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: 4. Juli 2024

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

LaTeX	*Lamport TeX*
IDE	*Integrierte Entwicklungsumgebung*
iOS	*iPhone Operating System*
NFL	*National Football League*
MVP	*Minimal Viable Product*
STH-App	*Sport Talent Hub App*
VS Code	*Visual Studio Code*

1 Kurzbeschreibung

Diese Projektdokumentation behandelt das Konzept und die Implementierung der plattformunabhängigen STH-App für den Austausch zwischen Spielern und Sportmanagern. 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 ihre einzigartigen Fähigkeiten, Erfahrungen und die gemeinsame Vision, eine innovative Smartphone-App zu gestalten, sowie ihre Leidenschaft für Sport, erlangte das Team den notwendigen Spirit und die Dynamik, um ein herausragendes Produkt zu erschaffen.
Die STH-App, kurz für „Sports Talent Hub“, wurde entwickelt, um Sportlern und Sportmanagern eine reibungslose Interaktion und Vernetzung zu bieten. Spieler haben die Möglichkeit, ihre Profile zu erstellen, ihre sportlichen Erfolge und Höhepunkte zu teilen und mit möglichen Managern in Verbindung zu treten. Sportmanager können gleichzeitig Talente identifizieren, kontaktieren und für unterschiedliche sportliche Projekte einstellen. Nach vielen Recherchen und Überlegungen entwickelte das Projektteam eine Smartphone-App, die plattformunabhängig ist und den Austausch und das Anwerben zwischen Spielern und Sportmanagern ähnlich der Social Media App Instagram ermöglicht. Die STH-App zielt darauf ab, eine Plattform zu schaffen, die effizient und benutzerfreundlich ist und Sportlern sowie Managern die Kommunikation und Zusammenarbeit in einem dynamischen und interaktiven Umfeld ermöglicht.
Das Konzept und die Umsetzung eines MVP (Minimum Viable Product) für die Anwendung sollen zwischen dem 10. März 2024 und dem 5. Juli 2024 entwickelt und in dieser Dokumentation erläutert werden. Damit das Projekt flexibel auf Veränderungen reagieren und kontinuierliche Verbesserungen umsetzen kann, wird es agil in einer angepassten Form von Scrum durchgeführt. Die Gründung eines Teams und eine deutliche Verteilung der Rollen innerhalb des Projekts sind für den Erfolg von großer Bedeutung. Jeder Teilnehmer des Projekts bringt bestimmte Kompetenzen und Erfahrungen mit sich, die das gesamte Projektteam stärken. Als Projektleiter übernimmt Jonas Waigel zusätzlich zur App-Entwicklung die Koordination der Tätigkeiten und stellt sicher, dass die Zeitpläne und Ziele eingehalten werden. Die erfolgreiche Implementierung der App wird wesentlich von Fitim Makolli, Hasan Deveci und Vatsegkan Zournatsidis durch ihre Fachkenntnisse in den Feldern Entwicklung, Design und Qualitätssicherung beeinflusst.

2 Ziele

2.1 Projektbegründung

Mit der STH-App soll eine neuartige und benutzerfreundliche Plattform entwickelt werden, die den Austausch von Sportlern und Sportmanagern verbessert. Es ist unverzichtbar, in einer immer stärker digitalisierten und vernetzten Welt effiziente Kommunikationswege und wirksame Instrumente bereitzustellen, um Talente zu identifizieren und zu fördern. Das Ziel der STH-App ist es, diese Lücke zu überbrücken und einen wichtigen Ort für die Interaktion zwischen Spielern und Managern zu schaffen.
Eines der zentralen Ziele der STH-App besteht darin, das Networking und den Austausch zwischen Sportlern und Sportmanagern zu fördern. Die Implementierung einer plattformunabhängigen App ermöglicht eine Kommunikation über geografische und organisatorische Grenzen hinweg. Sportler haben die Möglichkeit, ihre Profile und Leistungen vorzustellen, während Sportmanager gezielt nach Talenten suchen und diese kontaktieren können. Dadurch wird nicht nur die berufliche Laufbahn der Sportler gefördert, sondern es wird auch Sportmanagern ermöglicht, begabte Spieler zu finden und zu gewinnen.
Die weiteren Ziele, aufgeteilt in Muss- / Soll- / Kann- und Nicht-Ziele werden im nachfolgenden Punkt aufgelistet.

2.2 Aufstellung der einzelnen Ziele

Muss-Ziele

- Die STH-App muss als plattformunabhängige App mit einem 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 inklusive aller Screens in der App bereitstellen.
- Die STH-App muss für jeden Screen ein einheitliches Design aufweisen und bis zur Präsentation Design-Anpassungen und Verbesserungen durch das Projektteam durchgeführt werden.

Soll-Ziele

- Die STH-App soll bis zur Abgabe der Projektarbeit am 05.07.2024 auf verschiedenen Test-Geräten und Simulatoren / Emulatoren getestet werden und Fehler als neue Aufgaben eingestellt werden.
- Gefundene Fehler bei App-Tests sollen bis zum 05.07.2024 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 App Store 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 Home-Screen 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.

2.3 Methodik

Die STH-App nutzt agile Methoden, speziell Scrum, des Projektmanagements, die speziell auf die Projektanforderungen zugeschnitten sind. Um sicherzustellen, dass die STH-App den Grundsätzen des kontinuierlichen Verbesserungsprozesses entspricht und auf neue Anforderungen von Interessengruppen flexibel reagieren kann, wurde die Scrum-Methodik gewählt. Das STH-App-Projekt ist dabei in verschiedene Sprints unterteilt.

Für das STH-App-Projekt soll jeder Sprint eine Dauer von 2 Wochen haben. Am Ende eines Sprints wird die entstandene Funktion bewertet. Zu Beginn jedes Sprints findet das Sprint-Planning statt, um die Ziele und Aufgaben für den kommenden Sprint festzulegen. Zunächst werden die anstehenden Aufgaben im Product Backlog Reihenfolge des Product Backlogs, welches alle anstehenden Aufgaben und Features enthält. Während der Sprint-Planung werden die Aufgaben identifiziert, die im nächsten Sprint erledigt werden sollen. Das Team analysiert die benötigte Zeit für jede spezifische Aufgabe. Während des Sprints finden regelmäßige Stand-up-Meetings statt, die als Weeklys bekannt sind. Hier berichtet jedes Teammitglied kurz über seine Arbeit, erreichte Fortschritte und bestehende Herausforderungen. Nach jedem Sprint findet ein Sprint-Review statt, bei dem die Ergebnisse des Sprints präsentiert und bewertet werden. Das Team informiert die Stakeholder über die durchgeführten Funktionen und Aufgaben und gibt Feedback. Nach der Überprüfung des Sprints findet die Sprint-Retrospektive statt, in der das Team den vergangenen Sprint reflektiert und potenzielle Verbesserungen identifiziert. In der Retrospektive werden positive Ereignisse, aufgetretene Probleme und mögliche Verbesserungen erörtert, um kontinuierlich am Entwicklungsprozess zu arbeiten. Der Entwicklungsprozess beinhaltet mehrere Etappen, die mit der Anforderungsanalyse starten und detaillierte Beschreibungen der funktionalen und nicht-funktionalen Anforderungen bereitstellen.

2.4 Technologie / Theorie / Hintergrund Beschreibung

Die STH-App verwendet Firebase und Flutter. Das sind fortschrittliche Technologien und Frameworks, die sicherstellen, dass eine gute Leistung und Benutzerfreundlichkeit aufgewiesen wird. Firebase wird als Backend-Service verwendet, um eine schnelle und sichere Datenverwaltung zu gewährleisten. GitHub wird genutzt, um die Versionskontrolle und das zentrale Code-Repository sicherzustellen. Visual Studio Code ist die verwendete Entwicklungsumgebung (IDE), die die Produktivität von Entwicklern steigert, indem sie zahlreiche Erweiterungen und Tools, vor allem für die verwendeten Frameworks, bietet. Die erfolgreiche Umsetzung der STH-App erfordert die Anwendung agiler Entwicklungsmethoden und moderner Technologien. Durch die Verwendung agiler Prinzipien, moderner Entwicklungstools und theoretischer Erkenntnisse aus verschiedenen Fachbereichen wird eine effiziente, benutzerfreundliche und leistungsstarke Plattform geschaffen, die den Austausch zwischen Managern und Sportlern ermöglicht.

3 Risikoanalyse

Die Risikoanalyse dient dazu, kritische Einflüsse innerhalb und außerhalb des Projektes zu identifizieren. Für die STH App gilt vergleichbar zu allen anderen Projekten, dass der Projekterfolg nur mit begleitenden potenziellen Risiken ermöglicht werden kann. Für die Klassifizierung und Einschätzungen der Risiken dient eine Risikomatrix. Hierbei wird aufgezeigt, welche Risiken mit welcher Eintrittswahrscheinlichkeit und zugehöriger Auswirkung eintreten können. Anhand dessen kann bewertet werden, welche Risiken besonders beobachtet werden müssen und ob es Risiken gibt, die den Projekterfolg maßgeblich gefährden.

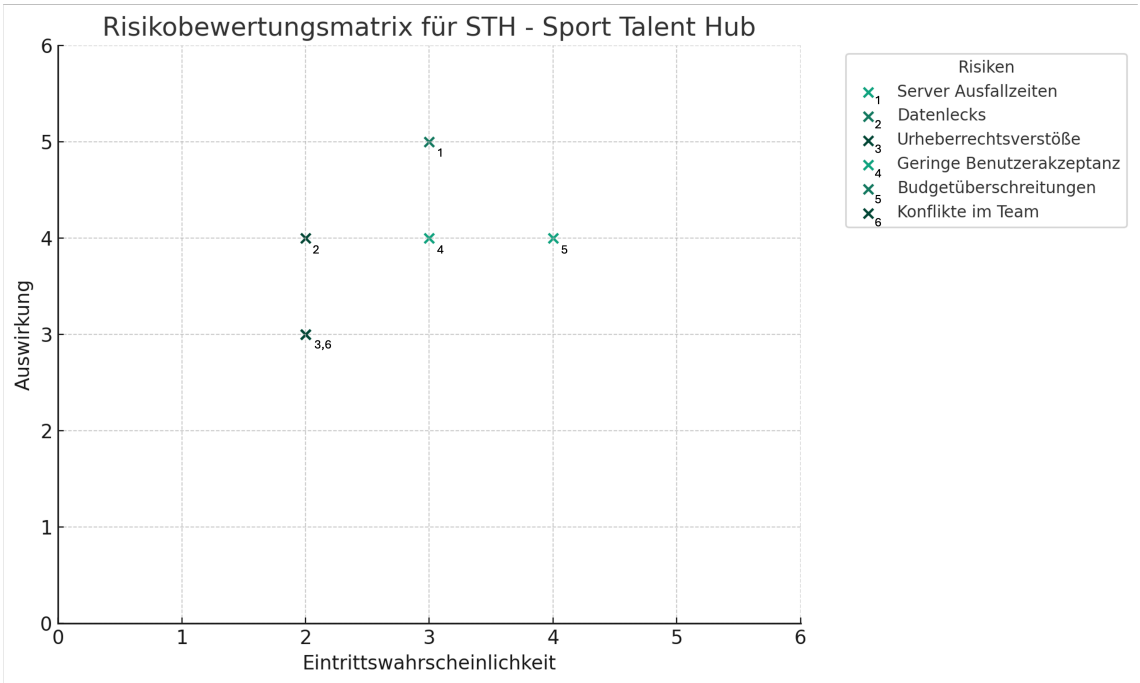
3.1 Vorgehen Risikoanalyse

Um eine Risikobewertungsmatrix erstellen und veranschaulichen zu können, benötigt es folgende Schritte. Zunächst müssen die Risiken erkannt werden, die im Projekt auftreten können. Dabei werden sowohl interne als auch externe Faktoren beleuchtet. Zu den internen Risiken gehören z.B. Konflikte im Team und zu den externen Server Ausfallzeiten beim externen Dienstleister. Hierbei ist zu beachten, dass mit jedem vorangegangenen Projektfortschritt auch neue Risiken hinzukommen können. Daraufhin werden die Risiken nach ihrer Eintrittswahrscheinlichkeit klassifiziert. Die Einordnung hilft dabei Risiken zu priorisieren. Dabei werden Risiken mit hohen Eintrittswahrscheinlichkeiten genauer betrachtet und in Zukunft im Blick behalten. Zuletzt werden die Risiken nach ihrer Auswirkung eingestuft. Risiken mit hoher Eintrittswahrscheinlichkeit und hohen Auswirkungen können hierbei zum Scheitern des Projektes bzw.żum Nichterfolg führen. Deshalb ist es besonders wichtig diese Art von Risiken nicht nur zu beobachten, sondern kontinuierlich zu messen, welche Folgen der Eintritt haben kann.

3.2 Grafische Darstellung Risikobewertungsmatrix

Die Risikobewertungsmatrix für das Projekt STH App ist in der folgenden Abbildung zu sehen:

Bild 2: Risikomatrix



Quelle: Eigene Darstellung

Für eine App mit diversen online Funktionen ist ein Server Ausfall fatal. Deshalb wurde dieses Risiko als eines der mit am höchsten verbunden Auswirkungen klassifiziert. Die Eintrittswahrscheinlichkeit bewegt sich hierbei im mittleren Bereich, da nach dem aktuellen Stand der Technik und der Rahmenverträge mit Dienstleistern bzw.Ṙechenzentren bei einem Ausfall meist auf eine redundante Serverlandschaft ausgewichen werden kann.
Ein Datenleck, welches durch einen Angriff auf die Backendsysteme entstehen kann, ist für den Erfolg und gleichzeitig für die Auswirkungen einer Smartphone-App schwerwiegend. Innerhalb der STH App können sensible Daten eingegeben und abgespeichert werden, welche nicht an außenstehende gelangen dürfen. Zudem sind die immensen Kosten bezüglich der Vertragsstrafen bei der Nichteinhaltung der DSGVO ein großes finanzielles Risiko.
Ein weiteres Risiko für die Applikation sind Urheberrechtsverstöße. Diese können auftreten, indem Nutzer Urheberrechtsgeschützte Inhalte veröffentlichen, welche nicht Ihnen gehören. Eine Begünstigung der Urheberrechtsverstöße könnte auf die Applikation selbst zurückzuführen sein und damit finanziell intensive Vertragsstrafen auslösen.
Eine geringe Benutzerakzeptanz kann durch ein vorher schlecht ausgearbeitetes UI/UX Konzept ausgelöst werden. Die Eintrittswahrscheinlichkeit ist hierbei im mittleren Bereich, da es nicht einfach ist, den Nutzern ein qualitativ hochwertiges und gut durchdachtes User-Interface zu liefern. Die Auswirkung liegt hierbei ebenso im mittleren Bereich,

da das Design schnell angepasst werden kann.
Die Budgetüberschreitung ist wie in fast jedem Projekt ein potenzielles Risiko, welches durchaus durch eine schlechte Planung auftreten kann. Sobald die finanziellen Ressourcen ausgeschöpft sind, kann nicht mehr an der Applikation gearbeitet und weiterentwickelt werden. Das könnte unter Umständen zu einem frühzeitigen Scheitern des Projektes führen.
In jedem Projektteam ist immer ein gewisses Konfliktpotenzial vorhanden. Dieses Risiko kann jederzeit und vor allem in Hochphasen wie z.B. kurz vor dem Start der Veröffentlichung der Applikation, auftreten. Allerdings ist das Risiko durch eine gute Projektleitung gut einschätzbar und präventiv vermeidbar bzw.żu lösen.

4 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.

5 Implementierungsplan und Implementierungsdurchführung

In diesem Abschnitt wird die Implementierung des Prototypen genau geplant. Für die Planung haben sich die Gruppenmitglieder der STH-App in gemeinsamen Meetings abgestimmt. Im folgenden werden die Schritte genauer erläutert.

5.1 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
- Verantwortliche/r: Das Team
- Ressourcen: VS Code, Flutter SDK, Android Emulator und iOS Simulator
- Aufgabenstellung: Einrichtung der Entwicklungsumgebung für Flutter

Arbeitspaket 2 vom 08.03.2024 bis 11.03.2024

- Bezeichnung: Initial Setup LaTeX
- Verantwortliche/r: Das Team
- Ressourcen: VS Code inclLaTeX-Plugins, Container-Plattform (Docker)
- Aufgabenstellung: Einrichtung der Entwicklungsumgebung für LaTeX

Arbeitspaket 3 vom 08.03.2024 bis 11.03.2024

- Bezeichnung: Initial Setup GitHub
- Verantwortliche/r: 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
- Verantwortliche/r: Das Team
- Ressourcen: Brainstorming
- Aufgabenstellung: Definition der App Funktionalitäten

Arbeitspaket 5 vom 14.03.2024 bis 15.03.2024

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

Arbeitspaket 6 vom 15.03.2024 bis 22.03.2024

- Bezeichnung: Home-Screen
- Verantwortliche/r: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung des Home-Screens inklusive Menüleiste (Bottom Navigationbar) am unteren Bildschirmrand und App Bar am oberen Bildschirmrand

Arbeitspaket 7 vom 18.03.2024 bis 01.04.2024

- Bezeichnung: Chat-Screen
- Verantwortliche/r: 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
- Verantwortliche/r: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung des Account-Profile-Screens inklusive Avatar und personenbezogene Daten

Arbeitspaket 9 vom 22.03.2024 bis 05.04.2024

- Bezeichnung: Search-Screen
- Verantwortliche/r: Herr Vatsegkan Zournatsidis
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Initiale Implementierung des Search-Screens

Arbeitspaket 10 vom 22.03.2024 bis 01.04.2024

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

Arbeitspaket 11 vom 25.03.2024 bis 08.04.2024

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

Arbeitspaket 12 vom 25.03.2024 bis 01.04.2024

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

Arbeitspaket 13 vom 25.03.2024 bis 01.04.2024

- Bezeichnung: Logo Design
- Verantwortliche/r: 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
- Verantwortliche/r: 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 05.04.2024

- Bezeichnung: GetStream-Chat
- Verantwortliche/r: 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
- Verantwortliche/r: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Anpassung der App-Leiste auf dem Chat-Screen

Arbeitspaket 17 vom 29.03.2024 bis 04.04.2024

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

Arbeitspaket 18 vom 01.04.2024 bis 08.04.2024

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

Arbeitspaket 19 vom 01.04.2024 bis 08.04.2024

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

Arbeitspaket 20 vom 02.04.2024 bis 08.04.2024

- Bezeichnung: Chat UI Adjustments
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Anpassung der App-Leiste für die Navigation und das Verhalten beim Seitenwechsel

Arbeitspaket 32 vom 12.04.2024 bis 19.04.2024

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

Arbeitspaket 33 vom 15.04.2024 bis 16.04.2024

- Bezeichnung: Page Linking
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: Herr Vatsegkan Zournatsidis
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Hinzufügen vom Logo auf die Startseite

Arbeitspaket 36 vom 22.04.2024 bis 29.04.2024

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

Arbeitspaket 37 vom 22.04.2024 bis 01.05.2024

- Bezeichnung: Avatar Transmission
- Verantwortliche/r: 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 06.05.2024

- Bezeichnung: Media Transmission
- Verantwortliche/r: 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 26.04.2024 bis 29.04.2024

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

Arbeitspaket 40 vom 29.04.2024 bis 30.04.2024

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

Arbeitspaket 41 vom 29.04.2024 bis 06.05.2024

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

Arbeitspaket 42 vom 29.04.2024 bis 06.05.2024

- Bezeichnung: Profile Navigation
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: Herr Hasan Deveci
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Implementierung neuer Chat-User als Testdaten

Arbeitspaket 47 vom 06.05.2024 bis 15.05.2024

- Bezeichnung: Hashtag Implementation
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: 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
- Verantwortliche/r: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework
- Aufgabenstellung: Fehlerbehebungen für den Profile-Screen und Account-Profile-Screen

Arbeitspaket 51 vom 10.06.2024 bis 14.06.2024

- Bezeichnung: Search-Screen: search function
- Verantwortliche/r: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework, Firebase
- Aufgabenstellung: Implementierung der Benutzer in Firebase und Implementierung der Firebase-Funktion zum Suchen der User

Arbeitspaket 52 vom 14.06.2024 bis 14.06.2024

- Bezeichnung: Search-Screen: avatar icon
- Verantwortliche/r: Herr Fitim Makolli
- Ressourcen: VS Code, Flutter Framework, Firebase
- Aufgabenstellung: Implementierung leerer Avatar-Icons der aufgelisteten User

Arbeitspaket 53 vom 14.06.2024 bis 14.06.2024

- Bezeichnung: Update Home-Screen with profileImages
- Verantwortliche/r: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework, Firebase
- Aufgabenstellung: Hinzufügen von Avatarbildern aus dem Firebase Storage im Home-Screen

Arbeitspaket 54 vom 18.06.2024 bis 18.06.2024

- Bezeichnung: Update Search-Screen with profileImages
- Verantwortliche/r: Herr Jonas Waigel
- Ressourcen: VS Code, Flutter Framework, Firebase
- Aufgabenstellung: Hinzufügen von Avatarbildern aus dem Firebase Storage im Search-Screen

5.2 Meilensteinplanung

Im Rahmen dieses Projekts wurde eine umfassende Planung und Organisation der Arbeitspakete durchgeführt. Dies umfasste die Analyse der Tabelle mit den Arbeitspaketen sowie die Erstellung eines Gantt-Diagramms. Letzteres bietet eine visuelle Darstellung der zeitlichen Abfolge und Dauer der verschiedenen Aufgaben, was die Projektplanung und -kontrolle erleichtert.

5.3 Vorgehensweise zur Erstellung des Gantt-Diagramms

Zunächst wurden alle relevanten Daten aus den Arbeitspaketen extrahiert und aufbereitet. Die Tabelle enthält die zwei wesentlichen Spalten "Beschreibung" und "Dauer". Zur Erstellung des Gantt-Diagramms wurde die Website "onlinegantt.com" in Anspruch genommen. Diese Plattform bietet eine benutzerfreundliche Oberfläche zur Eingabe und Visualisierung von Projektplänen.
Für jedes Arbeitspaket wurde das Startdatum gemäß der Tabelle eingetragen. Dabei wurde die Aufgabenbeschreibung als Titel für die jeweiligen Arbeitspakete zusammengefasst. Die Dauer der Aufgaben wurde entsprechend der Tabelle kategorisiert und entweder in Tagen oder Wochen angegeben.
Dadurch wird die zeitliche Abfolge der Aufgaben visualisiert, um mögliche Überschneidungen oder Abhängigkeiten zu erkennen. Die Erstellung des Gantt-Diagramms zeigt die klare Struktur des Projekts im Zeitraum von 08.03.2024 bis 18.06.2024. Die logische und sequenzielle Anordnung der Aufgaben erleichtert die Nachverfolgung und Kontrolle des Projekts.

5.4 Planung der Arbeitspakete

Dabei beanspruchen einige Aufgaben, wie in etwa die Einrichtung des GitHub-Repositories oder die Erstellung der Projektstruktur, nur einen Tag. Andere Aufgaben, wie die Implementierung des Chat-Screens oder die Anpassung der Seiten-Navigation, erstrecken sich über mehrere Wochen aufgrund der Komplexität und der Zeit für das Lösen von auftretenden Fehlern beim Testen. Diese kürzeren Aufgaben wurden in verschiedenen Sprints parallel zu den umfangreicheren Aufgaben geplant, um die Gesamtdauer des Projekts zu minimieren und schneller Fortschritte sehen zu können.

5.5 Deliverables Liste

Um sicherzustellen, dass alle Ergebnisse des Projekts klar definiert und termingerecht geliefert werden, wurde eine Deliverables Liste erstellt. Diese Liste umfasst alle Ergebnisse, die während des Projektverlaufs an die Stakeholder übergeben werden müssen. Die folgende Aufstellung der Arbeitspakete zeigt detailliert, welche Aufgaben zu erledigen sind, wer dafür verantwortlich ist, welche Ressourcen benötigt werden und in welchem Zeitraum die Aufgaben abgeschlossen sein sollen.

Bild 3: Deliverables

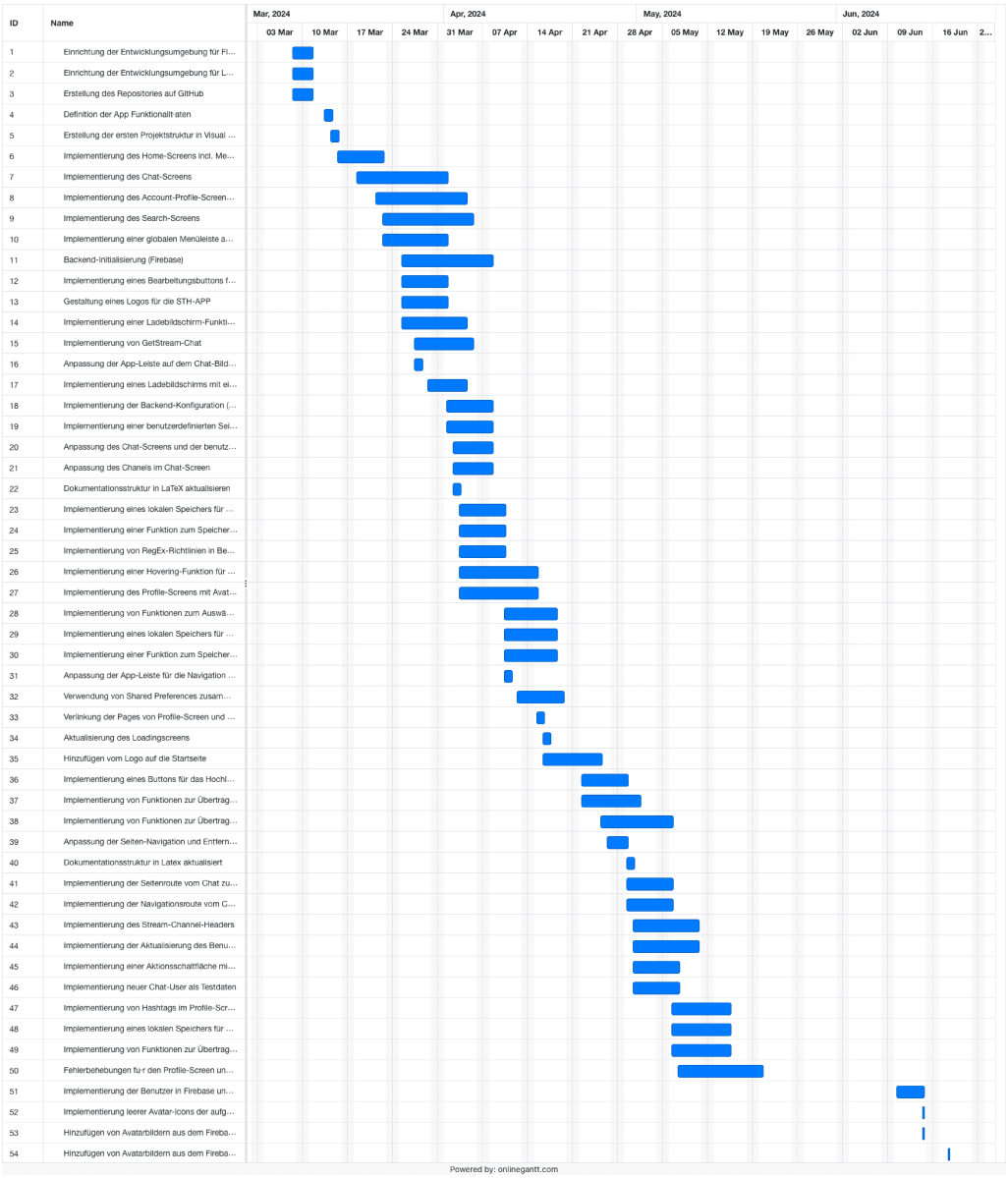
Arbeitspaket	Zeitraum	Verantwortliche/r	Ressourcen	Aufgabenstellung
Initial Setup Flutter	08.03.2024 - 11.03.2024	Das Team	VS Code, Flutter SDK, Android Emulator, iOS Simulator	Einrichtung der Entwicklungsumgebung fÝr Flutter
Initial Setup LaTeX	08.03.2024 - 11.03.2024	Das Team	VS Code inkl. LaTeX-Plugins, Container-Plattform (Docker)	Einrichtung der Entwicklungsumgebung fÝr LaTeX
Initial Setup GitHub	08.03.2024 - 11.03.2024	Das Team	Git-Client, GitHub-Konto	Erstellung des Repositories auf GitHub
Project Roadmap	13.03.2024 - 14.03.2024	Das Team	Brainstorming	Definition der App FunktionalitŠten
Project Structure	14.03.2024 - 15.03.2024	Herr Jonas Waigel	VS Code, Flutter Framework	Erstellung der ersten Projektstruktur in Visual Studio Code und Flutter
Home-Screen	15.03.2024 - 22.03.2024	Herr Jonas Waigel	VS Code, Flutter Framework	Implementierung des Home-Screens inklusive MenÝleiste (Bottom Navigationbar) und App Bar
Chat-Screen	18.03.2024 - 01.04.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Implementierung des Chat-Screens
Account Profile	21.03.2024 - 04.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung des Account-Profile-Screens inklusive Avatar und personenbezogene Daten
Search-Screen	22.03.2024 - 05.04.2024	Herr Vatsegkan Zournatsidis	VS Code, Flutter Framework	Initiale Implementierung des Search-Screens
Global-Menu-Bar	22.03.2024 - 01.04.2024	Herr Jonas Waigel	VS Code, Flutter Framework	Implementierung einer globalen MenÝleiste am unteren Bildschirmrand fÝr alle Pages
Initial Setup Backend	25.03.2024 - 08.04.2024	Herr Jonas Waigel	VS Code, Flutter Framework, Backend-Plattform (Firebase)	Backend-Initialisierung (Firebase)
Profile-Edit-Button	25.03.2024 - 01.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung eines Bearbeitungsbuttons fÝr den Account-Profile-Screen
Logo Design	25.03.2024 - 01.04.2024	Herr Vatsegkan Zournatsidis	ChatGPT 4.0	Gestaltung eines Logos fÝr die STH-APP
Loading-Screen	25.03.2024 - 04.04.2024	Herr Vatsegkan Zournatsidis	VS Code, Flutter Framework	Implementierung einer Ladebildschirm-Funktion beim Start der STH-App
GetStream-Chat	27.03.2024 - 05.04.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Implementierung von GetStream-Chat
Chat-App-Bar	27.03.2024 - 28.03.2024	Herr Jonas Waigel	VS Code, Flutter Framework	Anpassung der App-Leiste auf dem Chat-Screen
Logo Loading-Screen	29.03.2024 - 04.04.2024	Herr Vatsegkan Zournatsidis	VS Code, Flutter Framework	Implementierung eines Loading-Screen mit einem Logo
Backend Config	01.04.2024 - 08.04.2024	Herr Jonas Waigel	VS Code, Flutter Framework, Backend-Plattform (Firebase)	Implementierung der Backend-Konfiguration (Firebase), Authentifizierung und deren Methoden
Custom Page Routing	01.04.2024 - 08.04.2024	Herr Jonas Waigel	VS Code, Flutter Framework	Implementierung einer benutzerdefinierten Seitenroute beim Wechsel von Pages
Chat UI Adjustments	02.04.2024 - 08.04.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Anpassung des Chat-Screens und der benutzerdefinierte App-Leiste
Channel Adjustment	02.04.2024 - 08.04.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Anpassung des Chanels im Chat-Screen
Latex Docs Update	02.04.2024 - 03.04.2024	Herr Jonas Waigel	VS Code	Dokumentationsstruktur in LaTeX aktualisieren
Local Storage Setup	03.04.2024 - 10.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung eines lokalen Speichers fÝr den Account-Profile-Screen
Save/Load Functions	03.04.2024 - 10.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung einer Funktion zum Speichern/Laden fÝr den Account-Profile-Screen
RegEx Guidelines	03.04.2024 - 10.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung von RegEx-Richtlinien in Bezug auf personenbezogene Daten
Hovering Function	03.04.2024 - 15.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung einer Hovering-Funktion fÝr personenbezogene Daten
Profile UI	03.04.2024 - 15.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung des Profile-Screens mit Avatar, Bilder und Mediathek
Media Upload	10.04.2024 - 18.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung von Funktionen zum AuswŠhlen und Hochladen von Bilder und Videos
Profile Local Storage	10.04.2024 - 18.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung eines lokalen Speichers fÝr den Profile-Screen
Profile Save/Load	10.04.2024 - 18.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung einer Funktion zum Speichern/Laden fÝr den Profile-Screen
App Bar Navigation	10.04.2024 - 11.04.2024	Herr Jonas Waigel	VS Code, Flutter Framework	Anpassung der App-Leiste fÝr die Navigation und das Verhalten beim Seitenwechsel
Shared Preferences	12.04.2024 - 19.04.2024	Herr Jonas Waigel	VS Code, Flutter Framework, Backend-Plattform (Firebase)	Verwendung von Shared Preferences zusammen mit Firebase-Testdaten
Page Linking	15.04.2024 - 16.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Verlinkung der Pages von Profile-Screen und Account-Profile-Screen
Loading-Screen Update	16.04.2024 - 17.04.2024	Herr Vatsegkan Zournatsidis	VS Code, Flutter Framework	Aktualisierung des Loadingscreens
Start Page Logo	16.04.2024 - 25.04.2024	Herr Vatsegkan Zournatsidis	VS Code, Flutter Framework	HinzufÝgen vom Logo auf die Startseite
Media Button	22.04.2024 - 29.04.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung eines Buttons fÝr das Hochladen von Bilder und Videos
Avatar Transmission	22.04.2024 - 01.05.2024	Herr Fitim Makolli	VS Code, Flutter Framework, Backend-Plattform (Firebase)	Implementierung von Funktionen zur †bertragung vom Avatar zum Backend
Media Transmission	25.04.2024 - 06.05.2024	Herr Fitim Makolli	VS Code, Flutter Framework, Backend-Plattform (Firebase)	Implementierung von Funktionen zur †bertragung von Bilder und Videos an das Backend
Page Navigation	26.04.2024 - 29.04.2024	Herr Jonas Waigel	VS Code, Flutter Framework	Anpassung der Seiten-Navigation und Entfernung von Animationen beim Wechsel
Latex Docs Update	29.04.2024 - 30.04.2024	Herr Fitim Makolli	VS Code	Dokumentationsstruktur in LaTeX aktualisiert
Chat Routing	29.04.2024 - 06.05.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Implementierung der Seitenroute vom Chat zum Chanelscreen
Profile Navigation	29.04.2024 - 06.05.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Implementierung der Navigationsroute vom Chat-Screen zum Profile-Screen
Stream Headers	30.04.2024 - 10.05.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Implementierung des Stream-Channel-Headers
Username Update	30.04.2024 - 10.05.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung der Aktualisierung des Benutzernamens auf dem Profile-Screen
Action Button	30.04.2024 - 07.05.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Implementierung einer AktionsschaltflŠche mit Benutzerintegration
Test User Addition	30.04.2024 - 07.05.2024	Herr Hasan Deveci	VS Code, Flutter Framework	Implementierung neuer Chat-User als Testdaten
Hashtag Implementation	06.05.2024 - 15.05.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung von Hashtags im Profile-Screen
Hashtag Storage	06.05.2024 - 15.05.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung eines lokalen Speichers fÝr die Hashtags
Hashtag Transmission	06.05.2024 - 15.05.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Implementierung von Funktionen zur †bertragung von Hashtags an das Backend
Major Bug Fixes	07.05.2024 - 20.05.2024	Herr Fitim Makolli	VS Code, Flutter Framework	Fehlerbehebungen fÝr den Profile-Screen und Account-Profile-Screen
Search-Screen: search function	10.06.2024 - 14.06.2024	Herr Fitim Makolli	VS Code, Flutter Framework, Firebase	Implementierung der Benutzer in Firebase und Implementierung der Firebase-Funktion zum Suchen der User
Search-Screen: avatar icon	14.06.2024 - 14.06.2024	Herr Fitim Makolli	VS Code, Flutter Framework, Firebase	Implementierung leerer Avatar-Icons der aufgelisteten User
Update Home-Screen with profileImages	14.06.2024 - 14.06.2024	Herr Jonas Waigel	VS Code, Flutter Framework, Firebase	HinzufÝgen von Avatarbildern aus dem Firebase Storage im Home-Screen
Update Search-Screen with profileImages	18.06.2024 - 18.06.2024	Herr Jonas Waigel	VS Code, Flutter Framework, Firebase	HinzufÝgen von Avatarbildern aus dem Firebase Storage im Search-Screen

Quelle: Eigene Darstellung

5.6 Visualisierung

Durch das Gantt-Diagramm konnten potenzielle Engpässe und Risiken identifiziert werden. Ein besonderes Augenmerk gilt Aufgaben, die kritische Pfade darstellen, um Verzögerungen zu vermeiden.

Bild 4: Meilensteinplanung



Quelle: Eigene Darstellung über https://www.onlinegantt.com

6 Stakeholderanalyse

Für die STH App sind Stakeholder aus verschiedenen Bereichen vorhanden. Die Applikation erstrebt einen großen Einfluss auf die Sportindustrie und das speziell auf den Prozess des Anwerbens von neuen Talenten durch Vereine in unterschiedlichen Größen. Deshalb ist eine Stakeholderanalyse besonders wichtig, um die verschiedenen Gruppen an Interessenten identifizieren zu können. Für die Analyse werden folgende Gruppen an Stakeholdern betrachtet: Interne-, Externe-, Community- und Technische-Stakeholder.

Die Stakeholderanalyse wurde mit einem konkreten Vorgehen erstellt. Zuerst wurde erneut die Größe und Auswirkung der Applikation betrachtet. Dabei ist es auch wichtig, die Zielsysteme iOS und Android nicht außer Acht zu lassen, um konkrete Benutzergruppen definieren zu können. Innerhalb des Projektteams wurde in einem Meeting gemeinsam festgelegt, welche Stakeholder für die STH-App infrage kommen.

6.1 Management

Zunächst wurden die internen Stakeholder und das Management betrachtet. Dabei wurde das Projektteam und das Hochschulpersonal identifiziert. Das Projektteam der STH-App ist am Erfolg der Applikation interessiert, da das Projektteam sowohl an der Organisation und an der Entwicklung der STH-App beteiligt ist.

6.2 Partner / Stakeholder

Zu den Partnern und Stakeholdern hat das Projektteam folgende Personengruppen erkannt. Amateursportler, Sportagenturen, Manager, Vereine und Scouts sind externe Interessensgruppen, die den Erfolg der Applikation als relevant betrachten. Diese Gruppen an Personen haben einen direkten Bezug zur Applikation und stellen die zukünftigen Anwender dar. Das Interesse am Verlauf und Erfolg der Applikation ist in diesen Gruppen stark vertreten, da hier ein Mehrwert für all diese Gruppen erzielt werden kann. Aus der Hülle der Anwender heraus wurden weitere Stakeholdergruppen vom Projektteam erkannt.

Eine weitere relevante Gruppe ist die Community hinter den Anwendern. Dazu zählen Sportmedien und große Sportverbände, die keinen direkten Bezug zur Applikation haben, aber dennoch am Einfluss der STH-App in ihren Sportbereichen interessiert sind. Sportmedien werden über die Bekanntmachungen von neuen Verträgen über die STH-App berichten. Sportverbände sind an großen und performanten Sportvereinen interessiert, welche über die STH-App die Kommunikation für Spielertransfers durchführen können.

6.3 Externe Partner

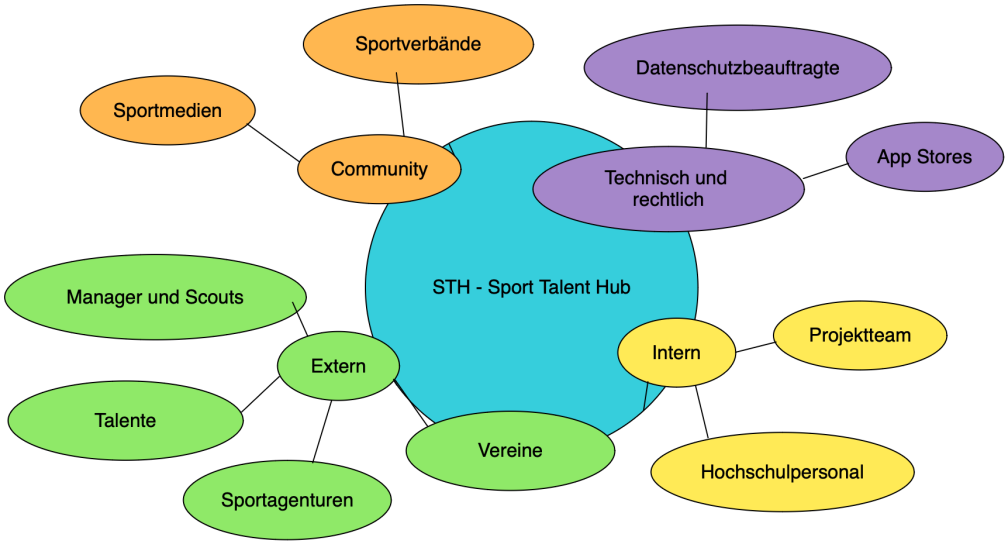
Zu den Externen Stakeholdern wurden zudem die technischen und rechtlichen Interessensgruppen erkannt.Innerhalb der STH-App werden persönliche Daten wie z.B. Name, Vorname, Wohnort, PLZ und die E-Mail hinterlegt. Die personenbezogenen Daten unterliegen strikten Regeln durch die Gesetzgebung der Datenschutz-Grundverordnung. Datenschutzbeauftragte werden ein großes Interesse an der korrekten Datenhaltung der STH-App nach der Veröffentlichung der Anwendung haben. Zudem sind die App-Stores wie der Google Play Store und der Apple App Store daran interessiert, dass diese Gesetzgebungen eingehalten werden und zudem auch die Richtlinien und Vorgaben der Stores eingehalten werden. Die Applikation muss für eine Veröffentlichung auf diesen Plattformen diesen Regelungen strikt folgen. Deshalb zählen diese Firmen und Personengruppen zu einem weiteren großen Teil der Stakeholder.

Die Stakeholderanalyse zeigt zum aktuellen Zeitpunkt der STH-App eine Vielzahl an Interessensgruppen auf, die am Erfolg oder an dem Verlauf des Projektes STH-App interessiert sind.

6.4 Grafische Darstellung Stakeholderanalyse

In der folgenden Abbildung sind die Stakeholdergruppen und deren Abhängigkeiten aufgezeichnet.

Bild 5: Stakeholderanalyse



Quelle: Eigene Darstellung

7 Projektressourcen

Im nächsten Schritt werden die Ressourcen beschrieben. Dazu zählen die am STH-App-Projekt beteiligten Personen und die materiellen Ressourcen. Da es sich um ein Studienprojekt handelt, das die Entwicklung eines MVP einer App zum Ziel hatte, sind die aufgeführten Ressourcen speziell auf diesen Rahmen zugeschnitten.

7.1 Personen

7.1.1 Projektteam

Das Projektteam bestand aus vier Mitgliedern: Jonas Waigel, Fitim Makolli, Hasan Deveci und Vatsegkan Zournatsidis. Jonas Waigel wurde als Projektleiter gewählt und war für die Koordination und Leitung des Teams verantwortlich. Alle Teammitglieder brachten ihre individuellen Fähigkeiten und Erfahrungen ein, um gemeinsam die STH-App zu entwickeln.

7.1.2 Personalaufwand

Der Personalaufwand wurde in Personentagen gemessen. Da das Projekt über einen Zeitraum von vier Monaten lief, wurde der wöchentliche Aufwand pro Teammitglied auf etwa 10 Stunden geschätzt, was insgesamt 40 Stunden pro Woche für das gesamte Team ergibt. Über die Projektlaufzeit hinweg summiert sich dies auf etwa 2.560 Stunden (40 Stunden/Woche $\times$ 15 Wochen).

7.1.3 Personalkosten

- **Jonas Waigel**
  - **Stundensatz:** 120,00 €
  - **Gesamtstunden:** 280 Stunden
  - **Zeitraum:** 08.03.2024 - 18.06.2024
  - **Gesamtkosten:** 33.600,00 €
- **Fitim Makolli**
  - **Stundensatz:** 110,00 €
  - **Gesamtstunden:** 350 Stunden
  - **Zeitraum:** 08.03.2024 - 18.06.2024
  - **Gesamtkosten:** 38.500,00 €

- **Hasan Deveci**
  - **Stundensatz:** 130,00 €
  - **Gesamtstunden:** 210 Stunden
  - **Zeitraum:** 08.03.2024 - 18.06.2024
  - **Gesamtkosten:** 27.300,00 €
- **Vatsegkan Zournatsidis**
  - **Stundensatz:** 150,00 €
  - **Gesamtstunden:** 420 Stunden
  - **Zeitraum:** 08.03.2024 - 18.06.2024
  - **Gesamtkosten:** 63.000,00 €

Gesamtkosten: 162.400,00 €

7.2 Ressourcen

7.2.1 Ressourcenliste

Die materiellen Ressourcen, die für die Entwicklung der STH-App erforderlich waren, umfassten:

- Hardware: Laptops/Computer für jedes Teammitglied
- Software: Entwicklungsumgebungen (Visual Studio Code, xCode und Android Studio), Versionskontrollsystem (GitHub)
- Plattformen: Firebase für das Backend, Flutter für die plattformübergreifende App-Entwicklung
- Kommunikationstools: Microsoft Teams für die teaminterne Kommunikation und für regelmäßige Meetings und Stand-ups
- Testgeräte: Smartphones (iOS und Android) und Simulatoren / Emulatoren von xCode und Android Studio für Testzwecke

7.2.2 Ressourcenkosten

- Hardware: 4 Laptops à 1200 € = 4800 €
- Software/ Entwicklungsumgebungen: Kostenfrei (durch Bildungsprogramme oder vorhandene Lizenzen)
- Plattformen: Firebase (kostenlose Nutzung innerhalb von bestimmten Grenzen), Flutter (kostenlos)
- Kommunikationstools: Microsoft Teams (kostenlose Nutzung für Bildungseinrichtungen)

- Testgeräte: Nutzung vorhandener Smartphones und kostenloser Simulatoren/Emulatoren

Insgesamt wurden alle benötigten Ressourcen effizient genutzt, um die Entwicklung des MVP der STH-App ohne zusätzliche finanzielle Belastungen zu realisieren. Dies ermöglichte es dem Team, sich vollständig auf die technischen, funktionalen und prozessualen Aspekte der App-Entwicklung sowie auf die Methoden des Projektmanagements zu konzentrieren und den Prototypen innerhalb des vorgegebenen Zeitrahmens erfolgreich umzusetzen.

8 Projektdurchführung und Dokumentation

Im Folgenden wird die Durchführung von der initialen Einrichtung bis hin zu der Erstellung des vollständigen Prototypen mit allen Ansichten und Funktionen genauer erläutert.

8.1 Initiale Einrichtung der Tools für das STH-Projekt

Dieses Kapitel beschreibt die initiale Einrichtung des Flutter-Code-Repositories, die Nutzung von Google Firebase und die Verwendung vorhandener Microsoft Teams Add-On Tools für die Projektorganisation.

8.1.1 Microsoft Teams Tools für die Organisation des Projektes und der Aufgaben für das Projektteam

Im Projektstart-Meeting hat das Projektteam zusätzlich zur Ideenfindung und Projektteam-Aufteilung entschieden, dass das STH-Projekt mit agilen Projektmanagement-Methoden angelehnt an Scrum durchgeführt werden soll. Zunächst wurde dafür ein Tool gesucht, das sowohl die Dokumentation von besprochenen Themen als auch die Planung von Aufgaben und die Kommunikation im Projektteam ermöglicht. Nach einigen Recherechen und Diskussionen hat sich das Projektteam für Microsoft Teams entschieden. Teams bietet mit seinen Add-On Tools die benötigten Möglichkeiten:
Ergebnisse aus Diskussionen und Weekly-Standup-Meetings können mit Microsoft OneNote dokumentiert werden. Aufgaben für das Projektteam können mit dem Add-On Microsoft Planner in einem Kanban Board übersichtlich dargestellt werden und der jeweilige Aufgabenstatus kann von jedem Projektmitglied eingesehen und bearbeitet werden. Zusätzlich wurde ein Team in Microsoft Teams erstellt, in dem die Kommunikation im Projektteam stattgefunden hat und Neuerungen oder Probleme bei der Entwicklung besprochen werden konnten.

8.1.2 Einrichtung des Flutter-Code-Repositories für das STH-Projekt

Im nächsten Schritt hat das Projektteam im Projektstart-Meeting ein Framework bzwėine Programmiersprache für das STH-Projekt gesucht, die sowohl eine gute Dokumentation, eine leicht verständliche Programmiersprache, Plattformunabhängigkeit für iOS/Android

und die Kompatibilität zum Kompilieren der Applikation für die Betriebssysteme Windows und MacOS bietet. Durch bereits vorhandenes Know-how wurde das Google-Framework Flutter gewählt.
Flutter ist ein Open-Source-Framework von Google, das eine Frontend-Entwicklung von Benutzeroberflächen für Apps bietet. Dabei gilt das Prinzip One-Codebase, welches bedeutet, dass Anwendungen für die unterschiedlichen Betriebssysteme iOS und Android aus einem Sourcecode kompiliert werden.
Für die Einrichtung des Repositories wird zunächst das Github Projekt "anwendungsprojekt_fom" erstellt und das Projektteam freigegeben. Im nächsten Schritt wurde auf den Entwicklungsrechnern der Projektmitglieder die Entwicklungsumgebung Visual Studio Code für die Code-Entwicklung und das Framework Flutter der Version 3.19.3 installiert und eingerichtet. "Flutter" bietet außerdem die Möglichkeit mithilfe des "flutter create" Terminal Commands ein initiales Flutter-Projekt erstellen zu lassen. Dieses initiale Setup beinhaltet eine voll funktionsfähige Beispiel-App, die für den ersten Test des Setups erfolgreich für Android und iOS kompiliert werden konnte. Für einen erfolgreichen iOS-Build wird zudem ein Apple-Gerät mit dem Betriebssystem macOS und die Apple-Software xCode benötigt, um iOS-spezifische Einheiten der App kompilieren zu können.
Für einen sogenannten "Clean Code", der für alle Projektmitglieder gut lesbar und verständlich ist, wurde festgelegt, dass für jede Änderung im Code der "Flutter Formatter (Terminal Command: dart format -l 120)und der Dart Analyzer (Terminal Command: dart analyze && dart fix –apply)" ausgeführt werden muss.

8.1.3 Google Firebase Tools als Backend

Firebase ist ein Cloud Service, der von Google bereitgestellt wird. Dieser bietet folgende wichtige Funktionen, die für die STH App essenziell sind:

- Der Cloud Firestore ist eine Cloud-Datenbank, in der verschiedene Werte von Variablen gespeichert werden können.
- Der Cloud Storage ist ein Cloud-Speicher, in dem Dateien unterschiedlicher Dateiformate hinterlegt werden können.
- Die Chat- und Kommunikationsfähigkeit

Über den TerminalCommand "flutter pub add" inclden jeweiligen Flutter packages/ dependencies $firebase\_auth, firebase\_core, firebase\_storage, cloud\_firestore$ wurden Flutter Packages dem Projekt hinzugefügt und können verwendet werden.

8.1.4 Zentrale Funktionen für Firebase, Shared Preferences, Flutter Page Routing, Flutter Bottom Navigationbar und Flutter App Bar

Zusätzlich wurden Funktionen, die an anderen Stellen des Codes wiederverwendet werden, zentralisiert. Dazu wurde im Repository ein Ordner "technical" angelegt und dieser in die jeweilige Grundfunktionalitäten unterteilt:

Firebase

- Authentifizierungsmethoden (Login / Register)
- Datei-Upload und -Download Funktionen
- Upload und Download von Variablenwerten

Shared Preferences

- Lokale Speicherung von Daten nach Download aus Firebase
- Update Funktion von editierten Daten

Custom Page Route

- Einmalige zentrale Konfiguration des Routing-Verhaltens zwischen Screens
- Flutter PageRouteBuilder inclġenerateRoute Funktion und neuer Screen als Übergabeparameter

Custom App Bar

- Einmalige zentrale Konfiguration der Flutter App Bar für ein einheitliches Design
- Boolean-Übergabeparameter für die Anzeige von Buttons/Icons
- String-Übergabeparameter für den Seitentitel

Custom Bottom Navigationbar

- Einmalige zentrale Konfiguration der Flutter Bottom Navigationbar für ein einheitliches Design
- Hervorhebung der aktuell ausgewählten Seite in der App
- Routing bei Auswahl eines Buttons in der Navigationbar

8.2 Loading-Screen

Der Loading-Screen ist ein wesentlicher Bestandteil der User Experience unserer STH-App. Der Loading-Screen ist der erste Bildschirm, der beim Starten der STH-App angezeigt wird. Dabei werden alle benötigten Ressourcen vorbereitet werden, die von der STH-App benötigt werden.

8.2.1 Gestaltung des Logos, die Entwicklung des Loading-Screens

Für die Gestaltung des Logos wurden mehrere Schritte durchgeführt. Das Logo wurde mithilfe von Canva erstellt und soll den Charakter sowie die Zielgruppe der STH-App widerspiegeln. Die Inspiration für das Logo wurde aus verschiedenen Quellen wie der NFL gewonnen. Die Farben und das Design sollen sportlich und ansprechend sein. Das Logo wurde in den Loading-Screen eingebaut und mit einem animierten Loading-Spinner ergänzt, der den initialen Start der STH-App veranschaulicht. Es wurde besonders darauf geachtet, dass der Nutzer nach genau vier Sekunden automatisch zur Startseite navigiert wird. Diese präzise Funktionalität ist entscheidend, da gleichzeitig im Hintergrund Ressourcen geladen werden, was für den Start der App notwendig ist. Das App-Logo wird mithilfe des Flutter-Dependencies flutter_launcher_icons implementiert:

```
flutter_launcher_icons:
  android: "launcher_icon"
  ios: true
  image_path: "assets/Images/FinalLogoSTHOriginal.png"
  min_sdk_android: 21
```

8.3 Home-Screen

Der Home-Screen ist der erste und damit wichtigste Anzeigebildschirm für die STH-App. Dieser wird direkt nach dem App-Start angezeigt und ist somit sofort sichtbar.

8.3.1 Anforderungen an den Home-Screen

Für die Funktionalitäten auf dem Home-Screen wurden zunächst vergleichbare Apps wie Instagram verglichen und die wichtigsten Funktionen ausgearbeitet.
Folgende Anforderungen wurden dabei erstellt:

- Profile/ Posts von Sportlern anzeigen
- Entry Point für die Chatfunktionalität

8.3.2 Erstellung der Flutter Widgets

Zunächst hat der Home-Screen die zentralen Elemente CustomAppBar und CustomNavigationBar erhalten. Im nächsten Schritt wurde das sogenannte Post-Widget erstellt, das im Home-Screen-Widget verwendet wird. Grund dafür ist, dass die unterschiedlichen Posts der Sportler vom Grund-Designkonzept gleich aufgebaut sein sollen. Die Posts sollen sich nur durch die inhaltlichen/ persönlichen Informationen unterscheiden.
Aufgebaut ist ein Post mit einem Profilbild links oben, dem Profilnamen, den Bildern des Sportlerposts und einem Chat-Button, mit dem der Manager direkt zum jeweiligen Chat mit dem Sportler springt.

8.3.3 Daten aus Firebase für den Home-Screen

Die Informationen (Profilname) und Dateien (Bilder und Profilbild) für die Sportlerposts auf dem Home-Screen werden aus Firebase geladen, indem die User ID der jeweiligen Sportler übergeben wird. Aufgrund von Beispieldaten gibt es hier datenschutzrechtlich keine Einschränkungen, für einen möglichen Rollout der Funktionen in der App werden allerdings im Ausblick noch einige Verbesserungen/ Möglichkeiten angesprochen.

8.4 Search-Screen

Der Search-Screen bietet die Möglichkeit, dass Nutzer, Sportler und Sportmanager gezielt nach bestimmten Personen anhand von Hashtags oder Namen zu suchen. Diese Funktion ist eng mit Firebase verknüpft, da dort die Profildaten der gesuchten Nutzer

gespeichert sind. Sobald ein Suchbegriff eingegeben wird, sei es ein Hashtag oder ein Name, zeigt der Search-Screen die entsprechenden Personen an. Die Suchfunktion verwendet Pattern-Matching, um präzise Ergebnisse zu liefern, basierend auf den eingegebenen Suchkriterien. Die Ergebnisse der Suche werden dynamisch von Firebase abgerufen und umfassen alle relevanten Werte für die gefundenen Benutzer. Jedes Suchergebnis wird in einem strukturierten Layout präsentiert, das neben grundlegenden Informationen auch Avatare der entsprechenden Benutzer zeigt. Dabei werden sowohl die personenbezogenen Informationen und das Profilbild für einen User in ein Array gespeichert und dem User im Suchergebnis zugeordnet.

8.5 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 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 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.

8.6 Implementierung des Chat-Screens

Zu jeder modernen Community Applikation gehört auch die Chatfunktionalität dazu. Der Austausch zwischen zwei oder mehreren Menschen ist besonders in der STH-App wichtig, um die Kommunikation zwischen Spielern und Managern zu ermöglichen. Die Anforderungen an den Chat-Screen standen schnell fest. Die Funktionalität muss mindestens den gleichen Interaktionsumfang ermöglichen, wie vergleichbare Apps wie z.B. Instagram, Snapchat oder X (ehemalig Twitter).

8.6.1 Anforderungen an den Chat-Screen

Die spezifischen Anforderungen für die Chat-Funktion in der STH-App wurden im Projektteam gemeinsam ausgearbeitet. Dabei wurden zunächst vergleichbare Applikationen genauer beleuchtet und die Mindestanforderungen anhand der dort gesichteten Funktionen definiert. Das Chatten in der STH-App muss mittels verschlüsselte eins zu eins Verbindungen funktionieren, in der zwei Menschen miteinander in einem Chat-Raum kommunizieren können. Zudem muss es auch die Möglichkeit geben Gruppenchats zu erstellen und mehrere Menschen dazu einzuladen. Auch hierbei müssen die Verbindungen stets verschlüsselt sein. Innerhalb eines Chats muss jeder Nutzer Nachrichten senden und empfangen können. Zu den Nachrichten gehören Textnachrichten, GIFs, Bilder und Videos. Jeder Nutzer muss gesendete und empfangene Nachrichten im Chat sehen können und die Chats müssen mit ihrem kompletten Verlauf beim ein- und ausloggen erhalten und neu geladen werden können.

8.6.2 Erstellung einer Flutter Widget-Seite

Nachdem die Anforderungen definiert wurden, kann nun die Erstellung der Chat-Screen Widget-Seite beginnen. Diese Seiten dienen als Vorbereitung dazu, Widgets platzieren und anzeigen zu können. Das Erstellen einer derartigen Seite kann in Flutter durch das Erstellen einer Klasse realisiert werden. Hierbei wird für die Klasse ein kontextbezogener Name eingesetzt und die Art der Widget-Seite definiert. Dabei wird zwischen Stateful

und Stateless Widgets unterschieden. Bei den Stateful Widget Klassen können sich deren Komponenten verändern. Die Stateless Widgets verändern sich nicht und können sich nicht in der Ansicht aktualisieren. Für den Chat-Screen ist das Stateful Klassenwidget die richtige Wahl, da sich die Anzahl und Inhalte der angezeigten Chats stetig verändern kann.

8.6.3 Integration des Stream-Chat-Flutter Pakets

Die Implementierung einer Chat-Funktion kommt in Social-Media Apps sehr häufig bzw. fast immer vor. Deshalb gibt es hierzu eine große Anzahl an Flutter Paketen, welche sich unter der Seite pub.dev genauer angeschaut werden können. Ein sehr mächtiges und umfängliches Paket nennt sich hierbei Stream-Chat-Flutter. Dieses Paket bringt eine Anzahl an vorbereiteten Widgets für die Anzeige und Erstellung von Chats mit sich. Darin beinhaltet sind App-Bars, Channel-Screens, Chat-Screens und Chatfunktionen. Die Einbindung dieses Pakets erfordert das Verändern der gesamten Struktur der Anwendung. Die Chat-Funktion stellt einen Hauptbestandteil der Anwendung dar und muss auf den Top of Widget-Tree implementiert werden. Das bedeutet, dass das Paket Stream-Chat-Flutter als Erstes in der Anwendung geladen werden muss, um es korrekt auszuführen. Nachdem die Struktur und das Laden der Pakete in der Anwendung angepasst wird, müssen Funktionen implementiert werden, die das Abrufen von Chats aus dem Stream-Chat-Client ermöglichen. Dabei greift die STH-App auf das separate Backend des Paketes zu und authentifiziert den aktuellen Nutzer der Applikation. Nach erfolgreicher Authentifizierung werden die zugehörigen Chatdaten und Channels abgerufen, um dem Nutzer alle Chatverläufe korrekt anzeigen zu können.

8.6.4 Benutzerauthentifizierung

Um die Benutzerauthentifizierung durch die Anwendung zu ermöglichen, müssen Zugangsschlüssel bzw. Tokens bereitstehen. Diese Tokens werden durch den Aufruf durch Firebase generiert und der jeweilige Nutzer der Anwendung registriert. Nach erfolgreicher Übermittlung der Tokens kann die STH-App nun sicherstellen, dass es sich um den angemeldeten Nutzer handelt und alle Chatverläufe inklusive persönliche Daten laden.

8.6.5 Implementierung der Chat-Widgets

Nach der erfolgreichen Benutzerauthentifizierung folgt der letzte Schritt. Die verschiedenen Chats des Nutzers und die Chatverläufe können mittels der Chat-Widgets aus dem Paket Stream-Chat-Flutter angezeigt werden. Das Paket beinhaltet einen vollständigen Channel-Screens. Das ist das Widget, welches zuerst beim Einstieg in die Chat-Funktionen angezeigt wird. Analog zu anderen bekannten Applikationen wird hierbei eine Übersicht über alle aktiven Chats des Nutzers angezeigt. Daraufhin folgt der Channel-Screen. Dieser stellt den einzelnen Chat, auf dem vom Channel-Screen aus geklickt wird, dar. Im Chat-Screen ist der gesamte Chatverlauf aus diesem Chat des Nutzers zu sehen. Zudem kann hier durch das Textfeld am unteren Ende der Seite neue Nachrichten durch den Nutzer eingegeben werden und an den Chatpartner versendet werden.

Die Implementierung des Channel-Screens erfolgt als Erstes. Hierfür wird eine eigene Klasse erstellt und das Chat-Screen Widget aus dem Paket wird auf dieser Seite platziert. Nun muss noch das richtige Routing definiert werden. Das stellt sicher, dass die verschiedenen Chats geladen werden können. Die Liste der verfügbaren Chats wird geladen und am oberen Ende der Seite wird die App-Bar mit dem Namen Chats angezeigt.
Nach der Integration des Channel-Screen-Widgets kann nun das Chat-Screen-Widget folgen. Für das Chat-Screen-Widget wird ebenso eine neue Klasse und Seite erstellt, worauf das Widget platziert werden kann. Nach der Platzierung wird durch das korrekte Routing sichergestellt, dass der richtige Chat vom angemeldeten User geöffnet wird. Durch die Integration des Paketes sind alle Funktionen automatisch verfügbar. Das Versenden und Empfangen von Nachrichten ist bereits möglich und das Reagieren auf Nachrichten ebenso. Zu normalen Textnachrichten können auch Bilder, GIFs, Videos und Emojis versendet werden. Die Grundfunktionalitäten einer ausgereiften Social-Media App stehen nun mit der Chat-Funktion.
Nun ist der letzte Schritt die Anwendung auf die korrekte Wegweisung bzw. Routing anzulernen. Dabei wird in der main.dart, welches die Hauptdatei darstellt und dabei auch die Strukturen für die gesamte Anwendung festlegt, das Routing eingearbeitet. Die neuen Screens müssen den gesamten Routing Optionen der Anwendung hinzugefügt werden. Die dabei verwendeten Klassen werden referenziert. Innerhalb der Datei CustomApp-Bar wird nun für den Chat-Button das korrekte Routing hinterlegt. Dabei wird auch festgelegt, in welchen Bildschirmen der Chat-Button sichtbar ist und in welchen nicht. Innerhalb des Chat-Screens verschwindet der Chat-Button, da der Anwender sich bereits in der Chatfunktionalität aufhält. In allen anderen Ansichten ist der Button sichtbar und das Routing erfolgt beim Auslösen. Nun ist der Channel-Screen und der Chat-Screen in der Anwendung vollständig mit korrektem Routing implementiert.

9 Fazit

Rückblickend auf das Projekt STH-App kann man sagen, dass die Ziele, die zu Beginn des Projekts festgelegt wurden, erfüllt wurden. Jetzt bietet die STH-App eine neue Möglichkeit, Sportmanager und Sportler über die Chat-Funktion in Kontakt zu bringen. Darüber hinaus sorgt das verwendete Framework Flutter für eine plattformunabhängige Interaktion zwischen Android- und Apple iOS-Geräten sowie für regelmäßige Aktualisierung. Das Design ist speziell für die STH-App konzipiert und zeichnet sich durch aussagekräftige Icons durch Konsistenz und eine gute Benutzererfahrung aus. Die Einbindung der Google Firebase Tools und insbesondere die unmittelbare Datenübertragung wurden als erfolgreich angesehen.
Der Einsatz moderner Tools und die sehr enge Zusammenarbeit der Teammitglieder waren die entscheidenden Faktoren für den aktuellen Stand der Anwendung. Der Stand des Projekts wurde wesentlich durch die wöchentlichen strukturierten Besprechungen der aktuellen Fortschritte, der aufgetretenen Probleme bei den Teammitgliedern und der weiteren Planung erreicht. Im Verlauf des Projekts kamen bei den Tests der App häufig Fehler auf und die Teammitglieder schlugen Verbesserungsvorschläge vor. Dennoch wurden alle gesetzten Meilensteine erreicht, obwohl es anfangs technische Probleme beim Einrichten der Entwicklungsumgebungen auf den verschiedenen Betriebssystemen als auch Kompilierfehler aufgrund von Änderungen am Sourcecode und Entwicklungsverzögerungen gab. Auf lange Sicht bietet die STH-App aufgrund ihrer unkomplizierten und zuverlässigen Bauweise sowie ihrer Benutzerfreundlichkeit ein beträchtliches Potenzial auf unterschiedlichen Märkten.
Wir konnten im Rahmen des Projekts wichtige Erfahrungen sammeln, vor allem im Hinblick auf die agile Arbeitsweise und die Entwicklung von Anwendungen. Auch das offene Feedback und die fortlaufende Verbesserung auf der Grundlage der App-Testergebnisse waren entscheidende Erfolgsfaktoren.

10 Ausblick

Um die STH-App weiterzuentwickeln und die Benutzererfahrung zu verbessern, sind folgende Erweiterungen und Verbesserungen vom Projektteam geplant.

- Verbesserung des Search-Screen-Algorithmus: Die Verbesserung des Suchalgorithmus führt zu einer Steigerung der Effizienz und Präzision der Suchergebnisse, was es den Nutzern ermöglicht, die gewünschten Informationen schneller und genauer zu finden.
- In-App-Linkouts einrichten, um direkt auf einen anderen Bildschirm zu wechseln: Mit dieser Funktion können Benutzer direkt zwischen verschiedenen Bildschirmen wechseln. Dies trägt zur Verbesserung des Navigierens in der Anwendung bei.
- Integration von KI-gestützten Funktionen: Die Verwendung von künstlicher Intelligenz ermöglicht es, auf die STH-App angepasste Empfehlungen und automatisierte Abläufe zu generieren, die die Benutzererfahrung noch individueller und effizienter machen. Zum Beispiel können KI-Funktionen auf dem Home-Screen oder Search-Screen dazu beitragen, Sportler oder Sportmanager zu finden.
- Lokalisierung und Übersetzungen von Texten in der App: Durch die Lokalisierung und Übersetzung der Inhalte der App in unterschiedliche Sprachen wird die Verbreitungsfähigkeit der STH-App vergrößert und die Benutzerfreundlichkeit für internationale Anwender gesteigert.
- Die Einführung von Personalisierungsmöglichkeiten: Es wird den Nutzern ermöglicht, die App individuell anzupassen, was zu einer Steigerung ihres Engagements und ihrer Zufriedenheit führt.

Darüber hinaus soll der Source-Code der STH-App mit technischen Versionsupdates des Frameworks Flutter und die dazugehörigen Dependencies aktualisiert werden sowie die Richtlinien von Apple und Google ebenfalls berücksichtigt werden. Dadurch wird sichergestellt, dass die Anwendung stets auf dem neuesten Stand der Technik bleibt und eine optimale Leistung erbringt. Andere bedeutende Aspekte sind die Leistung der STH-App auf den Nutzergeräten sowie Datenschutz und Sicherheit. Das Projektteam wird in den nächsten Monaten vermehrt auf Marktanalysen und Innovationsprozesse setzen, um die App fortlaufend zu verbessern und das Nutzererlebnis zu verbessern. Dabei entsteht eine ganzheitliche Teststrategie, die unterschiedliche Testverfahren wie User Interface Tests, A/B-Tests, manuelle Tests, Testautomatisierung, Unit-Tests und Crowdtests beinhaltet. Dies geschieht, um die Wünsche und Rückmeldungen der Nutzer gezielt zu erfassen und eine Vielzahl von Nutzern zu berücksichtigen. Auf lange Sicht plant das Projektteam, die STH-App mit Wartung und technischer Unterstützung zu veröffentlichen. Um fortlaufendes Feedback zu bekommen und regelmäßig Fehlerbehebungen sowie Verbesserungen

durch App-Updates durchzuführen, soll eine Community oder ein Forum entwickelt werden.
Zusammenfassend kann festgehalten werden, dass dieses Projekt in Zukunft viele Chancen und Möglichkeiten für den Erfolg der STH-App mit sich bringt. Die STH-App wird ihre Position als innovative und nützliche App für Sportmanager und Sportler weiter festigen und ausbauen.

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 4. Juli 2024

Eigenhändige Unterschrift