FOM Hochschule für Ökonomie und Management

Hochschulzentrum München

Seminararbeit

Im Rahmen des Moduls

Anwendungsprojekt

Über das Thema

STH - Konzept und Implementierung der plattformunabhängigen SportTalentHub App für den Austausch zwischen Spieler und Sportmanager

von

Jonas Waigel, Fitim Makolli, Hasan Deveci und Vatsegkan Zournatsidis

Gutachter: Prof. Dr. Joachim Berlak
Matrikelnummer: 590367, 590827, 587470 und 585568
Abgabedatum: 27. Mai 2024

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

STH-App *Sport Talent Hub App*

IDE *Integrierte Entwicklungsumgebung*

iOS *Internetwork Operating System*

LaTeX *Lamport TeX*

MVP Minimal Viable Product

1 Vorwort

Diese Projektdokumentation behandelt das Konzept und die Implementierung der plattformunabhängigen SportTalentHub App (kurz STH) für den Austausch zwischen Spieler und Sportmanager. Dabei werden zunächst die wichtigsten Voraussetzungen wie die Teambildung, die Projektinitiierung und die Projektskizze beschrieben. Das Projektteam besteht aus vier Projektmitgliedern (Jonas Waigel, Fitim Makolli, Hasan Deveci und Vatsegkan Zournatsidis). Zusätzlich hat das Team Jonas Waigel als Projektleiter gewählt. Durch die einzigartigen Fähigkeiten, Erfahrungen, die gemeinsame Vision, eine Smartphone-App zu gestalten und die Leidenschaft für Sport gab dem Team den notwendigen Spirit und die Dynamik, um ein tolles Produkt zu erschaffen.
Viele Recherchen, Überlegungen und die Begeisterung für Sport hat das Projektteam auf die Idee gebracht, eine plattformunabhängige Smartphone-App zu implementieren, die ähnlich zur Social Media App Instagram den Austausch und das Anwerben zwischen Spielern und Sportmanagern ermöglicht.
Dabei war es dem Team stets wichtig, dass zunächst anstehende Aufgaben detailliert besprochen werden und ein gemeinsames Einverständnis über Aufgaben und Themen herrschte. Das Konzept und die Implementierung eines MVP der App soll im Zeitrahmen von 10.03.2024 bis 06.07.2024 entstehen und in dieser Dokumentation beschrieben werden. Das Projekt soll agil in einer auf das Projekt abgewandelten Form von Scrum durchgeführt werden.

2 Ziele

In diesem Kapitel werden die Muss-, Soll-, Kann- und Nicht-Ziele für das Projekt STH-App aufgezeigt.

Muss-Ziele

- Die STH-App muss als plattformunabhängige App mit einen Source-Code implementiert werden, die auf verschiedenen Betriebssystemen und Geräten implementiert (vorzugsweise macOS und Windows) und ausgeführt (vorzugsweise iOS und Android) werden kann.
- Die Chat-Funktion in der STH App muss es ermöglichen, eine sichere Kommunikation zwischen den App-Nutzern aufbauen zu können.
- Die Projektdokumentation muss bis zum 21.06.2024 vom Projektteam komplett fertiggestellt und durch die anderen Projektmitglieder Korrektur gelesen sein.
- Das Projektteam muss bis 15.05.2024 einen fertigen MVP incl. aller Screens in der App bereitstellen.
- Die STH-App für jeden Screen ein einheitliches Design aufweisen und bis zur Präsentation Design-Anpassungen und Verbesserungen durch das Projektteam durchgeführt zu haben.

Soll-Ziele

- Die STH-App soll bis zur Abgabe der Projektarbeit am 06.07.2024 auf verschiedenen Test-Geräten und Simulatoren / Emulatoren getestet werden und Fehler als neue Aufgaben eingestellt werden.
- Gefundene Fehler bei App-Tests sollen bis zum 06.07.2024 durch das Projektteam gelöst werden, um das Projekt gut und mit einer funktionierenden App präsentieren zu können.
- Für die Veröffentlichung sollen die datenschutzrechtliche Themen und Datensicherheit durch weitere Implementierungen oder das Ausweisen wichtiger Dokumente durch das Projektteam gewährleistet sein.

Kann-Ziele

- Nach Abschluss aller Implementierungsschritte und ausreichenden Tests kann die STH App im Apple Appstore oder Google Play Store veröffentlicht werden.

- Bis zu einer offiziellen App Veröffentlichung kann eine Login- und Registrierungsfunktion implementiert werden, die es ermöglicht, nutzerspezifische Inhalte zu speichern, abrufen und filtern zu können.
- Bei regelmäßigen Updates der App kann eine weitere Funktion für ein Hilfe- und Supportcenter angeboten werden.

Nicht-Ziele

- Auf dem Homescreen darf es keine Möglichkeit geben, Posts oder Feeds zu kommentieren.
- Stories und Reels ähnlich wie bei der Social Media App Instagram sollen nicht möglich sein.
- Die STH App soll nicht dazu dienen, Sportmanager bewerten zu können oder Transaktionen bzw. Verträge abzuschließen sondern als Kommunikationsplattform zwischen Spieler und Sportmanager.
- Es dürfen keine Kosten für Spieler oder Sportmanager entstehen, um die App mit allen Funktionen nutzen zu können.
- Es dürfen keine laufenden Kosten wie Lizenzkosten für das Projektteam entstehen.

3 Produktanforderungen und technische Architektur

In diesem Kapitel werfen wir einen umfassenden Blick auf die Produktanforderungen und die technische Architektur der STH-App. Die STH-App bietet weit mehr als nur das Teilen von Inhalten. Sie fungiert als Social-Media-Plattform, auf der Sportmanager und Sportler miteinander kommunizieren und Fähigkeiten sowie personenbezogene Profildaten teilen können. Daher ist es von entscheidender Bedeutung, dass die Produktanforderungen einzelner Funktionen und Features separat zur Implementierung bereitgestellt werden, um den unterschiedlichen Anforderungen und Erwartungen beider Gruppen gerecht zu werden. Insbesondere mit der großen Anzahl an Funktionen ermöglicht die STH-App den Nutzern, ihre sportliche Karriere zu teilen und sich dabei mit anderen Benutzern vernetzen können. Der Home-Screen der STH-App bietet einen umfassenden Überblick über die sportlichen Aktivitäten und Leistungen durch Posts der jeweiligen Benutzer. Auf diese Weise können Manager beispielsweise die Profile von Athleten verfolgen. Gleichzeitig ermöglicht er den Sportlern, ihre Fähigkeiten zu veröffentlichen und sich mit anderen Athleten und Sportmanagern anderer Sportvereine zu vernetzen. Der Chat-Screen hingegen bietet für Sportmanager und Athleten einen zentralen Bereich für den direkten Austausch von Textnachrichten über Chatkanäle. Dies ermöglicht eine reibungslose Kommunikation und Zusammenarbeit, indem wichtige Informationen schnell und effizient ausgetauscht werden können, um die gemeinsamen Vereinbarungen und Ziele zu erreichen. Auf dem Search-Screen können Benutzer nach spezifischen personenbezogenen Daten oder Hashtags suchen. So wird beispielsweise das Auffinden neuer Benutzer oder das Suchen nach individuellen Informationen erleichtert, die den eigenen Interessen entsprechen. Beim Profile-Screen können Benutzer die digitale Visitenkarte eines jeden Sportlers oder Managers einsehen. Diese enthält personenbezogene Daten sowie eine Beschreibung der Person, Bilder, Videos und Qualifikationen und ermöglicht es den Benutzern, sich ein umfassendes Bild von anderen Nutzern zu machen und potenzielle Verbindungen zu knüpfen oder Zusammenarbeit zu erleichtern. Die Leistung der STH-App spielt eine ebenso wichtige Rolle, um sicherzustellen, dass die Benutzer ein reibungsloses Erlebnis genießen können. Dabei erfordert es eine effiziente Programmierung und Optimierung sowohl auf dem Frontend als auch auf dem Backend der App. Der Loading-Screen unterstützt dabei, sämtliche Initialisierungsfeatures der STH-App aufzurufen und damit alle verbundenen Services zu starten und zu Initialisieren. Neben den Produktanforderungen ist auch die technische Architektur der App von entscheidender Bedeutung. Hierbei spielt die technische Architektur einer App eine entscheidende Rolle, da sie die Grundlage für Effizienz und Skalierbarkeit bildet. Ein sorgfältig ausgewähltes Set von Technologien und Plattformen ist daher unerlässlich, um sicherzustellen, dass die Anwendung nicht nur reibungslos funktioniert, sondern auch die sich ständig wandelnden Anforderungen der Benutzer erfüllen kann. Als Entwicklungsplattform dient Visual

Studio Code, eine leistungsstarke IDE, die speziell für die Entwicklung von Flutter-Apps optimiert ist. Visual Studio Code bietet eine Fülle von Funktionen und Erweiterungen, die den Entwicklungsprozess beschleunigen und vereinfachen. Die klare Definition der Produktanforderungen sowie die Festlegung der Arbeitspakete über GitHub-Commits, in denen die Aufgabenstellung detailliert beschrieben wird, bilden zusammen mit der soliden technischen Architektur das Fundament für eine effektive Zusammenarbeit in der Entwicklung und Dokumentation des Projekts. Flutter selbst fungiert als Cross-Platform-Framework für die Entwicklung des Frontends als auch der Schnittstelle zum Backend der STH-App. Die Verwendung von Flutter bietet eine Reihe von Vorteilen, darunter eine hohe Leistung, schnelle Entwicklung und einfache Wartung. Für das Backend der STH-App wird Firebase genutzt, eine Plattform von Google, die eine breite Palette von Diensten für die Entwicklung von Web- und Mobile-Apps bietet. Mit Funktionen wie Echtzeitdatenbanken, Authentifizierungsdiensten und Cloud-Speichern bietet Firebase eine robuste Lösung für die Anforderungen der STH-App.

4 Teilprojekte und Arbeitspakete

In diesem Kapitel liegt der Fokus auf den Arbeitspaketen, die für die Entwicklung der STH-App von entscheidender Bedeutung sind. Es wird zunächst erläutert, welche Aufgabenstellung jedes Arbeitspaket initialisiert, wobei sowohl die Verantwortlichen des Arbeitspakets als auch die damit verbundenen Ressourcen betrachtet werden.

Definition der Arbeitspakete:

Arbeitspaket 1 vom 08.03.2024 bis 11.03.2024

- Bezeichnung: Initial Setup Flutter
- Verantwortlichen: Das Team
- Ressourcen: VS Code, Flutter SDK, Smartphone Emulatoren
- Aufgabenstellung: Einrichtung der Entwicklungsumgebung für Flutter

Arbeitspaket 2 vom 08.03.2024 bis 11.03.2024

- Bezeichnung: Initial Setup LaTeX
- Verantwortlichen: Das Team
- Ressourcen: VS Code inklusive LaTeX-Plugins, Container-Plattform (Docker)
- Aufgabenstellung: Einrichtung der Entwicklungsumgebung für LaTeX

Arbeitspaket 3 vom 08.03.2024 bis 11.03.2024

- Bezeichnung: Initial Setup GitHub
- Verantwortlichen: Das Team
- Ressourcen: Git-Client, GitHub-Konto
- Aufgabenstellung: Erstellung des Repositories auf GitHub

Arbeitspaket 4 vom 13.03.2024 bis 14.03.2024

- Bezeichnung: Project Roadmap
- Verantwortlichen: Das Team
- Ressourcen: Brainstorming
- Aufgabenstellung: Definition der App Funktionalitäten

Arbeitspaket 5 vom 14.03.2024 bis 15.03.2024

- Bezeichnung: Project Structure
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Erstellung der ersten Projektstruktur in Visual Studio Code/ Flutter

Arbeitspaket 6 vom 16.03.2024 bis 23.03.2024

- Bezeichnung: Home Screen
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung des Home-Screens inklusive Menüleiste am unteren Bildschirmrand

Arbeitspaket 7 vom 18.03.2024 bis 01.04.2024

- Bezeichnung: Chat Screen
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung des Chat-Screens

Arbeitspaket 8 vom 21.03.2024 bis 04.04.2024

- Bezeichnung: Account Profile
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung des Account-Profile-Screens inklusive Avatar und personenbezogene Daten

Arbeitspaket 9 vom 23.03.2024 bis 06.04.2024

- Bezeichnung: Search Screen
- Verantwortlichen: Herr Vatsegkan Zournatsidis
- Ressourcen: ChatGPT 4.0
- Aufgabenstellung: Implementierung des Search-Screens

Arbeitspaket 10 vom 23.03.2024 bis 31.03.2024

- Bezeichnung: Global Menu Bar
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung einer globalen Menüleiste am unteren Bildschirmrand für alle Pages

Arbeitspaket 11 vom 24.03.2024 bis 07.04.2024

- Bezeichnung: Initial Setup Backend
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework, Backend-Plattform (Firebase)
- Aufgabenstellung: Backend-Initialisierung (Firebase)

Arbeitspaket 12 vom 24.03.2024 bis 31.03.2024

- Bezeichnung: Profile Edit Button
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung eines Bearbeitungsbuttons für den Account-Profile-Screen

Arbeitspaket 13 vom 24.03.2024 bis 31.03.2024

- Bezeichnung: Logo Design
- Verantwortlichen: Herr Vatsegkan Zournatsidis
- Ressourcen: ChatGPT 4.0
- Aufgabenstellung: Gestaltung eines Logos für die STH-APP

Arbeitspaket 14 vom 25.03.2024 bis 04.04.2024

- Bezeichnung: Loading Screen
- Verantwortlichen: Herr Vatsegkan Zournatsidis
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung einer Ladebildschirm-Funktion beim Start der STH-App

Arbeitspaket 15 vom 27.03.2024 bis 06.04.2024

- Bezeichnung: GetStream Chat
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung von GetStream-Chat

Arbeitspaket 16 vom 27.03.2024 bis 28.03.2024

- Bezeichnung: Chat App Bar
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Anpassung der App-Leiste auf dem Chat-Bildschirm

Arbeitspaket 17 vom 29.03.2024 bis 04.04.2024

- Bezeichnung: Logo Loading Screen
- Verantwortlichen: Herr Vatsegkan Zournatsidis
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung eines Ladebildschirms mit einem Logo

Arbeitspaket 18 vom 31.03.2024 bis 07.04.2024

- Bezeichnung: Backend Config
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework, Backend-Plattform (Firebase)
- Aufgabenstellung: Implementierung der Backend-Konfiguration (Firebase) und deren Methoden

Arbeitspaket 19 vom 01.04.2024 bis 07.04.2024

- Bezeichnung: Custom Page Routing
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung einer benutzerdefinierten Seitenroute beim Wechsel von den Pages

Arbeitspaket 20 vom 02.04.2024 bis 08.04.2024

- Bezeichnung: Chat UI Adjustments
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Anpassung des Chat-Screens und der benutzerdefinierte App-Leiste

Arbeitspaket 21 vom 02.04.2024 bis 08.04.2024

- Bezeichnung: Channel Adjustment
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Anpassung des Chanels im Chat-Screen

Arbeitspaket 22 vom 02.04.2024 bis 03.04.2024

- Bezeichnung: Latex Docs Update
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code
- Aufgabenstellung: Aufgabenstellung: Dokumentationsstruktur in LaTeX aktualisieren

Arbeitspaket 23 vom 03.04.2024 bis 10.04.2024

- Bezeichnung: Local Storage Setup
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung eines lokalen Speichers für den Account-Profile-Screen

Arbeitspaket 24 vom 03.04.2024 bis 10.04.2024

- Bezeichnung: Save/Load Functions
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung einer Funktion zum Speichern/Laden für den Account-Profile-Screen

Arbeitspaket 25 vom 03.04.2024 bis 10.04.2024

- Bezeichnung: RegEx Guidelines
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung von RegEx-Richtlinien in Bezug auf personenbezogene Daten

Arbeitspaket 26 vom 03.04.2024 bis 15.04.2024

- Bezeichnung: Hovering Function
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung einer Hovering-Funktion für personenbezogene Daten

Arbeitspaket 27 vom 03.04.2024 bis 15.04.2024

- Bezeichnung: Profile UI
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung des Profile-Screens mit Avatar, Bilder und Mediathek

Arbeitspaket 28 vom 10.04.2024 bis 18.04.2024

- Bezeichnung: Media Upload
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung von Funktionen zum Auswählen und Hochladen von Bilder und Videos

Arbeitspaket 29 vom 10.04.2024 bis 18.04.2024

- Bezeichnung: Profile Local Storage
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung eines lokalen Speichers für den Profile-Screen

Arbeitspaket 30 vom 10.04.2024 bis 18.04.2024

- Bezeichnung: Profile Save/Load
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung einer Funktion zum Speichern/ Laden für den Profile-Screen

Arbeitspaket 31 vom 10.04.2024 bis 11.04.2024

- Bezeichnung: App Bar Navigation
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Anpassung der App-Leiste für die Navigation und das Verhalten beim Seitenwechsel

Arbeitspaket 32 vom 13.04.2024 bis 19.04.2024

- Bezeichnung: Shared Preferences
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework, Backend-Plattform (Firebase)
- Aufgabenstellung: Verwendung von Shared Preferences zusammen mit Firebase-Testdaten

Arbeitspaket 33 vom 14.04.2024 bis 15.04.2024

- Bezeichnung: Page Linking
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Verlinkung der Pages von Profile-Screen und Account-Profile-Screen

Arbeitspaket 34 vom 16.04.2024 bis 17.04.2024

- Bezeichnung: Loading Screen Update
- Verantwortlichen: Herr Vatsegkan Zournatsidis
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Aktualisierung des Loadingscreens

Arbeitspaket 35 vom 16.04.2024 bis 25.04.2024

- Bezeichnung: Start Page Logo
- Verantwortlichen: Herr Vatsegkan Zournatsidis
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Hinzufügen vom Logo auf die Startseite

Arbeitspaket 36 vom 21.04.2024 bis 28.04.2024

- Bezeichnung: Media Button
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung eines Buttons für das Hochladen von Bilder und Videos

Arbeitspaket 37 vom 21.04.2024 bis 01.05.2024

- Bezeichnung: Avatar Transmission
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework, Backend-Plattform (Firebase)
- Aufgabenstellung: Implementierung von Funktionen zur Übertragung vom Avatar zum Backend

Arbeitspaket 38 vom 25.04.2024 bis 05.05.2024

- Bezeichnung: Media Transmission
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework, Backend-Plattform (Firebase)
- Aufgabenstellung: Implementierung von Funktionen zur Übertragung von Bilder und Videos an das Backend

Arbeitspaket 39 vom 27.04.2024 bis 28.04.2024

- Bezeichnung: Page Navigation
- Verantwortlichen: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Anpassung der Seiten-Navigation und Entfernung von Animationen beim Wechsel

Arbeitspaket 40 vom 28.04.2024 bis 29.04.2024

- Bezeichnung: Latex Docs Update
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code
- Aufgabenstellung: Dokumentationsstruktur in Latex aktualisiert

Arbeitspaket 41 vom 29.04.2024 bis 05.04.2024

- Bezeichnung: Chat Routing
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung der Seitenroute vom Chat zum Chanelscreen

Arbeitspaket 42 vom 29.04.2024 bis 05.04.2024

- Bezeichnung: Profile Navigation
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung der Navigationsroute vom Chat-Screen zum Profile-Screen

Arbeitspaket 43 vom 30.04.2024 bis 10.05.2024

- Bezeichnung: Stream Headers
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung des Stream-Channel-Headers

Arbeitspaket 44 vom 30.04.2024 bis 10.05.2024

- Bezeichnung: Username Update
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung der Aktualisierung des Benutzernamens auf dem Profile-Screen

Arbeitspaket 45 vom 30.04.2024 bis 07.05.2024

- Bezeichnung: Action Button
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung einer Aktionsschaltfläche mit Benutzerintegration

Arbeitspaket 46 vom 30.04.2024 bis 07.05.2024

- Bezeichnung: Test User Addition
- Verantwortlichen: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung neuer Chat-User als Testdaten

Arbeitspaket 47 vom 05.05.2024 bis 15.05.2024

- Bezeichnung: Hashtag Implementation
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung von Hashtags im Profile-Screen

Arbeitspaket 48 vom 06.05.2024 bis 15.05.2024

- Bezeichnung: Hashtag Storage
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung eines lokalen Speichers für die Hashtags

Arbeitspaket 49 vom 06.05.2024 bis 15.05.2024

- Bezeichnung: Hashtag Transmission
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung von Funktionen zur Übertragung von Hashtags an das Backend

Arbeitspaket 50 vom 07.05.2024 bis 20.05.2024

- Bezeichnung: Major Bug Fixes
- Verantwortlichen: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Fehlerbehebungen für den Profile-Screen und Account-Profile-Screen

5 Profile-Screen & Accountprofile-Screen

In diesem Kapitel wird sowohl auf die Umsetzung der Flutter-Komponenten für den Profile-Screen als auch den Accountprofile-Screen des Projekts eingegangen. Dabei liegt der Fokus auf die konzeptionelle Gestaltung und Implementierung beider App-Seiten, die es beispielsweise Benutzern ermöglichen, ihr Profilbild sowie multimediale Inhalte wie Bilder und Videos zu verwalten, aber auch personenbezogene Daten sicher abzuspeichern. Diese Funktionalitäten werden dabei unter Einsatz verschiedener Flutter-Pakete und Kernfunktionen der Dart-Programmiersprache umgesetzt.
Beginnend mit dem Profile-Screen, wird eine StatefulWidget-Klasse verwendet, die Zustandsänderungen verfolgt und das Benutzerinterface entsprechend aktualisiert. Verschiedene Funktionen werden definiert, um auf Benutzerinteraktionen zu reagieren, beispielsweise das Öffnen von Galerieinhalten, das Laden von Bildern und Videos aus dem lokalen Speicher, das Hochladen von Medieninhalten auf Firebase sowie das Löschen von Bildern und Videos aus der Ansicht. Darüber hinaus werden Methoden implementiert, um die Anwendungszustände zu verwalten, Benutzereingaben zu verarbeiten und Multimedia-Inhalte wie Bilder und Videos sicher zwischen dem lokalen Speicher, der in der Regel durch "shared preferences" verwaltet wird, und dem Backend in Firebase zu übertragen. Dadurch werden die Benutzerdaten effizient synchronisiert und sicher in der Cloud gespeichert, wodurch Datenschutz und Datensicherheit gewährleistet werden. Bei Bedarf können diese Daten dann nahtlos abgerufen und aktualisiert werden. Dabei spielen bei der Implementierung dieser Features folgende Methoden eine entscheidende Rolle:
"build" ist die Methode, die das Benutzerinterface basierend auf dem aktuellen Zustand der StatefulWidget-Klasse aufbaut. Sie ist der Ausgangspunkt für den Aufbau der UI-Elemente und ermöglicht die dynamische Erstellung und Aktualisierung des Bildschirms. Die Methode "initState" spielt eine wichtige Rolle bei der Initialisierung des Zustands der StatefulWidget-Klasse. Hier werden unter anderem Informationen aus dem lokalen Speicher geladen und andere vorbereitende Maßnahmen getroffen, um das Benutzerinterface entsprechend anzupassen und eine konsistente Benutzererfahrung zu gewährleisten. Diese beiden Methoden bilden das Rückgrat des Profile-Screens und ermöglichen es, mithilfe von weiteren Methoden, die Benutzeroberfläche zu strukturieren und den Zustand der App zu verwalten. Zusätzliche Methoden wie "openGallery" und "openVideo" sind von entscheidender Bedeutung, um dem Benutzer die Möglichkeit zu geben, Bilder und Videos auszuwählen und anzuzeigen. Durch das Öffnen neuer Bildschirmfenster bieten sie eine intuitive Benutzeroberfläche und ermöglichen es dem Benutzer, Inhalte zu erkunden. Darüber hinaus bieten sie Funktionen zum Anzeigen, Schließen und Löschen von Inhalten, was die Interaktivität und Benutzerfreundlichkeit der STH-App verbessert. Methoden wie "loadAvatarImage" und "loadUserName" spielen eine essenzielle Rolle, um den Avatar des Benutzers und seinen Benutzernamen aus dem lokalen Speicher zu laden.

Diese Informationen werden wiederum aus dem Accountprofile-Screen bezogen, was eine Integration und eine personalisierte Anpassung des Benutzerinterfaces ermöglicht. Durch das Laden dieser Daten können Benutzer ihre Profile individuell gestalten und ein konsistentes Benutzererlebnis über verschiedene Bildschirme hinweg gewährleisten. Im Kontext der Datenschutzsicherheit spielen Methoden wie "saveImagePathsToLocalStorage", "saveImage", "loadImagesFromStorage" und "uploadGalleryImages" eine entscheidende Rolle beim Speichern, Laden und Hochladen von Bildern. Diese Methoden ermöglichen es Benutzern, ihre Bilder sowohl lokal im lokalen Speicher über die SharedPreferences als auch in der Cloud über das Backend in Firebase sicher zu verwalten und zu teilen. Durch die Verwendung dieser Methoden wird die Integrität der Benutzerdaten gewahrt und gleichzeitig eine zuverlässige Speicherung und Übertragung von Bildinhalten gewährleistet. Ähnlich wichtig sind Methoden wie "saveVideoPathsToLocalStorage", "saveVideo", "loadVideosFromStorage" und "uploadVideos" für die Verwaltung von Videos. Sie ermöglichen es dem Benutzer, multimediale Inhalte nahtlos zu verwalten und zu teilen. Dabei werden die Videodaten sowohl lokal im lokalen Speicher gespeichert als auch über das Backend in Firebase hochgeladen, wo sie für weitere Verwendungszwecke verfügbar sind. Diese Methoden spielen somit eine essenzielle Rolle bei der Sicherung und Bereitstellung von Videoinhalten für die Benutzer der App. Um dem Benutzer die Kontrolle über seine Inhalte zu geben, dienen "deleteImage" und "deleteVideo" dazu, ausgewählte Bilder und Videos zu löschen. "generateThumbnail" ermöglicht es, Vorschaubilder für Videos anzuzeigen, um dem Benutzer eine Vorschau des Inhalts zu bieten. Die Methoden "saveHashtagsToLocalStorage", "loadHashtagsFromStorage" und "deleteHashtagFromLocalStorage" tragen zur Verwaltung von Hashtags bei und verbessern die Personalisierung und Relevanz der bereitgestellten Inhalte. Die Methode "showHashtagsModal" erlaubt es dem Benutzer, Hashtags auszuwählen und hinzuzufügen, um seine Interessen zu kennzeichnen. Sie fördert die Benutzerbeteiligung und ermöglicht die Personalisierung von Inhalten. Die Benutzeroberfläche des Profilbildschirms setzt sich aus einem Header-Bereich mit dem Benutzernamen und dem Profilbild sowie einem Abschnitt zur Anzeige und Verwaltung von Bild- und Videoinhalten zusammen. Die Anzeige kann zwischen Bildern und Videos umgeschaltet werden, und Benutzer haben die Möglichkeit, neue Medieninhalte auszuwählen und hochzuladen, welche von den Methoden pickMultiImage und pickVideo unterstützt werden. Für die Gestaltung der Benutzeroberfläche kommen verschiedene Flutter-Widgets wie Column, Stack, AppBar, CircleAvatar und GridView zum Einsatz, um eine ansprechende und benutzerfreundliche Erfahrung zu gewährleisten. Zusätzlich werden benutzerdefinierte Widgets wie CustomAppBar und CustomBottomNavigationBar verwendet, um die Navigation und Interaktion zu erleichtern und das Design konsistent zu halten.

Im Accountprofile-Screen hingegen erscheinen bearbeitbare personenbezogene Daten, wobei zur Validierung der Eingaben reguläre Ausdrücke (Regex) herangezogen werden. Den Nutzern wird die Gelegenheit geboten, ihre Profildaten anzupassen und zu sichern – darunter fallen der Name, die Telefonnummer, die Adresse und die E-Mail-Adresse. Beim Start des Bildschirms werden die aktuellen Profildaten aus den SharedPreferences geladen und in entsprechenden Textfeldern angezeigt. Nutzer können diese Informationen bearbeiten und die Änderungen anschließend speichern. Hier haben Nutzer die Freiheit, ein Profilbild zu wählen, das dann in einem kreisförmigen Avatar angezeigt wird. Das ausgesuchte Bild wird nicht nur lokal auf dem Gerät gespeichert, sondern auch im Backend

abgelegt. Bei Bedarf wird es von Firebase abgerufen, um anderen Seiten mit den entsprechenden Daten zu bereichern. Die Benutzeroberfläche des Accountprofilbildschirms besteht aus mehreren Textfeldern für die Bearbeitung von Profildaten sowie einem Abschnitt für das Profilbild. Im Bearbeitungsmodus werden die Textfelder kenntlich und sichtbar zur Bearbeitung hervorgehoben, sodass Nutzer ihre Daten ändern können. Fehlermeldungen erscheinen, falls ungültige Eingaben gemacht wurden, und ein Tooltip mit einem Hover-Effekt informiert die Nutzer über die Art des Fehlers. Auch hierbei wurden folgende Methode zur Implementierung benötigt, um unseren konzeptionellen Anforderungen gerecht zu werden:
Die Build-Methode erstellt die Benutzeroberfläche des Accountprofil-Screens. Sie umfasst Textfelder für die Bearbeitung von Profildaten sowie einen Bereich für das Profilbild. Die UI wird je nach Bearbeitungsstatus der Daten dynamisch angepasst. Die PickImage-Methode ermöglicht es Benutzern, ein Profilbild auszuwählen und es hochzuladen. Sie wird aktiviert, sobald der Benutzer auf die entsprechende Schaltfläche klickt. Dadurch wird die Bildauswahl gestartet, und das ausgewählte Bild wird für das Profil verwendet. Beim Start des Bildschirms ruft die LoadProfileData-Methode die aktuellen Profildaten aus den SharedPreferences ab. Diese Daten, einschließlich Name, Telefonnummer, Adresse und E-Mail, werden dann in den entsprechenden Textfeldern angezeigt, damit der Benutzer sie bearbeiten kann. Um das ausgewählte Profilbild lokal zu speichern und sicherzustellen, dass es für spätere Verwendungszwecke verfügbar ist, wird die Methode saveAvatarImage verwendet. Nachdem der Benutzer ein Bild ausgewählt hat, wird es lokal auf dem Gerät gespeichert. Zusätzlich dazu werden die aktualisierten Profildaten, einschließlich des Profilbildes, im Backend gespeichert, damit das Profilbild auch von anderen Seiten oder Anwendungen wiederverwendet werden kann. Die LoadAvatarImage-Methode wird aufgerufen, um das Profilbild aus der lokalen Speicherung zu laden. Dadurch wird das zuvor gespeicherte Profilbild beim Öffnen des Accountprofil-Screens geladen und im Kreisavatar angezeigt. Für die Validierung der personenbezogenen Daten werden unter Verwendung von regulären Ausdrücken (Regex) folgende Methoden verwendet. Diese dienen dazu, sicherzustellen, dass die eingegebenen Daten gültig sind, bevor sie gespeichert werden. Dabei werden sämtliche personenbezogenen Daten sowohl lokal auf dem Gerät als auch im Backend gespeichert, um die Integrität der Daten zu gewährleisten und sicherzustellen, dass sie für verschiedene Anwendungen verfügbar sind:

- Die ValidatePhone-Methode validiert die eingegebene Telefonnummer und akzeptiert nur Zahlen und optional ein Pluszeichen am Anfang
- Um die Gültigkeit der eingegebenen Adresse zu überprüfen, wird die ValidateAddress-Methode verwendet. Diese erwartet Buchstaben, Zahlen, Kommas, Punkte, Leerzeichen und Umlaute als gültige Zeichen
- Die ValidateEmail-Methode prüft, ob die eingegebene E-Mail-Adresse gültig ist und erwartet das übliche Format einer E-Mail-Adresse

Während des abschließenden Bearbeitungsvorgangs wird die Methode updateUserData aufgerufen. Diese Funktion dient dazu, die Profildaten in der Cloud Firestore-Datenbank zu aktualisieren. Dabei werden sämtliche neuen Informationen wie Name, Telefonnummer, Adresse und E-Mail in der Datenbank aktualisiert, um sicherzustellen, dass sie synchronisiert und auf dem neuesten Stand sind. Dies ermöglicht es, die aktualisierten Daten

für andere Zwecke zu beziehen und weiterzuverwenden. Sobald der Benutzer die Profildaten bearbeitet hat, werden diese mithilfe der SaveProfile-Methode in den SharedPreferences gespeichert, um sie beim nächsten Öffnen des Bildschirms wiederherzustellen. Die Interaktion zwischen dem Profilbildschirm und dem Accountprofilbildschirm sowie der Datenaustausch zwischen ihnen bestimmen die grundlegende Ausrichtung der appspezifischen Funktionen. Diese Funktionen erstrecken sich auf verschiedene Bereiche der Anwendung, wie beispielsweise den Home-Screen über das Backend, die dem Benutzer die Wiedergabe von Informationen in Form von Bildern, Videos und personenbezogenen Daten ermöglicht. Auch der Suchbildschirm profitiert von der Implementierung der Daten über das Backend, da dieser den Suchalgorithmus unterstützt und dem Benutzer dabei hilft, relevante Ergebnisse zu finden, die im Sinne einer Social-Media-App agieren.

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die angemeldete Prüfungsleistung in allen Teilen eigenständig ohne Hilfe von Dritten anfertigen und keine anderen als die in der Prüfungsleistung angegebenen Quellen und zugelassenen Hilfsmittel verwenden werde. Sämtliche wörtlichen und sinngemäßen Übernahmen inklusive KI-generierter Inhalte werde ich kenntlich machen. Diese Prüfungsleistung hat zum Zeitpunkt der Abgabe weder in gleicher noch in ähnlicher Form, auch nicht auszugsweise, bereits einer Prüfungsbehörde zur Prüfung vorgelegen; hiervon ausgenommen sind Prüfungsleistungen, für die in der Modulbeschreibung ausdrücklich andere Regelungen festgelegt sind. Mir ist bekannt, dass die Zuwiderhandlung gegen den Inhalt dieser Erklärung einen Täuschungsversuch darstellt, der das Nichtbestehen der Prüfung zur Folge hat und darüber hinaus strafrechtlich gem. § 156 StGB verfolgt werden kann. Darüber hinaus ist mir bekannt, dass ich bei schwerwiegender Täuschung exmatrikuliert und mit einer Geldbuße bis zu 50.000 EUR nach der für mich gültigen Rahmenprüfungsordnung belegt werden kann. Ich erkläre mich damit einverstanden, dass diese Prüfungsleistung zwecks Plagiatsprüfung auf die Server externer Anbieter hochgeladen werden darf. Die Plagiatsprüfung stellt keine Zurverfügungstellung für die Öffentlichkeit dar.

München, den 27. Mai 2024

Fitim Makolli