wird automatisch ausgeloggt und der Token wird gelöscht.

Diese Methoden werden als Coden wie folgt angezeigt:

```
14 class ProfileController extends Controller
15
16 /**
17  * Display the user's profile form.
18  */
19 public function edit(Request $request): Response
20 {
21     return Inertia::render('Profile/Edit', [
22         'mustVerifyEmail' => $request->user() instanceof MustVerifyEmail,
23         'status' => session('status'),
24     ]);
25 }
26
27 /**
28  * Update the user's profile information.
29  */
30 public function update(ProfileUpdateRequest $request): RedirectResponse
31 {
32     $request->user()->fill($request->validated());
33
34     if ($request->user()->isDirty('email')) {
35         $request->user()->email_verified_at = null;
36     }
37
38     $request->user()->save();
39
40     return Redirect::route('profile.edit');
41 }
42
43 /**
44  * Delete the user's account.
45  */
46 public function destroy(Request $request): RedirectResponse
47 {
48     $request->validate([
49         'password' => ['required', 'current_password'],
50     ]);
51
52     $user = $request->user();
53
54     Auth::logout();
55
56     $user->delete();
57
58     $request->session()->invalidate();
59     $request->session()->regenerateToken();
60
61     return Redirect::to('/');
62 }
63
64
```

Abb. 35: ProfileController (Screenshot)

➤ **Profile-Frontend-Komponenten:**

• **UpdateProfileInformationForm:**

- Ermöglicht das Bearbeiten von Namen und E-Mail.
- **E-Mail-Verifizierung:** Zeigt Meldungen oder Verifizierungslinks basierend auf dem Verifizierungsstatus an.
- Fehler wie nicht valide Eingaben werden deutlich dargestellt.

• **Update Password-Form:**

- **Felder:**
  - Aktuelles Passwort.
  - Neues Passwort.
  - Passwortbestätigung.
- Automatisches Zurücksetzen der Eingabe nach Fehlern.

• **DeleteUserForm:**

- Aktiviert das Löschen des Benutzerkontos.
- Verwendet modale Dialoge für Bestätigungen.
- Validiert das Passwort vor dem Löschen.

➤ **Design-Aspekte:**

• **Flexibilität durch Komponenten:**

- **Header:** Beschreibung und Titel der Abschnitte.
- **Formulare:** Verwenden von modularen Elementen wie InputLabel, TextInput und InputError, um Design und Fehlerbehandlung zu vereinheitlichen.

• **Modal-Handling:** wird für kritische Aktionen (z. B. Löschung) eingesetzt, um eine klare Benutzerführung zu gewährleisten.

• **Responsives Layout:** `max-w-x1` und `sm:px-6` sorgen für ein Layout, das auf verschiedenen Geräten optimal aussieht.

Die drei Komponenten nämlich **UpdateProfileInformationForm**, **UpdatePasswordForm** und **DeleteUserForm** werden in der Datei Edit.jsx als eine Seite dargestellt. Folgend zeigt sich die Edit.jsx Datei an.

```
1  import AuthenticatedLayout from '@/Layouts/AuthenticatedLayout';
2  import { Head } from '@inertiajs/react';
3  import DeleteUserForm from './Partials/DeleteUserForm';
4  import UpdatePasswordForm from './Partials/UpdatePasswordForm';
5  import UpdateProfileInformationForm from './Partials/UpdateProfileInformationForm';
6
7  export default function Edit({ mustVerifyEmail, status }) {
8    return (
9      <AuthenticatedLayout
10        header={
11          <h2 className="text-xl font-semibold leading-tight text-gray-800">
12            Profile
13          </h2>
14        }
15      >
16        <Head title="Profile" />
17
18        <div className="py-12">
19          <div className="mx-auto max-w-7xl space-y-6 sm:px-6 lg:px-8">
20            <div className="bg-white p-4 shadow sm:rounded-lg sm:p-8">
21              <UpdateProfileInformationForm
22                mustVerifyEmail={mustVerifyEmail}
23                status={status}
24                className="max-w-xl"
25              />
26            </div>
27
28            <div className="bg-white p-4 shadow sm:rounded-lg sm:p-8">
29              <UpdatePasswordForm className="max-w-xl" />
30            </div>
31
32            <div className="bg-white p-4 shadow sm:rounded-lg sm:p-8">
33              <DeleteUserForm className="max-w-xl" />
34            </div>
35          </div>
36        </div>
37      </AuthenticatedLayout>
38    );
39  }
40
```

Abb. 36: Edit Datei für die Profile Seite (Screenshot)

### 5.4.3. Chat Funktionalitäten

Dieser Teil ist wahrscheinlich der wichtigste Teil des Projekts, denn wenn es um eine Social-Networking-Plattform für Studenten geht, ist die Kommunikation ein wichtiger Aspekt. Bei der Implementierung der Chat-Funktionen wurde das WebSocket-Kommunikationsprotokoll verwendet, um die Kommunikation zwischen zwei Anwendungen, genauer gesagt die Echtzeit-Kommunikation zwischen zwei Benutzern, zu ermöglichen.

Außerdem musste der WebSocket-Dienst in Verbindung mit React und Laravel eingesetzt werden, um den Informationsaustausch zwischen Frontend, Backend und Datenbank zu erleichtern.

### Websockets

<<WebSockets sind ein Kommunikationsprotokoll, das eine bidirektionale Kommunikation zwischen Anwendungen ermöglicht.

Sie sind eine gute Wahl, wenn eine bidirektionale Kommunikation erforderlich ist, wie z. B. bei Chats und der Zusammenarbeit zwischen mehreren Spielern.>><sup>29</sup>

<<WebSockets are a communication protocol that enable bidirectional communication between applications. They are a great choice when two-way communication is needed such as chat and multiplayer collaboration.>> (Übersetz. Von Deepl, 08.12.2024)

### WebSocket Verbindung zwischen Client und Server

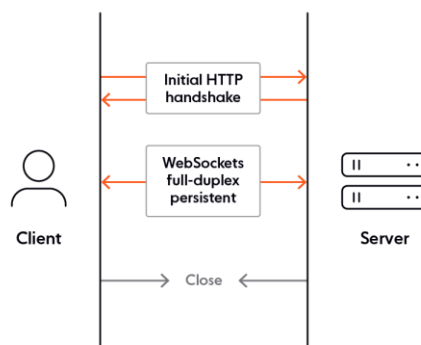


Abb. 37: WebSockets Connections

<sup>29</sup> Booker, A. (2024, 24. April). The complete guide to WebSockets with React. Ably Realtime. <https://ably.com/blog/websockets-react-tutorial#what-are-web-sockets>

Eine WebSocket-Verbindung zwischen Client und Server beginnt mit einem HTTP-Handshake, bei dem der Client eine Anfrage mit einem Upgrade-Header sendet, der seine Absicht signalisiert, Protocols zu wechseln. Wenn der Server WebSockets unterstützt, nimmt er diese Anfrage an und aktualisiert die Verbindung zum WebSocket-Protokoll. Dies schafft einen dauerhaften, bidirektionalen Kommunikationskanal über eine einzige TCP-Verbindung und ermöglicht einen Echtzeit-Datenaustausch. Diese Effizienz ist besonders nützlich für Anwendungen wie Chat-Systeme, in denen sowohl der Client als auch der Server Nachrichten unmittelbar senden und empfangen können, ohne die Mühe, wiederholte HTTP-Anfragen zu stellen.

Zunächst werden die Implementierung der WebSocket Server und die WebSocket Verbindung mit dem Client, in der StudentCommunity-WebApp Anwendung erklärt.<sup>30</sup>

### ➤ WebSocket Server

```

1  const WebSocket = require('ws');
2
3  const wss = new WebSocket.Server({ port: 8080 });
4
5  wss.on('connection', (ws, req) => {
6    console.log('Neue WebSocket-Verbindung');
7
8    ws.on('message', (message) => {
9      console.log('Empfangene Nachricht:', message);
10
11      try {
12        const parsedMessage = JSON.parse(message);
13
14        wss.clients.forEach((client) => {
15          if (client.readyState === WebSocket.OPEN) {
16            client.send(JSON.stringify(parsedMessage));
17          }
18        });
19      } catch (error) {
20        console.error('Fehler beim Parsen der Nachricht:', error);
21        ws.send(JSON.stringify({ error: 'Ungültige Nachricht' }));
22      }
23    });
24  });
25
26  ws.on('close', () => {
27    console.log('WebSocket-Verbindung geschlossen');
28  });
29 });
30
31 console.log('WebSocket-Server läuft auf ws://localhost:8080');
32

```

Abb. 38: WebSocket Server

<sup>30</sup> Booker, A. (2024b, April 24). The complete guide to WebSockets with React. Ably Realtime. <https://ably.com/blog/websockets-react-tutorial#how-web-sockets-work>

Auf einem bestimmten Port, hier Port 8080, lauscht dieser Server und nimmt eingehende Verbindungen von Clients entgegen. Sobald eine Verbindung aufgebaut ist, tritt das `connection`-Ereignis auf, welches den Beginn der Kommunikation mit dem entsprechenden Client ermöglicht. Ein `message`-Event ermöglicht es, Nachrichten zu erhalten und zu verarbeiten. Dabei erfolgt die Umwandlung der eingehenden Daten in ein JavaScript-Objekt und deren Weiterleitung an andere angeschlossene Clients, wenn nötig. Um eine stabile und sichere Kommunikation sicherzustellen, übernimmt der Server auch die Verwaltung von Verbindungsabbrüchen (über das `close`-Ereignis) und bearbeitet Fehler wie das fehlerhafte Parsen von Nachrichten.

### ➤ WebSocket Client Verbindung

```
31 // WebSocket-Verbindung
32 useEffect(() => {
33   if (!currentUser || !chat) return;
34
35   const wsUrl = `ws://localhost:8080?chatId=${chat.id}&userId=${currentUser.id}`;
36   const ws = new WebSocket(wsUrl);
37   setSocket(ws);
38
39   ws.onopen = () => console.log('WebSocket connected to', wsUrl);
40   ws.onmessage = (event) => {
41     const receivedMessage = JSON.parse(event.data);
42     setMessages((prev) => [...prev, receivedMessage]);
43   };
44   ws.onclose = () => console.log('WebSocket closed');
45   ws.onerror = (error) => console.error('WebSocket error:', error);
46
47   return () => ws.close();
48 }, [chat, currentUser]);
```

Abb. 39: WebSocket Verbindung

Eine Verbindung wird eingeleitet, indem eine neue `WebSocket`-Instanz erstellt wird. Dabei werden die Server-URL sowie weitere Parameter wie die `chatId` und `userId` übermittelt. Das `onopen`-Event bestätigt, dass die Kommunikation abgeschlossen ist, nachdem die Verbindung aufgebaut wurde. Die empfangenen Daten werden vom `onmessage`-Ereignis in ein JavaScript-Objekt umgewandelt und in die Nachrichtenhistorie aufgenommen. Zur Gewährleistung der Anwendungsstabilität werden Vorkommnisse wie `onclose` und `onerror` verwendet, um Fehler und Verbindungsabbrüche zu bearbeiten. Diese dauerhafte und bidirektionale Verbindung ist für eine Echtzeitkommunikation von entscheidender Bedeutung, da sie es dem Client und dem Server ermöglicht, Nachrichten unabhängig voneinander zu senden und zu empfangen.

## 6. Test und Evaluation

Testing ist eine bedeutende Phase des Softwareentwicklungsprozesses. Sie gewährleistet, dass jeder verfasste Code wie geplant funktioniert, bevor den nächsten Teil Ihres Projekts gearbeitet wird. Wenn eine Software-Komponente geschrieben wurde, mussten immer einen Test geschrieben, um zu überprüfen, ob das Verhalten den Erwartungen entspricht. Dieser Prozess trägt dazu bei, die Qualität des Codes zu erhalten.<sup>31</sup>

### 6.1. Feature- Tests

In Laravel werden normalerweise zwei Dinge gemeint: Unit Testing und Feature Testing. Unit Testing konzentriert sich nur auf ein kleines Stück Code (normalerweise eine Methodenklasse innerhalb einer Klasse), während Feature-Tests überprüfen, ob eine bestimmte Funktion, die aus mehreren Methoden oder Klassen besteht, die miteinander interagieren, so funktioniert, wie Sie entwickelt haben.<sup>32</sup>

---

<sup>31</sup>, How to Get Started with Unit Testing in Laravel | Better Stack Community. (2024, 23. April). <https://betterstack.com/community/guides/testing/laravel-unit-testing/>

<sup>32</sup> How to Get Started with Unit Testing in Laravel | Better Stack Community. (2024, 23. April). <https://betterstack.com/community/guides/testing/laravel-unit-testing/>



Aber bevor ein Test ausgeführt wird, müssen ein paar Konfigurationen erledigt werden. Es gibt eine Datei `phpunit.xml` in der Projekt-Root-Direktion. Es handelt sich um die Konfigurationsdatei für PHPUnit, einen Testrahmen für PHP-Applikationen. Sie ist in drei Hauptbereiche unterteilt, wie nachstehend dargestellt:



Abb. 40: `phpunit.xml` Datei (Screenshot)

In der `<testsuites>`-Sektion sind zwei verschiedene Arten von Tests für uns vordefiniert. Dieser Abschnitt sagt PHPUnit, welche Tests in den Directories `./tests/Unit` und `./tests/Feature` gespeichert sind.

In der `<coverage>`-Sektion werden dann, welcher Code und welche Directories im Test behandelt werden und wie sie behandelt werden definiert.

In der `<php>`-Sektion könnten abschließend Umgebungsvariablen für die Testumgebung definiert werden. Beim Durchführen eines Tests werden die Variablen, die hier definiert sind, überschrieben durch die, die in der `.env-Datei` definiert sind, also im Projekt-Root.

### **Feature Test:**

Wie früher genannt, überprüft der Feature Test bestimmte Funktion, die aus mehreren Methoden oder Klassen besteht. Für die Anwendung der StudentCommunity-WebApp wurden verschiedene Funktionen aus dem Controller Ordner überprüft. Diese beziehen sich auf Controller in dem Auth Ordner, nämlich `AuthenticationTest`, `EmailverificationTest`, `RegistrationTest`, `PasswordConfirmationTest`, `PasswordUpdateTest` und `PasswordResetTest`. Aber als Beispiel für Testing wurden alle Features gleichzeitig mit dem Befehl `docker-compose exec apache ./vendor/bin/phpunit` ausgeführt. Als Folgendes werden die Registration-Test Datei und Samples von Testergebnissen angezeigt.

```
tests > Feature > Auth > RegistrationTest.php > RegistrationTest > test_new_users_can_register
1  <?php
2
3  namespace Tests\Feature\Auth;
4
5  use Illuminate\Foundation\Testing\RefreshDatabase;
6  use Tests\TestCase;
7
8  class RegistrationTest extends TestCase
9  {
10     use RefreshDatabase;
11
12     public function test_registration_screen_can_be_rendered(): void
13     {
14         $response = $this->get('/register');
15
16         $response->assertStatus(200);
17     }
18
19     public function test_new_users_can_register(): void
20     {
21         $formData = [
22             'matriculationNumber' => '123456', // A valid matriculation number
23             'name' => 'John Doe', // A valid name
24             'email' => 'johndoe@example.com', // A unique valid email
25             'password' => 'password123', // A strong password
26             'password_confirmation' => 'password123', // Match confirmation
27             'phoneNumber' => '1234567890', // Optional valid phone number
28             'studyProgramme' => 'Computer Science', // A valid study program
29             'faculty' => 'Science', // A valid faculty name
30         ];
31         $response = $this->post('/register', $formData);
32
33         $response->assertRedirect(route('verification.notice'));
34         $this->assertAuthenticated();
35     }
36 }
```

Abb. 41: Registration Test Datei (Screenshot)

- Der initiale Test (`test_registration_screen_can_be_rendered`) gewährleistet die korrekte Darstellung der Registrierungsseite.
- Der zweite Test (`test_new_users_can_register`) wird verwendet, um die Registrierung eines Benutzers zu simulieren und zu überprüfen, ob die Registrierung erfolgreich ist, ob eine Weiterleitung durchgeführt wird und ob der Benutzer registriert ist.
- `RefreshDatabase` wird verwendet, um die Datenbank nach jedem Test wieder in ihren ursprünglichen Zustand zu bringen.

## Testergebnisse:

### Test 1:

<b>PASS</b>	Tests\Unit\ExampleTest	
✓	that true is true	0.12s
<b>FAIL</b>	Tests\Feature\Auth\AuthenticationTest	
✓	login screen can be rendered	8.03s
✗	users can authenticate using the login screen	1.05s
✗	users can not authenticate with invalid password	0.34s
✗	users can logout	0.31s
<b>FAIL</b>	Tests\Feature\Auth\EmailVerificationTest	
✗	email verification screen can be rendered	0.26s
✗	email can be verified	0.24s
✗	email is not verified with invalid hash	0.24s
<b>FAIL</b>	Tests\Feature\Auth\PasswordConfirmationTest	
✗	confirm password screen can be rendered	0.26s
✗	password can be confirmed	0.33s
✗	password is not confirmed with invalid password	0.29s
<b>FAIL</b>	Tests\Feature\Auth\PasswordResetTest	
✓	reset password link screen can be rendered	0.23s
✗	reset password link can be requested	0.30s
✗	reset password screen can be rendered	0.26s
✗	password can be reset with valid token	0.25s

Abb. 42: Feature Test 1 (Screenshot)

Bei dem Test 1, wurden die Features AuthenticationTest, E-Mail-VerificationTest, PasswordConfirmationTest und PasswordResetTest nicht-bestanden. Es sollte angemerkt werden, dass die Methoden ‚Login-Bildschirm kann gerendert werden‘ und ‚Passwort-Link-Bildschirm kann gerendert werden‘ bei den Features AuthenticationTest und PasswordResetTest erfolgreich sind. Für einen erfolgreichen Feature-Test müssen jedoch alle Methoden in dem Feature als bestanden betrachtet werden.

### Test 2:

<b>PASS</b>	Tests\Unit\ExampleTest	
✓	that true is true	0.14s
<b>PASS</b>	Tests\Feature\Auth\AuthenticationTest	
✓	login screen can be rendered	9.88s
✓	users can authenticate using the login screen	1.78s
✓	users can not authenticate with invalid password	0.95s
✓	users can logout	0.48s
<b>PASS</b>	Tests\Feature\Auth\EmailVerificationTest	
✓	email verification screen can be rendered	0.38s
✓	email can be verified	0.45s
✓	email is not verified with invalid hash	0.43s
<b>PASS</b>	Tests\Feature\Auth\PasswordConfirmationTest	
✓	confirm password screen can be rendered	0.26s
✓	password can be confirmed	0.22s
✓	password is not confirmed with invalid password	0.44s
<b>PASS</b>	Tests\Feature\Auth\PasswordResetTest	
✓	reset password link screen can be rendered	0.27s
✓	reset password link can be requested	7.02s
✓	reset password screen can be rendered	2.71s
✓	password can be reset with valid token	2.37s
<b>PASS</b>	Tests\Feature\Auth\PasswordUpdateTest	
✓	password can be updated	1.70s
✓	correct password must be provided to update password	0.49s

Abb. 43: Feature Test 2 (Screenshot)

Für Test 2, wurden alle getesteten Features bestanden. Alle Methoden in den Features wurden auch erfolgreich. Außer dem Feature Test werden in dem nächsten Teil die Usability Tests bearbeitet.

## 6.2. Usability Tests

Usability-Tests spielen eine wichtige Rolle bei der Bewertung der Benutzerfreundlichkeit und Leistungsfähigkeit einer Plattform. Bei einem Usability-Test können Beobachtung und Befragung kombiniert werden, um herauszufinden, wie Benutzer mit einer Website oder Anwendung interagieren, welche Elemente sie attraktiv finden und wo potenzielle Verwirrungen auftreten. Das Ziel besteht darin, Schwachstellen zu erkennen und gezielt die Benutzererfahrung zu verbessern.

### Zielsetzungen und Vorzüge von Usability-Tests

Durchführung von Usability-Tests bringt viele Vorzüge mit sich:

- **Qualitätskontrolle:** Gewährleistet eine benutzerfreundliche und funktionale Plattform.
- **Verbesserung der Bedienbarkeit:** Die Nutzerfreundlichkeit wird verbessert, indem komplexe oder verwirrende Bereiche identifiziert und vereinfacht werden.
- **Zielerreichung:** Kontrolliert, ob die Plattform ihren primären Zweck erfüllt, etwa die Förderung von Kommunikation und Kooperation zwischen Studenten.
- **Steigerung der Effizienz:** Verringerung von Fehlern und Verbesserung der Navigation führt zu einer Steigerung der Nutzerzufriedenheit.

Die Tests werden zur Beantwortung der folgenden Fragen verwendet:

- Ist die Navigation intuitiv und klar strukturiert?
- Welche Bedeutung hat der Inhalt der Plattform für die Zielgruppe?
- Befriedigt die Plattform die erhofften Erwartungen und Ziele?
- Ist die Plattform generell leicht zu bedienen?<sup>33</sup>

---

<sup>33</sup> Becker, L. (2024, 7. November). Usability Test: Definition, Vorteile & Methoden | EOM.de. EOM.de. <https://eom.de/insights/usability-tests-nutzerfreundlichkeit-messen-zielgruppe-verstehen-33771.html>

## **Leitfaden für Usability-Tests mit Fokusgruppen**

Die Fokusgruppenmethode wurde für die Usability-Tests der Web-App der StudentCommunity verwendet. Zu den Teilnehmern gehörten Studenten der Hochschule Kaiserslautern. Es wurden die nachfolgenden Schritte berücksichtigt.

### **➤ Zielvorgabe:**

- Die Nutzerfreundlichkeit der wesentlichen Funktionen wie Registrierung, Login, Profilverwaltung und Interaktionen in Gruppen wird bewertet.
- Erfassung der Meinungen der Nutzer bezüglich Navigation und Gestaltung

### **➤ . Auswahl der Teilnehmer:**

Es wurden Studenten aus verschiedenen Studiengängen eingeladen, um eine Vielzahl von Meinungen zu hören.

### **➤ Der Ablauf des Tests:**

- Die Teilnehmer haben bestimmte Aufgaben erledigt, etwa ein Profil erstellen oder in einer Gruppe posten.
- Die Teilnehmer erhielten während des Gebrauchs die Möglichkeit, ihre Gedanken laut auszusprechen („Think-Aloud-Methode“).
- Eindrücke über Design, Navigation und Benutzerführung wurden in moderierten Diskussionen festgehalten.

### **➤ Die Resultate der Usability-Tests**

#### **• Positive Schlussfolgerungen:**

Die Usability-Tests haben gezeigt, dass die Plattform mit ihrer benutzerfreundlichen und klaren Navigation überzeugend ist. Auch das übersichtliche und minimalistische Design wurde sehr gelobt, da es die Handhabung vereinfacht und optisch ansprechend gestaltet ist. Die Gruppen- und Chatfunktionen wurden insbesondere als sehr hilfreich für die Kooperation und den Austausch in der Gemeinschaft betont.

- **Vorschläge zur Verbesserung:**

Die Tests zeigten auch Bereiche mit Verbesserungspotenzial auf. So wurde die Einführung eines Onboarding-Prozesses vorgeschlagen, um neuen Nutzern den Einstieg zu erleichtern. Zudem sollten Fehlermeldungen präziser formuliert werden, um den Nutzern klare Hilfestellung bei Problemen zu geben. Abschließend wurde angemerkt, dass die mobile Version der Plattform optimiert werden könnte, um die Benutzerfreundlichkeit auf Smartphones zu verbessern.

## **7. Fazit und Ausblick**

### **7.1. Zusammenfassung der Ergebnisse**

Eine auf den Technologien Laravel und React basierende soziale Netzwerkplattform für Studenten wurde in dieser Arbeit erstellt. Das Ziel bestand darin, eine Plattform zu schaffen, die es den Studierenden ermöglicht, sich auszutauschen und miteinander zu kooperieren. Eine individuell anpassbare Kommunikation, ein Veranstaltungskalender und eine Jobbörse gehörten zu den Modulen, die im Verlauf der Arbeit erfolgreich umgesetzt wurden.

Die ausgearbeitete Plattform ist bekannt für ihre Nutzerfreundlichkeit, Sicherheit und Flexibilität. Funktionelle Prüfungen haben gezeigt, dass die grundlegenden Anforderungen erfüllt sind. Die Sicherheit und Stabilität der Plattform wurden maßgeblich durch den Einsatz moderner Softwarearchitekturen sowie die Einbindung von Sicherheitsmechanismen wie CSRF-Schutz verbessert. Außerdem ist es auch klar zu sagen, dass die Arbeit in Bezug auf die funktionale und nicht-funktionale Anforderungen nicht alle Implementiert wurden und werden als Verbesserungsvorschlägen betrachtet.

### **7.2. Persönliches Fazit**

Die Realisierung der Arbeit ermöglichte es, theoretisches Wissen sowie praktische Fähigkeiten in der Full-Stack-Entwicklung anzuwenden und auszubauen. Die Verwendung von Laravel und React hat sich als die beste Option für die Entwicklung einer leistungsstarken und skalierbaren Anwendung erwiesen. Durch die Beschäftigung mit zeitgemäßen Technologien und Prinzipien des Softwaredesigns konnte die Kenntnisse über Softwareentwicklung auf ein höheres Niveau gebracht werden.

Es war eine Herausforderung, die Usability für verschiedene Benutzergruppen zu gewährleisten. Die fortlaufende Bewertung mithilfe von Usability-Tests war dabei unverzichtbar. In diesen Tests wurde festgestellt, dass die Plattform zwar gut aufgenommen wurde, aber es gibt auch Möglichkeiten zur Verbesserung, vor allem im Hinblick auf die mobile Nutzung.



### 7.3. Zukünftige Arbeiten und Erweiterungsmöglichkeiten

Die ausgearbeitete Plattform bietet eine zuverlässige Grundlage, die in der Zukunft weiterentwickelt werden kann. Im Folgenden sind einige Möglichkeiten zur Erweiterung aufgeführt:

- **Erweiterung der Funktionen:** Einbindung zusätzlicher Module, wie zum Beispiel eines Lernmanagementsystems oder Werkzeuge zur Zusammenarbeit in Projekten.
- Die mobile Nutzung wird verbessert, indem eine native mobile App entwickelt oder die bestehende responsive Web-App verbessert werden.
- **Internationalisierung:** Die Plattform soll in verschiedenen Sprachen angeboten werden, damit Studierende aus verschiedenen Ländern angesprochen werden können.
- **Funktionen, die auf KI basieren:** Einführung von Funktionen wie smarten Jobempfehlungen oder personalisierten Lernressourcen unter Verwendung von Künstlicher Intelligenz.
- **Skalierbarkeit und Leistungsfähigkeit:** Durch die Nutzung von Cloud-Technologien wird die Skalierbarkeit verbessert und die Datenbankstruktur wird optimiert.
- Um die Benutzerfreundlichkeit weiter zu verbessern, werden ausführliche Usability-Tests durchgeführt, an denen ein größerer Benutzerkreis beteiligt ist.
- **Bearbeitung der fehlenden Anforderungen:** Alle funktionale oder nicht-funktionale Anforderungen, die nicht entwickelt konnten, sollen nochmal betrachtet werden.

zusammenfassend kann festgehalten werden, dass die Arbeit nicht bloß eine funktionsfähige Plattform geschaffen hat, sondern auch viele Möglichkeiten für künftige Entwicklungen aufzeigt. Damit trägt sie zur Förderung der Vernetzung und Kooperation unter den Studierenden bei.

## Literaturverzeichnis

Andrea. (2023, 17. April). Das Wasserfallmodell einfach erklärt. Projekte Leicht Gemacht. <https://projekte-leicht-gemacht.de/blog/projektmanagement/klassisch/wasserfallmodell/>

Becker, L. (2024, 7. November). Usability Test: Definition, Vorteile & Methoden | EOM.de. EOM.de. <https://eom.de/insights/usability-tests-nutzerfreundlichkeit-messen-zielgruppe-verstehen-33771.html>

Booker, A. (2024, 24. April). The complete guide to WebSockets with React. Ably Realtime. <https://ably.com/blog/websockets-react-tutorial#what-are-web-sockets>

Booker, A. (2024b, April 24). The complete guide to WebSockets with React. Ably Realtime. <https://ably.com/blog/websockets-react-tutorial#how-web-sockets-work>

Garden, T. (2020, 2. Oktober). Grundlagen Datenbankdesign: So funktioniert der Einstieg. talentgarden. <https://blog.talentgarden.com/de/blog/data/grundlagen-datenbankdesign-so-funktioniert-der-einstieg>

Helbert, D. (2021, 3. Juni). Laravel und Docker: Die perfekte Entwicklungsumgebung. Laravel Tutorials. <https://laravel.dirk-helbert.de/laravel-docker/>

How to Get Started with Unit Testing in Laravel | Better Stack Community. (2024, 23. April). <https://betterstack.com/community/guides/testing/laravel-unit-testing/>

Kuphal, A. (2010, S. 2). Soziale Netzwerke und ihre Vor- und Nachteile: Speziell: Cybermobbing. GRIN Verlag.

Kinsta. (2023b, Oktober 9). Was ist React.js? Ein Blick auf die beliebte JavaScript-Bibliothek. Kinsta®. <https://kinsta.com/de/wissensdatenbank/was-ist-react-js/>

Kinsta. (2023, 6. Oktober). Das Laravel PHP-Framework - Web-App-Konstruktion für jedermann. Kinsta®. <https://kinsta.com/de/wissensdatenbank/warum-solltest-du-laravel-verwenden/>

Kinsta. (2023, 6. Oktober). Das Laravel PHP-Framework - Web-App-Konstruktion für jedermann. Kinsta®. <https://kinsta.com/de/wissensdatenbank/was-ist-laravel/> Kinsta.

(2023, 6. Oktober). Das Laravel PHP-Framework - Web-App-Konstruktion für jedermann.

Kinsta®. <https://kinsta.com/de/wissensdatenbank/warum-solltest-du-laravel-verwenden/>

Inertia.js - the modern monolith. (o. D.). <https://inertiajs.com/>

LaRavel - the PHP framework for web artisans. (o. D.).

<https://laravel.com/docs/11.x/eloquent#generating-model-classes>

LaRavel - the PHP framework for web artisans. (o. D.).

<https://laravel.com/docs/11.x/migration>

LaRavel - the PHP framework for web artisans. (o. D.).

<https://laravel.com/docs/11.x/controllers>

LaRavel - the PHP framework for web artisans. (o. D.).

<https://laravel.com/docs/11.x/ Routing>

LaRavel - the PHP framework for web artisans. (o. D.).

<https://laravel.com/docs/11.x/middleware>

LaRavel - the PHP framework for web artisans. (o. D.-b).

<https://laravel.com/docs/11.x/authentication#introduction>

LaRavel - the PHP framework for web artisans. (o. D.-c).

<https://laravel.com/docs/11.x/authentication#ecosystem-overview>

LaRavel - the PHP framework for web artisans. (o. D.-d).

<https://laravel.com/docs/11.x/authentication#laravels-api-authentication-services> LaRavel

- the PHP framework for web artisans. (o. D.-b).

<https://laravel.com/docs/11.x/sanctum#main-content>

LaRavel - the PHP framework for web artisans. (o. D.-b).

<https://laravel.com/docs/11.x/sanctum#spa-authentication>

LaRavel - the PHP framework for web artisans. (o. D.-b).

<https://laravel.com/docs/11.x/sanctum#csrf-protection>

LaRavel - the PHP framework for web artisans. (o. D.).

<https://laravel.com/docs/11.x/installation>

Metag, J. & Schäfer, M. S. (2017). Hochschulen zwischen Social Media-Spezialisten und Online-Verweigerern. Eine Analyse der Online-Kommunikation promotionsberechtigter Hochschulen in Deutschland, Österreich und der Schweiz. *Studies in Communication And Media*, 6(2), 160–195.

<https://doi.org/10.5771/2192-4007-2017-2-160>

Software-Architektur | PFP Software GmbH. (o. D.).

<https://pfp.gmbh/beratung/architektur/>

Stanković, D. (2023, 23. November). Laravel Breeze: The Ultimate Installation Guide for Laravel Starter Kit's Authentication System. Medium.

<https://medium.com/@stdejan/laravel-breeze-the-ultimate-installation-guide-for-laravel-starter-kits-authentication-system-a59ecf40dcf6>

Steinweg, C. (2013, S. 92). Projektkompass Softwareentwicklung: Geschäftsorientierte Entwicklung von IT-Systemen. Springer-Verlag.

Unger, R. & Chandler, C. (2012, Kapitel 1). A Project Guide to UX-Design: For user experience designers in the field or in the making. New Riders.

Vogel, O., Arnold, I., Chughtai, A., Ihler, E., Kehrer, T., Mehlig, U. & Zdun, U. (2009, S. 476). Software-Architektur: Grundlagen - Konzepte - Praxis. Springer-Verlag.

# Anhang

## 1. Gitlab Ticketübersichten

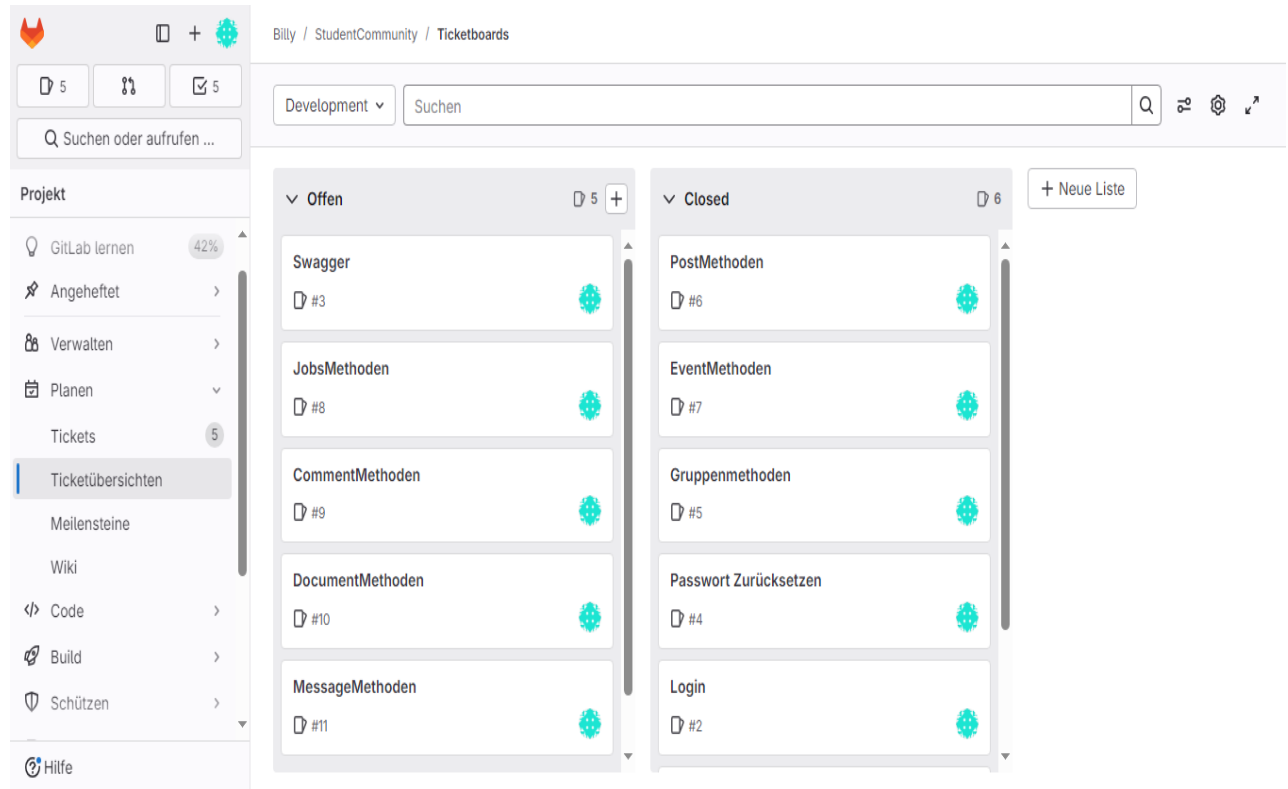


Abb. 44: Taskübersichten (Screenshot)

## 2. Swagger Dokumentation

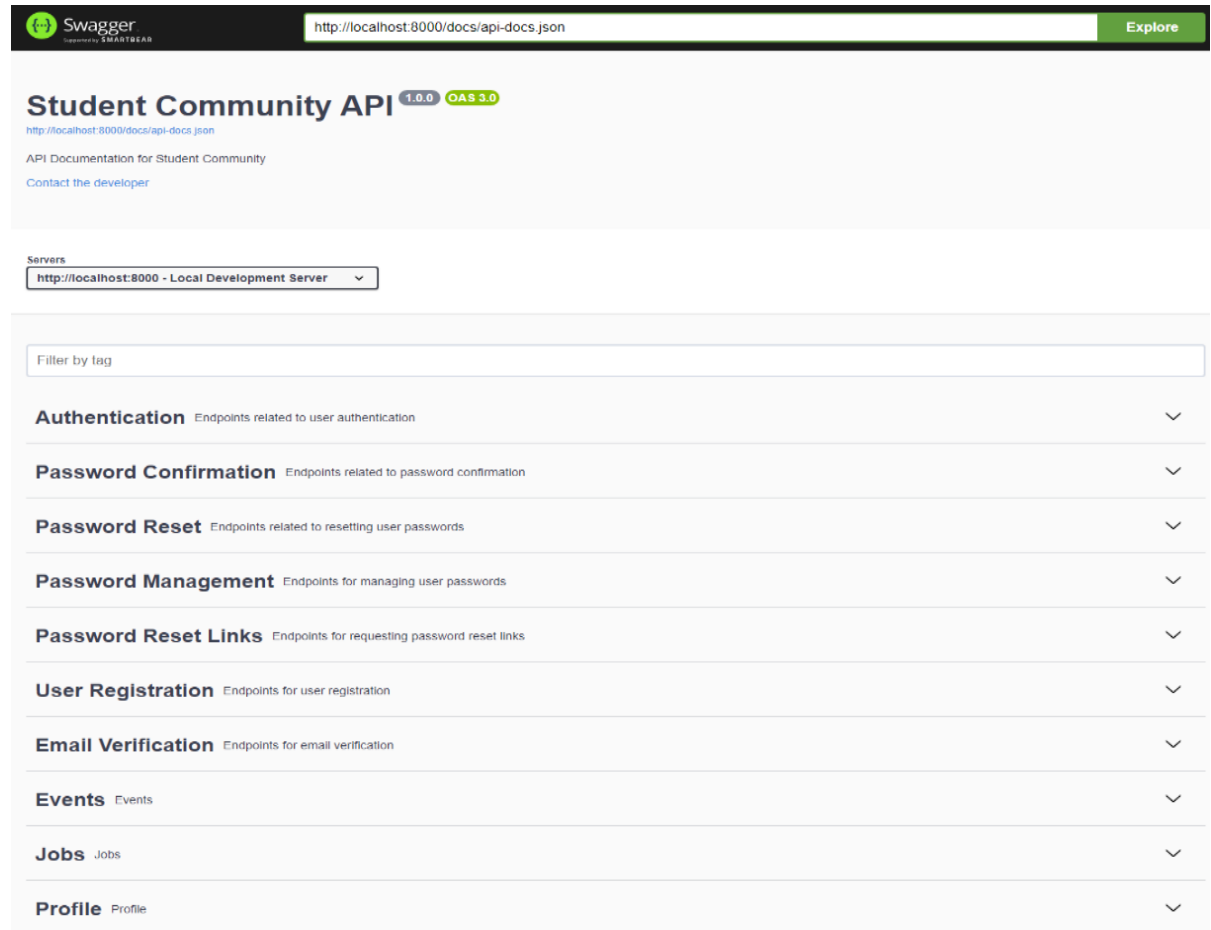


Abb. 45: Swagger Dokumentierung (Screenshot)

### 3. Dokumentation der Nutzung von KI-Tools

KI-basiertes Hilfsmittel	Einsatzform	Betroffene Teile der Arbeit	Bemerkungen
ChatGPT, <a href="https://chatgpt.com/">https://chatgpt.com/</a>	Erstellung von Textvorschlägen, im Text	Abstract, Vorwort, Danksagung und Teile der Implementierung, bzw. das Kapitel 5.2, wobei der Inhalt aus dem Laravel Doc die auf Englisch aufgesetzt ist, wurde auf Deutsch umformuliert.	Ziel der Nutzung beim Kapitel 5.2, war nur ein guter Inhalt im deutsche Sprache Form zu haben , von den Inhalten, die aus dem Laravel Dokument stammen. Aber die Quellen sind vorhanden.
DeepL Translator (DeepL SE) <a href="https://www.deepl.com/translator">https://www.deepl.com/translator</a>	Übersetzung von Definitionen.	Definition von WebSockets in Kapitel 5.4.3	

*Tabelle 1: Dokumentation der Nutzung von KI-Tools*



## Eidesstattliche Erklärung

Hiermit versichere ich, dass ich folgende Prüfung

Bachelorarbeit

---

*Art (zum Beispiel Hausarbeit, Projektarbeit, Bachelorarbeit)*

Entwicklung einer sozialen Netzwerkplattform für Studenten: Konzeption, Implementierung und Evaluation „Full-Stack-Anwendung mit Laravel und React.“

---

*Titel*

eigenständig erbracht, keine anderen als die angegebenen Quellen und erlaubten Hilfsmittel benutzt und die aus fremden Quellen direkt oder indirekt übernommenen Gedanken (Texte, Textteile oder Textbausteine) als solche kenntlich gemacht habe.

Falls ich KI-generierte Outputs verwendet habe, habe ich dies gemäß der Vorgabe des/der Prüfenden kenntlich gemacht.

Die Arbeit habe ich in dieser oder ähnlichen Form oder in Auszügen noch keiner Prüfungsbehörde zu Prüfungszwecken vorgelegt.

Sofern sowohl eine schriftliche wie auch elektronische Version der Arbeit von mir abgegeben werden, erkläre ich, dass beide Versionen der Arbeit identisch sind.

Mir ist bekannt, dass Zuwiderhandlungen gegen den Inhalt dieser Erklärung einen Täuschungsversuch darstellen, der grundsätzlich das Nichtbestehen der Prüfung zur Folge hat.

Zweibrücken, 09.12.2024

---

*Ort, Datum*

Billy Max Tsane Tsafack

---

*Vorname und Name in Druckschrift*



---

*Unterschrift*