



TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Fakultät Elektrotechnik Feinwerktechnik Informationstechnik

Projekt: Digital Dahoam

Gruppenprojektbericht im Studiengang Software Engineering

vorgelegt von

Daniel Reitberger, Andre Reif, Rebecca Vogler, Jan Scholz, Boas
Dünkel, Marc Jonas Roser

Betreuer: Johannes Walser

Vorgelegt am 01.04.2023

© 2023

Dieses Werk einschließlich seiner Teile ist **urheberrechtlich geschützt**. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung des Autors unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen sowie die Einspeicherung und Verarbeitung in elektronischen Systemen.

Glossar

API Application Programming Interface.

BaaS Backend as a Service: Plattform, die die Bereitstellung von Backend-Funktionalität ermöglicht inklusive Datenbank, Authentifizierung, Datei-Upload, etc.

Bug Fehler in der Software.

Bugfix Behebung eines Fehlers in der Software.

CD Continuous Delivery / Continuous Deployment: Automatisches Ausrollen der neuen Funktionalität.

CI Continuous Integration: Frühes Integrieren kleiner Änderungen in den Hauptzweig (Git Branches).

Confluence Software zur Zusammenarbeit und Dokumentation.

Jira Software zum Projektmanagement.

jsx Javascript XML: Erweiterung von Javascript, die es ermöglicht, HTML-Elemente in Javascript zu definieren.

MUI Material-UI (Material Design Komponenten für React).

Node.js Javascript-Plattform, die es ermöglicht, Javascript außerhalb des Browsers auszuführen.

React Frontend-Framework basierend Node.js, das von Facebook entwickelt wird. Es ermöglicht die Entwicklung von Benutzeroberflächen für Webanwendungen.

tsx TypeScript XML: Erweiterung von TypeScript, die es ermöglicht, HTML-Elemente in TypeScript zu definieren.

UI Benutzeroberfläche, der Teil der Anwendung, der für den Nutzer sichtbar und nutzbar ist. Englisch: „User Interface“.

Usability Bedienbarkeit oder Nutzbarkeit einer Anwendung.

USP Unique Selling Point.

Wireframe Grobkonzept einer Benutzeroberfläche.

Inhaltsverzeichnis

1. Einleitung	1
2. Projekt	2
2.1. Projektbeschreibung	2
2.1.1. Ausgangssituation	2
2.1.2. Projektziel	3
2.1.3. Aufgaben	3
2.2. Team	4
3. Prozess	5
3.1. Scrum	5
3.2. Projektmanagement	5
3.3. Sprintübersicht	5
4. Analyse	6
4.1. Wettbewerbsanalyse	6
4.2. Anforderungsanalyse	6
4.3. Usabilityanalyse	6
5. Architektur	7
6. Technologien	8
6.1. Grundlage für Technologieentscheidung	8
6.2. TypeScript	9
6.2.1. Einsatz im Projekt	9
6.2.2. Grund für Technologieentscheidung	9
6.3. React	9
6.3.1. Allgemeines in Bezug auf die Implementierung mit React	9
6.3.2. Einsatz im Projekt	11
6.3.3. Grund für Technologieentscheidung	11
6.4. Material-UI	12
6.4.1. Einsatz im Projekt	12
6.4.2. Grund für Technologieentscheidung	12
6.5. Firebase	13
6.5.1. Firebase Authentication	13
6.5.2. Firebase Realtime Database	13
6.5.3. Firebase Cloud Firestore	13
6.5.4. Firebase Storage	14
6.5.5. Einsatz im Projekt	14

6.5.6. Grund für Technologieentscheidung	14
6.6. Github	14
6.6.1. Einsatz im Projekt	14
6.6.2. Grund für Technologieentscheidung	15
6.7. Vercel	15
6.7.1. Einsatz im Projekt	15
6.7.2. Grund für Technologieentscheidung	15
6.8. Jira	16
6.8.1. Einsatz im Projekt	16
6.8.2. Grund für Technologieentscheidung	16
7. Implementierung	17
7.1. Landing Page	17
7.2. Profil	18
7.2.1. Anmeldung & Registrierung	18
7.3. Authentifizierung	19
7.3.1. Verwendung des „Firebase-Authentication“-Services	19
7.3.2. Zugriff auf die Profilinformationen	20
7.3.3. Authentifizierungsprozess	21
7.3.4. Persönliches Accountmanagement	23
7.3.5. Profilseite & Profilbilder	23
7.3.6. Blockierte Nutzer	24
7.4. Beiträge	24
7.4.1. Beiträge erstellen	25
7.4.2. Alle Beiträge	26
7.4.3. Profilseite	27
7.4.4. Merkzettel	27
7.4.5. Marktplatz	28
7.4.6. Dashboard	28
7.5. Chat	29
7.5.1. Firebase Realtime Database für den Chat	29
7.5.2. Autorisierung der Nachrichten und Chaträume	29
7.5.3. Benutzung des Chats	30
7.5.4. Verknüpfung von Beiträgen mit dem Chatraum des Beitragautors	31
8. Testing	33
9. Fazit	34
A. Anhang	A
I. Bilder	A

<i>Inhaltsverzeichnis</i>	V
II. Code	C
Abbildungsverzeichnis	E
Tabellenverzeichnis	F
Literaturverzeichnis	G

1. Einleitung

Dieser Projektbericht beschreibt die Entwicklung einer prototypischen Plattform für einen interaktiven Austausch innerhalb eines Ortes, einer Stadt oder einer Gemeinde. Er wurde im Rahmen des Masterstudiengangs *Software-Engineering* an der Technischen Hochschule Georg Simon Ohm Nürnberg erstellt.

Zu Beginn wird auf die Beschreibung des Projekts eingegangen, die Anforderungen an das Produkt werden definiert und die Aufgaben des Projekts werden aufgezeigt. Damit wird eine Grundlage gelegt, die für die weitere Arbeit im Projekt benötigt wird. Außerdem wird das Team in seinen verschiedenen Rollen vorgestellt.

Im nächsten Kapitel wird der Prozess der Entwicklung der App beschrieben. Dies geschah über eineinhalb Jahre hinweg und umfasst die Planung, die Konzeption und die Implementierung der Plattform. Mit der agilen Softwareentwicklungsmethode Scrum wurde ein iterativer Prozess durchgeführt, der die Entwicklung der Anwendung in mehreren Sprints abwickelte.

Im vierten Kapitel werden technische Aspekte von „Digital Dahoam“ beschrieben. Hierbei werden Technologien und Frameworks vorgestellt, die für die Entwicklung der Plattform verwendet wurden. Diese werden jeweils mit einem kurzen Überblick vorgestellt, der genaue Zweck bzw. Einsatz im Projekt erklärt und die Entscheidung für die Verwendung dieser Technologien wird erläutert.

Die Architektur der Plattform wird im fünften Kapitel genauer beschrieben und die einzelnen Komponenten vorgestellt. Außerdem wird die Kommunikation zwischen den Komponenten beschrieben.

Details zur Umsetzung der Website erscheinen in Kapitel 7. Hierbei wird die Umsetzung der einzelnen Funktionen der Plattform beschrieben. Zunächst wird die Umsetzung der Benutzeroberfläche beschrieben, danach werden die einzelnen Funktionen der Plattform vorgestellt. Außerdem wird durch Screenshots die Benutzeroberfläche der Plattform dargestellt.

Anschließend werden die Aktivitäten in den Sprints beschrieben. Dabei wird kurz über die einzelnen User Stories berichtet, die in den Sprints umgesetzt wurden. Außerdem wird die Arbeit im Team beschrieben und die Ergebnisse der einzelnen Sprints zusammengefasst.

Der Bericht wird abgeschlossen mit einem Fazit, in dem die Ergebnisse des Projekts zusammengefasst werden. Außerdem wird ein Ausblick auf mögliche Weiterentwicklungen der Plattform gegeben und eine Bewertung der Arbeit im Projekt abgegeben.

2. Projekt

Zu aller erst wird das Projekt vorgestellt und die Aufgabenstellung erläutert. Dabei wird auch auf die Ausgangssituation eingegangen und das Projektziel definiert. Dadurch wird der Leser in die Lage versetzt, die weitere Projektdokumentation besser zu verstehen. Anschließend wird das Team vorgestellt und die einzelnen Rollen und Aufgaben der Teammitglieder erläutert.

Die nachfolgende Projektbeschreibung stammt direkt aus dem Projektaushang und wurde nur leicht angepasst. Die Projektbeschreibung ist in der Projektarbeit nicht weiter ausführlich dargestellt, da sie nur eine kurze Einführung in das Projekt darstellt.

2.1. Projektbeschreibung

Das Projekt „Digital Home Town“ ist ein „smart city-Konzept“, welches zur digitalen Vernetzung verschiedener Generationen und Interessenten im Sozialraum dient. Lebendiger Austausch, voneinander lernen, kommunale Ressourcen effizient nutzen, als Stadt virtuell zusammenwachsen, darum geht es in diesem Projekt.

2.1.1. Ausgangssituation

Grundsätzlich geht es darum eine Plattform zu bauen, die Zielgruppen- und generationenübergreifend eine Stadt vernetzt und verschiedene Features anbietet. So können virtuelle Kursräume Informationen zur Verfügung stellen, durch Chattools Diskussionsforen eingerichtet werden und die physikalische Infrastruktur treffend verteilt werden, z. B. durch ein Buchungstool (Turnhallenbelegung, Sportheim). So können z. B. schulische Inhalte auch anderen Generationen zur Verfügung gestellt werden und umgekehrt zeithistorische Informationen für Schüler dargeboten werden sowie kommunale Ressourcen optimal genutzt und ausgelastet werden. Ferner entsteht ein digitaler Marktplatz für lokale Betriebe, um die Wirtschaftskraft in der Region zu stärken. Diese Beispiele stellen nur exemplarische Ansätze und Features dar. Denn die Plattform wächst in einer organischen Evolution mit den Anforderungen der nutzenden Gesellschaft.

Zwei besondere Anforderungen bringt dieses Projekt mit sich:

- Das fertige Produkt muss von Beginn an generationenübergreifend eine hohe Userakzeptanz erlangen, indem die Zugänglichkeit und Bedienbarkeit leicht, schnell und barrierefrei gewährleistet ist.
- Um ein solches Projekt langfristig zu realisieren, müssen Schnittstellen modular mitgedacht werden, damit auch sich verändernde Bedarfsstrukturen berücksichtigt werden können.

2.1.2. Projektziel

Ziel ist es in diesem Projekt eine prototypische Plattform zu entwickeln, die in ihrer Multifunktionalität verschiedenste Anforderungen für die Nutzer erfüllt. Ob es nun . . . ,

. . . ein Planungstool, ähnlich einem mit anderen Usern geteilten Kalender,

. . . ein Chat oder Blog-Tool zur virtuellen Interaktion und diskursiven Auseinandersetzung ist,

. . . virtuelle Kursräume und Datenarchive sind, die generationenübergreifendes Lernen ermöglichen,

. . . eine Tauschbörse oder einfach nur

. . . eine originäre Homepage ist,

mit allen Features wird das Zusammenleben in einer Kommune leichter und dem aktuellen Stand der Technik gerecht. Deshalb sollen diese sämtlich in diese eine Plattform einbezogen werden.

2.1.3. Aufgaben

Die zu bearbeitenden Aufgaben umfassen folgende Punkte:

- Einbindung des Kunden und Abstimmung zu spezifischen Anforderungen
- Anforderungsanalyse der zu priorisierenden Features
- Konzeption einer multimedialen Plattform
- Entwicklung eines Rechte- und Rollenkonzepts
- Entwicklung einzelner Features
- Umsetzung der Spezifikationen im Sinne der Entwicklung eines lauffähigen Prototyps.
- Konsequente Berücksichtigung der Usability für alle Adressaten
- Test der lauffähigen Funktionalitäten.

2.2. Team

Im Rahmen des Projekts „Digital Home Town“ arbeiteten folgende Personen zusammen:

- **Daniel Reitberger** (Scrum Master)
- **Andre Reif** (Product Owner, Entwickler)
- **Rebecca Vogler** (Product Owner, Tester)
- **Boas Dünkel** (Entwickler)
- **Jan Scholz** (Entwickler)
- **Jonas Roser** (Entwickler)

Dieses Team hat sich selbstständig für das Thema „Smart City“ entschieden und wird von Herrn Johannes Walser betreut. Dieser ist der Ansprechpartner für alle Fragen und Anliegen, die sich im Projekt ergeben. Dazu war er der „Kunde“ des Projekts, also derjenige, der die Anforderungen an das Produkt definiert hat.

Alle Teammitglieder haben sich in der Projektarbeit gegenseitig unterstützt und sich gegenseitig weiterentwickelt.

3. Prozess

3.1. Scrum

3.2. Projektmanagement

3.3. Sprintübersicht

4. Analyse

4.1. Wettbewerbsanalyse

4.2. Anforderungsanalyse

4.3. Usabilityanalyse

5. Architektur

Auf Grundlage der Technologie Entscheidungen wurde folgende Architektur für das Projekt entwickelt. Diese ist in folgender Abbildung dargestellt und wird in diesem Kapitel noch genauer erklärt:

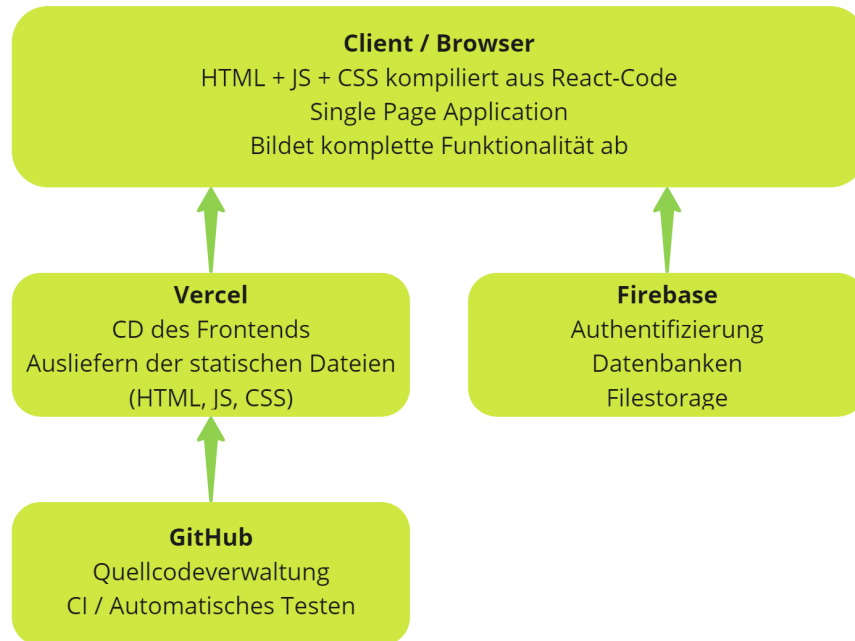


Abbildung (5.1) – Übersicht der technischen Softwarearchitektur

Das Frontend wird in React entwickelt. Dieses enthält die Single Page Application (SPA) für die Benutzeroberfläche. Die gesamte Funktionalität wird in diesem Frontend implementiert. Um im Team bei der Entwicklung effizient zusammenzuarbeiten, wird als Quellcodeverwaltungstool Git bzw. Github verwendet. Wird eine Änderung am Quellcode veröffentlicht, werden automatisch die CI / CD Prozesse gestartet. Sind die Tests erfolgreich, wird die neue Version über Vercel gehostet.

Wird der entsprechende Internetlink im Browser aufgerufen, erhält der anfragende Client (Browser) die statischen Dateien für die Benutzeroberfläche. Der Browser führt das darin enthaltene Javascript aus, das ein dynamisches Laden der Inhalte der Datenbank (Firebase) ermöglicht.

6. Technologien

Für die Entwicklung eines Softwaresystems ist insbesondere bei der Implementierung entscheidend, welche Technologien verwendet werden. Aus diesem Grund widmet sich dieses Kapitel werden die technischen Aspekte der Plattform beschrieben. Dabei werden die Technologien und Frameworks vorgestellt, die für die Entwicklung der Plattform verwendet wurden.

Die Plattform wurde mit den folgenden Technologien und Frameworks entwickelt:

- **TypeScript** als Programmiersprache
- **React** als Frontend-Framework
- **MUI** als UI-Framework
- **Firebase** als Backend as a Service (BaaS) Plattform
- **Vercel** als Hosting-Plattform
- **Github** als Versionsverwaltungs-Plattform
- **Jira** als Projektmanagement-Plattform

Diese werden im Folgenden kurz vorgestellt.

6.1. Grundlage für Technologieentscheidung

Die einzige aus der Aufgabenstellung ersichtliche Vorgabe ist, dass das Softwareprodukt „Digital Hometown“ eine Plattform für den Austausch bieten soll. Die Wahl der Technologie ist uns hierbei offengelassen. Was daraus jedoch hervorgeht ist, dass es sich um eine für möglichst viele Nutzer verwendbare Web- oder Mobilanwendung handeln soll. Insbesondere bei der Wahl einer Webanwendung, die dem Nutzer jeglichen Installationsaufwand erspart, ist die Hemmschwelle sehr gering, ein Softwareprodukt auszuprobieren.

Da, wie beschrieben, für eine solche Plattform des sozialen Austauschs eine ausreichend große Nutzerzahl entscheidend ist, fiel die Entscheidung schnell auf eine Webanwendung. Obwohl beim aktuellen Stand der Plattform „Digital Dahoam“ nicht in erster Linie auf die Benutzbarkeit auf mobilen Endgeräten gelegt wurde, sei an dieser Stelle erwähnt, dass sich Webanwendungen mit etwas mehr Aufwand sehr gut auch für mobile Geräte wie Smartphones oder Tablets entwickeln lassen. Der Fachbegriff hierfür ist die Umsetzung einer „Progressive Web-App“.

6.2. TypeScript

TypeScript ist eine Erweiterung von Javascript, die statische Typisierung und Klassen hinzufügt. Dadurch wird die Entwicklung von Software vereinfacht, da die Typisierung die Lesbarkeit des Codes verbessert und die Klassen die Wiederverwendbarkeit von Code ermöglichen. Die Programmiersprache wird von Microsoft entwickelt und ist Open Source.¹

6.2.1. Einsatz im Projekt

TypeScript wurde im Projekt verwendet, um die Entwicklung der Plattform zu vereinfachen. Der komplette Code der Website wurde mit TypeScript, HTML und CSS geschrieben, wobei TypeScript hierbei die Hauptrolle spielt.

6.2.2. Grund für Technologieentscheidung

Da TypeScript in großen Teilen der Javawelt inzwischen als de facto Standard ist um vor allem große Anwendungen sicher und effizient zu entwickeln, wurde diese Technologie für die Entwicklung der Plattform verwendet. React (6.3), das größte Frontend-Framework der Welt, wird inzwischen auch in TypeScript entwickelt.²

6.3. React

React ist ein Open-Source Frontend-Framework, das von Facebook entwickelt wird. Es ermöglicht die Entwicklung von Benutzeroberflächen für Webanwendungen. Dabei wird die Benutzeroberfläche in einzelne Komponenten aufgeteilt, die unabhängig voneinander entwickelt werden können. Diese Komponenten werden in einer `.jsx` Datei definiert, die eine Kombination aus Javascript und HTML ist. Die Komponenten werden in einer React Anwendung in einer `.jsx` Datei eingebunden. In einer neueren Version, ist es auch möglich mit TypeScript zu arbeiten. Die neue Dateierweiterung für diese TypeScript ist `.tsx`. Diese Dateien werden dann in eine Javascript Datei kompiliert, die von einem Browser ausgeführt werden kann.³

6.3.1. Allgemeines in Bezug auf die Implementierung mit React

React basiert auf dem Model-View-Controller Design Pattern. Der Browser Document Object Model (DOM) fungiert dabei als die View-Komponente. Die Model-Komponente, ist der Virtual DOM, das vom Controller (React) manipuliert wird.

¹Vgl. TypeScript 2022 [1]

²Vgl. TypeScript 2023 [2]

³Vgl. React 2022 [3]

6.3.1.1. Einrichten und Starten der React-Anwendung

Für die Entwicklung von React-Anwendungen eignen sich alle moderne Entwicklungsumgebungen. Im Projektteam wurde sich auf den weitverbreiteten Texteditor „Visual Studio Code“ geeinigt. Neben einer Entwicklungsumgebung wird Node.js benötigt, um Javascript-Code auf der Entwicklungsmaschine ausführen zu können. Die notwendigen Abhängigkeiten werden mit dem Paketmanager „yarn“ installiert. Mit dem in der Datei package.json definierten Alias yarn dev startet der lokale Node.js Entwicklungsserver automatisch, nachdem alle benötigten Pakete installiert wurden. Handelt es sich um eine lauffähige Version, wird automatisch im Browser die Startansicht der entwickelten React-Anwendung geöffnet.

6.3.1.2. React Components

React besteht aus Komponenten, die automatisch neu gerendert werden, wenn sich die Parameter der Komponente ändern. Komponenten können als Functional- bzw. als Class-Komponenten implementiert werden. Während die Verwendung von Class-Komponenten in älteren React-Versionen üblich war, wird in den aktuellen Versionen meist die funktionelle Implementierung verwendet.

```
1  function Component(props: {name: string}) {  
2    return <div>Hallo {props.name}!</div>  
3  }
```

Beispiel einer React-Komponente

6.3.1.3. React Hooks

Die Komponenten bilden die Basis jeder React-Anwendung. Durch die React-Hooks wird es einer Komponente ermöglicht, dynamische Bestandteile und einen Zustand zu besitzen. Es gibt mehrere Hooks für verschiedene Anwendungsfälle und es lassen sich auch eigene Hooks definieren. Eines der wichtigsten React-Hooks ist useState. Durch useState wird es ermöglicht, eine Variable über ein oder mehrere Komponenten hinweg zu benutzen und manipulieren. Die Verwendung eines useState Hooks wird in Code 2 gezeigt. Hier wird auch ein weiterer wichtiger Hook aufgeführt. Der useEffect Hook ermöglicht es, auf die Änderung eines Zustands zu reagieren.

```
1  function Component() {  
2    // count ist der aktuelle Wert  
3    // setCount ist die Funktion, um den Wert zu ändern  
4    const [count, setCount] = React.useState<number>(0)  
5  
6    React.useEffect(() => {  
7      // wird ausgeführt, wenn count sich ändert  
8      console.log("count changed")  
9    }, [count])  
10 }
```

```

11  return (
12    <div>
13      <p>{count} mal geklickt.</p>
14      <button onClick={() => setCount(count + 1)}></button>
15      Button
16    </div>
17  )
18 }

```

Beispiel einer React-Komponente mit Hooks

Es lassen sich beliebig viele Komponenten verschachteln. Das Durchreichen der Parameter wird bei größeren Projekten sehr aufwändig – insbesondere in Bezug auf die Wartbarkeit. Um dies zu entschärfen, gibt es weitere Konzepte wie der React Context, der im Folgenden beschrieben wird.

6.3.1.4. React Context

Der React Context ermöglicht es einen Zustand über mehrere Komponenten hinweg zu benutzen, ohne ihn mittels Parameter an alle Unterkomponenten durchzureichen. Man kann den React Context mit einer globalen Variable vergleichen. Ein typischer Anwendungsfall für den React Context ist das Verwenden von Authentifizierungsdaten wie der Name über die gesamte Anwendung hinweg.

6.3.2. Einsatz im Projekt

React wurde verwendet, um die gesamte Website aufzubauen. Sie bildet alles ab, was der Benutzer sieht und mit der Plattform interagiert.

6.3.3. Grund für Technologieentscheidung

React wurde für die Entwicklung der Plattform verwendet, da es das meistgenutzte Frontend-Framework der Welt ist. Außerdem gab es ein großes Interesse der verschiedenen Entwickler, sich in dieses Framework einzuarbeiten.

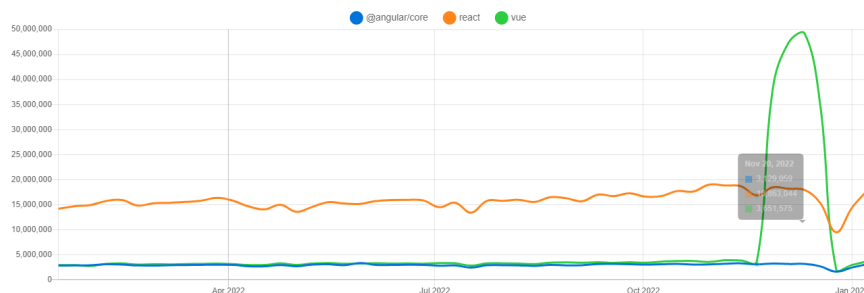


Abbildung (6.1) – Vergleich Downloadzahlen verschiedener Frontend Frameworks [4]

6.4. Material-UI

Material-UI (kurz MUI) ist eine Bibliothek, die es ermöglicht, Komponenten aus dem Material Design zu verwenden. Diese Komponenten sind konfigurierbar und können so an die eigenen Bedürfnisse angepasst werden. Trotzdem entsprechen Sie alle einer einheitlichen Designsprache, die von Google entwickelt wurde. ⁴

MUI bietet Komponenten für folgende Bereiche an:

- **Navigation** – Komponenten für die Navigation.
- **Inputs** – Komponenten für die Eingabe von Daten.
- **Layout** – Komponenten für das Layout der Website.
- **Data Display** – Komponenten für die Anzeige von Daten.
- **Feedback** – Komponenten für die Rückmeldung an den Benutzer.
- **Surfaces** – Komponenten für Oberflächen.
- **Utils** – Komponenten für die Unterstützung.

Diese Bibliothek wird von Material-UI SAS. entwickelt und ist Open-Source auf Github verfügbar. MUI bietet eine direkte Integration mit React. ⁵

6.4.1. Einsatz im Projekt

Da React nur sehr rudimentäre Komponenten für die Benutzeroberfläche bereitstellt, wurde MUI verwendet, um die Benutzeroberfläche zu gestalten. Es bietet eine große Auswahl an Komponenten, die direkt in React verwendet werden können.

6.4.2. Grund für Technologieentscheidung

MUI vereinfachte die Gestaltung der Benutzeroberfläche, da es eine große Auswahl an Komponenten bietet, die direkt in React verwendet werden können. Aus diesem Grund konnten die Entwickler schnell mit der Gestaltung der Benutzeroberfläche beginnen.

⁴Vgl. Google 2021 [5]

⁵Vgl. Material-UI 2022 [6]

6.5. Firebase

Firebase ist eine BaaS Plattform, die die Bereitstellung von Backend-Funktionalität ermöglicht inklusive Datenbank, Authentifizierung, Datei-Upload, etc. Dabei wird die Funktionalität in einzelne Module aufgeteilt, die unabhängig voneinander verwendet werden können. Die Plattform wird von Google entwickelt⁶ und besteht aus mehreren Komponenten, die nun kurz vorgestellt werden.

6.5.1. Firebase Authentication

Firebase Authentication ist ein Modul von Firebase, das die Authentifizierung von Nutzern ermöglicht. Dabei werden verschiedene Authentifizierungsmethoden unterstützt, wie z. B. E-Mail und Passwort, Google, Facebook, etc. Außerdem ist es möglich eigene Authentifizierungsmethoden zu implementieren, sowie Nutzer über Telefonnummern zu authentifizieren. Dabei können Nutzer auch in mehreren Geräten gleichzeitig eingeloggt sein.

Bis auf die Authentifizierungsmethode über Telefonnummern, die nur in den USA verfügbar ist, werden alle Authentifizierungsmethoden kostenlos angeboten.⁷

6.5.2. Firebase Realtime Database

Firebase Realtime Database ist ein Modul von Firebase, das die Bereitstellung einer Datenbank ermöglicht. Diese Datenbank ist eine NoSQL Datenbank, ähnlich wie MongoDB. Hier werden die Daten nicht relational in Dokumenten gespeichert, sondern in einer Baumstruktur. Updates in der Datenbank werden in Echtzeit an alle Nutzer gesendet, die sich mit der Datenbank verbinden.⁸

6.5.3. Firebase Cloud Firestore

Firebase Cloud Firestore ist ein Modul von Firebase, das die Bereitstellung einer Datenbank ermöglicht. Dabei wird die Datenbank in einzelne Dokumente aufgeteilt, die in einer Baumstruktur organisiert sind. Firestore ist die Weiterentwicklung der Realtime Database und bietet einige Vorteile gegenüber dieser. So ist die Datenbank in mehrere Regionen aufgeteilt, was die Verfügbarkeit erhöht. Außerdem ist es möglich, die Datenbank in mehrere Projekte aufzuteilen, was die Sicherheit erhöht. Abfragen in der Datenbank können mit Indexen optimiert werden, was die Performance verbessert.⁹

⁶Vgl. Firebase 2022 [7]

⁷Vgl. Firebase Authentication 2022 [8]

⁸Vgl. Firebase Realtime Database 2022 [9]

⁹Vgl. Firebase Cloud Firestore 2022 [10]

6.5.4. Firebase Storage

Firebase Storage ist ein Modul von Firebase, das die Bereitstellung von Datei-Upload ermöglicht. Dabei können Dateien in einem Bucket gespeichert werden, das in einzelne Ordner aufgeteilt ist. Der Datei-Upload kann über die Firebase Konsole oder über die Firebase SDK's erfolgen. Diese Funktion ist vergleichbar mit AWS S3 Buckets.¹⁰

6.5.5. Einsatz im Projekt

Firebase wurde im Projekt für die Bereitstellung der Datenbank und der Authentifizierung verwendet. Außerdem speichert es die Bilder, die von den Nutzern hochgeladen werden.

6.5.6. Grund für Technologieentscheidung

Durch Firebase konnte die Entwicklung der Backend-Funktionalität beschleunigt werden, da die Entwickler sich nicht um die Bereitstellung dieser Funktionalität kümmern mussten.

6.6. Github

Github ist eine Plattform, die es ermöglicht, Softwareprojekte zu verwalten. Diese Plattform wird von Github Inc. entwickelt. Github bietet eine direkte Integration mit Git. Die wichtigsten Funktionen von Github sind in Tabelle 6.1 aufgeführt.¹¹

Funktion	Beschreibung
Versionsverwaltung	Github bietet Versionsverwaltung auf Grundlage von Git an.
Projektmanagement	Anforderungen können als Issues angelegt und verwaltet werden.
Dokumentation	Mithilfe von Markdown Wikis.
Teamarbeit	In Form von Kommentaren, Reviews, etc.
Hosting	Bereitstellung statischer Seiten.
CI/CD	Automatisierte Tests und Deployment.

Tabelle (6.1) – Funktionen von Github

6.6.1. Einsatz im Projekt

Github wurde im Projekt für die Versionsverwaltung und CI verwendet. Die Versionsverwaltung wurde durch die Integration mit Git ermöglicht. Außerdem wurden Github Actions benutzt, welches automatisierte Tests und andere Sanity-Checks ermöglicht.

¹⁰Vgl. Firebase Storage 2022 [11]

¹¹Vgl. Github 2022 [12]

6.6.2. Grund für Technologieentscheidung

Github wurde im Projekt eingesetzt, da es eine gute Integration mit Git bietet und somit die Versionsverwaltung vereinfacht. Außerdem wurde durch die verfügbare CI Funktionalität die Qualität des Codes verbessert und stetig getestet werden. Die Entwicklungsgeschwindigkeit wurde dadurch vereinfacht.

6.7. Vercel

Vercel ist eine Hosting-Plattform, die es ermöglicht, statische Webseiten zu hosten. Diese Plattform wird von Vercel Inc. entwickelt. Vercel bietet eine direkte zu 6.6 und erstellt neue Deployments und Builds, sobald ein neuer Commit in Github verfügbar ist. Dies funktioniert auch mit mehreren Branches und Pull Requests. Jedes Deployment wird mit einer eigenen URL versehen, sodass mehrere Versionen der gleichen Webseite gleichzeitig verfügbar sind. So können Pull Requests getestet werden, bevor sie in den Master Branch gemerged werden.¹²

6.7.1. Einsatz im Projekt

Vercel wurde im Projekt für die Bereitstellung der Webanwendung verwendet. Sobald bei Github ein neuer Commit verfügbar ist, wird automatisch ein neues Deployment erstellt. Dies geschah für den main-Branch auf der Domain <https://daham.roser.dev> und für den dev-Branch auf der Domain <https://dev.daham.roser.dev>. Jeder andere Branch bekam sein eigenes Deployment auf einer eigenen Domain, welche von Vercel erstellt wurde.

6.7.2. Grund für Technologieentscheidung

Vercel wurde verwendet, um die Bereitstellung der Webanwendung zu vereinfachen. Nach Errichtung des Github Repositories und initialer Projekterstellung wurde die Integration mit Vercel in wenigen Klicks aktiviert und hostet seitdem kostenlos die Webanwendung.

¹²Vgl. Vercel 2022 [13]

6.8. Jira

Jira ist eine Plattform, die es ermöglicht, Softwareprojekte zu verwalten. Diese Plattform wird von Atlassian entwickelt.

Die wichtigsten Funktionen von Jira sind in Tabelle 6.2 aufgeführt.¹³

Funktion	Beschreibung
Projektmanagement	Anforderungen können als Issues angelegt und verwaltet werden.
Dokumentation	Mithilfe von Markdown Wikis.
Teamarbeit	In Form von Kommentaren, Reviews, etc.

Tabelle (6.2) – Funktionen von Jira

6.8.1. Einsatz im Projekt

In Jira wurden die einzelnen User Stories verwaltet und bearbeitet, sowie in Sprints eingeplant. Durch ein Kanbanboard wurden die einzelnen User Stories in den einzelnen Sprints angezeigt. Jira bietet eine direkte Integration mit Github.

6.8.2. Grund für Technologieentscheidung

Jira wurde verwendet, um die Verwaltung der User Stories zu vereinfachen. Durch die direkte Integration mit Github wurden die einzelnen User Stories mit den dazugehörigen Commits verknüpft. Dadurch konnte die Entwicklung der einzelnen User Stories nachvollzogen werden. Außerdem ist es ein sehr simples Tool, welches für die Verwaltung von User Stories sehr gut geeignet ist.

¹³Vgl. Atlassian 2023 [14]

7. Implementierung

In diesem Kapitel wird die Implementierung der Anwendung beschrieben.

Da der komplette Code und viele Screenshots den Rahmen dieses Kapitels sprengen würden, wird in den einzelnen Kapiteln nur ein Auszug gezeigt. Den kompletten Code findet man direkt im Github Repository unter <https://github.com/Jonasdero/digital-hometown-frontend>.

7.1. Landing Page

Bei dem Aufruf der Plattform „Digital Dahoam“ wird der Nutzer auf die Landing Page (s. Abbildung 7.1) weitergeleitet. Diese ist abhängig davon, ob der Nutzer eingeloggt ist oder nicht.

Auf der Landing Page wird die Vision der Plattform vorgestellt. Bei dem Projekt „Digital Dahoam“ geht es darum, dass sich Menschen in der Nachbarschaft vernetzen können. Die Plattform soll zudem die Menschen dazu motivieren, anderen zu helfen, bzw. selbst um Hilfe zu bitten. Insbesondere für neu zugezogene Menschen soll die Plattform eine Möglichkeit sein, die Umgebung zu entdecken und interessante Vereine kennenzulernen.

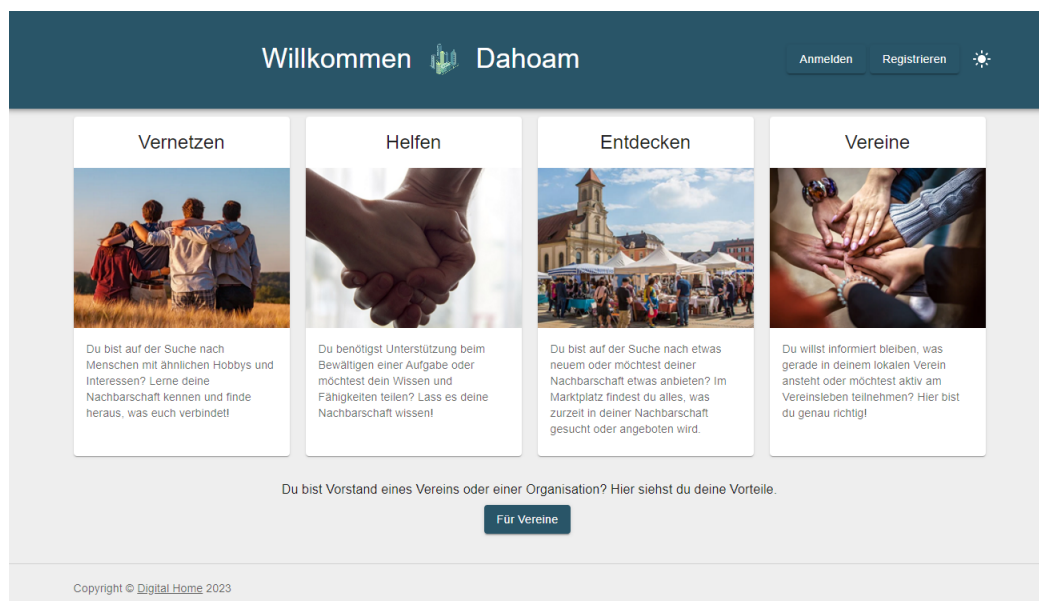


Abbildung (7.1) – Bildschirmaufnahme der Startseite für nicht eingeloggte Nutzer

Will sich ein Benutzer bspw. ein Vereinsvorstand mit seinem Verein registrieren, kann er über den Button „Für Vereine“ zur Landing Page für Vereine navigieren. Über die Landing Page für Vereine kann der Vorstand seinen Verein bei der Plattform registrieren oder sich einloggen (s. ??). Der Inhalt der Landing Page beschreibt, welche Vorteile die Plattform für Vereine bietet.

7.2. Profil

Die Profile eines Nutzers sind in der Anwendung sehr wichtig. Jeder Nutzer verwaltet sein eigenes Profil, welches er mit anderen Nutzern teilt. Dazu gehören Informationen wie Name, E-Mail-Adresse, Geburtsdatum, Geschlecht und ein Profilbild. Damit andere Nutzer die Informationen des Profils sehen können, muss das Profil öffentlich sein. Zudem sollen Benutzer der Anwendung durch Interessen, Profilbilder und persönliche Beschreibungen voneinander unterscheidbar sein und sich so besser vernetzen können. Um sich mit anderen Nutzern zu verbinden, ist es wichtig, dass diese Informationen in der Anwendung gespeichert werden.

Außerdem soll die soziale Interaktion zwischen Nutzer ermöglicht werden. Dazu gehören Funktionen wie das Folgen von anderen Nutzern oder auch das Schreiben von Nachrichten zwischen zwei Nutzern oder in Gruppen. Dies soll direkt vom Profil eines Nutzers aus möglich sein.

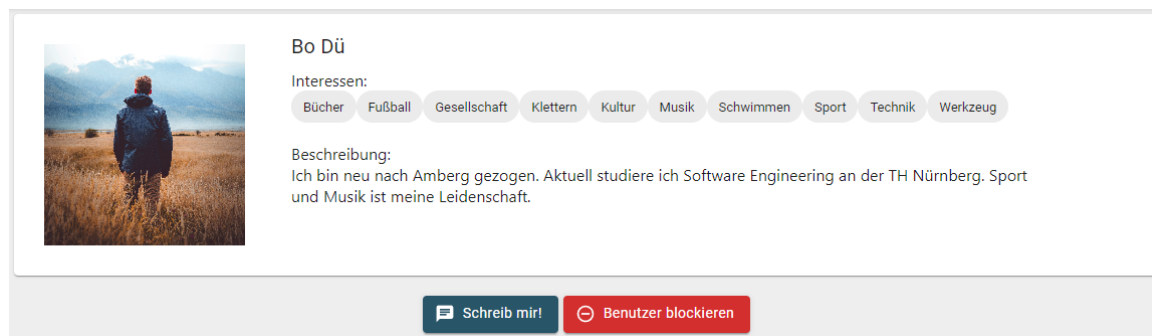


Abbildung (7.2) – Übersicht eines Benutzerprofils

Hier sieht man die verschiedenen Informationen, die ein Nutzer in seinem Profil angeben kann. In den nächsten Seiten wird die Implementierung dieser Funktionen beschrieben. Diese wird aufgeteilt in die verschiedenen Bereiche: *Anmeldung & Registrierung*, *persönliches Accountmanagement*, *Profilseite & Profilbilder* und *blockierte Nutzer*.

7.2.1. Anmeldung & Registrierung

Den Nutzern wird die Möglichkeit gegeben, sich mit einer E-Mail-Adresse und einem Passwort anzumelden oder eine direkte Anmeldung über OAuth mit Google zu nutzen. Dies wird beides durch die Firebase Authentication API ermöglicht.

Sobald ein Nutzer sich registriert hat, wird von Firebase intern ein Benutzerprofil erstellt, welches wichtige Nutzermetadaten und Authentifizierungsinformationen enthält. Außerdem wird bei der Anmeldung über Google das Profilbild mit in Firebase gespeichert. Die Anmeldemaske sieht wie folgt aus:

Um diese Daten in der Anwendung zu speichern, wird ein eigener Benutzerdatensatz angelegt, sobald ein Nutzer registriert wurde und Daten aus dem internen Benutzerprofil von

The image shows a login form titled 'Anmelden' (Login) with a lock icon. It contains two input fields: 'Email *' and 'Passwort *' (Password *). Below the password field is a dark blue button labeled 'Anmelden'. Underneath the button are two social media icons: Google (G) and Facebook (f). At the bottom, there are two links: 'Passwort vergessen?' (Forgot password?) and 'Noch keinen Account? Registriere dich hier!' (No account yet? Register here!).

Abbildung (7.3) – Anmeldemaske

Firebase mit übertragen. Hierbei wird unterschieden, ob der aktuelle Benutzer ein normaler Nutzer oder ein Verein ist. Diese werden in unterschiedlichen „Collections“ gespeichert. Bei der Anmeldung wird dann geprüft, welchen Typ der Benutzer hat und die Daten entsprechend geladen. Dadurch kann mit einer Datenstruktur gearbeitet werden, die für beide Typen geeignet ist. Andere Ansichten basieren dann häufig auf der Unterscheidung zwischen Nutzern und Vereinen. Die Struktur des Datensatzes kann man in Abbildung A.1 sehen.

7.3. Authentifizierung

Essentiell für ein soziales Netzwerk ist es, dass die Benutzer sich mit einem Profil registrieren können. Wie das im Detail funktioniert und wie die Authentifizierung implementiert wurde, wird in diesem Kapitel beschrieben.

7.3.1. Verwendung des „Firebase-Authentication“-Services

Die Implementierung der Authentifizierung wird mit dem Backend-as-a-Service-Anbieter Firebase durchgeführt. Firebase bietet eine Schnittstelle, die es ermöglicht, Profile anzulegen sowie den Anmelde- und Registrierungsprozess mit wenigen Zeilen Code zu implementieren.

Durch die verschiedenen Authentifizierungsmethoden, die Firebase anbietet, kann der Nutzer sich mit E-Mail und Passwort, Google oder Facebook anmelden. Die Authentifizierungsmethoden können einfach in der Firebase-Konsole aktiviert werden.

Es wird zu dem die Möglichkeit geboten, dass Nutzer, die ihre Zugangsdaten vergessen haben, eine E-Mail mit einem Link zum Zurücksetzen des Passworts erhalten.

Die Beschriebenen Funktionen decken also einen standardmäßigen Anmelde- und Registrierungsprozess ab.

7.3.2. Zugriff auf die Profilinformationen

Wie bei der React-Einführung beschrieben, ist es aufwendig, Informationen wie die des Profils über die gesamte Anwendung hinweg durchzureichen. Dafür wurde auch bereits die Möglichkeit des React Contexts vorgestellt. Da dieses Konzept auch für die Profilinformationen eingesetzt wird hier nochmal ein Beispiel der Implementierung gezeigt.

Das in Listing 7.1 stellt das Interface des Contexts dar, das für alle Components befüllt wird. Die Profilinformationen sind im Attribute `currentUser` gespeichert.

```

1 interface AuthContextI {
2   currentUser: User | Club | undefined | null
3   setCurrentUser: React.Dispatch<React.SetStateAction<User | Club |
4     undefined | null>>
5   logOut: () => void
6   logIn: (email: string, password: string) => void
7   signUpWithEmail: (email: string, password: string, displayName: string,
8     isOrg: boolean) => Promise<void>
9   signUpOAuth: (providerName: "google" | "facebook", isOrg: boolean) => void
10  // usw.
11 }

```

Listing (7.1) – Auszug aus dem Interface des Authentifizierungskontexts

Um den Vorteil des Contexts für die Codequalität hervorzuheben wird in Listing 7.2 dargestellt, wie einfach es möglich ist, die zentral befüllten Attribute in einer beliebigen React-Komponente zu verwenden. Hierbei wird das Prinzip der Higher-Order-Components (HOC) genutzt, das zuvor implementiert wurde. Es ist lediglich notwendig, die implementierte Komponente mit dem HOC `withAuth` zu umschließen. Dieses HOC stellt der jeweiligen Komponente die Attribute des Contexts als Props zur Verfügung.

```

1 function SignOut({ logOut }: AuthContextI) {
2   useEffect(() => logOut(), [logOut])
3   return <Navigate to="/" />
4 }
5
6 export default withAuth(SignOut)

```

Listing (7.2) – Verwendung des Authentifizierungskontext-HOCs

Das Component `SignOut` bekommt hier beispielsweise durch das HOC automatisch die Funktion `logout` als Prop übergeben, die dann zum Ausloggen des Benutzers verwendet wird.

7.3.3. Authentifizierungsprozess

Im folgenden wird nun anhand von Bildschirmaufnahmen gezeigt, wie der Authentifizierungsprozess abläuft.

Beim Klick auf den Registrieren-Button der Landing-Page wird man auf eine Seite weitergeleitet, über die Benutzername, Email und Passwort eingegeben werden können (s. Abbildung 7.4). Fehlerhafte Eingaben der E-Mail-Adresse, sowie die redundante Eingabe des Passworts werden hierbei validiert.

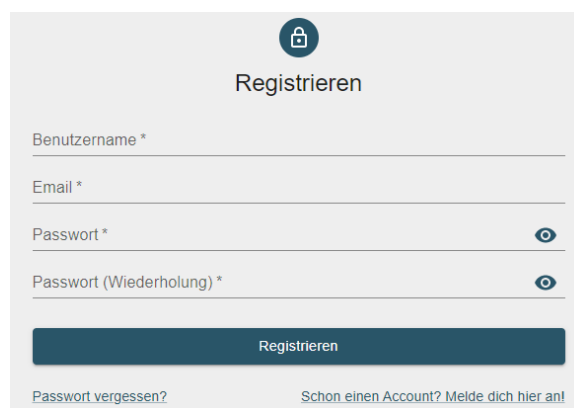
The image shows a registration form titled 'Registrieren' with a lock icon. It contains four input fields: 'Benutzername *', 'Email *', 'Passwort *' (with an eye icon for toggling visibility), and 'Passwort (Wiederholung) *' (also with an eye icon). Below the fields is a dark blue 'Registrieren' button. At the bottom, there are two links: 'Passwort vergessen?' and 'Schon einen Account? Melde dich hier an!'.

Abbildung (7.4) – Registrierungsseite

Nach der erfolgreichen Registrierung des Nutzers, wird er zunächst auf eine Seite geführt, auf der er seine Profilinformationen um das Geburtsdatum und die Postleitzahl ergänzen kann (s. Abbildung 7.5).

Nach der erfolgreichen Registrierung wird der Nutzer auf sein Profil weitergeleitet, wo er u. a. Interessen hinterlegen kann, um von Nutzern gefunden zu werden, welche die gleichen Interessen haben. Details hierzu sind dem Projektbericht von Jonas Roser zu entnehmen.

Eine besonders einfache Möglichkeit, sich bei der Plattform zu registrieren, bzw. anzumelden ist die Authentifizierung über Google. Beim Klick auf das Google-Icon öffnet sich direkt der Google-OAuth-Dialog (s. Abbildung 7.6). Dies steigert die Usability der Plattform, da der Nutzer nicht mehr die E-Mail-Adresse und das Passwort eingeben muss.

Wenn der Nutzer sich mit einer E-Mail-Adresse registriert hat, kann er sich auf folgender Seite anmelden (s. Abbildung 7.7).

Abbildung (7.5) – Ergänzen der Profilinformationen beim ersten Anmelden

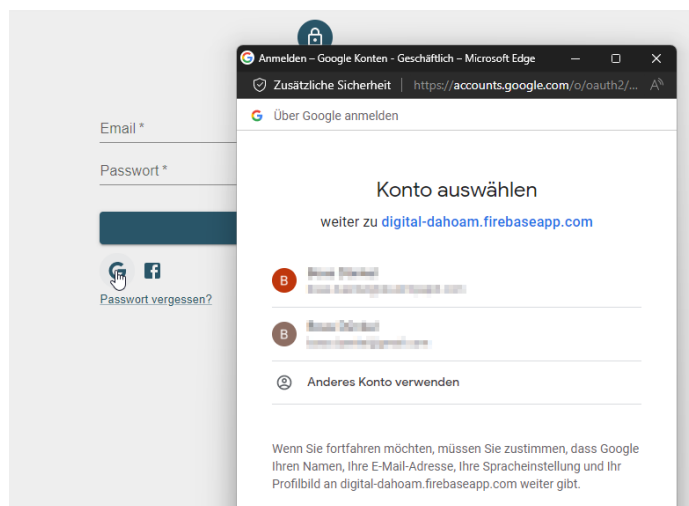
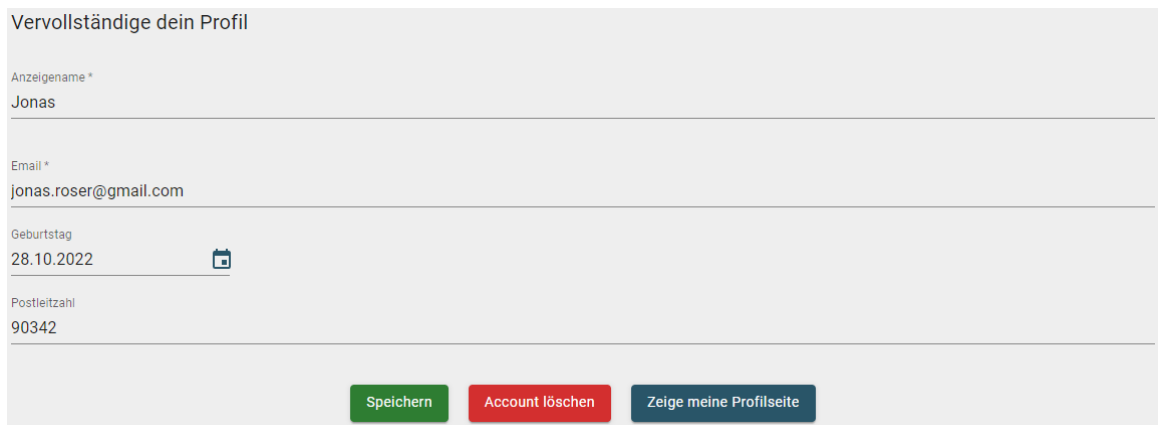


Abbildung (7.6) – Authentifizierungs-Dialog beim Anmelden mit Google

Abbildung (7.7) – Anmeldeseite

7.3.4. Persönliches Accountmanagement

Um die Daten des Benutzerprofils zu verwalten, gibt es eine Seite, auf der der Nutzer seine persönlichen Daten ändern kann. Diese werden dann in der Anwendung aktualisiert und in der Datenbank gespeichert. Er hat hier die Möglichkeit seinen Namen, seine E-Mail, sein Geburtsdatum und seine Postleitzahl zu ändern. Hier kann außerdem der komplette Account gelöscht werden, um die Daten des Nutzers zu löschen.



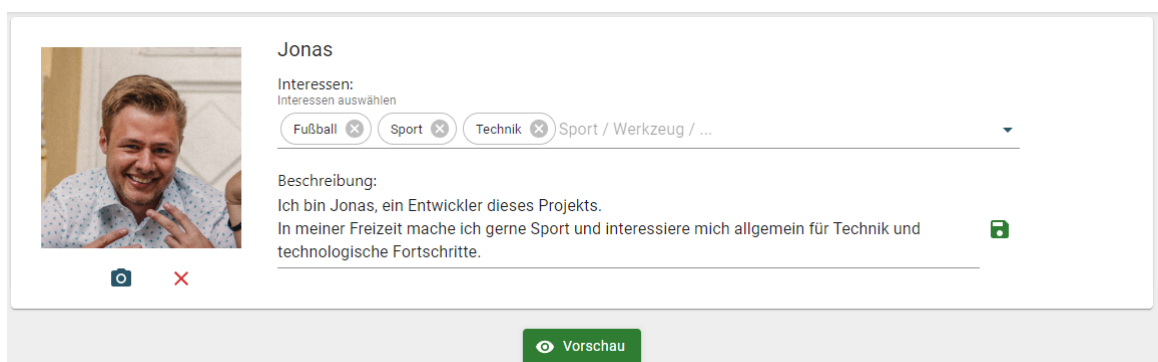
The screenshot shows a form titled 'Vervollständige dein Profil'. It contains four input fields: 'Anzeigename *' with the value 'Jonas', 'Email *' with 'jonas.roser@gmail.com', 'Geburtsdag' with '28.10.2022' and a calendar icon, and 'Postleitzahl' with '90342'. At the bottom, there are three buttons: 'Speichern' (green), 'Account löschen' (red), and 'Zeige meine Profilseite' (dark blue).

Abbildung (7.8) – Benutzereinstellungen

7.3.5. Profilseite & Profilbilder

Auf der Profilseite des Nutzers kann er seine persönlichen Daten einsehen und bearbeiten. Hier können Interessen und eine Beschreibung hinzugefügt werden. Außerdem kann er sein Profilbild ändern, indem er auf das Profilbild oder auf den Knopf mit der Kamera klickt. Dies wird dann im Firebase Storage gespeichert und in der Datenbank verlinkt.

Das Ganze ist so aufgebaut, dass Nutzer zwischen einer Vorschau, also der Sicht, die auch andere Nutzer von seinem Profil sehen und der Bearbeitungssicht unterscheiden können.



The screenshot shows a user profile for 'Jonas'. On the left is a profile picture of a man. To the right, the name 'Jonas' is displayed. Below the name is a section for 'Interessen:' with the subtext 'Interessen auswählen'. There are three tags: 'Fußball', 'Sport', and 'Technik', each with a close button. To the right of these tags is a dropdown menu showing 'Sport / Werkzeug / ...'. Below the interests is a 'Beschreibung:' section with the text 'Ich bin Jonas, ein Entwickler dieses Projekts. In meiner Freizeit mache ich gerne Sport und interessiere mich allgemein für Technik und technologische Fortschritte.' and a green plus icon to add more text. At the bottom, there is a green button labeled 'Vorschau' with an eye icon.

Abbildung (7.9) – Persönliches Profil

7.3.6. Blockierte Nutzer

Um die Privatsphäre der Nutzer zu schützen, können diese andere Nutzer blockieren. Blockierte Nutzer können dann nicht mehr auf das Profil des Blockierenden zugreifen und auch keine Nachrichten mehr schreiben.

Über die Profilseite kann ein Nutzer einen anderen Nutzer blockieren.

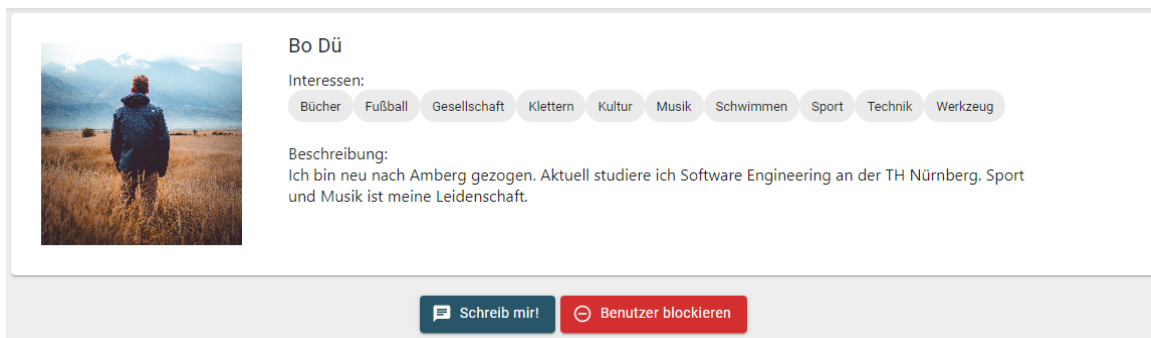


Abbildung (7.10) – Blockieren eines Nutzers

Blockierte Nutzer kann man dann in der Anwendung unter dem Menüpunkt *Blockiert* einsehen. Dort wird jeder Nutzer oder Verein aufgeführt, welcher blockiert ist. Mit einem Klick auf das „X“ wird der Block aufgehoben.

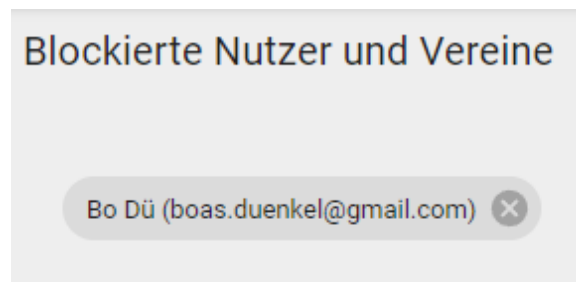


Abbildung (7.11) – Blockierte Nutzer

7.4. Beiträge

Beiträge sind ein anderes wichtiges Feature von „Digital Dahoam“. Hiermit können Nutzer Informationen austauschen, die für andere Nutzer interessant sein könnten. Beiträge können von allen Nutzern erstellt werden, die sich registriert haben. Es gibt folgende Typen von Beiträgen:

- **Anfrage:** Hier können Nutzer eine Anfrage stellen, die dann von anderen Nutzern beantwortet werden kann. Anfragen können auch benutzt werden, wenn bestimmte Gegenstände im Haushalt fehlen, z. B. ein bestimmtes Werkzeug oder ein bestimmtes Lebensmittel.

- **Angebot:** Hier können Nutzer ein Angebot erstellen, das dann von anderen Nutzern angenommen werden kann. Dies kann wie Ebay-Kleinanzeigen benutzt werden, um überflüssige Gegenstände zu verkaufen.
- **Information:** Hier können Nutzer Informationen teilen, die für andere Nutzer interessant sein könnten.
- **Veranstaltung:** Hier können Nutzer Veranstaltungen erstellen, die dann von anderen Nutzern besucht werden können.

7.4.1. Beiträge erstellen

Beiträge können mit einem Klick auf den Button *Beiträge erstellen* erstellt werden. Hier öffnet sich ein Pop-up, welchem der Nutzer den Titel, die Beschreibung, den Beitragstyp und die Kategorie des Beitrags eingeben kann. Außerdem werden für jeden Beitrag das Startdatum, also ab wann der Beitrag gültig ist und angezeigt wird, und das Enddatum, also bis wann der Beitrag gültig ist und angezeigt wird, festgelegt. Für Veranstaltungen gibt es zusätzlich noch den Ort und das Datum. Sobald auf *absenden* geklickt wird, wird der Beitrag in der Datenbank gespeichert und auf der Startseite angezeigt.

Erstelle einen Beitrag

Gib deinem Beitrag einen Titel *

Beitragstyp auswählen *

Gültigkeit ab 26.01.2023 Gültigkeit bis 26.01.2024

Schreibe deinen Beitrag *

Beitragskategorien auswählen

Abbruch Absenden

Pflichtfelder

Abbildung (7.12) – Beiträge erstellen

7.4.2. Alle Beiträge

Unter *Alle Beiträge* findet man alle Beiträge, die ein valides Gültigkeitsdatum haben. Diese werden mit Titel, Beschreibung, Kategorie und Beitragstyp angezeigt. Für Veranstaltungen gibt es zusätzlich noch den Ort und das Datum.



Abbildung (7.13) – Beitrag

Ein Problem, was es während der Implementierung zu lösen galt, war die richtige Filterung der Beiträge. Die hier verwendete Sortierfunktion (siehe II) wurde dann für die restlichen Listen verwendet und die gefilterten Beiträge dort nur noch verfeinert. Diese musste folgendes erfüllen:

- Eigene Beiträge werden immer angezeigt.
- Beiträge, die nicht mehr gültig sind, werden nicht angezeigt.
- Beiträge von blockierten Nutzern werden nicht angezeigt.
- Beiträge müssen nach Datum sortiert werden.

Mit dem Klick auf die 3 kleinen Punkte in der rechten oberen Ecke öffnet man das Beitragsmenü. Hier gibt es folgende Punkte:

- **Beitrag zum Merkzettel:** Hinzufügen eines Inhalts zum Merkzettel.
- **Details:** Anzeigen der Details des Beitrags. (siehe A.2)
- **Zum Autor:** Direkter Link zum Profil des Autors.
- **Nachricht an Autor:** Direkter Link zur Nachrichtenfunktion, um dem Autor eine Nachricht zu schicken.

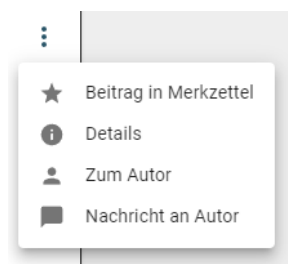


Abbildung (7.14) – Beitragsmenü

7.4.3. Profilseite

Auf der Profilseite kann der Nutzer werden seine Beiträge angezeigt. Hier werden auch abgelaufene Beiträge angezeigt und grau hinterlegt.

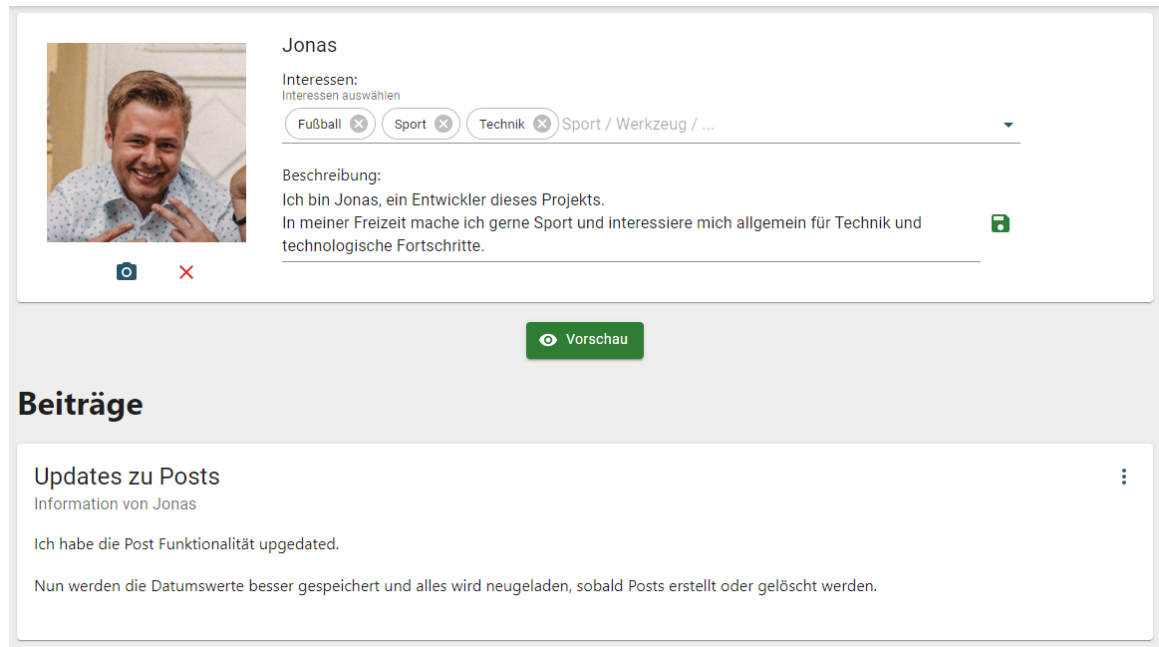


Abbildung (7.15) – Beiträge des Nutzers

7.4.4. Merkzettel

Beiträge, die für den Merkzettel markiert sind erscheinen dort, und können dort auch wieder entfernt werden. Außerdem werden sie nach Beitragstyp gruppiert.



Abbildung (7.16) – Merkzettel

7.4.5. Marktplatz

Auf dem Marktplatz können Personen und Beiträge durchsucht werden. Durch den Filter auf Kategorien, Informationen oder Veranstaltungen findet man schnell den richtigen Beitrag.

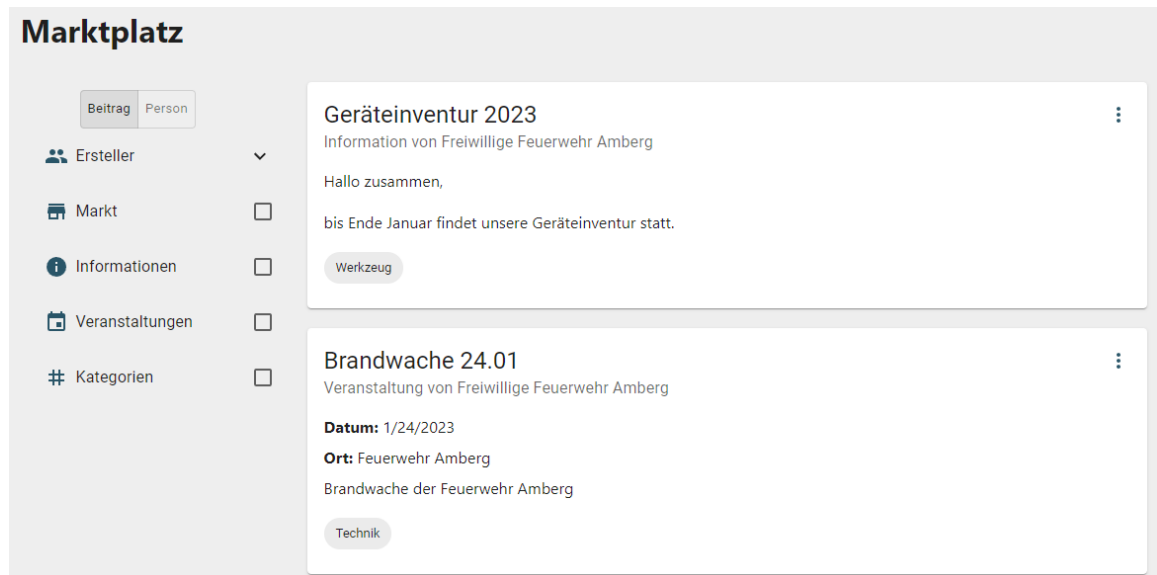


Abbildung (7.17) – Marktplatz

7.4.6. Dashboard

Auf der Startseite werden aktuelle Beiträge aus der Nähe angezeigt, sowie die Veranstaltungen, die der Nutzer auf dem Merkzettel hat.

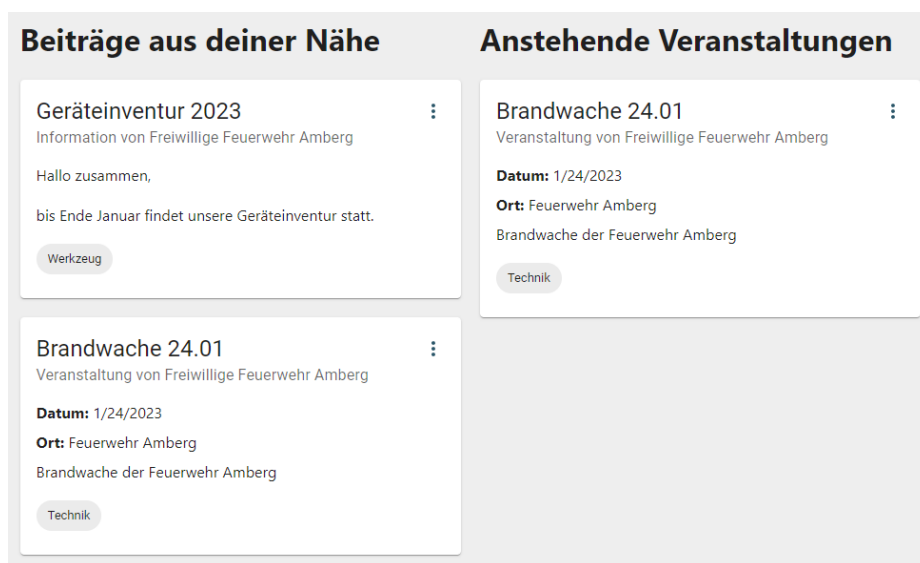


Abbildung (7.18) – Dashboard

7.5. Chat

Ein wichtiges Tool für das soziale Netzwerk „Digital Hometown“ ist der Chat, da hier die Personen in Kontakt treten und sich austauschen können.

Auf technischer Ebene sind für einen Chat mehrere Technologien erforderlich, wodurch die Umsetzung nicht trivial ist. Neben den Laden der Bereits gesendeten Nachrichten und dem Absenden von Nachrichten, ist es für eine gute Usability notwendig, dass neue Nachrichten sofort bei allen Chatteilnehmern sichtbar sind. Diese Anforderung kann in effizienter Weise durch die Verwendung von bidirektionalen WebSocket-Verbindungen ermöglicht werden.

7.5.1. Firebase Realtime Database für den Chat

Für den Chat wird die Firebase Realtime Database (Realtime DB) verwendet. Dadurch wird die Implementierung deutlich vereinfacht, da die entsprechende JavaScript Bibliothek einfach in die React Anwendung integriert werden kann. Eine Echtzeitsynchronisation zwischen den Clients (Browseranwendungen der User) und der Datenbank kann mit geringem Aufwand umgesetzt werden. Die Realtime DB ist eine NoSQL Datenbank. Die Firebase DB wird in diesem Projekt ausschließlich für den Chat verwendet und hat dabei zwei Hauptwurzelemente (s. Abbildung 7.19) verwendet werden. Bei dem Design wurde darauf geachtet, dass die einzelnen Hauptwurzelemente keine zu große Verschachtelung aufweisen, um eine gute Performanz zu gewährleisten.



Abbildung (7.19) – Hauptwurzelemente der Realtime DB

7.5.2. Autorisierung der Nachrichten und Chaträume

Eine wichtige Anforderung für die Implementierung einer Chatfunktionalität ist es, dass Nachrichten nur von Mitgliedern des jeweiligen Chatraums gelesen werden können. Dies wird mit den Realtime DB Regeln umgesetzt. Ein Auszug hiervon ist in Listing 7.3 dargestellt. Kurz zusammengefasst ermöglicht diese Konfiguration, dass nur Nachrichten von Benutzern gelesen und geschrieben werden können, die ein Mitglied des Chatraumes sind.

```

1 {
2   "rules": {
3     "messages": {
4       "$roomId": {
5         ".read": "root.child('rooms/' + $roomId + '/members').hasChild(auth
          .uid)",
  
```

```

6      ".write": "root.child('rooms/' + $roomId + '/members').hasChild(
      auth.uid)"
7    },
8    "messages": {
9      ".indexOn": "sendAt"
10   }
11 },
12 ...
13 }
14 }

```

Listing (7.3) – Realtime DB Regeln für Chatnachrichten

7.5.3. Benutzung des Chats

Im folgenden wird anhand von Screenshots die Benutzung des Chats beschrieben. Abbildung 7.20 zeigt die Standardansicht des Chats. Auf der linken Seite sind die Chaträume aufgelistet. Die Chaträume sind in zwei Kategorien unterteilt. Zum einen gibt es die Chaträume, die mit Einzelpersonen erstellt wurden und zum anderen die Chaträume mit mehreren Personen (Gruppenchats). In der Liste der Chaträume werden Gruppenchats mit einem Gruppenicon hervorgehoben.

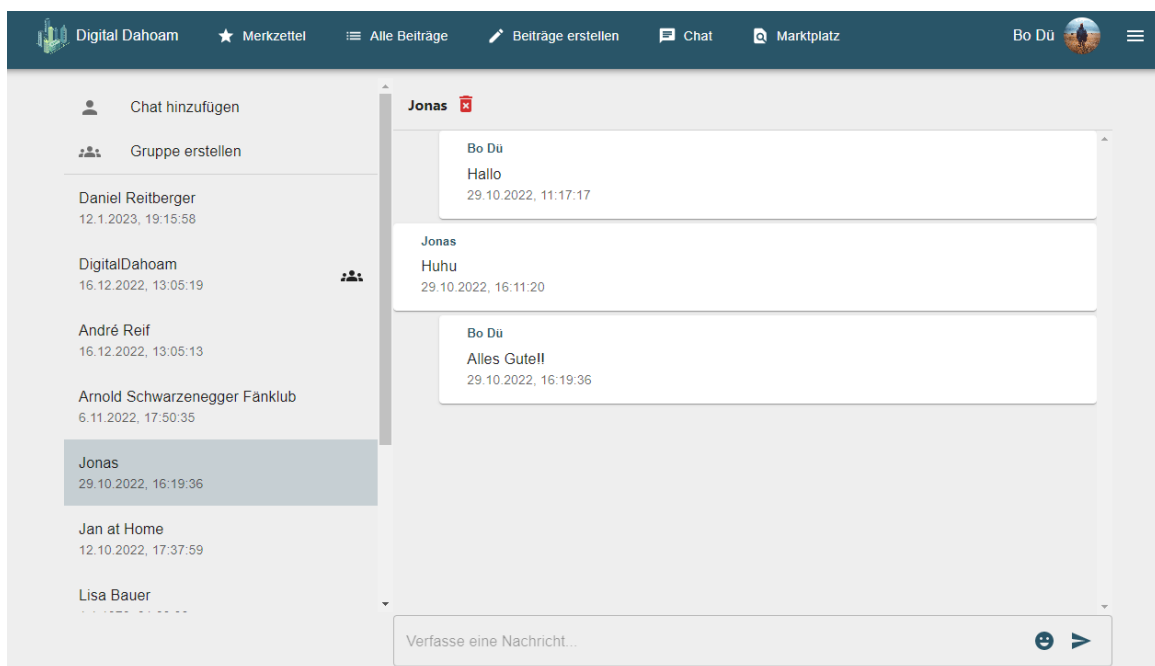


Abbildung (7.20) – Ansicht des Chats in der Anwendung

Über den Button „Chat hinzufügen“ wird die in Abbildung 7.21 dargestellte Ansicht geöffnet, wo alle Personen der Plattform angezeigt werden. Hier kann ein neuer Chat mit einer Einzelperson erstellt werden.

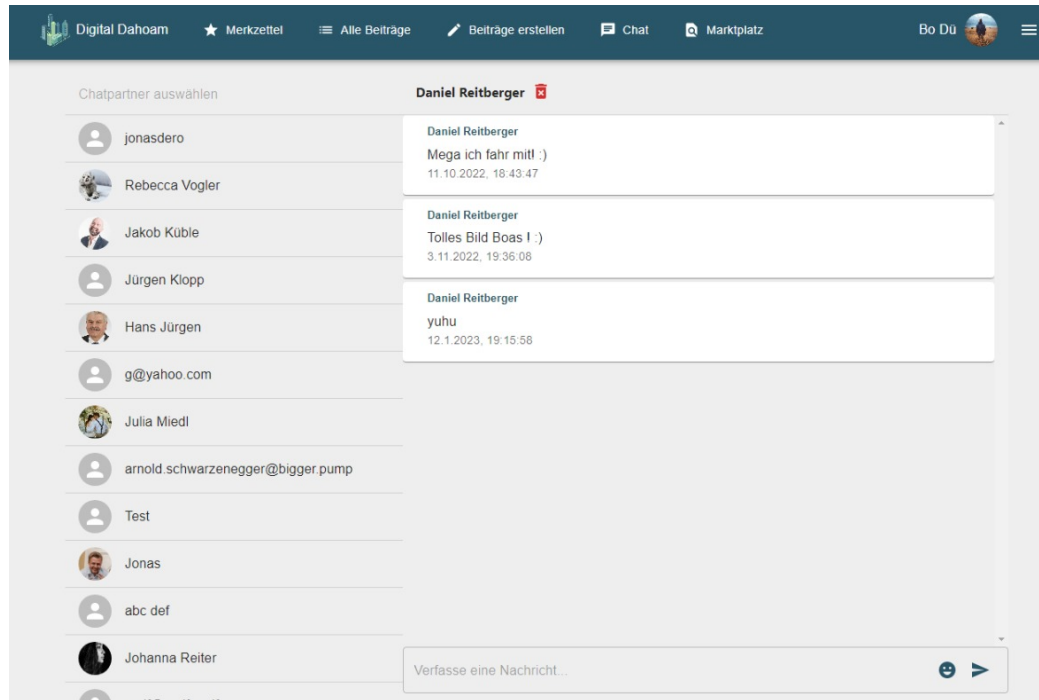


Abbildung (7.21) – Hinzufügen von Chats mit Einzelpersonen

Das Hinzufügen von Gruppenchats erfolgt über den Button „Gruppe hinzufügen“. Der Ablauf sieht hier wie folgt aus. Zuerst wird automatisch ein neuer Chatraum erstellt. Wird dieser ausgewählt, lässt sich über einen Button die in Abbildung 7.22 dargestellte Ansicht öffnen. Hier können die Mitglieder des Chatraumes hinzugefügt werden, indem die entsprechenden Personen aus der Liste ausgewählt werden. Es besteht zudem die Möglichkeit, der Chatgruppe einen anderen Namen zu geben.

7.5.4. Verknüpfung von Beiträgen mit dem Chatraum des Beitragautors

Die Motivation des in diesem Absatz beschriebenen Features wird mit folgendem Fallbeispiel erläutert.

Angenommen, ein Benutzer meldet sich an, da er an Fußball interessiert ist und gerne mit Menschen aus seiner Nachbarschaft gerne zum Fußball spielen treffen möchte. Er sieht nun einen Beitrag, der genau diesem Bedürfnis entspricht. Wie in Abbildung 7.23 dargestellt, kann er nun über den Button „Nachricht an Autor“ direkt zum Chatraum des Beitragautors gelangen. Dort kann er den Beitragautor anschreiben und sich mit ihm zum Fußball spielen treffen.

Im Hintergrund muss dafür der gemeinsame Chatraum ermittelt werden. Falls dieser noch nicht existiert, muss er erstellt werden. Wird dann der Nutzer auf die Chatseite weitergeleitet, öffnet sich automatisch der entsprechende Chatraum.

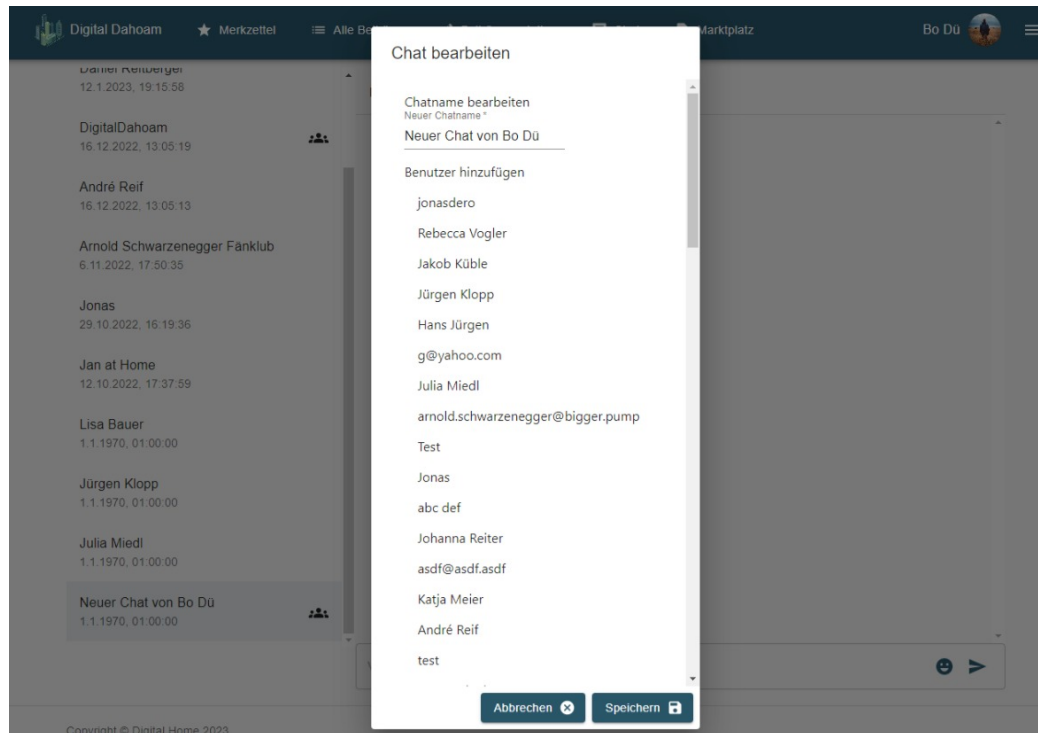


Abbildung (7.22) – Chatgruppe umbenennen und Mitglieder hinzufügen



Abbildung (7.23) – Möglichkeit von einem Beitrag direkt zum Chatraum des Beitragautors zu gelangen

8. Testing

9. Fazit

A. Anhang

I. Bilder

```
blocked: []  
  
dateOfBirth: 1666990800000  
  
desc: "Ich bin Jonas, ein Entwickler dieses Projekts. In meiner Freizeit mache ich  
gerne Sport und interessiere mich allgemein für Technik und  
technologische Fortschritte."  
  
displayName: "Jonas"  
  
email: "jonas.roser@gmail.com"  
  
favoritePosts: ["mFCoElDAPOTxuJCCbJCE", "...]  
  
friends  
  
id: "DCNu11SgrTYrBWu1bHxPjhYvllQ2"  
  
interests: ["Fußball", "Sport", "Tech...]  
  
isOrg: false  
  
photoURL: "https://firebasestorage.googleapis.com/v0/b/digital-  
dahoam.appspot.com/o/images%2FDCNu11SgrTYrBWu1bHxPjhYvllQ2?  
alt=media&token=a90ec139-31e7-4e65-bc97-a8b150ecc187"  
  
postCode: 90342
```

Abbildung (A.1) – Benutzerdatensatz in Firestore

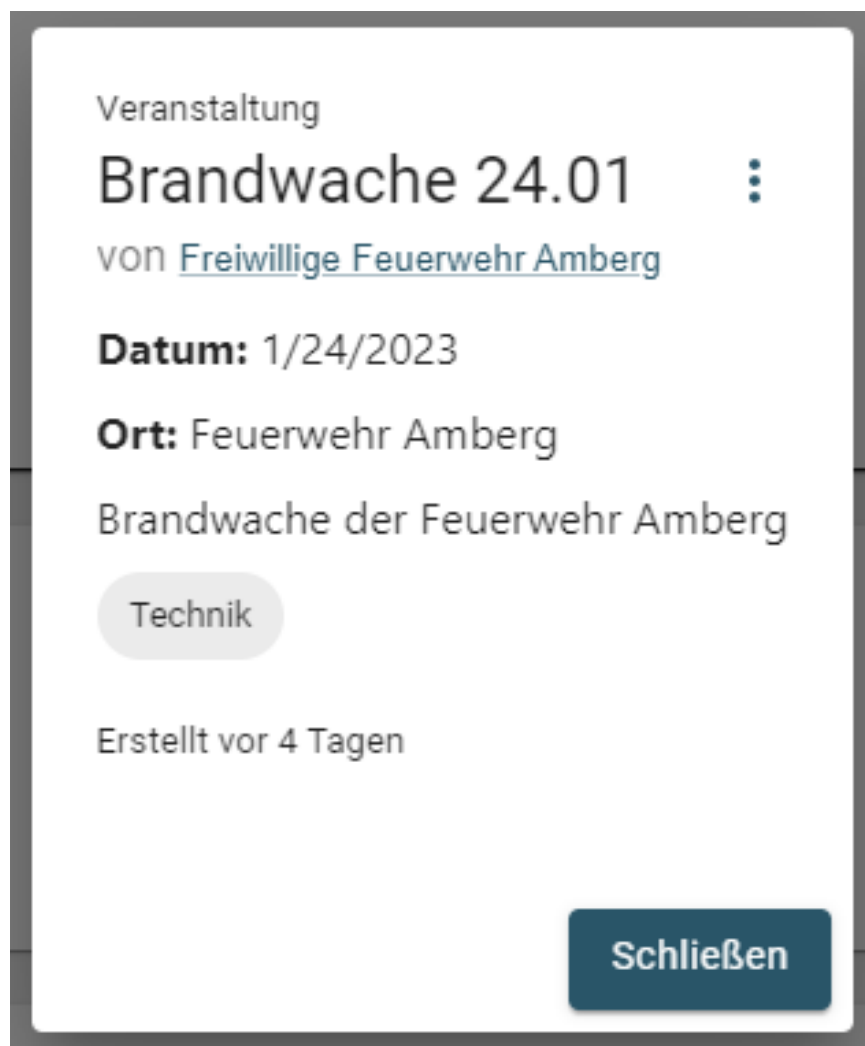


Abbildung (A.2) – Beitragsdetails

II. Code

```
1 const filtered = posts
2 .filter((post) => !currentUser?.blocked?.includes(post.author.id))
3 .filter((post) => {
4   // check validity
5   if (currentUser?.id === post.author.id) {
6     return true
7   }
8
9   if (!(post.validityStart && post.validityEnd)) {
10    // no validity set
11    return true
12  }
13  console.log(
14    moment(post.validityStart).toDate().getTime(),
15    moment(post.validityStart).toDate().getTime() <= new Date().getTime(),
16  )
17  if (
18    moment(post.validityStart).toDate().getTime() >= new Date().getTime() &&
19    moment(post.validityEnd).toDate().getTime() <= new Date().getTime()
20  ) {
21    return true
22  } else {
23    return false
24  }
25 })
```

Filterfunktion der Beiträge

Abbildungsverzeichnis

5.1.	Übersicht der technischen Softwarearchitektur	7
6.1.	Vergleich Downloadzahlen verschiedener Frontend Frameworks [4]	11
7.2.	Übersicht eines Benutzerprofils	18
7.3.	Anmeldemaske	19
7.8.	Benutzereinstellungen	23
7.9.	Persönliches Profil	23
7.10.	Blockieren eines Nutzers	24
7.11.	Blockierte Nutzer	24
7.12.	Beiträge erstellen	25
7.13.	Beitrag	26
7.14.	Beitragsmenü	26
7.15.	Beiträge des Nutzers	27
7.16.	Merkzettel	27
7.17.	Marktplatz	28
7.18.	Dashboard	28
A.1.	Benutzerdatensatz in Firestore	A
A.2.	Beitragsdetails	B

Tabellenverzeichnis

6.1. Funktionen von Github	14
6.2. Funktionen von Jira	16

Literaturverzeichnis

- [1] Typescript, “JavaScript With Syntax For Types.” 2022. [Online]. Available: <https://www.typescriptlang.org/>
- [2] —, “Documentation - React,” Jan. 2023. [Online]. Available: <https://www.typescriptlang.org/docs/handbook/react.html>
- [3] React, “React – A JavaScript library for building user interfaces,” 2022. [Online]. Available: <https://reactjs.org/>
- [4] NPM, “@angular/core vs react vs vue | npm trends,” Jan. 2023. [Online]. Available: <https://npmtrends.com/@angular/core-vs-react-vs-vue>
- [5] Google, “Design - Material Design,” May 2021. [Online]. Available: <https://m2.material.io/design>
- [6] MUI, “MUI: The React component library you always wanted,” 2022. [Online]. Available: <https://mui.com/>
- [7] G. Firebase, “Firebase,” 2022. [Online]. Available: <https://firebase.google.com/>
- [8] G. Authentication, “Firebase Authentication,” 2022. [Online]. Available: <https://firebase.google.com/docs/auth?hl=de>
- [9] F. Realtime Database, “Firebase Realtime Database,” 2022. [Online]. Available: <https://firebase.google.com/docs/database?hl=de>
- [10] G. Firestore, “Firestore,” 2022. [Online]. Available: <https://firebase.google.com/docs/firestore?hl=de>
- [11] G. CloudStorage, “Cloud-Speicher für Firebase | Cloud Storage for Firebase,” 2022. [Online]. Available: <https://firebase.google.com/docs/storage?hl=de>
- [12] Github, “Build software better, together,” 2022. [Online]. Available: <https://github.com>
- [13] Vercel, “Develop. Preview. Ship. For the best frontend teams – Vercel,” 2022. [Online]. Available: <https://vercel.com/>
- [14] Atlassian, “DH Board - Agile board - Jira,” Jan. 2023. [Online]. Available: <https://digitaldahoam.atlassian.net/jira/software/projects/DH/boards/1>