Hochschule Esslingen
University of Applied Sciences

Entwicklung von Software in der EPLAN GmbH & Co. KG

Praxissemesterbericht

im Studiengang
Softwaretechnik und Medieninformatik

vorgelegt von

Baran Ibrahim Cal
Matr.-Nr.: 768960

am 13. März 2025
an der Hochschule Esslingen

Betreuer: Prof. Dipl.-Informatikerin Astrid Beck

Kurzfassung

Das Praxissemester dient dazu, die im Studium erworbenen theoretischen Kenntnisse in der praktischen Arbeit anzuwenden und zu vertiefen. Durch die Mitarbeit an realen Projekten können nicht nur technische Fähigkeiten erweitert, sondern auch wertvolle Erfahrungen in der Teamarbeit und Projektorganisation gesammelt werden. Zudem bietet das Praktikum die Möglichkeit, Einblicke in den beruflichen Alltag zu gewinnen und sich auf zukünftige Herausforderungen im Berufsleben vorzubereiten.

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

PBI	Product Backlog Item
PC	Personal Computer
HTML	Hypertext Markup Language
npm	Node Package Manager
WYSIWYG	What You See Is What You Get
Enum	Enumeration
UI	User Interface
UI	User Interface

1 Einleitung

Im Rahmen des Studiums im Bachelorstudiengang Softwaretechnik und Medieninformatik an der Hochschule Esslingen war ein Praxissemester, welches mindestens 100 Arbeitstage umfassen muss, vorgesehen. Das Praktikum bietet die Möglichkeit, Kenntnisse die theoretisch als auch praktisch während des Studiums erworben wurden, in einem realen Arbeitsumfeld anzuwenden und wertvolle Erfahrungen in der Softwareentwicklung zu sammeln. Dabei ist es wichtig ein Unternehmen zu finden welches verschiede Projekte betreut und dem Praktikanten mehr Verantwortung gibt, um dessen fachliche Kompetenz weiterzuentwickeln. Die Entscheidung für ein Praktikum bei E-PLAN fiel aufgrund des ersten und sehr informativen Gesprächs, sowie eines Kennenlerntages, wo man das gesamte Team kennenlernen und einiges über das Unternehmen lernen durfte. Das Praktikum dauerte sechs Monate und das im Zeitraum vom 02.09.24 bis zum 28.02.25.

2 EPLAN GmbH & Co. KG

Anfang der 1980er Jahre kamen die ersten PCs auf den Markt. In dieser Zeit wagte Winfried Müller den Versuch, Stromlaufpläne mithilfe kostengünstiger Software zu erstellen. Diese simple, aber wegweisende Idee legte den Grundstein für einen weltweiten Standard und führte zur Gründung des Unternehmens, das heute als EPLAN bekannt ist (EPLAN Gmbh & Co. KG, 2024a).

Am 28. Juni 1984 wurde im Rheinland die Wiechers & Partner Datentechnik GmbH gegründet – ein Unternehmen, das mit der Software EPLAN ein innovatives Produkt für die Elektrokonstruktion auf den Markt brachte.

Die Idee entstand aus der Erkenntnis, dass das manuelle Zeichnen von Elektro-Schaltplänen mit Bleistift und Lineal nicht nur mühsam, sondern auch fehleranfällig und zeitaufwendig war. Um dieses Problem zu lösen, wurde eine Software für die Elektrokonstruktion von Maschinen und Anlagen entwickelt, die den gesamten Konstruktionsprozess erheblich vereinfachte, standardisierte und automatisierte. Besonders unter Ingenieuren, die nach effizienteren Lösungen suchten, fand die Innovation schnell Anklang und setzte sich rasch am Markt durch. Die Software markierte einen entscheidenden Schritt in der Digitalisierung der Elektrotechnik und legte den Grundstein für einen neuen Industriestandard. Zu jener Zeit waren elektrotechnische Planungen eine große Herausforderung, da sie ausschließlich mit leistungsstarken Großrechnern durchgeführt werden konnten – eine Technologie, die sich nur große Unternehmen leisten konnten. Dies führte dazu, dass kleinere Betriebe kaum Zugang zu effizienten digitalen Lösungen für die Elektrokonstruktion hatten.

Mit der neu entwickelten Software änderte sich dies grundlegend: Erstmals war es möglich, dass jeder mit einem handelsüblichen PC grafisch interaktive Schaltpläne erstellen konnte. Diese Innovation brachte eine enorme Erleichterung mit sich, da das computergestützte Engineering (CAE) den Planungsprozess erheblich vereinfachte und beschleunigte.

Durch die Automatisierung vieler manueller Schritte sparte die Software wertvolle Zeit. Diese Effizienzsteigerung fand großen Anklang in der Branche und trug maßgeblich dazu bei, dass sich die Lösung schnell am Markt etablierte. Abbildung 1 veranschaulicht diesen frühen Entwicklungszeitraum, indem sie einen Blick auf die Anfänge von EPLAN gibt.

Abbildung 1: Die Ursprünge von EPLAN

Bereits ein Jahr später expandierte EPLAN in die Automobilindustrie, was das wachsende Potenzial der Software weiter unterstrich. 1986 erkannte Prof. Friedhelm Loh frühzeitig die enormen Möglichkeiten der Innovation und investierte in das aufstrebende Software-Start-up. Mit diesem Schritt wurde EPLAN Teil der Friedhelm Loh Group, die sich heute als einer der weltweit führenden Anbieter für Engineering-Lösungen etabliert hat. Diese strategische Integration ebnete den Weg für die Entwicklung und Einführung weiterer innovativer Produkte (EPLAN GmbH & Co. KG, 2024b)

2.1 EPLAN Heute

Heute, 40 Jahre nach der Gründung, feierte EPLAN sein Jubiläum in Köln und blickt auf eine erfolgreiche Unternehmensgeschichte zurück. Mit Hauptsitz in Monheim am Rhein (EPLAN GmbH & Co. KG, 2024c), rund 68.000 Kunden, 274.000 Lizenzen, 50 Standorten weltweit und 1.460 Mitarbeitern zählt EPLAN heute zu den führenden Anbietern für Engineering-Software (EPLAN GmbH & Co. KG o.D.). Als Teil von Rittal

Software Systems innerhalb der Friedhelm Loh Group treibt das Unternehmen die Digitalisierung und Automatisierung in der Elektrokonstruktion maßgeblich voran (EPLAN Gmbh & Co. KG, 2024a). Im Laufe der Jahre hat EPLAN sein Tätigkeitsfeld kontinuierlich erweitert und ist heute in zahlreichen Branchen vertreten. Dazu gehören unter anderem der Maschinen- und Anlagenbau, der Schaltschrankbau, die Automobilindustrie sowie die Energie- und Gebäudetechnik. Für diese Branchen bietet EPLAN eine Vielzahl an innovativen Lösungen, darunter das EPLAN Data Portal, EPLAN eSTOCK und EPLAN eBUILD. Während des Praktikums erfolgte die Mitarbeit in einem Team, das sich speziell mit der Weiterentwicklung und Optimierung von EPLAN eBUILD beschäftigt (EPLAN GmbH & Co. KG, 2024d).

2.2 EPLAN eBUILD

EPLAN eBUILD ist eine Softwarelösung, die speziell entwickelt wurde, um Kunden im Engineering-Bereich eine effizientere Möglichkeit zur Erstellung von Schaltplänen zu bieten, ohne dabei Prozesse von Grund auf neu gestalten zu müssen. Die Software ermöglicht es, bewährte Vorlagen wiederzuverwenden und daraus automatisiert neue Schaltpläne zu generieren. Neben der Möglichkeit, Vorlagen wiederzuverwenden, bietet eBUILD weitere Vorteile, wie beispielsweise einen flexiblen Zugriff auf Bibliotheken dank der Cloud-Anbindung. Dies ermöglicht eine flexible Nutzung ohne die Notwendigkeit einer Installation. Zudem trägt eBUILD dazu bei, Engineering-Prozesse zu standardisieren (EPLAN GmbH & Co. KG, 2024e). Dabei setzt sich eBUILD aus zwei Funktionsbereichen zusammen. um einen können Benutzer eigene Vorlagen-Bibliotheken erstellen, die auf der EPLAN Makrotechnologie basieren und von erfahrenen Anwendern im Designer eingerichtet werden. Diese Bibliotheken sind dann in der Cloud für das gesamte Unternehmen zugänglich. Auf diese Weise lassen sich gesamte Engineering-Prozesse automatisieren und effizienter gestalten. Die Bibliotheken können im Project Builder jederzeit wiederverwendet werden, wodurch häufig genutzte Elemente von Schaltplänen schnell und mit nur wenigen Klicks zusammengestellt werden können. Diese Funktion spart Zeit bei wiederkehrenden Aufgaben und ist benutzerfreundlich, sodass keine lange Einarbeitung erforderlich ist (EPLAN GmbH & Co. KG, 2024f). Abbildung 2 veranschaulicht den Einsatz von eBUILD.

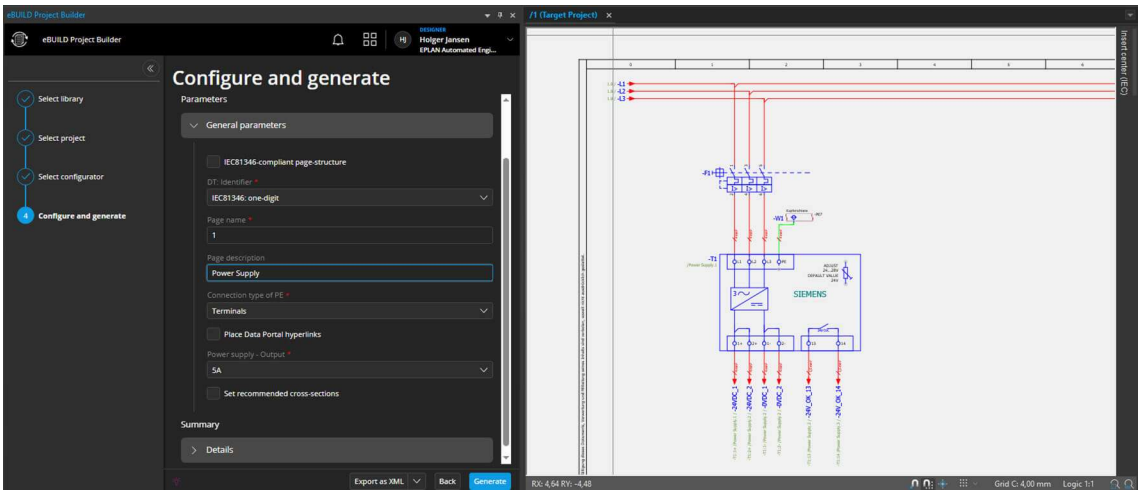

Abbildung 2: EPLAN eBUILD

2.3 Unternehmensalltag

Der Arbeitsalltag während des Praktikums verlief insgesamt strukturiert und gleichmäßig. Im Büro begann die Arbeit in der Regel um 7 Uhr, während die Arbeitszeiten im Homeoffice flexibler waren und meist zwischen 6 und 8 Uhr starteten. Der Arbeitstag endete nach 8,5 Stunden, typischerweise zwischen 14:30 und 17:30 Uhr. Zu Beginn jedes Arbeitstages steht zunächst das Prüfen der E-Mails an, um über aktuelle Entwicklungen, Aufgaben und wichtige Informationen auf dem neuesten Stand zu bleiben. Zusätzlich bietet Outlook eine praktische Übersicht, in der alle anstehenden Termine des Tages, der Woche oder sogar des gesamten Monats auf einen Blick sichtbar sind. Dies erleichtert die Planung, sorgt für eine bessere Organisation und stellt sicher, dass keine Meetings übersehen werden. Nach der ersten Arbeitsphase, die bis 11 Uhr andauert, folgt das tägliche Team-Meeting, das sogenannte "Daily". In diesem Austausch besprechen die Teammitglieder den aktuellen Stand ihrer Aufgaben und gehen gemeinsam die jeweiligen PBIs durch. Zudem bietet das Meeting die Möglichkeit, Herausforderungen oder Probleme anzusprechen, um gemeinsam Lösungen zu finden. Dieser regelmäßige Austausch fördert die Zusammenarbeit, sorgt für Transparenz im Team und hilft dabei, effizient an den Projekten zu arbeiten. Nach dem Daily wird im Büro bis 12 Uhr weitergearbeitet, bevor die meisten eine Mittagspause einlegen. Einige holen sich etwas zu essen, während andere ihr mitgebrachtes Essen genießen. In der Regel verbringen die Mitarbeiter die Pause gemeinsam, was den Teamgeist stärkt. Nach mindestens 30 Minuten geht es zurück an die Arbeit, bis schließlich der Arbeitstag endet.

2.4 Arbeitsweise

Von den fünf Arbeitstagen sind Dienstag und Mittwoch fest für die Arbeit im Büro vorgesehen. An diesen Tagen treffen sich die Kollegen vor Ort, um gemeinsam zu arbeiten. Montag, Donnerstag und Freitag verbringen die meisten im Homeoffice und erledigen ihre Aufgaben von zu Hause aus. Zwar besteht auch an diesen Tagen die Möglichkeit, ins Büro zu kommen, jedoch ist dies eher die Ausnahme, sodass man dort nur auf wenige Kollegen treffen würde. Die gesamte Entwicklungsabteilung in EPLAN, einschließlich des Teams, arbeitet agil nach der Scrum-Methode. Die Arbeit erfolgt in zweiwöchigen Sprints, die durch ein tägliches Daily-Meeting begleitet werden. Jeder Sprint beginnt mit zwei Planungsterminen (Plannings), in denen die anstehenden Aufgaben festgelegt werden. In der darauffolgenden Woche findet das Refinement statt, bei dem offene Anforderungen präzisiert und priorisiert werden. Eine Woche nach dem Refinement und ein Tag vor dem nächsten Planning, folgt die Retrospektive und das Sprint Review, in der das Team den vergangenen Sprint reflektiert und Verbesserungen bespricht und ebenso die erzielten Ergebnisse präsentiert und bewertet. Die meisten dieser Meetings finden bevorzugt an einem Dienstag oder Mittwoch statt, da sie im Büro effektiver durchgeführt werden können als im Homeoffice. Abgesehen von der Retrospektive und den Daily-Meetings waren die meisten dieser Termine nicht direkt relevant für das Projekt. Sie wurden jedoch in den ersten Monaten regelmäßig besucht, um einen besseren Einblick in die Abläufe und die agile Arbeitsweise des Teams zu erhalten. Jede Woche fand ein Meeting mit dem Betreuer statt, um den aktuellen Fortschritt zu besprechen, Probleme zu klären und organisatorische Themen zu erledigen. Bei Herausforderungen konnten jederzeit auch die Kollegen um Unterstützung gebeten werden, die gerne halfen, sofern es möglich war. Während des gesamten Projekts wurden sämtliche Probleme, Fortschritte, Rückschläge und Ideen in OneNote dokumentiert, um den Verlauf nachvollziehbar zu halten und eine strukturierte Übersicht zu gewährleisten.

3 Richtexteditor

3.1 Projektübersicht

Während des Praktikums erfolgte eine Mitarbeit in einer Entwicklungsabteilung mit Fokus auf die Entwicklung und Umsetzung eines Projekts. Dieses Projekt ist ein Teil eines größeren Gesamtprojekts und trägt zu dessen Umsetzung bei. Die finale Software, die auch das eigene Teilprojekt umfasst, wird zu einem späteren Zeitpunkt veröffentlicht und steht anschließend den Kunden von EPLAN zur Verfügung.

Während des Praktikums bei EPLAN lag ein besonderer Fokus des Teams auf der Weiterentwicklung des UI-Updates für den eBUILD Designer. In diesem Zusammenhang sollte der Praktikant den bestehenden Texteditor für die Beschreibungen in den Bibliotheken des Designers durch eine moderne Lösung ersetzen. Der alte Texteditor, der noch von Kunden verwendet wird, basiert auf Markdown und bietet keine der fortschrittlichen Funktionen, die heutzutage in modernen Texteditoren Standard sind. Markdown ist eine einfache Auszeichnungssprache, mit der man Text formatieren kann, ohne komplizierte Befehle oder spezielle Software zu benötigen. Man schreibt den Text und nutzt bestimmte Zeichen, um Elemente wie Überschriften, Fett- oder Kursivdruck, Listen oder Links zu erstellen. Zum Beispiel macht man eine Überschrift mit einer Raute (# Überschrift), fettgedruckten Text mit zwei Sternchen (**fett**) und Listen mit Bindestrichen (- Punkt). Sämtliche Inhalte müssen manuell eingegeben werden, was besonders bei umfangreichen Texten sehr umständlich und zeitraubend ist. Diese fehlenden Funktionen machen den Texteditor nicht nur ineffizient, sondern auch unpraktisch und benutzerunfreundlich. Daher war es notwendig, diesen veralteten Texteditor durch eine zeitgemäße und benutzerfreundliche Alternative zu ersetzen, die eine schnellere und effektivere Bearbeitung der Texte ermöglicht. Der neue Editor sollte Funktionen wie automatische Formatierung, einfaches Kopieren und Einfügen sowie bessere Möglichkeiten zur Texterstellung bieten, um die Benutzererfahrung zu verbessern und die Effizienz zu steigern. Abbildung 3 zeigt den aktuell verwendeten Editor mit einem Beispieltext in Markdown, der die Einschränkungen des bisherigen Systems verdeutlicht.

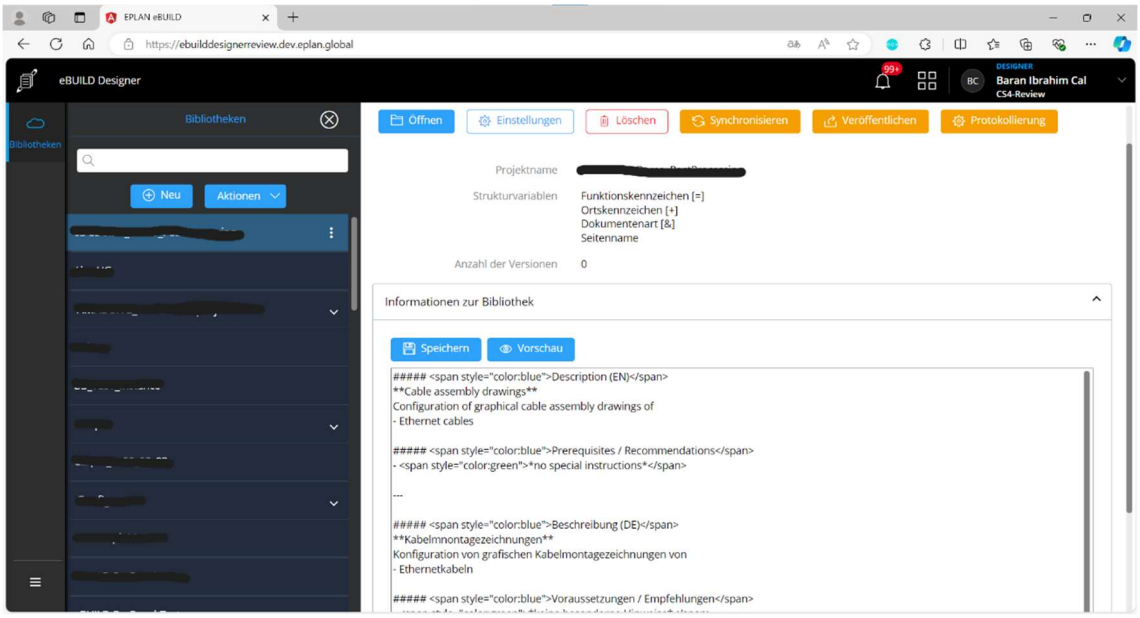

Abbildung 3: Der noch verwendete Editor

Um die Mängel des alten Texteditors zu beheben, wurde zunächst auf alternative Texteditoren gesetzt, die zwar weiterhin Markdown anzeigen und speichern konnten, aber zusätzliche Funktionen wie die Möglichkeit, die Textgröße anzupassen oder Text zu unterstreichen, boten. Diese Editoren waren ein Schritt in die richtige Richtung, da sie mehr Flexibilität ermöglichten. Allerdings war irgendwann klar, dass solche Editoren immer noch nicht alle Möglichkeiten eines modernen Rich-Text-Editors bieten, der HTML anzeigen und speichern kann und weitaus umfangreichere Funktionen und Features bietet. Dies führte dazu auf einen Rich-Text-Editor zu umzusteigen.

Der aktuelle Texteditor, der „Angular WYSIWYG Text Editor“, wurde vom Praktikanten selbst implementiert, nachdem mit dem Betreuer gemeinsam eine gründliche Auswahl der verfügbaren Editoren getroffen hatte. Die Integration des Editors in eBUILD erfolgte in einem separaten Git-Repository, was jedoch mit vielen Schwierigkeiten verbunden war. Anstatt den Editor über ein einfaches npm install zu installieren, mussten die Editordateien aus GitHub heruntergeladen werden und in den neuen Branch eingefügt werden. Dies wurde so umgesetzt, weil bei der Verwendung von npm install nicht auf die Icons in den Buttons zugegriffen werden konnte, was dazu führte, dass der E-PLAN-Style nicht korrekt implementiert werden konnte. Dies führte zu zahlreichen Fehlern, die nur nach und nach behoben werden konnten.

Der „Angular WYSIWYG Text Editor“ bietet jedoch eine Reihe von Funktionen, die in modernen Texteditoren mittlerweile Standard sind. Dazu gehören das Ändern der Textfarbe, das Verlinken von Texten und Bildern sowie die Möglichkeit, Bilder zu vergrö-

ßern oder zu verkleinern. Diese Features ermöglichen eine deutlich flexiblere Nutzung und verbessern die Benutzerfreundlichkeit des Editors.

Eine weitere Herausforderung bestand darin, den bereits gespeicherten Text der Kunden, der im Markdown-Format vorlag, in HTML zu konvertieren und korrekt darzustellen. Diese Aufgabe wurde durch den Einsatz der Bibliothek „markdown-it“ gelöst, die Markdown nahezu perfekt in HTML umwandelt. Allerdings sollte die Konvertierung nicht bei jedem Seitenaufruf erfolgen, sondern nur einmalig, sodass der Text nach der ersten Umwandlung vom neuen Editor automatisch im HTML-Format gespeichert wird. Um dies zu erreichen, wurde im Backend eine Enum mit den Werten „Markdown“ und „HTML“ erstellt, wobei „Markdown“ als Standardwert festgelegt wurde. Wenn der Text im Frontend durch den neuen Editor gespeichert wird, wird das Enum für diesen Text auf „HTML“ gesetzt. Beim Öffnen der Bibliothek im Frontend wird das Enum abgefragt, und je nach Wert erfolgt entweder eine Konvertierung des Textes oder nicht. So wird sichergestellt, dass die Konvertierung nur einmalig erfolgt und der Text anschließend in HTML gespeichert bleibt. Die Abbildung 4 zeigt den aktuellen „Angular WYSIWYG Text Editor“ im Dark Mode mit seinen zentralen Funktionen.

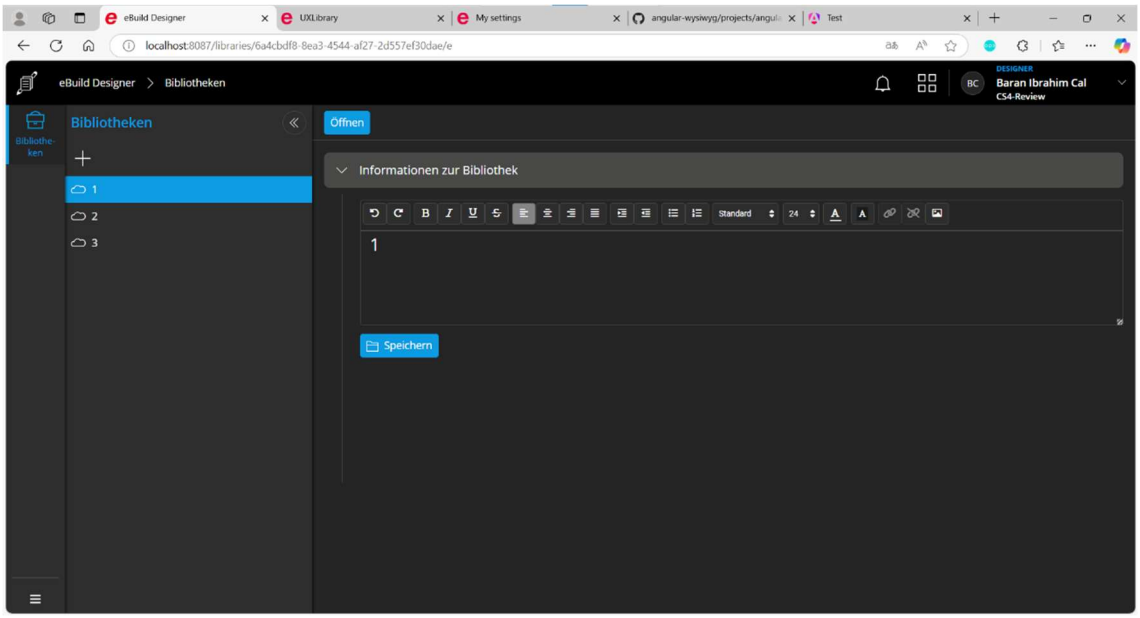

Abbildung 4: Der implementierte Angular WYSIWYG Text Editor

3.2 Verwendete Tools

Für das Projekt wurde ein leistungsfähiger Laptop bereitgestellt, auf dem bereits alle essenziellen Tools und Programme vorinstalliert waren, um einen reibungslosen Einstieg in die Entwicklungsarbeit zu gewährleisten. Die Wahl der Entwicklungsumgebun-

gen richtete sich nach den Anforderungen des jeweiligen Anwendungsbereichs – ob Frontend oder Backend.

Da das Frontend von eBUILD vollständig in Angular entwickelt wurde und das Backend in C#, war die Auswahl an geeigneten Entwicklungsumgebungen begrenzt. Für die Frontend-Entwicklung standen WebStorm und Visual Studio Code zur Verfügung. Trotz vorhandener Erfahrung mit Visual Studio Code aus dem Studium fiel die Wahl auf WebStorm, da es von einigen Teammitgliedern bevorzugt wurde und erweiterte Funktionen für die Angular-Entwicklung bietet.

Im Backend wurde Visual Studio 2022 verwendet, da es optimal auf die Arbeit mit C# abgestimmt ist. Neben der Entwicklung spielte auch regelmäßige Kommunikation eine wichtige Rolle. Da ein Teil der Arbeit remote erfolgte, fanden Meetings und Teamgespräche an diesen Remotetagen über Microsoft Teams statt, was eine effiziente Kommunikation und Zusammenarbeit sicherstellte.

4 Ablauf des Praktikums

Das Praktikum begann offiziell im September, einen Monat nach dem Start als Werkstudent. Bereits am ersten Tag als Werkstudent erhielt ich einen Arbeitslaptop sowie eine neue Tastatur, Maus, ein Headset und einen Rucksack. Zudem wurde mir ein eigener Arbeitsplatz mit zwei zusätzlichen Monitoren zugewiesen. Die Einarbeitungsphase begann ebenfalls schon in dieser Zeit. Während dieser Phase lernte ich nicht nur das Unternehmen EPLAN kennen, sondern auch die Kollegen und die benutzten Tools. Ein wichtiger Teil der Einarbeitung bestand darin, sich selbstständig mit dem Code im Frontend und Backend vertraut zu machen. Dies ermöglichte mir, die Struktur und Funktionsweise der bestehenden Software besser zu verstehen.

Nach Abschluss der Einarbeitungsphase und dem offiziellen Beginn des Praktikums fanden mehrere kleinere Meetings mit dem Betreuer sowie dem Standortleiter in Stuttgart statt. In diesen Besprechungen ging es hauptsächlich um das anstehende Projekt. Dabei wurde versucht, die Anforderungen klar zu definieren und auf ein realistisches Maß zu begrenzen.

Als erste Aufgabe wurde festgelegt, ein kleines, lokales Angular-Projekt zu erstellen, in dem verschiedene Texteditoren getestet und integriert werden sollten. Ziel war es, deren Funktionen und Benutzerfreundlichkeit zu vergleichen, um eine fundierte Entscheidung für den späteren Einsatz zu treffen. Zusätzlich sollte ein kleines Backend in C# entwickelt werden, dass die Inhalte der Editoren speichert und bei Bedarf wieder ausgibt. Diese erste Aufgabe diente als Grundlage für die weitere Umsetzung des Projekts und half dabei, sich tiefer in die technischen Anforderungen einzuarbeiten.

Bei der Auswahl geeigneter Editoren mussten mehrere Anforderungen erfüllt werden. Die Editoren sollten kostenlos und Open Source sein, um eine flexible und nachhaltige Nutzung zu ermöglichen. Außerdem war es essenziell, dass sie mit Angular kompatibel sind. Ursprünglich war die Vorgabe, dass sie moderne Standalone-Komponenten anstelle veralteter Module nutzen sollten. Allerdings stellte sich schnell heraus, dass viele Open-Source-Editoren noch auf das ältere Modul-System setzen, was die Integration erschwerte. Zusätzlich war es wichtig, dass die Editoren regelmäßig Updates erhalten, um langfristige Sicherheit und Funktionalität zu gewährleisten. Auch die Anpassbarkeit des Designs spielte eine zentrale Rolle, da ein nicht individuell anpassbarer Editor für das Projekt ungeeignet gewesen wäre. Besonders relevant war die Art der Textverarbeitung: Während einige Editoren Inhalte in Markdown speichern, arbeitet die Mehrheit mit HTML. Da der bisherige Editor Markdown nutzte, bestand die ursprüngliche Anforderung darin, vorrangig Markdown-Editoren zu evaluieren – oder idealerweise eine Lösung zu finden, die beide Formate unterstützt.

Nach der ersten umfassenden Suche nach geeigneten Markdown-Editoren zeigte sich schnell, dass kaum einer alle gestellten Anforderungen vollständig erfüllte. Dennoch konnten vier potenzielle Editoren identifiziert werden, die zumindest teilweise passten:

- **CKEditor 5**: Dieser Editor unterstützt sowohl Markdown als auch HTML, jedoch nicht gleichzeitig. Man muss sich für ein Format entscheiden.

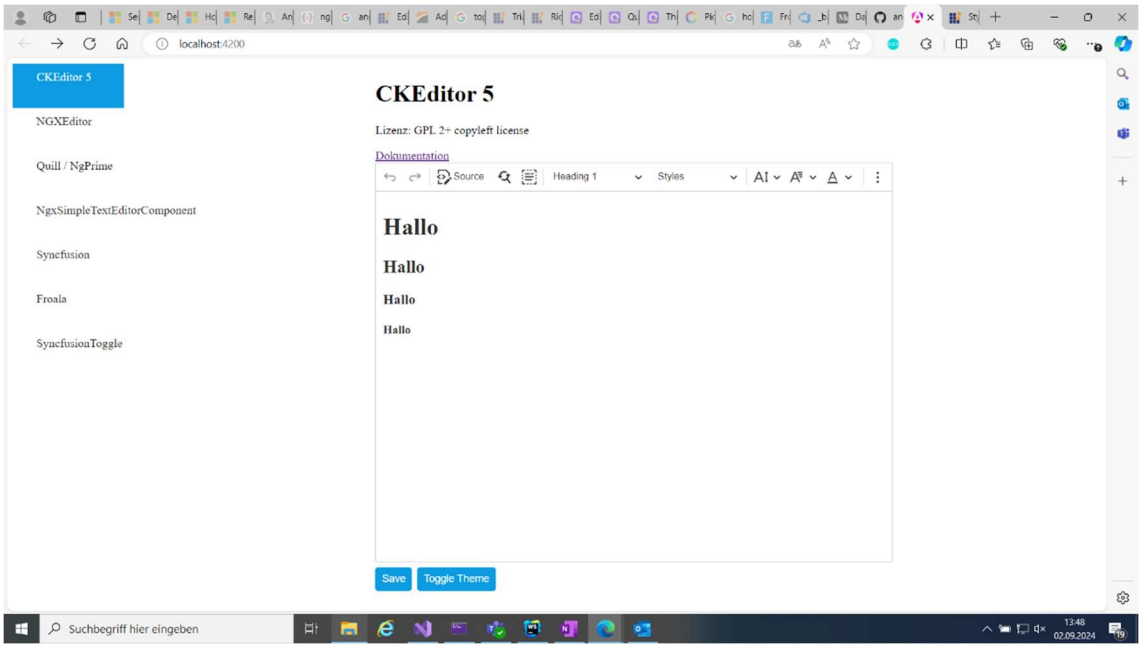

Abbildung 5: Testseite mit allen Editoren (Fokus auf den CKEditor 5)

- **NgxEditor**: Ein reiner HTML-Editor, der keine Markdown-Unterstützung bietet.

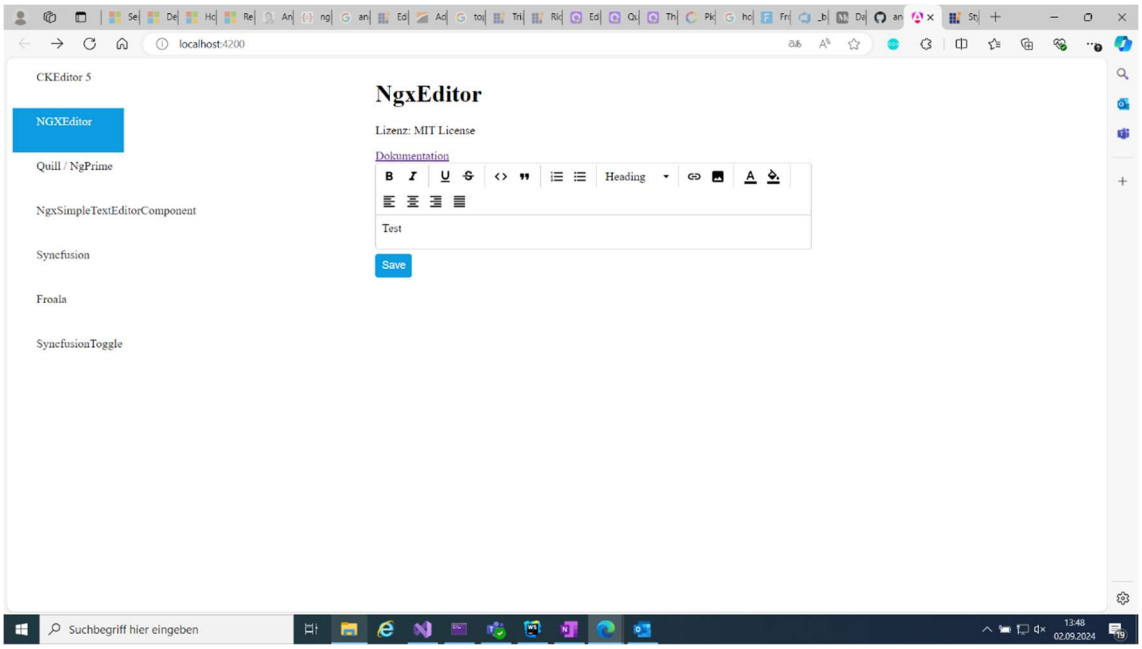

Abbildung 6: Testseite mit allen Editoren (Fokus auf den NgxEditor)

- **NgxSimpleTextEditor**: Ebenfalls ein reiner HTML-Editor mit grundlegenden Funktionen.

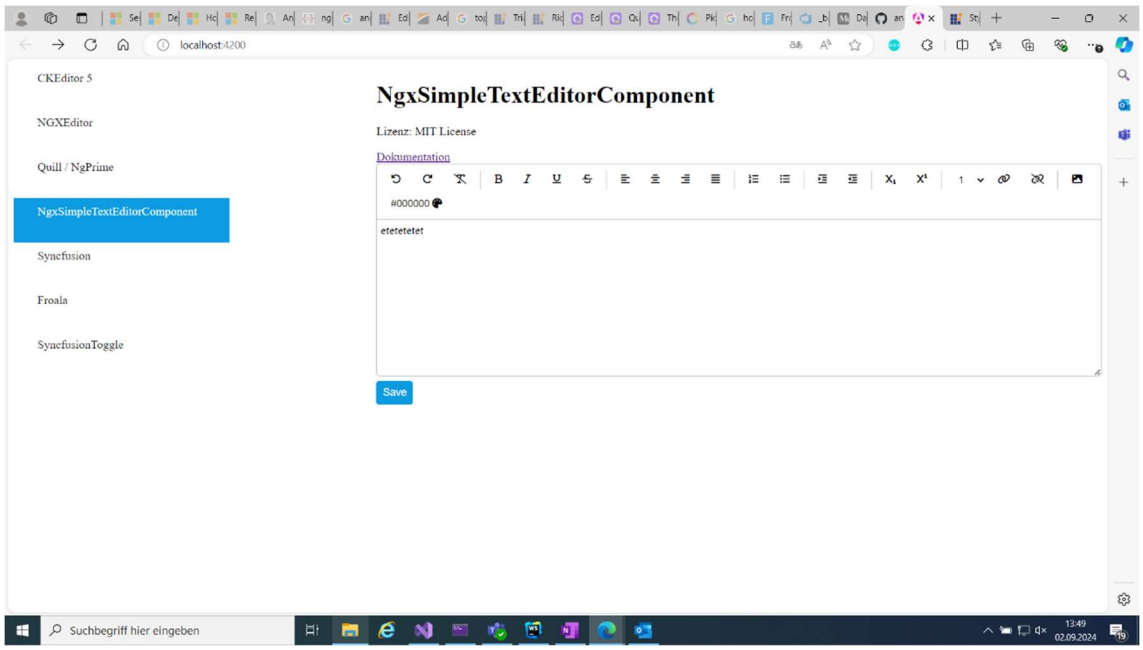

Abbildung 7: Testseite mit allen Editoren (Fokus auf den NgxSimpleTextEditor)

- **Quill**: Arbeitet ausschließlich mit HTML, zeichnet sich aber durch eine aktive Community aus, die regelmäßige Updates und Verbesserungen bereitstellt.

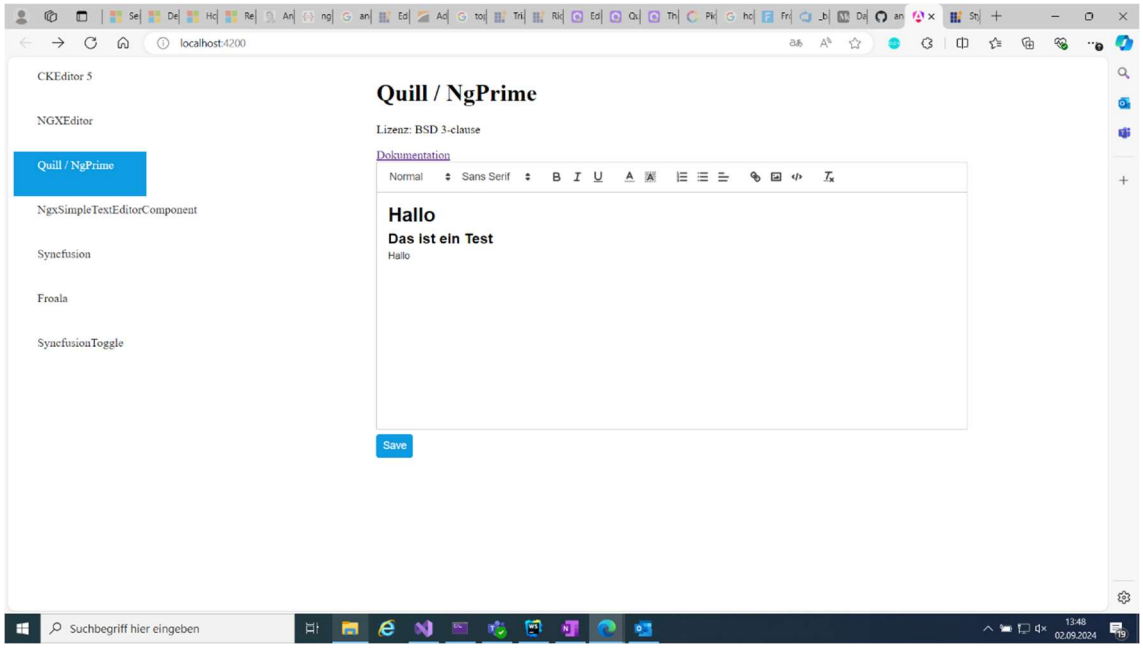

Abbildung 8: Testseite mit allen Editoren (Fokus auf den Quill/NgPrime)

Von diesen vier Editoren kam nur der **CKEditor 5** den Anforderungen am nächsten, da er zumindest zwischen Markdown und HTML im Code wählen lässt. Eine Lösung, die beide Formate gleichzeitig lesen und speichern kann, wurde jedoch nicht gefunden.

Diese Editoren wurden anschließend in eine Testseite integriert, um ihre Funktionalität zu überprüfen. Dabei ergaben sich zunächst Herausforderungen, da viele der Editoren noch auf veraltete Angular-Module anstatt auf moderne Standalone-Komponenten setzten. Durch intensives Experimentieren und Testen gelang es jedoch, auch diese Editoren erfolgreich zu implementieren.

Die Testseite selbst besteht aus einer einzigen Ansicht und nutzt zusätzlich Komponenten des internen EPLAN-UI-Teams, beispielsweise einen Speicher-Button. Auf der Backend-Seite wurden die Anfragen aus dem Frontend verarbeitet, sodass jeder gespeicherte Textinhalt in einer separaten Datei abgelegt wurde.

Nachdem bereits eine erste Suche nach Editoren durchgeführt wurde, soll nun eine weitere Recherche erfolgen, die sich gezielt auf kostenpflichtige Lösungen konzentriert. Dabei geht es nicht nur darum, passende Editoren zu identifizieren, sondern auch die jeweiligen Preismodelle im Detail zu analysieren. Besonders relevant ist die Unterscheidung zwischen Abonnement-basierten Modellen mit monatlichen oder jährlichen Gebühren und einmaligen Lizenzkosten für eine dauerhafte Nutzung. Zusätzlich wird darauf geachtet, ob sie einen echten Mehrwert für das Projekt darstellen. Aspekte wie Benutzerfreundlichkeit, regelmäßige Updates, Support-Angebote sowie Integrationsmöglichkeiten spielen hierbei eine entscheidende Rolle.

Ein weiteres zentrales Auswahlkriterium für geeignete Editoren ist, dass ihre Nutzung nicht durch ein festgelegtes Aufruf- oder Nutzungslimit innerhalb eines bestimmten Zeitraums eingeschränkt wird. Ein Beispiel für einen Editor mit einer solchen Begrenzung ist TinyMCE, der in einigen kostenpflichtigen Tarifen die Anzahl der möglichen Nutzungen pro Monat limitiert. Dies kann in vielen Anwendungsfällen problematisch sein, insbesondere wenn der Editor regelmäßig und intensiv genutzt wird.

Daher liegt der Fokus dieser zweiten Recherche darauf, Editoren zu finden, die uneingeschränkt nutzbar sind – unabhängig davon, wie häufig oder intensiv sie verwendet werden. Eine solche uneingeschränkte Nutzung ist essenziell, um langfristige Skalierbarkeit und Flexibilität zu gewährleisten, ohne dass durch künstliche Beschränkungen Mehrkosten oder technische Einschränkungen entstehen.

In dieser zweiten Suche wurden nur Syncfusion und Froala als potenzielle Editoren gefunden. Diese sollen nun ebenfalls in das Testprojekt integriert und auf ihre Funktionalität hin getestet werden. Syncfusion bietet neben einem Editor noch weitere Komponenten an, die bei EPLAN genutzt werden könnten. Nach der bisherigen Evaluierung zählen CKEditor 5, Syncfusion und Froala zu den besten Editoren, da sie Markdown lesen

als auch schreiben können. Dadurch bieten sie die größte Flexibilität und eignen sich besonders gut für den geplanten Einsatz.

Nachdem alle Editoren integriert wurden, stand das Styling im Mittelpunkt. Zunächst wurde geprüft, ob sich die Editoren optisch anpassen lassen, um das Design von E-PLAN zu übernehmen. Einer der Editoren wurde daraufhin beispielhaft im Dark Mode gestaltet, mit der Option, auch einen Light Mode zu implementieren, sowie einem Light-/Dark-Mode-Switch. Editoren, die sich nicht entsprechend anpassen ließen, wurden als ungeeignet eingestuft. Am Ende konnten die anpassbaren Editoren erfolgreich gestylt und um einen Light-/Dark-Mode erweitert werden, wobei eine Herausforderung darin bestand, die exakten EPLAN-Farben korrekt umzusetzen.

Der CKEditor 5 lässt sich vollständig anpassen, während es bei Syncfusion zunächst Probleme gab, bereits die Hintergrundfarbe der Toolbar zu ändern. Nach Rücksprache mit dem Support konnte dieses Problem jedoch behoben werden, sodass Syncfusion nun ebenfalls vollständig anpassbar ist. Bei Froala hingegen blieb die Anpassung selbst nach Kontakt mit dem Kundensupport erfolglos. Möglicherweise liegt hier ein Fehler vor, der noch weiter überprüft werden müsste. Zusätzlich zeigte sich, dass viele Funktionen des CKEditor 5 oder auch Syncfusion im Markdown-Modus nicht verfügbar sind, da Markdown von Natur aus eingeschränkte Formatierungsmöglichkeiten bietet. In der Abbildung 5 sieht man nochmal alle Editoren, die in der Testseite implementiert wurden. Der Fokus liegt aber auf dem CKEditor im Light Mode.

Nach einer Diskussion mit dem Betreuer wurden die Anforderungen an die Editoren angepasst. Syncfusion wurde aufgrund der hohen Kosten aus der Auswahl genommen. Die Preise für Syncfusion sind wie folgt: Ein Team von bis zu 5 Entwicklern kostet 395$ pro Monat, ein Team von bis zu 10 Entwicklern 695$ pro Monat, und für Teams mit mehr als 10 Entwicklern muss eine individuelle Preisanfrage gestellt werden. Diese Preise machten Syncfusion zu teuer, weshalb eine Eigenentwicklung die bessere Option darstellt, bevor auf Syncfusion zurückgegriffen werden würde. Stattdessen wurde beschlossen, einen kostenlosen und Open-Source-Editor zu bevorzugen, da diese Lösungen langfristig kostengünstiger und flexibler sind. Ein WYSIWYG-Editor mit HTML-Unterstützung wird nun bevorzugt, anstatt eines Markdown-basierten Editors.

Ziel dieser Entscheidung ist es, die Übersicht und Struktur zu verbessern, sodass sich sowohl neue als auch erfahrene Nutzer besser zurechtfinden und die Zusammenarbeit an einer Bibliothek effizienter wird. Gleichzeitig soll der Programmieraufwand minimiert werden, da die Nutzer in der Regel Elektrotechniker und keine Programmierer sind. Um diesen Anforderungen gerecht zu werden, wird Markdown durch HTML ersetzt, um den Nutzern den Umgang mit HTML zu erleichtern, ohne dass sie sich mit komplexen Programmieraufgaben auseinandersetzen müssen.

Da die bisherigen Daten im Markdown-Format gespeichert wurden, ist eine Konvertierung zu HTML erforderlich. Diese Konvertierung kann entweder im Frontend oder im Backend erfolgen. Der Editor sollte grundlegende Funktionen wie Fettdruck, Kursivschrift, Unterstreichung und Durchstreichung unterstützen. Zudem sollte er die Möglichkeit bieten, nummerierte und ungeordnete Listen zu erstellen. Weitere wichtige Funktionen umfassen die Anpassung von Hintergrund- und Textfarben, das Erstellen von Absätzen, die Änderung der Text- und Schriftgrößen sowie das Einfügen von Bildern und URLs. Ein zusätzliches Kriterium ist nicht nur das Styling des Editors, wobei die Integration eines Light- und Dark-Mode gewünscht wird, sondern ebenso sollte es möglich sein, eigene Funktionen in die Toolbar des Editors einzufügen.

Die größte Herausforderung besteht darin, einen kostenlosen und Open-Source-Editor zu finden, der all diese Funktionen bietet und gleichzeitig anpassbar ist. Viele kostenlose Editoren bieten nur begrenzte Styling-Möglichkeiten, und es bleibt fraglich, ob es realistisch ist, eigene Funktionen in diese Editoren zu integrieren.

Schließlich wurde eine Möglichkeit gefunden, alle Editoren zu stylen, sodass nun alle HTML-Editoren in Betracht gezogen werden können. Die Styling-Optionen wurden auf zwei Arten umgesetzt: Entweder in der Haupt-style.scss-Datei oder innerhalb der Komponente durch die Verwendung von :host::ng-deep in der style.css, was zwar veraltet gilt, aber weiterhin genutzt werden kann. Die bevorzugte Methode ist das Styling über die Haupt-style.scss, da die andere Option in Zukunft möglicherweise nicht mehr unterstützt wird und daher keine langfristige Lösung darstellt. In Abbildung 9 sieht man diese Haupt CSS-Datei, in der alle Editoren angepasst wurden.

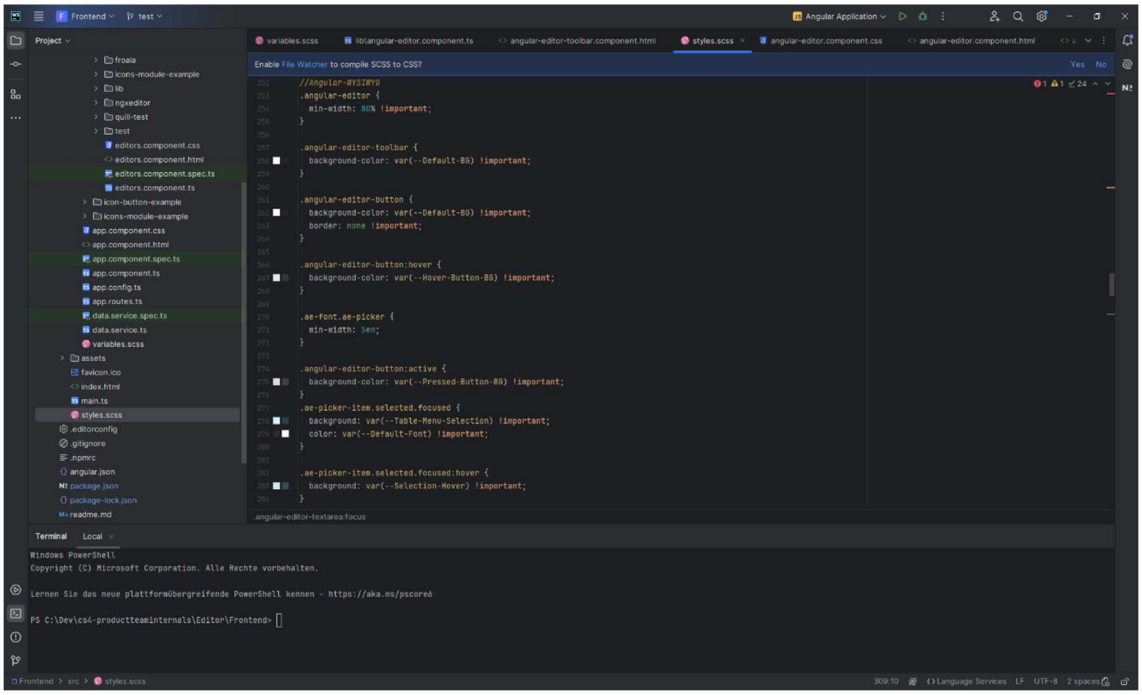

Abbildung 9: style.css

In die engere Auswahl der Editoren kamen somit NgxEditor, Jodit, Angular-WYSIWYG und Quill. Diese Editoren wurden für eine Präsentation vorbereitet, um sie dem Team vorzustellen. Während und nach der Präsentation kamen verschiedene Fragen auf, die nun geklärt werden müssen. Es gab Fragen zum Speicherformat der Editoren. Es wurde festgestellt, dass die Editoren Bilder im Base64-Format speichern.

Ein weiteres Thema war die Konvertierung von Markdown zu HTML. Diese ist grundsätzlich möglich, allerdings muss entschieden werden, ob die Konvertierung im Backend oder Frontend erfolgen soll. Eine Lösung im Backend gilt als sicherer, während eine Lösung im Frontend schneller und interaktiver wäre. Daher wird angestrebt, beide Ansätze zu testen. Ein Problem im Frontend besteht darin, dass es nicht einfach ist, zu überprüfen, ob eine Variable im Markdown-Format vorliegt, um sie dann nur bei Bedarf in HTML zu konvertieren. Daraufhin kam die Frage auf, ob eine Konvertierung von Markdown zu HTML auch etwas Sinnvolles ausgibt. Daher soll ein Test mit Beispieldaten durchgeführt werden, um die Konvertierung zu überprüfen.

Schließlich wurde auch die Konfiguration der Toolbar angesprochen. Es besteht der Wunsch, die Toolbar nach Bedarf anzupassen, um Funktionen hinzuzufügen oder zu entfernen.

Nach weiteren Tests wurde vom Betreuer entschieden, die Konvertierung im Frontend durchzuführen. Dies ermöglicht eine schnellere und interaktive Umsetzung ohne zusätzliche Backend-Belastung. Das Ergebnis der Konvertierung weicht nur minimal vom Original ab und ist daher ausreichend präzise. Zudem hat sich gezeigt, dass die Anpassung der Toolbar bei allen in Betracht gezogenen Editoren problemlos möglich ist.

Der Betreuer entschied sich schließlich für Angular-WYSIWYG als Editor, der nun in einem neuen Branch des eBUILD- Repositories integriert werden sollte. Statt einer einfachen Installation über npm install wurden die Editor-Dateien direkt in das eBUILD-Projekt eingefügt. Dies brachte zahlreiche Fehler mit sich, die jedoch mit viel Aufwand behoben wurden, bis der Editor schließlich erfolgreich lief. Auf die einzelnen Fehler wird an dieser Stelle nicht näher eingegangen.

Durch diese direkte Integration war es möglich, vollständig auf alle Styles zuzugreifen und sogar die Icons anzupassen. Anschließend musste die einmalige Konvertierung der bestehenden Markdown-Daten in HTML umgesetzt werden. Dazu wurde im Backend ein Enum mit den Werten Markdown und HTML erstellt, wobei Markdown als Standardwert gesetzt wurde. Beim Laden der Bibliothek im Frontend wird dieser Wert abgefragt – wenn der Inhalt noch in Markdown vorliegt, erfolgt die Konvertierung zu HTML, und der Enum-Wert wird entsprechend auf HTML gesetzt. Ist der Inhalt bereits in HTML gespeichert, gibt es keine weitere Konvertierung.

Zum Abschluss wurde der Editor an das EPLAN-Design angepasst, sodass er sowohl im Light- als auch im Dark-Mode entsprechend den Einstellungen korrekt dargestellt wird. Den Light Mode kann man in Abbildung 6 sehen und den Dark Mode in Abbildung 7.

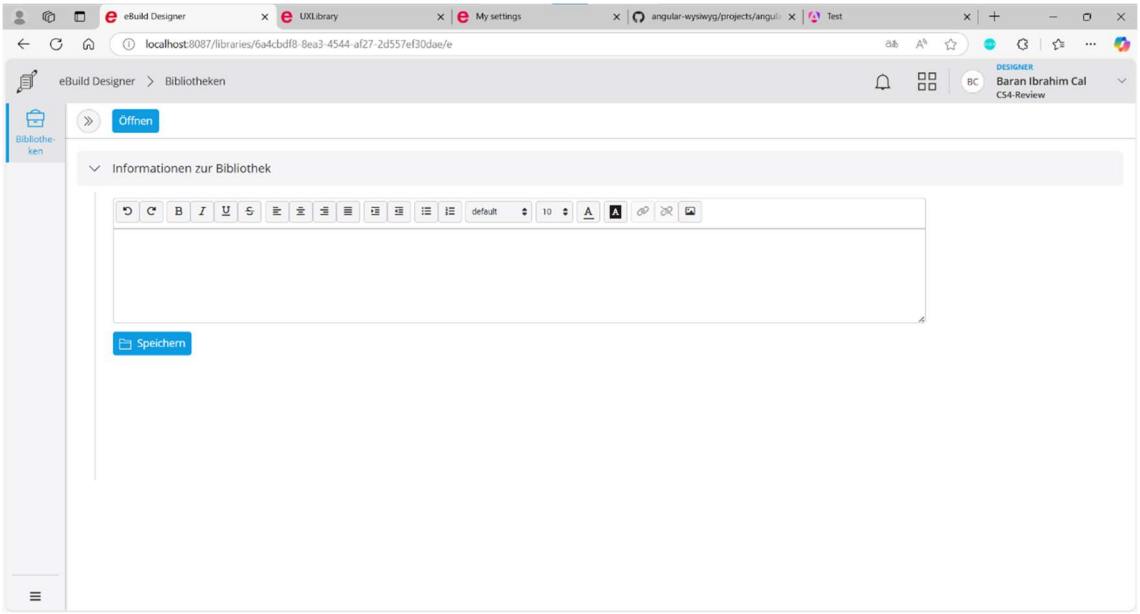

Abbildung 10: Implementierter Angular-WYSIWYG Editor im Light Mode

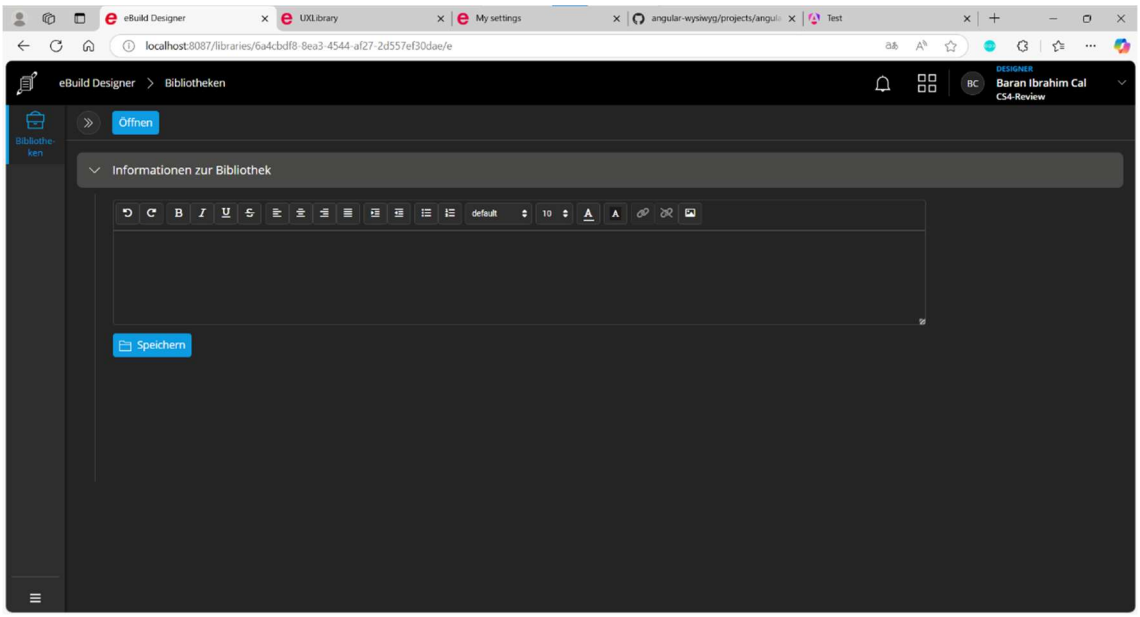

Abbildung 11: Implementierter Angular-WYSIWYG Editor im Dark Mode

5 Zusammenfassung

Während des Praktikums lag der Fokus auf der Implementierung von verschiedenen Editoren für die Bibliotheksbeschreibung im eBUILD Designer. Dabei wurde der bestehende Texteditor für die Bibliotheken des Designers durch eine moderne Alternative ersetzt. Der bisherige Editor basierte auf Markdown und war in seiner Funktionalität stark eingeschränkt, was die Bearbeitung von Texten umständlich machte. Ziel war es, eine benutzerfreundliche Lösung mit erweiterten Funktionen zu integrieren, die eine effizientere Textbearbeitung ermöglicht. Zunächst wurden verschiedene Markdown- und HTML-Editoren evaluiert. Dabei zeigte sich, dass viele kostenlose Editoren nicht alle erforderlichen Funktionen boten. Auch kostenpflichtige Editoren wie Syncfusion und Froala wurden getestet, jedoch aufgrund hoher Kosten oder mangelnder Anpassbarkeit ausgeschlossen. Schließlich fiel die Wahl auf den Angular WYSIWYG Text Editor, der eine umfangreiche Funktionalität bietet und sich in das bestehende System integrieren ließ. Ein zentraler Bestandteil der Umsetzung war die Konvertierung von Markdown zu HTML, um bestehende Kundendaten weiterhin nutzbar zu machen. Diese Konvertierung wurde einmalig durchgeführt, sodass zukünftige Texte direkt im HTML-Format gespeichert werden. Die Integration des neuen Editors stellte einige technische Herausforderungen dar, insbesondere bei der direkten Implementierung der Editor-Dateien, um vollständige Kontrolle über das Styling und die Icons zu behalten. Trotz anfänglicher Schwierigkeiten wurde der Editor erfolgreich in das *eBUILD*-Projekt integriert und an das EPLAN-Design angepasst.

6 Ausblick

Das Projekt befindet sich noch in der Entwicklungsphase und ist noch nicht vollständig abgeschlossen. Der nächste Schritt besteht darin, das Projekt in die Testumgebung zu überführen. Dafür müssen zunächst mehrere Pipelines erfolgreich durchlaufen und validiert werden, um eine reibungslose Integration sicherzustellen.

Sobald der Editor fertiggestellt ist, stehen weitere Erweiterungen und Optimierungen an. Ein zentrales zukünftiges Entwicklungsziel ist die Implementierung eines leistungsfähigen Formeleditors.

Zusätzlich fehlen noch umfassende Tests, um die Stabilität, Performance und Sicherheit des Projekts zu gewährleisten. Dazu gehören sowohl Unit-Tests für einzelne Komponenten als auch Integrationstests, um das Zusammenspiel verschiedener Module zu überprüfen. Automatisierte Tests sollen sicherstellen, dass neue Funktionen keine bestehenden Prozesse beeinträchtigen, während manuelle Tests dabei helfen, potenzielle Benutzerprobleme frühzeitig zu erkennen.

Literaturverzeichnis

EPLAN Gmbh & Co. KG. 2024a. *EPLAN feiert 40 Jahre: Vom Start-up zum Marktführer.* [Online] 2024a. [Zitat vom: 4. Februar 2025.] https://www.eplan.de/unternehmen/news/eplan-feiert-40-jahre-vom-start-up-zum-marktfuhrer/.

EPLAN GmbH & Co. KG. 2024d. *EPLAN - Efficient Engineering.* [Online] 2024d. [Zitat vom: 5. Februar 2025.] https://www.eplan.de/.

EPLAN GmbH & Co. KG 2024b. *40 Jahre Eplan: das Unternehmen – und die Geschichten dahinter.* [Online] 4. 07 2024b. [Zitat vom: 4. Februar 2025.] https://www.eplan.com/de-de/blog/backstage/40-jahre-eplan-das-unternehmen-und-die-geschichten-dahinter.

EPLAN GmbH & Co. KG 2024e. *EPLAN eBUILD - Neue Methotik für Ihr Engineering.* [Online] 2024e. [Zitat vom: 5. Februar 2025.] https://www.eplan.de/loesungen/eplan-ebuild/.

EPLAN GmbH & Co. KG 2024f. *EPLAN eBUILD Automatisiertes Engineering in der Cloud.* [Online] 2024f. [Zitat vom: 6. Februar 2025.] https://www.eplan.de/loesungen/eplan-ebuild/eplan-ebuild/.

EPLAN GmbH & Co. KG 2024c. *EPLAN GmbH & Co. KG - Headquarter Monheim am Rhein.* [Online] 2024c. [Zitat vom: 4. Februar 2025.] https://www.eplan.de/unternehmen/standorte/europa/deutschland/headquarter-monheim/.

Ehrenwörtliche Erklärung

Name: Cal	Vorname: Baran Ibrahim
Matrikel-Nr.: 768960	Studiengang: Medieninformatik

Hiermit versichere ich, Baran Ibrahim, Cal, dass ich den vorliegenden Praxissemesterbericht mit dem Titel Entwicklung von Software in der EPLAN GmbH & Co. KG selbständig und ohne fremde Hilfe verfasst und keine anderen als die angegebene Literatur und Hilfsmittel verwendet habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinne nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quelle kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden.

Stuttgart, 07.03.25 _____

Ort, Datum

Unterschrift