

Nochba – App für Nachbarschaftshilfe

DIPLOMARBEIT

verfasst im Rahmen der

Reife- und Diplomprüfung

an der

Höheren Abteilung für Informatik

Eingereicht von:

Arsham Edalatkhan
Sandin Habibovic
Martin Hausleitner

Betreuer:

Rainer Stropek

Projektpartner:

IKT Linz GmbH - Open Common Linz

Leonding, März 2023

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Die Arbeit wurde bisher in gleicher oder ähnlicher Weise keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Die vorliegende Diplomarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

Leonding, April 2022

S. Schwammal & S. Schwammal

Abstract

Brief summary of our amazing work. In English. This is the only time we have to include a picture within the text. The picture should somehow represent your thesis. This is untypical for scientific work but required by the powers that are. Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.



Zusammenfassung

Zusammenfassung unserer genialen Arbeit. Auf Deutsch.

Das ist das einzige Mal, dass eine Grafik in den Textfluss eingebunden wird. Die gewählte Grafik soll irgendwie eure Arbeit repräsentieren. Das ist ungewöhnlich für eine wissenschaftliche Arbeit aber eine Anforderung der Obrigkeit.

Bitte auf keinen Fall mit der Zusammenfassung verwechseln, die den Abschluss der Arbeit bildet! Suspendisse vel felis.

Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.



Inhaltsverzeichnis

1 Einleitung	1
1.1 Motivation	1
1.2 Aufgabenstellung	1
1.3 Zusammenfassung des Projektergebnisse	1
1.4 Projektvorgehensmodell	1
2 Technologien	3
2.1 Technologieevaluierung	3
2.2 Technologie entscheidung	7
3 Systemarchitektur	9
3.1 Flutter	9
3.2 Firebase	9
3.3 Algolia Search	13
4 Umsetzung	14
4.1 Continuous Integration/Delivery	14
4.2 Mobile Anwendung	14
4.3 UI/UX Design	21
4.4 Backend	31
5 Evaluating getroffener Entscheidungen	40
6 Zusammenfassung	41
Literaturverzeichnis	VI
Abbildungsverzeichnis	VII
Tabellenverzeichnis	VIII
	III

1 Einleitung

1.1 Motivation

1.2 Aufgabenstellung

1.3 Zusammenfassung des Projektergebnisse

1.4 Projektvorgehensmodell

1.4.1 Projektorganisation

Discord

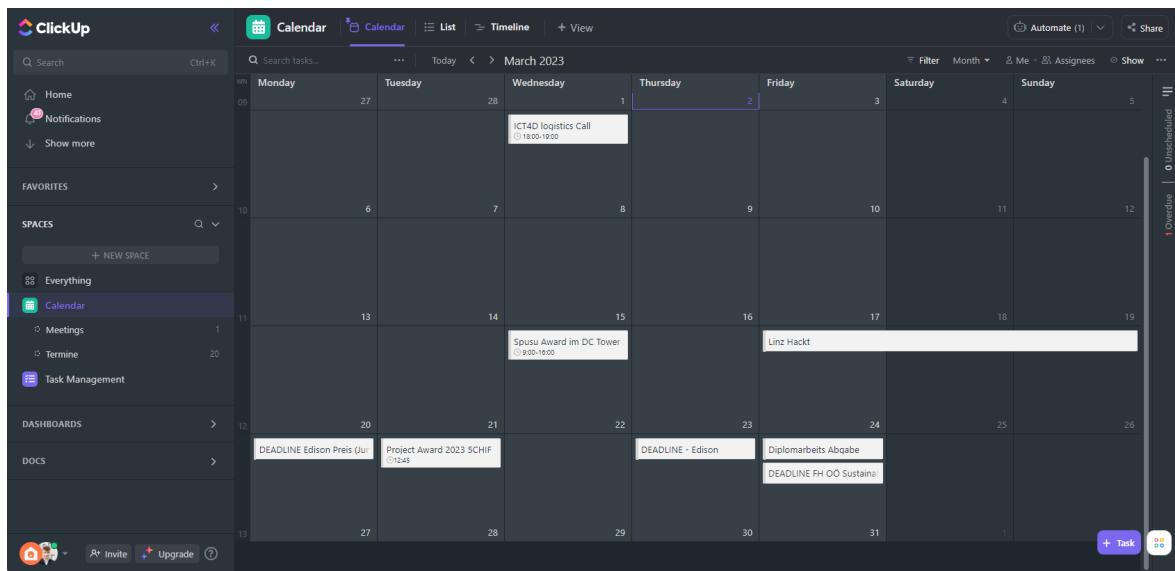
Unsere primäre Kommunikationsplattform zwischen unserem Team und unseren Diplomarbeit-Betreuern ist Discord.

Da wir in den Sommerferien 2022 mit unserer Arbeit begonnen haben und zu diesem Zeitpunkt Corona-Regelungen galten, mussten wir eine Meeting-Plattform finden, die sowohl Video- als auch Text-Chat ermöglicht. Aufgrund unserer Vertrautheit mit Discord und der Tatsache, dass es kostenlos ist, fiel die Entscheidung leicht.

Während unserer Diplomarbeit haben wir über 40 Text-Kanäle erstellt, um alle relevanten Informationen, Notizen und Dokumente zu speichern. Die Meeting-Notizen wurden jedoch auf ClickUp gespeichert.

ClickUp

Um unser Projekt effizient zu organisieren, nutzen wir nicht nur eine WhatsApp-Gruppe, sondern auch das Projektmanagement-Tool ClickUp. Die Entscheidung für ClickUp fiel uns leicht, da ich persönlich bereits viele Projektmanagement-Programme ausprobiert habe und ClickUp bei meinen letzten Projekten am besten abgeschnitten hat. Zudem ist es sehr intuitiv und einfach zu erlernen, wie auch meine beiden Teamkollegen bestätigen können.



In unserer Diplomarbeit haben wir viele Mentoring-Sessions, Meetings mit unserem Diplomarbeitbetreuer und weitere Termine wie Wettbewerbe. Ohne eine geeignete Methode zur Verwaltung all dieser Termine kann es schnell unübersichtlich werden. Daher haben wir das Kalender-Feature von ClickUp genutzt, um alle unsere Einträge abzubilden. Auf der Abbildung sieht man den ClickUp Kalender-View, wie wir ihn benutzt haben. Ein großer Vorteil dabei war, dass der Kalender automatisch mit unseren persönlichen Kalendern synchronisiert wurde, wie beispielsweise meinem Google Kalender. Dadurch haben wir nie einen Termin verpasst, was insbesondere für mich als Projektleiter sehr hilfreich war, da ich nicht jeden daran erinnern musste, alle Termine in seinem Kalender einzutragen.

Ein weiterer wichtiger Punkt ist das Task-Management in unserem Projekt. Da ich schnell viele Aufgaben koordinieren musste, war es sinnvoll, alle Aufgaben in einer simplen To-Do-Liste abzubilden. Vor den Semesterferien haben wir beispielsweise eine eigene Abteilung für Ferien-Tasks erstellt, um einen Überblick über alle Aufgaben während der Semesterferien zu haben. Obwohl wir die Tasks sehr nicht aktiv genutzt haben, war es sehr praktisch, um den Fortschritt zu überwachen und sicherzustellen, dass alles rechtzeitig erledigt wurde.

Zusammenfassend war ClickUp eine große Bereicherung für unser Projektmanagement, da es uns viel Zeit bei der Verwaltung von Terminen und Aufgaben gespart hat. Auch die Benutzung als neuer User war sehr einfach, sodass wir uns nicht lange einlesen mussten. Besonders faszinierend ist, dass ClickUp sehr intuitiv ist und viele nützliche Features bietet.

2 Technologien

2.1 Technologieevaluierung

Ein zentraler Aspekt jedes Projektbeginns ist die Evaluierung der Technologie. Wir haben uns bewusst viel Zeit genommen, um eine schnelle und unkomplizierte Entwicklung zu ermöglichen und später mögliche Bedauern zu vermeiden. Die Anforderungen für das Projekt umfassen:

- Entwicklung von Apps für Android und iOS
- Möglichkeit zur zukünftigen Veröffentlichung der Anwendung als Webapp
- Unterstützung von Datenbankabfragen auf Grundlage von Geo-Koordinaten
- Integrierte Authentifizierungs- und Autorisierungsschicht
- Realtime-Datenbank, um Echtzeit-Chatrooms für Konversationen zu führen.

2.1.1 Frontend

Im Bereich der Frontend-App-Entwicklung stehen verschiedene Möglichkeiten zur Verfügung, um eine App zu programmieren:

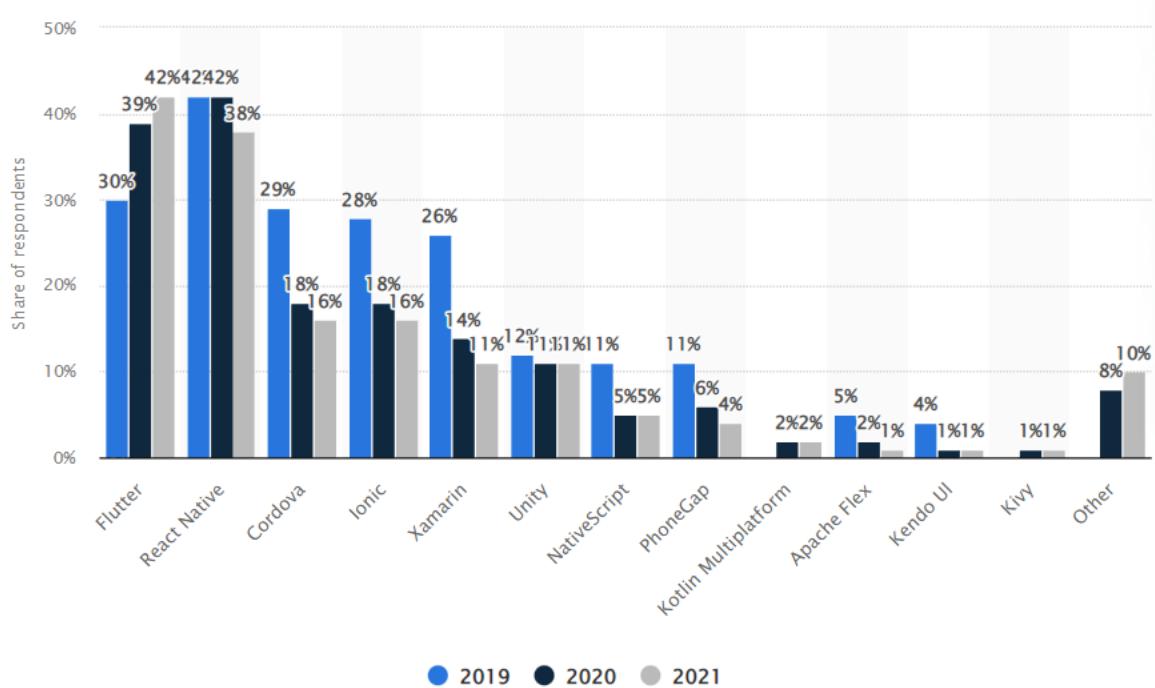
- Native Entwicklung für jede Plattform (iOS, Android, Web)
- Nutzung eines Frameworks für plattformübergreifende Entwicklung von Apps

Native Entwicklung für jede Plattform (iOS, Android, Web): Die native Entwicklung ist eine Möglichkeit, eine App speziell für eine bestimmte Plattform wie iOS, Android oder das Web zu entwickeln. Hierbei wird die App in der nativen Sprache der Plattform programmiert, d.h. in Swift für iOS, Java oder Kotlin für Android und HTML, CSS und JavaScript für das Web. Der Vorteil der nativen Entwicklung besteht darin, dass die App die native Funktionalität des Geräts vollständig nutzen und eine optimale Performance bieten kann. Entwickler können auf alle nativen APIs zugreifen und die Benutzeroberfläche der App kann besser an das jeweilige Betriebssystem angepasst werden. Allerdings erfordert die native Entwicklung spezialisierte Kenntnisse in den

verschiedenen Plattformen und Programmiersprachen. Dies bedeutet, dass mehrere Entwickler benötigt werden und die Entwicklung teurer und zeitaufwändiger sein kann.

Ein Framework für plattformübergreifende Entwicklung ermöglicht die Erstellung einer einzigen Codebasis, die auf verschiedenen Plattformen wie iOS, Android und dem Web ausgeführt werden kann. Ein solches Framework nutzt häufig eine gemeinsame Programmiersprache wie JavaScript und bietet eine Vielzahl von Bibliotheken und Tools, um die Entwicklung zu erleichtern. Das Framework ermöglicht es Entwicklern, schnell und effizient Apps zu erstellen, ohne sich mit den Unterschieden in den Plattformen und Programmiersprachen auseinandersetzen zu müssen. Ein weiterer Vorteil von Frameworks für plattformübergreifende Entwicklung besteht darin, dass sie eine schnellere Markteinführung ermöglichen und den Entwicklungsaufwand reduzieren können. Allerdings kann die Verwendung eines Frameworks die Performance beeinträchtigen, da es schwierig sein kann, die App für jede Plattform zu optimieren.

Wir haben schnell beschlossen, dass wir diese Sache nicht mit nativer Programmierung angehen werden. Keiner in unserem Team ist mit einer der Plattformen vertraut, was zu einem doppelten Arbeitsaufwand führen würde. Für ein kleines Team von nur drei Personen wäre dies zeitlich einfach zu aufwendig.



<https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/>

Um eine effektive und plattformübergreifende Entwicklung zu gewährleisten, haben wir uns dazu entschieden, ein Cross-Platform-Framework zu verwenden. Wir haben die Statistik darüber betrachtet, welche Plattformen am meisten genutzt werden und dabei festgestellt, dass Flutter mit 42 Prozent das meistgenutzte Framework ist und die Tendenz steigend ist. Auf Platz 2 liegt React Native, das fast genauso beliebt ist.

	Flutter	React Native	Cordova	Ionic	Xamarin
Version 3.2022	2.10	0.68	11.0.0	6.0.10	5.10
iOS	iOS 8+	iOS 9+	iOS 8+	iOS 8+	iOS 8+
Android	Android 4.1	Android 4.1+	Android 4.1+	Android 4.1+	Android 4.1+
Web	✓	✓	✓	✓	✓
Language Support	Dart	JavaScript	HTML, CSS, JavaScript	HTML, CSS, JavaScript	C#
UI Widgets	Material Design, Cupertino (iOS)	Jede	-	Material Design	Xamarin Forms, Native UI
Hot Reload	Yes	Yes	No	Yes	Yes
License	BSD 3-Clause	MIT	Apache 2.0	MIT	MIT
Package Manager	Pub	NPM, Yarn	NPM	NPM	NuGet
Packages verfügbar 2.2023	32914	1216	6880	400+	1588
Github Stars 2.2023	150k	108k	5k	48k	15k

Quellen: [<https://reactnative.directory/>] (<https://reactnative.directory/>)

[<https://pub.dev/>] (<https://pub.dev/>) [<https://www.npmjs.com/search?q=keywords:ecosystem:cordova>]
[<https://github.com/xamarin>] (<https://github.com/xamarin>) [<https://github.com/apache/cordova-android>] (<https://github.com/apache/cordova-android>)

[<https://ionic.io/docs/supported-plugins>] (<https://ionic.io/docs/supported-plugins>)

[<https://market.ionicframework.com/plugins>] (<https://market.ionicframework.com/plugins>)

In der Vergleichstabelle haben wir die wichtigsten Informationen zu den verschiedenen Frameworks zusammengefasst. Dabei haben wir festgestellt, dass alle Plattformen unsere Anforderungen erfüllen. Xamarin hat für uns als Team, das seit Jahren C# programmiert, ein gewisses Interesse geweckt. Allerdings hat Dart eine ähnliche Syntax wie C#, weshalb es für uns als Team auch eine geeignete Option darstellt.

Flutter hat als UI-Widget-Library sowohl Material Design 2 und 3 als auch eine iOS Library, die das native iOS-Design 1:1 abbildet. Dadurch kann der Nutzer den Unterschied zwischen nativen und Flutter-Designs kaum erkennen. Xamarin hat ebenfalls eine UI-Library namens Xamarin.Forms, die auf allen Plattformen funktioniert und native UI-Libraries für jede Plattform enthält.

React Native ist zwar in JavaScript geschrieben, aber man kann keine React-Komponenten importieren und verwenden. Wenn man jedoch bereits Erfahrung mit React.js hat, kann man sich als Entwickler leichter in React Native einarbeiten. Die letzten beiden Plattformen Cordova und Ionic basieren auf HTML, CSS und JavaScript, was den Vorteil bietet, dass man auf jede JavaScript-Library zugreifen kann.

Ein wichtiger Faktor bei der Wahl des Frameworks ist auch der Hot Reload. Cordova bietet diese Funktion nicht, alle anderen Plattformen hingegen schon, was für uns ein dealbreaker ist.

Die Package-Manager wurden ebenfalls verglichen und der von Flutter hat uns am meisten überzeugt, da jedes Package ein Beispiel hat und man anhand von Pub Points sehen kann, wie gut das Package in verschiedenen Kriterien abschneidet. Zudem kann man sehen, welche Plattformen das Package unterstützt.

Obwohl npm der meistgenutzte Package-Manager ist, ist er nicht spezifisch auf ein bestimmtes Framework ausgerichtet und es fehlen einige Funktionen, die Pub hat. NuGet kennen wir bereits von der CSharp-Entwicklung und unserer Meinung nach wirkt er sehr altmodisch und fehlt ebenfalls einige Funktionen, die Pub hat.

Ein weiterer wichtiger Faktor bei der Wahl des Frameworks sind die verfügbaren Packages. Flutter bietet hier eine große Auswahl, wobei zu beachten ist, dass bei Frameworks, die auf JavaScript basieren, auf alle JavaScript-Libraries zugegriffen werden kann. Insgesamt sind die meisten Flutter-Packages auf mobile Apps ausgerichtet.

Die Community wurde anhand der Anzahl der Github-Stars gemessen, wobei Flutter mit gut einem Drittel vor React Native liegt. Obwohl Flutter noch nicht so lange auf dem Markt ist, spricht dies dafür, dass Entwickler mit Flutter sehr zufrieden sind.

Ein wichtiger Aspekt bei der Verwendung von Cross-Platform-Frameworks ist die Performance, da diese aufgrund ihrer Nicht-Nativität variieren kann. In einem Forschungspapier, in dem alle oben genannten Plattformen mit Ausnahme von Cordova verglichen wurden, wurde in Punkt 2.6 ein aussagekräftiger Vergleich durchgeführt. Dabei wurde festgestellt, dass Flutter hinsichtlich der Performance auf Platz 1 liegt,

gefolgt von React Native auf Platz 2. Im Gegensatz dazu schneidet Xamarin bei diesem Vergleich nicht so gut ab. Die letzte Position nimmt Iconic ein, was aufgrund der Tatsache, dass es lediglich eine HTML-, CSS- und JavaScript-Seite in einem Webview anzeigt, nachvollziehbar ist. Im Vergleich zu den anderen Plattformen kann es aufgrund des Fehlens eigener Renderer nicht mithalten.

Quellen: <http://uu.diva-portal.org/smash/get/diva2:1626535/FULLTEXT01.pdf>

2.1.2 Backend

Da ich über umfangreiches Wissen in verschiedenen Backend-Technologien verfüge, fiel uns die Entscheidung für unser Projekt etwas leichter. Wir konzentrierten uns bei der Auswahl auf "Backend-as-a-Service"-Plattformen, da sie bereits viele Funktionen implementiert haben und wir somit nicht alles neu programmieren mussten.

In Betracht kamen für unser Projekt drei Technologien: Supabase, Firebase und Appwrite. Alle drei bieten SDKs für Flutter und React Native. Darüber hinaus verfügen sie alle über eine Datenbank, die Geoqueries für das Hauptfeature des Beitragradius unterstützt, sowie Storage für Beitragsfotos und viele mehr

Supabase hot zum Zeitpunkt der Evaluierung im Februar 2022 jedoch noch keine Cloud-Functions wie Firebase und Appwrite, was es schwierig machte, eine robuste Business-Logik umzusetzen. Obwohl Supabase am 1. April 2022 experimentell Edge Functions eingeführt hat, ist dies für eine Produktions-App nicht empfehlenswert.

Daher blieben Appwrite und Firebase als die beiden besten Optionen übrig. Obwohl Appwrite nicht mit den Features von Firebase mithalten konnte, ist Appwrite 100 Prozent Open Source und Firebase Closed Source, was für Appwrite spricht. Da wir uns bei unserem Cross-Platform-Framework noch nicht sicher waren, war es uns nun leichter, uns für Firebase zu entscheiden, da es das beste Flutter-SDK hatte, was man anhand der verfügbaren Features erkennen konnte.

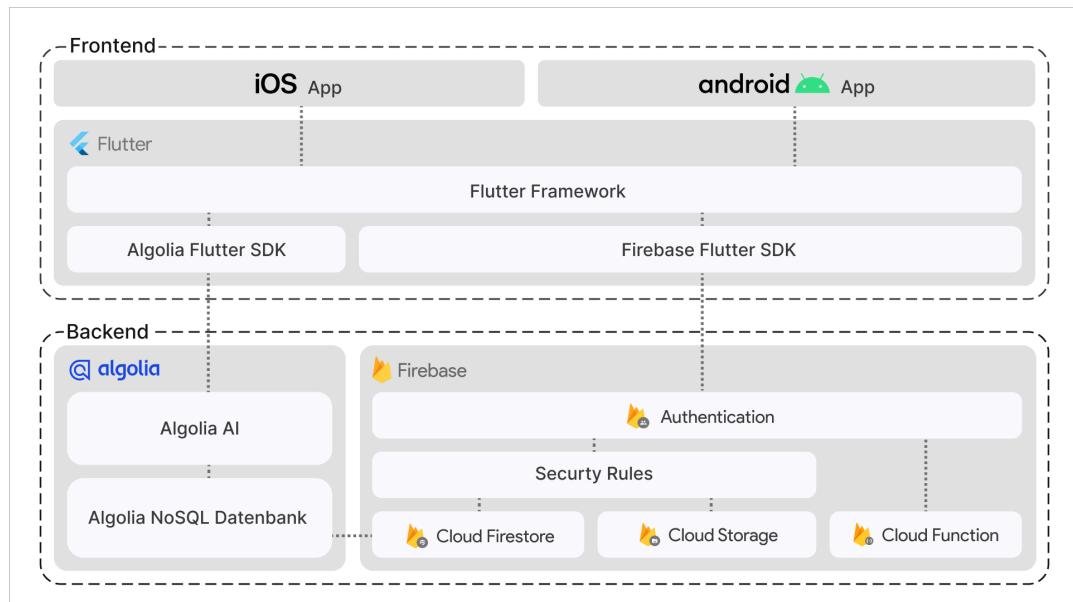
2.2 Technologie entscheidung

Die Kombination aus Flutter und Firebase hat uns am meisten überzeugt, da sie gut harmonieren. Flutter ist ein schnelles Framework mit einer breiten Palette an Erweiterungen für mobile Anwendungen und verfügt über eine starke Community. Der Trend geht in die Richtung, dass Flutter Marktführer wird, was uns ein gutes Gefühl

bezüglich unserer Entscheidung gibt. Zu Beginn gab es jedoch Zweifel bei der Arbeit mit Flutter, da es aufgrund der vielen Klammern schnell unübersichtlich wurde. Dank der super VS Code-Integration mit einem Reformat-Feature, das das Klammernproblem löst, konnten wir diese Herausforderung meistern. Auch nach einem Jahr Entwicklung mit Flutter sind wir immer noch beeindruckt davon, wie einfach es ist, Features zu implementieren und wie straight-forward die Entwicklung ist. Der Syntax von Dart, der Programmiersprache hinter Flutter, ist schnell erlernbar und ermöglicht es uns, schnell viele Programmierpatterns zu beherrschen.

Firebase hat sich ebenfalls als sehr gute Entscheidung herausgestellt, da die Entwicklung schnell und einfach ist. Ein großer Vorteil ist, dass wir uns keine Sorgen um das Skalieren machen mussten. Mit den Features ist Firebase nach wie vor die Nr. 1. Allerdings gibt es zwei große Nachteile, die unsere Backend-Entscheidung aktuell ändern würden. Die Firestore-Datenbank bietet keine Suchfunktion, weshalb wir Algolia genutzt haben. Einerseits bietet Algolia viele fortgeschrittene Suchfunktionen, die andere Datenbank-Suchmaschinen nicht haben. Allerdings ist es keine perfekte Lösung. In unserer Diplomarbeit ist auch das Thema Transparenz und Open Source von großer Bedeutung. Firebase schränkt uns hierbei ein, da es Closed Source ist. Daher würden wir nach aktuellem Standpunkt Supabase nutzen, da es mittlerweile auch Cloud Functions bietet und Open Source ist. Supabase verwendet zudem PostgreSQL, das eine Suchfunktion schon integriert hat.

3 Systemarchitektur



3.1 Flutter

Hier kommt die Beschreibung der Technologie und deren Vorteile/Nachteile, auf Basis von wissenschaftlichen Studien und Erfahrungen aus der Praxis.

Quellen:

<https://docs.flutter.dev/resources/architectural-overview>

<https://flutter.dev/docs/resources/technical-overview>, <https://pub.dev/packages/flutter>

3.2 Firebase

Firebase[1] ist eine von Google entwickelte Plattform für die Entwicklung von Web- und mobilen Anwendungen. Die Plattform bietet eine Vielzahl von Tools und Diensten, die es Entwicklern ermöglichen, schnell und einfach skalierbare Anwendungen zu erstellen.

Die Systemarchitektur von Firebase basiert auf der Cloud-Computing-Technologie, bei der die Anwendungslogik und Daten in der Cloud gehostet werden. Dies bedeutet, dass Entwickler keine physischen Server oder Infrastrukturen verwalten müssen, um ihre Anwendungen zu betreiben. Stattdessen können sie sich auf die Entwicklung der Anwendungslogik konzentrieren und die Firebase-Plattform übernimmt den Rest.

Firebase bietet auch eine Echtzeit-Datenbank, die es Entwicklern ermöglicht, Daten in Echtzeit zwischen ihren Anwendungen zu synchronisieren. Darüber hinaus bietet Firebase eine Vielzahl von Tools und Diensten, darunter Authentifizierung, Benachrichtigungen, Hosting, Speicherung und vieles mehr.

Dank dieser Systemarchitektur und der bereitgestellten Tools und Dienste können Entwickler mit Firebase schnell und einfach skalierbare Anwendungen erstellen und betreiben.

3.2.1 Firebase Authentication

Beschreibung der Firebase Authentication Technologie, inklusive Sicherheitsregeln und Best Practices.

Quellen:

<https://firebase.google.com/docs/auth>

<https://medium.com/flutter-community/firebase-authentication-in-flutter-752d14209a8a>

Security Rules

In unserer App ist die Sicherheit ein sehr wichtiger Faktor, um die Vertraulichkeit und Integrität der Daten und der Nutzer zu gewährleisten. Firebase bietet für seine Cloud-basierten Datenbank- und Speicherlösungen - Cloud Firestore und Cloud Storage - die sogenannten Security Rules[2][3][4], um den Zugriff auf die Daten und Ressourcen zu kontrollieren. Diese Regeln definieren, wer auf welche Art und Weise auf welche Daten zugreifen darf.

Die Security Rules für Cloud Firestore und Cloud Storage sind in einer eigenen Sprache geschrieben und werden serverseitig auf Firebase-Servern ausgeführt. Die Syntax basiert auf einer ähnlichen Struktur wie JSON und erlaubt komplexe Abfragen. Die Regeln können für eine bestimmte Sammlung oder einen bestimmten Pfad definiert werden und erlauben es, bestimmte Bedingungen für Lese- oder Schreibzugriffe zu definieren.

Listing 1: Security Rules Beispiel

```

1      match /posts/{postId} {
2          allow read: if request.auth.uid != null;
3          allow write: if request.auth.uid == resource.data.author;
4      }

```

Diese Regel definiert, dass jeder Nutzer Lesezugriff auf alle Posts hat, aber nur der Autor des Posts ihn auch ändern darf.

Insgesamt bieten die Security Rules für Cloud Firestore und Cloud Storage eine leistungsstarke und flexible Möglichkeit, die Zugriffsrechte auf die Daten und Ressourcen in einer Firebase-App zu steuern. Mit Hilfe der Security Rules können wir sicherstellen, dass die Daten ihrer Nutzer geschützt und nur von berechtigten Personen abgerufen oder geändert werden können.

3.2.2 Cloud Firestore

Vorstellung der Datenbanktechnologie Firestore

Quellen:

<https://firebase.google.com/docs/firestore>

<https://medium.com/flutter-community/firebase-cloud-firestore-in-flutter-26c6e8c6f90c>

Datenmodell

Hier kommt die Erklärung des Datenmodells in Firebase Firestore und dessen Auswirkungen auf die App-Architektur.

Quellen:

<https://firebase.google.com/docs/firestore/data-model>

<https://www.raywenderlich.com/6628345-cloud-firestore-for-flutter-getting-started>

Weitere wichtige Punkte:

- Präsentation des eigenen Datenmodells in der Arbeit.
- Diagramme werden verwendet, um das Modell zu präsentieren und Entscheidungen zu erläutern und zu begründen.
- Anforderungen der App-Architektur werden dabei berücksichtigt werden.
- Performance, Skalierbarkeit und Strukturierung können thematisiert werden.

- Zur Veranschaulichung unserer eigenen Datenmodell-Entwicklung kann auf ein Beispiel-Datenmodell-Diagramm auf der offiziellen Firebase-Website verwiesen werden, welches uns als Orientierungshilfe diente.

Beispiel-Datenmodell-Diagramm Quelle:

https://firebase.google.com/docs/firestore/data-model#structure_your_data

3.2.3 Cloud Storage

Beschreibung der Cloud Storage Technologie in Firebase und deren Einsatz in der App.

Quellen:

<https://firebase.google.com/docs/storage>

<https://medium.com/flutter-community/firebase-cloud-storage-in-flutter-flutter-an-firebase-tutorial-c5de7835c6cd>

3.2.4 Firebase Cloud Functions

Firebase Cloud Functions[5] ermöglichen es uns, unsere Business-Logik wie beispielsweise den Registrierungsprozess und viele andere Logikvorgänge einfach und effektiv in der Cloud abzubilden. Wir haben uns entschieden, die Funktionen in TypeScript zu schreiben, da es uns viele Vorteile bietet, wie zum Beispiel eine statische Typisierung, verbesserte IDE-Unterstützung und eine erhöhte Code-Lesbarkeit.

Der Registrierungsprozess ist ein gutes Beispiel dafür, wie wir Firebase Cloud Functions einsetzen können. Anstatt eine monolithische Anwendung zu erstellen, die alles in einem einzigen Server handhabt, können wir die Logik in kleinere Funktionen aufteilen, die jeweils eine spezifische Aufgabe erfüllen.

Firebase Cloud Functions bietet viele Vorteile, wie zum Beispiel die Möglichkeit, die Funktionen einfach zu skalieren und automatisch zu verteilen, um hohe Lasten zu bewältigen. Außerdem können wir die Funktionen einfach testen und debuggen, indem wir lokale Emulatoren verwenden, bevor wir sie in der Cloud bereitstellen.

Insgesamt sind Firebase Cloud Functions eine großartige Möglichkeit, um unsere Business-Logik in der Cloud abzubilden und unsere Anwendung zu skalieren und

zu verbessern. Durch die Verwendung von TypeScript können wir sicherstellen, dass unser Code sauber und robust bleibt und leicht gewartet werden kann.

3.3 Algolia Search

Vorstellung der Algolia Search Technologie und deren Integration in die App-Architektur.

Algolia AI

Quellen:

<https://www.algolia.com/doc/>

<https://www.algolia.com/doc/guides/sending-and-managing-data/send-and-update-your-data/tutorials/firebase-algolia/>

Firebase Cloud Function

Beschreibung der Firebase Cloud Functions und deren Rolle in der Algolia Integration.

Quellen:

<https://firebase.google.com/docs/functions>

<https://www.algolia.com/doc/guides/sending-and-managing-data/send-and-update-your-data/tutorials/firebase-algolia/>

4 Umsetzung

4.1 Continuous Integration/Delivery

4.1.1 GitHub Actions

allg. actions warum kosten

Build IOS

Build Android

4.1.2 Fastline

Build Number increment

4.1.3 Firebase App Distribution

4.2 Mobile Anwendung

4.2.1 Dateistuktur

In Flutter gibt es keine fixe Dateistuktur für eine App, man kann seine Struktur also selbst überlegen und gestalten. Im Folgenden beschreibe ich, wie wir unsere Dateistuktur für eine Flutter-App aufgebaut haben.

In Flutter gibt es keine feste Dateistuktur, stattdessen kann man die Struktur der Dateien und Ordner selbst bestimmen. Für unser Flutter-Projekt haben wir uns für eine Struktur entschieden, die sich an bewährten Praktiken orientiert.

Unsere Dateistuktur sieht wie folgt aus:

- **logic** - Hier befindet sich die Geschäftslogik der App, einschließlich der Firestore-Cloud-Funktionen und Repositories, die API-Aufrufe ausführen.
- **pages** - Hier werden Widgets entworfen, die jeweils eine Seite der App darstellen.
- **routes** - Hier werden die Routen definiert, die tiefere Links ermöglichen.

- **shared** - Hier werden UI-Widgets wie Buttons oder andere Widgets gespeichert, die oft wiederverwendet werden.
- **views** - Hier befinden sich Ansichten, die von mehreren Seiten der App verwendet werden können.

Im Nachhinein hätten wir die Dateistruktur anders gestaltet, z.B. hätten wir das UI als eigenes Package definiert und die pages und views besser unterteilt.

4.2.2 State Management

In Flutter gibt es verschiedene Möglichkeiten[6][7], um mit dem State Management umzugehen. State Management bezieht sich auf die Art und Weise, wie Daten innerhalb einer App verwaltet werden. In jeder App gibt es bestimmte Daten, die von verschiedenen Komponenten und Widgets verwendet werden und sich im Laufe der Zeit ändern können. State Management bezieht sich auf die Methoden, die verwendet werden, um diese Daten innerhalb der App zu verwalten und zu aktualisieren.

GetX

GetX verwendet ein reaktives Ansatz zur Verwaltung des Zustands, was bedeutet, dass Änderungen im Zustand automatisch die UI aktualisieren, ohne dass der Entwickler manuell Code schreiben muss, um diese Aktualisierungen durchzuführen. Dies spart viel Entwicklungszeit und macht es einfach, auf Benutzerinteraktionen zu reagieren.

Mit GetX können wir auch eine einheitliche Datenquelle haben, auf die alle Komponenten zugreifen können, was die Wartung und Erweiterung der Anwendung erleichtert. Darüber hinaus bietet GetX auch eine einfache Möglichkeit, Abhängigkeiten zu verwalten und Zustandsinformationen zwischen Bildschirmen zu teilen.

Insgesamt hat uns die Verwendung von GetX im Flutter-Framework sehr geholfen, eine effektive und skalierbare Anwendung zu erstellen, die auf die Bedürfnisse unserer Benutzer abgestimmt ist. getx controller service etc...

4.2.3 Authentifizierung

Anmelde Flow

Diagramm erklärung screenshots

Regestrierungs Flow

Diagramm erklärung screenshots

Firebase Authentifizierung

allg.

4.2.4 Feed

Die Feed-Page ist die erste und wichtigste Anzeige auf der App. Hier werden alle aktuellen Beiträge und Updates von der Nachbarschaft angezeigt.

4.2.5 Beiträge

Beiträge sind das Hauptkommunikationsmittel auf der App. Jedem Beitrag muss ein Titel, eine Beschreibung und eine Reichweite, unter der, der Beitrag sichtbar ist, angegeben werden. Weiters ist es möglich einem Beitrag ein Bild und Tags anzuhängen.

Kategorien

Um Beiträge besser zuordnen zu können, muss der User den Beitrag vor dem Veröffentlichen in eine bestimmte Kategorie einteilen. Diese Kategorien ermöglichen es Usern, die Art Ihrer Anfrage ihm vorhinein besser zu spezifizieren und die Suche nach Beiträgen einer bestimmten Art zu vereinfachen. Bestimmte Kategorien werden weiters in Unterkategorien aufgeteilt, da diese ein zu weit gefächertes Genre an Anfragen umfassen.

Es existieren folgende Kategorien bzw. Unterkategorien:

- Mitteilung
 - Frage
 - Appell
 - Warnung
 - Empfehlung
 - Gefunden
- Suche
 - Hilfe
 - Verloren

- Ausleihen
- Event

Mitteilung: Die Kategorie der Mitteilung dient dazu die Nachbarn über ein bestimmtes Ereignis oder Meldung zu informieren oder zu befragen.

Suche: Die Kategorie der Suche dient dazu mit den Nachbarn im Falle einer Hilfesuche oder eines verloren gegangenen Objekts in Kontakt zu treten.

Ausleihen: Die Kategorie des Ausleihens dient dazu die Nachbarn nach der Erlaubnis, sich ein bestimmtes Werkzeug oder Objekt ausborgen zu dürfen, zu bitten.

Event: Die Kategorie des Events dient dazu die Nachbarn auf eine bestimmte Veranstaltung aufmerksam zu machen.

Tags

Als Tag wird ein Schlüsselwort beschrieben, was man an ein Informationsgut anhängen kann, um es besser beschreiben zu können und/oder besser auffindbar zu machen. In der App werden Tags als eine Erweiterung der Kategorien verwendet, um es Usern zu ermöglichen Ihren Beitrag einem selbstdefinierten Typ zuzuordnen.

Info

Jeder Beitrag hat eine eigene Sektion, wo wichtige Entscheidungsinformationen angegeben werden, wie der Stadtteil und die ungefähre Entfernung zum gegebenen Nachbarn und das Erstelldatum des Beitrags.

Kommentare

Die Kommentarfunktion ermöglicht es den Usern unter einem Beitrag Ihre Meinung, Feedback oder sonstiges zu hinterlassen.

Beitrag oder Kommentar Melden

Um auf unangebrachte Beiträge oder Kommentare schnell reagieren zu können, gibt es die Möglichkeit Beiträge oder Kommentare zu melden. Diese Meldungen werden auf Firestore gespeichert und können dann im Einzelnen überprüft werden. Fürs Melden muss ein Grund ausgewählt und eine genauere Beschreibung angegeben werden. Gründe fürs Melden eines Beitrags oder Kommentars:

- Unangebrachter Inhalt
- Belästigung
- Betrug
- Spam
- Sonstiges

4.2.6 Filter

Der Filter bietet die Option die Beiträge nach bestimmten Kriterien zu filtern und die Suche nach bestimmten Beiträge zu vereinfachen. Der Filter unterteilt sich in einen Menüfilter und Hauptfilter. Die Ansicht vom Menüfilter befindet sich direkt über den Beiträgen und ermöglicht eine schnelle Filterung nach einzig allein den Hauptkategorien. Die Ansicht vom Hauptfilter taucht erst nach dem Antippen vom Filtersymbol auf und beinhaltet eine größere Auswahl an Filteroptionen. Darunter zählt neben dem Filtern nach Hauptkategorien auch die zusätzliche Möglichkeit genauer nach Unterkategorien zu suchen. Außerdem besteht auch die Option die Beiträge nach Datum oder Likes zu sortieren oder die Beiträge aufsteigend oder absteigend zu ordnen. Die wichtigste Filterkomponente ist der Range-Slider, womit die Beiträge nach der Reichweite gefiltert werden können, da die Entfernung zum Nachbarn eine der wichtigsten Entscheidungsfaktoren zum Antworten auf einem Beitrag ist.

4.2.7 Suche

4.2.8 Typesense

4.2.9 Algolia

diagram

Firestore Sync

Algolia SDK

4.2.10 Chat

package genommen warum

Flyer Package

4.2.11 Profil

Die Profilanzeige ist die öffentliche Informationsstelle über den User. In dieser Anzeige werden als erster Eindruck der Name und das Profilbild vom User angezeigt. Genauere Informationen über den User können im Bereich Nutzerinfo gefunden werden. Darunter zählen:

- Geburtstag
- Beruf
- Bio

Außerdem beinhaltet die Anzeige eine eigene Beitragssicht, wo alle Beiträge vom jeweiligen User eingesehen werden können.

Profil Melden

Falls das Profil vom User unangebrachten Inhalt aufweist, besteht die Möglichkeit das Profil zu melden.

4.2.12 Benachrichtigungen

Benachrichtigungen dienen dazu die Nachbarn über mögliche Hilfsbereitstellungen zu informieren, bevor der Kontakt überhaupt entsteht. Die Benachrichtigung zeigt den Nachbarn an, der in Kontakt treten möchte, und möglicherweise den Beitrag auf dem geantwortet wurde. Außerdem ist die Benachrichtigung mit einem „Annehmen“- und „Ablehnen“- Button ausgestattet, womit das Hilfsangebot angenommen oder abgelehnt werden kann.

4.2.13 Einstellungen

Die Einstellungssicht der Anwendung bietet eine Reihe an nützlichen Funktionen, die dem User mehr Kontrolle über sein Konto geben. Dazu gehört zu einem die Option, die Sprache der App umzustellen, um dem User zu ermöglichen, die Anwendung in der bevorzugten Sprache zu nutzen. Zum derzeitigen Stand kann die App sich in zwei Sprachen übersetzen lassen: Deutsch und Englisch. Darüber hinaus können Benutzer auch entscheiden, ob sie Benachrichtigungen erhalten möchten oder nicht, und diese

Einstellungen jederzeit ein- oder ausschalten. Diese Einstellungen bieten den Benutzern eine höhere Privatsphäre und Personalisierungsmöglichkeiten, um die Anwendung besser an ihre individuellen Bedürfnisse anzupassen. Zu den weiteren Einstellungen gehört das Umändern der E-Mail oder des Passworts, um die Sicherheit des Kontos zu gewährleisten und unbefugten Zugriff zu verhindern. Die letzte Funktion in der Einstellungssicht ist das Löschen des eigenen Kontos, womit alle Daten des Users gelöscht werden. Vor dem endgültigen Löschen des Kontos wird der User allerdings aufgefordert, seine Entscheidung zu bestätigen.

4.2.14 Feedback

Im Zuge unserer Testphase suchen wir nach einer effizienten Methode, um Feedback zu sammeln. Um ein fundiertes Verständnis einer Fehlermeldung zu erlangen, erweist sich die Bereitstellung eines Screenshots als besonders nützlich. Daher haben wir uns nach Packages umgesehen, welche die Möglichkeit bieten, Screenshots zu erstellen und ergänzende Informationen als Kontext hinzuzufügen. Wir haben uns für das Package <https://pub.dev/packages/feedback> entschieden. Dieses Package gestattet dem Nutzer, einen Screenshot anzufertigen und Annotationen hinzuzufügen, bevor ein entsprechender Kontext in Textform eingefügt werden kann. Eine weitere praktische Funktion besteht darin, dass der Nutzer bei Bedarf noch einmal navigieren und den Screenshot wiederholen kann. Zur Initiierung dieses Feedback-Views haben wir das Package <https://pub.dev/packages/shake> genutzt, welches bei einer entsprechenden Bewegung des Mobilgeräts aktiviert wird. Wir haben uns bei dieser Funktion an anderen Anwendungen orientiert, die ähnliche Mechanismen für die Feedback-Erhebung nutzen.

Das von uns gewählte Vorgehen zur Analyse des Feedbacks erfolgt über die Trello-API, welche es uns ermöglicht, ein entsprechendes Dashboard zur Erfassung und Bearbeitung der Meldungen zu erstellen. Die erfassten Informationen, wie beispielsweise der Screenshot, der Text, das Datum und relevante Systeminformationen, dienen uns als Grundlage für eine weitere Auswertung und Fehlerbehebung. Trello erleichtert uns die Verwaltung des Feedbacks auf kostenlose Weise und stellt somit ein adäquates Instrument zur Unterstützung des Entwicklungsprozesses dar.

4.3 UI/UX Design

Das UI/UX-Design einer App ist von entscheidender Bedeutung für den Erfolg der App und die Zufriedenheit der Benutzer. Insbesondere bei einer Social-Media-Nachbarschafts-App ist ein gut durchdachtes UI/UX-Design unerlässlich, um eine positive Benutzererfahrung zu gewährleisten.

Ein gutes UI-Design ist wichtig, um sicherzustellen, dass die Benutzer die App einfach und intuitiv bedienen können. Es sollte eine klare Struktur und Navigation haben, damit die Benutzer schnell zu den gewünschten Funktionen gelangen können. Wenn die Benutzer die App als kompliziert oder verwirrend empfinden, werden sie möglicherweise frustriert und geben die Nutzung der App auf.

Auch ein gutes UX-Design ist wichtig, um sicherzustellen, dass die Benutzer mit der App zufrieden sind. Es sollte eine ansprechende und ansprechende Benutzeroberfläche bieten, die dem Benutzer ein angenehmes Nutzungserlebnis vermittelt. Wenn die Benutzer die App als langweilig oder uninteressant empfinden, werden sie möglicherweise nicht wiederkommen.

Darüber hinaus sollte das UI/UX-Design einer Social-Media-Nachbarschafts-App bestimmte Funktionen und Merkmale berücksichtigen, die für eine erfolgreiche Community-Plattform erforderlich sind. Beispielsweise sollte es einfach sein, Beiträge zu erstellen und zu teilen, auf Kommentare zu antworten und Nachrichten an andere Benutzer zu senden. Es sollte auch Möglichkeiten geben, um Benutzerprofile zu erstellen und zu verwalten sowie umfassende Datenschutz- und Sicherheitsfunktionen zu bieten.

Insgesamt ist das UI/UX-Design einer Social-Media-Nachbarschafts-App von entscheidender Bedeutung für den Erfolg der App und die Zufriedenheit der Benutzer. Es ist wichtig, dass das Design auf die Bedürfnisse und Anforderungen der Benutzer zugeschnitten ist und ein einfaches, intuitives und ansprechendes Nutzungserlebnis bietet.

4.3.1 Inspiration

Während des Designprozesses für die App habe ich Recherchen im Bereich App-Design durchgeführt. Mein Ziel war es, die App so intuitiv wie möglich zu gestalten, um eine benutzerfreundliche Erfahrung zu gewährleisten. Dazu habe ich mich an bekannten Social-Media-Apps orientiert, wie zum Beispiel Twitter, Instagram und TikTok, die

bereits auf dem Markt sehr erfolgreich sind und von vielen Menschen vertraut genutzt werden.

Darüber hinaus hat auch die erfolgreichste Nachbarschafts-App in Deutschland, "Nebenan", viel zur Grundlage der App beigetragen. Allerdings habe ich festgestellt, dass ihre App sehr kompliziert aufgebaut und unübersichtlich ist, was für uns eine Chance darstellte, es besser zu machen.

Zusätzlich haben wir uns von Websites wie Dribble und Mobbin inspirieren lassen, die uns geholfen haben, ein einfaches und schlichtes Design für die App zu entwickeln. Insgesamt war die Recherche und Inspiration für das Design der App ein wichtiger Schritt, um sicherzustellen, dass unsere Benutzer eine ansprechende und intuitive Erfahrung haben.

4.3.2 Prototyping

Prototyping ist ein wichtiger Schritt bei der Entwicklung von mobilen Apps. Es ermöglicht Entwicklern, Designern und anderen Stakeholdern, eine frühzeitige Vorstellung davon zu bekommen, wie die App funktionieren wird und wie sie aussehen wird. Durch die Erstellung eines Prototyps können auch Fehler im Design und in der Funktionalität identifiziert werden, bevor die App in die eigentliche Entwicklung geht.

Es gibt verschiedene Arten von Prototypen, darunter Low-Fidelity- und High-Fidelity-Prototypen. Low-Fidelity-Prototypen sind einfache Skizzen oder Wireframes, die nur die grundlegenden Funktionen und das Layout der App abbilden. High-Fidelity-Prototypen sind detaillierter und können interaktive Funktionen und Designelemente enthalten.

Es gibt viele Tools und Plattformen, die zur Erstellung von Prototypen verwendet werden können. Ein Beispiel ist Adobe XD, das es Entwicklern und Designern ermöglicht, schnelle und einfache Prototypen zu erstellen. Andere Tools wie Sketch, Figma oder InVision sind ebenfalls beliebt.

Der Vorteil von Prototyping ist, dass es Entwicklern und Designern ermöglicht, frühzeitig Feedback von Benutzern zu erhalten und Änderungen vorzunehmen, bevor die App tatsächlich entwickelt wird. Außerdem können Entwickler und Designer Zeit und Ressourcen sparen, indem sie sich auf die richtigen Funktionen und das richtige Design konzentrieren und unnötige Funktionen und Designs vermeiden.

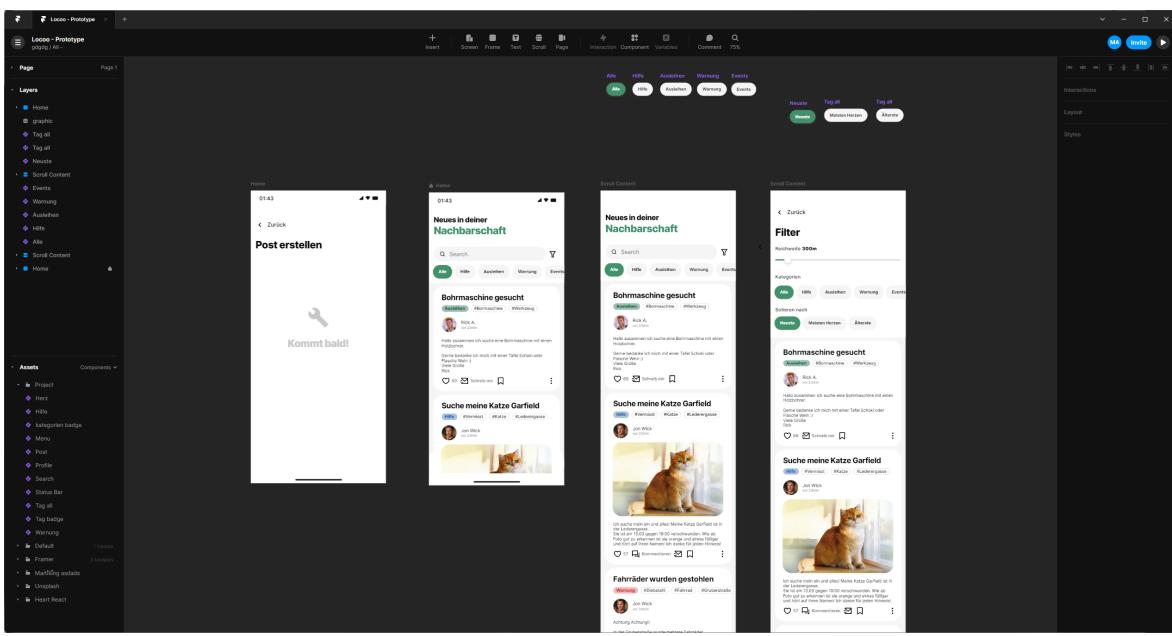
Im Fall von Flutter als Entwicklungsplattform für die App ist das Prototyping ebenfalls wichtig. Flutter bietet verschiedene Widgets und Design-Tools, mit denen Entwickler schnell und einfach Designelemente erstellen und implementieren können. Durch ein gut durchdachtes Prototyping kann Zeit und Aufwand beim Entwickeln gespart werden, da Design-Entscheidungen bereits getroffen wurden und Entwickler sich auf die Umsetzung konzentrieren können.

Insgesamt ist Prototyping ein unverzichtbarer Schritt bei der Entwicklung von mobilen Apps. Es ermöglicht Entwicklern und Designern, frühzeitig Feedback von Benutzern zu erhalten und Änderungen vorzunehmen, bevor die App tatsächlich entwickelt wird. Durch die Verwendung von Tools wie Adobe XD, Sketch oder Figma können Prototypen schnell und einfach erstellt werden, was Zeit und Ressourcen spart. Prototyping ist auch für die Verwendung von Flutter als Entwicklungsplattform wichtig, da es Entwicklern ermöglicht, Zeit und Aufwand zu sparen und sich auf die Umsetzung zu konzentrieren.

Framer

Framer ist eine Software, die es Benutzern ermöglicht, schnell und einfach ansprechende Prototypen von mobilen Anwendungen und Websites zu erstellen. Das Tool wurde ursprünglich als Prototyping-Tool für Designer entwickelt, um Designs schnell zu testen und zu verfeinern, bevor sie in die Entwicklung übergehen.

Erfolgreiche Apps wie Spotify haben Framer im Rahmen ihres Prototyping-Prozesses verwendet, um schnell und effizient funktionierende App-Designs zu erstellen. Framer ist eine schnelle und effiziente Möglichkeit, um Ideen in die Tat umzusetzen, ohne sich durch langwierige Entwicklungsprozesse zu quälen.



Framer wurde während des Hackerthon Linz hackt verwendet, um den ersten Prototypen zu erstellen. Die Wahl von Framer war aufgrund seiner Geschwindigkeit und Effizienz in der Erstellung von funktionsfähigen App-Designs und der Erfahrung, die der Benutzer bereits mit dem Tool hatte, getroffen worden.

Seit der Erstellung des ersten Prototyps hat sich das Geschäftsmodell von Framer jedoch geändert. Es ist jetzt ein Website-Baukasten, der es Benutzern ermöglicht, einfach und schnell ansprechende Websites zu erstellen, ohne Kenntnisse in der Webentwicklung zu benötigen. Obwohl es nun als Website-Baukasten fungiert, behält Framer immer noch einige seiner Kernfunktionen als Prototyping-Tool bei, was es zu einer guten Option für Designer und Entwickler macht, die schnell Prototypen erstellen möchten.

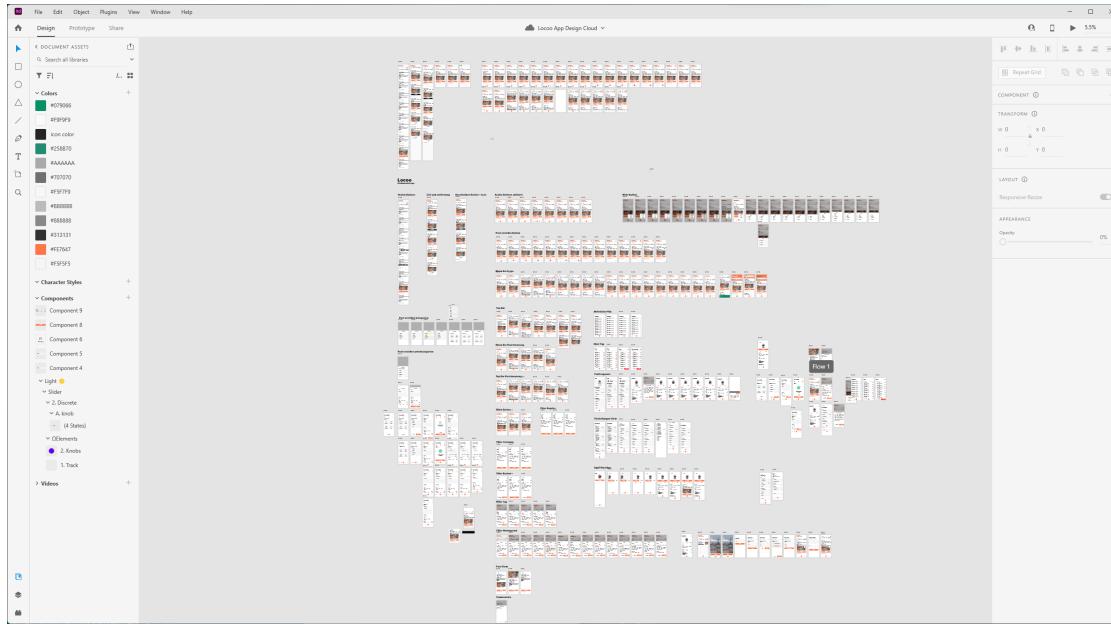
Insgesamt hat Framer gezeigt, dass es ein schnelles und effektives Tool ist, um Designideen in die Tat umzusetzen. Obwohl es nun als Website-Baukasten fungiert, ist es immer noch eine nützliche Option für Designer und Entwickler, die schnell und einfach Prototypen erstellen möchten.

Adobe XD

Als ich zum ersten Mal mit Framer arbeitete, war ich begeistert von den vielen Features und Möglichkeiten, die es bietet. Aber im Laufe der Zeit stellte ich fest, dass es für mein Projekt zu aufwändig war und ich nach einer einfacheren Lösung suchte. So bin ich auf Adobe XD umgestiegen, ein Tool, mit dem ich schon jahrelange Erfahrung hatte.

Obwohl Adobe XD nicht so viele Features wie Framer bietet, ist es aufgrund seiner Einfachheit und Benutzerfreundlichkeit ein ideales Tool, um schnell einen Prototypen

zu erstellen. Ich hatte alle Hauptscreens fertig gestaltet, bevor wir mit der Entwicklung mit Flutter begannen. Die anderen wichtigen Screens habe ich dann später gestaltet, als ich mit Flutter vertrauter war und besser abschätzen konnte, wie aufwändig es in Flutter umzusetzen war.



Wie man auf dem obigen Bild sehen kann, ist unser Prototyp eine Sammlung verschiedener Designs und Evolutionen des Designs. Adobe XD war für unser Projekt perfekt, da ich der alleinige Designer war und somit keine Kollaborationsfunktion benötigte. Allerdings, da ich jetzt mit dem bekanntesten Designtool Figma vertraut bin und es billiger ist, wenn mehrere Personen an einem Design arbeiten und auch mehr Funktionen bietet, würde ich aktuell mit Figma weiterarbeiten.

Zusammenfassend lässt sich sagen, dass Adobe XD ein mächtiges Tool für UI/UX-Designer ist, das einfach zu bedienen und ideal für kleine bis mittelgroße Projekte ist. Es bietet zwar nicht so viele Funktionen wie andere Tools, ist aber für schnelle Prototypenerstellung und einfache Zusammenarbeit mit Entwicklern und Stakeholdern sehr gut geeignet.

4.3.3 Design

allg.

Design System

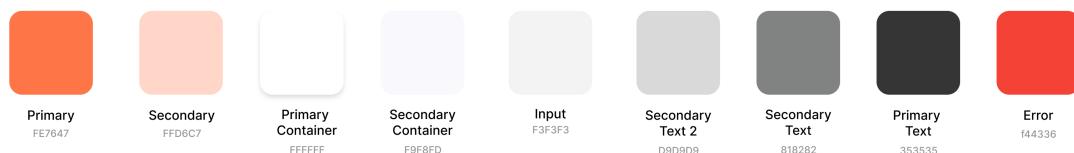
bild von allen componetns

Farben

Die Farbpalette einer Marke ist ein wichtiger Teil des UI-Designs und trägt dazu bei, ein einheitliches Erscheinungsbild zu schaffen. Bei der Gestaltung einer Nachbarschafts-App haben wir uns verschiedene Farbpaletten von anderen Apps angesehen und festgestellt, dass die meisten großen Apps grün gestaltet sind.

Obwohl Grün oft mit Natur und Gemeinschaft in Verbindung gebracht wird, wollten wir uns von diesen etablierten Konventionen abheben. Wir haben uns für Orange entschieden, da es auffällig und ungewöhnlich ist und somit das Potenzial hat, unsere App von anderen Nachbarschafts-Apps zu unterscheiden.

Darüber hinaus passt Orange gut zu unseren Werten, da es für Wärme, Freundschaft und Optimismus steht - Eigenschaften, die wir in unserer App fördern möchten. Wir hoffen, dass unsere Farbauswahl dazu beiträgt, dass unsere Nutzer sich in unserer App wohl und willkommen fühlen und die Farbe sich als wiedererkennbares Markenzeichen etabliert.



Bevor wir uns für unseren finalen Farbton entschieden haben, haben wir eine Vielzahl von verschiedenen orangen Farbtönen für unseren Prototypen ausprobiert und sie dann für ein paar Tage beobachtet, um ein Gefühl für die Farbe zu bekommen.

Schließlich haben wir uns für die endgültigen Farben entschieden, wie in der Abbildung dargestellt.

Unsere Hauptcontainerfarbe für die meisten Ansichten ist Weiß, ebenso wie die Hintergrundfarbe der Posts. Allerdings mussten wir eine Farbe finden, die gut als Hintergrund passt. Hier haben wir ein helles Grau verwendet.

Für unsere Textfarbpalette haben wir fast Schwarz verwendet. Dies wurde bewusst so gewählt, da Schwarz dadurch weicher wirkt. Um Untertitel weniger präsent zu gestalten, haben wir einen etwas helleren Grauton gewählt.

Icons

Remixicon



Material Icons



Eigen designtes Icon



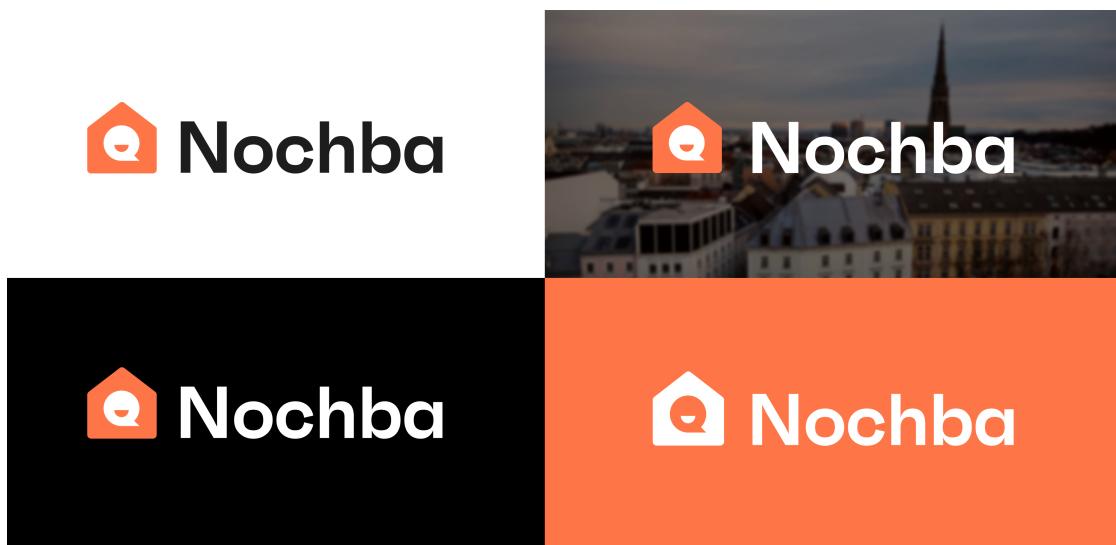
In der Abbildung oben sind alle verwendeten Icons in unserer App zu sehen. Es war uns ein wichtiges Anliegen, die App mit so vielen Icons wie möglich auszustatten, um die Bedienung intuitiver und benutzerfreundlicher zu gestalten. Nachdem wir uns mehrere Icon-Packs angesehen hatten, haben wir uns hauptsächlich für die frei verfügbaren Remixicons entschieden. Uns war wichtig, ein eher unbekanntes Icon-Pack zu verwenden, um unsere App von anderen abzuheben und ihr ein einzigartiges Aussehen zu geben. Besonders das verspielte, runde Design hat uns gut gefallen. Ein weiterer großer Vorteil von Remixicons ist, dass es ein Flutter-Paket gibt, was die Implementierung in Flutter vereinfacht hat.

Obwohl Remixicons über 2.271 Icons verfügt, haben uns bei einigen Icons die Material Icons von Google besser gefallen. Daher haben wir einige Icons von Google Material Icons verwendet, insbesondere die abgerundeten Icons, um das Design insgesamt nicht zu hart erscheinen zu lassen. Als ich kein passendes Icon finden konnte, um den Abstand zwischen zwei Nachbarn zu symbolisieren, habe ich mich selbst daran gesetzt, ein passendes Icon im Stil der anderen zu entwerfen.

Fonts

fotos von fonts welche fonts wie ist die typographie aufgebaut

Logo



Logo Historie

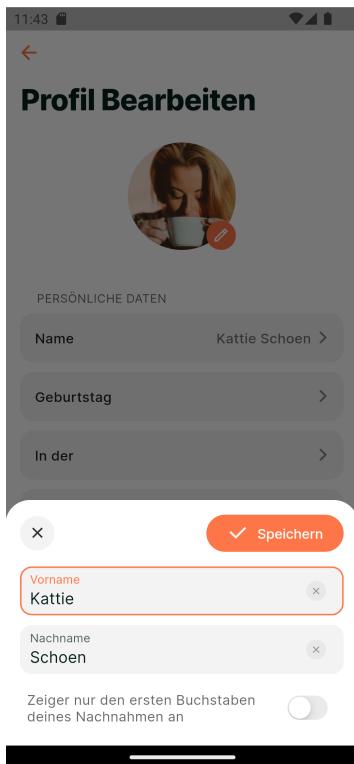
Im Zuge des Designprozesses haben wir hohe Ansprüche an das Logo unserer App gestellt, da das Logo unser Projekt repräsentiert. Es war uns wichtig, dass es einfach gehalten und leicht erkennbar ist. Wir haben uns Zeit gelassen, um den endgültigen Entwurf des Logos zu finalisieren, da uns die vorherigen Entwürfe nicht vollständig zufriedenstellten. Ich habe gezielt nach anderen Logos gesucht, die eine Verbindung zu Nachbarschaften oder Häusern herstellen und Ideen auf unserem Discord-Server gespeichert. Es sollte immer ein Haus erkennbar sein, da Häuser schnell mit Nachbarschaften in Verbindung gebracht werden.

In den früheren Versionen der App hatte ich Schwierigkeiten, ein ansprechendes App-Icon zu gestalten, da ich kein eigenständiges Icon hatte. Aus diesem Grund habe ich mich dazu entschlossen, den Schriftzug und das App-Icon (Logo) separat zu gestalten, wie es auch bei anderen Apps üblich ist. Das endgültige Logo habe ich dann erst Ende 2022 entworfen, da mir alle anderen Designs nicht gefielen. Es ist ein Haus mit einer sprechenden Sprechblase, die lächelt, geworden. Dies soll symbolisieren, dass man innerhalb des Hauses sprechen kann und das Lächeln soll verdeutlichen, dass man Freude mit seinen Nachbarn teilen kann. Generell hat ein Lächeln immer eine positive Wirkung auf den Betrachter und es verdeutlicht, dass es sich bei unserer App um eine Anwendung für den Austausch unter Menschen handelt.

Wir haben bewusst eine runde, verspieltere Schriftart gewählt, da sie besser mit der Farbmischung harmoniert als die vorherige Schriftart.

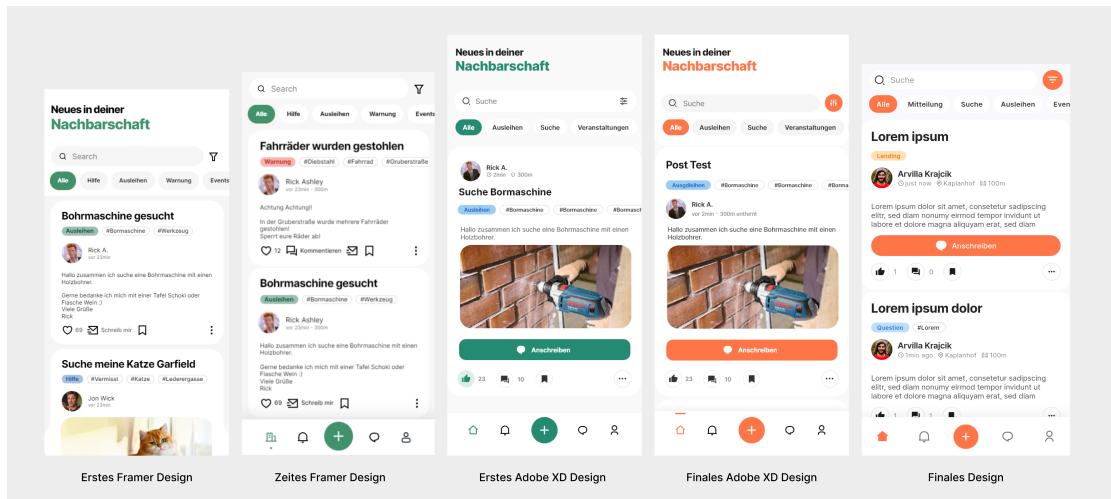
4.3.4 App Design

Thumb Zone Prinzip



Wir haben versucht, das Thumb Zone Prinzip in unserem Design-Muster zu implementieren, indem wir wichtige UI-Elemente immer im unteren Bereich der App platziert haben. Wenn man die Abbildung betrachtet, kann man deutlich erkennen, dass wir das Layout für die Namensbearbeitung extra unten platziert haben, um alle Klick-Elemente bequem mit dem Daumen bedienen zu können. Wir haben auch alle wichtigen Buttons wie "Speichern" in unserer primären Farbe gestaltet und rechts ausgerichtet, um einen leichteren Zugriff zu gewährleisten. Außerdem haben wir es bei unserer App so implementiert, dass man das Textfeld nicht durch Klicken ändern muss, sondern einfach die Enter-Taste auf der Tastatur drücken kann, um das Eintippen von Daten zu erleichtern. Unwichtige Buttons haben wir in grau gestaltet, wie z.B. den X-Button in der Abbildung. Ein kleines, aber nützliches Feature, das wir eingebaut haben, ist der Löschen-Button neben dem Textfeld, der es dem Benutzer ermöglicht, den gesamten Text bequem zu löschen. Leider ist uns gegen Ende die Zeit ausgegangen, um diese Prinzipien vollständig umzusetzen. In zukünftigen Versionen der App möchten wir sicherstellen, dass diese Prinzipien einheitlich angewendet werden.

Design Historie



eingehen mit fotos und beschreibung auf die haupt screens der app

4.3.5 Websites Design

The website landing page for Nochba features a large orange button labeled "Bald verfügbar". Below it, the text "Verbinde dich mit deiner Nochbaschaft" is displayed. A "Jetzt Mitmachen!" button is present. To the right, a mobile phone screen shows a post from "John Ullrich" about a missing power drill. The footer includes logos for DORFTV, Kronen Zeitung, OÖNachrichten, MeinBezirk.at, L_nz, and DORFT.

Da unser Projekt aufgrund unserer Teilnahme an Linz hackt in den Medien präsent war und wir keinen zentralen Anlaufpunkt für Informationen hatten, haben wir uns entschlossen, eine schlichte Landing Page mit grundlegenden Informationen zu gestalten. Das wichtigste Feature unserer Website ist die Möglichkeit für Nutzer, sich für die Testphase zu registrieren. Bei der Gestaltung der Landing Page ließ ich mich von den Vorlagen von Framer inspirieren und nutzte hauptsächlich fertige Abschnitte. Wir hielten uns an unsere Farbpalette und achteten darauf, dass die Webseite möglichst

einfach gestaltet ist. Ursprünglich hatten wir nicht geplant, eine Website für unser Projekt zu erstellen.

4.4 Backend

4.4.1 Firebase

Firebase ist ein BaaS, das uns Entwicklern eine erhebliche Menge an Stress abnimmt. Das Hosting von Datenbanken oder in unserem Fall Cloud-Funktionen funktioniert mit nur wenigen Klicks. Wir müssen uns keine Gedanken über Skalierung oder Ausfälle machen. Unser Firebase-Backend hosten wir auf der Google Cloud in Frankfurt (EUR3 Europe-West), um eine geringe Latenz zu gewährleisten. Zu Beginn haben wir den kostenlosen Plan "Spark" genutzt, was für uns ausreichend war, da wir noch nicht die Firebase-Cloud-Funktionen genutzt haben. Um Geld zu sparen, haben wir dann lange Zeit mit dem lokalen Firebase-Emulator an den Cloud-Funktionen gearbeitet. Im Januar 2023 sind wir dann auf den "Blaze"-Plan umgestiegen, um die Firebase-Cloud-Funktionen nutzen zu können. Bis zum Stand März 2022 haben wir nur wenige Euro für den Verbrauch gezahlt. Wir waren auch sehr beeindruckt von der exzellenten Dokumentation und den vielen Tutorials auf YouTube, die Firebase anbietet.

4.4.2 Firebase SDKs

Firebase SDK ist eine Sammlung von Software Development Kits, die von Firebase bereitgestellt werden, um Cloud-basierte Anwendungen zu erstellen. Firebase SDK ist für eine Vielzahl von Programmiersprachen verfügbar, darunter JavaScript, Swift, Kotlin, Java und unter anderem Dart von Flutter. Die Firebase SDK bietet eine breite Palette von Funktionen, mit denen leistungsstarke Anwendungen erstellt werden können, darunter Authentifizierung, Cloud Messaging, Cloud Firestore, Realtime Database, Cloud Functions. Mit diesen Funktionen können schnell und einfach Funktionen wie Benutzerverwaltung, Datenverwaltung, Messaging und Benachrichtigungen in Anwendungen integriert werden. Außerdem gibt es zu Firebase SDK eine umfangreiche Dokumentation und Support-Tools.

4.4.3 Firebase Authentication

Email und Passwort authentication

4.4.4 Cloud Firestore

NoSQL-Database, Dokument-basierte Speicherung, Subcollections

4.4.5 Cloud Storage

Speicherung von Profilbilder und Post-Bilder

4.4.6 Firebase Cloud Functions

Firebase Cloud Functions sind eine großartige Möglichkeit, um Business-Logik abzubilden. Es handelt sich um serverlose Funktionen, die in JavaScript oder TypeScript geschrieben werden und einzeln auf der Google Cloud gehostet werden. Je nach Nachfrage werden sie automatisch skaliert oder komplett abgeschaltet. Ein Nachteil ist, dass die Startup-Zeit länger sein kann als bei einem herkömmlichen Backend, das immer läuft. Allerdings kann man durch Cloud Functions viel Geld sparen, da nur die Prozessorlaufzeit bezahlt werden muss.

Firebase Cloud Functions ermöglichen es Entwicklern, auf verschiedene Ereignisse in Firebase-Produkten zu reagieren. Zum Beispiel, wenn sich Daten in der Firestore-Datenbank ändern oder ein neuer Nutzer in Firebase Authentication registriert wird. Wenn ein solches Ereignis eintritt, wird die entsprechende Cloud Function automatisch ausgeführt.

Wir haben alle Cloud Functions in TypeScript programmiert, da TypeScript Typsicherheit und erweiterte Fehlererkennung bietet.

Regestrierung mit einen Verifizierungscode

Die Cloud-Funktion "checkVerificationCode" wird ausgeführt, wenn ein Nutzer sich mit einem Verifizierungscode registrieren möchte. Wenn der Benutzer nicht authentifiziert ist, wird eine `HttpsError` ausgelöst. Dann wird der Verifizierungscode überprüft, um sicherzustellen, dass er die korrekte Formatierung hat. Wenn der Code ungültig ist, wird eine weitere `HttpsError` ausgelöst.

Als Nächstes wird der Verifizierungscode mit der Datenbank abgeglichen, um sicherzustellen, dass er aktiv und noch nicht zu oft verwendet wurde. Wenn der Code erfolgreich validiert wird, wird die Adresse des Benutzers abgerufen und deren Koordinaten

mithilfe von einer API call anOpenStreetMap mit der Funktion "getOSMCoordinatesFromAddress" ermittelt. Dann wird die Entfernung zwischen der Adresse des Benutzers und der Adresse, die dem Verifizierungscode zugeordnet ist, berechnet mit der Funktion "getDistanceFromLatLonInMeters". Wenn die Entfernung nicht innerhalb des zulässigen Bereichs liegt, wird eine weitere HttpsError ausgelöst.

Schließlich werden die Informationen des Benutzers und des Verifizierungscodes in der Datenbank aktualisiert, um anzusehen, dass der Benutzer erfolgreich verifiziert wurde. Die Cloud-Function gibt true zurück, um anzusehen, dass die Verifizierung erfolgreich war.

In jeder Phase der Funktion wird ein Logger verwendet, um Informationen über den Status der Funktion zu protokollieren.

Regestrierung mit Gerät Koordinaten

Die Cloud-Funktion "checkAddressWithDeviceLocation" erfordert eine authentifizierte Anfrage und erhält eine Adresse, eine Längen- und Breitengradkoordinate vom Gerät des Benutzers. Die Funktion prüft, ob alle erforderlichen Daten vorhanden sind und ruft dann die Funktion "getOSMCoordinatesFromAddress" auf, um die Koordinaten der angegebenen Adresse zu erhalten. Es wird auch die Funktion "getDistanceFromLatLonInMeters" aufgerufen, um die Entfernung zwischen der Adresse und den Koordinaten des Geräts des Benutzers zu berechnen.

Wenn die Entfernung größer ist als ein vordefinierter maximaler Abstand, wird eine Fehlermeldung ausgegeben und die Funktion gibt "false" zurück. Andernfalls speichert die Funktion die Koordinaten der Adresse und die Entfernung zwischen den Koordinaten des Geräts des Benutzers und der Adresse in der Firestore-Datenbank. Die Funktion ruft auch die Funktion "getOSMSuburbFromCoords" auf, um den Vorort der Adresse zu erhalten, und speichert diesen ebenfalls in der Firestore-Datenbank.

Die Funktion gibt "true" zurück, wenn die Verifizierung erfolgreich abgeschlossen ist.

In jeder Phase der Funktion wird ein Logger verwendet, um Informationen über den Status der Funktion zu protokollieren.

Verifizierungscode generieren

Die Cloud-Funktion "generateVerificationCode" definiert Konstanten für das Intervall zwischen der Generierung von Codes, die maximale Anzahl von Codes und die Reichweite in Metern.

Anschließend wird der letzte code des nutzers aus der Datenbank geholt, um zu überprüfen, ob seit dem letzten generierten Code ausreichend Zeit vergangen ist. Wenn ja, wird der zuletzt generierte Code zurückgegeben.

Wenn nicht, wird eine Schleife gestartet, um einen neuen Code zu generieren mit der Funktion "generateRandomVerificationCode". Der generierte Code wird dann mit Firestore abgeglichen, um sicherzustellen, dass er nicht bereits verwendet wurde.

Wenn der generierte Code eindeutig ist, wird überprüft ob der Benutzer verifiziert ist. Wenn dies nicht der Fall ist, wird ein Fehler zurückgegeben. Andernfalls wird der generierte Code in die Firestore-Datenbank eingefügt um den letzten generierten Code und das Datum der Generierung zu speichern.

Das Skript gibt dann den generierten Code zurück. In jeder Phase der Funktion wird ein Logger verwendet, um Informationen über den Status der Funktion zu protokollieren.

Koordinaten von einer Adresse bestimmen

Ein wichtiger Teil für die Verifizierung ist, dass wir die Adresse des Nutzers in Koordinaten umwandeln, damit wir im Backend damit arbeiten können. Zunächst haben wir die Google Geocode API genutzt, allerdings kostet diese Geld. Wir haben eine bessere Option gefunden, die besser zu unseren Projekt-Philosophie passt: nämlich Nominatim, eine Open-Source-Geocoding-API für OpenStreetMap-Daten. Nominatim bietet eine kostenlose API für genau unser Problem an.

Die Funktion **getOSMCoordinatesFromAddress** nutzt die Bibliothek **axios**, um eine Anfrage an die Nominatim-API zu senden. Die API wird genutzt, um Koordinaten für eine Adresse zu erhalten.

Die Funktion nimmt einen Parameter **address** vom Typ **string** entgegen, welcher die Adresse enthält, für die Koordinaten abgerufen werden sollen.

Mithilfe von **axios.get** wird eine HTTP GET-Anfrage an die Nominatim-API gesendet. Die Adresse wird als URL-Parameter im Format **q=Adresse** übergeben. Die API liefert

die Koordinaten als JSON-Objekt zurück, welches in der Variable **data** gespeichert wird.

Die Funktion überprüft dann, ob die Anfrage ein Ergebnis zurückgeliefert hat. Falls nicht, wird eine Fehlermeldung mit der Adresse ausgegeben.

Wenn ein Ergebnis vorhanden ist, wird das erste Ergebnis (in der Variable **firstResult**) verwendet, um eine neue Instanz der **GeoPoint**-Klasse zu erstellen. Diese Klasse ist Teil der Firebase-Admin-Bibliothek und ermöglicht die Speicherung von geografischen Koordinaten in einer Firestore-Datenbank.

Schließlich gibt die Funktion die neuen Koordinaten als **GeoPoint**-Objekt zurück, das die Breiten- und Längengrade in Zahlen enthält.

Entfernung von zwei Nutzern berechnen

Die Cloud-Funktion "getDistanceFromTwoUsers" berechnet die Entfernung zwischen zwei Benutzern. Die Funktion erwartet eine PostId und einen Authentifizierten Benutzer. Dann wird eine Überprüfung durchgeführt, ob der Post mit der angegebenen ID existiert und ob der Post eine gültige Reichweite hat. Es werden auch Überprüfungen durchgeführt, ob die Benutzerkoordinaten vorhanden sind, und ob die Entfernung zwischen beiden Benutzern innerhalb des Postbereichs liegt.

Die Funktion nutzt die importierten Funktionen `getDistanceFromLatLonInMeters` und `getNearestDistance`, um die Entfernung in Metern zu berechnen. Allerdings wird die berechnete Distanz grob gerundet, um die Privatsphäre der Nutzer zu wahren. Dabei werden die Längen- und Breitengradkoordinaten von zwei Benutzern miteinander verglichen, die aus der Firestore-Datenbank abgerufen werden. Wenn die Entfernung größer als die Reichweite des Posts ist, wird ein Fehler ausgegeben.

Schließlich gibt die Funktion die Entfernung zurück.

Entfernung von zwei geographischen Koordinaten berechnen

Listing 2: `getDistanceFromLatLonInMeters` Funktion

```

1     export function getDistanceFromLatLonInMeters(
2         lat1: number,
3         lon1: number,
4         lat2: number,
5         lon2: number
6     ) {
7         if (lat1 < -90 || lat1 > 90 || lon1 < -180 || lon1 > 180) {
8             throw new Error(
9                 "Invalid coordinate: lat1 must be between -90 and 90, lon1 must be
10                between -180 and 180"
11             )
12     }
13 }
```

```

10      );
11  }
12  if (lat2 < -90 || lat2 > 90 || lon2 < -180 || lon2 > 180) {
13    throw new Error(
14      "Invalid coordinate: lat2 must be between -90 and 90, lon2 must be
15      between -180 and 180"
16    );
17  const R = 6371; // Radius der Erde in km
18  const dLat = deg2rad(lat2 - lat1); // deg2rad unten
19  const dLon = deg2rad(lon2 - lon1);
20  const a =
21    Math.sin(dLat / 2) * Math.sin(dLat / 2) +
22    Math.cos(deg2rad(lat1)) *
23      Math.cos(deg2rad(lat2)) *
24      Math.sin(dLon / 2) *
25      Math.sin(dLon / 2);
26  const c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
27  const d = R * c * 1000; // Distanz in Metern
28
29  if (lat1 === lat2 && lon1 === lon2) return 0;
30  return d;
31}
32
33 function deg2rad(deg: number) {
34   return deg * (Math.PI / 180);
35}

```

Der vorliegende Code implementiert eine Funktion, die die Distanz in Metern zwischen zwei geographischen Koordinaten (Breitengrad und Längengrad) auf der Erdoberfläche berechnet. Die Funktion verwendet die Haversine-Formel, die auf der Kugelgeometrie basiert, um die kürzeste Entfernung zwischen zwei Punkten auf der Erdoberfläche zu berechnen.

Die Funktion "getDistanceFromLatLonInMeters" hat vier Parameter: "lat1" und "lon1" sind die Breiten- und Längengrade des ersten Punktes, während "lat2" und "lon2" die Breiten- und Längengrade des zweiten Punktes sind, zwischen denen die Distanz berechnet werden soll.

Zu Beginn des Codes werden die Eingabeparameter auf ihre Gültigkeit geprüft und eine Fehlermeldung wird ausgegeben, falls eine der Koordinaten außerhalb des Bereichs von -90 bis 90 für die Breite und -180 bis 180 für die Länge liegt.

Die Funktion berechnet dann die Differenzen der Breiten- und Längengrade sowie den Radius der Erde (R) in Kilometern. Die Differenzen werden dann in Radianen umgewandelt und die Haversine-Formel wird angewendet, um die Entfernung in Kilometern zu berechnen. Schließlich wird das Ergebnis in Meter umgewandelt und zurückgegeben.

Die Funktion "deg2rad" wird als Hilfsfunktion definiert, um Grad in Radianen umzurechnen.

Die Funktion gibt 0 zurück, falls die beiden Eingabeparameter denselben Wert haben, um zu vermeiden, dass eine sehr kleine Distanz als Ergebnis ausgegeben wird, wenn es sich um denselben Punkt handelt.

Quelle: <https://www.movable-type.co.uk/scripts/latlong.html>

Bestimmung der nächstgelegenen Entfernung

Um die Privatsphäre der Nachbarn zu gewährleisten, benötigen wir eine Funktion, die uns den nächstgelegenen Abstand zu den Nachbarn gibt, ohne den genauen Abstand preiszugeben. Die Funktion heißt "getNearestDistance" und bekommt einen Meterwert als Parameter.

Die Funktion erstellt ein Array mit den Optionen [100, 200, 500, 1000, 5000, 10000, 15000] und setzt die Variable "nearest" auf den ersten Wert im Array.

Dann wird eine Schleife ausgeführt, die durch jedes Element im Array "options" geht und prüft, welches Element am nächsten zum angegebenen Abstand "distance" liegt. Wenn ein Element näher ist als das bisher am nächsten liegende Element, wird "nearest" aktualisiert.

Schließlich wird überprüft, ob "nearest" größer oder gleich 1000 ist, und je nachdem wird der Abstand entweder in Kilometern oder Metern zurückgegeben. Wenn der Abstand größer oder gleich 1000 ist, wird die Einheit "km" hinzugefügt, ansonsten wird "m" hinzugefügt. Wir haben absichtlich keine Switches oder If-Bedingungen benutzt, da wir es jetzt mit der aktuellen Funktion schneller schaffen, die Abstände zu ändern. Wir evaluieren noch, wie das Array mit den Optionen aussieht.

Nachbarschaft von Nutzer bestimmen

Um unseren App-Nutzern ein besseres Verständnis für die Nachbarschaften zu geben, in denen ihre Nachbarn wohnen, zeigen wir bei jedem Benutzer die Nachbarschaft an, in der sie leben. Diese Information wird automatisch in der Datenbank gespeichert, wenn der Nutzer verifiziert wird. Um diese Funktion zu ermöglichen, verwenden wir die folgende Funktion, die die geografischen Koordinaten des Benutzers verwendet, um die entsprechende Nachbarschaft mithilfe der OpenStreetMap-API abzurufen:

Die Funktion heißt "getOSMSuburbFromCoords" und nimmt zwei Parameter entgegen: "lat" für die geografische Breite und "lon" für die geografische Länge des Benutzers. Diese

Funktion gibt eine Promise zurück, die eine Zeichenfolge (String) mit dem Namen der Nachbarschaft des Benutzers enthält. Die Funktion verwendet die "axios"-Bibliothek, um eine GET-Anfrage an die OpenStreetMap-API zu senden. Diese Anfrage enthält die geografischen Koordinaten des Benutzers und die gewünschte Zoomstufe (18), um die Nachbarschaft zu finden. Wenn die Anfrage erfolgreich ist, gibt die Funktion den Namen der Nachbarschaft zurück, der aus den Daten extrahiert wird, die von der API zurückgegeben werden. Wenn der Name der Nachbarschaft nicht verfügbar ist, gibt die Funktion den Namen der Stadt oder der Gemeinde zurück, in der sich der Benutzer befindet. Wenn auch diese Informationen nicht verfügbar sind, gibt die Funktion null zurück. Wenn bei der Anfrage ein Fehler auftritt, wird eine Fehlermeldung ausgelöst.

Verifizierungscode format überprüfen

Um die Laufzeit bei der Verifizierung von Cloud-Funktionen zu optimieren, ist es sinnvoll, am Anfang der Verifizierung eine Überprüfung durchzuführen, ob der Verifizierungscode das richtige Format hat. Dazu wird die Funktion `verifyVerificationCode` genutzt werden, welche einen `string` als Parameter erwartet und einen Boolean-Wert zurückgibt. In der Funktion wird ein regulärer Ausdruck (Regex) definiert, um sicherzustellen, dass der Verifizierungscode den Anforderungen entspricht. Der Regex lautet `/^[\a-zA-Z0-9]{10}$/`, was bedeutet, dass der Code aus genau 10 alphanumerischen Zeichen bestehen muss. Mit der Methode `test` wird der übergebene Code auf Übereinstimmung mit dem Regex geprüft und das Ergebnis als Boolean-Wert zurückgegeben.

Verifizierungscode generator

Die Funktion `generateRandomVerificationCode` erzeugt einen zufälligen Code mit einer Länge von 10 Zeichen, indem sie eine Kombination aus Groß- und Kleinbuchstaben des englischen Alphabets sowie Ziffern von 0 bis 9 verwendet. Dabei wird `Math.random()` zur Generierung einer Zufallszahl zwischen 0 und 1 genutzt und mit der Länge des Zeichenfolgen-Arrays multipliziert, um eine zufällige Position innerhalb des Arrays auszuwählen. Anschließend wird das ausgewählte Zeichen an das Ergebnis angehängt und dieser Schritt wird für jedes Zeichen wiederholt, bis eine Zeichenkette der Länge 10 generiert wurde.

Diese Funktion wird in der Cloud-Funktion `generateVerificationCode` verwendet, um einen zufälligen Code zu generieren. Es ist wichtig, dass der Code-Generator in der Lage ist, eine ausreichende Anzahl von Codes zu generieren, damit es keine Kollisionen gibt. Da jeder Code zufällig generiert wird und die Funktion eine zufällige Zeichenkette aus 62 möglichen Zeichen erzeugt, ist es äußerst unwahrscheinlich, dass der gleiche Code zweimal generiert wird. Die Anzahl der möglichen Kombinationen beträgt 62^{10} , was ungefähr 8.39×10^{17} Möglichkeiten entspricht. Daher ist die Wahrscheinlichkeit, dass zwei identische Codes generiert werden, vernachlässigbar.

4.4.7 Algolia Search

4.4.8 Algolia SDK

Typesense Search

4.4.9 Typesense SDK

5 Evaluating getroffener Entscheidungen

6 Zusammenfassung

Aufzählungen:

- Itemize Level 1
 - Itemize Level 2
 - Itemize Level 3 (vermeiden)
- 1. Enumerate Level 1
 - a. Enumerate Level 2
 - i. Enumerate Level 3 (vermeiden)

Desc Level 1

Desc Level 2 (vermeiden)

Desc Level 3 (vermeiden)

Literaturverzeichnis

- [1] Google. (2021) Firebase. Online verfügbar: <https://firebase.google.com/>
- [2] Firebase. (2023) Security Rules | Firebase. Online verfügbar: <https://firebase.google.com/docs/rules>
- [3] M. Hausleitner. (2023) Firestore Security Rules Nochba. Online verfügbar: <https://github.com/Martin-Hausleitner/Nochba/blob/main/firestore.rules>
- [4] ——. (2023) Storage Security Rules Nochba. Online verfügbar: <https://github.com/Martin-Hausleitner/Nochba/blob/main/storage.rules>
- [5] Google. (2023) Firebase Cloud Functions. Online verfügbar: <https://firebase.google.com/docs/functions>
- [6] F. Team. (2021) Flutter Interactive. Online verfügbar: <https://docs.flutter.dev/development/ui/interactive>
- [7] AppDevCommunity. Flutter State Management Approaches with Examples: setState, BLoC, Rx, Provider, Riverpod. Online verfügbar: <https://medium.com/app-dev-community/flutter-state-management-approaches-with-examples-setstate-bloc-rx-provider-riverpod-aad48f79a255>

Abbildungsverzeichnis

Tabellenverzeichnis

Quellcodeverzeichnis

1	Security Rules Beispiel	11
2	getDistanceFromLatLonInMeters Funktion	35

Anhang