HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Floorball Tactics-Board

Bachelorarbeit

Betreuer: Mirko Stocker
Experte: Leo Büttiker
Gegenleser: Prof. Dr. Farhad Mehta

Autoren: Luca Aquino, Silvan Habegger

Abstract

Im Unihockey-Sport kommen oft Taktik-Tafeln zur Visualisierung und Absprache von Taktiken und Übungen zum Einsatz. Während dies ein bewährtes Mittel des Informationsaustasches ist, sind die fertig aufgezeichneten Spielzüge trotz standardisierter Symbolsprache ohne weiterführende Beschreibung für Dritt-Personen nur schwer interpretiertbar, da eine zeitliche Dimension nicht abgebildet werden kann. Zudem ist ein Austausch von Taktiken meist nur durch einfaches Abzeichnen, Kopieren oder mit dem Zukauf von Lehrbüchern möglich. Eine zentrale Austauschplattform gibt es nicht.

Die Umsetzung wurde in einem an RUP angelehnten Projektvorgehen vorgenommen, in deren Rahmen agile Methoden zum Zuge kamen.

Die Android-Applikation wurde in Java 7 implementiert und in eine dreischichtige Model-View-Presenter Architektur aufgeteilt. Die Views sowie die Navigation halten sich stark an Googles Material-Design-Guidelines und sind möglichst benutzerfreundlich konzipiert. Um Taktiken und Gruppen lokal zu speichern, wurde greenDao ORM für SQLite Datenbanken verwendet. Diese Library gilt als schnellster und bekanntester OR-Mapper in seinem Segment. Als Datenaustausch-Format kommt JSON zum Einsatz.

Für die Server-Applikation wurde auf Scala in Zusammenspiel mit MongoDB gesetzt. Scala ist durch die funktionalen Sprachkonzepte ebenso wie MongoDB als Document-Oriented-Database sehr gut skalierbar. Die Administrations-Web-Oberfläche wurde in HTML, CSS und JavaScript implementiert, und bezieht Daten über dieselbe JSON-Schnittstelle wie die Android-Applikation.

In dieser Arbeit wurde eine App entwickelt, die es Autoren einer Taktik ermöglicht, mit geringem Aufwand Spielzüge in gewohnter Notation zu erfassen, sowie Spieler besonders einfach gleichzeitig laufen zu lassen. Erstellte Übungen können Gruppen zugeordnet werden, womit dem Benutzer eine strukturierte Organisationsmöglichkeit angeboten wird.

Der Austausch von Taktiken wird durch einen “Store“ ermöglicht, in welchem Taktiken und Gruppen als kostenlose Downloads angeboten werden. Dieser soll einerseits die Mannschafts-interne Kommunikation verbessern, wie auch die Trainer zu neuen Ideen für Übungen und Taktiken anregen. Um den Store verwalten zu können, wurde ein Administrationstool in Form einer Webanwendung implementiert.

Management Summary

Ausgangslage

Im Unihockey-Sport kommen oft Taktik-Tafeln zum Einsatz. Das sind im wesentlichen Whiteboards mit einem Spielfeld als Hintergrund, auf denen gezeichnet werden kann und Magnete als Spieler dienen. Solche Taktik-Tafeln werden benutzt, um im Training eine Übung zu visualisieren oder den Spielern taktische Instruktionen für ein Spiel zu vermitteln.

Um sich als Trainer solche Taktiken und Übungen (im weiteren Verlauf als "Taktik" zusammengefasst) festzuhalten und eine Sammlung zu erstellen, muss entweder die Taktik zu Papier gebracht oder mit rudimentären elektronischen Tools wie Word, Powerpoint oder Exercisedrawer erfasst werden. Die Übungen sind in beiden Fällen statisch festgehalten und eine genaue Beschreibung des Ablaufs ist zwingend notwendig - insbesondere, wenn zeitlich parallele Abläufe vorhanden sind. Trotz solcher Beschreibungen nimmt die Interpretation dieser Taktiken vorallem bei höherer Komplexität viel Zeit in Anspruch.

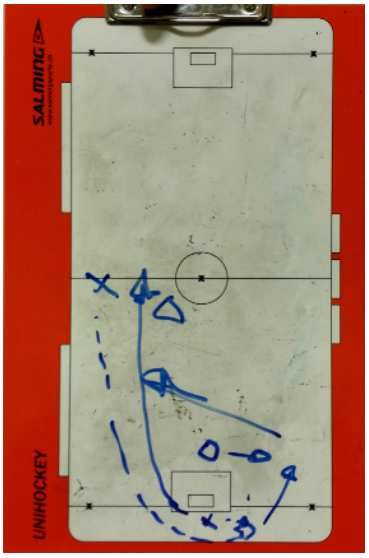


Abbildung 1: Taktik-Tafel

Einen Austausch erfasster Taktiken zwischen Trainern findet nur unter Bekannten oder in Kursen statt. Vereinzelt sind PDFs im Internet zu finden oder gewisse, auf Unihockey spezialisierte, Shops vertreiben kleine Sammlungen. Dadurch ist viel Know-How nicht zugänglich oder geht verloren. Auch der Austausch von Taktiken zwischen Trainer und Spieler erfolgt oft nur durch reines Aufzeichnen im Training oder durch Verteilen einer Zusammenstellung auf Papier. Beide Fälle sind nicht optimal, da ein Spieler die Übung eventuell vor oder nach dem Training nochmals anschauen möchte und falls er diese auf Papier hat, diese auch verstehen muss. Der zeitliche Aufwand ist somit falsch verteilt, da ein Trainer zum Aufzeichnen wenig Aufwand benötigt, während das Interpretieren der Zeichnung für die Spieler viel Zeit in Anspruch nimmt.

Vorgehen, Technologien

Das Projekt wurde in einem an RUP angelehnten Vorgehen durchgeführt, in dem die Planung der einzelnen Tasks in einwöchigen Iterationen vorgenommen wurde. Im Verlaufe des Projekts wurden die eingeplanten Tasks jeweils dynamisch an die Bedürfnisse und verändernden Anforderungen angepasst. Die Anforderungen wurden vom Trainer der UHC-Lokomotive-Stäfa geprüft und seine Visionen und Inputs aufgenommen.

Die Android-Applikation ist in Java 7 (Standard für Entwicklung eines Android Apps) implementiert und in eine dreischichtige, gut wartbare Architektur aufgeteilt. Die Views sowie die Navigation halten sich stark an Google's Material-Design-Guidelines und sind möglichst intuitiv konzipiert. Um Taktiken und Gruppen lokal zu speichern, wurde greenDao ORM für SQLite Datenbanken verwendet. Diese Library gilt als schnellster und bekanntester OR-Mapper in seinem Segment. Als Datenaustausch-Format kommt JSON zum Einsatz.

Für die Server-Applikation wurde auf Scala in Zusammenspiel mit MongoDB gesetzt. Scala ist durch die funktionalen Sprachkonzepte ebenso wie MongoDB als Document-Oriented-Database sehr gut skalierbar.

Die Administrations-Web-Oberfläche wurde in HTML, CSS und JavaScript implementiert, und bezieht Daten über dieselbe JSON-Schnittstelle wie die Android-Applikation.

Ergebnisse

Es wurde in dieser Arbeit ein App entwickelt, die es Trainern ermöglicht, Taktiken und Übungen zu erfassen, diese zu organisieren und in einer lokalen Sammlung übersichtlich abzulegen. Es ist möglich, diese nach Belieben zu durchsuchen, zu filtern und zu ändern. Zusätzlich können den Taktiken Namen und Beschreibungen gegeben werden, um diese einfach zu identifizieren oder Variationen und Gedanken festzuhalten.

Das Aufzeichnen von Taktiken wurde möglichst intuitiv und einfach gehalten, ohne Kompromisse im Umfang der Features einzugehen. Es können Spieler hinzugefügt und deren Laufwege eingezeichnet werden.
Bälle werden Spielern zugewiesen. Befindet sich ein Spieler in Ballbesitz, kann er passen und schiessen. Für das Erfassen zeitlich paralleler Spielzüge wurde ein besonders neuartiges und benutzerfreundliches Konzept erarbeitet, das bisher definierte Aktionen während dem Einzeichnen der neuen Bewegung wiedergibt.

Abbildung 2: Resultat Tactics-Board Android Applikation

Ausblick

Die App steht auf Google-Play gratis als Download zur Verfügung und soll stetig weiterentwickelt werden.

Der Online-Store bietet grosses Potential. So soll später das einfach gehaltene Laden von Einträgen verbessert und performanter gemacht werden. Es soll auch möglich werden, für Taktiken Geld zu verlangen, um so eine Zusammenarbeit mit National-Verbänden oder lokalen Unihockey-Läden zu fördern (da diese ihre Taktiken / Übungen kommerziell vertreiben).

Weiteres Potential für die Applikation besteht darin, das Taktik-Board auf andere Sportarten auszuweiten. Eine solche Portierung ist problemlos auf alle Mannschaftssportarten mit einem definiertem Spielfeld möglich. Dadurch würde der international eher kleine Markt von Unihockey Spielern und Trainern in einen potentiell weitaus grösseren transformiert werden.

Ebenfalls ausbaufähig ist das Konzept der relativ starren Gruppen. Diese können, wie vom Trainer der UHC-Lokomotive-Stäfa vorgeschlagen, durch die etwas dynamischeren “Tags“ abgebildet werden, was eine mehrfache Zuordnung zu Gruppen ermöglicht und so Planung und Gestaltung von Trainings besser unterstützt.

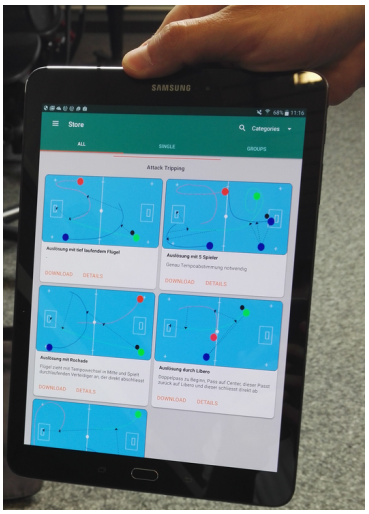


Abbildung 3: Online-Store der Applikation

Eigenständigkeitserklärung

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben haben.
- dass wir keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt haben.

Rapperswil, 17.06.2016:

Luca Aquino

Silvan Habegger

Aufgabenstellung

Aufgabenstellung Bachelorarbeit Luca Aquino und Silvan Habegger „Unihockey Taktik App“

1. Betreuer und Experte

Diese Bachelorarbeit ist ein eigenes Thema der Studierenden und wird von Mirko Stocker, HSR, IFS, m1stocke@hsr.ch betreut.

Experte:

- [zu bestimmen]

2. Studierender

Diese Arbeit wird als Bachelorarbeit an der Abteilung Informatik durchgeführt von

- Luca Aquino, laquino@hsr.ch
- Silvan Habegger, shabegge@hsr.ch

3. Einführung

Taktiktafeln werden in den unterschiedlichsten Teamsportarten für das Training und die Analyse eingesetzt. Typischerweise werden diese ad hoc auf einem Whiteboard gezeichnet. Silvan Habegger spielt selber Unihockey und würde die alten Taktik-Tafeln gerne durch eine modernere Alternative ersetzen, die zusätzliche Funktionen bietet, jedoch genauso einfach zu bedienen ist und optisch ansprechend aussieht.

4. Ziele der Arbeit

Das Ziel dieser Arbeit ist die Erstellung einer App, mit der Tatktiktafeln gezeichnet werden können. Durch Animationen der einzelnen Schritte soll das Verständnis gefördert werden. Die gespeicherten Tafeln sollen über einen Server mit anderen Benutzern ausgetauscht werden können.

Die App sollte in einer modernen Android (5+) Version entwickelt und einfach erweiterbar sein. Zwei Schwerpunkte liegen bei der Implementierung einer Oberfläche, welche den Material-Design-Prinzipien folgt, und bei der Implementation eines Multi-Language-Supports.

Taktiktafeln werden in unterschiedlichen Sportarten eingesetzt, diese Arbeit konzentriert sich aber auf Unihockey.

5. Aufgabenstellung

Die App umfasst die folgenden Hauptfunktionalitäten:

1. Eine Taktik-Tafel, auf welcher Schuss-, Pass und Laufwege eingezeichnet werden können
2. Spielzüge können Step-by-Step definiert und abgespielt werden
3. Eine Anzeige lokal gespeicherter Spielzüge
4. Eine Möglichkeit gespeicherte Tafeln mit anderen Trainern auszutauschen

Zusätzliche Funktionen, die je nach Zeitbudget umgesetzt werden könnten:

1. Ein Statistik Bereich, in welchem pro Spiel/Saison Spieler- und Spielstatistiken aufgezeichnet werden können
2. Statistik mit geblockten Schüssen erweitern
3. Statistik mit +/- Bilanz erweitern (Ganze Blöcke müssten auswählbar sein)
4. Druckversion der Statistik mittels PDF
5. Trainingspräsenz-Tracker

6. Zur Durchführung

Es finden wöchentliche Besprechungen mit dem Betreuer statt. Zusätzliche Besprechungen sind nach Bedarf durch die Studierenden zu veranlassen.

Alle Besprechungen, ausser der Kick-off Besprechung, sind von den Studierenden mit einer Traktandenliste vorzubereiten und zu leiten. Als erstes Traktandum soll immer der Stand des Projektes präsentiert werden (Was wurde gemacht? Was wurde erreicht? Wie viel Zeit wurde in was investiert?). Die Ergebnisse der Besprechungen sind durch die Studierenden zu protokollieren und an den Betreuer anschliessend zuzustellen.

Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. An Meilensteinen (oder auch Zwischenversionen) gemäss Projektplan sind einzelne Arbeitsresultate in vorläufigen Versionen abzugeben. Über die abgegebenen Arbeitsresultate erhalten die Studierenden ein vorläufiges Feedback. Eine definitive Beurteilung erfolgt auf Grund der am Abgabetermin abgelieferten Resultate.

7. Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen (siehe https://www.hsr.ch/Allgemeine-Infos-Diplom-Bach.4418.0.html). Die zu erstellenden Dokumente bzw. Berichtsteile sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollten den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Alle Resultate sind vollständig auf CD/DVD in 3 Exemplaren abzugeben.

8. Termine

Siehe auch Terminplan auf https://www.hsr.ch/Termine-Diplom-Bachelor-und.5142.0.html.

Montag, den 22. 2. 16	**Beginn der Bachelorarbeit**
8.6.16	Abgabe Kurzbeschreibung und A0-Poster an Betreuer zur Überprüfung Vorlagen stehen unter den allgemeinen Infos Diplom-, Bachelor- und Studienarbeiten zur Verfügung.
15.6.16, 10.00	**Fertigstellung des A0-Posters bis 10.00 Uhr und Weiterleitung als PDF-Dokument an das Studiengangsekretariat.**

17.6.16, 12.00	**Abgabe der Arbeit an den Betreuer bis 12.00 Uhr.**
17.6.16, 16.00	Präsentation der Bachelorarbeiten, 16 bis 20 Uhr
8.8. - 26.8.16	Mündliche Prüfung zur Bachelorarbeit
30.9.16, 9.00	Bachelorfeier und Ausstellung der Bachelorarbeiten

9. Beurteilung

Eine erfolgreiche Bachelorarbeit zählt 12 ECTS-Punkte pro Studierenden. Für 1 ECTS Punkt ist eine Arbeitsleistung von 30 Stunden budgetiert (Siehe auch Modulbeschreibung der Bachelorarbeit https://unterricht.hsr.ch/staticWeb/allModules/19419_M_BAI.html.).

Für die Beurteilung ist der verantwortliche Dozent zuständig.

Gesichtspunkt	Gewicht
1. Organisation, Durchführung	1/6
2. Berichte (Abstract, Mgmt Summary, technischer u. persönliche Berichte) sowie Gliederung, Darstellung, Sprache der gesamten Dokumentation	1/6
3. Inhalt*)	3/6
4. Mündliche Prüfung zur Bachelorarbeit	1/6

*) Die Unterteilung und Gewichtung von 3. Inhalt wird im Laufe dieser Arbeit präzisiert.

Im Übrigen gelten die Bestimmungen der Abteilung Informatik für Bachelorarbeiten.

Rapperswil, den 11. Februar 2016

Mirko Stocker

Dozent für Informatik
Institut für Software (IFS)
Hochschule für Technik Rapperswil

Inhaltsverzeichnis

1 Analyse **10**
- 1.1 Ausgangslage . . . 10
- 1.2 Idee & Mehrwert . . . 10
- 1.3 User-Stories . . . 10
- 1.4 Use Cases . . . 12
- 1.5 Use Cases “brief“ . . . 13
- 1.6 Use Cases “fully dressed“ . . . 16
- 1.7 Functional Requirements . . . 21
- 1.8 Non-Functional Requirements . . . 21
- 1.9 Domain Modell . . . 24
- 1.10 Tests . . . 25

2 Architektur **27**
- 2.1 Evaluation Presentation-Layer Pattern . . . 27
- 2.2 Evaluation Libraries . . . 30
- 2.3 Evaluation Technologien Backend . . . 31
- 2.4 Prozess-Interaktionen . . . 33
- 2.5 Architektur-Übersicht . . . 35

3 UI, Design & Usability **37**
- 3.1 Konzept . . . 37
- 3.2 Farbgebung . . . 37
- 3.3 Icons und Bildsprache . . . 38
- 3.4 Usability . . . 39
- 3.5 Navigationskonzept . . . 41
- 3.6 Wireframes . . . 44

4 Projekt-Management **53**
- 4.1 Meilensteine . . . 53
- 4.2 Zeitaufwand . . . 53
- 4.3 Risikomanagement . . . 54

5 Ergebnisse **55**
- 5.1 Zielerreichung . . . 55
- 5.2 Änderungen Navigationskonzept . . . 55
- 5.3 Abgedeckte Use Cases . . . 56
- 5.4 Zusätzlich implementierte Funktionen . . . 62
- 5.5 Weggelassene Use Cases . . . 63
- 5.6 Problemlösungen & Erkenntnisse . . . 64

6 Schlussfolgerungen **70**
- 6.1 Ergebnisdiskussion . . . 70
- 6.2 Resultate . . . 71
- 6.3 Ausblick . . . 71

7 Anhang **73**
A System-Tests . . . 73
B Schritt für Schritt Anleitung Google Sign-In and Keystore Hashes . . . 89
C Code-Beispiele . . . 90
D Semantisches Sprachmodell . . . 94
E Benutzerhandbuch . . . 95
F Feedback von Michael Kuhn, Trainer UHC Lokomotive Stäfa . . . 107
G Persönliche Berichte . . . 109

Kapitel 1

Analyse

Bemerkung: *"Taktik" steht im Kontext des Dokuments auch für einen Spielzug im Spiel oder einer Übung.*

1.1 Ausgangslage

Im Unihockey-Sport kommen oft Taktiktafeln zum Einsatz. Das sind im wesentlichen Whiteboards mit einem Spielfeld als Hintergrund, auf denen gezeichnet werden kann und Magnete als Spieler dienen. Solche Taktiktafeln werden benutzt, um im Training eine Übung zu visualisieren oder den Spielern eine taktische Instruktionen für ein Spiel zu vermitteln.

Der Austausch von Taktiken findet nur in kleinem Rahmen statt. Dies ist vorallem auf die limitierten Möglichkeiten und auf das Nichtvorhandensein zentraler Austauschplattformen zurückzuführen.

1.2 Idee & Mehrwert

Aktuell bedarf das Aufzeichnen einer Taktik weit weniger Zeit als das Interpretieren durch Drittpersonen. Während das Aufzeichnen aber nur einmal stattfindet, das Lesen und die Interpretation aber mehrfach, scheint der zeitliche Aufwand falsch verteilt.

Das Floorball Tactics-Board soll diesem Problem Abhilfe schaffen. Durch eine Applikation, mit der es möglich ist, Abläufe abzuspielen, soll der Aufwand für das Auffassen einer Übung deutlich verringert werden. Mit benutzerfreundlichen Verfahren für das Erfassen einer Taktik soll zudem der zusätzliche Zeitaufwand für Trainer und Autoren minimal gehalten werden.

Mit dem Online-Store wird Benutzern eine Möglichkeit geboten, Know-How mit anderen Spielern und Trainern auszutauschen.

1.3 User-Stories

Die User-Stories beschreiben Funktionswünsche und -ansprüche verschiedener Benutzer. Diese informellen Anregungen dienen als Leitlinien für die Ausarbeitung der Use Cases.

1.3.1 User-Story 1: Spielzug planen

Als Anwender möchte ich eine neue Taktik abbilden können, um diese mit meinem Team zu besprechen.

1.3.2 User-Story 2: Taktiken speichern

Als Anwender möchte ich meine erstellten Taktiken abspeichern können, damit ich diese zu einem späteren Zeitpunkt wieder ansehen kann.

1.3.3 User-Story 3: Taktik duplizieren

Als Anwender möchte ich eine bestehende Taktik duplizieren, um eine darauf basierende Taktik zu erstellen.

1.3.4 User-Story 4: Taktik abspielen

Als Anwender der Applikation möchte ich eine Taktik wie ein Video abspielen können, um den Spielfluss und die Zusammenhänge der Spielzüge besser zu erkennen.

1.3.5 User-Story 5: Meine Taktiken gruppieren

Als Anwender möchte ich meine bestehenden Taktiken gruppieren, um eine bessere Übersicht zu haben.

1.3.6 User-Story 6: Meine Taktiken durchsuchen

Als Anwender möchte ich nach einer bestehenden Taktik suchen, um schneller zu der von mir gewünschten Taktik zu gelangen.

1.3.7 User-Story 7: Meine eigenen Taktiken teilen

Als Anwender möchte ich meine Taktiken mit anderen Anwendern teilen, damit auch andere von meiner Taktik profitieren.

1.3.8 User-Story 8: Eine nicht von mir erstellte Taktik beziehen

Als Anwender möchte ich von anderen Anwendern erstellte Taktiken beziehen, damit ich von diesen profitieren kann.

1.4 Use Cases

Im Use Case Diagramm sind lediglich die komplexeren Use Cases abgebildet. Die verwendeten Actors repräsentieren jeweils eine typische Rolle eines Benutzers. Grundsätzlich sind aber alle Use Cases von jedem Actor ausführbar.

Alle der Übersichtlichkeit wegen ausgelassenen Use Cases haben keine weiteren Abhängigkeiten zu anderen Use Cases und sind ebenfalls jedem Actor zugänglich.

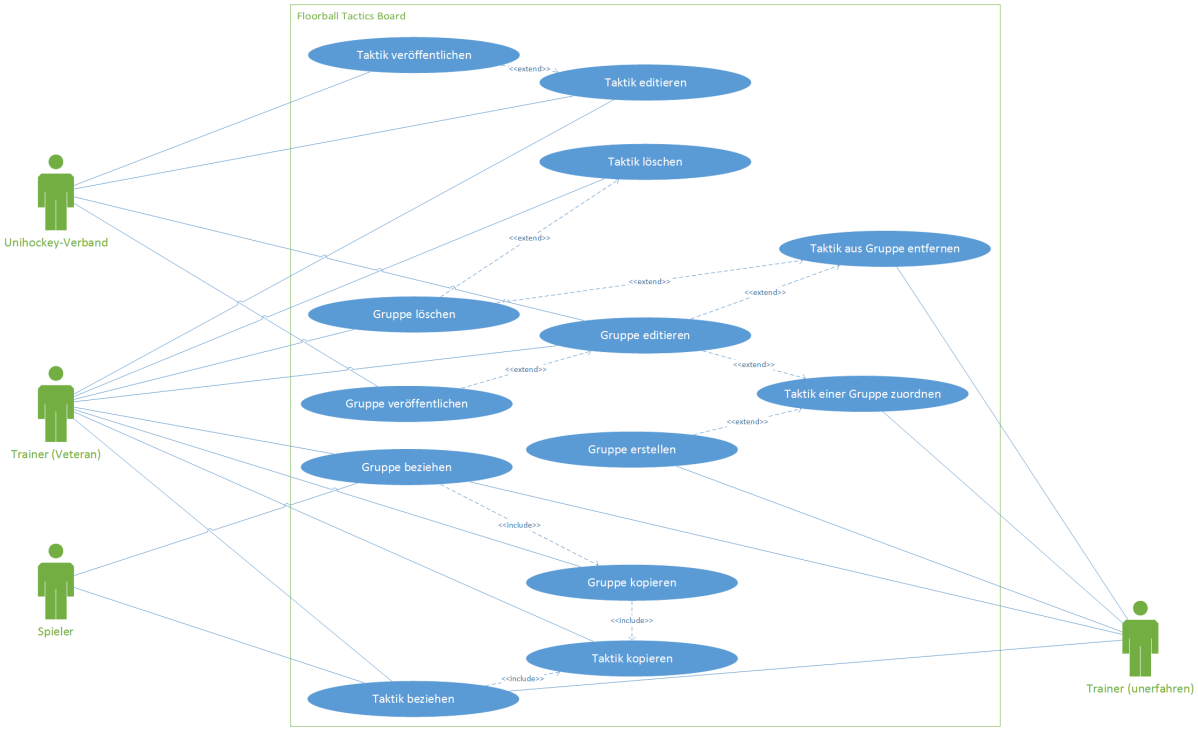


Abbildung 1.1: Use Case Diagramm

Actors

- **Unihockey-Verband**
  Interesse des Unihockey-Verbandes ist es, die Qualität nationaler Mannschaften zu steigern. Deshlab veröffentlicht dieses Germium Taktiken und Gruppen zum freien Gebrauch oder teilweise gegen ein Entgeld.
- **Trainer (Veteran)**
  Der bereits erfahrene Trainer lässt sich von Taktiken und Übungen anderer Leute inspirieren und passt diese bei Bedarf an die Bedürfnisse seiner Mannschaft an.
- **Trainer (unerfahren)**
  Der noch unerfahrene Trainer sucht sich im Taktik-Store Hilfe durch Taktiken und Übungen anderer Trainer. Diese organisiert er durch die Gruppenverwaltung, um bei der Planung seiner Trainings schneller auf die Übungen Zugriff zu haben.
- **Spieler**
  Spieler beziehen Gruppen und Taktiken, die im Training besprochen wurden, aus dem Store.

1.5 Use Cases "brief"

Im Folgenden wurde die vollständige Liste der Use Cases zur besseren Übersicht in 4 Themenbereiche unterteilt.

Taktik CRUD

UC101 **Taktik erstellen**
Erstellt einen Eintrag für eine neue Taktik in der lokalen Taktikübersicht durch Eingabe eines Namens und einer Beschreibung. Leitet auf die Taktik-Board Ansicht weiter, wo der Taktik-Inhalt bearbeitet werden kann.

UC102 **Taktik löschen**
Löscht eine Taktik aus der lokalen Taktikübersicht.

UC103 **Taktik editieren**
Name und Beschreibung können direkt aus der lokalen Taktikübersicht bearbeitet werden. Für die Bearbeitung des Taktik-Inhalts wird auf die Taktik-Board Ansicht weitergeleitet.

UC104 **Taktik suchen**
Der Anwender kann in der Übersicht der bestehenden Taktiken die Liste filtern, indem er die Suchparameter (Textsuche und Gruppenzugehörigkeit) definiert. Bestehende Filter werden angezeigt und können wieder entfernt werden.

UC105 **Taktik kopieren**
Der Benutzer kann eine Taktik duplizieren.

Taktik abspielen

UC201 **Taktik abspielen (automatisch)**
Die Schritte, einer vom Anwender gewählten Taktik, werden automatisiert wiedergegeben.

UC202 **Zeitliche Navigation innerhalb Taktik**
Der Anwender kann innerhalb des Ablaufs einer Taktik bewegen.

Taktikinhalt bearbeiten

Für Use Cases in dieser Kategorie, die eine Aktion auf dem Spielfeld mit Start- und Endzeitpunkt als Resultat haben, ist die funktionale Anforderung 1.7.1 zu beachten.

UC301 **Spieler hinzufügen**
Der Anwender fügt dem Spielfeld auf dem Taktik-Board einen zusätzlichen Spieler hinzu.

UC302 **Spieler entfernen**
Der Anwender entfernt auf dem Taktik-Board nicht benötigte Spieler.

UC303 **Spieler verschieben**
Der Anwender verschiebt die (Start-)Position eines Spielers. Eine Verschiebung ist nur zum Zeitpunkt 0 einer Taktik möglich.

UC304 **Spieler editieren**
Der Anwender bearbeitet auf dem Taktik-Board hinzugefügte Spieler (Startposition, Farbe, Nummer).

UC305 **Passweg hinzufügen**
Der Anwender definiert auf dem Taktik-Board ein Passspiel von einem Spieler zu einem anderen.

UC306 **Passweg entfernen**
Der Anwender entfernt auf dem Taktik-Board einen zuvor erstellten Passweg.

UC307 **Schussweg hinzufügen**
Der Anwender definiert auf dem Taktik-Board eine Schussbahn, die der Ball beschreiben soll.

UC308 **Schussweg entfernen**
Der Anwender löscht auf dem Taktik-Board einen bestehenden Schussweg.

UC309 **Ball hinzufügen**
Der Anwender fügt in der Taktik-Board Ansicht dem Spielfeld einen Ball hinzu.

UC310 **Ball verschieben**
Der Anwender verschiebt die (Start-)Position eines Balles. Eine Verschiebung ist nur zum Zeitpunkt 0 einer Taktik möglich.

UC311 **Ball entfernen**
Der Anwender entfernt in der Taktik-Board-Ansicht einen Ball vom Spielfeld.

UC312 **Spielerbewegung hinzufügen**
Der Anwender definiert auf dem Taktik-Board den Weg und die Dauer der Bewegung eines Spielers. Eine Bewegung kann parallel zu anderen Bewegungen ausgeführt werden.

UC313 **Spielerbewegung entfernen**
Der Anwender entfernt auf dem Taktik-Board eine zuvor definierte Bewegung eines Spielers.

UC314 **Ballbewegung hinzufügen**
Der Anwender definiert auf dem Taktik-Board die Bewegung des Balls (ohne Anspielziel).

UC315 **Ballbewegung entfernen**
Der Anwender entfernt auf dem Taktik-Board eine bestehende Ball-Bewegung.

UC316 **Ganzfeldansicht anzeigen**
Der Anwender wechselt auf dem Taktik-Board die Anzeige auf die Ganzfeldansicht.

UC317 **Halbfeldansicht anzeigen**
Der Anwender wechselt auf dem Taktik-Board die Anzeige auf die Halbfeldansicht.

UC318 **Notiz hinzufügen**
Der Anwender fügt dem Taktik-Board eine Handnotiz hinzu (hat keinen Einfluss auf das Spielgeschehen).

UC319 **Notiz entfernen**
Der Anwender kann dem Taktik-Board zuvor erstellte Notizen wieder entfernen.

Gruppen CRUD

UC401 **Gruppe erstellen**
Der Anwender erstellt in der lokalen Taktikübersicht eine Gruppe (Struktureinheit) und weist dieser Taktiken zu.

UC402 **Gruppe löschen**
Der Anwender löscht eine bestehende Gruppe und kann dabei wahlweise die enthaltenen Taktiken behalten oder löschen.

UC403 **Gruppe editieren**
Der Anwender editiert den Namen, die Beschreibung und/oder die zugeordneten Taktiken einer Gruppe.

UC404 **Gruppe kopieren**
Der Anwender erstellt ein Duplikat einer Gruppe und aller enthaltenen Taktiken.

UC405 **Taktik einer Gruppe zuordnen**
Der Anwender weist eine Taktik einer Gruppe zu.

UC406 **Taktik aus Gruppe entfernen**
Der Anwender entfernt eine Taktik aus einer Gruppe.

Taktiken austauschen

UC501 **Taktik veröffentlichen**
Der Anwender veröffentlicht eine Taktik und kann wahlweise während dieses Prozesses den Titel und die Beschreibung der Taktik anpassen.

UC502 **Gruppe veröffentlichen**
Der Anwender veröffentlicht eine Gruppe aus der Taktikübersicht und kann wahlweise während dieses Prozesses den Titel und die Beschreibung der Gruppe anpassen, wie auch Taktik-Zuweisungen editieren.

UC503 **Taktik beziehen**
Der Anwender bezieht eine Taktik aus dem Store, von welcher ein editierbares Duplikat in der lokalen Taktikübersicht erstellt wird.

UC504 **Gruppe beziehen**
Der Anwender bezieht eine Gruppe von Taktiken aus dem Store. Ein editierbares Duplikat der Gruppe wird der lokalen Taktikübersicht hinzugefügt.

Admin Web-UI

UC601 **Kategorien verwalten**
Verfügbare Kategorien müssen von den Administratoren erstellt und wieder entfernt werden können.

UC602 **Taktiken / Gruppen bereinigen**
Der Anwender kann Taktiken und Gruppen im Web-UI auflisten und, falls dazu berechtigt, löschen.

UC603 **User verwalten**
Benutzer sollen für die Funktionen des Web-UIs berechtigt werden können.

1.6 Use Cases “fully dressed“

In diesem Kapitel sind komplexere Use Cases mit Extensions oder Inclusions anderer Use Cases im Format “Fully Dressed“ aufgelistet. Da die Aktionen für Gruppen und Taktiken sich in vielen Punkten überschneiden, ist jeweils nur eine der Aktionen aufgelistet. Unterschiede der weggelassenen Kategorie können der “Brief“-Beschreibung entnommen werden.

UC402: Gruppe löschen

Actors
Anwender
Scope
Lokale Taktikübersicht
Preconditions

- Anwender befindet sich auf der lokalen Taktikübersicht (F01)
- Es existiert eine Gruppe, der mindestens eine Taktik zugeordnet ist

Main Success Scenario

1. Der Anwender selektiert eine Gruppe mit mindestens einer zugeordneten Taktik
2. Der Anwender betätigt die Löschen-Aktion
3. Das System zeigt einen “Aktion-Bestätigen Dialog“ an, in dem spezifiziert werden kann, ob zugehörige Taktiken gelöscht oder erhalten bleiben sollen
4. Der Anwender wählt “Taktiken löschen“ und bestätigt die Aktion

   Extension *Taktik löschen*
5. System löscht die Gruppe und die ausgewählten Taktiken

Alternative Scenarios

4a Aktion abbrechen

1. Der Anwender bricht die Aktion ab
2. System kehrt auf die lokale Taktikübersicht zurück

4b Option “Taktiken behalten“

1. Der Anwender wählt “Taktiken behalten“ und bestätigt die Aktion
2. Das System löscht die Gruppe und weist die enthaltenen Takiken einer Meta-Gruppe zu
3. Das System kehrt auf die lokale Taktikübersicht zurück

Postconditions

- Die Gruppe wurde gelöscht
- Falls Main Success Scenario durchgeführt wurde: Die zugeordneten Taktiken existieren (ohne Gruppenzuordnung)
- Falls Alternative Scenario *4b* durchgeführt wurde: Die zugeordneten Taktiken wurden gelöscht

UC403: Gruppe editieren

Actors
Anwender
Scope
Lokale Taktikübersicht
Preconditions

- Anwender befindet sich auf der lokalen Taktikübersicht (F01)
- Es existiert eine Gruppe

Main Success Scenario

1. Der Anwender selektiert eine Gruppe
2. Der Anwender löst die "Bearbeiten"-Aktion aus
3. Das System öffnet die Gruppe-bearbeiten Ansicht
4. Der Anwender bearbeitet Eigenschaften der Gruppe
   Extensions *Taktik einer Gruppe zuordnen* und *Taktik aus Gruppe entfernen*
5. Der Anwender schliesst die Bearbeitungsaktion ab
6. Das System speichert die Änderungen

Alternative Scenarios

*a Aktion abbrechen

1. Der Anwender bricht die "Bearbeiten"-Aktion ab
2. System verwirft die Änderungen

Postconditions

- Die eingegebenen Änderungen wurden gespeichert

UC502: Gruppe veröffentlichen

Actors
Anwender
Scope
Lokale Taktikübersicht
Preconditions

- Anwender befindet sich auf der lokalen Taktikübersicht (F01)
- Es existiert eine Gruppe mit mindestens einer zugeordneten Taktik

Main Success Scenario

1. Der Anwender selektiert eine Gruppe
2. Der Anwender löst die "Veröffentlichen"-Aktion aus
3. Das System öffnet die "Gruppe veröffentlichen" Ansicht
4. Der Anwender schliesst die "Veröffentlichen"-Aktion ab
5. Das System lädt die Gruppe auf den Server

Alternative Scenarios

4a Aktion abbrechen

1. Der Anwender bricht die Aktion ab
2. System verwirft die Änderungen und kehrt auf die lokale Taktikübersicht zurück

4b Gruppe bearbeiten

1. Anwender bearbeitet die Gruppe
   Extension *Gruppe editieren*
2. Weiter mit Main Success Scenario Schritt 4

Postconditions

- Die Gruppe inkl. aller zugehörigen Taktiken wurden auf dem Server gespeichert

UC105: Taktik kopieren

Actors
Anwender
Scope
Lokale Taktikübersicht
Preconditions

- Anwender befindet sich in der lokalen Taktikübersicht (F01)
- Mindestens eine Taktik existiert

Main Success Scenario

1. Der Anwender selektiert eine Taktik
2. Der Anwender löst die "Duplizieren"-Aktion aus
3. Das System zeigt die "Duplizieren"-Ansicht an
4. Der Anwender bestätigt die Aktion
5. Das System dupliziert die Taktik

Alternative Scenarios

4a Aktion abbrechen

1. Der Anwender bricht die Aktion ab

4b Name angeben

1. Anwender gibt einen Namen ein, unter welchem die Taktik gespeichert werden soll
2. Weiter mit Main Success Scenario Schritt 4

Postconditions

- Die Taktik wurde dupliziert

UC503: Taktik beziehen

Actors
Anwender
Scope
Taktik-Store
Preconditions

- Anwender befindet sich im Taktik-Store (F04)
- Mindestens eine Taktik wird angezeigt

Main Success Scenario

1. Der Anwender selektiert eine Taktik
2. Der Anwender löst die "Download"-Aktion aus
3. Das System zeigt die "Duplizieren"-Ansicht an
4. Der Anwender bestätigt die Aktion
5. Das System dupliziert die ausgewählte Taktik in die lokalen Taktikübersicht

Alternative Scenarios

3a Aktion abbrechen

1. Der Anwender bricht die Aktion ab

3b Name angeben

1. Anwender gibt einen Namen ein, unter welchem die Taktik gespeichert werden soll
2. Weiter mit Main Success Scenario Schritt 4

Postconditions

- Die Taktik wurde vom Online-Store in den lokalen Speicher kopiert

1.7 Functional Requirements

Funktionale Anforderungen beziehen sich auf Aktionen, die von einem Benutzer ausgeführt werden können und definieren Use Case übergreifende Rahmenbedingungen.

1.7.1 FR01: Paralleles Abspielen von Spielzügen

Für Aktionen auf dem Taktik-Board, die einen definierbaren Start- und Endzeitpunkt haben, muss eine zeitliche Überschneidung möglich sein. Dies bezieht sich insbesondere auf Bewegungen verschiedener Aktoren (Bälle und andere Spieler), die unabhängig voneinander stattfinden müssen. Sie dürfen somit nicht an die vollendete Ausführung einer Aktion anderer Aktoren gebunden sein.

1.7.2 FR02: Abspielen von Spielzügen während dem Aufnehmen einer neuen Bewegung

Damit der Benutzer seine Aktion einem zeitlichen Kontext zuordnen kann, sollen bereits erfasste Spielzüge, die gleichzeitig wie die zu erfassende Aktion stattfinden, abgespielt werden.

1.8 Non-Functional Requirements

Die Nicht-Funktionalen Anforderungen sind für ein Testgerät festgelegt, das mindestens die folgenden Spezifikationen erfüllt:

- CPU 4 x 1.9GHz
- Arbeitsspeicher 2GB RAM
- Auflösung 2048 x 1536
- Android 5.0 oder höher
- Verbindung WiFi oder mittel bis guter 3G Empfang

1.8.1 NFR01: Ladezeiten View

Ziel

Hohe Benutzerfreundlichkeit durch möglichst kurze Ladezeiten.

Szenario Komponenten

Stimulus
Anwender navigiert zu anderer View

Environment
Runtime, Navigation auf Seite ohne Online-Funktionen

Response
System rendert die gewünschte Seite in <1s.

Response Messung
Vergangene Zeit von Eingabe der Aktion bis zur vollständigen Darstellung der View.

1.8.2 NFR02: Statusvisualisierung

Ziel

Der Anwender weiss jederzeit, in welchem Zustand sich das System befindet.

Szenario Komponenten

Stimulus
Starten der Applikation, Ausführen einer Aktion, Navigation zu anderer View

Environment
Systemstart / Runtime

Response
Benutzer wird über Resultat der Aktion informiert (Statusmeldung) oder wird bei länger dauernden Abläufen über den Fortschritt informiert (Ladebalken o.ä.)

Response Messung
Jede Aktion liefert sofortiges Feedback, auch wenn längere Ladezeiten anfallen sollten.

1.8.3 NFR03: Ladezeiten von Online-Funktionen

Ziel

Die Ladezeiten von Funktionen und Views mit Onlineanbindung dürfen das Benutzererlebnis nicht negativ beeinflussen.

Szenario Komponenten

Stimulus
Navigation zu einer View mit Online-Abfragen oder ausführen einer Aktion, die eine Kommunikation zum Server veranlasst.

Environment
Runtime

Response
Die View-Struktur wird ohne Inhalt innerhalb von <1s aufgebaut. Vom Server geladene Komponenten (Inhalt für View-Struktur), werden nachträglich die View geladen.
Eine Anfrage mit Inhalt von maximal 10 Taktiken wird vom Server innerhalb von <2s beantwortet. D.h. der Inhalt ist nach 2s verfügbar.

Response Messung
View-Aufbau: Vergangene Zeit von Initiierung der Aktion bis zum vollständigen Aufbau der Struktur der View Request: Vergangene Zeit von Initiierung der Aktion bis geladene Elemente angezeigt werden

1.8.4 NFR04: Verfügbarkeit unter Last

Ziel

Online-Funktionen müssen im Betrieb mit vielen Usern verfügbar sein.

Szenario Komponenten

Stimulus
Ausführen einer Aktion mit Online-Anbindung / Navigation zu einer View, die Inhalte vom Online-Store bezieht.

Environment
Runtime, parallel mit 100 anderen Anwendern

Response
Online-Inhalte werden unter einer Backend-Systemlast von 100 Usern innerhalb von max. 5s geladen.

Response Messung
Vergangene Zeit von Initiierung der Aktion bis geladene Elemente angezeigt werden

1.8.5 NFR05: Android-Mindestversion

Dem Benutzer soll ein modernes UI mit aktuellen Komponenten und intuitivem Bedienkonzept präsentiert werden. Dafür muss eine Android-Version vorausgesetzt werden, die den Entwicklern nötige Tools zur Verfügung stellt, mit denen dies erreicht werden kann.

1.9 Domain Modell

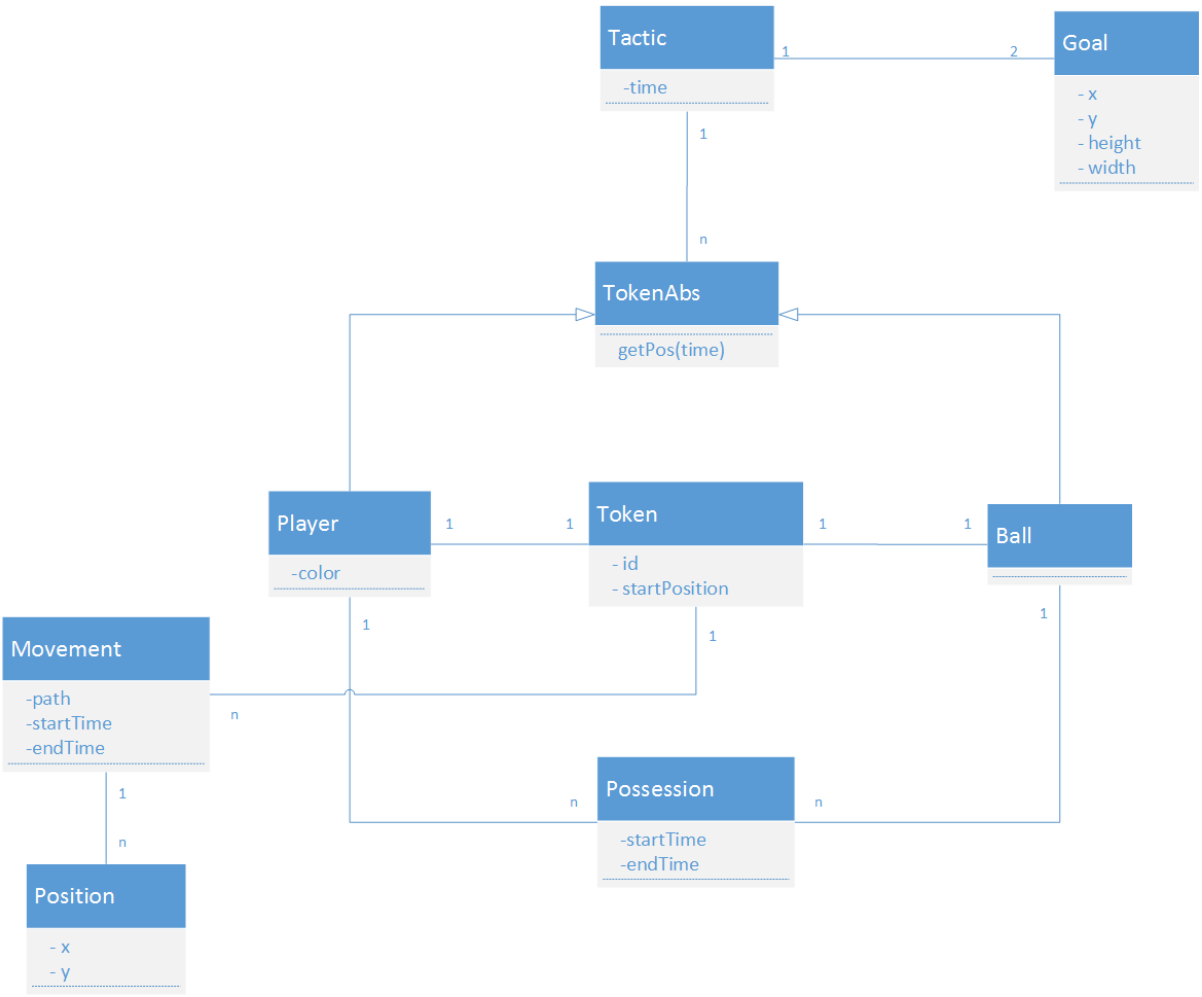


Abbildung 1.2: Domain-Modell

Movement

Bewegungen (*Movement*) finden über eine bestimmte Dauer, definiert durch Start- und Endzeitpunkt, statt und können einem beliebigen *Token* zugeordnet werden. Die Position eines Tokens zu einem bestimmten Zeitpunkt X kann über dessen Methode *getPos(time)* ermittelt werden.

Darin wird untersucht, ob eine *Movement*-Instanz zu Zeitpunkt X stattfindet, mit welcher dann die Berechnung der aktuellen Position durchgeführt wird. Liegt keine aktive Bewegung vor, wird bei Spielern (*Player*) deren letzte Position zurückgegeben. Die Position eines Balls (*Ball*) wird in diesem Fall über den aktuellen Ballbesitz ermittelt.

Possession

Mit dem Ballbesitz (*Possession*) werden zweierlei Dinge festgestellt. Zum einen wird die Verfügbarkeit von Aktionen (Passen und Schiessen) eines Spielers überprüft, zum anderen wird dadurch die aktuelle Position des Balles bestimmt.

1.10 Tests

Zur Überprüfung der App werden System-, Android- und Unit-Tests entwickelt. Die Systemtests sind in Anhang A zu finden. Android- und Unit-Tests sind im Programmcode zu finden.

1.10.1 Durchführung Android-Tests

Android-Tests brauchen einen Context (z.B. eine Activity), da sie entweder auf UI-Elemente oder eine Datenbank angewiesen sind. Jede TestSuite muss für sich alleine ausgeführt werden.

13.5.16	**1.6.16**	**13.6.16**	**TestSuite**
√	√	√	ApplicationTest
√	√	√	ServiceTestSuite
√	√	√	UtilTestSuite

1.10.2 Durchführung Unit-Tests

Unit-Tests funktionieren ohne Context. Die Testklassen müssen ebenfalls für sich alleine ausgeführt werden, da Android-Studio sonnst auch AndroidTests mit demselben Packagename ausführt (jedoch ohne Context, was zu Fehlern führt).

13.5.16	**1.6.16**	**13.6.16**	**Unit-Test**
√	√	√	MovementTest
√	√	√	AdapterUtilTest
√	√	√	CoordinateConverterTest
√	√	√	GeometryUtilTest

1.10.3 Durchführung Systemtest

Die Systemtests wurden in Absprache mit dem Betreuer erst gegen Ende des Projekts entwickelt, weil sich das User Interface fortlaufend änderte und erst gegen Ende des Projekts eine stabile Version erreichte.

13.6.16	**Systemtest**
√	T101A: Taktik erstellen
√	T101B: Taktik erstellen mit Standard-Werten
√	T101C: Taktik erstellen in eigener Gruppe
√	T101D: Halb-Feld Taktik erstellen
√	T102A: Taktik löschen
√	T103A: Taktik Eintrag editieren
√	T103B: Taktik editieren
√	T101A: T104A: Taktik Name suchen
√	T104B: Taktik Name suchen mit nicht existierendem Suchtext
√	T104C: Taktik Beschreibung suchen
√	T104D: Nicht angezeigte Taktik suchen
√	T104E: Suchfeld anwählen
√	T104F: Suchfeld Inhaltlöschen
√	T104G: Gruppe ausblenden
√	T104H: Gruppe einblenden
√	T105A: Taktik kopieren
√	T105B: Geteilte Taktik kopieren
√	T201A: Taktik abspielen (aus Card)
√	T201B: Taktik abspielen (aus Taktiktafel)
√	T201C: Taktik mit Pass/Schuss abspielen
√	T202B: Zeitliche Navigation innerhalb Taktik(aus Card)
√	T202B: Zeitliche Navigation innerhalb Taktik(aus Taktiktafel)
√	T301A: Spieler hinzufügen
√	T302A: Spieler löschen
√	T303A: Spieler verschieben
√	T303B: Spieler verschieben nach Zeitpunkt 0
√	T304A: Spieler editieren
√	T305A: Passweg hinzufügen
√	T307A: Schussweg hinzufügen
√	T309A: Ball hinzufügen
√	T3011A: Ball entfernen
√	T3012A: Spielerbewegung hinzufügen
√	T3012B: Parallele Spielerbewegung hinzufügen
√	T3013B: Spielerbewegung entfernen
√	T316A: Ganzfeldansicht anzeigen
√	T317A: Halbfeldansicht anzeigen
√	T401: Gruppe erstellen
√	T101A: Taktik erstellen
√	T402: Gruppe löschen
√	T403A: Gruppe editieren
√	T404A: Gruppe kopieren
√	T405A: Taktik einer Gruppe zuordnen
√	T406A: Taktik aus Gruppe entfernen
√	T501A: Taktik veröffentlichen
√	T502A: Gruppe veröffentlichen
√	T503A: Taktik beziehen 1. Phase
√	T503B: Taktik beziehen 2. Phase
√	T504A: Gruppe beziehen 1. Phase
√	T504B: Gruppe beziehen 2. Phase
√	T601A: Kategorien verwalten - hinzufügen
√	T602A: Kategorien verwalten - löschen
√	T603A: Taktiken bereinigen
√	T603B: Gruppen bereinigen

Kapitel 2

Architektur

2.1 Evaluation Presentation-Layer Pattern

In Android-Applikationen ist oft eine hohe Kopplung zwischen Activities und anderen Klassen vorhanden und deren Aufgabenbereich ist schwer auszumachen. Da das Floorball Tactics-Board im Vergleich mit einfachen CRUD-Applikationen über eher komplexe Views mit vielen Interaktionsmöglichkeiten verfügt, soll durch die Verwendung eines View-Patterns eine gute Wartbarkeit des Codes erreicht werden.

Anforderungen

Bei der Wahl des Patterns ist zu beachten, dass die Parallelität von Aufnehmen einer Bewegung und Abspielen einer Taktik aus 1.7.2 nicht beeinträchtigt wird und der Abspielmechanismus aus mehreren Quellen gestoppt werden kann, unter anderem

- Pause-Button (manuell durch User)
- Ende von Aufnehmen einer Aktion
- Erreichen der Endzeit einer Taktik

Im Zusammenhang mit Use Case 201 soll bei länger andauernden Berechnungen auf nicht blockierende Positionierungs-Abfragen und kurze Reaktionszeiten auf User-Interaktionen geachtet werden.
Um eine mögliche zukünftige Portierung auf andere Betriebssysteme nicht zu verunmöglichen, soll das Pattern zudem die Wiederverwendbarkeit des Codes begünstigen.

Model-View-Controller (MVC)

Das Model-View-Controller-Pattern ist besonders in Webapplikationen stark verbreitet. Die Rollen der drei Komponenten definieren sicht wie folgt:

- **Model** Bearbeitet Daten und führt Bussiness-Logik aus
- **View** Grafische Benutzeroberfläche, die dem Benutzer angezeigt wird; sendet Befehle an den Controller
- **Controller** Reagiert auf Benutzerinteraktionen, bereitet Interaktionen für Model- und View-Schicht auf

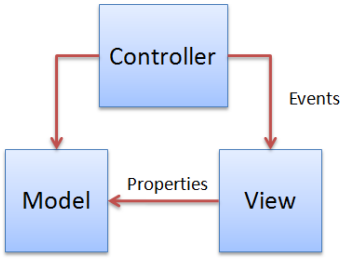


Abbildung 2.1: MVC-Pattern [4]

Model-View-Presenter (MVP)

Dieses Pattern stammt vom MVC-Pattern ab und verfolgt das Ziel, die Separation-of-Concerns zu verbessern, wie auch automatisierbare UI-Tests zu ermöglichen.

- **Model** Bearbeitet Daten und führt Bussiness-Logik aus
- **View** Grafische Benutzeroberfläche, die dem Benutzer angezeigt wird; enthält Referenz auf den Presenter, an den Aktionen delegiert werden
- **Presenter** Kümmert sich um die Aufbereitung von Model-Daten und verarbeitet die Interaktionen mit der View

Der Presenter ist in diesem Fall stärker an die View gekoppelt, als der Controller im MVC-Pattern, und kann unter anderem auch Einfluss auf die View nehmen.

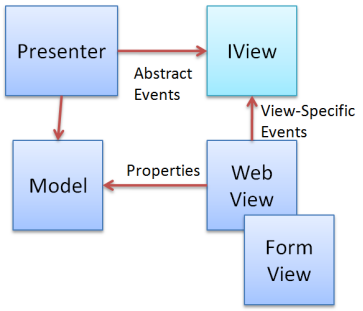


Abbildung 2.2: MVP-Pattern [4]

Model-View-ViewModel (MVVM)

Dieses Pattern findet seinen Anwendungsbereich besonders im Zusammenhang mit einer Markup-Sprache, an die Daten gebunden werden. Dieses Databinding wird durch Verwendung eines ViewModels vom Domain-Objekt entkoppelt. Das ViewModel hält zudem den Status von Informationen, falls ein solcher benötigt wird.

- **Model** Enthält Daten (ohne Verhalten und Logik), entspricht dem Domain Objekt
- **View** Grafische Benutzeroberfläche, die dem Benutzer angezeigt wird und nimmt Inputs entgegen. Die View ist aktiv und enthält Logik, Databindings und kann Events verarbeiten.
- **ViewModel** Regelt die Interaktionen zwischen View und Model, kümmert sich um die Daten-Weitergabe und -Beschaffung (z.B. über einen Service) und kann Events auf der View auslösen

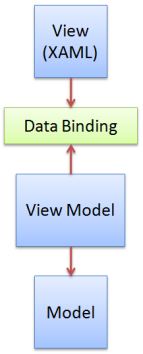


Abbildung 2.3: MVVM-Pattern [3]

Vergleich Presentation-Layer Pattern

MVC

Vorteile	Nachteile
• Starke Trennung der Schichten • Wiederverwendbarkeit Controller und Model	• Synchrones Redrawing bei Positionsabfragen von Tokens; benötigt weitere Mechanismen um längere Wartezeiten korrekt zu überbrücken • Bei sauberer Verwendung sollte kein Status im Controller gehalten werden (Timer)

MVP

Vorteile	Nachteile
• Presenter kann Events an View senden UI-Redraw über requestLayout() nach langen Berechnungen (Android internes Redraw-Queueing) • Von der View entkoppeltes Testing möglich • Status kann im Presenter gehalten werden (Abspielstatus und -zeit)	• Starke Kopplung Presenter<->View

MVVM

Vorteile	Nachteile
• Model-Status in ViewModel abbildbar (für längere Berechnungen)	• Starke Fragmentierung auf Model-Basis (Interaktionen auf mehreren Models gleichzeitig notwendig) • DataBinding mit komplexen Views umständlich • Nicht mit allen Grafik-Libraries kompatibel

Entscheidung

Aufgrund der guten Integrationsmöglichkeit mit dem Redraw-Queueing von Android, wie auch der Unterstützung von Parallelität von Aktionen (Steuerung Abspielen gekoppelt an andere Aktionen), entscheiden wir uns für das *Model-View-Presenter Pattern (MVP)*.

Führt das Pattern bei einfachen Views zu zusätzlichem Aufwand ohne sichtlichen Mehrwert, nehmen wir uns die Freiheit, den Presenter wegzulassen und die Aktionen direkt in der View zu implementieren, sofern dies nicht grob gegen definierte Konzepte verstösst.

2.2 Evaluation Libraries

Für Android, insbesondere für Java, existiere viele Frameworks und Libraries, die dem Entwickler Arbeit und Zeit einsparen sollen. Um für die grössten Aufgabenbereiche die passendste Library zu finden, sind diese in einem Vergleich zu evaluieren.

2.2.1 Grafik

Betrifft die Ansicht, das Zeichnen und Abspielen einer Taktik, wie in Wireframes 3.6.2 und 3.6.3 dargestellt, und ist unabhängig vom Selektions- und Administrationsbildschirm.

- *Im Zusammenhang mit* dem Taktik-Board
- *in Anbetracht* der grafischen Umsetzung der definierten Features
- *haben wir uns entschieden,* diese mit nativen Android-Komponenten zu implementieren
- *und gegen* den Einsatz einer Grafik-Library
- *um* die Einarbeitungszeit zu verkürzen und das Einhalten des Android-Styleguides zu vereinfachen
- *dafür akzeptieren wir als Konsequenz* Funktionalität, die eine Grafiklibrary mitbringt, zu verlieren

2.2.2 Geometrie

- *Im Zusammenhang mit* den Berechnungen von Pfaden und Positionen
- *in Anbetracht* der besseren Testbarkeit der Features
- *haben wir uns entschieden,* die JTS Geometry Library von VividSolutions einzubinden
- *und gegen* den Einsatz der Android-Grafik-Komponenten
- *um* eine sauberere Umsetzung und zugleich das Testing dieser Berechnungen mit nativem JUnit (ohne das Starten eines Android-Emulators) zu ermöglichen
- *dafür akzeptieren wir als Konsequenz*, dass eine zusätzliche Library zu importieren und Umrechnungen von JTS Pfaden auf Android-Grafik-Pfade anzustellen

2.2.3 Object-Relational Mapping

- *Im Zusammenhang mit* dem im Android-Projekt verwendeten Object-Relational Mapping Framework
- *in Anbetracht* von Performance und Marktreife
- *haben wir uns für* greenDAO entschieden
- *und gegen* den Einsatz von ActiveAndroid oder SugarORM
- *um* eine ausgereiftere, kontinuierlich unterstützte und weiterentwickelte Library zu erhalten
- *dafür akzeptieren wir als Konsequenz*, Unannehmlichkeiten wie das Fehlen von Annotations und die eigenhändige Konfiguration von Beziehungen in Kauf nehmen zu müssen.

2.2.4 Datenaustausch

- *Im Zusammenhang mit* dem Datenaustausch der Applikation mit dem Server
- *in Anbetracht* Wartbarkeit und Umsetzungsaufwand
- *haben wir uns für* Retrofit zusammen mit Gson und OkHttp entschieden
- *und gegen* Google Volley
- *um* einen gut wartbaren, getrennten Code zu erhalten
- *dafür akzeptieren wir als Konsequenz*, mehr Initialaufwand zu haben, um die Mechanismen zu verstehen.

2.2.5 Bitmap-Caching

- *Im Zusammenhang mit* dem Memoryverbrauch von Bitmaps
- *in Anbetracht* von Ladezeiten und Umsetzungsaufwand
- *haben wir uns für* den Android Universal Image Loader entschieden
- *und gegen* die eigene Entwicklung eines Caches
- *um* eine kompakte, schnelle und konfigurierbare Umsetzung zu erhalten
- *dafür akzeptieren wir als Konsequenz*, nicht genau voraussagen zu können, wie und wann die Bitmaps gecached werden.

2.3 Evaluation Technologien Backend

Während der Umfang der Server-Applikation bekannt ist, muss eine Technologie für dessen Umsetzung eruiert werden. Diese werden anhand aufgelisteter Kriterien ausgewählt und evaluiert.

Übergreifendes Kriterium: Skalierbarkeit

Mit dem Ziel, die entwickelte Android-App der Öffentlichkeit zugänglich zu machen, sind für die Server-Komponente Technologien einzusetzen, die mehrere parallele Benutzer unterstützen. Da die Grösse des Zielpublikums aber schwer einzuschätzen ist, muss bereits zu Beginn auf gute Skalierbarkeit geachtet werden, um die NFR 1.8.3 zu erfüllen.

Gewichtung: hoch

2.3.1 Sprache

Kriterium 1: Google Libraries

Während der Implementation des Prototypen wurde die Entscheidung getroffen, Benutzer über ein Google Konto zu authentifizieren. Dadurch bietet sich eine Programmiersprache an, in der Google Libraries verwendet werden können, um die in das OAuth-Verfahren investierten Zeitaufwand zu minimieren.

Gewichtung: Tief

Kriterium 2: Geräteunabhängiger Datenaustausch

Das Projekt verfügt über zwei verschiedene grafische Oberflächen; die Android-Applikation und das Web Admin UI. Diese sollen beide von derselben Schnittstelle Gebrauch machen, wodurch ein geräteunabhängiges Datenaustauschformat gewählt werden soll.

Gewichtung: Hoch

Technologie-Vergleich

Die folgenden Kandidaten werden für genannte Kriterien als gleichwertig eingestuft. Die Wahl wurde deshalb auf Basis persönlicher Kriterien des Teams getroffen.

	NodeJS	**Playframework (Scala)**	**Spring (Java)**
Kenntnisse	Mittel	Tief	Hoch
Interesse	Tief	Hoch	Tief
Wartbarkeit	Mittel	Gut	Gut

Letzere beiden Optionen schneiden gemäss diesen Faktoren gleich ab. Das Team entscheidet sich für das Playframework in Scala.

2.3.2 Datenbank

Bei der Datenbank ist, wie auch bei der Programmiersprache, auf Skalierbarkeit zu achten, da sonst die skalierte Webapplikation beim Datenbankzugriff in einen Engpass geraten kann.

Folgende Eigenschaften der Server-Komponente wurden für die Datenbank-Evaluation als ausschlaggebend erachtet:

- Query-Komplexität: tief
- Wenig CREATE-Operationen
- Viele READ-Operationen
- Keine / Wenig UPDATE-Operationen
- Wenig DELETE-Operationen

Mehrere Kandidaten liefern für die definierten Kriterien gute Resultate. Unter anderem schneiden die bereits etablierten Technologien *MySQL*, *MongoDB* und *CouchDB* für unseren Verwendungszweck etwa gleich gut ab.

Aufgrund der sehr guten Integrationsmöglichkeit in die gewählte Programmiersprache durch das Framework *ReactiveMongo* wurde ***mongoDB*** als zu verwendende Technologie bestimmt.

2.4 Prozess-Interaktionen

Für das Taktik-Board existieren mehrere Abläufe, die verschachtelte Interaktionen zwischen Objekten erfordern. Um eine bessere Übersicht zu bieten, werden diese im Folgenden erläutert.

2.4.1 Zeitliche Navigation

Mit Bedienung der Zeit-Navigation kann der Anwender zu einem bestimmten Zeitpunkt des Taktikablaufs springen. Es wird die Position aller Tokens des Spielfelds zum gewünschten Zeitpunkt ermittelt, an welcher diese dann dargestellt werden.

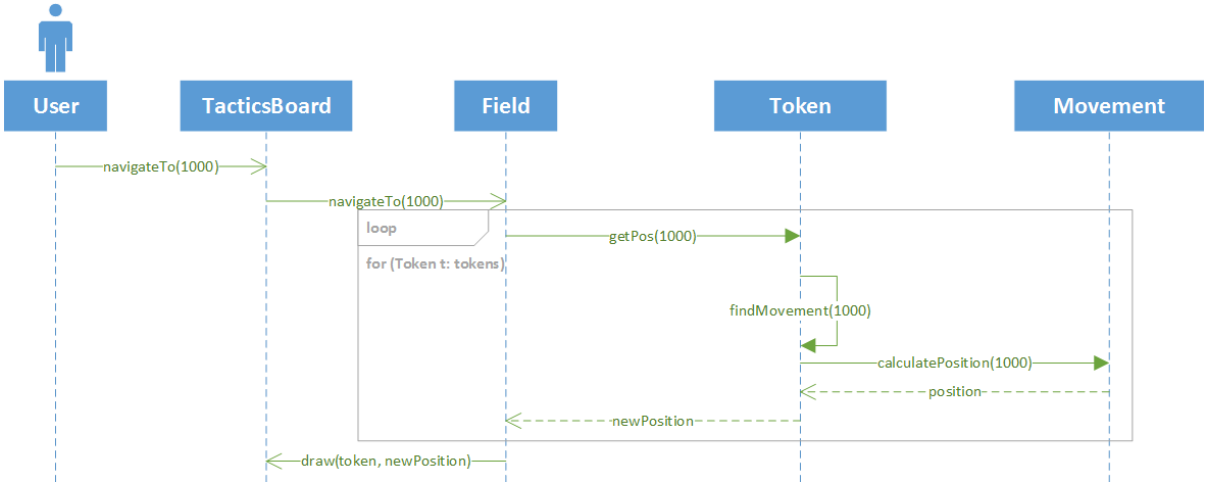


Abbildung 2.4: Sequenzdiagramm “Zeitliche Navigation“

2.4.2 Taktikwiedergabe

Der Abspielmechanismus verfügt über einen von der Activity entkoppelten Timer, der ein Redraw-Aufruf an die View sendet. Diese fragt die Positionen aller Spielfeld-Elemente ab und rendert sie entsprechend.

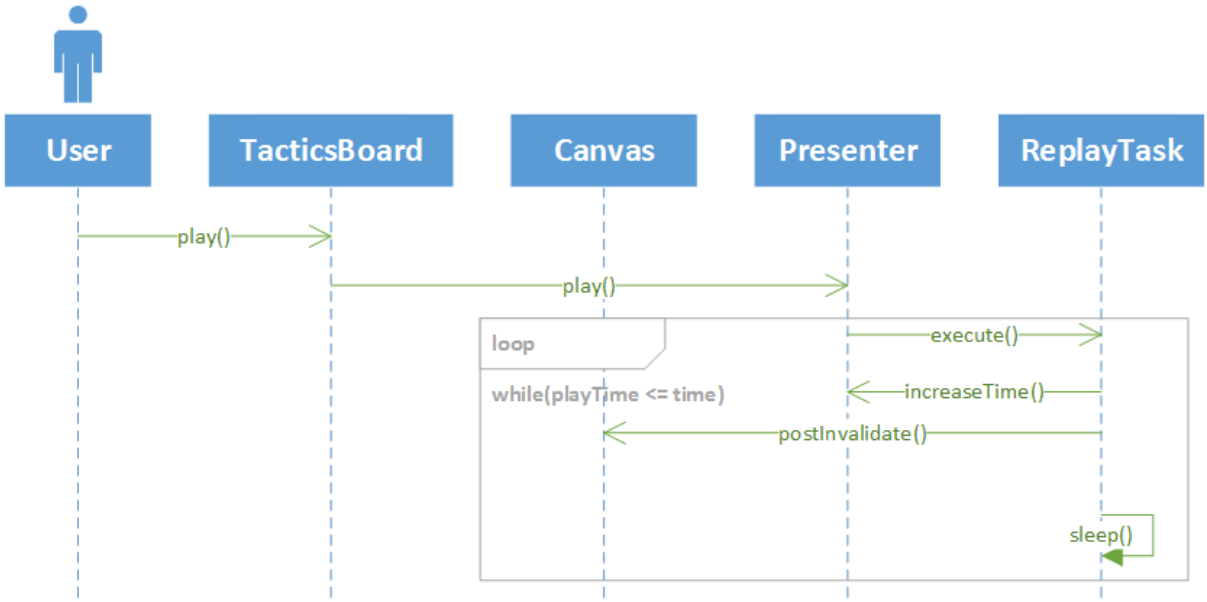


Abbildung 2.5: Sequenzdiagramm “Taktikwiedergabe“

2.4.3 Passen

Kernaufgaben dieser Komponente sind:

- Ermitteln Start-Position und -Zeitpunkt
- Für Laufpässe:

  Ermitteln, zu welchem Zeitpunkt sich das Anspielziel an der gewünschten Position befindet
- Für Pässe an statisches Ziel (Ziel ist für längere Zeit am gleichen Ort):

  Definieren einer maximalen Geschwindigkeit des Balls
- Berechnen des Pfades (gerade Verbindung zwischen Start- und Endposition) sowie Start- und Endzeit (gemäss der letzten zwei Punkte)

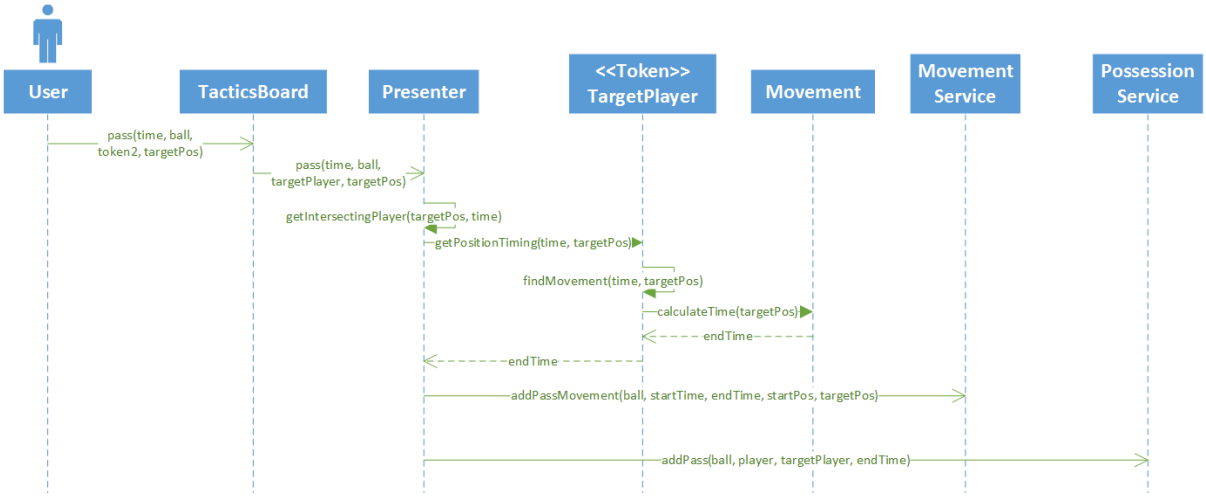


Abbildung 2.6: Sequenzdiagramm “Passen“

2.5 Architektur-Übersicht

Das Floorball Tactics-Board teilt sich grundlegend in zwei Komponenten auf, Android-Applikation und Backend-Applikation. Beide verfügen jeweils über eine Datenbank, in denen Informationen abgelegt werden. Für die Server-Komponente existiert zudem eine rudimentäre Web-Administrationsoberfläche.

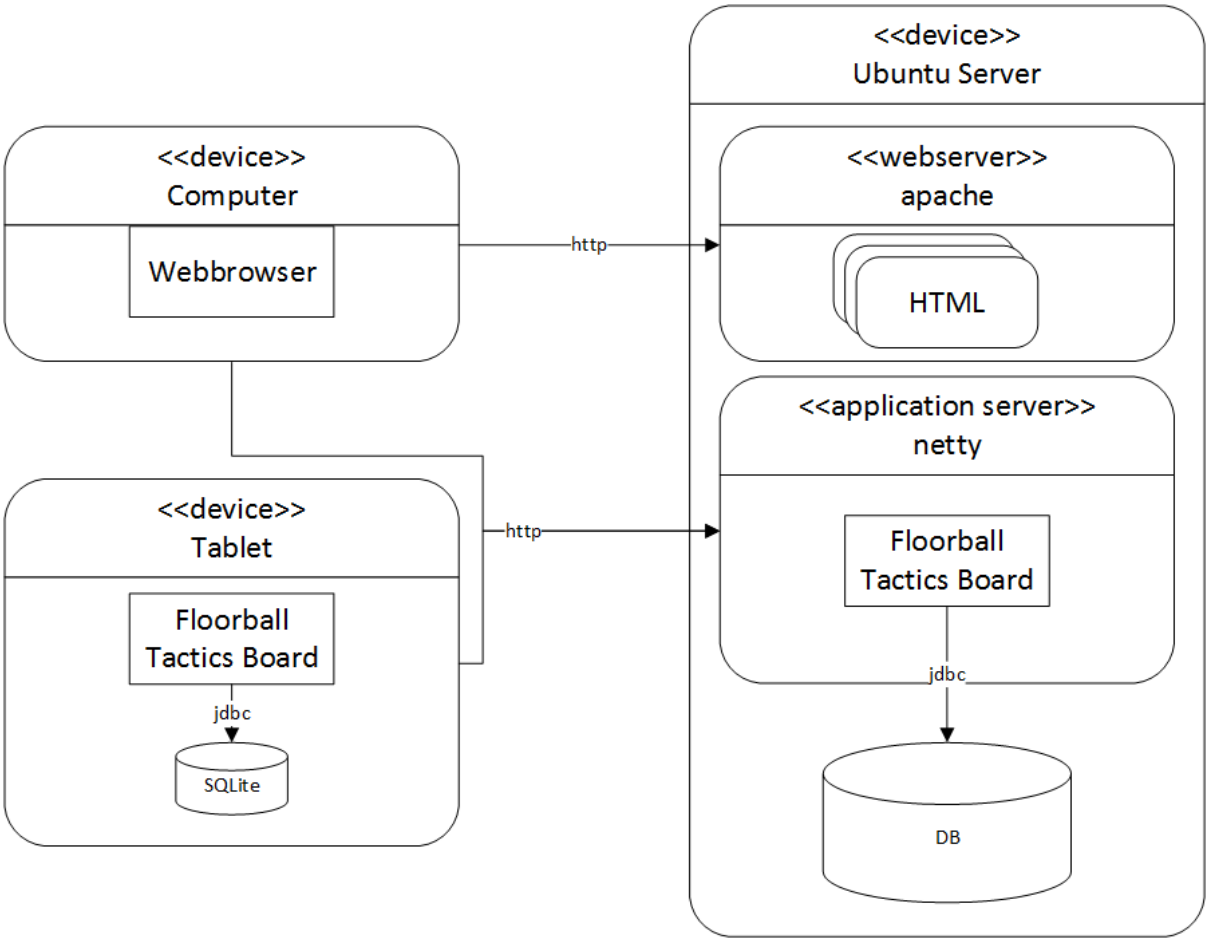


Abbildung 2.7: Deployment-Diagramm

Besagte Komponenten sind jeweils in mehrere Schichten aufgeteilt. Die Zugehörigkeit zu den Layern ist der Packagebeschreibung zu entnehmen und ist im Kontext der jeweiligen Komponente gültig.

Presentation-Layer	Grafische Aufbereitung von Informationen und Verarbeitung von Benutzereingaben
Bussiness-Logic-Layer	Verarbeitung von Daten
Data-Access-Layer	Zugriff auf abgelegte Daten (Datenbank, Backend, ...)

2.5.1 Android-Applikation

Package-Struktur

ch.codinglynx.tacticsboard.application.model *Data-Access-Layer*
Enthält Domain-Objekte, die zur Datenhaltung verwendet werden. Zusätzlich zu den Domain-Objekten wird von *greenDAO* ein DAO (Data-Access-Object) zu jedem Domain-Objekt generiert.

ch.codinglynx.tacticsboard.application.serializer *Data-Access-Layer*
Umfasst Serialisierungs- und Deserialisierungs-Logik, sowie Klassen, die mit Retrofit im Zusammenhang stehen.

ch.codinglynx.tacticsboard.application.service *Bussiness-Logic-Layer*
Sämtliche Klassen, die Bussiness-Logik abhandeln, werden dem *service*-Package zugeordnet.

ch.codinglynx.tacticsboard.application.ui *Presentation-Layer*
Alle mit der Benutzeroberfläche in Zusammenhang stehenden Klassen sind in diesem Package abgelegt. Dies umfasst Activities, Presenter, Fragments, Dialoge, Adapter, Listener und SearchProvider.

ch.codinglynx.tacticsboard.application.util
Das util-Package beherbergt Utility-Klassen, die in mehreren Kontexten Verwendung finden.

2.5.2 Backend

Package-Struktur

app/controllers *Presentation-Layer*
Controller aus dem MVC-Pattern sind in diesem Package aufzufinden.

app/dto *Data-Access-Layer*
dto steht für *DataTranserObject*, welche in diesem Fall für den Datenaustausch zwischen Klassen verwendet werden. DTOs werden nicht persistiert und existieren nur zur Laufzeit.

app/filters *Bussiness-Logik-Layer*
Beinhaltet Filter, die in der Request-Chain durchlaufen werden.

app/models *Data-Access-Layer*
Enthält Domain-Objekte, die in dieser Form auf der Datenbank persistiert werden.

app/services *Bussiness-Logic-Layer*
Service-Klassen, die komplexe Abläufe abhandeln, werden hier eingeordnet.

public *Presentation-Layer*
Bildet zusammen mit den Scala-Controllern den Presentation-Layer der Server-Applikation. Im *public*-Ordner enhaltene Dateien bilden die Web Administrationsoberfläche. Umfasst JavaScript-, CSS- und HTML-Dateien.

Kapitel 3

UI, Design & Usability

3.1 Konzept

Das Design der Applikation orientiert sich an den Material-Design Guidelines von Google und verwendet deren Komponenten zu den beschriebenen Zwecken und gemäss denn spezifizierten Richtlinien. Eine Ausnahme bildet die Tactics-Board Ansicht (F02), wo *FloatingActionButtons* verwendet werden, um Spieler und Bälle zu symbolisieren.

3.2 Farbgebung

Um den Wiedererkennungswert der Applikation zu steigern, wird das Standard-AppTheme auf das folgendes Schema angepasst:

3.2.1 Primärfarbe

Legt die Grundfarbe des Applikations-Themes fest, betrifft unter anderem Status- und Actionbar.

Normal	**Dunkel**	**Medium**	**Hell**
#009688	#00796B	#4DB6AC	#B2DFDB

3.2.2 Akzentfarbe

Akzentfarben betreffen Input-Elemente wie Buttons, Textfelder und Checkboxen. Aktive Elemente werden durch den helleren Farbton gekennzeichnet.

Normal	**Hell**
#FF5722	#FF8A65

3.2.3 Textfarbe

Primär	**Sekundär**
#212121	#727272

3.2.4 Hintergrund

Zusätzlich zum Standard-Android Hintergrund ist die Farbgebung der Taktik Editier- und Abspiel-Ansicht aufgelistet.

Applikation	**Taktik**	**Spielfeld**
#E0E0E0	#FFFFFF	#81D4FA

3.2.5 Spieler-Tokens

Für Spieler- und Ball-Tokens wurden Farben mit einem starken Akzent zum Spielfeldhintergrund gewählt. Die Farbe einzelner Tokens kann danach vom Benutzer festgelegt werden.

Spieler	**Ball**
#3F51B5	#000000

3.3 Icons und Bildsprache

In der Applikation werden Icons als Metaphern für Schaltflächen verwendet. Für diese wurden, wenn immer möglich, dem Benutzer bekannte Android-Icons verwendet. Für applikationsspezifische Aktionen (Schiessen, Passen) wurden eigene Icons auf Basis der Android-Icons erstellt.

3.4 Usability

Um den Benutzern eine angenehme User-Experience zu bieten, wurden diverse Vorkehrungen und Entscheidungen getroffen.

3.4.1 Tests

Ein Testing für das User-Interface wird benötigt, um die Praxistauglichkeit der erarbeiteten Konzepte auf die Probe zu stellen.

UI-Tests

Unter dem Begriff "UI-Tests" sind automatisierte Tests zu verstehen. In der Android-Entwicklerszene findet das Framework *Espresso* für diese Zwecke starken Anklang.

Ein Testing dieser Art wird für das Floorball Tactics-Board im Verlaufe der Bachelorarbeit als nicht sinnvoll erachtet, weil

- die Benutzeroberfläche noch zu grossen Änderungen unterliegt und dieselben Tests gegebenenfalls mehrfach geschrieben werden müssen
- das Schreiben derartiger Tests für das Taktik-Board aufgrund komplexer Komponenten und Events viel Zeit in Anspruch nimmt

Als alternative Test-Methode wurden manuelle Tests erfasst und in Anhang A mitgeliefert. Mit Android Studio Version 2.2 , das sich Stand 10.06.2016 noch in der Beta-Phase befindet, können Tests während der Ausführung aufgenommen werden, womit sich eine Möglichkeit zur Automatisierung der System-Tests bietet.

Usability-Tests

Usability-Tests wurden während der Entwicklung des Prototyps, besonders aber auch im späteren Verlauf des Projekts informell durchgeführt. Ergebnisse der Tests wurden jeweils mit dem Betreuer in wöchtentlichen Meetings besprochen und sind in den Sitzungs-Protokollen [1] und -Traktanden [2] festgehalten.

3.4.2 Sehstörungen

Um Benutzern mit Sehstörungen ein uneingeschränktes Benutzererlebnis zu ermöglichen, wurde bei der Wahl der Standard-Farben auf ein gutes Kontrastverhältnis geachtet.

Überprüft wurden die Farben über die in Android integrierten Entwickler-Tools, mit denen eine Auswahl verschiedener Sehschwächen emuliert werden kann.

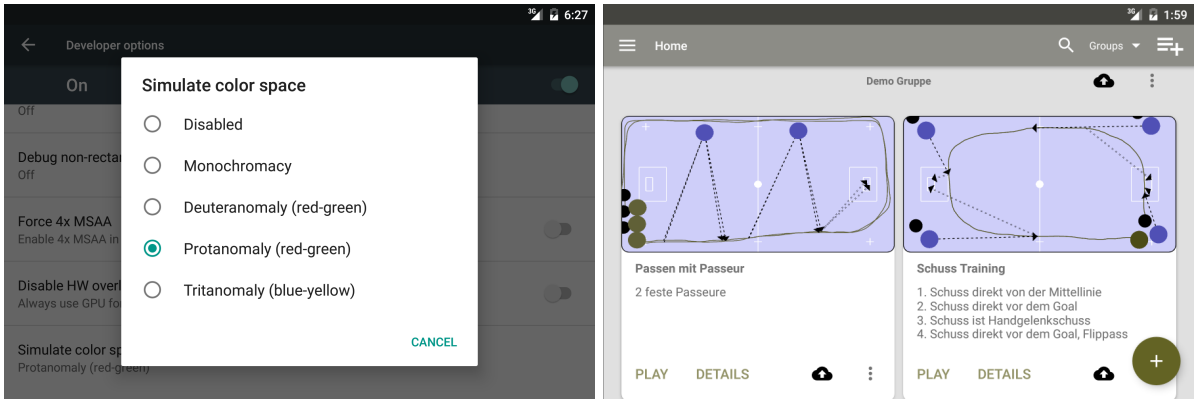


Abbildung 3.1: Android Farbprofile und Taktikübersicht mit Rot-Grün-Sehschwäche

3.4.3 Skip-Features

Die Taktik-Verwaltung funktioniert vorallem durch die Vergabe von Namen, Beschreibungen und deren Einordnen. In der traditionellen Taktik-Planung können diese Informationen ebenfalls vergeben werden, so zum Beispiel in einer Taktik-Sammlung. Öfter ist es jedoch der Fall, dass Taktiken nur kurz im Training aufgezeichnet werden, ohne dass derartige Informationen spezifiziert werden müssen - schon gar nicht vor dem eigentlichen Aufzeichnen.

Floorball Tactics-Board nähert sich diesen Abläufen an, indem es bei der Namens- und Beschreibungsvergabe ein "Überspringen"-Feature anbietet. Mit diesem werden Standard-Werte in die Felder eingefügt, und es kann direkt in den Bearbeitungsmodus der Taktik gesprungen werden.

Titel und Informationen können, wenn gewünscht, im Nachhinein durch die Editier-Funktion angepasst werden.

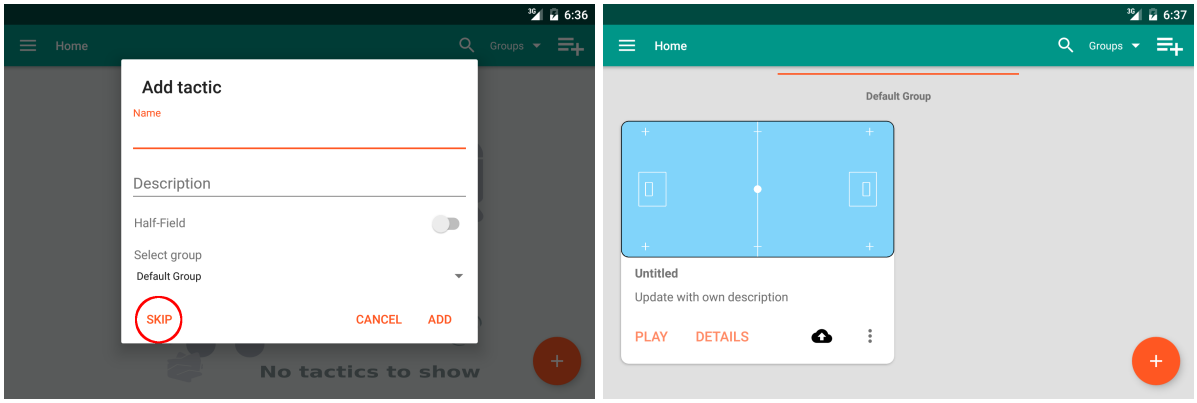


Abbildung 3.2: Skip-Button beim Erstellen einer Taktik und die unbenannte Taktik in der Übersicht

3.4.4 First-Start Anleitung und User Manual

Um dem Benutzer beim ersten Start eine Übersicht zu bieten, welche Elemente zur Verfügung stehen und wie diese benutzt werden, wird ein kurzes Tutorial angezeigt, das auch übersprungen werden kann. Für genauere Informationen steht ein Benutzerhandbuch unter "Help" in der App zur Verfügung. Siehe auch Anhang E.

3.5 Navigationskonzept

Um dem Benutzer die Navigation zwischen den Views und Activities zu erleichtern, braucht es ein ausgearbeitetes Konzept. Dieses hält sich an die Anleitung auf der Android Developer Seite [11].

3.5.1 Entity-Relation-Diagram

Zeigt die Beziehungen zwischen dem Benutzer und verschiedenen Elementen der App. Aus dem Entitiy-Relation-Diagramm können weitere Schlüsse in Bezug auf Benutzerinteraktionen gezogen werden.

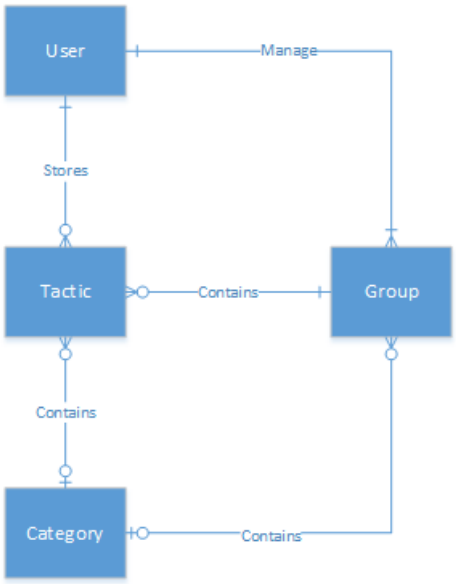


Abbildung 3.3: Entity-Relation-Diagramm

3.5.2 Back Navigation

Up-Button

Der Up-Button (meist in der Actionbar links) navigiert in der hierarchischen Anordnung der Frames nach oben (näher zum Homescreen). Element 1 in Abbildung 3.4.

Back-Button

Der Back-Button navigiert in der Screenhistory (Frame-Stack) rückwärts. Element 2 in Abbildung 3.4.

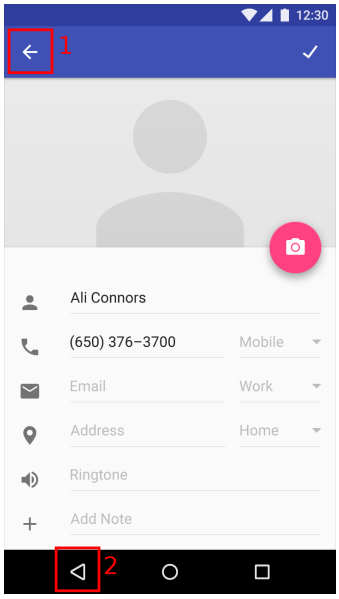
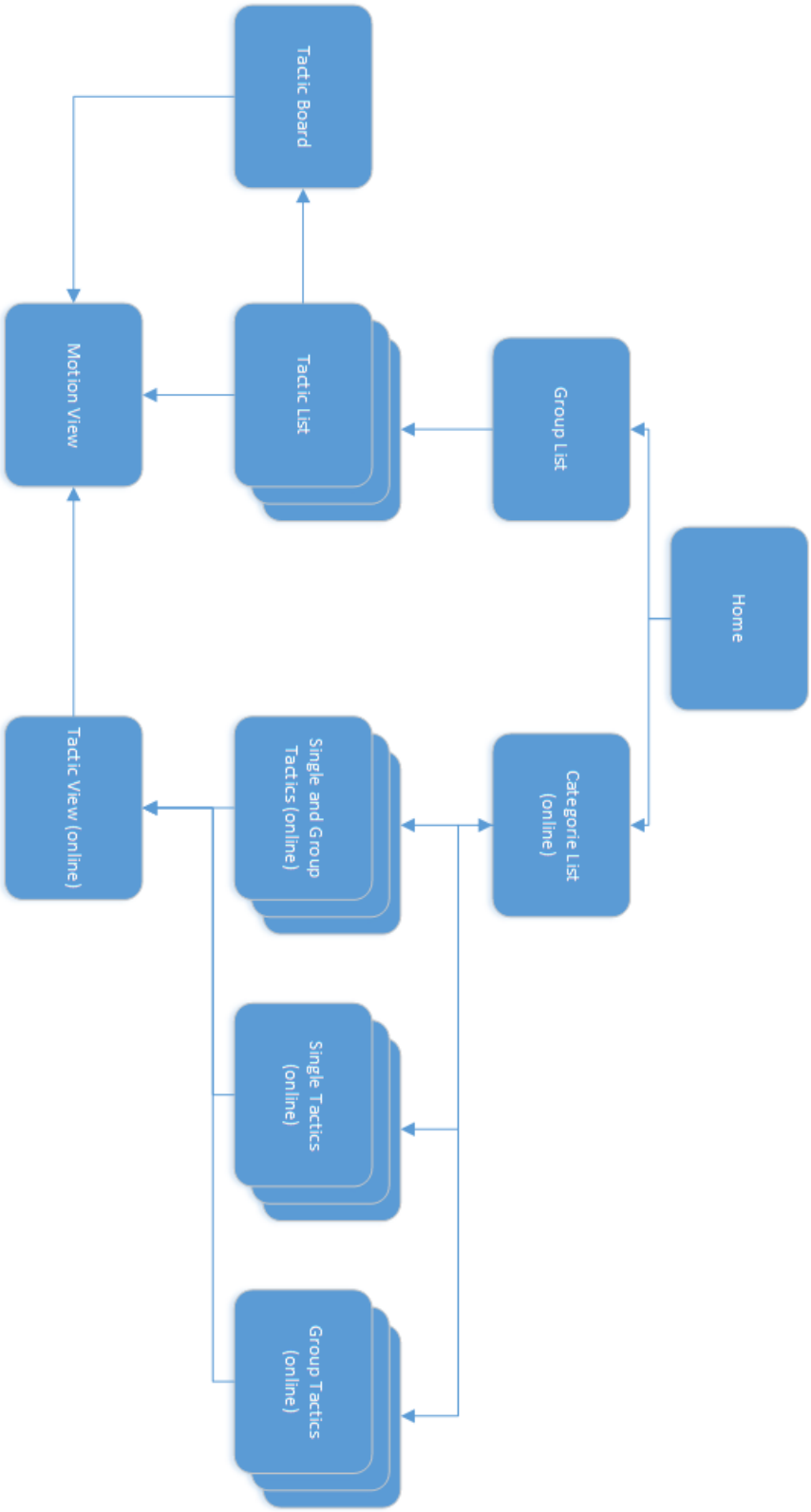


Abbildung 3.4: Android Bildschirm

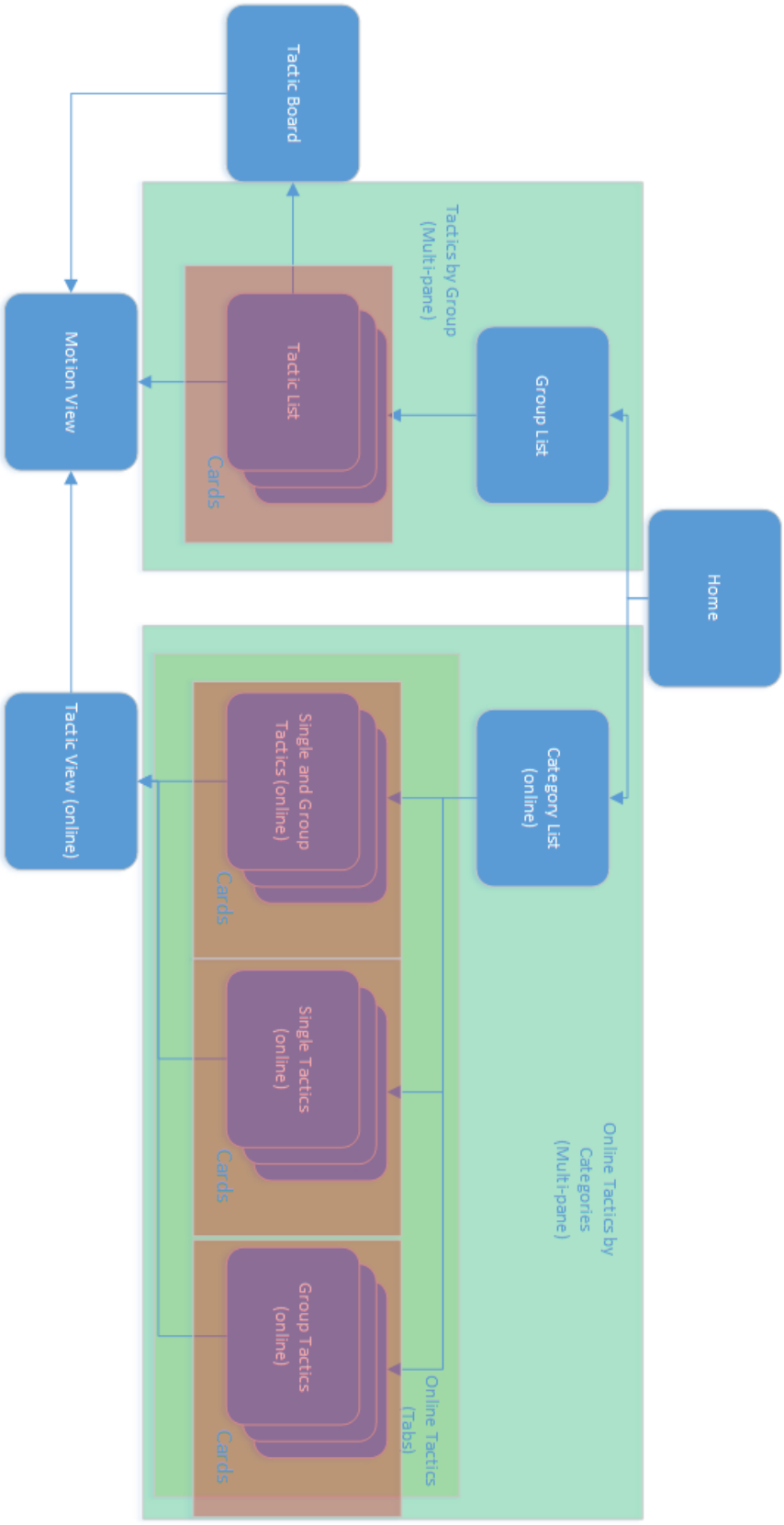
3.5.3 Screen-Relation-Diagram

Zeigt die verschiedenen benötigten Screens, welche für die App und deren grobe Navigation notwendig sind. Das Diagramm ist versteht sich als Anhaltspunkt, um Navigationselemente ausfindig zu machen und wiederspiegelt nicht die genaue Navigation.



3.5.4 Screen-Map-Diagram

Zeigt erste Entscheidungen betreffend Navigation und UI-Elementen.



3.6 Wireframes

Die Wireframes bilden das Skelett der zu implementierenden Applikation. Sie verstehen sich als ein Leitfaden für die Umsetzung und nicht als anzustrebendes Ideal.

3.6.1 F01: Lokaler Taktikstore (Homescreen)

Übersicht über alle Taktiken, eingeteilt in ihre jeweiligen Gruppen. Verfügt über eine Komponente zum Filtern nach Name / Beschreibung oder zugeordneter Gruppe. Dient als Startbildschirm der Applikation.

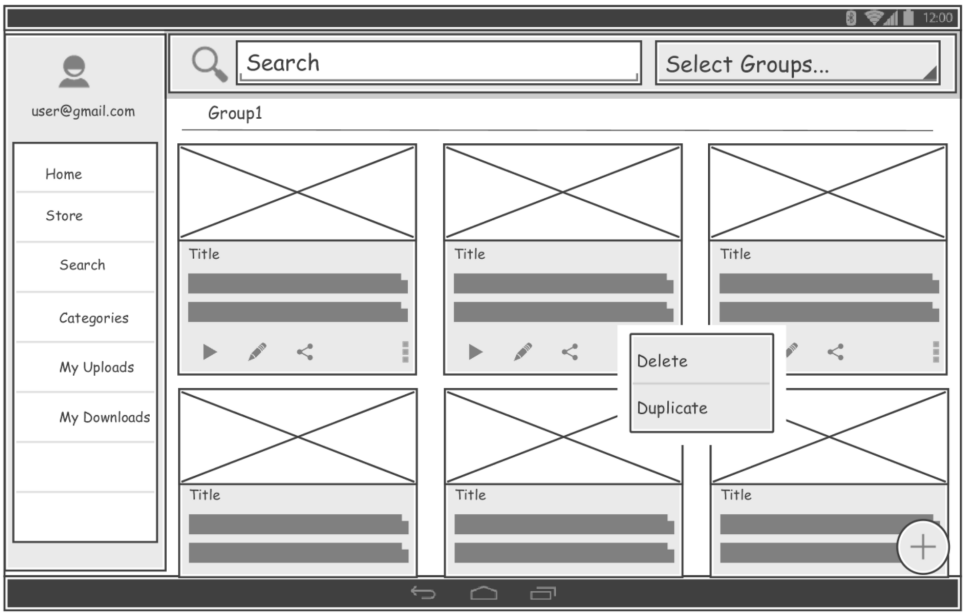


Abbildung 3.5: Lokale Taktikübersicht

Aktionen

Name	Navigationsziel
Up-Button	- (nicht vorhanden)
Neue Taktik erstellen	F02
Taktik abspielen	F03
Taktik bearbeiten	F02
Taktik teilen	F01.1 Dialog
Taktiken filtern	-
Taktik löschen	-
Taktik duplizieren	-

F01.1: Teilen-Dialog

Dialog für Teilen einer Taktik direkt aus dem Homescreen.

3.6.2 F02: Taktiktafel Ansicht

Ansicht der Taktik im Bearbeitungsmodus. Spielzüge können definiert und abgespielt werden.

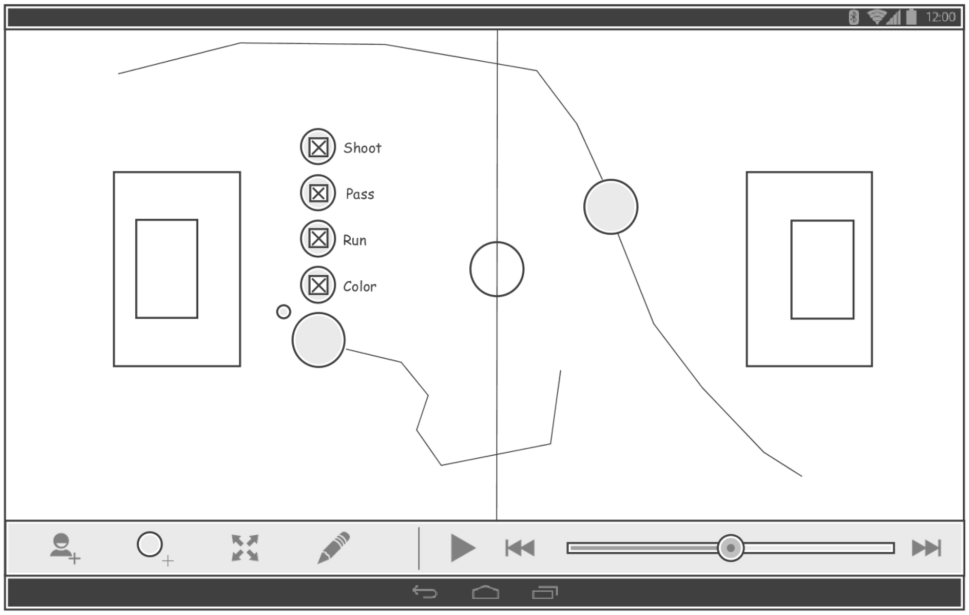


Abbildung 3.6: Taktiktafel Bearbeitungsmodus

Aktionen

Name	**Navigationsziel**
Up-Button	-(nicht vorhanden)
Spieler hinzufügen	-
Spieler bewegen	-
Spieler entfernen	-
Ball hinzufügen	-
Passen	-
Schiessen	-
Laufen mit Ball	-
Laufen ohne Ball	-
Taktik-Play abspielen	F03
Taktik-Play pausieren	-
Taktik-Play spulen	-
Ganzfeld umschalten	-
Halbfeld umschalten	-
Skizzenmodus benutzen	-

3.6.3 F03: Taktik abspielen

Die ausgewählte Taktik wird abgespielt. Entspricht der Ansicht F02 ohne Editier-Aktionen, verfügt demnach nur Interaktionsmöglichkeiten, um im zeitlichen Verlauf der Taktik zu navigieren.

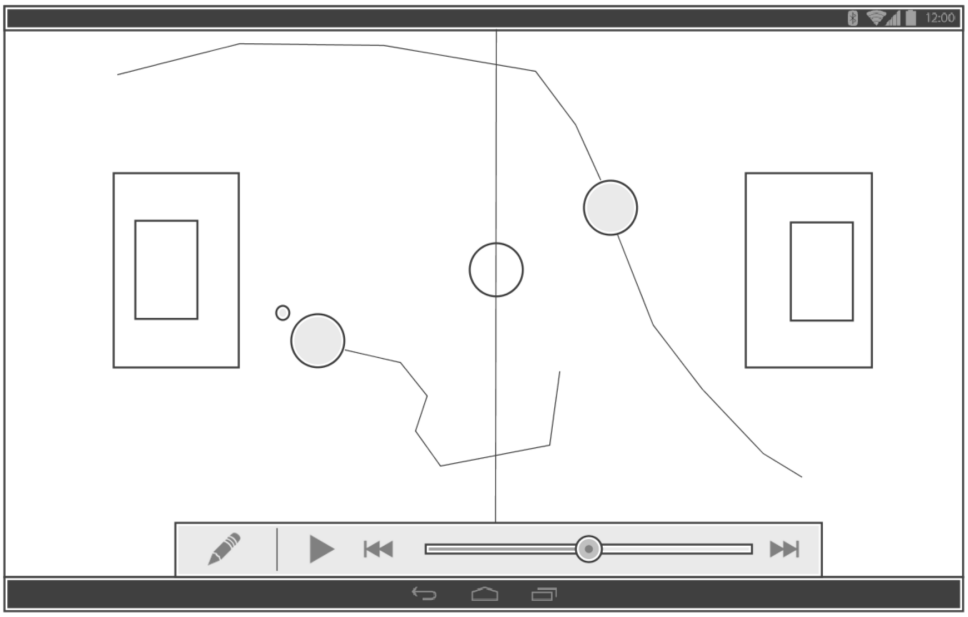


Abbildung 3.7: Taktik Abspielmodus

Aktionen

Name	**Navigationsziel**
Up-Button	-(nicht vorhanden)
Taktik-Play abspielen	-
Taktik-Play pausieren	-
Taktik-Play spulen	-

3.6.4 F04: Taktik-Store

F04.1: Taktiken suchen

Es kann nach Taktiken gesucht werden. Es besteht die Möglichkeit, nur nach Einzeltaktiken, nach ganzen Gruppen oder einer kombinierter Ansicht beider zu suchen.

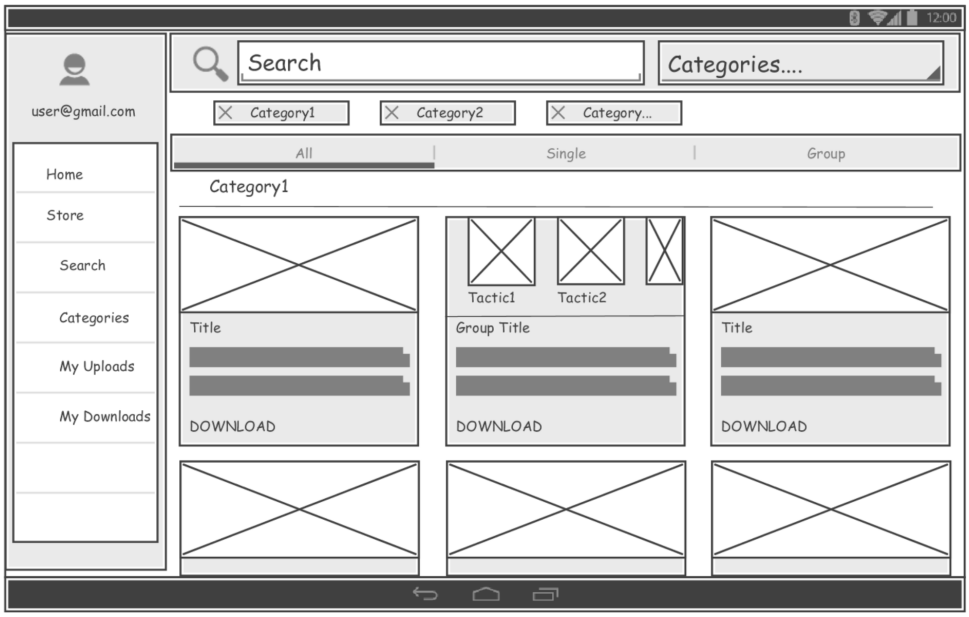


Abbildung 3.8: Taktik-Store Suche

Aktionen

Name	**Navigationsziel**
Up-Button	F01
Nach Taktiken suchen	-
Nach Kategorien Filtern	-
Taktik downloaden	-
Taktik- anklicken	F04.5
Kategorien-Filter entfernen	-
Tabs umschalten	-

F04.2: Kategorie Ansicht

Es werden alle verschiedenen Taktikkategorien angezeigt.

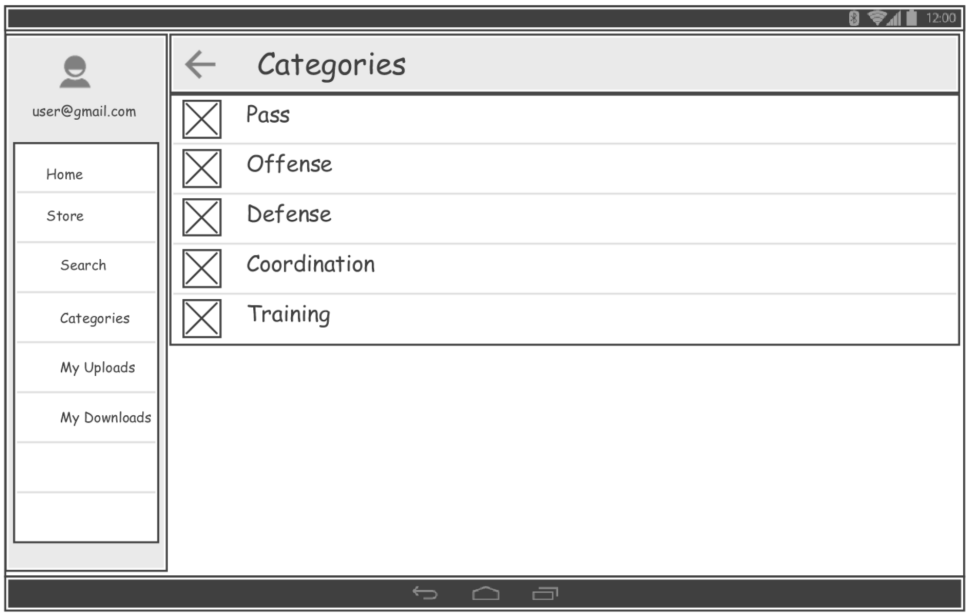


Abbildung 3.9: Kategorie-Übersicht

Aktionen

Name	**Navigationsziel**
Up-Button	F01
Kategorie auswählen	F04.1

F04.3: Meine geteilten Taktiken

Es werden alle Taktiken angezeigt, welche der Anwender anderen zur Verfügung gestellt hat.

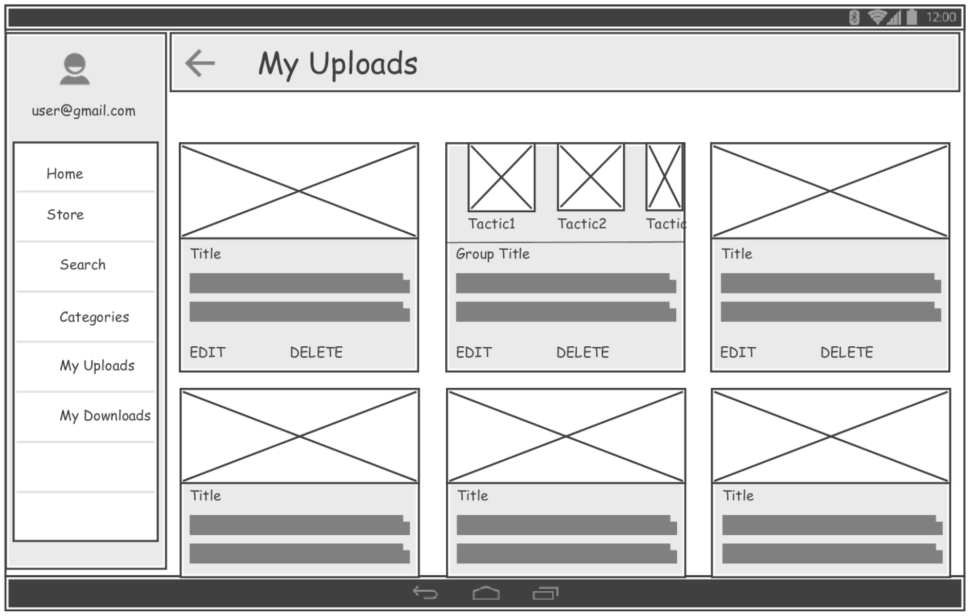


Abbildung 3.10: Übersicht Uploads

Aktionen

Name	**Navigationsziel**
Up-Button	F01
Taktik-Beschreibung editieren	Dialog
Taktik entfernen	-
Taktik uploaden	Dialog

F04.4: Meine bezogenen Taktiken

Es werden alle Taktiken angezeigt, welche der Anwender von anderen bezogen hat.

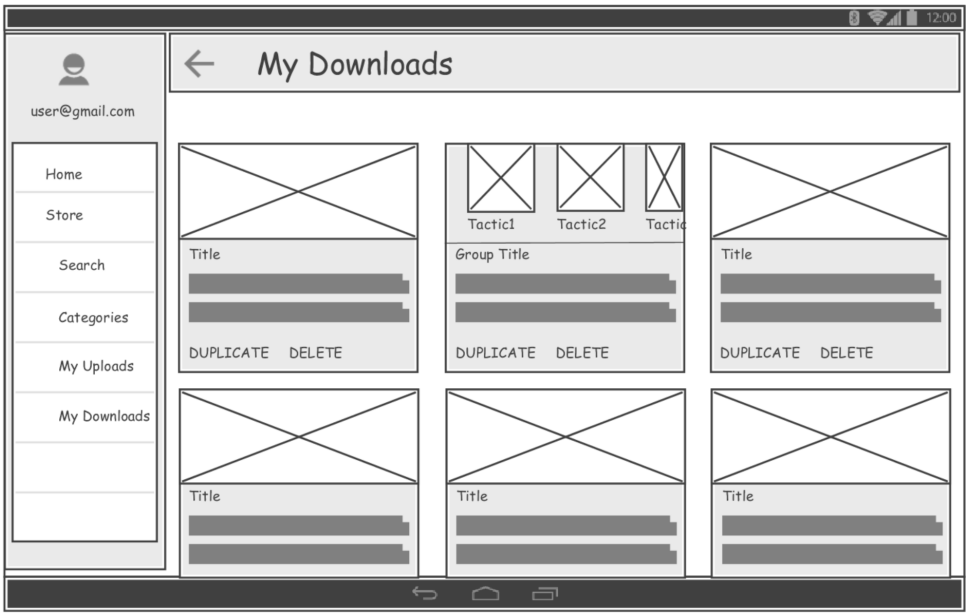


Abbildung 3.11: Übersicht Downloads

Aktionen

Name	**Navigationsziel**
Up-Button	F01
Taktik Duplizieren	-
Taktik entfernen	-
Taktik anklicken	F04.5

F04.51: Taktikvorschau einzeln

Vorschau einer einzelnen Taktik. Dazu werden Bilder und verschiedene Angaben angezeigt. Gehört eine Einzeltaktik zu einer ganzen Gruppe, kann zu dieser navigiert werden.

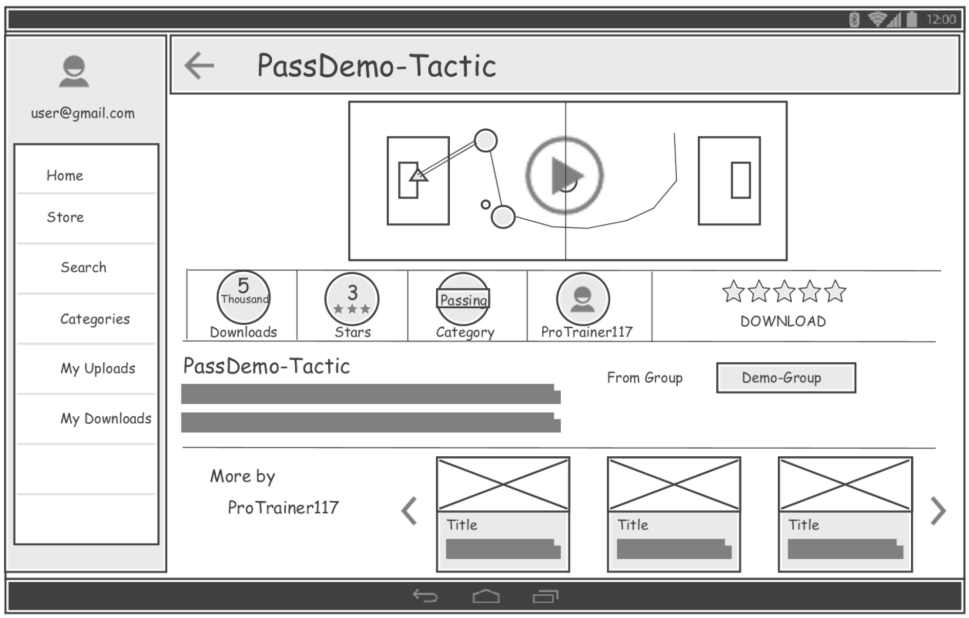


Abbildung 3.12: Vorschau Taktik

Aktionen

Name	**Navigationsziel**
Up-Buttonn	F04.1
Taktik abspielen	F03
Taktik downloaden	-
Taktik bewerten	-
Mehr von Uploader	F04.51/F04.52
Zur Gruppe	F04.52

F04.52: Taktikvorschau Gruppe

Vorschau einer einzelnen Taktik. Dazu werden Bilder und verschiedene Angaben angezeigt. Handelt es sich um eine Gruppenvorschau, so sind mehrere Vorschauen verfügbar.

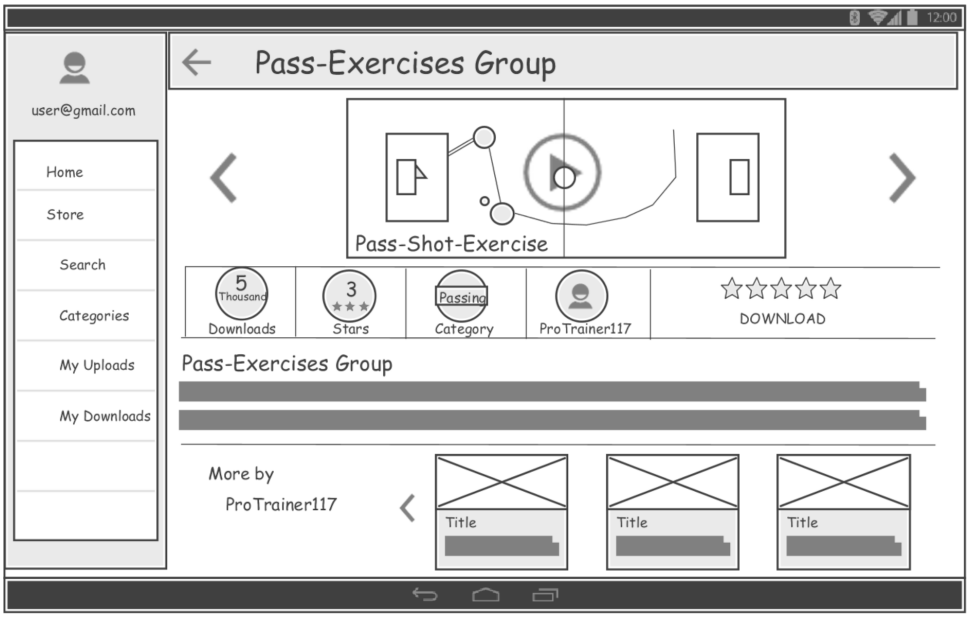


Abbildung 3.13: Vorschau Gruppe

Aktionen

Name	**Navigationsziel**
Up-Button	F04.1
Taktik abspielen	F03
Taktik downloaden	-
Taktik bewerten	-
Mehr von Uploader	F04.51/F04.52
Einzelne Taktik anzeigen	F04.51

Kapitel 4

Projekt-Management

4.1 Meilensteine

Nr	Name	Datum geplant	Datum ist
M01	Entwurf des Projektplans erstellen	02.03.2016	01.03.2016
M02	Infrastruktur und Entwicklungswerkzeuge eingerichtet	04.03.2016	04.03.2016
M03	Requirements Engineering abgeschlossen	09.03.2016	08.03.2016
M04	Android Prototyp lauffähig	16.03.2016	23.03.2016
M05	Server Prototyp lauffähig	16.03.2016	23.03.2016
M06	Kernfunktionen für Taktik Board implementiert	13.04.2016	27.04.2016
M07	Animation Taktiken lauffähig	27.04.2016	21.04.2016
M08	Taktik-Store implementiert	25.05.2016	27.05.2016
M09	Feature freeze	01.06.2016	02.06.2016
M10	Abgabe	17.06.2016	17.06.2016

4.2 Zeitaufwand

Insgesamt wurden bis Mittwoch, 15.06.2016, 932.45 Stunden in Redmine geloggt.

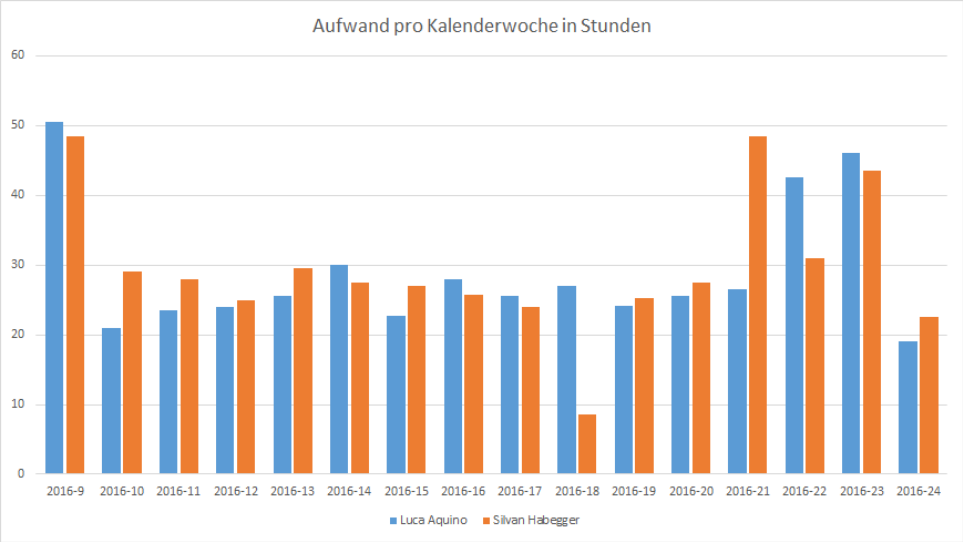


Abbildung 4.1: Zeitaufwand Floorball Tactics-Board

4.3 Risikomanagement

Bemerkung: Tabelle ist von KW9 (zweite Projekt-Woche). Neubeurteilungen wurden durchgeführt, sind hier jedoch nicht aufgelistet.

Name	**Beschreibung**	**Eintritts-W'keit**	**Schaden**	**Wert**	**Massnahmen**
Design nicht umsetzbar	Das definierte Design der Screens ist mit Android nicht umsetzbar.	10%	10h	**2h**	• Prototyping der wichtigsten grafischen Designelemente • Wenn möglich reine Android-Komponenten verwenden, keine eigenen Designelemente • Einhalten der Material-Design-Patterns
Key-Feature nicht umsetzbar	Definieren und Abspielen von Spielzügen ist nicht möglich	5%	50h	**2.5h**	• Prototyping der wichtigsten Features
Zeitliche Begrenzung	Definierte Features lassen sich aus Zeitgründen nicht umsetzten	20%	40h	**8h**	• Genügend Zeitpuffer im Projektplan einrechnen • Frühes Erfassen von Zeitverschiebungen verbunden mit Anpassung des Projektplans
Library benutzen	Eingesetzte Libraries verursachen Zusatzaufwand, da sie nicht gut definiert / umgesetzt sind	25%	30h	**7.5h**	• Proof of Technology für die wichtigsten Libraries durchführen • Die wichtigsten Libraries in Prototyp einbinden
Benutzerbedürfnisse verpassen	Es wird ein App entwickelt, das niemand braucht	15%	200h	**30h**	• App nicht mit unnötigen Funktionen aufblähen; schlicht halten • Einheitliches und übersichtliches Design konzipieren • Intuitive Navigation konzipieren • Requirements einem echten Trainer zur Kontrolle geben
TOTAL	-	-	-	**40h**	-

Kapitel 5

Ergebnisse

5.1 Zielerreichung

Die Resultate dieser Arbeit umfassen eine Android- und eine Server-Applikation, die zusammen den Funktionsumfang der ausgearbeiteten User-Stories abdecken. Im Verlauf des Projektes konnten weitere Erkenntnisse hinzugewonnen werden, was teilweise zu Anpassungen der Prioritäten führte und Auswirkung auf Details der Use Cases hatte.

Detaillierte Resultate und Umfang der Änderungen an Use Cases ist den Kapiteln 5.3, 5.4 und 5.5 zu entnehmen.

5.2 Änderungen Navigationskonzept

Bei der Erstellung der Wireframes stellte sich der Einsatz eines Navigation-Drawers als die intuitivste Variante für die Navigation innerhalb der App heraus. Dadurch sind alle Views, ausgenommen das Taktik-Board, direkt über den Drawer erreichbar. Bei der Benützung der App wurde festgestellt, dass ein “Stacking“ der Views und die Navigation mit dem Up-Button nicht benötigt wird. Dies hatte Änderungen im Navigationskonzept zur Folge. Die Views werden nicht mehr “gestackt“ und der Up-Button (siehe 3.5.2) wird überflüssig.

5.3 Abgedeckte Use Cases

Übersichtshalber werden die abgedeckten Use Cases, wo sinnvoll, gruppiert behandelt.

5.3.1 Taktik CRUD

Use Cases: UC101, UC102, UC103, UC105, UC405, UC406

Die Verwaltung von Taktiken wird im Homescreen vorgenommen. Alle Aktionen sind im Menu der jeweiligen Taktik verfügbar. Gruppenzuordnungen können direkt im Erstell- oder Bearbeitungsdialog vorgenommen werden, weshalb UC405 - UC406 unter diesem Kapitel aufgelistet sind.

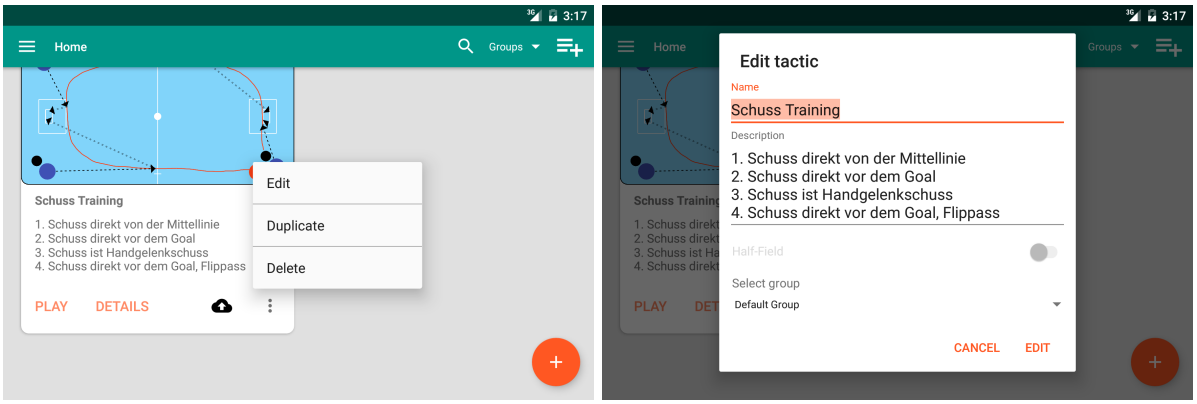


Abbildung 5.1: Taktik-Aktionen und Editier-Dialog in der lokalen Taktikübersicht

Änderungen: Um Wartezeiten zu verkürzen, wurden Dialoge an Stelle einer separaten Activity für das Bearbeiten und die Detailansicht von Taktiken verwendet.

5.3.2 Gruppen CRUD

Use Cases: UC401 - UC404

Als Ordnungseinheit von Taktiken sind die Gruppen-Verwaltungsfunktionen ebenfalls in der lokalen Taktikübersicht eingeordet. Sie verfolgen das Ziel, eine bessere Struktur in die Bibliothek zu bringen und können als eine Einheit in den Store gestellt werden.

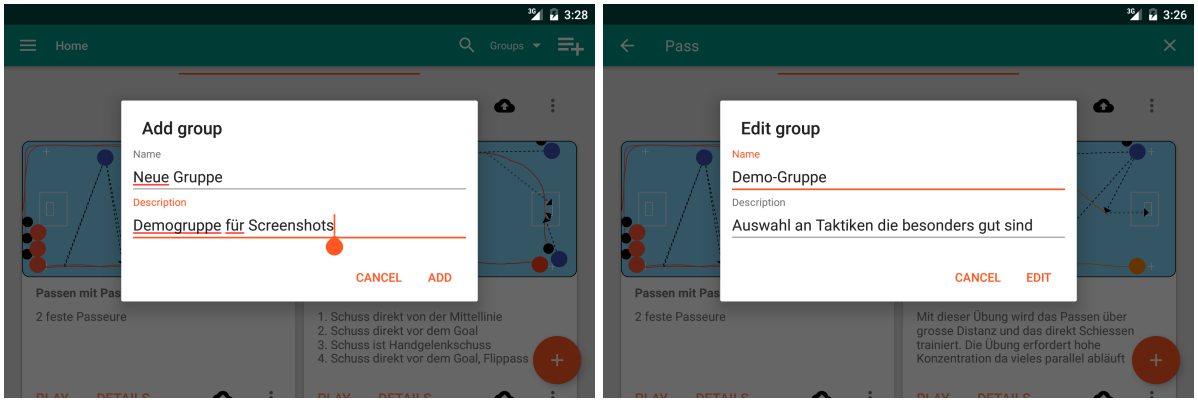


Abbildung 5.2: Erstell- und Bearbeitungs-Dialog für Gruppen

5.3.3 Taktik suchen

Use Cases: UC104
Wireframes: 3.6.1

Der Benutzer kann die lokal vorhandenen Taktiken und Gruppen durchsuchen und nach Gruppe filtern. Suchresultate sind weiterhin als Unterelement ihrer entsprechenden Gruppe angezeigt.

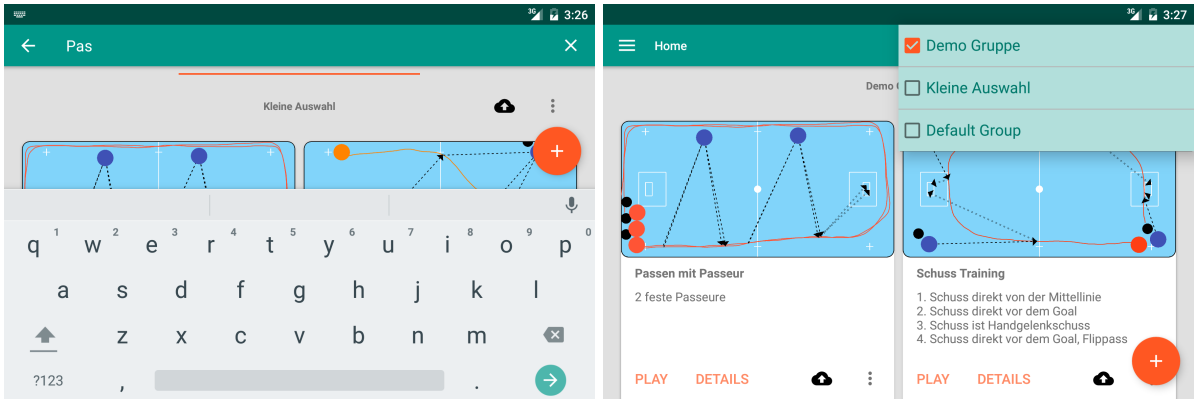


Abbildung 5.3: Textsuche und Gruppenfilter in der lokalen Taktikübersicht

5.3.4 Taktik-Board

Use Cases: UC201 - UC202; UC301 - UC319
Wireframes: 3.6.2 und 3.6.3

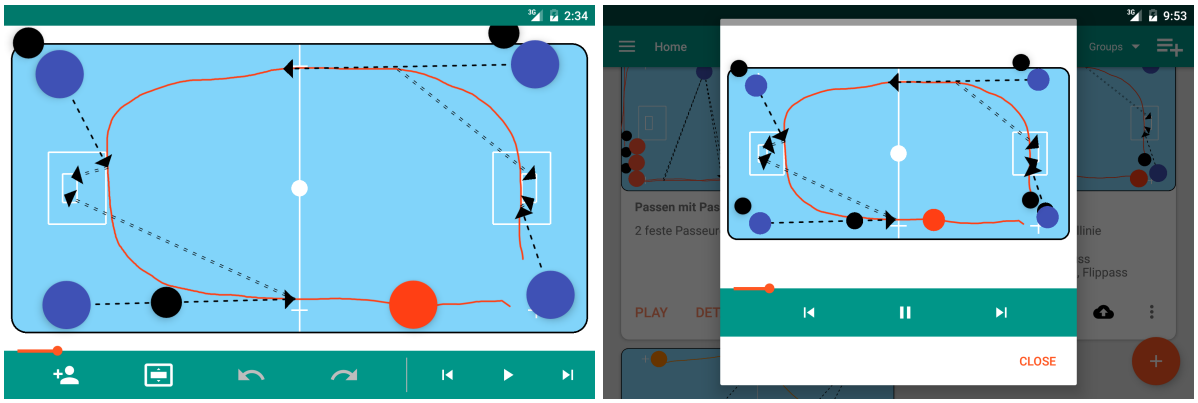


Abbildung 5.4: Taktik-Board und Playback-Ansicht

Änderungen: Das Taktik-Board, eine der Kernfunktionen der Applikation, unterliegt wegen Verfeinerung der Anforderungen innerhalb des agilen Projektrahmens den meisten Veränderungen. Um die Benutzerfreundlichkeit zu erhöhen, wurden im Vergleich zu Wireframe 3.6.2 folgende Änderungen umgesetzt:

- Abspiel-Ansicht in Dialog anstatt eigener Activity implementiert
- Schüsse und Pässe nur auf gültige Ziele möglich (Kollissionsdetektion)
- Bälle werden direkt dem Spieler, nicht dem Spielfeld, zugewiesen
- Bälle sind passive Objekte, deren Aktionen auf dem ballbesitzenden Spieler ausgelöst werden müssen
- “Undo- / Redo-Buttons“ für Spieler-Bewegungen (anstatt einfacher Löschaktion)
- Halbfeld- / Ganzfeldansicht muss beim Erstellen einer Taktik definiert werden
- Zusatzfeature Spielfeldskalierung 5.4.2

5.3.5 Taktik- / Gruppen-Upload

Use Cases: UC501 - UC502
Wireframes: 3.6.4

Taktiken und Gruppen können in den Store gestellt werden.

Die Aktion ist an das Cloud-Icon in der Taktikübersicht gekoppelt. Dieses findet als Status-Visualisierung und gleichzeitig als Auslöser der Upload-Aktion Verwendung.

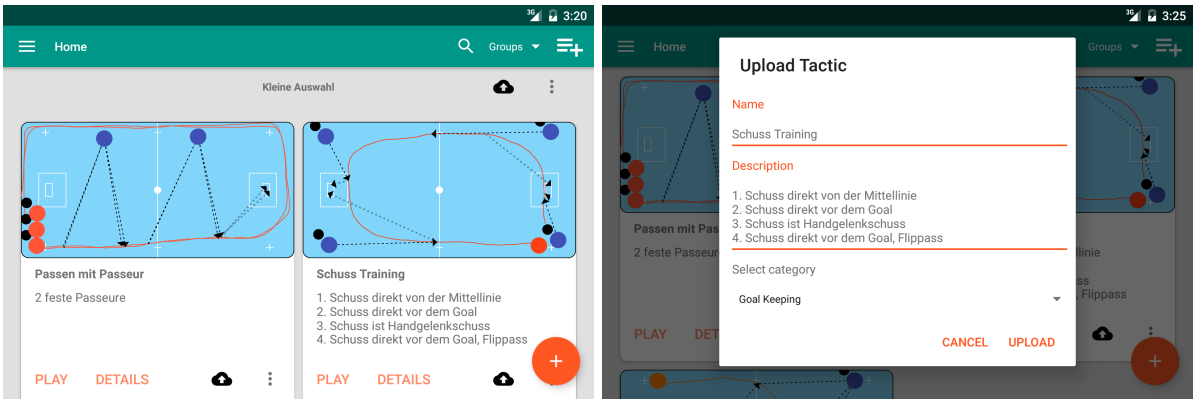


Abbildung 5.5: Upload-Buttons für Taktiken und Gruppen mit Upload-Dialog-Beispiel

Änderungen: Anders als im verknüpften Wireframe gibt es keine "My-Uploads"-Ansicht. Diese Funktion wurde direkt in die Taktikübersicht integriert. Dazu wurde, entgegen den ursprünglichen Anforderungen, definiert, dass es nicht möglich ist, einen Store-Eintrag zu bearbeiten.

5.3.6 Taktik- / Gruppen-Download

Use Cases: UC503 - UC504
Wireframes: 3.6.4

Taktiken und Gruppen können über den Store heruntergeladen werden. Wird diese Aktion ausgeführt, so sind die gewünschten Taktiken in der Ansicht "Downloads" verfügbar, wo Detailansicht, Abspielmodus und Duplizier-Funktion auslösbar sind.

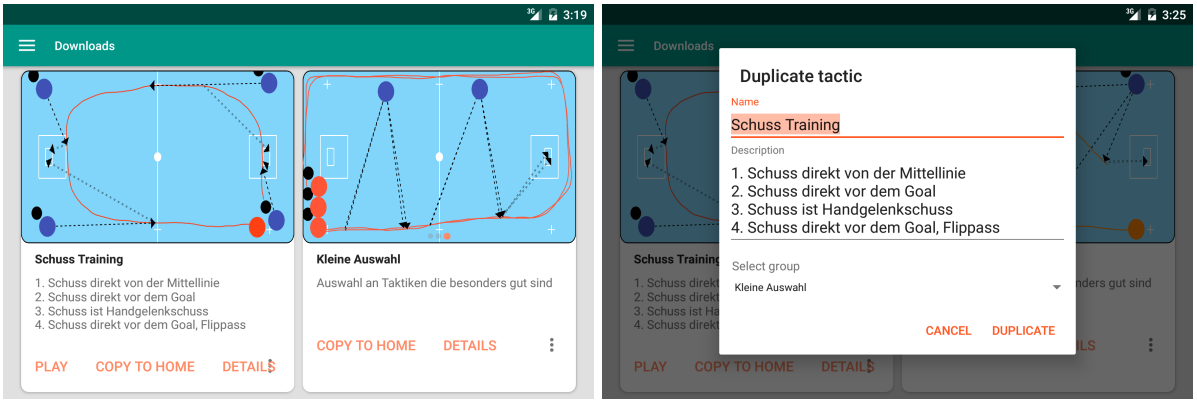


Abbildung 5.6: Downloads-Übersicht mit Duplizier-Dialog (für das Kopieren einer Taktik oder Gruppe in die lokale Übersicht)

5.3.7 Store

Use Cases: UC501 - UC502
Wireframes: 3.6.4

Der Store bietet den Anwendern der Applikation die Möglichkeit, Taktiken auszutauschen. Store-Einträge können analog zur lokalen Taktikübersicht durchsucht und gefiltert werden. Zudem wird über eine Einteilung in Tabs eine Möglichkeit geboten, die Resultate auf Gruppen oder einzelne Taktiken einzuschränken.

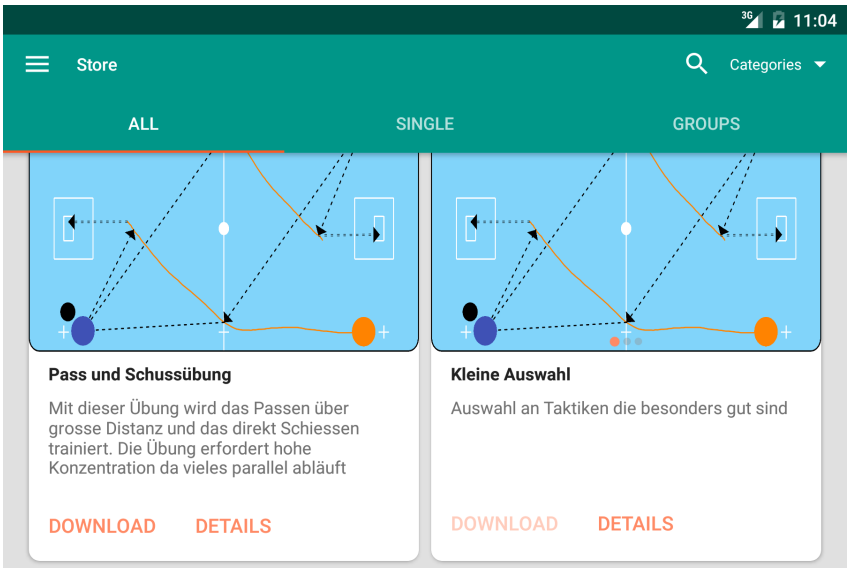


Abbildung 5.7: Taktiken und Gruppen im Online-Store

5.3.8 Detailansicht Taktik / Gruppe

Wireframes: 3.6.4 und 3.6.4

Ist eine Beschreibung oder ein Titel zu ausführlich für die einfache Card-Ansicht, so können die Informationen einer Taktik oder Gruppe in der Detailansicht angezeigt werden. Dies gilt sowohl für die lokale Übersicht, als auch für Einträge im Online-Store und der Downloads-Übersicht.

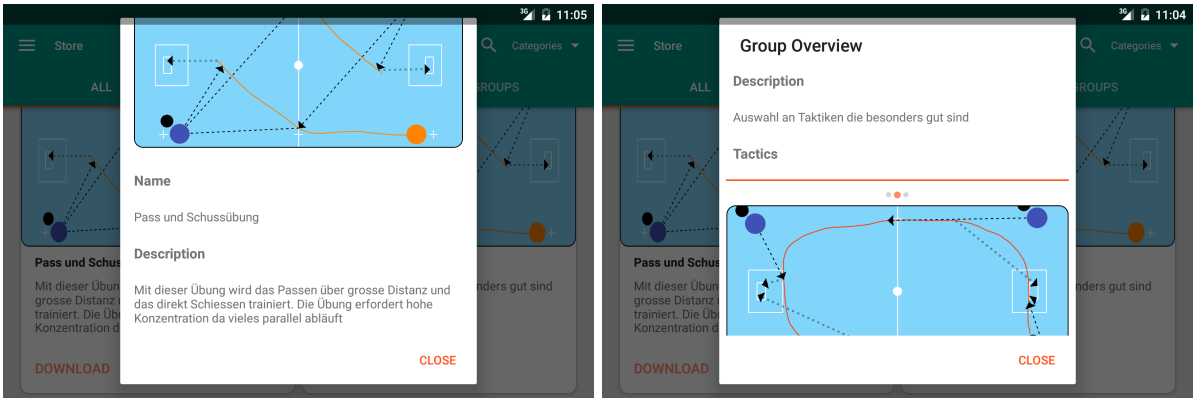


Abbildung 5.8: Detailansicht Taktik und Gruppe

Änderungen: Die Wireframes verwenden die Detailansicht (separate Activity) lediglich im Store und stellen zusätzliche Informationen wie die Anzahl Downloads oder den Ersteller dar. Die implementierte Detailansicht verzichtet auf diese Komponenten und verwendet die Detailansicht in mehreren Kontexten als Dialog.

5.3.9 Admin Web-UI

Use Cases: UC601 - UC603

Die Web-Administrationsoberfläche bietet einfache Verwaltungsfunktionen, die primär der Bereinigung des Stores dienen. Die Authentifizierung findet, gleich wie in der Android-Applikation, über Google OAuth statt. Berechtigungen können von bestehenden Administratoren vergeben werden.

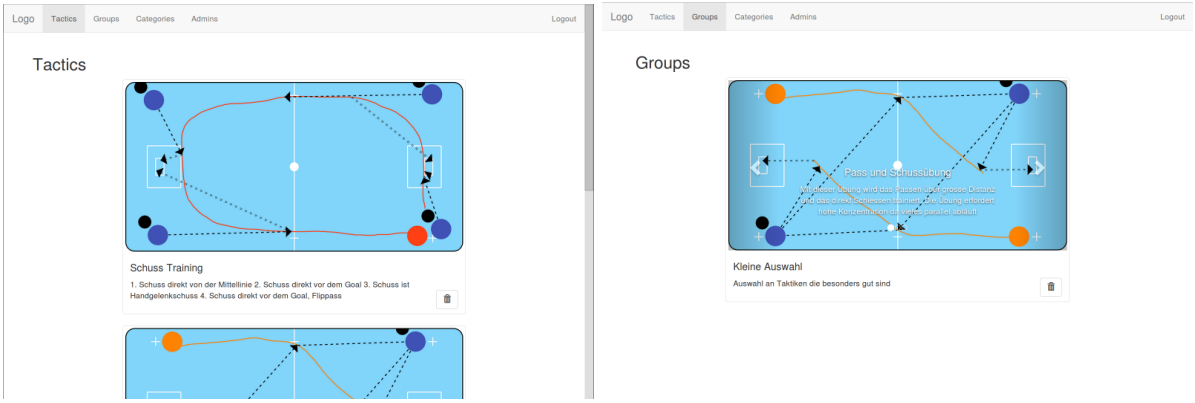


Abbildung 5.9: Taktik- und Gruppen-Administration

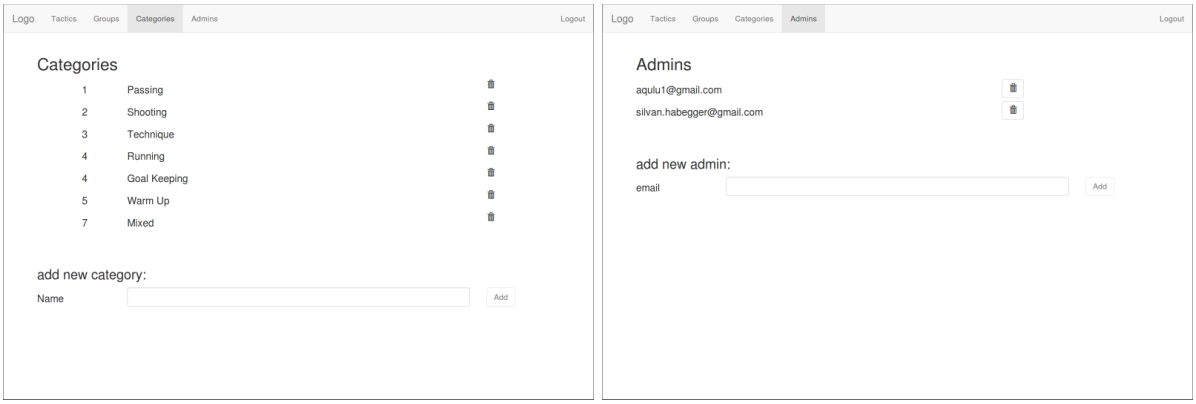


Abbildung 5.10: Kategorie- und Admin-Verwaltung

5.4 Zusätzlich implementierte Funktionen

Durch Anpassung des Fokus von Features oder zur Verbesserung der Usability wurden im Verlaufe der Arbeit mehere Funktionen implementiert, die nicht im Requirements-Engineering aufgetaucht sind.

5.4.1 UC1XX: Erweiterte Suchresultate

Kontext: Suche in der lokalen Ansicht
Beschreibung: Sind in der lokalen Taktikübersicht Gruppenfilter gesetzt, so sind die angezeigten Taktiken bereits eingeschränkt. Wird zusätzlich ein Suchbegriff eingegeben, der auch auf ausgeblendete Resultate zutrifft, so wird der Benutzer darauf hingewiesen, dass einige auf den Suchbegriff zutreffende Taktiken nicht angezeigt werden. Mit einem Klick auf den Hinweis werden diese temporär eingeblendet.

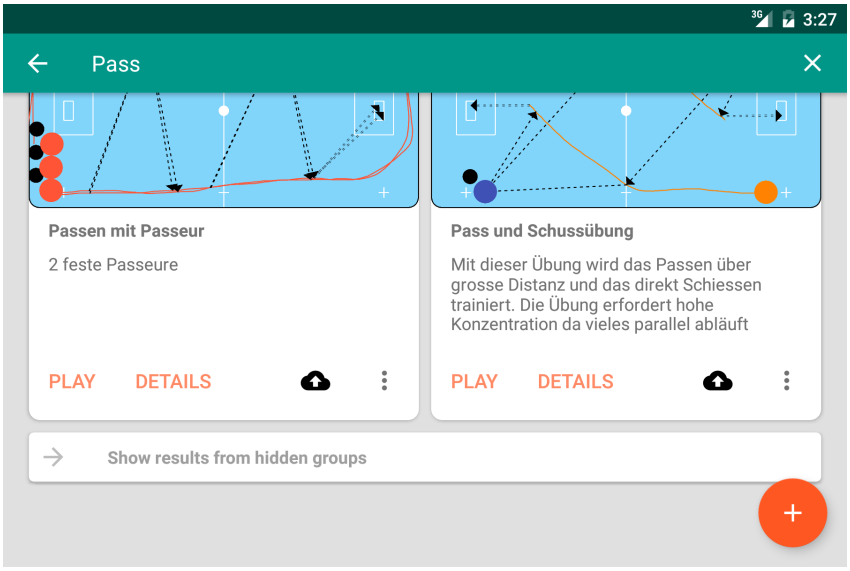


Abbildung 5.11: Hinweis auf versteckte Suchergebnisse

5.4.2 UC3XX: Spielfeld-Skalierung

Kontext: Taktik-Board
Beschreibung: Öffnet der Anwender das Taktik-Board, so wird standardmässig eine Tafel mit korrekten Proportionen angezeigt. Der Anwender kann mit dem Skalierungs-Button zwischen einem den Bildschirm ausfüllenden Feld und dem korrekt proportionierten Feld hin und her wechseln. Bereits definierte Spielzüge werden mitskaliert.

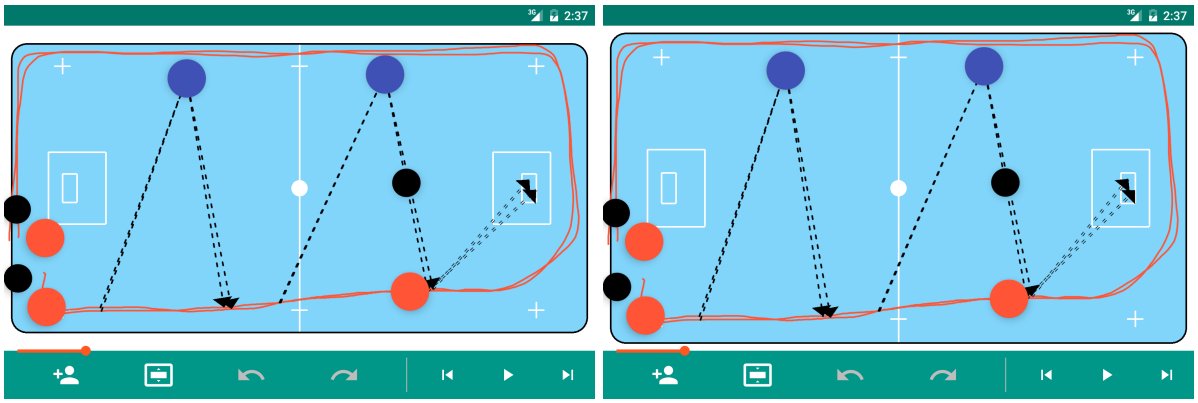


Abbildung 5.12: Feldskalierung Standard und Bildschirm füllend

5.5 Weggelassene Use Cases

Mit dem Anpassen von Funktionen ging oft auch das Weglassen eines bereits definierten Use Cases einher. Diese Entscheidungen sind mit Begründung hier aufgelistet.

5.5.1 UC306 & UC308: Passweg / Schussweg entfernen

Use Cases 306 und 308 wurden wegen Prioritätenänderung während des Projektes weggelassen. In vorgängigen Besprechungen wurde in nützlicher Zeit keine intuitive Lösung für das Entfernen eines Pass- oder Schusswegs gefunden.

Alternative Funktionen: Durch Löschen eines Balls werden auch dessen Schusswege entfernt. Nach dem Hinzufügen eines neuen Balls kann der Pass / Schuss neu definiert werden.

5.5.2 UC310, UC314 & UC15: Ball verschieben / Ballbewegung hinzufügen und entfernen

Mit der Entscheidung, Bälle als passive Objekte zu behandeln, fielen diese Features weg. Die Position eines Balls wird durch dessen besitzenden Spieler oder durch Ausführen eines Schusses / Passes definiert.

5.5.3 UC318 - UC319: Notiz hinzufügen und entfernen

Das schon zu Beginn als optional angedachte Feature, Notizen oder Freihandzeichnungen einer Taktik hinzuzufügen, wurde aus dem Projektscope weggelassen.

5.6 Problemlösungen & Erkenntnisse

Die Beschreibung der gelösten Probleme und der daraus gezogene Erkenntnisse sollen dazu dienen, während der Arbeit erarbeitetes Know-How festzuhalten.

5.6.1 Google-OAuth

Um die Benutzung unseres Server APIs auf die Android App einzuschränken und keine eigene Verwaltung von Benutzern umsetzen zu müssen, wird ein OAuth-Verfahren genutzt. Da die meisten Android Nutzer einen Google Account besitzen, gibt es nur eine kleine Einschränkungen der potentiellen Nutzer. Ein weiterer Vorteil ist, dass sich diese keinen zusätzlichen Account zulegen müssen. Unter Anhang B findet sich eine detaillierte Step-By-Step Anleitung, wie OAuth aufzusetzen ist. Unter Anhang C 1 sind Code-Beispiele zu finden.

Überblick Verfahren

- Der Benutzer meldet sich mit Betätigung des Login Buttons bei Google an. Die Login-Anfrage ist auf das App registriert. Anfragen ausserhalb des Apps werden nicht akzeptiert. Der Android-seitige Login-Prozess wird von einer Google Library durchgeführt.
- Bei erfolgreichem Login wird ein Token zurückgeliefert.
- Bei den geschützten Requests zum Backend wird dieser Token im Header mitgesendet.
- Die Server-Applikation verifiziert den Token über eine Google Library.
- Ist die Verifizierung erfolgreich, wird der Request bearbeitet. Ansonsten wird der HTTP-Statuscode "401 Unauthorized" zurückgegeben und im App erscheint der Hinweis, dass man sich anmelden muss.

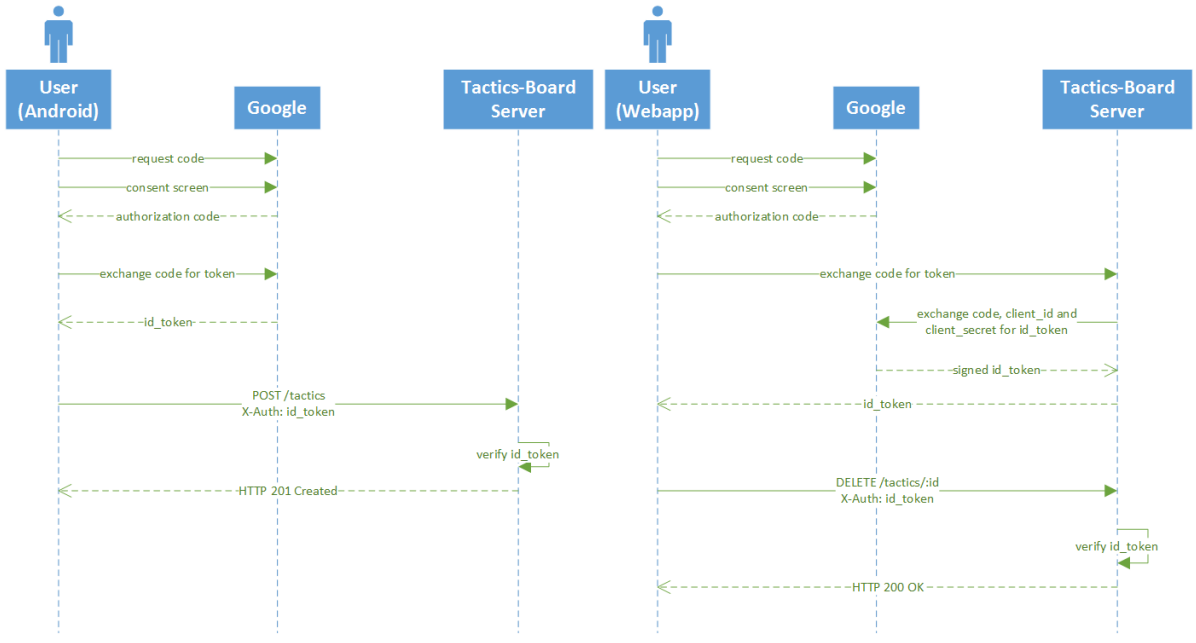


Abbildung 5.13: Google OAuth Android und Web-App

5.6.2 Upload / Download

Die Kommunikation mit dem Server ist mit den Libraries Retrofit, GSON und OkHttp gelöst. Retrofit erzeugt einen typsicheren Http-Client, der GSON als JSON-Serializer / -Deserializer und OkHttp zur Kommunikation nutzt.

JSON Serializer / Deserializer

GSON ist eine Google Library, die Java Objekte in das JSON-Format und zurück konvertiert. Der Vorteil in dessen Benützung liegt darin, dass für Klassen eigene Adapter registriert werden können. Diese erlauben das Konvertieren von verschachtelte Objekt-Hierarchien ohne grösseren Zusatzaufwand. Zusätzlich bietet es Gewissheit über die Namen der JSON Properties, da diese Information anzugeben ist. Unter Anhang C 2 sind Code-Beispiele zu finden.

Retrofit Api Service Creator

Mit Retrofit können die benötigten API-Calls typsicher als Interface angegeben werden. Die Implementation des Interface wird mit OkHttp erzeugt.

OkHttp Interceptor

Um den Token im Http-Header zu setzen, kann entweder eine Annotation "@Headers(..)" im API-Interface verwendet werden oder es wird ein OkHttp-Interceptor hinzugefügt, welcher die Requests abfängt und bearbeitet. Ein Code Beispiel ist unter Anhang C 4 zu finden.

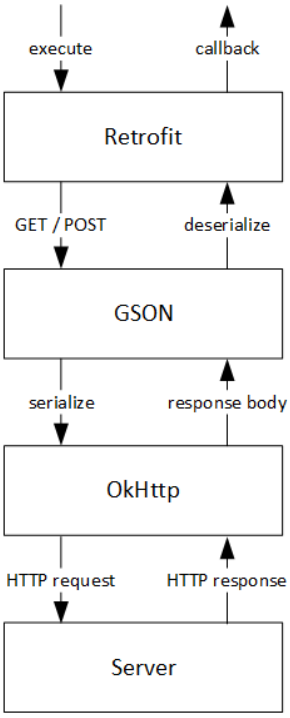


Abbildung 5.14: Retrofit Kommunikationsstack

```
@GET("tactics")
Call<List<TacticPreview>> getTacticPreviews(@Query("categoryId") Long categoryId,
    @Query("searchQuery") String searchQuery);
```

Beispiel eines API-Call Interfaces. Weitere Beispiele sind im Anhang C unter 3 zu finden.

5.6.3 Umgang mit Bilder

In Android ist es wichtig, auf den Memory-Gebrauch zu achten. Da es kein Memory-Swapping gibt, ist die einzige Möglichkeit, bei vollem RAM mehr Speicher zu allozieren, anderen freizugeben [15]. Dazu kann im Falle von Memory-Knappheit mittels LRU (Least Recently Used) Cache oder Weak-References dem Garbage Collector mitgeteilt werden, welche Objekte abgeräumt werden können.

Bitmaps im Memory

Speziell zu beachten sind dabei Referenzen auf Bitmaps[14]. Die Bilder sollten vor dem Speichern auf die gebrauchte Grösse zugeschnitten und zusätzlich komprimiert werden. Da das Komprimieren und Dekomprimieren von Bitmaps ein aufwendiger Prozess ist, sollte dies erstens im Hintergrund geschehen und zweitens möchte man die Bitmaps cachen. Dies muss über mehrere Stufen (Memory- und Disk-Cache) geschehen und darf nicht zu einer OutOfMemoryException führen [13]. Um diesen Anforderungen gerecht zu werden, wird die Library "Android Universal Image Loader" , kurz UIL, eingesetzt. Diese ist speziell dafür konzipiert, alle genannten Punkte umzusetzen. Normalerweise können dem UIL nur Internet-URIs übergeben werden, welche asynchron heruntergeladen, zugeschnitten und gecached werden. Mit einem eigenen UIL-Downloader, der ein String entgegen nimmt und ein ByteArrayInputStream zurück gibt, kann dieses Verfahren auch für das Laden lokale Bitmaps adaptiert werden. Da diese Aktionen asynchron ablaufen, muss ein Listener spezifiziert werden, der das Bitmap als Parameter erhält. Ergänzend muss dem Base64-String des Bitmaps die TaktikId getrennt durch den String "thumbnail" angehängt werden,

da es sonst beim Laden von zwei gleichen Bitmaps zu Komplikationen führen kann. Code Beispiele sind im Anhang C unter 5 zu finden.

Bitmap Up- & Download

Um bei Up- und Download Bandbreite zu sparen, wird das Bitmap als Base64-String kodiert als anstatt wie standardmässig vorgesehen als Byte-Array. Dadurch lässt sich die Grösse um ca. Faktor 10 verkleinern (eigene Beobachtung).

Vektor Grafiken

Alle fest installierten Drawables (ausgenommen der Screenshots für das User Manual) wurden, wenn immer möglich, als Vektorgrafiken gespeichert. Eine Vektorgrafik ist eine Beschreibung von Formen und Farben, die beliebig skaliert werden kann und dabei fast keinen Speicherplatz einnimmt. Dadurch verkleinert sich die App-Grösse erheblich. Vektorgrafiken können über die Material-Icons Seite [1] heruntergeladen und mit dem Vector-Asset-Studio importiert werden.

5.6.4 Multithreading in Android

Android verfügt über mehrere Modelle, wie Aktionen parallel ausgeführt werden können.

- *AsyncTask:*
  Ein vom UI-Thread unabhängiger Task wird gestartet. Primär modelliert für zeitintensive Tasks; im vorliegenden Fall werden Berechnungen ausgeführt, bis die innere while-Schleife durch das Setzen eines Modus terminiert wird.

  Nach Ausführung des primären Tasks wird die Methode *onPostExecute* ausgeführt, die auf dem UI-Thread läuft und so für abschliessende Arbeiten verwendet werden kann.

- *Handler:*
  Der "Handler" verfügt über eine Task-Queue, in die Instanzen des Typs *Runnable* eingereiht werden können. Dies kann sowohl direkt, über die Methode *post*, oder auch zeitverzögert, über Aufruf von *postDelayed* geschehen. Der Handler arbeitet diese Tasks nacheinander in einem vom UI entkoppelten Thread ab.

  Die Schleife innerhalb des Handlers wird über Rekursion gelöst. So wird, falls die Abbruch-Bedingung nicht erfüllt wurde, derselbe Task erneut in die Task-Queue eingereiht.

- *Timer:*
  Ein Timer entspricht einer while-Schleife, die um einen Initial- und einen Folge-Delay erweitert wurde und in diesen Zeitabständen die zugewiesenen *TimerTask*-Instanzen abarbeitet. Timer-Methoden werden standardmässig auf dem UI-Thread ausgeführt, können aber auch in einen Daemon-Thread ausgelagert werden.

  Der Timer führt die zugewiesenen Tasks aus, bis die Methode *cancel* aufgerufen wird.

Für das Floorball Tactics-Board werden zumeist AsyncTasks verwendet. Diese bieteen für normale vom UI entkoppelte Verwendungszwecke den grössten Komfort. Das Team behält sich die Möglichkeit vor, andere Task-Modelle einzusetzen, falls diese eine sauberere Lösung für den jeweiligen Aufgabenbereich bieten.

5.6.5 Update DB-Shema

Wird im GreenDaoGenerator ein Objekt aktualisiert, muss die Schema-Versionnummer erhöht werden. Dadurch wird das Schema-Update auf die SQLite-Datenbank übertragen, was aber zur Folge hat, dass alle bisherigen Einträge verloren gehen. Diesem Problem kann Abhilfe geschaffen werden, indem man die "onUpgrade" Methode des DaoMaster.OpenHelper überschreibt [17]. Dadurch wird eine temporäre DB

[1] https://design.google.com/icons/

erstellt, alle Einträge gespeichert, die eigentliche DB geupdatet, die Einträge der temporären DB in die originale zurückkopiert und die temporäre Datenbank gelöscht.

5.6.6 Object-Relational Mapping

Mit der Entscheidung, greenDAO als OR-Mapper einzusetzen, wurden gleichzeitig einige Unannehmlichkeiten in Kauf genommen. Es stellte sich heraus, dass greenDAO zum jetzigen Zeitpunkt kein Polymorphismus und auch keine bidirektionalen Verbindungen zwischen Entitäten unterstützt.

Bidirektionale Verbindungen

Ballbesitz auf dem Tactics-Board wird mit einer n-zu-n Verbindung von Ball zu Spieler modelliert. Um die aktuelle Position eines Balls zu bestimmen, wird dementsprechend die Position des besitzenden Spielers ausgelesen, was mit einer fehlenden bidirektionalen Verbindung Umstände bereitet. Um diesem Problem entgegen zu wirken, wurden folgende Massnahmen eruiert:

- Auslagerung der Logik in eine Service-Klasse. In dieser kann über ein DAO die Spieler-Instanz nachgeladen und auf diese zugegriffen werden.
- Spieler- und Ball-Listen werden als Instanz-Variablen der Klasse *TacticsBoardPresenter* geführt. Da die ID des Spielers über die n-zu-n Verbindung zugänglich ist, kann die Spieler-Liste nach dieser ID abgesucht werden, und ohne zusätzlichen Datenbank-Zugriff dessen Position (somit auch die Position des Balls) bestimmt werden.

Polymorphismus

Im Domain-Modell wurden die Klassen Ball und Spieler als Subklassen von "Token" definiert. Die Klasse "Movement" modelliert Bewegungen eines solchen Tokens; für Spieler sind dies Lauf-Pfade, für Bälle wiederum Pass- oder Schuss-Pfade. Da greenDAO selbst kein Polymorphismus unterstützt (und ID Kollisionen bei parallelem führen der Eigenschaften in den jeweiligen Tabellen "Player" und "Ball" wahrscheinlich sind), können folgende Lösungswege dem Problem Abhilfe schaffen:

- *Separate Movement-Tabellen*
  Für Ball und Spieler werden jeweils separate Movement-Tabellen geführt. Für diese Movements kann ein Interface oder eine abstrakte Klasse erstellt werden, um dupliziertem Code vorzubeugen.
- *Manuelles modellieren einer to-One Beziehung zur Tabelle "Token"*
  Erstellen einer Tabelle "Token", mit welcher die ID zur Tabelle "Movement" modelliert werden kann. ID-Kollissionen wird so vorgebeugt, da der Foreign-Key zugleich als Primary-Key der den Subklassen verwendet werden kann. Bedenken wurden lediglich durch das umständliche Nachladen von Entity-Änderungen durch greenDAO vorgerufen.

  Trotzdessen scheint dies als der momentan intuitivste Variante.

In Floorball Tactics-Board findet letztere Methode verwendung. Eine Token-Tabelle verwaltet die IDs von Player und Ball und enthält zudem Felder für geteilte Informationen.

5.6.7 Pixel und Auflösung

Es gibt in Android zwei verschiedene Grössenangaben. Die Pixel (px) definieren, wie viele Pixel eine View breit / hoch sein soll. Die native Auflösung kann von Gerät zu Gerät jedoch stark variieren. Die Punktdichte wird in "ppi" (Pixel per Inch) oder "dpi" (Dots per Inch) angegeben. Somit kann ein Icon, das 15 x 15 Pixel misst, auf zwei Geräten mit der selben Bildschirm-Grösse aber unterschiedlichen Punktdichten unterschiedlich gross dargestellt werden. Mit den "dp" (Density Pixels) wird dem Abhilfe geschaffen. Ein Density Pixel entspricht einem Pixel bei 160dpi. Diese Dimension wird für die Darstellung nach folgender Formel in Pixel umgerechnet: $px = dp * (dpi/160)$.

Während der Laufzeit erhält man Grössenangaben immer in Pixel, selbst wenn man sie unter *res/values/dimens.xml* als dp angegeben hat. Um diese in dp rechnen zu können, wurden zwei statische Methoden definiert, die Pixel zu dp und dp zu Pixel umrechnet.

5.6.8 Geräteunabhängiges Rendering einer Taktik

Durch das Taktik-Store Feature können Taktiken zwischen Geräten ausgetauscht werden, die teilweise unterschiedliche Auflösungen und Displayformate aufweisen. Um dies zu ermöglichen, werden die abgespeicherten Koordinaten in ein displayunabhängiges Format konvertiert und in diesem ausgetauscht. Dafür werden die Pixel-Koordinaten durch die jeweilige Spielfeldbreite / -höhe geteilt.
Die so erhaltenen Werte werden ausgetauscht und können durch eine Multiplikation mit den Spielfelddimensionen eines beliebigen Gerätes für dieses umgerechnet werden und sind damit unabhängig von ihrer ursprünglichen Auflösung.

5.6.9 Kollisionsdetektion

Für eine vereinfachte Kollisionsdetektion und aufgrund der mangelnden Testbarkeit von Android-Klassen aus dem *graphics*-Package, werden die Pfad-Berechnungen mit der JTS Geometry Library durchgeführt. Mit dieser müssen keine künstlichen Breiten auf Pfadobjekten eingeführt werden und die Abhängigkeit zum Grafik-Kontext der Activity (auf welchen aus den Klassen des *model*-Packages zugegriffen werden müsste) entfällt. Zudem kann mit bildschirmunabhängigen Einheiten anstelle von "Pixel" gerechnet werden.

5.6.10 Spielzüge abspielen

Jeder einzelne Pfadpunkt eines Spielers wird in einer Sammlung von Punkten, die über Start- und Endzeit verfügt, abgespeichert. Wird die Taktik abgespielt, startet ein Timer, der in einem bestimmten Intervall dafür sorgt, dass alle Spielsteine an die korrekte Position geschoben werden. Im Detail wird für jeden Spieler dessen Sammlung an Positionen abgefragt, ob sie zu diesem Zeitstempel in Bewegung sind. Falls ja, wird der entsprechende Punkt aus der Positionssammlung berechnet und zurückgegeben. Die Canvas wird daraufhin aktualisiert und der Abspielzeitstempel (Timer) erhöht.

Um die Spieler darzustellen, werden nicht die Floating Action Buttons, welche zuvor den Spieler repräsentierten, verwendet. In der "*OnDraw*"-Methode des Canvas wird für jeden Spieler ein ausgefüllter Kreis an entsprechender Position gezeichnet. Da "*OnDraw*" auch vom System ausgeführt werden kann, müssen die Positions- und Farbinformationen in der Objektinstanz abgelegt werden und können nicht als Parameter der "*OnDraw*" Funktion übergeben werden.

5.6.11 Android Cards in Recycler-View

Mit der Einführung von Android 5 Lollipop hat Google das Material-Design, dessen Design-Guidlines und mit diesen die "Cards" veröffentlicht [2]. Dabei sind Karten ein Einstiegspunkt zu einer detaillierteren Ansicht. Sie eignen sich besonders gut für Sammlungen von Elementen mit Bildern. Die Recycler-View dient der Anzeige dieser Cards. Sie kann Card-Views wiederverwenden und muss nicht immer eine neue kreieren. Dadurch werden Ressourcen gespart die Benutzbarkeit erhöht.

Multi-Layout-Recycler-View

Werden verschiedene Card-Layouts in der gleichen View angezeigt, muss die Methode *getItemViewType* überschrieben werden, die einen Integer zurückgibt. Damit kann angegeben werden, welches Layout an

[2] https://material.google.com/components/cards.html

welcher Position zum Einsatz kommt. Zusätzlich braucht es für jedes Layout einen ViewHolder Typ. In den Methoden *onCreateViewHolder* und *onBindViewHolder*, die eine Card aufbauen, wird je nach ViewType etwas anderes ausgeführt. Möchte man die Cards dazu noch verschieden breit anzeigen lassen, muss auf dem LayoutManager der RecyclerView ein SpanSizeLookup definiert werden, der für jede Position die richtige Anzahl Spalten zurückgibt.

Kartenbreite berechnen

Dem Layout Manager kann angegeben werden, wie viele Cards nebeneinander angezeigt werden sollen. Es kann jedoch nicht definiert werden, wie breit eine Card mindestens oder maximal sein soll. Um eine möglichst gute Usability zu erreichen, soll die Breite jedoch immer etwa gleich sein.

Um die Spaltenanzahl zu berechnen, braucht es die Breite des Bildschirms (je nach Ausrichtung des Geräts verschieden: Portrait / Landscape). Erschwerend kommt hinzu, dass jede Karte ein Margin (initial vom XML-layout definiert) und die Activity ein Padding besitzt, welche die effektive Kartenbreite beeinflusst und miteinbezogen werden muss:

```
int cardMargin = getDpfromPixels(homeActivity.getResources()
.getDimensionPixelSize(R.dimen.home_activity_tactic_card_margin));
int activityPadding = getDpfromPixels(homeActivity.getResources()
.getDimensionPixelSize(R.dimen.activity_horizontal_padding));
int width =
    getDpfromPixels(homeActivity.getResources().getDisplayMetrics().widthPixels)-2*activityPadding;
```

Die Anzahl erhält man mit width / "minimale Karten Breite", wobei mindestens 1 erforderlich ist (ist die Bildschirmbreite kleiner als die minimale Breite, so darf eine Karte auch schmaler sein als die definierte minimale Breite).

Bei festgelegter Spaltenanzahl kann die Breite der einzelnen Karten durch Setzten von Margins (Rand) verändert werden, was auch gleich die maximale Kartenbreite definiert. Die Margins müssen gesetzt werden, da die Breite der Karte sonst bis zum Doppelten ihrer ursprünglichen Grösse anwachsen kann (z.B.: bei minimaler Breite von 300 und Bildschirmbreite von 599). Die effektive Kartenbreite erhält man mit:

```
int cardWidth = (width - 2 * activityPadding - numberOfCards * 2 * cardMargin) / numberOfCards;
```

Falls diese grösser als die angegebene maximale Breite ist, werden links und rechts von der CardView Margins gesetzt. Diese berechnen sich folgendermassen:

```
int margin = (width - numberOfCards * 2 * cardMargin - numberOfCards * maxWidth - 2 *
    activityPadding) / 2;
```

5.6.12 Orientation-Change mit Navigation Drawer

Bei einem Orientation-Wechsel wird die aktuelle Activity ab- und wieder aufgebaut. Wird mit dem Navigation-Drawer und Fragments gearbeitet, muss beim Wiederaufbau der Activity das Fragment neu initialisiert und gesetzt werden. Zusätzlich muss dem Navigation-Drawer die angewählte Position mitgeteilt und das Fragment neu instantiiert werden. Da die Fragments in diesem App auf das Optionsmenu zugreifen, darf dies erst geschehen, nachdem das Menu kreiert wurde. Die zuletzt vor dem Drehen des Bildschirms angewählte Position im Navigation-Drawer muss als SavedInstanceState gespeichert werden, um nach dem Wiederaufbau der Activity zugreifbar zu sein.

Kapitel 6

Schlussfolgerungen

6.1 Ergebnisdiskussion

Die Entwicklung einer Android-Applikation in diesem Umfang war für beide Teammitglieder noch Neuland. Ein detailliertes Fazit über die eingesetzten Methoden und Werkzeuge bietet Aufschluss über positive und negative Entscheidungen.

6.1.1 Vorgehensweise

Projektmanagement

Die in Rücksprache mit dem Betreuer gewählte Projektmanagement-Methode erwies sich als optimal. Weil es sich bei dem Projekt um ein eigenes Thema handelt, war sich das Projektteam des groben Funktionsumfangs und den Anforderungen bereits im Voraus im Klaren.

Die im kleineren Rahmen angewendeten agilen Methoden boten den nötigen Freiraum, kurzfristig den Projektfokus anzupassen. Dies konnte mehrfach genutzt werden, um Features ohne Kompromisse wunschgemäss zu implementieren.

Evaluationen & Prototyp

Aufgrund des neuen Technologieumfelds wurde der Kompromiss eingegangen, zu Beginn einen Prototyp zu entwickeln. Dieser diente primär der Evaluation von Libraries, deren Ergebnisse in Kapitel 2.2 festgehalten sind.

Im Vorfeld fand bereits ein Vergleich der Frameworks statt. Der jeweils passendste Kandidat wurde für die Umsetzung des Prototyps eingesetzt. Weil sich nur wenige dieser Entscheidungen als nicht optimal herausstellten, konnten grosse Teile des Codes wiederverwendet werden, wodurch der Zeitverlust fruchtloser Implementationen minimiert wurde.

6.1.2 Technologien

Android

Android, zu Beginn eine sehr angenehme Plattform für die Entwicklung, barg auch kleine Unannehmlichkeiten. Zum einen werden dokumentierte Features wie der "Mini Drawer"[1] oder das "Floating Action Menu" [2] nicht in den Google Libraries mitgeliefert, andererseits musste zur Behebung von Problemen wie dem Caching von Bildern 5.6.3 oder Abstürzen beim Ändern der Geräteorientierung 5.6.12 viel Zeit aufgewendet werden.

Positive Aspekte umfassen die mittlerweile gut dokumentierten Funktionen, die grosse Community und die zahlreichen Libraries und Workarounds von Drittanbietern. Ebenfalls lieferte Android Studio eine Vielzahl an nützlichen Werkzeugen mit.

[1] https://material.google.com/patterns/navigation-drawer.html#navigation-drawer-behavior

[2] https://material.google.com/components/buttons-floating-action-button.html#buttons-floating-action-button-transitions

Scala

Die Entwicklung der Backend-Applikation mit Scala und dem Play-Framework bot dem darin noch unerfahrenen Team einen angenehmen Einstieg. Zwar wurde für einige Zeilen Code weit mehr Zeit aufgewendet, als dies in Java der Fall wäre, was aber durch die Robustheit und den Funktionsumfang gerechtfertigt ist.

Negativer Aspekt bei der Verwendung von Scala und dem Play-Framework ist lediglich die etwas längere Einarbeitungszeit, sofern man sich nicht an die Mischung aus funktionalen und Objektorientierten Sprachkonzepten gewöhnt ist. Das Team musste dazu ebenfalls etwas Extra-Aufwand investieren, war sich dessen aber schon im Voraus bewusst und bereit, diese Zeit zu investieren.

6.2 Resultate

Die im Rahmen des Projektes entwickelte Applikation entspricht grösstenteils den in der Startphase ausgearbeiteten Anforderungen. Bei der Priorisierung der Features wurde deren Qualität höher als die Abdeckung aller zuvor definierten Use Cases gewichtet. Einige Funktionen fielen durch Änderungen an Konzepten und Interaktionsmöglichkeiten weg, wovon die Usability des Endprodukts profitierte.

6.3 Ausblick

Die umgesetzte Applikation bietet in mehreren Aspekten Potential für Weiterentwicklung und Verbesserung. Da in der Entwicklungsphase stets ein Augenmerk auf Wartbarkeit und Wiederverwendbarkeit von Code und Konzepten gelegt wurde, kann die Applikation und der noch eher einfach gehaltene Store problemlos erweitert werden. Das Team ist nicht abgeneigt, weiterhin an dem Projekt zu arbeiten.

6.3.1 Tactics-Board

Dem Tactics-Board soll der eingeplante “Freedraw“ Modus hinzugefügt werden. Dieser dient dazu, dem Benutzer viele Freiheiten, welche er auf einer gewöhnlichen Taktiktafel hat, zurückzugeben. Dadurch können Hindernisse, Markierungen und Notizen eingezeichnet werden. Ebenfalls wird es möglich von den Vorteilen der digitalen Verwaltung, sowie des Online-Austausches Gebrauch zu machen ohne dabei die etwas aufwändige Gestaltung durch erfassen von Bewegungen, Pässen und Schüssen zu benutzen. Somit kann das Board für jegliche Art von Taktik und Übung gebraucht werden, Einschränkungen gibt es keine mehr.

Umsetzungswahrscheinlichkeit: hoch

6.3.2 Adaption auf andere Sportarten

Die für das Projekt erarbeiteten Konzepte und Verfahren sind auf verschiedene Ballsportarten übertragbar. Für dieses Feature müssten folgende Tasks umgesetzt werden:

- Kategorisierung Taktik nach Sportart (einmalig)
- Zeichnen Spielfeld und Berechnung Tor-Positionen (pro Sportart)

Wichtig ist hierbei, dass die Navigation der Sportarten sauber konzipiert wird, damit dieses Feature bestehende Benutzer nicht unnötige Zeit kostet. Eine Auslagerung neuer Sportarten in eine neue App ist keine Option, da die Popularität auf mehrere Apps aufgeteilt würde und Wartbarkeit und Kontrolle um ein vielfaches sinkt.

Umsetzungswahrscheinlichkeit: hoch

6.3.3 Verwaltung von Taktiken

In einem Gespräch mit Michael Kuhn, Trainer der UHC-Lokomotive-Stäfa, wurde ein alternative Organisationsmöglichkeit von Taktiken besprochen (siehe Anhang F). Diese könnten anhand von sogenannten “Tags“ gleich mehreren Kategorien angehören, anstatt die Zugehörigkeit auf nur eine Gruppe zu beschränken.

Umsetzungswahrscheinlichkeit: hoch

6.3.4 Zusammenstellung von Trainings

Michael Kuhn schwebte auch die Idee eines Trainingsgestalters vor. Wurden Taktiken mit Tags versehen, die den Bereich, die Anzahl Spieler, Anzahl Torhüter sowie die Dauer abdecken, ist es möglich, sich ein Training automatisch zusammenstellen zu lassen. Die Umsetzung dazu ist nicht ganz trivial und benötigt einen sorgfältig ausgearbeiteten Algorithmus.

Umsetzungswahrscheinlichkeit: mittel (In entfernter Zukunft)

6.3.5 Erweiterungen Store

Da der Store im Moment noch sehr rudimentär gehalten ist, bietet dieser Bereich das grösste Entwicklungspotential. Als Vorbild fungiert dabei der Google Play Store.

Benutzerinfos hinzufügen

Indem der Autor einer Taktik gespeichert und angezeigt wird, soll dem Benutzer die Möglichkeit geboten werden, weitere Taktiken dieses Autors anzuzeigen.

Umsetzungswahrscheinlichkeit: mittel

Laden verbessern

Steigen die Nutzerzahlen und das Volumen hochgeladener Taktiken, ist der momentane Ladevorgang nicht mehr tragbar. Es gibt dabei mehrere Möglichkeiten, den Ladevorgang zu optimieren:

- Laden einer bestimmter Anzahl an Taktiken. Scrollt der Benutzer bis zum Ende des Bildschirms, werden weitere geladen. Dies verringert sich Ladezeiten und schont das Datenvolumen des Benutzers.
- Separates Laden von Bildern und Name / Beschreibung. Die Karten werden inklusive Text angezeigt, und Bilder in einem davon unabhängigen Schritt nachgeladen. Damit kann der Caching-Algorithmus des Android Universal Image Loader eingesetzt werden, womit sich die Nutzung des Datenvolumen und die Benutzbarkeit verbessert.

Umsetzungswahrscheinlichkeit: sehr hoch

Taktiken kommerziell anbieten

Taktiken sollen gegen Geld angeboten werden können. Für diese Umsetzung muss das App sehr stabil laufen und Sicherheitsrisiken müssen ausgeschlossen werden können. Für dieses Feature sind mehrere Verkaufsmodelle denkbar.

- Angebot vom Entwicklerteam erstellte Taktiken. Der gesamte Gewinn kommt uns zu.
- Kooperation mit dem Unihockey-Verband oder anderen Interessenten. Der Gewinn wird aufgeteilt.
- Benutzer können Taktiken verkaufen. Der Gewinn abzüglich einer Provision für das Entwicklerteam geht an den Benutzer.

Umsetzungswahrscheinlichkeit: mittel

6.3.6 Portierung auf andere Betriebssysteme

Um die Reichweite der Applikation zu vergrössern, ist eine Portierung auf iOS oder Cross-Plattform Frameworks wie Xamarin oder react-native möglich. Viele Konzepte wurden auf Basis von Features von Android entwickelt und sind nicht leicht auf andere Plattformen adaptierbar. Eine Portierung wäre mit erheblichen Aufwand verbunden, der mit “ehrenamtlicher Arbeit“ kaum zu stemmen ist.

Umsetzungswahrscheinlichkeit: tief

Kapitel 7

Anhang

A System-Tests

Die System-Tests haben das Ziel, die Use Cases und das Zusammenspiel vom User-Interface mit dem System zu verifizieren. Es ist zu erwähnen, dass alle Cancel-Aktionen nicht aufgeführt sind, weil das Ergebnis dieser Aktion immer der Ausgangslage entsprechen sollte. Die Testnummern, zum Beispiel T101, stimmen mit den Use Case Nummern, hier UC101, überein.

1 T101A: Taktik erstellen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv, “Default Group“ ist zu sehen.

Ablauf

1. Auf orangen “Plus-Button“ klicken
2. Im Textfeld “Name“ den Text “Test-Taktik 1234“ eingeben
3. Im Textfeld “Description“ den Text “4312“ eingeben
4. Im Textfeld “Name“ den Text “Test-Taktik 1234“ eingeben
5. “Default Group“ auswählen in “Select Group“
6. “ADD“ anwählen
7. Im Taktikboard den “Backbutton“ betätigen

Erwartetes Resultat

Eintrag in “Default Group“ mit leerem Ganz-Feld als Bild, als Name (fetter Text) “Test-Taktik 1234“ und als Beschreibung (unterhalb des Namen) “4312“. Upload Icon soll aktiv sein (Pfeil in Wolke und kein Häkchen).

2 T101B: Taktik erstellen mit Standard-Werten

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und “Default Group“ ist zu sehen.

Ablauf

1. Auf orangen “Plus-Button“ klicken
2. “Half-Field“ nicht anwählen
3. “Default-Group“ auswählen

4. “SKIP“ klicken
5. Im Taktikboard den “Backbutton“ betätigen

Erwartetes Resultat

Eintrag in “Default Group“ mit leerem Ganz-Feld als Bild, als Name (fetter Text) “Untitled“ und als Beschreibung (unterhalb des Namen) “Update with own description“. Upload Icon soll aktiv sein (Pfeil in Wolke und kein Häkchen).

3 T101C: Taktik erstellen in eigener Gruppe

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv. UC401 Gruppe erstellen, wurde ausgeführt mit dem Namen “Test-Gruppe-01“ und die erstelle Gruppe ist zu sehen.

Ablauf

1. Auf orangen “Plus-Button“ klicken
2. “Half-Field“ nicht anwählen
3. “TestGruppe-01“ auswählen
4. “SKIP“ klicken
5. Im Taktikboard den “Backbutton“ betätigen

Erwartetes Resultat

Eintrag in “TestGruppe-01“ mit leerem Ganz-Feld als Bild, als Name (fetter Text) “Untitled“ und als Beschreibung (unterhalb des Namen) “Update with own description“. Upload Icon soll aktiv sein (Pfeil in Wolke und kein Häkchen).

4 T101D: Halb-Feld Taktik erstellen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und “Default Group“ ist zu sehen.

Ablauf

1. Auf Orangen “Plus-Button“ klicken
2. “Half-Field“ anwählen
3. “Default Group“ auswählen
4. “SKIP“ klicken
5. Im Taktikboard den “Backbutton“ betätigen

Erwartetes Resultat

Eintrag in “Default Group“ mit leerem Halb-Feld als Bild, als Name (fetter Text) “Untitled“ und als Beschreibung (unterhalb des Namen) “Update with own description“. Upload Icon soll aktiv sein (Pfeil in Wolke und kein Häkchen).

5 T102A: Taktik löschen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt.

Ablauf

1. In der Taktik-Karte auf die drei Pünktchen klicken
2. “Delete“ klicken
3. “Delete“ im erschienen Dialog anklicken

Erwartetes Resultat

Taktik-Eintrag, auf dem die Löschen-Aktion ausgeführt wurde ist nicht mehr zu sehen.

6 T103A: Taktik Eintrag editieren

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. UC401 Gruppe erstellen, wurde ausgeführt mit dem Namen “Test-Gruppe-01“ und die erstelle Gruppe sowie die “Default Group“ ist zu sehen.

Ablauf

1. In der Taktik-Karte auf die drei Pünktchen klicken
2. “Edit“ anklicken
3. Name auf “Test-1234“ ändern
4. Description auf “Desc“ ändern
5. Gruppe auf “Test-Gruppe-01“ ändern
6. “Edit“ betätigen

Erwartetes Resultat

Die Taktik wechselt die Gruppe zu “Test-Gruppe-01“, den Namen auf “ Test-1234“ und die Beschreibung auf “Desc“.

7 T103B: Taktik editieren

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt.

Ablauf

1. Auf das Bild, den Namen oder die Beschreibung klicken
2. Im Taktik-Board UC301 Spieler hinzufügen sowie UC303 Spieler verschieben durchführen und den Spieler innerhalb des Feldes platzieren
3. Den “Backbutton“ betätigen

Erwartetes Resultat

Im Bild erscheint der Spieler.

8 T104A: Taktik Name suchen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. Die “Default Group“ ist zu sehen.

Ablauf

1. auf die Lupe in der Toolbar klicken
2. im Textfeld “Test-Taktik“ eingeben

Erwartetes Resultat

Die erstellte Taktik ist zu sehen.

9 T104B: Taktik Name suchen mit nicht existierendem Suchtext

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. Die “Default Group“ ist zu sehen.

Ablauf

1. auf die Lupe in der Toolbar klicken
2. im Textfeld “mich gibt es nicht“ eingeben

Erwartetes Resultat

Die erstellte Taktik ist nicht zu sehen. Im Hintegrund erscheint ein Bild, dass einen Hinweis darauf ist, dass keine Taktiken gefunden wurden.

10 T104C: Taktik Beschreibung suchen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. Die “Default Group“ ist zu sehen.

Ablauf

1. auf die Lupe in der Toolbar klicken
2. im Textfeld “This is a test“ eingeben

Erwartetes Resultat

Die erstellte Taktik ist zu sehen.

11 T104D: Nicht angezeigte Taktik suchen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. UC401 Gruppe erstellen, wurde ausgeführt mit dem Namen “Test-Gruppe-01“. Die “Default Group“ ist nicht zu sehen, die erstellte Gruppe schon.

Ablauf

1. auf die Lupe in der Toolbar klicken
2. im Textfeld “Test-Taktik“ eingeben
3. auf das erschienene Feld “show results from hidden groups“ klicken

Erwartetes Resultat

Es erscheint ein Eintrag mit dem Text “Show results from hidden groups“. Klickt man darauf erscheint die Taktik mit dem Namen “Test-Taktik“.

12 T104E: Suchfeld anwählen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv.

Ablauf

1. auf die Lupe in der Toolbar klicken

Erwartetes Resultat

Der Titel, die Lupe, die Dropdown-Liste “Groups“ sowie das Gruppen-hinzufügen-Icon verschwinden. Es erscheint einen Pfeil nach links und ein Hinweis “Search. . . “.

13 T104F: Suchfeld Inhalt löschen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. UC104 Taktik filtern wurde mit dem Suchtext “blabla“ ausgeführt. Die Taktik ist nicht zu sehen.

Ablauf

1. Suchtext entweder mit Back-Space oder dem “x“ am rechten Rand löschen

Erwartetes Resultat

Die erstellte Taktik ist zu sehen und der Hintergrund enthält kein Bild.

14 T104G: Gruppe ausblenden

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“.

Ablauf

1. auf den Drop-Down-Button “Groups“ klicken.
2. im Eintrag “Default Group“ das Häkchen abwählen

Erwartetes Resultat

Die “Default Group“ sowie die erstellte Taktik sind nicht zu sehen. Im Hintergrund erscheint ein Bild, dass einen Hinweis darauf ist, dass keine Taktiken gefunden wurden.

15 T104H: Gruppe einblenden

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. Die “Default Group“ ist ausgeblendet.

Ablauf

1. auf den Drop-Down-Button “Groups“ klicken.
2. im Eintrag “Default Group“ das Häkchen anwählen

Erwartetes Resultat

Die “Default Group“ sowie die erstellte Taktik sind zu sehen. Im Hintergrund erscheint kein Bild.

16 T105A: Taktik kopieren

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“.

Ablauf

1. In der Taktik-Karte auf die drei Pünktchen klicken
2. “Duplicate“ klicken

Erwartetes Resultat

Es Erscheint ein neuer Taktik-Eintrag mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“.

17 T105B: Geteilte Taktik kopieren

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und “UC101 Taktik erstellen wurde durchgeführt“ mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. UC501 Taktik veröffentlichen wurde durchgeführt.

Ablauf

1. In der Taktik-Karte auf die drei Pünktchen klicken
2. “Duplicate“ klicken

Erwartetes Resultat

Es Erscheint ein neuer Taktik-Eintrag mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“ sowie der Gruppe “Default Group“. Das Upload Icon der duplizierten Taktik ist ein Pfeil in der Wolke.

18 T201A: Taktik abspielen (aus Card)

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv, “UC101 Taktik erstellen“ wurde durchgeführt und besitzt mindestens eine Bewegung (UC312).

Ablauf

1. In der Taktik-Karte auf “Play“ drücken
2. Auf das “Dreiecksymbol (Play-Button)“ drücken

Erwartetes Resultat

Der Spieler läuft die im UC312 erfasste Bewegung ab.

19 T201B: Taktik abspielen (aus TaktikTafel)

Voraussetzungen

“UC101 Taktik erstellen“ wurde durchgeführt, besitzt mindestens eine Bewegung (UC312) und F02 Tactics-Board ist aktiv.

Ablauf

1. In der TaktikTafel auf das “Dreiecksymbol (Play-Button)“ drücken

Erwartetes Resultat

Der Spieler läuft die im UC312 erfasste Bewegung ab.

20 T201C: Taktik mit Pass/Schuss abspielen

Voraussetzungen

“UC101 Taktik erstellen“ wurde durchgeführt. Die Taktik besitzt mindestens einen Pass und einen Schuss. F02 Tactics-Board ist aktiv.

Ablauf

1. In der TaktikTafel auf das “Dreiecksymbol (Play-Button)“ drücken

Erwartetes Resultat

Der Spieler läuft die Bewegung ab, und der Ball die angegebene Route mit Pass und Schuss.

21 T202A: Zeitliche Navigation innerhalb Taktik(aus Card)

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv, “UC101 Taktik erstellen“ wurde durchgeführt und besitzt mindestens eine Bewegung (UC312).

Ablauf

1. In der Taktik-Karte auf “Play“ drücken
2. Den Schieberegler nach rechts bewegen

Erwartetes Resultat

Der Spieler läuft die im UC312 erfasste Bewegung übereinstimmend zum Bewegen des Reglers ab.

22 T202B: Zeitliche Navigation innerhalb Taktik(aus TaktikTafel)

Voraussetzungen

“UC101 Taktik erstellen“ wurde durchgeführt, besitzt mindestens eine Bewegung (UC312) und F02 Tactics-Board ist aktiv. Der Schieberegler befindet sich ganz links.

Ablauf

1. Den Schieberegler nach rechts bewegen

Erwartetes Resultat

Der Spieler läuft die im UC312 erfasste Bewegung übereinstimmend zum Bewegen des Reglers ab.

23 T301A: Spieler hinzufügen

Voraussetzungen

“UC101 Taktik erstellen“ wurde durchgeführt und F02 Tactics-Board ist aktiv.

Ablauf

1. Den “Add Player (+ mit User-Icon)“ Button drücken.

Erwartetes Resultat

Es ist ein zusätzlicher Spieler in Form eines blauen-Kreises erschienen.

24 T302A: Spieler löschen

Voraussetzungen

“UC101 Taktik erstellen“ wurde durchgeführt, besitzt mindestens eine Bewegung (UC312) und F02 Tactics-Board ist aktiv. Nur ein Spieler ist in der Taktik.

Ablauf

1. Den Spieler mit dem UC312 durchgeführt wurde anklicken.
2. In der Toolbar (Menu Leiste am unteren Rand) das “löschen Symbol“ anklicken.

Erwartetes Resultat

Der Spieler und seine Bewegung ist nicht mehr zu sehen. Der Schieberegler darf sich nicht mehr Bewegen lassen. Undo Button (Pfeil nach Links) ist nicht aktiv.

25 T303A: Spieler verschieben

Voraussetzungen

“UC101 Taktik erstellen“ und “UC301 Spieler hinzufügen“wurde durchgeführt. F02 Tactics-Boards ist aktiv. Der Spieler besitzt keine Bewegung.

Ablauf

1. Den Spieler mit dem UC312 durchgeführt wurde, anklicken.
2. Im erschienen Menu “Move“wählen und den Spieler verschieben.

Erwartetes Resultat

Der Spieler befindet sich an der Position, bei der man den Finger vom Bildschirm genommen hat.

26 T303B: Spieler verschieben nach Zeitpunkt 0

Voraussetzungen

“UC101 Taktik erstellen“ und “UC301 Spieler hinzufügen“wurde durchgeführt und F02 Tactics-Board ist aktiv.

Ablauf

1. “UC312 Spielerbewegung hinzufügen“ durchführen
2. Den Spieler mit dem UC312 durchgeführt wurde anklicken.

Erwartetes Resultat

Im erschienen Menu darf kein Eintrag “Move“auftauchen.

27 T304A: Spieler editieren

Voraussetzungen

“UC101 Taktik erstellen“ und “UC301 Spieler hinzufügen“wurde durchgeführt, besitzt mindestens eine Bewegung (UC312) und F02 Tactics-Board ist aktiv.

Ablauf

1. Auf den Spieler klicken.
2. In der Toolbar (Menu Leiste am unteren Rand) das “Farb-Paletten Symbol“ anklicken.
3. Ein grüner Farbton auswählen.
4. Mit “Pick“ bestätigen

Erwartetes Resultat

Der Spieler und seine Bewegung ändern die Farbe auf den gewählten grün Farbton.

28 T305A: Passweg hinzufügen

Voraussetzungen

“UC101 Taktik erstellen“ und zweimal “UC301 Spieler hinzufügen“ wurde durchgeführt, der erste besitzt eine Bewegung (UC312) der zweite einen Ball (UC309) und F02 tactics-Board ist aktiv. Der Schieberegler befindet sich ganz links.

Ablauf

1. Auf den Spieler mit Ball klicken.
2. Im erschienen Menu “Pass“ wählen
3. Mit einem Finger das Pass-Ende auf die Bewegung des ersten Spieler ziehen, bis ein roter Kreis erscheint.
4. Finger vom Bildschirm nehmen

Erwartetes Resultat

Es erscheint ein gestrichelter, schwarzer Pfeil vom Spieler mit Ball bis zur Bewegung des ersten Spielers.

29 T307A: Schussweg hinzufügen

Voraussetzungen

“UC101 Taktik erstellen“ und “UC301 Spieler hinzufügen“ wurde durchgeführt. Der Spieler besitzt einen Ball (UC309) und F02 Tactics-Board ist aktiv.

Ablauf

1. Auf den Spieler klicken.
2. Im erschienen Menu “Shoot“ wählen
3. Mit einem Finger das Schuss-Ende auf ein Goal ziehen, bis ein roter Kreis erscheint.
4. Finger vom Bildschirm nehmen

Erwartetes Resultat

Es erscheint ein doppelt gestrichelter, schwarzer Pfeil vom Spieler mit Ball zur Goal-Linie.

30 T309A: Ball hinzufügen

Voraussetzungen

“UC101 Taktik erstellen“ und “UC301 Spieler hinzufügen“ wurde durchgeführt. F02 Tactics-Board ist aktiv.

Ablauf

1. Auf den Spieler klicken.
2. Im erschienen Menu “Add Ball“ wählen

Erwartetes Resultat

Es erscheint ein schwarzer Punkt neben dem Spieler. Klickt man erneut auf den Spieler erscheint “Add Ball“ nicht mehr, dafür “Pass“ und “Shoot“.

31 T3011A: Ball entfernen

Voraussetzungen

“UC101 Taktik erstellen“ wurde durchgeführt. Die Taktik besitzt mindestens einen Pass (UC305) und einen Schuss (UC307). F02 Tactics-Board ist aktiv.

Ablauf

1. Auf den Ball klicken.
2. In der Toolbar (Menu Leiste am unteren Rand) das “löschen Symbol“ anklicken.

Erwartetes Resultat

Der Ball und seine Bewegungen (Pass und Schuss) werden nicht mehr angezeigt.

32 T3012A: Spielerbewegung hinzufügen

Voraussetzungen

“UC101 Taktik erstellen“ und “UC301 Spieler hinzufügen“ wurde durchgeführt. F02 Tactics-Board ist aktiv.

Ablauf

1. Auf den Spieler klicken.
2. Im Menu “Run“ wählen
3. den gewünschten Laufweg mit dem Finger zeichnen.
4. Den Finger vom Bildschirm nehmen

Erwartetes Resultat

Es erscheint eine durchgezogen Linie in der selben Farbe wie der Spieler mit dem aufgezeichneten Weg.

33 T3012B: Parallele Spielerbewegung hinzufügen

Voraussetzungen

“UC101 Taktik erstellen“ und zweimal “UC301 Spieler hinzufügen“ wurde durchgeführt. Der eine Spieler besitzt eine Bewegng. F02 Tactics-Board ist aktiv und der Schieberegler befindet sich ganz links.

Ablauf

1. Auf den Spieler ohne Bewegung klicken
2. Im Menu “Run“ wählen
3. den gewünschten Laufweg mit dem Finger zeichnen.
4. Den Finger vom Bildschirm nehmen

Erwartetes Resultat

Währen des Zeichnens sollte sich der Spieler, der schon einen Laufweg hatte, fortbewegen. Beim Zweiten Spieler erscheint ebenfalls eine durchgezogen Linie. Beim Abspielen bewegen sich die Spieler parallel.

34 T3013B: Spielerbewegung entfernen

Voraussetzungen

“UC101 Taktik erstellen“ und “UC301 Spieler hinzufügen“ wurde durchgeführt. F02 Tactics-Board ist aktiv.

Ablauf

1. “UC312 Spielerbewegung hinzufügen“ durchführen
2. In der Toolbar (Menu Leiste am unteren Rand) den Undo-Button(Pfeil nach links) betätigen

Erwartetes Resultat

Die erfasste Bewegung verschwindet wieder.

35 T316A: Ganzfeldansicht anzeigen

entspricht T101A. Überprüfen das auch wirklich Ganzfeld zu sehen ist.

36 T317A: Halbfeldansicht anzeigen

entspricht T101D.

37 T401: Gruppe erstellen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv

Ablauf

1. Auf den Add-Group Button klicken (In der Toolbar ganz rechts)
2. Als Name “Test-Group“ eingeben
3. Description leer lassen
4. “Add“ klicken

Erwartetes Resultat

Zuoberst erscheint eine neue Gruppenabtrennung mit dem Namen “Test-Group“. Das Upload Symbol ist inaktiv. Die Gruppe erscheint ebenfalls im Group-DropDown.

38 T402: Gruppe löschen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv. UC401 Gruppe erstellen, wurde ausgeführt. Die erstelle Gruppe ist zu sehen und besitzt mindestens eine Taktik.

Ablauf

1. Rechts neben dem Gruppentitel auf die drei Pünktchen klicken
2. Den Eintrag “Delete“ wählen
3. Den Dialog mit “Delete“ bestätigen

Erwartetes Resultat

Die Gruppe und die beinhaltenden Taktiken sind nicht mehr zu sehen, es gibt auch keinen Eintrag im Group-DropDown.

39 T403A: Gruppe editieren

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv. UC401 Gruppe erstellen, wurde ausgeführt mit dem Namen “Test-Gruppe-01“ und die erstelle Gruppe ist zu sehen.

Ablauf

1. Rechts neben dem Gruppentitel auf die drei Pünktchen klicken
2. Den Eintrag “Edit“ wählen
3. Den Namen auf “Edited“ ändern
4. “Edit“ klicken

Erwartetes Resultat

Der Gruppentitel hat sich zu “Edited“ geändert. Ebenso im Gruppen-DropDown.

40 T404A: Gruppe kopieren

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv. Es existiert eine Gruppe mit mindestens einer Taktik, die Gruppe und die Taktik(en) wurden auf den Server geladen.

Ablauf

1. Rechts neben dem Gruppentitel auf die drei Pünktchen klicken
2. Den Eintrag “Duplicate“ wählen

Erwartetes Resultat

Es erscheint ein Neuer Gruppen Eintrag, sowie mit neuen Taktiken. Die Namen und Beschreibungen sind die gleichen wie von der “Kopiervorlage“. Das Upload-Symbol hat sich jedoch von einem Häckchen zu einem Pfeil geändert.

41 T405A: Taktik einer Gruppe zuordnen

Entspricht T103A.

42 T406A: Taktik aus Gruppe entfernen

Entspricht T103A.

43 T501A: Taktik veröffentlichen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und UC101 Taktik erstellen wurde durchgeführt mit dem Namen “Test-Taktik“, der Beschreibung “This is a test“. Es besteht eine Verbindung zum Server und man ist eingeloggt.

Ablauf

1. In der Taktik-Karte auf den Upload-Button klicken (Wolke mit Pfeil)
2. Name auf “Test-1234“ ändern
3. Description auf “Desc“ ändern
4. Kategorie “Pass“ wählen
5. “UPLOAD“ betätigen

Erwartetes Resultat

Es erscheint ein Hinweis, dass die Taktik erfolgreich hochgeladen wurde und das Upload-Symbol hat sich zu einem Häkchen in einer Wolke verändert (und ist inaktiv). Im Store (F04) gibt es in der Kategorie “Pass“ einen EIntrag mit dem Namen “Test-1234“ und der Beschreibung “Desc“.

44 T502A: Gruppe veröffentlichen

Voraussetzungen

F01 Lokaler Taktikstore (Homescreen) ist aktiv und “UC401 Gruppe erstellen“ wurde mit dem Namen “Test-Gruppe“ und Beschreibung “Test1234“ durchgeführt. Die Gruppe besitzt mehrere Taktiken. Es besteht eine Verbindung zum Server und man ist eingeloggt.

Ablauf

1. Rechts neben dem Gruppen-Titel auf den Upload-Button klicken (Wolke mit Pfeil)
2. Name auf “Test-Gruppe-1234“ ändern
3. Description auf “Gruppe-Desc“ ändern
4. Kategorie “Pass“ wählen
5. “UPLOAD“ betätigen

Erwartetes Resultat

Es erscheint ein Hinweis, dass die Gruppe erfolgreich hochgeladen wurde und das Upload-Symbol hat sich zu einem Häkchen in einer Wolke verändert (und ist inaktiv). Im Store (F04) gibt es in der Kategorie “Pass“ einen Eintrag mit dem Namen “Test-Gruppe-1234“ und der Beschreibung “Gruppe-Desc“. Die Beinhaltenden Taktiken haben die gleichen Namen und Beschreibungen wie die originalen (Inhalt von “Test-Gruppe“).

45 T503A: Taktik beziehen 1. Phase

Voraussetzungen

F04 Taktikstore ist aktiv und es gibt eine Taktik mit Namen “Test“ und Beschreibung “Test-Desc“. Die Taktik wurde noch nicht heruntergeladen.

Ablauf

1. In der Taktik-Karte den “Download-Button“ klicken

Erwartetes Resultat

Es erscheint ein Hinweis, dass die Taktik erfolgreich heruntergeladen wurde und der Download-Button ist inaktiv. Unter Downloads (F04.4) gibt es einen Eintrag mit Namen “Test“ und Beschreibung “Test-Desc“. Die Taktik kann abgespielt werden.

46 T503B: Taktik beziehen 2. Phase

Voraussetzungen

T502A wurde durchgeführt. “F04.4 Downloads“ ist aktiv.

Ablauf

1. In der Taktik-Karte den “COPY HOME-Button“ klicken
2. Den Namen auf “Downloaded“ändern
3. Die Beschreibung auf “Downloaded-Desc“ändern
4. Als Gruppe die “Default-Group“ auswählen
5. “Duplicate“betätigen

Erwartetes Resultat

Unter dem Homescreen (F01) gibt es einen Eintrag mit Namen “Downloaded“ und Beschreibung “Downloaded-Desc“. Die Taktik kann editiert und hochgeladen werden.

47 T504A: Gruppe beziehen 1. Phase

Voraussetzungen

F04 Taktikstore ist aktiv und es gibt eine Gruppe mit Namen “Test-Gruppe“ und Beschreibung “Test-Gruppe-Desc“. Die Gruppe wurde noch nicht heruntergeladen.

Ablauf

1. In der Gruppen-Karte den “Download-Button“ klicken

Erwartetes Resultat

Es erscheint ein Hinweis, dass die Gruppe erfolgreich heruntergeladen wurde und der Download-Button ist inaktiv. Unter Downloads (F04.4) gibt es einen Eintrag mit Namen “Test-Gruppe“ und Beschreibung “Test-Gruppe-Des“. Die Taktiken der Gruppe können abgespielt werden.

48 T504B: Gruppe beziehen 2. Phase

Voraussetzungen

T503A wurde durchgeführt. “F04.4 Downloads“ ist aktiv.

Ablauf

1. In der Karte den “COPY HOME-Button“ klicken
2. Den Namen auf “Downloaded-Gruppe“ändern
3. Die Beschreibung auf “Downloaded-Gruppe-Desc“ändern
4. “Duplicate“betätigen

Erwartetes Resultat

Unter dem Homescreen (F01) gibt es eine neue Gruppe mit Namen “Downloaded-Gruppe“ und Beschreibung “Downloaded-GruppeDesc“. Die Gruppe und alle beinhaltenden Taktiken können editiert und hochgeladen werden.

49 T601A: Kategorien verwalten - hinzufügen

Voraussetzungen

Mit Webbrowser angemeldet im Admin-Tool und man besitzt admin-Rechte.

Ablauf

1. Auf den Reiter Categories gehen
2. Als Name “Test-Cat“ eingeben
3. Auf “Add“klicken
4. Android: Applikation neu starten

Erwartetes Resultat

Es erscheint einen Eintrag in der Liste unterhalb des Categories-Titel. Der Eintrag hat eine Nummer höher als der letzte. Falls es der erste Eintrag ist hat er die Nummer 1. Der Name ist “Test-Cat“. Im Android unter Kategorien (F04.2) erscheint ebenfalls ein Eintrag mit dem Namen “Test-Cat“.

50 T602A: Kategorien verwalten - löschen

Voraussetzungen

Mit Webbrowser angemeldet im Admin-Tool und man besitzt admin-Rechte. Es existiert eine Kategorie mit dem Namen “Test-Cat“.

Ablauf

1. Auf den Reiter Categories gehen
2. Rechts neben der Kategorie “Test-Cat“ den Delete-Button klicken.
3. Android: Applikation neu starten

Erwartetes Resultat

Der Eintrag mit dem Namen “Test-Cat“ verschwindet im Webtool, wie auch im Android.

51 T603A: Taktiken bereinigen

Voraussetzungen

Mit Webbrowser angemeldet im Admin-Tool und man besitzt admin-Rechte. Es existiert eine Taktik mit dem Namen “Uploaded-Tactic“ auf dem Server

Ablauf

1. Auf den Reiter Tactics gehen
2. Die Taktik mit dem Namen “Uploaded-Tactic“ suchen
3. Auf “Delete-Button“klicken
4. Android: Applikation neu starten

Erwartetes Resultat

Der Eintrag mit dem Namen “Uploaded-Tactic“ verschwindet im Webtool, wie auch im Android.

52 T603B: Gruppen bereinigen

Voraussetzungen

Mit Webbrowser angemeldet im Admin-Tool und man besitzt admin-Rechte. Es existiert eine Gruppe mit dem Namen “Uploaded-Group“ auf dem Server

Ablauf

1. Auf den Reiter Tactics gehen
2. Die Taktik mit dem Namen “Uploaded-Group“ suchen
3. Auf “Delete-Button“klicken
4. Android: Applikation neu starten

Erwartetes Resultat

Der Eintrag mit dem Namen “Uploaded-Group“ verschwindet im Webtool, wie auch im Android.

B Schritt für Schritt Anleitung Google Sign-In and Keystore Hashes

Damit Google die Anfragen von Apps authentifizieren und Rückschlüsse auf den Urheber der Anfrage ziehen kann, wird jede App mit einem Schlüssel signiert. Der dazugehörige SHA-1 Fingerabdruck muss bei Google hinterlegt sein, um die Authentifizierung zu ermöglichen. Mögliche Arten von Schlüsseln sind der Debug-Key, der während der Entwicklung Einsatz findet, und der Release-Key, mit dem Apps für Veröffentlichungen auf Google-Play signiert werden. Ein in den Konfigurationen hinterlegtes JSON File beschreibt die Art der Zugriffe.

Lösungsansatz/Umsetzung

1. Auf die Google Developer Seite wechseln: https://developers.google.com/identity/sign-in/android/start
2. Zu Punkt zwei "Get a configuration file" navigieren.
3. Name des Projekts, des Package sowie des Landes angeben
4. Google Sign-In hinzufügen und sein SHA-1 Fingerprint angeben.
5. Den DebugKey erhält man mit folgendem Befehl: keytool -exportcert -list -v -alias androiddebugkey -keystore /.android/debug.keystore passwort: android
6. Weiter auf "Continue to Generate configuration files" klicken
7. Das File "google-services.json" downloaden
8. Dieses File im Android-Projekt unter app/ ablegen.
9. Zu Google Api Console wechseln: https://console.developers.google.com
10. Im neuen Fenster rechts neben der Sucheingabe das einzurichtende App-Projekt auswählen
11. links im Navigationsmenu auf Zugangsdaten. Hier müssen mehrere Dinge erledigt werden.
    - unter "API-Schlüssel"->"Android key for ..."anpassen falls von mehreren PCs aus entwickelt wird. Dann müssen alle SHA-1 Fingerprints registriert werden. Wird draufgeklickt erscheint eine Editierseite. Hier kann der Paketname bei bedarf angepasst werden so wie der zuvor hinzugefügte SHA-1 Fingerabdruck. Ebenfalls kann ein neuer Fingerabdruck hinzugefügt werden, was notwendig ist für alle PCs die im Projekt involviert sind, sowie für den Release Key. Danach speichern betätigen.
    - unter "OAuth-2.0-Client-IDs"den "Web client entfernen"
    - "Anmeldedaten erstellen"anklicken (oberhalb) und OAuth-Client-ID auswählen ->Android ->Name ist egal ->SHA-1 Fingerprint des zweiten Entwicklers und Package angeben ->Erstellen.
    - wiederhole den letzten Punkt für alle Entwickler und für alle Release-Keys
    - wiederhole den letzten Punkt für eine Backend-Authentifizierung aber wähle "Webanwendung"->gib einen Namen zB: Scala Backend ->den Rest leer lassen ->Speichern. Die generierte Client ID wird in der Applikatiion gebraucht.
    - Webbrowser schliessen.
12. Für eine vollständige Authentisierung ist sowohl Code im Android-Projekt, wie auch im Backend zu konsultieren. Für eine Beispiel-Implementierung dienen folgende Klassen:
    - **Backend** *floorballtactics-server/prototype/app*
      controllers/UserController.scala
      services/UserService.scala
    - **Android** *floorballtactics-android/Prototyp/app/src/main/java*
      ch.codinglynx.tacticsboard.prototype.HomeActivity.java
      ch.codinglynx.tacticsboard.prototype.signin.OAuthTask.java

C Code-Beispiele

In diesem Abschnitt werden intressante und wichtige Codeausschnitte aufgeführt. Sie dienen als Beispielimplementationen oder als Anregungen zur Lösungsfindung.

1 Google OAuth

Signin Prozess:

```
GoogleSignInOptions gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
        .requestIdToken(getString(R.string.oauth_clientid))
        .requestEmail()
        .build();
this.apiClient = new GoogleApiClient.Builder(this)
        .enableAutoManage(this, this /* OnConnectionFailedListener */)
        .addApi(Auth.GOOGLE_SIGN_IN_API, gso)
        .addApi(AppIndex.API).build();

Intent signInIntent = Auth.GoogleSignInApi.getSignInIntent(this.apiClient);
startActivityForResult(signInIntent, RC_SIGN_IN);
```

Android Google-Konto und Token Abfragen:

```
private void handleSignInResult(GoogleSignInResult result) {
        if (result.isSuccess()) {
                // Signed in successfully, show authenticated UI.
                signInAccount = result.getSignInAccount();
                Picasso.with(this)
                        .load(signInAccount.getPhotoUrl())
                        .placeholder(R.drawable.ic_person_white_24px)
                        .error(R.drawable.ic_person_white_24px)
                        .into(this.accountPicture);
                this.accountName.setText(signInAccount.getDisplayName());
                this.accountEmail.setText(signInAccount.getEmail());
                RestClient.setToken(signInAccount.getIdToken());
        } else {
                Toast.makeText(this, "login failed", Toast.LENGTH_SHORT).show();
        }
}
```

2 Serializer / Deserializer

```
public class TacticSerializer implements JsonSerializer<Tactic> {

        @Override
        public JsonElement serialize(Tactic src, Type typeOfSrc,  JsonSerializationContext
            context) {
               JsonObject tacticJSON = new JsonObject();
               Gson gson = GsonHolder.getGsonInstance();
               tacticJSON.addProperty("id", src.getId());
               tacticJSON.addProperty("name", src.getName());
               ...
               tacticJSON.add("players", gson.toJsonTree(src.getPlayers()));
               tacticJSON.add("balls", gson.toJsonTree(src.getBalls()));
        return tacticJSON;
        }
}

public class TacticDeserializer implements JsonDeserializer<Tactic> {

        @Override
        public Tactic deserialize(JsonElement json, Type typeOfT, JsonDeserializationContext
            context) throws JsonParseException {
               JsonObject jsonObject = json.getAsJsonObject();
               Gson gson = GsonHolder.getGsonInstance();
               Long id = jsonObject.get("id").getAsLong();
               String name = jsonObject.get("name").getAsString();
               ...
               List<Player> players = (List<Player>) gson.fromJson(jsonObject.get("players"),
                   new TypeToken<List<Player>>() { }.getType());
               List<Ball> balls = (List<Ball>) gson.fromJson(jsonObject.get("balls"), new
                   TypeToken<List<Ball>>() { }.getType());
               Tactic tactic = new Tactic(id, name, description, time, thumbnail,
                   thumbnailWidth, 0.0, 0.0, 0.0, 0.0, isHalfField, isFullScreenMode, groupId);
               tactic.setPlayers(players);
               tactic.setBalls(balls);
               return tactic;
        }
}
public static Gson getGsonInstance() {
        GsonBuilder gsonBuilder = new GsonBuilder();
        ...
        gsonBuilder.registerTypeAdapter(Tactic.class, new TacticSerializer());
        ...
        gsonBuilder.registerTypeAdapter(Tactic.class, new TacticDeserializer());
        return gsonBuilder.create();
}
```

3 Retrofit Api Service Creator

Beispiel eines Interfaces:

```
public interface ApiService {
        @POST("tactics")
        Call<Void> uploadTactic(@Body Tactic tactic);
        ..
        @GET("tactics")
        Call<List<TacticPreview>> getTacticPreviews(@Query("categoryId") Long categoryId,
            @Query("searchQuery") String searchQuery);

        @GET("tactics/{objId}")
        Call<Tactic> getTacticWithId(@Path("objId") String objId);
```

```
..
}
```

Retrofit Service Implementierung erzeugen:

```
private static ApiService apiService;
private static View snackbarView;
private static OkHttpClient.Builder httpClient = new OkHttpClient.Builder();
private static String oAuthToken;

private static Retrofit.Builder builder =
new Retrofit.Builder()
        .baseUrl(BASE_URL)
        .addConverterFactory(GsonConverterFactory.create(GsonHolder.getGsonInstance()));

public static ApiService getApiService() {
        return apiService = createService(ApiService.class, oAuthToken);
}
```

Request absetzten und Response handhaben:

```
public static class GetTactic extends AsyncTask<Map.Entry<String, TacticCallback>, Void, Void>
    {
        @Override
        protected Void doInBackground(final Map.Entry<String, TacticCallback>... params) {
                final String tacticObjId = params[0].getKey();
                final TacticCallback callback = params[0].getValue();
                Call<Tactic> call = RestClient.getApiService().getTacticWithId(tacticObjId);
                call.enqueue(new Callback<Tactic>() {
                        @Override
                        public void onResponse(Call<Tactic> call, Response<Tactic> response) {
                                ..
                                callback.onResponse(response.body());
                                ...
                        }

                @Override
                public void onFailure(Call<Tactic> call, Throwable t) {
                        ...
                        return null;
                }
        }
}
```

4 OkHttp Interceptor

```
public static <S> S createService(Class<S> serviceClass, final String authToken) {
        ...
        httpClient.addInterceptor(new Interceptor() {
                @Override
                public okhttp3.Response intercept(Chain chain) throws IOException {
                Request original = chain.request();

                        Request.Builder requestBuilder = original.newBuilder()
                                .header("X-Auth", authToken)
                                .method(original.method(), original.body());

                        Request request = requestBuilder.build();
                        return chain.proceed(request);
                }
        });
```

```
        OkHttpClient client = httpClient.build();
        Retrofit retrofit = builder.client(client).build();
        return retrofit.create(serviceClass);
}
```

5 Android Universal Image Loader

ImageLoader instantiieren:

```
private static ImageLoader imageLoader;

 private static ImageLoader getImageLoader(Context context) {
    if(imageLoader == null) {
    DisplayImageOptions defaultOptions = new DisplayImageOptions.Builder()
            .showImageOnLoading(R.drawable.thumbnail_placeholder)
            .cacheOnDisk(true)
            .cacheInMemory(true)
            .bitmapConfig(Bitmap.Config.RGB_565)
            .imageScaleType(ImageScaleType.EXACTLY)
            .delayBeforeLoading(500)
            .build();

        ImageLoaderConfiguration config = new ImageLoaderConfiguration.Builder(context)
            .defaultDisplayImageOptions(defaultOptions)
            .imageDownloader(new Base64ImageDownloader(context))
            .memoryCache(new WeakMemoryCache())
            .threadPoolSize(2)
            .build();
            imageLoader = ImageLoader.getInstance();
            imageLoader.init(config);
        }
    return imageLoader;
}

public static void loadImagefromBase64Data(Context context, String data, String unique,
     SimpleImageLoadingListener listener){
    data = data.concat("thumbnail"+unique);
    getImageLoader(context).loadImage(data, listener);
}
```

Eigener ImageDownloader:

```
public class Base64ImageDownloader extends BaseImageDownloader {

        ...

        @Override
        protected InputStream getStreamFromOtherSource(String base64Data, Object extra) throws
             IOException {
                base64Data = base64Data.split("thumbnail")[0];
                return new ByteArrayInputStream(Base64.decodeBase64(base64Data);
        }
}
```

D Semantisches Sprachmodell

1 Android

- *Back-Button* Ein Android spezifischer Button der meist vom Betriebssystem verwaltet wird. Er wird gebraucht um in der Screen-History rückwärts zu Navigieren[7].
- *Up-Button* Ein Button links in der Actionbar. Wird gebruacht um Hierarchisch in den Screens nach oben zu Navigieren[7].
- *Actionbar* Persistent angezeigte Leiste im obersten Bildschirmbereich des Apps mit wichtigen Funktionen[6].
- *Navigation Drawer* Schnell-Navigationsleiste am linken Bildschirmrand, die angezeigt oder versteckt werden kann durch ziehen nach links respektive rechts [12].
- *Cards* Ein UI-Element, meist mit einem Bild, Text und Aktionen. Cards sind immer gleich breit können jedoch unterschiedliche Höhen haben, dadurch sind auf einem Display wie echte Karten gliederbar.
- *Toolbar* Ein grafisches Kontroll-Element welches Buttons, Icons und andere In- oder Output Elemente besitzt[23].
- *DropDown* Eine ausklappbare Auswahl vo Grafischen Elementen. Oft in aufgelisteter Form.
- *Fragments*Ein Fragment ist ein Verhalten oder ein Teil eines User-Interfaces einer Activity [10].
- *Canvas* Eine Canvas hällt seine “draw“ calls und zeichnet diese in ein Bitmap [8].

2 Unihockey

- *Grossfeld* Bezeichnet eines der zwei verschiedenen Spielfeldgrössen im Unihockey. Das Grossfeld entspricht etwa einer Dreifachturnhalle [1]. Gespielt wird mit 5 Feldspieler und einem Torwart pro Mannschaft.
- *Kleinfeld* Bezeichnet eines der zwei verschiedenen Spielfeldgrössen im Unihockey. Das Kleinfeld entspricht etwa einer Einfachturnhalle [2]. Gespielt wird mit 3 Feldspieler und einem Torwart pro Mannschaft.
- *Taktik* Beschreibung wie sich eine Mannschaft auf dem Feld organisiert / verhaltet.
- *Spielzug* Beschreibung wie eine Ablauf von Pass-, Schuss- und Laufwegen auszusehen hat.
- *Passweg* Spieler A passt den Ball zu einem Spieler B auf der Route des Passweges.
- *Schussweg* Spieler A schiesst den Ball aus Position X aufs Tor.
- *Ganzfeld* Ansicht eines ganzen Feldes, beinhaltet 2 Tore und 2 Halbfelde.
- *Halbfeld* Ansicht einer Hälfte des Spielfeldes, beinhaltet nur ein Tor.
- *Gruppe* Eine Ansammlung von Taktiken gruppiert nach Zusammengehörigkeiten (vom Anwender erstellt und verwaltet).
- *Kategorie* Einteilung von einzelnen Taktiken und Gruppen in von uns vorgegebene Themen.
- *Floorball* Englische Bezeichnung für Unihockey. Oft generalisierend verwendet da Unihockey in anderen Ländern unterschiedliche Namen hat.

[1] ca 40m x 20m
[2] ca 24m x 14m

3 Technisch

- *ORM* Object-Relational-Mapping. Eine Technik um Objekte einer Objektorientierten Programmiersprache in einer Datenbank abzulegen [21].
- *Memory-Swap* Auslagerung eines RAM Speicherbereichs in einen Sekundären Speicher, wie der Festplatte, durch das Betriebssystem [20]
- *LRU-Cache* Least-Recently-Used Cache. Ist ein Cache-Speicher Strategie, bei der die am längsten ungenützten Elemente aus dem Cache entfernt werden wenn der Speicher knapp wird.
- *Weak-References* Eine Weak-Reference schützt das Objekt nicht vor dem Garbage-Collector [24].
- *Garbage-Collection* Räumt nicht mehr gebrauchte Objekte aus dem Speicher falls keine Referenz auf dem Objekt besteht und demzufolge das Objekt nicht mehr erreichbar ist [24].
- *Bitmap* Steht für eine Rastergrafik und im Speziellen für ein Binärbild [19].
- *Pre-Compiler* Auf Deutsch ein "Vorkompilierer". Precompiler kommen zum Einsatz, wenn mit einer bestehenden Programmiersprache (Zielsprache) neue Konstrukte mit einer eigenen Syntax verarbeitet werden sollen [22].
- *Separation-of-Concerns* Die Trennung von Verantwortlichkeiten auf spezialisierten Bereiche.

4 Spezifisch

- *Taktik* gilt in der ganzen Arbeit als Überbegriff für Taktiken, Spielzüge und Übungen.
- *Gruppe* Eine Ansammlungen von Taktiken / Spielzügen, die ähnliche Ziele verfolgen. Vom Benutzer verwaltet.
- *Kategorie* Eine Ansammlungen von Taktiken, sowie Gruppen von Taktiken, die ähnliche Ziele verfolgen. Ist vom App vorgegeben, der Nutzer kann die Kategorien nicht editieren.
- *Tags* Erweiternde Informationen in Form von kurzen Begriffen um eine Taktik oder Gruppe zu kategorisieren.

E Benutzerhandbuch

Das Benutzerhandbuch ist in Englisch geschrieben, da es auch in der App unter "Help"Verwendung findet:

The Floorball-Tactic App helps you create, manage and share your tactics and exercises. With an emphasis on easy and intuitive handling, Floorball-Tactics App brings you a new experience to your work as a coach. If you have any problems, read this short manual or tell us about your difficulties by shooting an email to codinglynx@gmail.com or just leave a comment on our Google Play page.

1. Home:
In this view you can order, manage and play back all of your locally stored tactics.

1.1. Creating a tactic:
Create your new tactic by clicking the orange, round button in the bottom right corner. Fill in the name and description or just leave it blank. Choose the group, the new tactic should belong to and hit "ADD". Your new tactic automatically opens on a blank tactics board; now it's time to unleash your creativity! If you want to open a new tactic board without filling in all details, just hit "SKIP" on the previously described dialog. The tactic will be added to the "Default Group" as "Untitled" with the description "Update with own description".

1.2. Creating a group:
To manage and organize your tactics, you have the opportunity to add new groups to your collection. Just click on the "stack with a plus" button in the upper right corner of the screen and fill in the name and description of your new group.

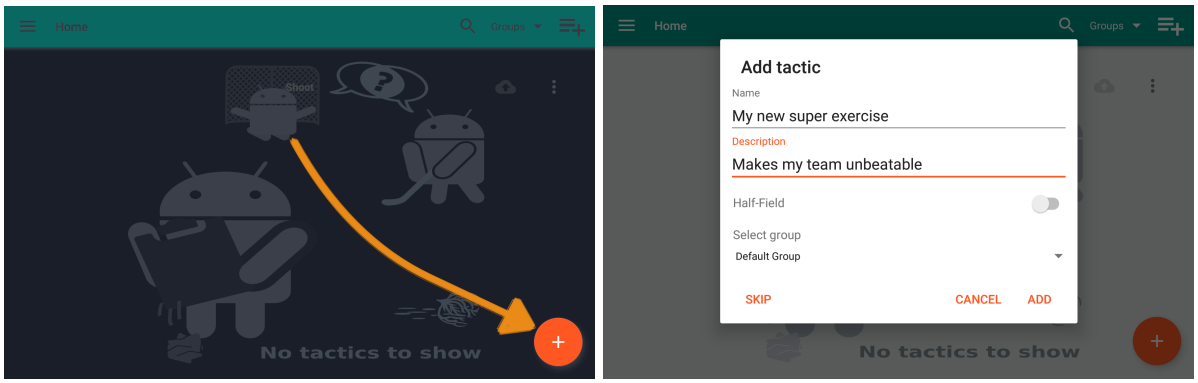


Abbildung 7.1: Add a new tactic

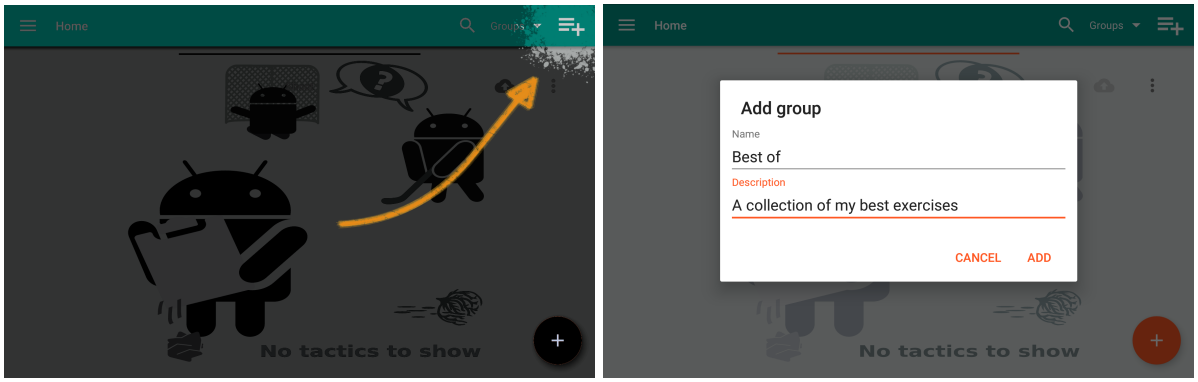


Abbildung 7.2: Add a new group

1.3. Filtering groups:

If you want to filter your collection by group to get a better overview, you can hide and show them using this feature.

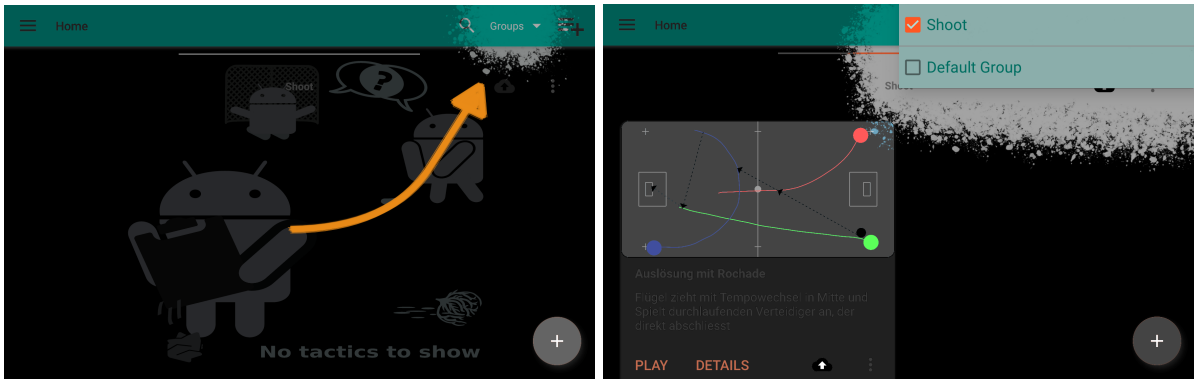


Abbildung 7.3: Filter by group

1.4. Searching tactics:

Searching a tactic by name or description can help you to find your desired tactic quickly. You can just hit the magnifier and input your search-term. The app will continuously update the results as you type. If you filtered the view by groups before and the search input matches any hidden tactics, the app lets you know by displaying a hint on the bottom. You can temporarily show these tactics by touching the information. After the next search update they will disappear again.

1.5. Editing a tactic:

To change the name, description or the group of a tactic, hit the three dots on the bottom right corner

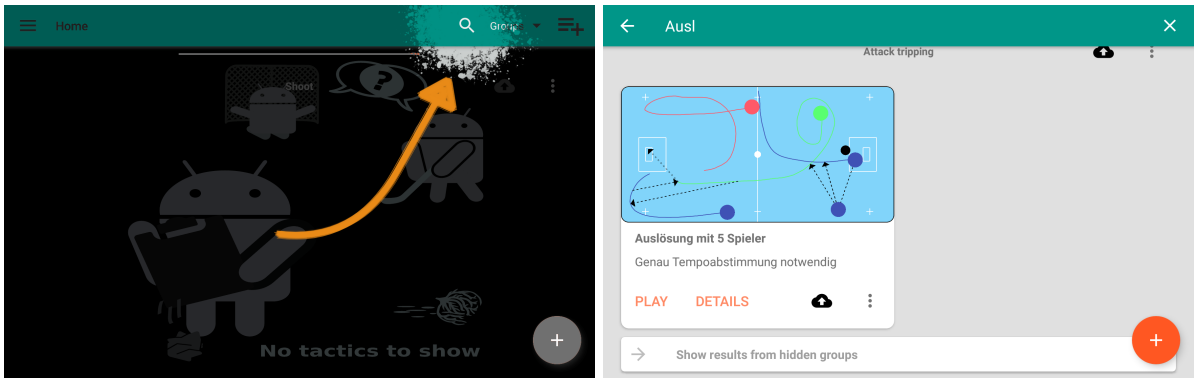


Abbildung 7.4: Search tactics

of a tactic and choose “edit“. A dialog, identical to the one you used to create a tactic, pops up, allowing you to change the values to your desire.

1.6. Editing groups:

You can change a group the same way you change a tactic. Hitting the three dots on the right side of the group's title and choosing "edit"will open the edit-dialog.

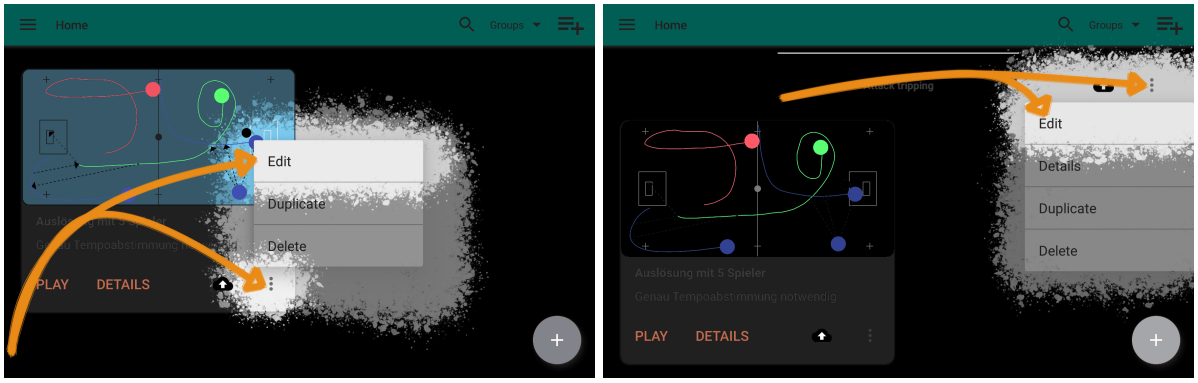


Abbildung 7.5: Edit a tactic or group

1.7. Uploading a tactic / group:

Sharing is caring, so give others the opportunity to benefit from your experience! To upload a tactic or even a whole group, you need to log in using your Google-Account. Click on the head-icon in the Navigation-Drawer and follow the instructions. After you successfully logged in, your mail address and your profile picture will be shown (if you have one). Now you are permitted to upload your own tactics. Just click on the black cloud with a arrow inside of it on the tactic / group (except for the "Default Group") you want to upload. If you're connected to the Internet, you'll be shown another edit-dialog, where you can change the name or description and choose a category in the store before starting the upload. After successfully uploading your tactic / your group, a check-icon inside of the black cloud will appear. It's not possible to re-upload a tactic or group, but if you want to upload an updated version, you can duplicate the group or tactic and just upload that one. Duplicated tactics and groups are allowed to be uploaded again, but it's recommended to first do some redesigning and editing using the tactics board.

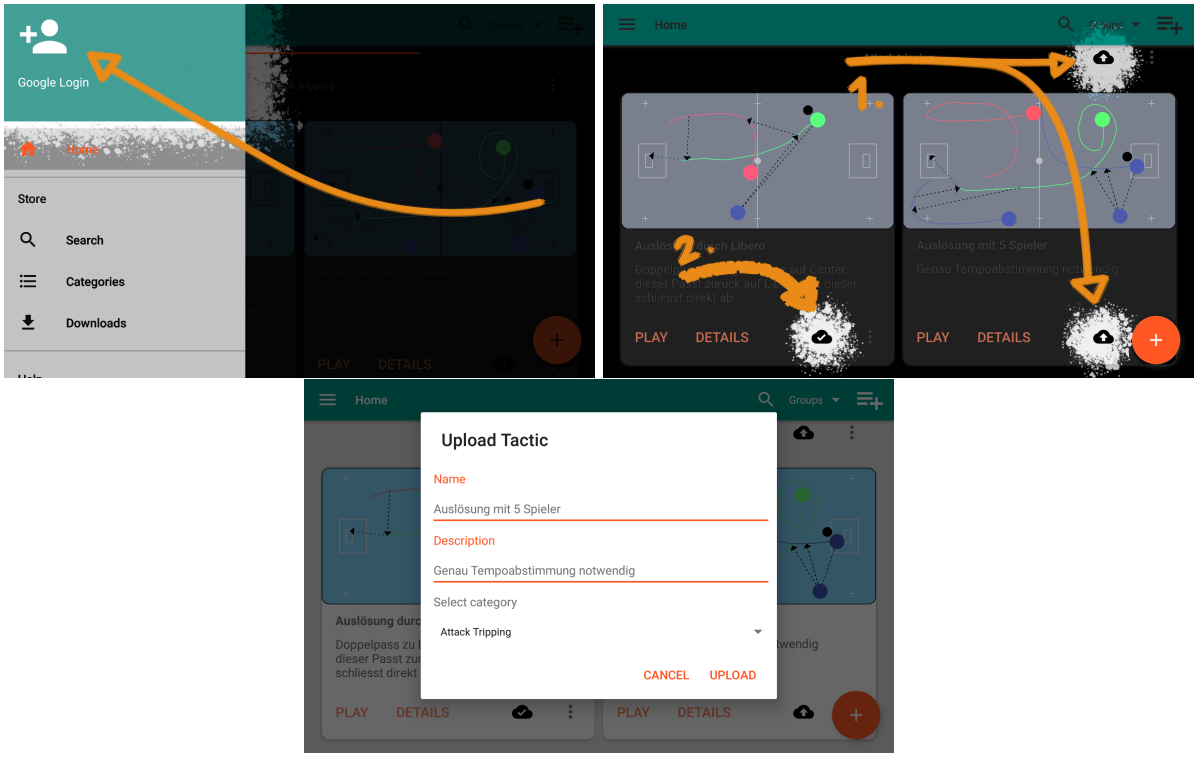


Abbildung 7.6: Upload tactic or group

1.8. Duplicating a tactic / group:
To duplicate your tactic or a group, you can click on the three dots and choose “duplicate“. Once the tactic or group is uploaded, you can not re-upload again. However, duplicating it allows you to upload the copied version.

1.9 Deleting a tactic / group:
Delete your tactic or group by clicking the three dots and selecting “delete“. Be aware that deleted tactics and groups are gone forever and cannot be restored (except you uploaded it).

2. Tactics board:
The tactics board let’s you design your own tactics and play them back. To provide you with the best possible experience, you should be familiar with a few important concepts.

2.1. Adding players to the tactics board
Players are the main part of a tactic. You can add a new player at any time. As many as you want. Players exist from the beginning until the end of a tactic, even if you add them later.

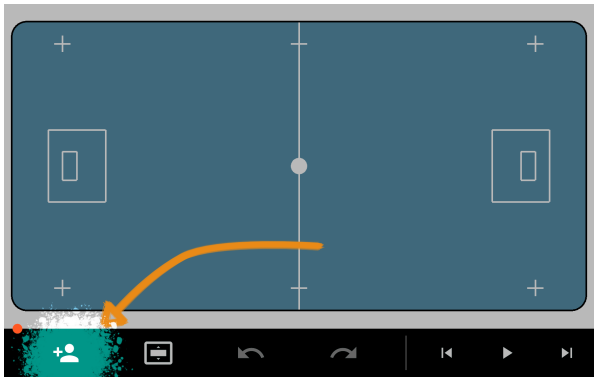


Abbildung 7.7: Add a player

2.2. Deleting a player:
If you want to delete a player, tap the desired token and hit the trash-can icon in the toolbar on the bottom. The player and his routes get deleted. This step is not revertable! If a pass or any other ball actions are registered to this player, unexpected behavior may occur. Instead of pestering you with restrictions and error-messages, we’ll just leave it to you to avoid those scenarios.

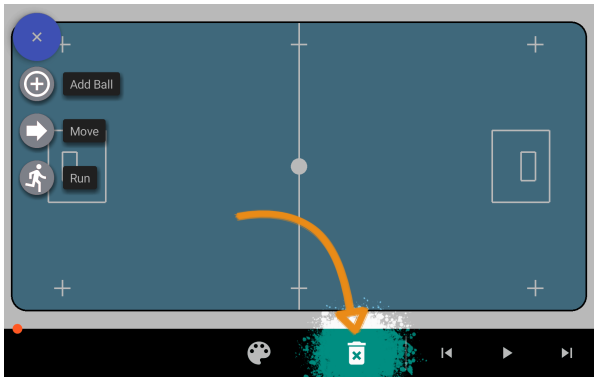


Abbildung 7.8: Delete a player

2.3. Moving a player:
Players can be moved to a new starting position. Keep in mind that moving your player is different to drawing a player's running path. Running means to add a new route to the player, while moving is just changing the position the player is starting from. After adding a running route, it's no longer possible to move the player.

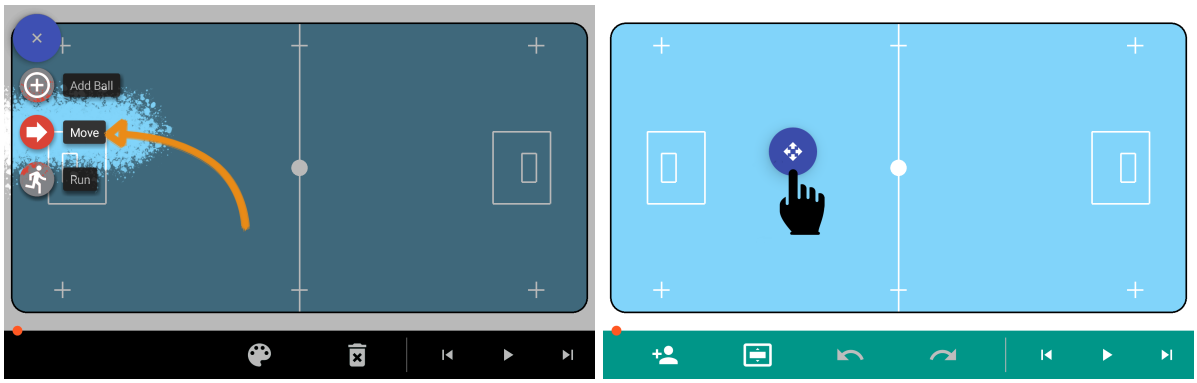


Abbildung 7.9: Move a player

2.4. Adding routes to a player:
To create a route on a player, just select it and hit "run". Now you can drag the token anywhere you want it to go and the player will follow the defined path. If you're not satisfied with the drawn route, undo it using the back-arrow in the toolbar on the bottom.

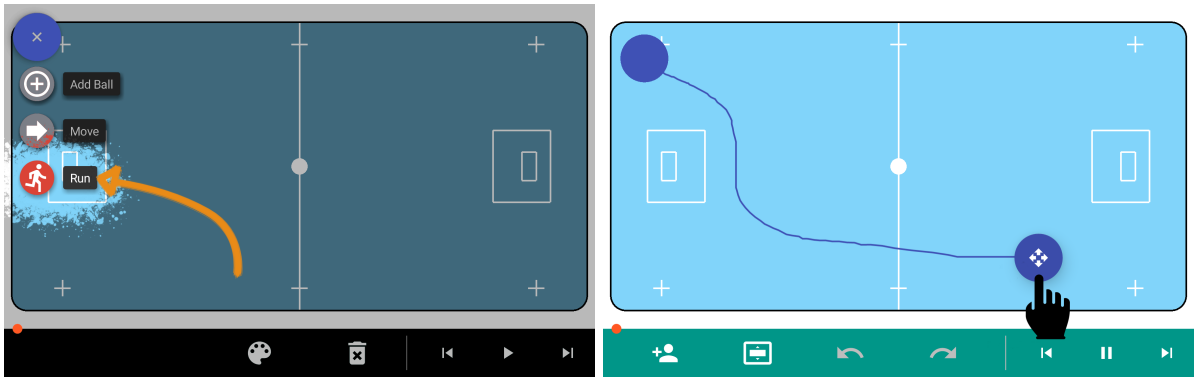


Abbildung 7.10: Defining a player's running route

2.5. Orchestrating simultaneous player movements:
Its possible and very easy to let players run simultaneously. Adjust the time by moving the dot in the orange slider on the top of the toolbar to the moment your player should start their simultaneous run. Select the player, which has no route at this point of time and hit run. As soon as your finger touches the player, the other one will start running it's route. Just draw while the other tokens movements are in progress and the route will be saved synchronously.

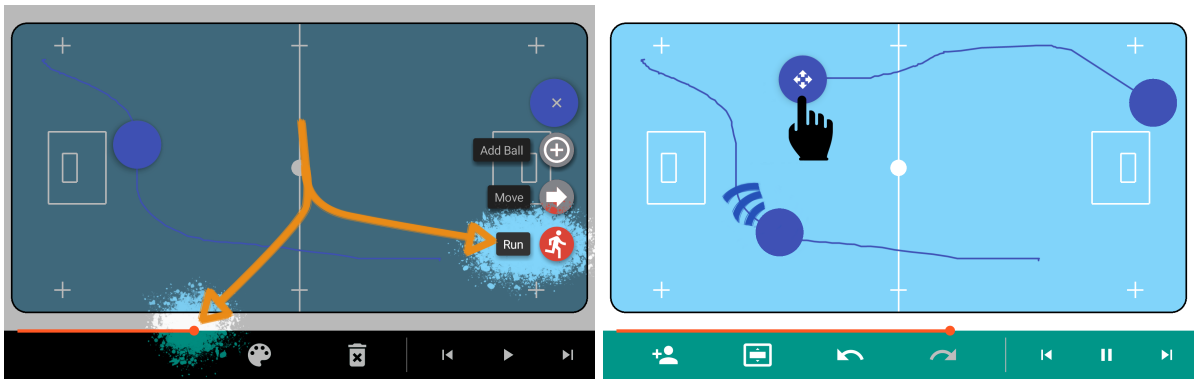


Abbildung 7.11: Define simultaneous player movements

2.6. Adding a ball:
You can add a ball to a player. The player has possession of the ball starting from the point of time you add it until you initiate a pass, a shot or the end of the tactic. Keep in mind that a player can only possess one ball at a time, but you can select him as your passing target, after he has lost possession of the first ball by a pass or a shot.

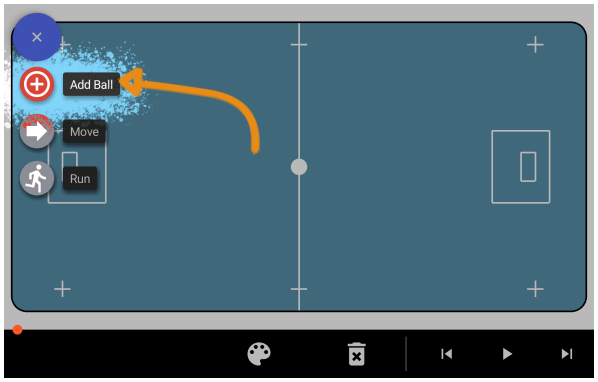


Abbildung 7.12: Add a ball

2.7. Deleting a ball:
Delete a ball by selecting it and hitting the trash-can in the toolbar on the bottom. The ball and it's route, including passes and shots will be deleted. This step is not revertable!

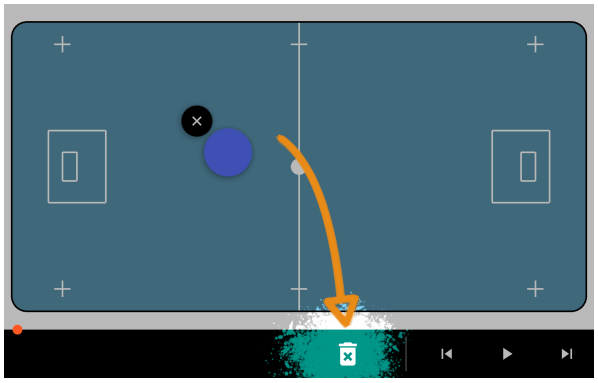


Abbildung 7.13: Delete a ball

2.8. Passing:
Passing is a main feature in the app. A pass can be performed by any player possessing a ball to an other player. It's possible to pass to another players route, but keep in mind, that this is only possible for positions the target is still moving to and not positions the target already passed by by the moment you initiate the pass! You may have to rewind to a time the target didn't pass the desired position, if you want to pass to that exact spot. To pass a ball, click on a player in possession of a ball and choose "Pass". Slide your finger across the screen to another player or it's route. If you see a red circle around your finger, a pass can be performed. The passing speed is begin calculated by staring time of the pass and the time when the target player arrives at the selected spot.

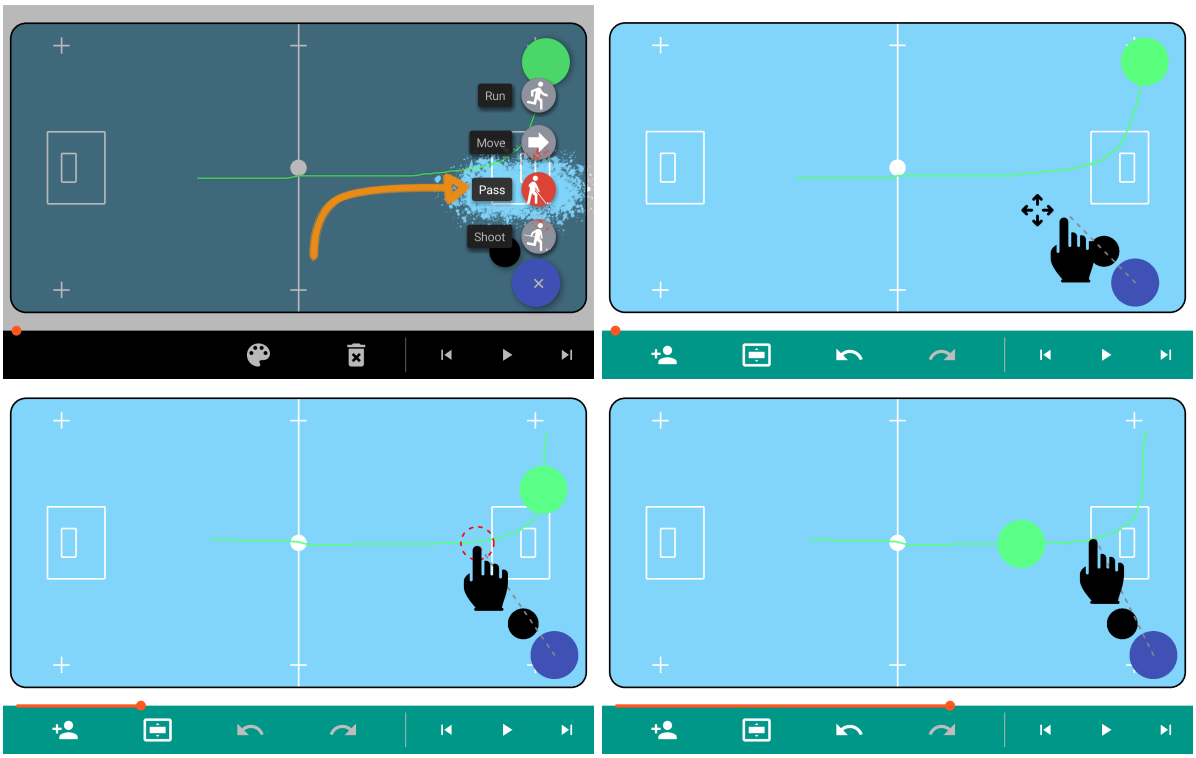


Abbildung 7.14: Passing

2.9. Shooting:
If a player is in possession of a ball, he can perform a shot on a goal. Click on a player in possession of a ball and choose “Shoot“. Slide your finger across the screen over the goal of your desire. A red circle around your finger indicates valid targets for the shot.

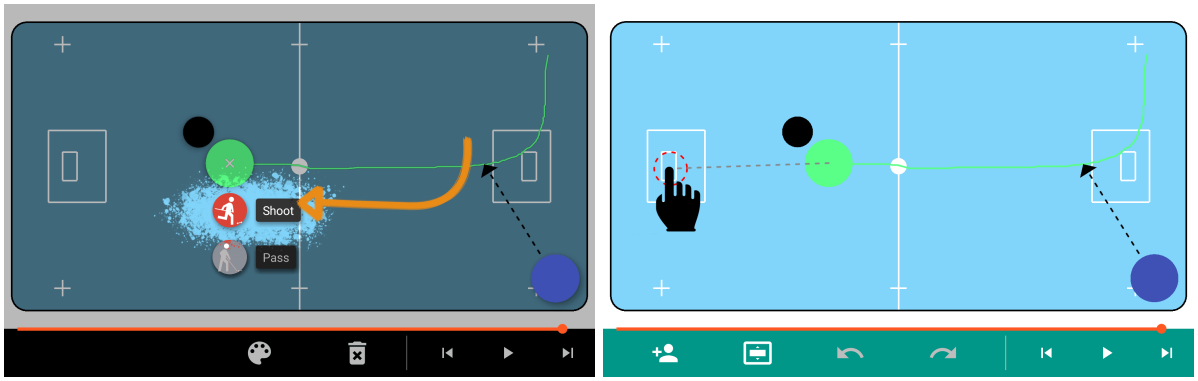


Abbildung 7.15: Shooting

2.10. Changing the color of a player:
After selecting a player on the field, hit the color-palette button. A color-picking dialog pops up, where you can adjust the player’s color to your liking.

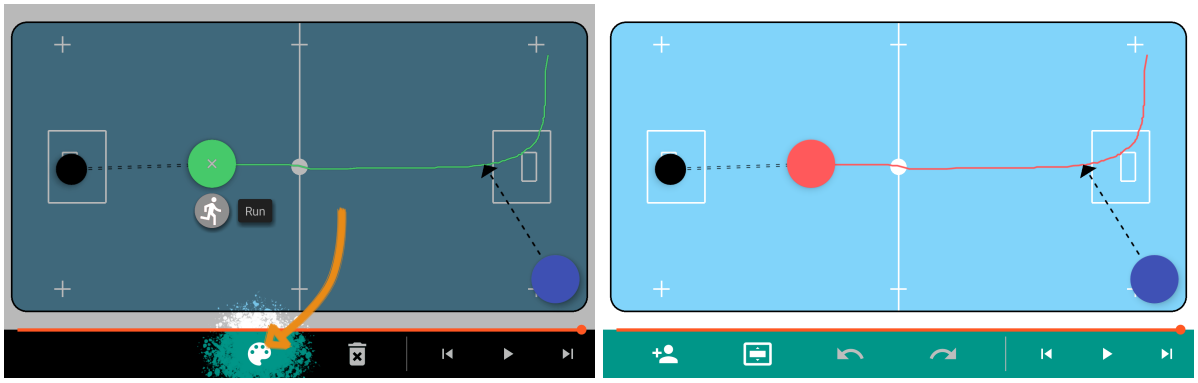


Abbildung 7.16: Change a player’s color

2.11. Changing the field size:
You can change the field size from it’s original aspect ratio to a fullscreen mode. The previously defined routes and positions will be stretched in order to maintain their original definitions.

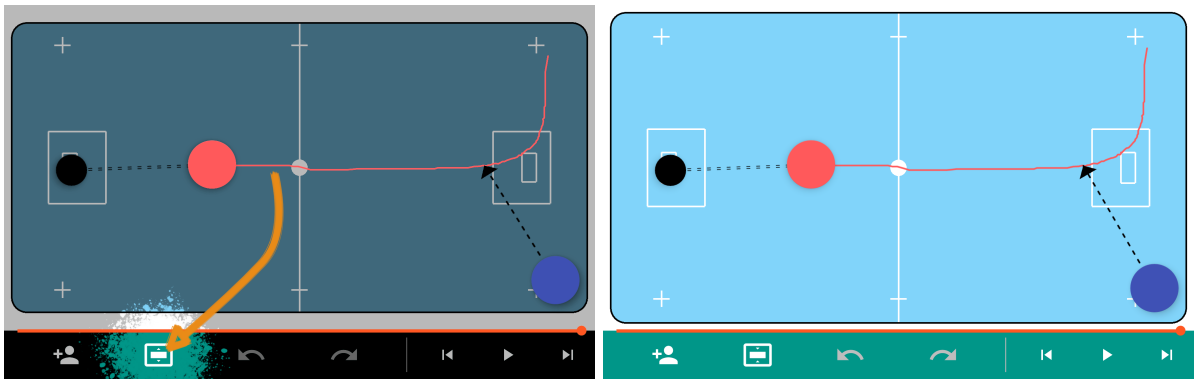


Abbildung 7.17: Delete a ball

2.12. Saving a tactic:
To save a tactic, just press the back button to return to the home screen. The tactic will be saved and updated automatically.

3. Store:
Have a look at other user's tactics and exercises - download, play and edit them! With the tactic store you can benefit from other people's work!

3.1. Overview:
You can scroll the store view just like you would on the home view, but tactics and groups are presented the same way in the store. You can click the group entry's picture to switch to it's next tactic picture. On both entries, tactics and groups alike, you can have a look at the detail view. You can also swipe to the "single" or "groups"tab, where tactics and groups are not mixed up.

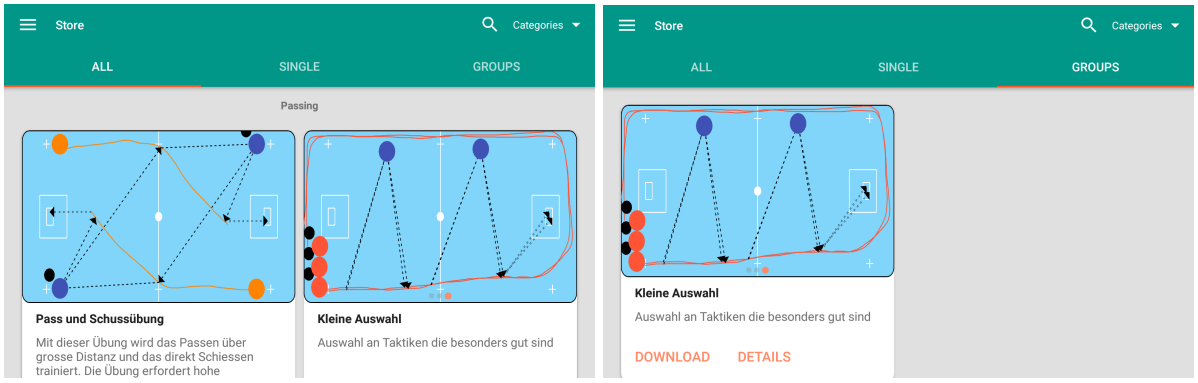


Abbildung 7.18: Store

3.2. Filtering categories:
To get a better overview, you can filter the entries by category, just like the group-filtering in the home view. Click on the "Categories" dropdown and select or unselect categories to your liking.

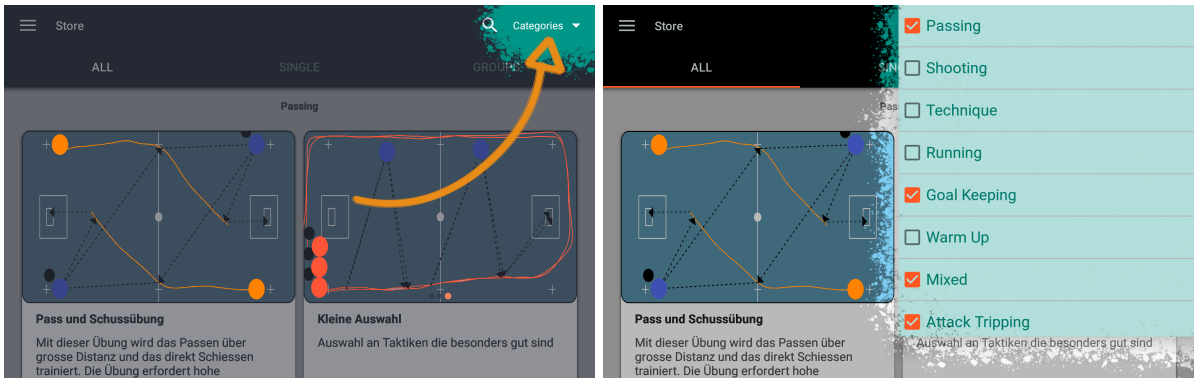


Abbildung 7.19: Filter categories

3.3. Search:

To perform a search in the store, click on the magnifier icon and enter your search term. Unlike the home screen, you have to hit the enter or the magnifier-key on the keyboard. The search then loads all tactics and groups matching input. To clear the search restrictions, use the arrow in the top left or the cross on the top right corner of the screen.

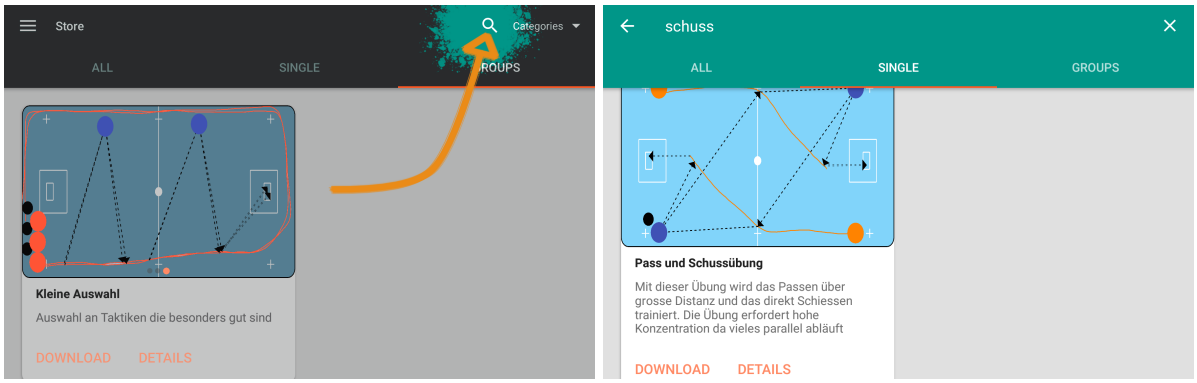


Abbildung 7.20: Search store

3.4. Downloading a tactic / group:

If you want to download a tactic or a whole group, just click on the “Download“ button. If you’re not connected to the internet, the app will inform you thusly. Depending on your connection speed the download may take a while.

3.5. Categories:

In the categories view, you can see all available categories. If you click on one of them, you’ll be taken to the store, where you’ll be presented all tactics and groups of that category.

Abbildung 7.21: Categories overview

3.6. Downloads:

You'll find all of your downloaded tactics and groups in the downloads view. You can play the tactics back by hitting the play button on the card. If you want to play back a tactic of a group, open the details dialog by clicking on the "Details" button and hit the play button on the tactic of your choice.

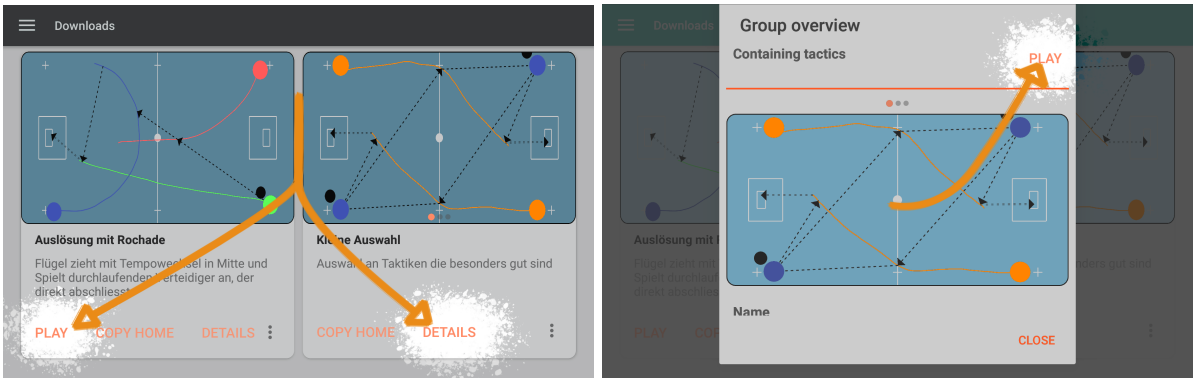


Abbildung 7.22: Downloads overview and play

3.7. Copy to Home:

If you want to edit a downloaded tactic or just add it to your home screen, you can copy a tactic or a whole group to home. Click on the "Copy Home" button. A dialog pops up, allowing you to change the name, description and - if you're copying a single tactic - assign it a group.

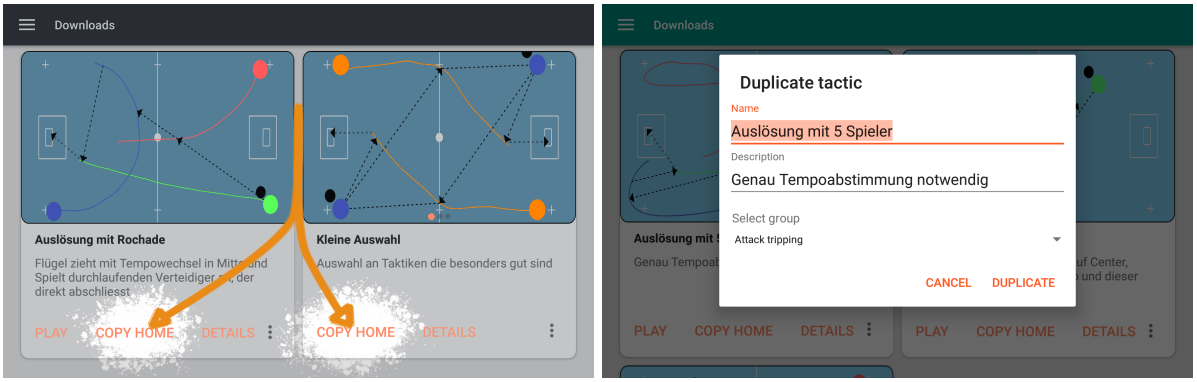


Abbildung 7.23: Copy to Home

3.8. Deleting from downloads:

Delete your downloaded tactics or groups by clicking on the three dots and selecting "delete".

F Feedback von Michael Kuhn, Trainer UHC Lokomotive Stäfa

1. Grundsätzlich besteht das Projekt aus Unterprojekten
   (a) Taktikboard: Eingabe und Wiedergabe einer Übung/ eines Spielzugs
   (b) Verwalten von Übungen (speichern, suchen, gruppieren, etc.)
      i. Lokal (jeder Trainer für sich)
      ii. Sharing im Internet (Trainer untereinander)

   Ich persönlich würde diese Aspekte im Dokument stärker trennen. Das Taktikboard an sich wäre eine intressante Anwendung(speichern unter einem Filename müsste möglich sein). Auch die Verwaltung von Übungen(z.B. nur als Bild vorhanden) für sich wäre eine interessante Anwendugn, auch schon nur lokal.

2. Was genua ist eine "Gruppe/Kategorie"? Das wurde mir nicht so ganz klar. Wenn ich das richtig verstehe, ist eine Taktik (=Übung) immer in genau einer Gruppe und in keiner oder einer Kategorie (korrekt?). Für mich stellt sich die Frage ob das flexibel genug ist? Ich kann mir verschiedene Kriterien vorstellen, nach welchen ich Übungen kategorisieren möchte, zB:
   (a) Wann ich die Übung im Training verwenden würde: Einspielübung vs taktische Übung, etc.
   (b) Was die Übung zum Ziel hat: Goalietraining, Schusstraining, Zweikampftraining, Spielverständnis, etc.
   (c) Zeitaufwand einer Übung (5min, 10min, 20min, etc.)
   (d) Schwierigkeitsgrad und Zielgruppe (v.a. wichtig beim Sharing) einer Übung (Junioren vs. Hobbysportler vs NLA, anspruchsvoll vs. flexibel vs einfach)
   (e) Wie viele Spieler involviert sind: Übung mit 3 Goalies, kann ich die Übung machen, wenn wir nur 12 Feldspieler haben? Die Übung funktioniert in 3er Gruppen, d.h. idealerweise ist die Anzahl Spieler z.B. ein Vielfaches von 3+2 zum Goalie-Einschiessen, etc.
   (f) Variationen einer Übung: Zuerst ohne Verteidiger, dann mit Verteidiger, etc. oder zuerst mit Ballannahme, dann mit Direktschuss

   Eine Alternative zu diesen relativ starren Gruppen (wenn ichs denn richtig verstanden habe) könnte "Tagging"sein. D.h. eine "Taktik"kann mit mehreren Tags versehen werden (n-to-m Beziehung). D.h. eine Übung könnte z.B. mit *Goalietraining, Goalie-Verschieben, Einspielübung, und Direktschusstraining* getagged sein, oder mit *Hinterherlaufen, Druck-auf-Slot, Entscheidungen-Treffen.* Anzahl Spieler/Zeitaufwand wäre damit wohl nicht gelöst (oder nur "unstrukturiert"). Auch über die Variationen lohnt es sich evtl. nochmals nachzudenken ...

3. Verwaltung (lokal)
   (a) Wunschfunktion: Ich komm ins Training, gebe die Anzahl Spieler und ein Thema (z.B Hinterherlaufen/Gegenbewegung) ein, und kriege für jeden Trainingsblock einen Vorschlag (z.B. Einspielen im 8, parallel dazu in der anderen Hälfte Übung zu kreuzen, danach Schussübung mit Kreuzen/ Auslösen mit Hinterherlaufen, danach Spielform "Vorwärtspässe nur direkt")
   (b) Nicht alle Übungen die dazu interessant wären, lassen sich einfach als Spielzug darstellen (z.B. Über-/Unterzahl). D.h. es müsste dazu auch möglich sein, Übungen die nicht als Taktikboard (sondern z.B. nur als Text) dargestellt sind, zu verwalten.
   (c) Bewerten von Taktiken (lokal und global)
   (d) Wie genau funktioniert die Suche? Textuell (andere Varianten)? In Notitzen/Gruppennamen/-Kategorien? Texteingabe (tendenziell mühsam auf mobile) vs browsing?

4. Sharing:
   (a) Was passiert wenn ich eine Taktik share, und dann später (lokal) eine Änderung mache? Kann ich diese Synchronisieren? Kriegen User, welche die Taktik bezogen haben, ein (optionales?) Update?

5. Taktikboard

   (a) Sicher hilfreich wäre es, einfach Notitzen zu einer Übung ablegen zu können (nicht nur beim Erstellen, sondern auch direkt im Training, wenn man z.B. fürs nächste mal etwas verbessern oder in der Erklärung hervorheben möchte). Vielleicht auch via Audio (schneller), entweder mit Text-to-Speech oder einfach zur späteren Nachbearbeitung. Dann müsste man auch schnell alle Taktiken mit solchen (zur Nachbearbeitung gedachten) Audio-Notitzen finden können.

6. Details:

   (a) Spielerbewegung mit Ball = Ballbewegung?

   (b) Was bedeuten 100 User? 100 weekly User? Oder 100 User die einen Zugriff innerhalb der gleichen Sekunde machen?

   (c) Rechtliche Aspekte beim sharing (edit und reshare). Terms of Service?

Literaturverzeichnis

[1] Silvan Habegger & Luca Aquino. Sitzungsprotokolle. http://sinv-56027.edu.hsr.ch/redmine/projects/floorball-tactics-board/wiki/Meeting_Protokoll. Username: guest; Password: guest; Accessed: 2016-06-17.

[2] Silvan Habegger & Luca Aquino. Sitzungstraktanden. http://sinv-56027.edu.hsr.ch/redmine/projects/floorball-tactics-board/wiki/Meeting_Traktanden. Username: guest; Password: guest; Accessed: 2016-06-17.

[3] codeproject.com. MVVM-pattern beschreibung. http://www.codeproject.com/Articles/100175/Model-View-ViewModel-MVVM-Explained. Accessed: 2016-06-03.

[4] codeproject.com. View Patterns Vergleich. http://www.codeproject.com/Articles/31210/Model-View-Presenter. Accessed: 2016-06-03.

[5] Martin Fowler. GUI architectures. http://martinfowler.com/eaaDev/uiArchs.html. Accessed: 2016-06-03.

[6] Google. Actionbar. http://developer.android.com/design/patterns/actionbar.html. Accessed: 2016-03-03.

[7] Google. Back and Up Button. http://developer.android.com/design/patterns/navigation.html. Accessed: 2016-03-03.

[8] Google. Canvas. https://developer.android.com/reference/android/graphics/Canvas.html. Accessed: 2016-06-14.

[9] Google. Cards. https://www.google.com/design/spec/components/cards.html. Accessed: 2016-03-03.

[10] Google. Fragments. https://developer.android.com/guide/components/fragments.html. Accessed: 2016-06-14.

[11] Google. Navigation. https://developer.android.com/training/design-navigation/descendant-lateral.html. Accessed: 2016-06-13.

[12] Google. Navigation Drawer. https://www.google.com/design/spec/patterns/navigation-drawer.html. Accessed: 2016-03-03.

[13] Google Inc. Caching bitmaps. https://developer.android.com/training/displaying-bitmaps/cache-bitmap.html. Accessed: 2016-06-10.

[14] Google Inc. Managing bitmaps memory. https://developer.android.com/training/displaying-bitmaps/manage-memory.html. Accessed: 2016-06-10.

[15] Google Inc. Managing your apps memory. https://developer.android.com/training/articles/memory.html. Accessed: 2016-06-10.

[16] Antonio Levia. MVP in Android. http://antonioleiva.com/mvp-android/. Accessed: 2016-06-03.

[17] Pedro Okawa. Greendao schema update and data migration. http://stackoverflow.com/a/30334668. Accessed: 2016-06-10.

[18] Mirko Stocker. *BA-Aufgabenstellung-AquinoHabegger.pdf*.

[19] The Free Encyclopedia Wikipedia. Bitmap. https://de.wikipedia.org/wiki/Bitmap. Accessed: 2016-06-14.

[20] The Free Encyclopedia Wikipedia. Memory-Swap. https://de.wikipedia.org/wiki/Virtuelle_Speicherverwaltung. Accessed: 2016-06-14.

[21] The Free Encyclopedia Wikipedia. OR-Mapper. https://en.wikipedia.org/wiki/Object-relational_mapping. Accessed: 2016-06-14.

[22] The Free Encyclopedia Wikipedia. Precompiler. https://de.wikipedia.org/wiki/Precompiler. Accessed: 2016-06-14.

[23] The Free Encyclopedia Wikipedia. Toolbar. https://en.wikipedia.org/wiki/Toolbar. Accessed: 2016-06-14.

[24] The Free Encyclopedia Wikipedia. Weak-Reference. https://en.wikipedia.org/wiki/Weak_reference. Accessed: 2016-06-14.

Abbildungsverzeichnis

1 Taktik-Tafel . . . 2
2 Resultat Tactics-Board Android Applikation . . . 3
3 Online-Store der Applikation . . . 3

1.1 Use Case Diagramm . . . 12
1.2 Domain-Modell . . . 24

2.1 MVC-Pattern [4] . . . 28
2.2 MVP-Pattern [4] . . . 28
2.3 MVVM-Pattern [3] . . . 28
2.4 Sequenzdiagramm “Zeitliche Navigation“ . . . 33
2.5 Sequenzdiagramm “Taktikwiedergabe“ . . . 33
2.6 Sequenzdiagramm “Passen“ . . . 34
2.7 Deployment-Diagramm . . . 35

3.1 Android Farbprofile und Taktikübersicht mit Rot-Grün-Sehschwäche . . . 39
3.2 Skip-Button beim Erstellen einer Taktik und die unbenannte Taktik in der Übersicht . . . 40
3.3 Entity-Relation-Diagramm . . . 41
3.4 Android Bildschirm . . . 41
3.5 Lokale Taktikübersicht . . . 44
3.6 Taktiktafel Bearbeitungsmodus . . . 45
3.7 Taktik Abspielmodus . . . 46
3.8 Taktik-Store Suche . . . 47
3.9 Kategorie-Übersicht . . . 48
3.10 Übersicht Uploads . . . 49
3.11 Übersicht Downloads . . . 50
3.12 Vorschau Taktik . . . 51
3.13 Vorschau Gruppe . . . 52

4.1 Zeitaufwand Floorball Tactics-Board . . . 53

5.1 Taktik-Aktionen und Editier-Dialog in der lokalen Taktikübersicht . . . 56
5.2 Erstell- und Bearbeitungs-Dialog für Gruppen . . . 57
5.3 Textsuche und Gruppenfilter in der lokalen Taktikübersicht . . . 57
5.4 Taktik-Board und Playback-Ansicht . . . 58
5.5 Upload-Buttons für Taktiken und Gruppen mit Upload-Dialog-Beispiel . . . 59
5.6 Downloads-Übersicht mit Duplizier-Dialog (für das Kopieren einer Taktik oder Gruppe in die lokale Übersicht) . . . 59
5.7 Taktiken und Gruppen im Online-Store . . . 60
5.8 Detailansicht Taktik und Gruppe . . . 60
5.9 Taktik- und Gruppen-Administration . . . 61
5.10 Kategorie- und Admin-Verwaltung . . . 61
5.11 Hinweis auf versteckte Suchergebnisse . . . 62
5.12 Feldskalierung Standard und Bildschirm füllend . . . 62
5.13 Google OAuth Android und Web-App . . . 64
5.14 Retrofit Kommunikationsstack . . . 65

7.1 Add a new tactic . . . 96

7.2 Add a new group . . . 96
7.3 Filter by group . . . 96
7.4 Search tactics . . . 97
7.5 Edit a tactic or group . . . 98
7.6 Upload tactic or group . . . 98
7.7 Add a player . . . 99
7.8 Delete a player . . . 99
7.9 Move a player . . . 100
7.10 Defining a player's running route . . . 100
7.11 Define simultaneous player movements . . . 101
7.12 Add a ball . . . 101
7.13 Delete a ball . . . 102
7.14 Passing . . . 102
7.15 Shooting . . . 103
7.16 Change a player's color . . . 103
7.17 Delete a ball . . . 103
7.18 Store . . . 104
7.19 Filter categories . . . 104
7.20 Search store . . . 105
7.21 Categories overview . . . 105
7.22 Downloads overview and play . . . 106
7.23 Copy to Home . . . 106