



HTL Dornbirn

Höhere Lehranstalt für Wirtschaftsingenieurwesen
Ausbildungsschwerpunkt Betriebsinformatik

Projekt

„MasMessenger“

Eine benutzerfreundliche Kommunikationsplattform für
effizienten Austausch und Vernetzung

Ausgeführt im Schuljahr 2024/25 von:

Maximilian Gmeinder

Ali Dadak

Matej Grkovic

Auftraggeber:

Michael Leeb

Dornbirn, am 22.02.2025

Dieses Projekt „MasMessenger“ handelt sich um die Entwicklung einer modernen Webanwendung zur Verbesserung der digitalen Kommunikation. Die Anwendung ermöglicht es Nutzern weltweit miteinander zu kommunizieren, sowie eine Registrierung und die Erstellung eines persönlichen Kontos. Nach erfolgreicher Anmeldung können Benutzer sowohl in Einzel- als auch in Gruppenchats miteinander interagieren.

Die Umsetzung dieser Arbeit erfolgt größtenteils in Visual Studio unter Verwendung von Vue.js mit dem Framework Vuetify, wodurch eine ansprechende und benutzerfreundliche Oberfläche geschaffen wird. Dabei liegt der Fokus darauf, die Anwendung einfach und verständlich zu gestalten, sodass sich Nutzer schnell zurecht finden - unabhängig von ihrem Alter oder ihren technischen Vorkenntnissen und Erfahrungen. Zudem wird darauf geachtet, dass Nachrichten und Daten zuverlässig und ohne Verzögerung verarbeitet werden.

Durch diese neue Webanwendung wird eine moderne, flexible und benutzerfreundliche Kommunikationsplattform bereitgestellt, die den Austausch zwischen Nutzern vereinfacht und optimiert – ohne Einschränkung von deren Entfernung, solange eine Internetverbindung vorhanden ist.

Abstract English

This project “MasMessenger” is about the development of a modern web application to improve digital communication. The application allows users worldwide to communicate with each other, as well as register and create a personal account. After successful registration, users can interact with each other in both individual and group chats.

The implementation of this work is largely carried out in Visual Studio using Vue.js with the Vuetify framework, creating an appealing and user-friendly interface. The focus is on making the application simple and easy to understand so that users can quickly find their way around - regardless of their age or previous technical knowledge and experience. Care is also taken to ensure that messages and data are processed reliably and without delay.

This new web application provides a modern, flexible and user-friendly communication platform that simplifies and optimizes the exchange between users - regardless of their distance, as long as an Internet connection is available.

Vorwort

Das vor Ihnen liegende Projekt „MasMessenger“ wurde von Maximilian Gmeinder, Ali Dadak und Matej Grkovic verfasst. Die Arbeit wurde im Rahmen des Schulunterrichts Softwareentwicklung und Produktmanagement an der HTL Dornbirn ausgearbeitet. Die Umsetzung dieses Projekts sowie das Schreiben der Arbeit erfolgte vom November 2024 bis Februar 2025.

Da wir als Team häufig „Social Media“ nutzen und regelmäßig über Plattformen miteinander sowie mit anderen Personen kommunizieren, sind uns einige Aspekte aufgefallen, die uns bei den bereits vorhandenen Kommunikationsplattformen stören. Sei es einerseits die unübersichtliche Struktur, eingeschränkte Anpassungsmöglichkeit oder die fehlende Privatsphäre und Sicherheit – diese Erfahrungen haben uns dazu inspiriert, eine eigene Kommunikationswebseite zu entwickeln.

Das Ziel unseres Projektes war von Anfang an klar, wir wollten, wie im Abstract schon erklärt eine benutzerfreundliche Plattform erschaffen, die eine einfache und flexible Kommunikation mit seinen Mitmenschen ermöglicht. Dabei haben wir uns bewusst für Technologien entschieden, die eine einfache Nutzung für alle Altersgruppen gewährleistet.

Inhaltsverzeichnis

1. Einleitung.....	6
2. Projektmanagement	7
2.1 Projektauftrag	7
2.2 Projektzieleplan.....	8
2.3 Projektumweltanalyse	9
2.4 Objektstrukturplan	11
2.5 Projektstrukturplan.....	12
2.6 Projektorganigramm-Grafik.....	13
2.7 Meilensteinplan	14
3. Tools & Technologien	15
3.1 Vue & Vuetify.....	15
3.2 C#	16
3.3 Swagger	16
3.4 SQL-Datenbanken.....	17
3.5 DevExpress XPO.....	18
3.6 Firebase	19
4. Theoretisches Konzept.....	20
4.1 Designvorschläge	21
5. Praktische Umsetzung	22
5.1 GitHub-Repository erstellen	22
5.2 Einrichten der Datenbank	23
5.3 CRUD-Operation mit Swagger	24
5.4 Programmieren der Weboberfläche	25
5.5 Wechsel von SQL-Datenbank zu Firebase	26
5.6 Fertigstellen des Projektes.....	27
6. Aufgetretene Probleme	28
7. Verwendete Hilfen	29
8. Quellenverzeichnis	30
9. Abbildungsverzeichnis	31
10. Tabellenverzeichnis.....	31

1. Einleitung

Die Webanwendung wird im Frontend mit Vue.js und dem dazu passenden Framework Vuetify in Visual Studio Code entwickelt, um eine moderne und benutzerfreundliche Oberfläche zu schaffen. Das Backend basiert auf .NET8 und wird mit C# in Visual Studio programmiert. Um die Arbeit mit den CRUD-Operationen (Create, Read, Update und Delet) zu erleichtern und eine strukturierte API-Dokumentation bereitzustellen, kommt Swagger zum Einsatz.

Die Daten werden innerhalb einer Microsoft SQL-Datenbank gespeichert. Dabei sorgt devExpress XPO für eine effiziente Verwaltung der Benutzer, sowie der Nachrichten, indem jedem Nutzer automatisch eine eindeutige ID zugewiesen wird. Während der Entwicklungsphase läuft die Anwendung lokal, bevor sie nach erfolgreichem Testen über, dass Azure Portal in eine Cloud hochgeladen wird. Dies gilt ebenfalls auch für die Webanwendung. Diese wird durch den Einsatz von Docker über das Azure Portal gehostet, dadurch ermöglichen wir eine freizugängliche Kommunikationsplattform.

2. Projektmanagement

2.1 Projektauftrag



Projektauftrag	
Projektstartereignis: Unterrichtsaufgabe	Projektstarttermin: 04.11.2024
Inhaltliches Projektendereignis: Erreichen der Projektziele Formales Projektendereignis: Abgabe der Teamsaufgabe mit Präsentation	Projektendtermine: <ul style="list-style-type: none">• 21.01.2025• 04.02.2025
Projektziele: Ziel dieses Projekts ist die Entwicklung einer modernen Webanwendung, die Nutzern eine Anmeldung und Kontoerstellung ermöglicht. Nach Registrierung können Benutzer in Einzel- und Gruppenchats kommunizieren. Die Anwendung wird in Visual Studio mit Vue.js und Vuetify für ein ansprechendes, benutzerfreundliches Interface umgesetzt.	Nicht-Projektziele: <ul style="list-style-type: none">• App• Kosten• Kein Chat-To-Stranger• Keine Werbeanzeigen• Keine Öffentliche Profile
Hauptaufgaben (Projektphasen): PM Das Projektmanagement umfasst die Planung und Organisation aller Ressourcen sowie die Definition der Projektziele und des Zeitplans. Analyse Die Analysephase ermittelt und dokumentiert alle funktionalen und technischen Anforderungen für die Benutzer- und Admin-Funktionen. Dazu gehört ebenfalls einen Designvorschlag der Nutzeroberfläche. Coding In der Coding-Phase wird die Benutzeroberfläche in Vue.js/Vuetify entwickelt und mit der SQL-Datenbank verbunden. Die Daten werden ebenfalls mit Hilfe von devExpress XPO abgespeichert. Das Backend wird in Visual Studio mit C# erarbeitet (CRUD-Operationen der User). Testen & Dokumentation Es werden umfassende Tests durchgeführt, um die Funktionalität und Sicherheit der Anwendung zu gewährleisten.	
ProjektauftraggeberIn: Leeb Michael	ProjektleiterIn: Gmeinder Maximilian
Projektteam: Ali Dadak & Matej Grkovic	
 Michael Leeb, (ProjektauftraggeberIn)	 Maximilian Gmeinder, (ProjektleiterIn)

Tabelle 1: Projektauftrag

2.2 Projektzieleplan

Projektzieleplan	
Zielart	Projektziele
Hauptziele	<ul style="list-style-type: none">• Entwicklung einer modernen Webanwendung• Kontoerstellung: Login & Registrierung soll für den Nutzer möglich sein• Nutzer sollen in Einzel & Gruppenchats kommunizieren können• Benutzerfreundliche Weboberfläche
Nebenziele	<ul style="list-style-type: none">• Kein Chat-To-Stranger• Keine Öffentliche Profile
Nicht-Ziele	<ul style="list-style-type: none">• App• Kosten• Werbeanzeigen

Tabelle 2: Projektzieleplan

2.3 Projektumweltanalyse

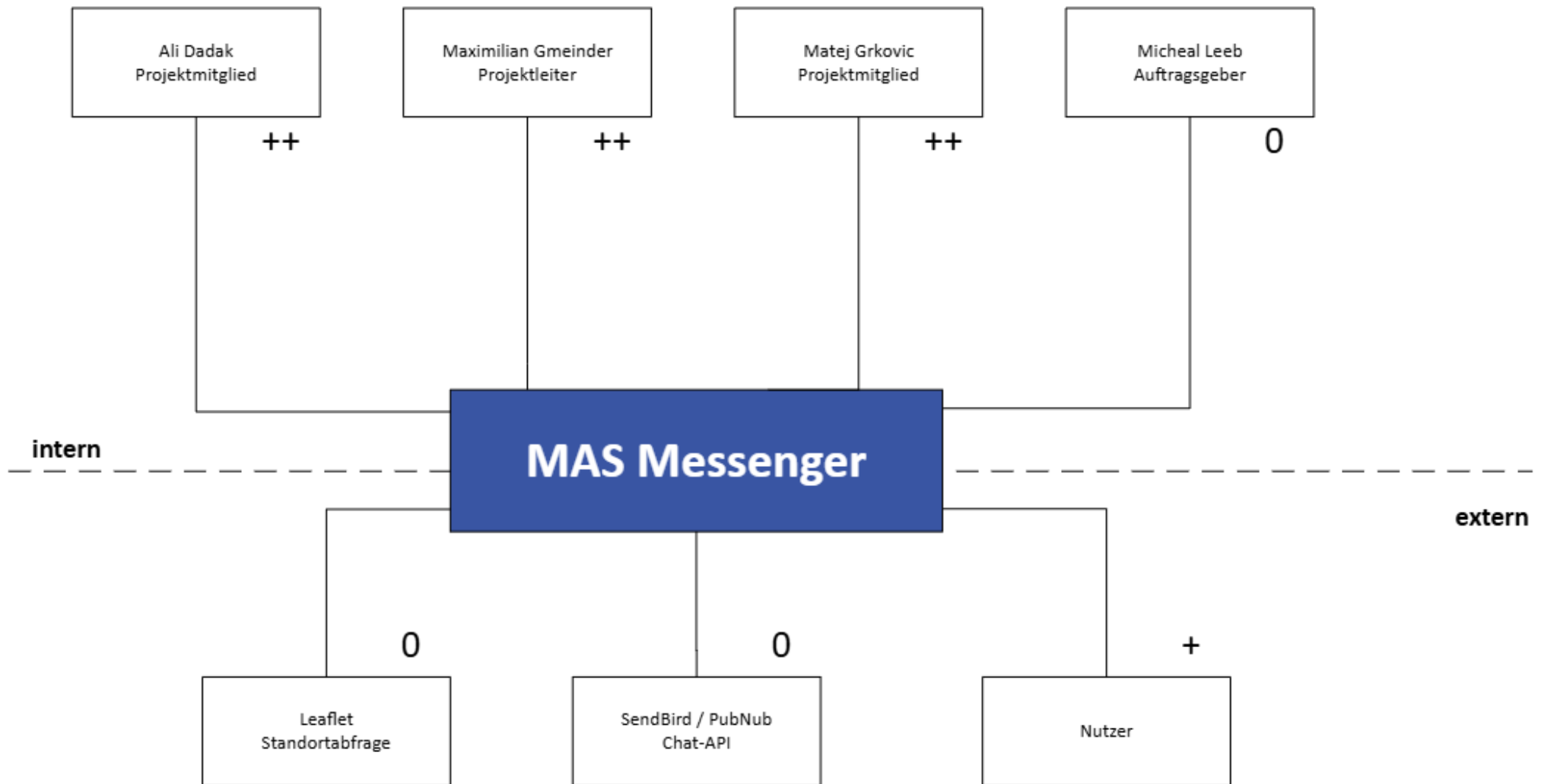


Abbildung 1: Projektumweltanalyse

Legende:

++ → sehr positiv

+ → positiv

0 → neutral

- → negativ

Projektumweltanalyse – Beziehungen		
Umwelten	Beziehung	Maßnahmen
Projektteam	Wir als Projektteam werden viel Zeit miteinander verbringen, was auf Dauer zu Konflikten führen könnte, besonders in Situationen bei denen Meinungsverschiedenheiten auftreten können.	Wir als Team werden ständig auf einen respektvollen Umgang untereinander achten. Probleme bzw. Meinungsverschiedenheiten werden an- und ausgesprochen, damit diese den weiteren Verlauf des Projektes nicht beeinflussen. Abgesehen davon werden Vorschläge von den einzelnen Teammitglieder berücksichtigt und je nach Gebrauch angewendet.
Projektauftraggeber	Der Auftraggeber in unserem Fall Herr Professor Leeb kann bei Problemen helfen, wenn unklare Situationen vorliegen.	Bei Problemen wird der Auftraggeber informiert. Das Auftreten von Problemen wird von Beginn an in der Zeitplanung berücksichtigt, ebenfalls wird darauf geachtet Projektzwischenstände vorzuweisen.
Nutzer	Die Nutzer haben eine positive Einstellung gegenüber unserem Projekt und sind an einer guten Erfahrung interessiert.	Im Falle einer Veröffentlichung Wir werden regelmäßig Feedback von Nutzern einholen, um sicherzustellen, dass ihre Bedürfnisse und Erwartungen erfüllt werden.
Unternehmen der verwendeten Tools	Die Unternehmen der verwendeten Tools spielen eine unterstützende Rolle in unserem Projekt, indem sie Ressourcen bereitstellen, die unsere Entwicklung erleichtern.	Es wird sichergestellt, dass die verwendenden Tools den Projektanforderungen entsprechen und keine negativen Auswirkungen auf den Projektverlauf haben, ebenfalls werden Überprüfungen durchgeführt, um ihre Effizienz zu gewährleisten.

Tabelle 3: Projektumweltanalyse-Beziehungen

2.4 Objektstrukturplan

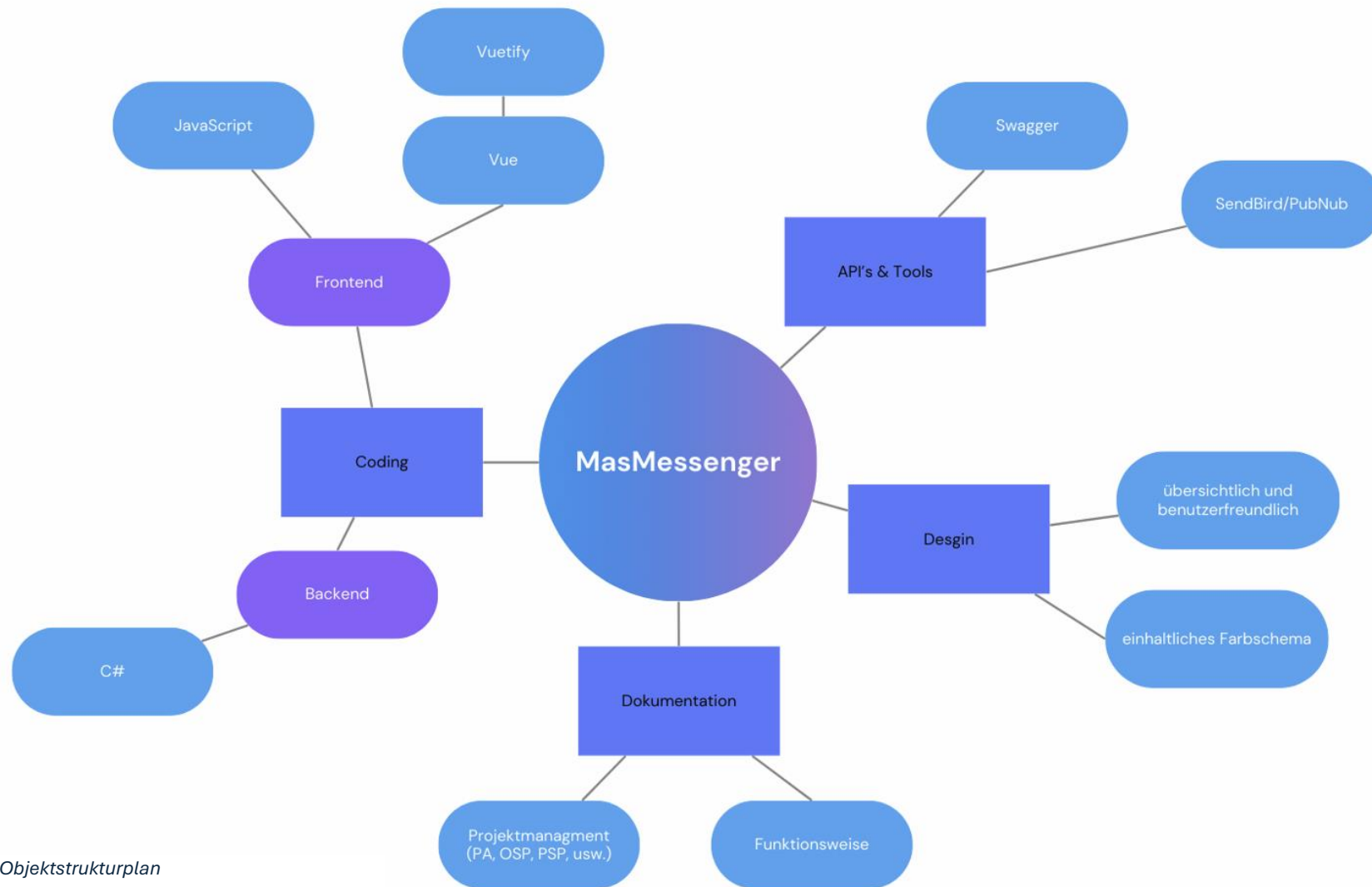


Abbildung 2: Objektstrukturplan

2.5 Projektstrukturplan

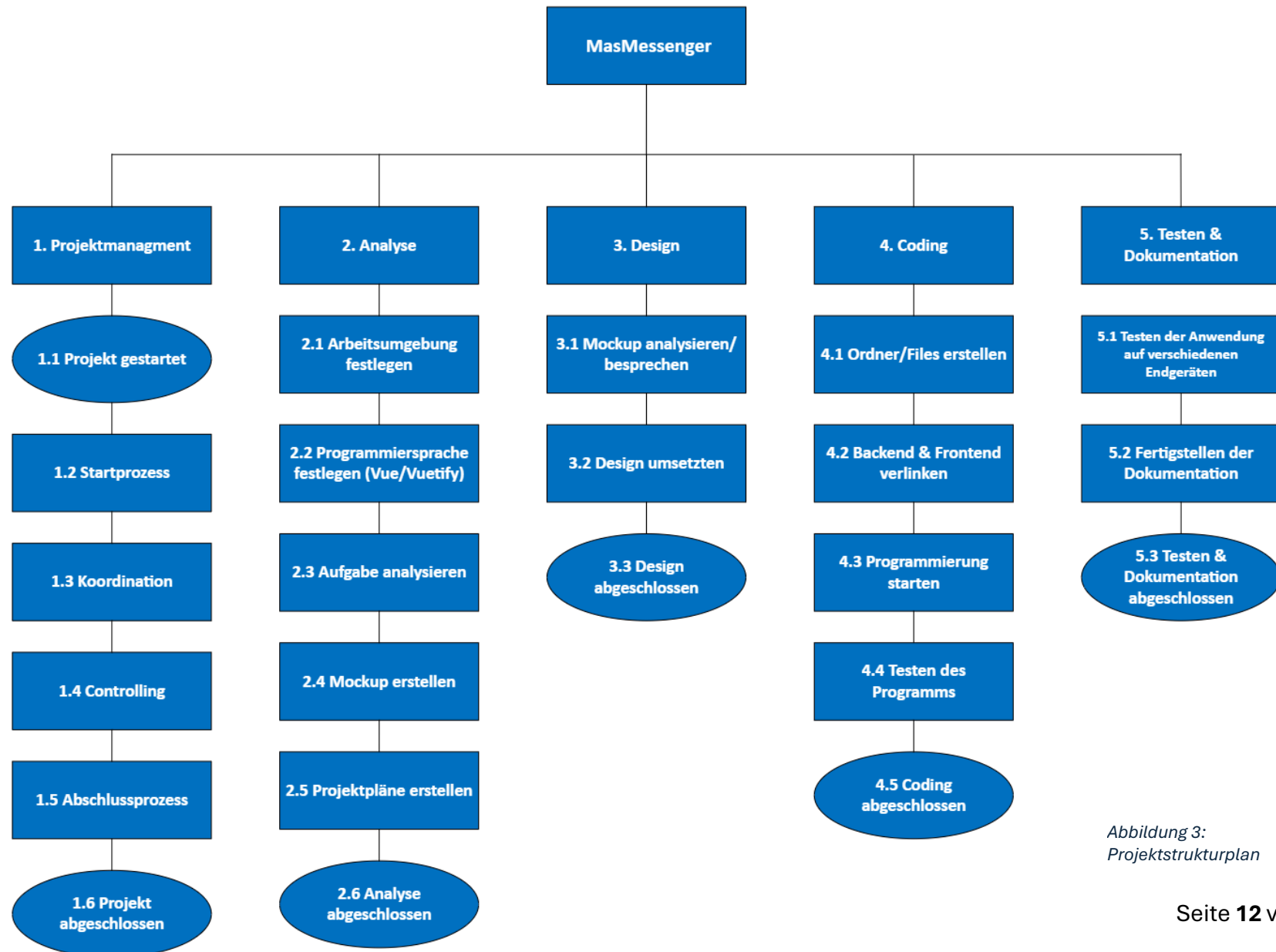


Abbildung 3:
Projektstrukturplan

2.6 Projektorganigramm-Grafik

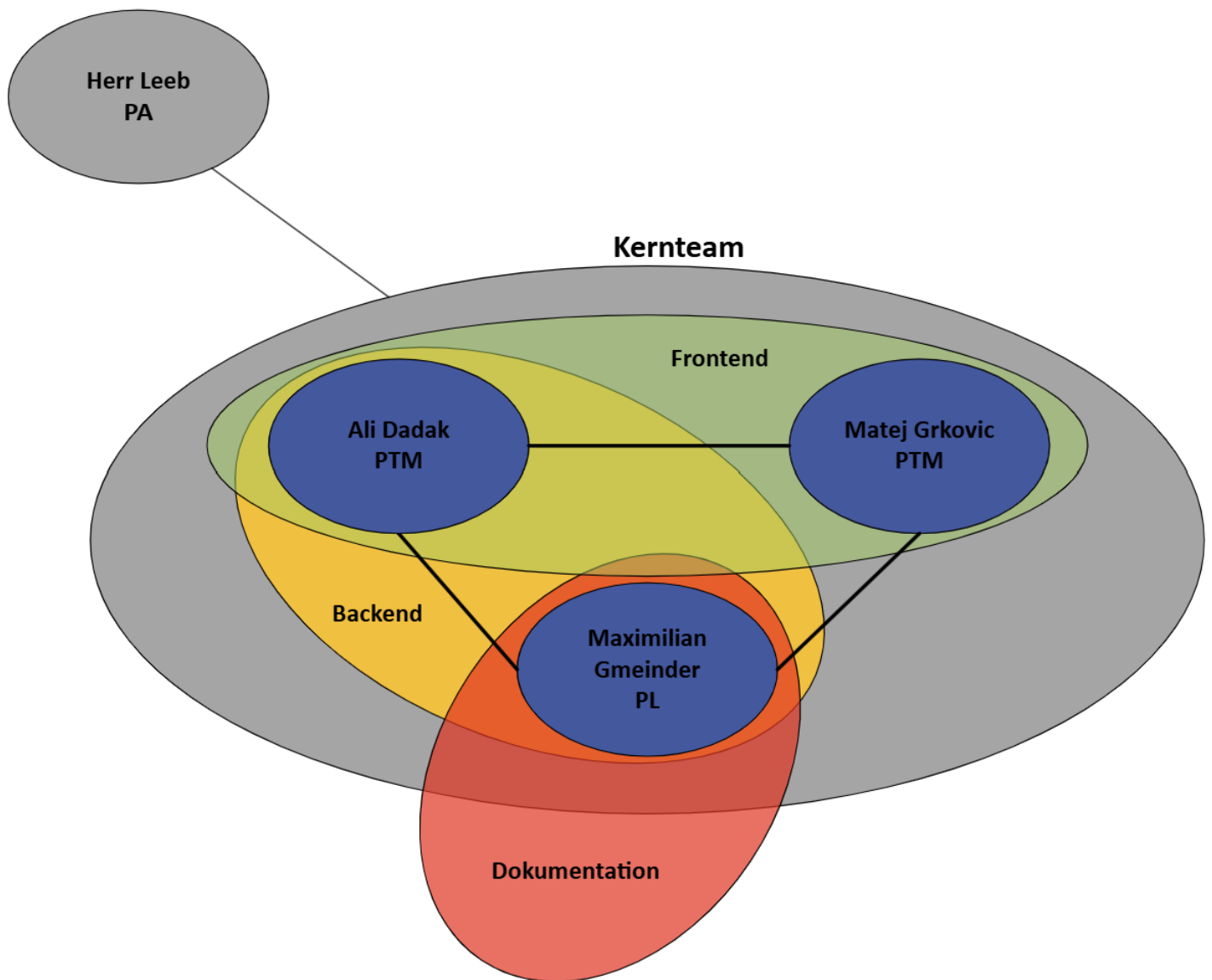


Abbildung 4: Projektorganigramm

PL...Projektleiter

PT...Projektmitglied

PA...Projektauftraggeber

2.7 Meilensteinplan

Projektmeilensteinplan			
PSP-Code	Meilenstein	Aktuelle Plantermine	Ist-Termine
1.1	Projekt gestartet	4.11.2024	4.11.2024
2.4	Mockup erstellt	6.11.2024	11.11.2024
2.6	Analyse abgeschlossen	13.11.2024	13.11.2024
3.3	Design abgeschlossen	18.11.2024	18.11.2024
4.3	Programmierung starten	20.11.2024	20.11.2024
4.5	Coding abgeschlossen	20.01.2025	17.02.2025
5.3	Testen & Dokumentation abgeschlossen	27.01.2025	19.02.2025
1.6	Projekt abgeschlossen	4.02.2025	23.02.2025

Tabelle 4: Projektmeilensteinplan

3. Tools & Technologien

3.1 Vue & Vuetify

Vue.js ist ein modernes und benutzerfreundliches JavaScript-Framework, das für die Entwicklung von interaktiven und dynamischen Webseiten erleichtert. Vue sorgt für eine effizientere Erstellung von Benutzeroberflächen und Single-Page-Webanwendungen. Dabei bietet das Framework eine klare Struktur und vereinfacht das Arbeiten mit Daten und Benutzerinteraktionen.

Das JavaScript-Framework wird häufig für interaktive Webanwendungen eingesetzt, darunter Dashboards, Echtzeit-Chats oder komplexere Benutzeroberflächen. Der größte Vorteil den Vue.js mit sich bringt ist die praktische komponentenbasierte Architektur: die Möglichkeit flexible wiederwendbare Bausteine zu erstellen, sorgt einerseits für eine höhere Effizienz, andererseits lässt sich das ständige Schreiben des gleichen Codes vermeiden.

Weitere Vorteile von dem JavaScript-Framework Vue, ist die besondere Benutzerfreundlichkeit, dabei spielt die Erfahrung der Nutzer kaum eine Rolle. Vue lässt sich problemlos in bereits bestehende Projekte integrieren und bietet zahlreiche Erweiterungsmöglichkeiten. Beispiele dafür wären „Vuex“ für das State-Management oder auch „Vuetify“ für vorgefertigte Design-Elemente.

Vuetify ist eine Erweiterung für das Framework Vue.js, diese stellt viele bereits vorgefertigte Design-Elemente bereit. Darunter sind Navigationsleisten, Karten, Formulare aber auch einfachere Elemente wie Buttons oder Textfelder mit weiteren Features. Die Erweiterung Vuetify hilft Entwicklern effizient zu arbeiten und schnell moderne und benutzerfreundliche Oberflächen zu erstellen, dabei Design-Elemente selbst gestalten zu müssen. Außerdem hilft Vuetify dabei, responsive und optisch ansprechende Webanwendungen zu entwickeln, was für ein einheitliches Erscheinungsbild sorgt.

3.2 C#

C# ist eine moderne, objektorientierte Programmiersprache von Microsoft, die sich besonders für die Entwicklung von Anwendungen auf der .NET-Plattform eignet. Sie wird häufig für Desktop-Programme, Webanwendungen, Spiele (mit Unity) und sogar mobile Apps verwendet.

Durch ihre klare Syntax und die enge Integration mit .NET ermöglicht C# eine effiziente und strukturierte Programmierung. Die Sprache unterstützt viele moderne Konzepte wie asynchrone Verarbeitung, objektorientierte Programmierung und starke Typisierung, was zu stabiler und gut wartbarer Software führt.

Dank der großen Community und zahlreicher Bibliotheken kann C# flexibel in verschiedenen Bereichen eingesetzt werden – von Unternehmenssoftware bis hin zu modernen Web-APIs mit .NET.

3.3 Swagger

Swagger ist ein leistungsstarkes Tool zur Dokumentation und Entwicklung von APIs. Es ermöglicht Entwicklern, REST-APIs übersichtlich zu beschreiben, zu testen und zu validieren.

Mit Swagger können Schnittstellen in einem standardisierten Format definiert werden, was die Zusammenarbeit zwischen Backend- und Frontend-Teams erleichtert. Zudem bietet es eine interaktive Oberfläche, mit der API-Endpunkte direkt im Browser getestet werden können. Besonders praktisch ist, dass CRUD-Operationen (Create, Read, Update, Delete) mit Swagger einfach umgesetzt werden können, was die Entwicklung und Verwaltung von APIs erheblich vereinfacht.

3.4 SQL-Datenbanken

SQL-Datenbanken sind relationale Datenbanksysteme, die Structured Query Language (SQL) verwenden, um Daten zu verwalten und abzufragen. Sie ermöglichen die Speicherung von Informationen in Tabellen, die durch Beziehungen miteinander verbunden sind.

Ein großer Vorteil von SQL-Datenbanken ist die Fähigkeit, komplexe Abfragen durchzuführen und große Datenmengen effizient zu verarbeiten. Mit SQL können Entwickler Daten einfügen, abrufen, aktualisieren und löschen, wodurch sie eine flexible Verwaltung der Daten ermöglichen.

Bekannte SQL-Datenbanken sind **MySQL**, **PostgreSQL**, **Microsoft SQL Server** und **SQLite**. Diese Systeme bieten robuste Funktionen zur Sicherstellung der Datenintegrität, Transaktionsverarbeitung und Sicherheitsmechanismen. SQL-Datenbanken sind ideal für Anwendungen, die konsistente Daten und komplexe Abfragen erfordern, wie z. B. Unternehmenssoftware, Webanwendungen und E-Commerce-Plattformen.

In unserem Fall wurde SQL als Datenbank für die Nutzerkonten sowie die verschickten Nachrichten verwendet, um eine effiziente Speicherung und Verwaltung dieser wichtigen Daten zu gewährleisten.

3.5 DevExpress XPO

DevExpress XPO (eXpress Persistent Objects) ist ein leistungsstarkes ORM (Object-Relational Mapping) Tool, das die Interaktion zwischen C#-Anwendungen und relationalen Datenbanken erleichtert. Es ermöglicht Entwicklern, Datenbankoperationen mithilfe von Objektmodellen durchzuführen, ohne sich um die zugrunde liegenden SQL-Abfragen kümmern zu müssen.

In unserem Projekt haben wir DevExpress XPO verwendet, um automatisch eine eindeutige Objekt-ID (OID) für jeden Nutzer zu generieren. Diese OID erleichtert es uns, Nutzer in der Datenbank zu identifizieren und effizient zu bearbeiten, zu löschen und zu aktualisieren. Durch die Nutzung von XPO können wir die Datenmanipulation einfach und schnell durchführen, was die Entwicklung unserer Anwendung erheblich vereinfacht.

3.6 Firebase

Firebase ist eine Entwicklungsplattform von Google, die eine Vielzahl von Tools und Diensten für die Erstellung von Web- und Mobilanwendungen bereitstellt. Sie bietet Funktionen wie Echtzeit-Datenbanken, Authentifizierung, Hosting und Cloud-Funktionen, die Entwicklern helfen, ihre Anwendungen schnell und effizient zu entwickeln.

Ein herausragendes Merkmal von Firebase ist die Echtzeit-Datenbank, die es ermöglicht, Daten sofort zu synchronisieren, sodass Nutzer in Echtzeit auf Änderungen zugreifen können. Diese Funktion ist besonders nützlich für Anwendungen, die eine hohe Interaktivität erfordern, wie Chat-Apps oder kollaborative Plattformen.

Firebase bietet auch eine einfache Möglichkeit zur Benutzerregistrierung und -authentifizierung über verschiedene Anmeldemethoden wie E-Mail, Google und Facebook. Dadurch wird die Implementierung von Sicherheits- und Benutzerverwaltungsfunktionen deutlich vereinfacht.

Im Laufe des Projektes haben wir entschieden, von der SQL-Datenbank und dem C#-Backend zu Firebase zu wechseln. Diese Entscheidung fiel, weil Firebase in Bezug auf die Implementierung eines Echtzeit-Chats einfacher und kompatibler ist. Durch die Nutzung von Firebase benötigen wir keine zusätzliche API, um die Daten zu verwalten und die Echtzeit-Funktionalität bereitzustellen. Dieser Wechsel erleichtert nicht nur die Entwicklung, sondern sorgt auch für eine nahtlose Benutzererfahrung, da die Daten in Echtzeit synchronisiert werden. Im weiteren Verlauf der Dokumentation werden wir genauer auf die Herausforderungen eingehen, die uns zu diesem Wechsel veranlasst haben, sowie auf die Vorteile, die sich aus der Integration von Firebase in unser Projekt ergeben haben.

4. Theoretisches Konzept

Das theoretische Konzept unseres Projekts bildet die Grundlage für die Entwicklung einer modernen Webanwendung. Das Frontend wird in Visual Studio Code erstellt und nutzt das Framework Vue in Kombination mit der Erweiterung Vuetify, um ansprechende und benutzerfreundliche Oberflächen zu gestalten. Im Backend implementieren wir eine .NET 8-Anwendung in Visual Studio, die mit Swagger für die Dokumentation und Testbarkeit der API ausgestattet ist. Dieses Backend stellt die zentrale Verbindung zwischen der SQL-Datenbank und dem Frontend her und wird in C# programmiert. Es umfasst die CRUD-Operationen, die erforderlich sind, um Benutzerkonten zu erstellen, zu bearbeiten, zu aktualisieren und zu löschen.

Sobald diese Struktur etabliert ist, folgt der nächste Schritt: Ein Nutzer gelangt auf die Startseite und hat die Möglichkeit, sich entweder anzumelden oder zu registrieren, je nachdem, ob er bereits über ein Konto verfügt. Die Registrierung erfolgt über eine POST-Methode, die es ermöglicht, einen neuen Account zu erstellen und in der Datenbank zu speichern. Bei erfolgreicher Registrierung wird der Nutzer auf die Startseite weitergeleitet, wo er die Option hat, zu Chats zu navigieren, neue Gespräche zu starten oder bestehende Unterhaltungen fortzuführen.

Zusätzlich soll den Nutzern die Möglichkeit geboten werden, ihr Profilbild zu ändern und persönliche Daten wie Benutzername, Geburtsdatum, Herkunft und Beruf anzupassen. Wenn Nachrichten verfasst werden, erfolgt die Übermittlung in Echtzeit an den anderen Gesprächspartner über die Chat-API. Diese Nachrichten werden zudem in einer separaten Tabelle innerhalb der Datenbank gespeichert, um eine effiziente Verwaltung und Abrufbarkeit der Unterhaltungen zu gewährleisten.

4.1 Designvorschläge

Der Designvorschlag für unsere Webanwendung zielt darauf ab, eine benutzerfreundliche und ansprechende Benutzeroberfläche zu schaffen, die die Interaktion der Nutzer fördert. Durch die Verwendung von klaren, modernen Layouts und einer ansprechenden Farbpalette wird ein einheitliches und attraktives Erscheinungsbild erzielt.

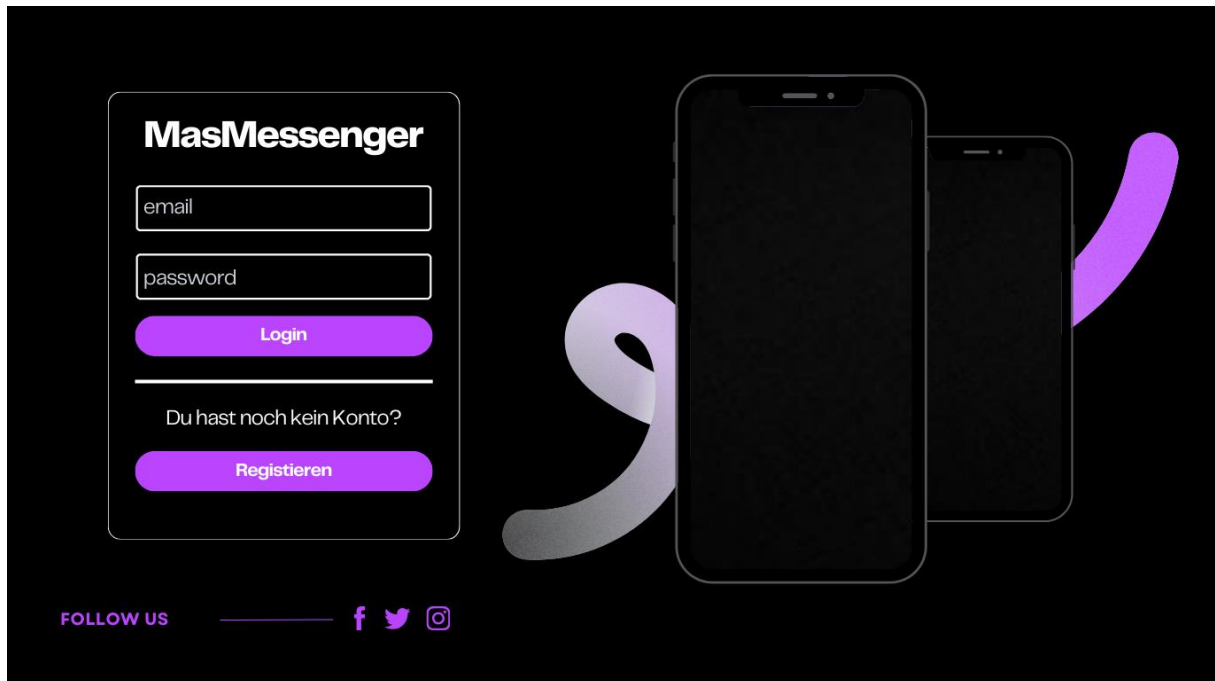


Abbildung 5: Designvorschlag Startseite

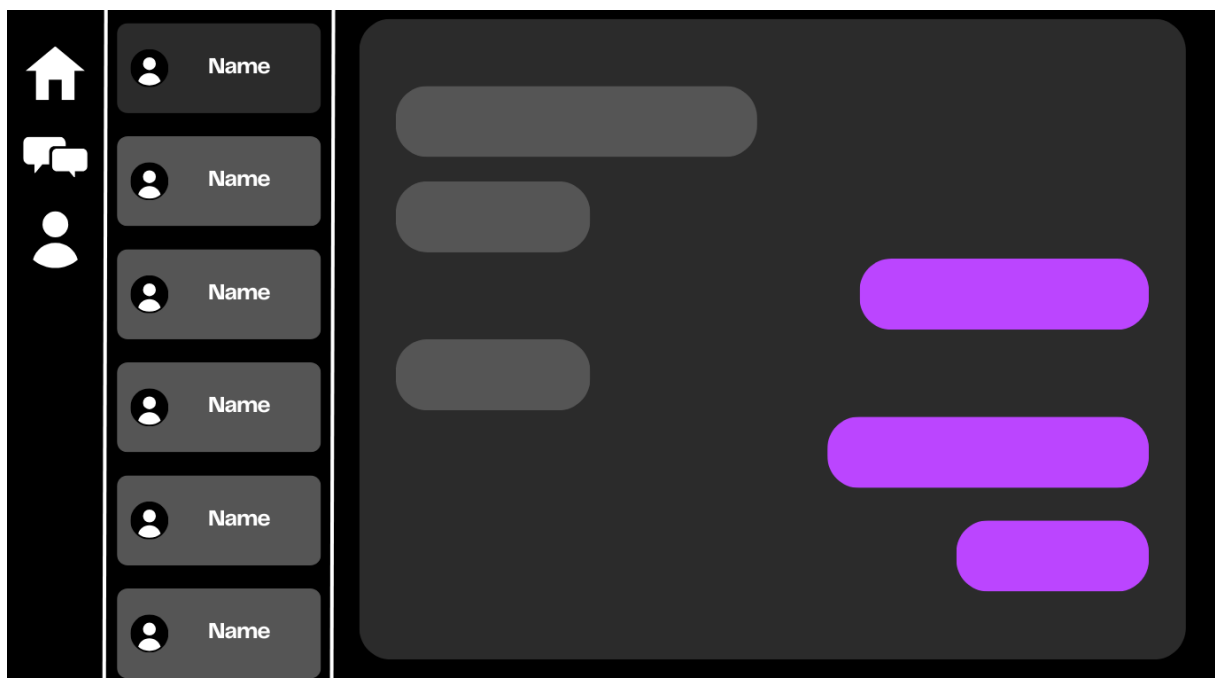


Abbildung 6: Designvorschlag Chats

5. Praktische Umsetzung

5.1 GitHub-Repository erstellen

Ein GitHub-Repository zu erstellen, bietet zahlreiche Vorteile: Es ermöglicht eine einfache Versionierung des Codes, fördert die Zusammenarbeit im Team und bietet eine zentrale Plattform für die Speicherung und Bereitstellung des Projekts. Zudem können durch Issues und Pull-Requests Probleme effizient verwaltet und neue Funktionen nahtlos integriert werden. Insgesamt optimiert ein GitHub-Repository die Entwicklung und Dokumentation von Softwareprojekten. Dabei wird auf eine ordentliche und überschaubare Ordnerstruktur geachtet, bei der sich jedes Projektteammitglied auskennt.

Name	Last commit message	Last commit date
..		
MasMessenger24	Update Startseite: Userlist erledigt	now
MasMessenger_Backend	Backend-C#	9 minutes ago
Sonstiges	Update der Doku	2 minutes ago

Abbildung 7: Ordnerstruktur - GitHub

5.2 Einrichten der Datenbank

Um die Datenbank für unser Projekt einzurichten, haben wir zunächst im C#-Backend eine Klasse erstellt, die die Struktur der Daten repräsentiert. Diese Klasse definiert die Eigenschaften, die für die Benutzerkonten und Nachrichten erforderlich sind. Basierend auf dieser Klasse haben wir anschließend eine entsprechende Tabelle in der SQL-Datenbank angelegt.

Ergebnisse Meldungen												
	OID	Name	Surname	Email	PasswordHash	PasswordSalt	Image	Birthday	Job	Nationality	Gender	Role
1	1	Nemo	Katanic	nemo@katanic	0x09373FB68...	0x2CBE7B3...	0xF...	2024-1...	Student	Serbian	Male	User
2	3	Matej	Grkovic	matej@gkrovic.com	0x84927CA4...	0xF8CDCD...	0xF...	2024-1...	AMS	Austria	Male	User
3	4	Max	Gmeinder	maxgmeinder@gmail.com	0xE58BD3A5...	0x78511A7...	0xF...	2024-1...	AMS	Austria	Male	User
4	9	Zebra	Cebrail	zebra@gmail.com	0xECAF767E...	0xA755516...	NULL	NULL	NULL	NULL	NULL	NULL
5	10	Sammy	Semmel	sammysemmel@gmail.com	0x0A0422D9...	0xBF091D8...	NULL	NULL	NULL	NULL	NULL	NULL
6	11	Julian	Barfus	juki@gmail.com	0xA2175D77...	0xB60864C...	NULL	NULL	NULL	NULL	NULL	NULL

Abbildung 8: Tabelle - SQL Datenbank

Hier ist die User Tabelle zusehen, die anhand der User.cs (Klasse) im Backend erstellt wurde.

5.3 CRUD-Operation mit Swagger

Nach dem Einrichten der Datenbank war der nächste Schritt, die CRUD-Operationen mithilfe von Swagger zu implementieren. Swagger ermöglicht es uns, unsere API interaktiv zu dokumentieren und zu testen, sodass wir auf einfache Weise Endpunkte wie Erstellen, Abrufen, Aktualisieren und Löschen von Datensätzen überprüfen können.

Im folgenden Screenshot sehen Sie die Swagger-Oberfläche, die alle verfügbaren API-Endpunkte auflistet. Hier können Sie die erforderlichen Parameter einsehen, Beispielanfragen starten und direkt die Funktionalität unserer CRUD-Operationen testen. Diese interaktive Dokumentation ist ein wertvolles Werkzeug, um die reibungslose Integration zwischen dem Backend und der Datenbank zu gewährleisten.

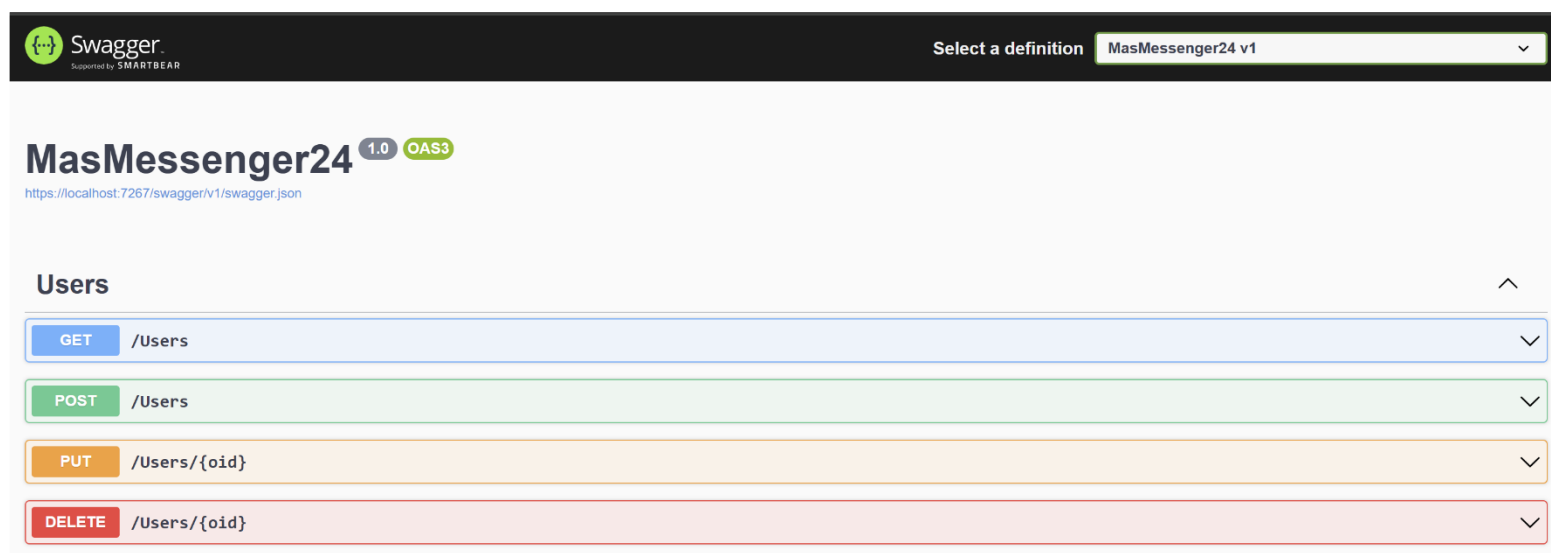


Abbildung 9: Swagger - CRUD Operation

5.4 Programmieren der Weboberfläche

Die Programmierung der Weboberfläche erfolgte basierend auf unserem detaillierten Designvorschlag. Dabei wurde Visual Studio Code als Entwicklungsumgebung genutzt und das Frontend mithilfe von Vue.js und Vuetify umgesetzt. Der Designvorschlag diente als Leitfaden für die Struktur und das Layout der Benutzeroberfläche: klare, intuitive Navigation und ansprechende, moderne Elemente standen im Fokus. Vue.js ermöglichte die Entwicklung dynamischer und reaktiver Komponenten, während Vuetify uns unterstützte, die Prinzipien des Material Designs konsequent anzuwenden. So entstand eine benutzerfreundliche und reaktionsschnelle Weboberfläche, die sowohl funktional als auch ästhetisch überzeugt.

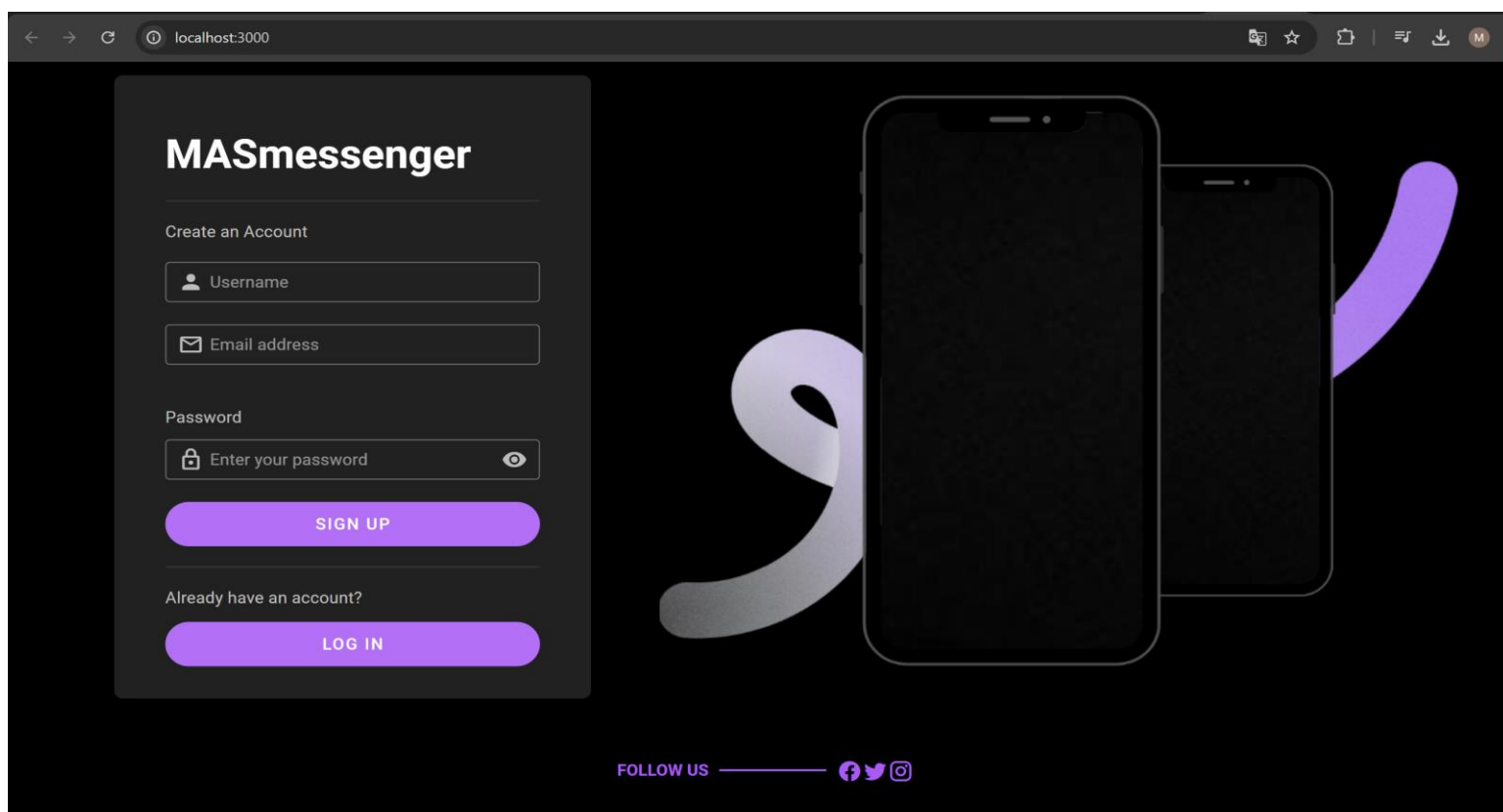


Abbildung 10: Weboberfläche - Startseite

5.5 Wechsel von SQL-Datenbank zu Firebase

Im Verlauf des Projekts stellten wir fest, dass die herkömmliche SQL-Datenbank in Kombination mit dem C#-Backend einige Herausforderungen bei der Umsetzung eines Echtzeit-Chats mit sich bringt. Besonders bei der schnellen Synchronisierung von Chatnachrichten und der Verwaltung von Live-Daten zeigte sich, dass SQL-basierte Lösungen nicht immer optimal geeignet sind.

Deshalb haben wir uns dazu entschieden, auf Firebase umzusteigen. Firebase bietet eine integrierte Echtzeit-Datenbank, die eine sofortige Synchronisation zwischen den Nutzern ermöglicht, ohne dass zusätzliche APIs implementiert werden müssen. Dadurch wurde der Entwicklungsaufwand reduziert und die Benutzererfahrung erheblich verbessert, da die Kommunikation nahezu verzögerungsfrei erfolgt.

In der Dokumentation bei dem Punkt „**6. Aufgetretene Probleme**“ gehen wir noch detailliert auf die spezifischen Probleme und den Wechsel zu Firebase ein.

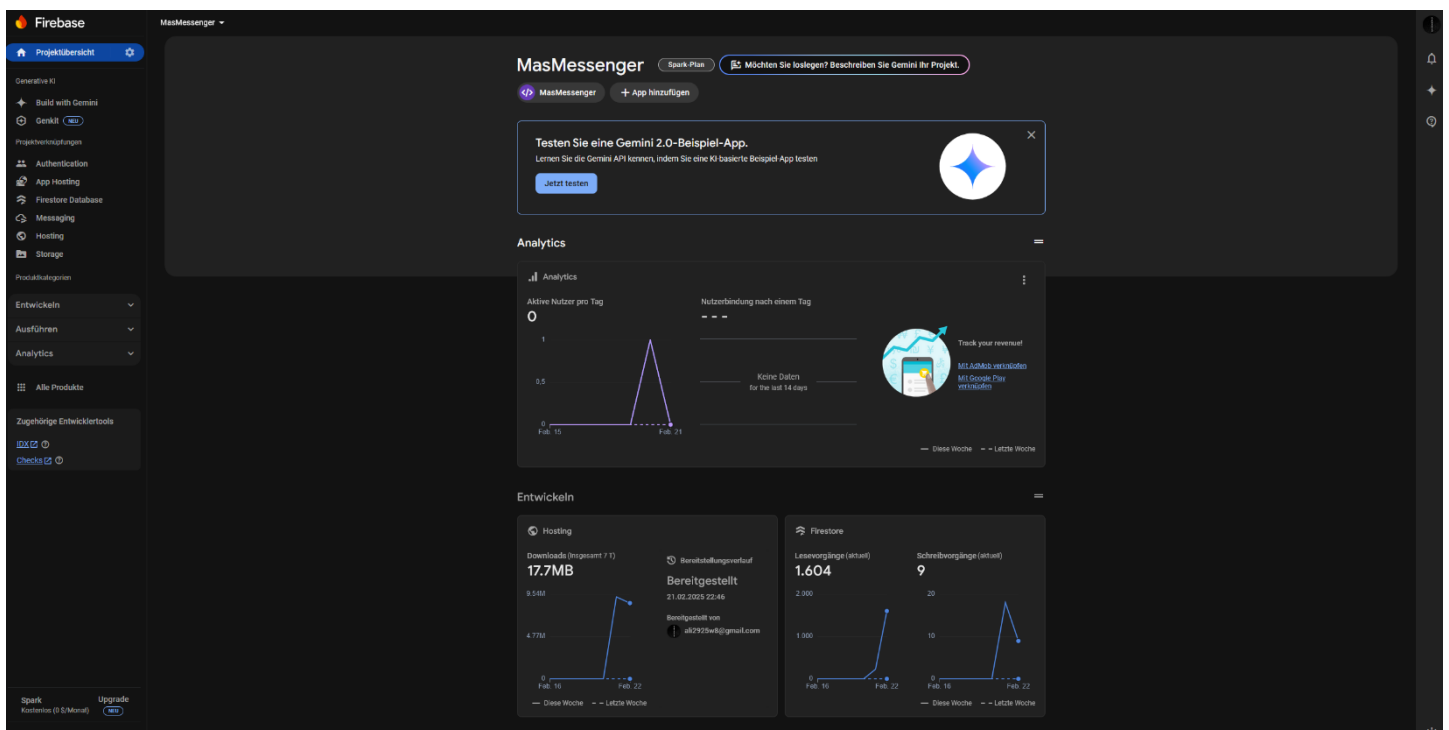


Abbildung 11: Firebase - Weboberfläche

5.6 Fertigstellen des Projektes

Nachdem wir zu Firebase gewechselt haben, lag unser Hauptaugenmerk auf der finalen Optimierung und Fertigstellung des Projekts. Die Echtzeit-Datenverwaltung wurde stabil integriert, sodass die Kommunikation reibungslos und nahezu verzögerungsfrei abläuft. Dank Firebase konnten wir die Anwendung direkt deployen – Docker war somit nicht mehr erforderlich.

In dieser letzten Phase wurden umfassende Tests durchgeführt, letzte Fehler behoben und die Benutzeroberfläche weiter verfeinert, um eine intuitive und zuverlässige Nutzung sicherzustellen. Alle Kernfunktionen, wie Registrierung, Anmeldung, Chat und Profilverwaltung, wurden gründlich validiert. Mit der erfolgreichen Fertigstellung haben wir einen soliden Grundstein für eine moderne und skalierbare Kommunikationsplattform geschaffen.

6. Aufgetretene Probleme

Im Großen und Ganzen lief das Projekt relativ gut, jedoch traten einige Herausforderungen auf, die uns zum Wechsel von der SQL-Datenbank zu Firebase veranlassten. Ursprünglich war die SQL-Datenbank korrekt implementiert und sowohl die Anmeldung als auch die Registrierung funktionierten einwandfrei. Das Problem trat jedoch nach der Anmeldung auf: Nachdem sich ein Nutzer erfolgreich eingeloggt hatte, wurde er auf eine neue Seite weitergeleitet, die nicht wusste, welcher Nutzer aktuell angemeldet war. Dadurch konnten die zugehörigen Chats nicht korrekt abgerufen werden.

Um dieses Problem zu lösen, dachten wir zunächst daran, einen JWT-Token zu verwenden, der der API übermittelt werden sollte, um das richtige Nutzerprofil zu identifizieren. Dieser Ansatz erwies sich jedoch als zeitintensiv und komplex. Zudem war die Suche nach einer passenden, kostenfreien Chat-API schwierig, da viele Optionen kostenpflichtig waren.

Diese Probleme führten zu der Entscheidung, auf Firebase umzusteigen. Firebase bietet eine integrierte Echtzeit-Datenbank und eine effizientere Möglichkeit, Nutzer zu authentifizieren und Chats in Echtzeit zu synchronisieren – ganz ohne zusätzliche, komplexe API-Implementierungen. Somit konnten wir die Nutzeridentifikation und Datenverwaltung wesentlich vereinfachen und die Echtzeitkommunikation stabil und performant umsetzen.

7. Verwendete Hilfen

Während der Entwicklung unserer Webanwendung haben wir **ChatGPT** als Unterstützung beim Programmieren sowie zur Korrektur von Rechtschreibfehlern genutzt. ChatGPT half insbesondere bei der Verbesserung der Codequalität, der Optimierung von Algorithmen und der Formulierung präziser API-Anfragen. Zudem diente es als hilfreiches Tool, um Fehler in der Dokumentation und den Benutzeroberflächen frühzeitig zu erkennen und zu korrigieren.

Darüber hinaus haben wir **DeepSeek** ausschließlich als Programmierhilfe eingesetzt. Es unterstützte uns gezielt bei der Lösung komplexer Entwicklungsprobleme, der Verbesserung von Code-Strukturen und der Optimierung von Performance-Aspekten.

8. Quellenverzeichnis

Überblick über Vue

<https://vuejs.org/>

Was man über Vue.js wissen sollte

<https://www.heise.de/blog/Was-man-ueber-Vue-js-wissen-sollte-4969211.html>

Vue 3 Tutorial für Einsteiger

<https://vuejs.de/artikel/vuejs-tutorial-deutsch-anfaenger/>

Vuetify

<https://vuetifyjs.com/en/introduction/why-vuetify/#what-is-vuetify3f>

Was ist Vuetify

<https://mfg.fhstp.ac.at/allgemein/was-ist-vuetify/>

Was ist C#?

<https://www.pi-informatik.berlin/pi-lexikon/softwareentwicklung/was-ist-csharp/>

Überblick über C#

<https://learn.microsoft.com/de-de/dotnet/csharp/tour-of-csharp/overview>

Swagger

[https://de.wikipedia.org/wiki/Swagger_\(Software\)](https://de.wikipedia.org/wiki/Swagger_(Software))

Swagger: Mehr Komfort bei der API-Entwicklung

<https://www.ionos.at/digitalguide/websites/web-entwicklung/was-ist-swagger/>

Was ist SQL und eine SQL-Datenbank

<https://www.assecor.de/glossar/sql-datenbank>

DevExpress XPO

<https://docs.devexpress.com/xpo/2263/getting-started>

Firebase – die Power aus dem Hintergrund

<https://entwickler.de/mobile/firebase-die-power-aus-dem-hintergrund-002>

Firebase

<https://firebase.google.com/docs/database?hl=de>

9. Abbildungsverzeichnis

Abbildung 1: Projektumweltanalyse	9
Abbildung 2: Objektstrukturplan	11
Abbildung 3: Projektstrukturplan	12
Abbildung 4: Projektorganigramm	13
Abbildung 5: Designvorschlag Startseite	21
Abbildung 6: Designvorschlag Chats	21
Abbildung 7: Ordnerstruktur - GitHub.....	22
Abbildung 8: Tabelle - SQL Datenbank	23
Abbildung 9: Swagger - CRUD Operation	24
Abbildung 10: Weboberfläche - Startseite	25
Abbildung 11: Firebase - Weboberfläche	26

10. Tabellenverzeichnis

Tabelle 1: Projektauftrag.....	7
Tabelle 2: Projektzieleplan.....	8
Tabelle 3: Projektumweltanalyse-Beziehungen	10
Tabelle 4: Projektmeilensteinplan	14