



HTL Villach
Future Inside



Dokumentation

Moskitos-Soccer-App



Raphael Moser
Elias Santner
Martin Sonnberger
Marco Wilscher

Inhalt

Einleitung.....	5
Storymap	7
Product Backlog.....	7
User	7
Userstory: Anmelden.....	7
Userstory: Benutzerdaten ändern.....	8
Userstory: Statistiken ansehen.....	9
Admin	10
Userstory: Spieler anlegen	10
Userstory: Spieler bearbeiten	11
Userstory: Spieler löschen.....	11
Userstory: Spiel anlegen.....	12
Userstory: Spiel ändern	12
Userstory: Spiel löschen	13
Supporting Stories	13
Story: Einstellungen.....	13
Story: Design.....	13
Story: Menü	13
Story: Offline.....	13
Klassendiagramm	14
ER-Diagramm	15
Webservice	16
General	16
Data Model.....	16
Methods	17
GameResource	17
updateGame.....	18
getGamesByDate	19
getGamesByPlayerId (and between 2 Dates).....	20
insertGame	21
deleteGame	22
PlayerResource	23
getAllPlayers	23
updatePlayer	24
setPositions	25
getPlayerByUsername	25

insertPlayer.....	27
deletePlayer.....	28
login	29
setPassword.....	29
setGeoLocation.....	30
getGeoLocation	30
getPlayersNearby	31
getAllParticipations	32
getParticipationByld	33
getParticipationByld	34
insertParticipation	35
updateParticipation.....	36
deleteParticipation	36
Architektur – Programmierung	37
Login Webservice	37
Login App.....	38
Kommunikation (App und Webservice)	39
Kommunikation in der App	39
Arbeitsaufteilung.....	40
User-Guide Moskitos SoccerApp.....	41
Einleitung.....	41
Offline-Modus	41
Login	41
Hauptansicht	41
Eigene Benutzerdaten bearbeiten	42
Bestenliste	43
QR-Code anzeigen	43
Einstellungen	44
Sprache wechseln	44
Nachrichtenart Überblendung	44
Impressum.....	44
Admin-Guide Moskitos SoccerApp.....	45
Einleitung.....	45
Offline-Modus	45
Hauptansicht	45
Benutzerdaten bearbeiten	45
Spieler hinzufügen.....	46

Spiel hinzufügen	46
Spielerauswahl	46
Teamzuweisung	46
Ergebniseingabe	47
Spiel bearbeiten/löschen	48
QR-Code Scanner	48
Einstellungen	48
Teamzuweisung	48
Prozessfortschritt	49

Einleitung

Auftraggeber: Prof. DI Gernot OBERLERCHER

Scrummaster: OStR Prof. DI Heidrun MÜLLER-STEGMÜLLER

Programmiertechnische Beratung: Prof. Mag. Gerald ORTNER

Beginn: 15.März 2017

Ende: 14.Juni.2017

Teammitglieder:

- Moser Raphael
- Santner Elias
- Sonnberger Martin
- Wilscher Mario

Auftrag: Der Fußballclub Moskitos benötigt eine Android-App, die es ermöglicht Spielergebnisse festzuhalten.

Jedem Spieler steht ein Benutzerkonto zur Verfügung, wobei diese sich in normale Benutzer und Admins unterteilen. In diesem werden neben Benutzername und Passwort, für die Anmeldung in der App, auch die bevorzugte Position am Spielfeld gespeichert.

In einem Spiel wird festgehalten, wann es stattgefunden hat, welcher Spieler in welchem Team gespielt hat, sowie dessen Position am Spielfeld und das Endergebnis. Die Teameinteilung und Positionszuweisung erfolgt bei der Spielerstellung entweder manuell vom Spielersteller oder automatisch durch eine Funktion in der App.

Die Statistik wird automatisch aus den Spielergebnissen generiert. Die Statistik besteht aus folgenden Punkten:

- Anzahl der teilgenommenen Spiele, die gespielte Position und ob dieses mit einem Sieg, Niederlage oder mit einem Unentschieden geendet hat
- Durchschnittliche Tordifferenz
- Geschossene Tore (Gesamt)
- Normale Tore
- Kopfballtore
- Kopfballtore bei Schnee
- Elfmertore
- Bekommene Tore
- Geschossene Gurken

Funktionsumfang: Der Funktionsumfang ist davon abhängig, ob es sich um einen normalen Benutzer oder Admin handelt, da dem Admin zusätzlich zu den normalen Funktionen, die Verwaltung der Spieler und Spiele zu Verfügung steht.

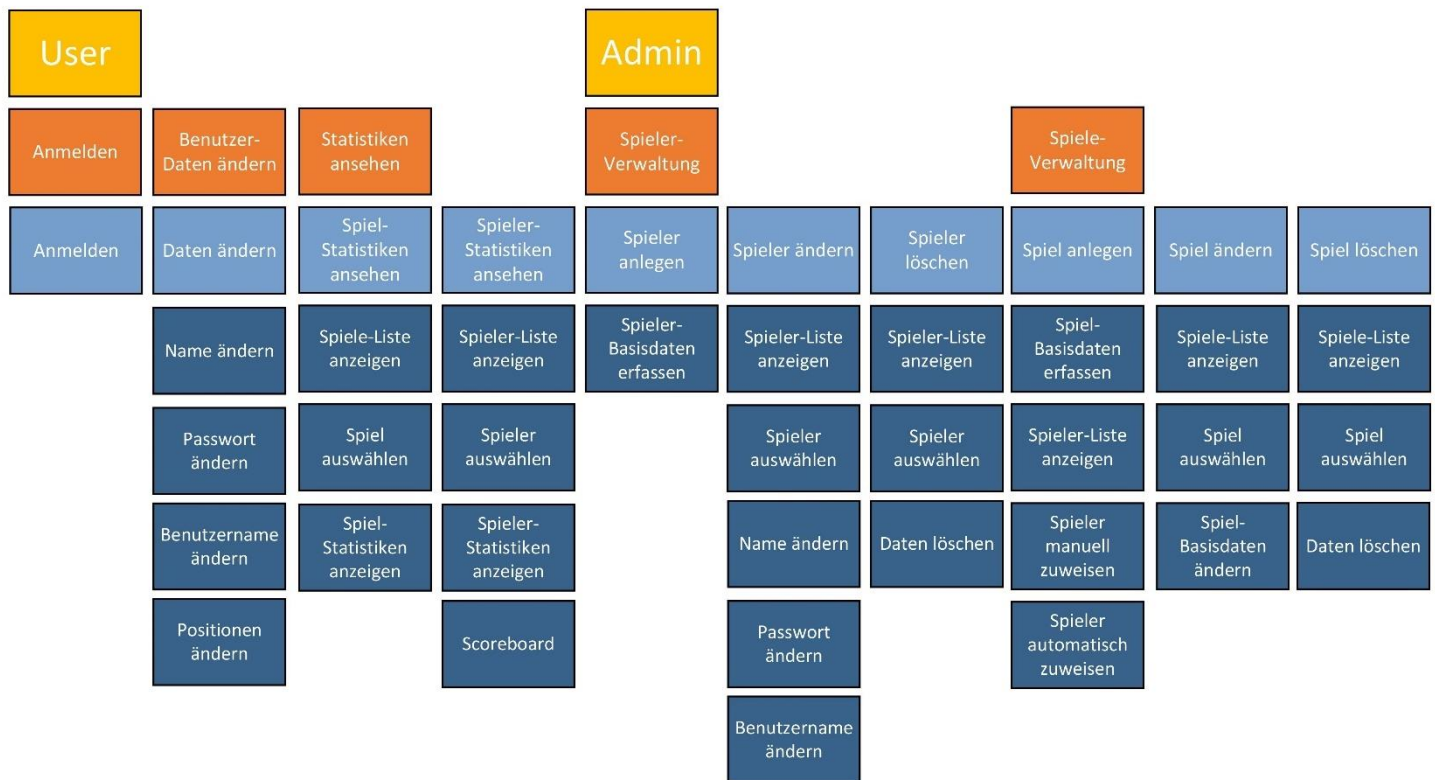
Benutzer:

- Wunschpositionen festlegen
- Spielergebnisse ansehen
- Spieler Statistiken ansehen

Admin:

- Neue Benutzer anlegen
- Bestehende Benutzer bearbeiten
- Bestehende Benutzer löschen
- Neue Spiele anlegen
- Bestehende Spiele bearbeiten
- Bestehende Spiele löschen

Storymap



Product Backlog

User

Diese Userstories sind allgemein für jeden User.

Userstory: Anmelden

Ziel dieser Userstory ist es, dass sich der User anmelden kann, um auf die ihm zur Verfügung stehenden Funktionen Zugriff zu erhalten.

Akzeptanzkriterium:

Durch Eingabe der richtigen Benutzername-Passwort-Kombination wird der User angemeldet und erhält so Zugriff auf die weiteren Funktionen.

Testfälle:

- 1. Vorbedingung:** User „Admin“ mit Passwort „Test“ existiert.
Aktion: User gibt „Admin“ als Benutzername und „Test“ als Passwort ein und klickt auf Anmelden.
Ergebnis: User erhält Zugriff auf die App.
- 2. Aktion:** User gibt **keinen** Benutzernamen oder **kein** Passwort ein und klickt auf Anmelden.
Ergebnis: User erhält eine Meldung, dass er Benutzername und Passwort eingeben muss, um sich anmelden zu können.

3. **Vorbedingung:** User „Ad“ mit Passwort „Test“ existiert **nicht**.
Aktion: User gibt „Ad“ als Benutzernamen und „Test“ als Passwort ein und klickt auf Anmelden.
Ergebnis: User erhält eine Meldung, dass Benutzername oder Passwort ungültig ist.
4. **Vorbedingung:** User „Admin“ hat **nicht** das Passwort „xxx“.
Aktion: User gibt „Admin“ als Benutzernamen und „xxx“ als Passwort ein und klickt auf Anmelden.
Ergebnis: User erhält eine Meldung, dass Benutzername oder Passwort ungültig ist.

Userstory: Benutzerdaten ändern

Ziel dieser Userstory ist es, dass der User seine Benutzerdaten ändern kann, damit sein Profil richtig festgehalten wird.

Akzeptanzkriterium:

- User kann seinen Namen ändern, solange zwischen 3 und 25 Zeichen ist.
- User kann seinen Benutzernamen ändern, solange dieser zwischen 3 und 20 Zeichen ist, von keinem anderen User genutzt wird und nur aus Zahlen und Buchstaben besteht.
- User kann sein Passwort ändern, solange dieses länger als 5 Zeichen ist.
- User kann seine Positionen auf denen er spielen möchte auswählen, wobei er mindestens eine Position ausgewählt haben muss.

Testfälle:

1. **Vorbedingung:** User ist eingeloggt und **kein** Spieler hat den Username „Test“.
Aktion: User gibt „Test“ als Benutzername ein und drückt auf Speichern.
Ergebnis: User bekommt Benachrichtigung, dass seine Benutzerdaten aktualisiert wurden.
2. **Vorbedingung:** User ist eingeloggt.
Aktion: User gibt „Test1“ als Passwort ein.
Ergebnis: User bekommt Benachrichtigung, dass seine Benutzerdaten aktualisiert wurden.
3. **Vorbedingung:** User ist eingeloggt und ein Spieler hat den Username „Admin“.
Aktion: User gibt „Admin“ als Benutzername ein und drückt auf Speichern.

Ergebnis: User bekommt Benachrichtigung, dass dieser Benutzername bereits in Nutzung ist.

4. **Vorbedingung:** User ist eingeloggt.

Aktion: User wählt **keine** Position auf der er spielen möchte aus und drückt auf Speichern.

Ergebnis: User bekommt Benachrichtigung, dass er mindestens eine Position auswählen muss.

5. **Vorbedingung:** User ist eingeloggt.

Aktion: User gibt „ad“ als Benutzername ein und drückt auf Speichern.

Ergebnis: User bekommt Benachrichtigung, dass sein Benutzername nicht kürzer als 3 Zeichen sein darf.

6. **Vorbedingung:** User ist eingeloggt.

Aktion: User gibt „ganzLangerBenutzerNameDerZiemlichLangIst“ als Name ein und drückt auf Speichern.

Ergebnis: User bekommt Benachrichtigung, dass sein Name nicht länger als 25 Zeichen sein darf.

7. **Vorbedingung:** User ist eingeloggt.

Aktion: User gibt „Test!?“ als Benutzername ein und drückt auf Speichern.

Ergebnis: User bekommt Benachrichtigung, dass sein Benutzername nur aus Buchstaben und Zahlen bestehen darf.

Userstory: Statistiken ansehen

Ziel dieser Userstory ist es, dass der User Statistiken einsehen kann.

Akzeptanzkriterium:

- User kann Statistik eines Spielers einsehen.
- User kann Statistik eines Spiels einsehen.
- User kann sich auf dem Scoreboard mit anderen Spielern vergleichen.

Testfälle:

1. **Vorbedingung:** Es gibt mindestens einen Spieler.
Aktion: User wählt Spieler aus, dessen Statistik er ansehen will.

Ergebnis: User sieht Statistik des Spielers.

2. **Vorbedingung:** Es gibt mindestens einen Spieler.
Aktion: User öffnet das Scoreboard.

Ergebnis: User sieht die Spieler sortiert nach der Leistung.

3. **Vorbedingung:** Es gibt mindestens ein Spiel.
Aktion: User wählt Spiel aus, von dem er die Statistik sehen will.

Ergebnis: User sieht Statistik des Spiels.

Admin

Diese Userstories sind nur für Admins zugänglich. Damit diese Userstories funktionieren besteht die Vorbedingung, dass der User Admin-Rechte hat.

Userstory: Spieler anlegen

Ziel dieser Userstory ist es, dass ein Admin neue Benutzerkonten anlegen kann, damit auch neu hinzukommende Spieler die App benutzen können.

Akzeptanzkriterium:

- Admin kann einen Namen für den neuen User eintragen, solange dieser zwischen 25 und 3 Zeichen ist.
- Admin kann einen Benutzernamen für den neuen User eintragen, solange dieser zwischen 3 und 20 Zeichen ist, von keinem anderen User genutzt wird und nur aus Buchstaben und Zeichen besteht.
- Admin kann ein Passwort für den neuen User anlegen, solange dieses länger als 5 Zeichen ist.
- Admin kann dem neuen User Admin-Rechte erteilen.

Testfälle:

1. **Vorbedingung:** User ist eingeloggt mit Admin-Rechte.
Aktion: User gibt **weder** Name, Benutzername noch Passwort ein und drückt auf Speichern.
Ergebnis: User bekommt Benachrichtigung, dass er Name, Benutzername und Passwort eingeben muss.

2. **Vorbedingung:** User ist eingeloggt mit Admin-Rechte und ein Spieler hat bereits den Username „Test1“.

Aktion: User gibt „Test1“ als Benutzername, „Test“ als Name und „12345“ als Passwort ein und drückt auf Speichern.

Ergebnis: User bekommt Benachrichtigung, dass dieser Benutzername bereits vergeben ist.

3. **Vorbedingung:** User ist eingeloggt mit Admin-Rechte und **kein** Spieler hat den Username „Test1“.

Aktion: User gibt „Test1“ als Benutzername ein und drückt auf Speichern.

Ergebnis: User bekommt Benachrichtigung, dass ein neuer Spieler angelegt wurde.

Userstory: Spieler bearbeiten

Ziel dieser Userstory ist es, dass ein Admin bestehende Benutzerkonten ändern kann, damit er den Spielern bei Problemen (z.B. Passwort vergessen) helfen kann.

Akzeptanzkriterium:

- User kann Namen des Spielers ändern, solange zwischen 3 und 25 Zeichen ist.
- User kann Benutzernamen eines Spielers ändern, solange dieser zwischen 3 und 20 Zeichen ist, von keinem anderen User genutzt wird und nur aus Zahlen und Buchstaben besteht.
- User kann Passwort eines Spielers ändern, solange dieses länger als 5 Zeichen ist.
- User kann die Positionen des Spielers ändern.

Testfälle:

Siehe Benutzerdaten ändern (**Vorbedingung:** User hat Admin-Rechte).

Userstory: Spieler löschen

Ziel dieser Userstory ist es, dass ein Admin Benutzerkonten, welche nicht mehr benutzt werden, löschen kann, damit alten Spielern der Zugriff auf die App verweigert wird.

Akzeptanzkriterium:

- User kann die Benutzerkonten von Spielern löschen.
- User kann sein Benutzerkonto nicht löschen.

Testfälle:

1. **Vorbedingung:** User ist eingeloggt und hat Admin-Rechte.

Aktion: User wählt bei einem Spieler die Option löschen aus.

Ergebnis: Spieler wird gelöscht (Spieldaten bleiben erhalten).

2. **Vorbedingung:** User ist eingeloggt und hat Admin-Rechte.

Aktion: User wählt bei sich selbst die Option löschen aus.

Ergebnis: User wird benachrichtigt, dass er sich selbst nicht löschen kann.

Userstory: Spiel anlegen

Ziel dieser Userstory ist es, dass ein Admin Spiele anlegen kann, damit Spiele festgehalten werden können und deren Ergebnisse.

Akzeptanzkriterium:

Admin kann Spiele anlegen, Spieler dem Spiel zuweisen bzw. in Teams und das Ergebnis eintragen.

Testfälle:

1. **Vorbedingung:** User ist eingeloggt und hat Admin-Rechte.

Aktion: User weist keine Spieler dem Spiel zu.

Ergebnis: User bekommt Benachrichtigung, dass er die Mindestanzahl an Spielern nicht erfüllt um ein Spiel zu erstellen.

2. **Vorbedingung:** User ist eingeloggt und hat Admin-Rechte und hat Spieler dem Spiel zugewiesen.

Aktion: User weist einem Spieler kein Team zu.

Ergebnis: User bekommt Benachrichtigung, dass jeder Spieler einem Team zugewiesen sein muss.

3. **Vorbedingung:** User ist eingeloggt und hat Admin-Rechte und hat jeden Spieler ein Team zugewiesen.

Aktion: User trägt Ergebnis des Spiels ein.

Ergebnis: User bekommt Benachrichtigung, dass die Daten gespeichert wurden.

Userstory: Spiel ändern

Ziel dieser Userstory ist es, dass ein Admin Spiele im Nachhinein bearbeiten kann, damit er mögliche Fehler korrigieren kann.

Akzeptanzkriterium:

Admin kann das Ergebnis von Spielen ändern.

Testfälle:

1. **Vorbedingung:** User ist eingeloggt und hat Admin-Rechte und es gibt mindestens ein Spiel.

Aktion: User wählt Spiel und ändert das Ergebnis.

Ergebnis: User bekommt Benachrichtigung, dass die Änderung gespeichert wurde.

Userstory: Spiel löschen

Ziel dieser Userstory ist es, dass ein Admin Spiele löschen kann, damit möglich falsch erstellte Spiele entfernt werden können.

Akzeptanzkriterium:

Admin kann Spiele löschen.

Testfälle:

1. **Vorbedingung:** User ist eingeloggt und hat Admin-Rechte und es gibt mindestens ein Spiel.

Aktion: User wählt Spiel und löscht es.

Ergebnis: User bekommt Benachrichtigung, dass das Spiel gelöscht wurde.

Supporting Stories

Story: Einstellungen

Ziel dieser Story ist es den Benutzern Einstellungsmöglichkeiten zur Verfügung stellen. Dazu gehören:

- Sprache
- Benachrichtigungsart
- Layoutauswahl

Story: Design

Ziel dieser Story ist es ein Einheitliches Design für die App zu erstellen.

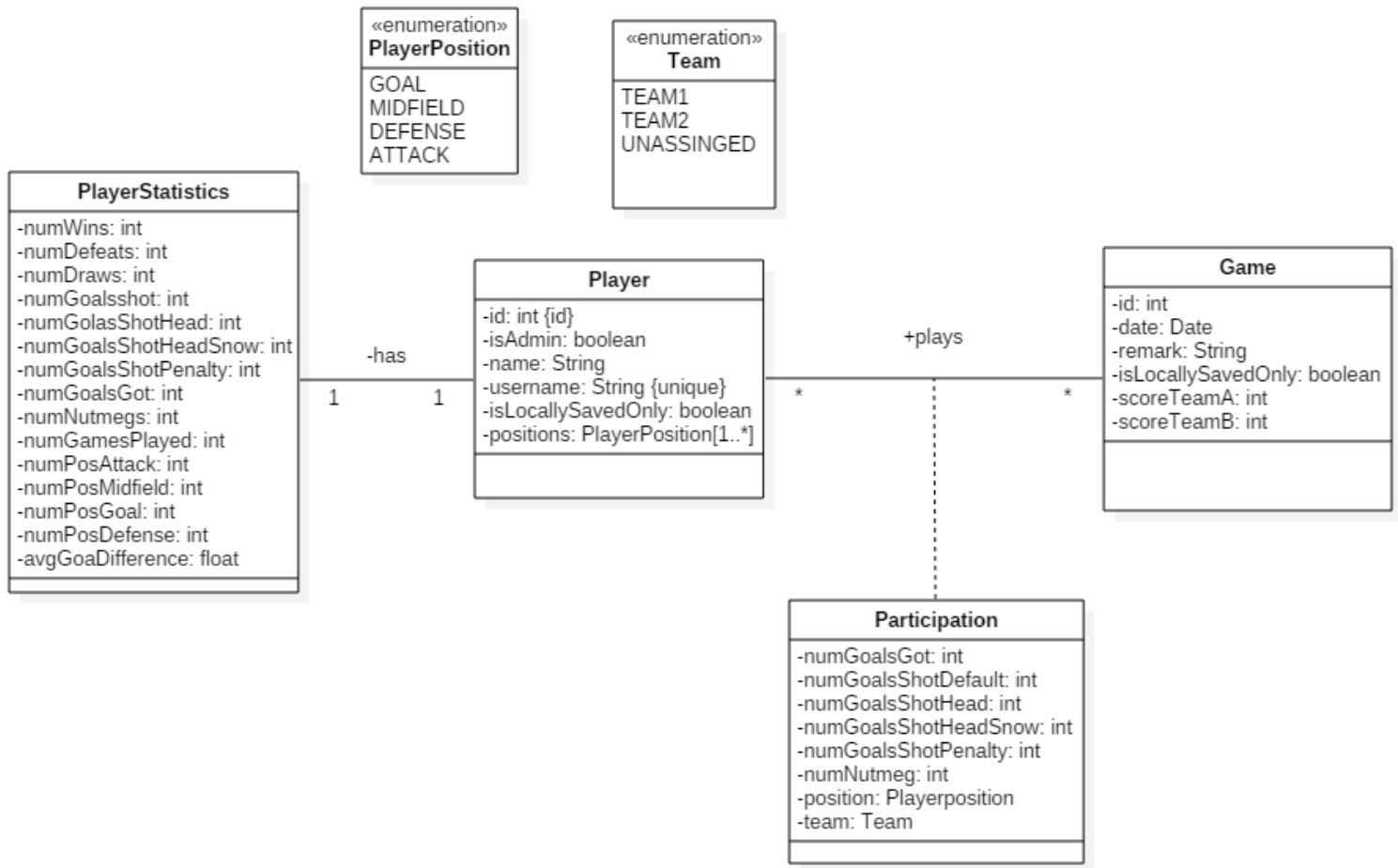
Story: Menü

Ziel dieser Story ist es das die App über ein Menü verfügt, welches Benutzerfreundlich und Übersichtlich ist.

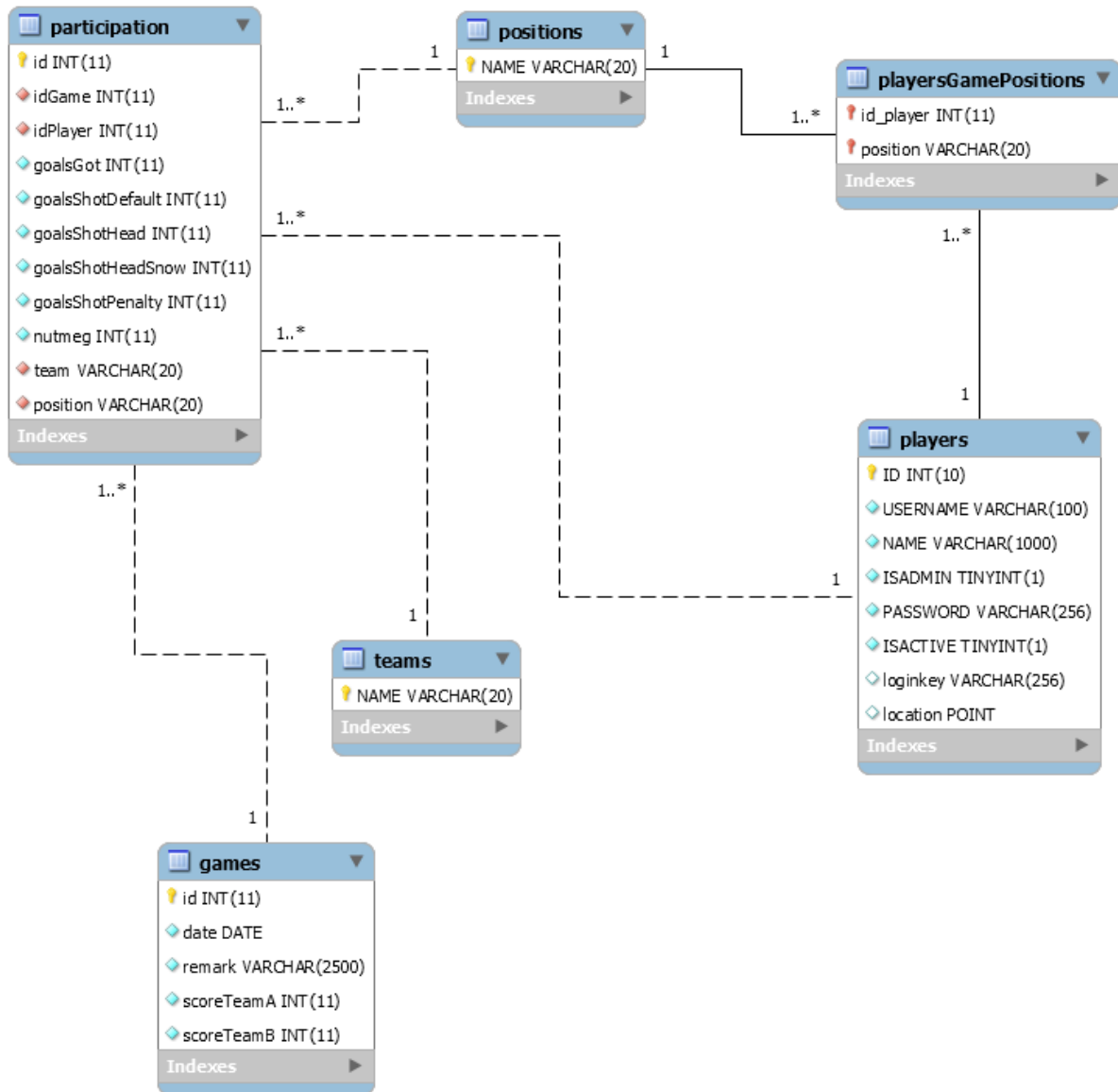
Story: Offline

Ziel dieser Story ist es das die App auch mit einer inaktiven Internetverbindung funktioniert, ohne dabei wichtige Funktionsfähigkeiten zu verlieren.

Klassendiagramm



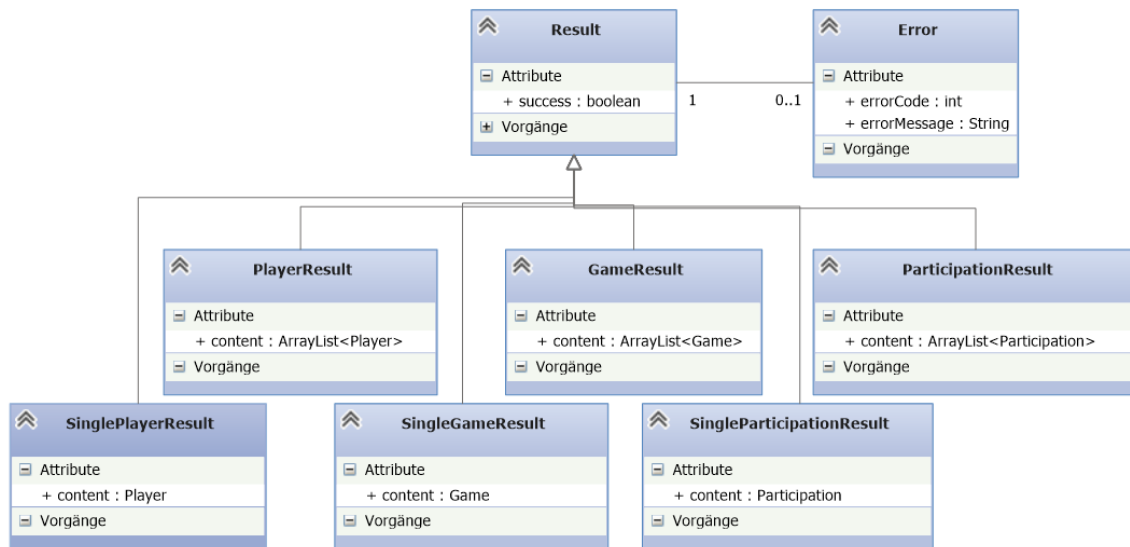
ER-Diagramm



Webservice

General

Data Model



Class	Description
Error	Represents an error. Error Code is always 0, because there are no pre – defined errors in this API.
Result	Information to the result of a web service call. If it fails and success equals false, the error object shows the reason for the fail. If success equals true, there should be content in the response. The only reason for a succeeded call with an empty content object is that there wasn't a single record matching the requests filter option.

Es wird nur JSON supported.

Methods

GameResource

getAllGames

Request

Method	URL
GET	url/game

Param	Type	Description	Example Value
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	<pre>{ "type": "gameResult", "success": true, "content": [{ "date": "2017-03-16", "id": 1, "remark": "Schneefall", "scoreTeamA": 56, "scoreTeamB": 0 }, { "date": "2017-03-15 ", "id": 2, "remark": "Peter verletzt", "scoreTeamA": 0, "scoreTeamB": 0 }, { "date": "2017-03-14 ", "id": 3, "remark": "", "scoreTeamA": 0, "scoreTeamB": 0 }] }</pre>
401	{"type": "gameResult", "success": false}
500	{"error": "Something went wrong. Please try again later."}

updateGame

Request

Method	URL
PUT	url/game

Param	Type	Description	Example Value
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre>{ "date": "2017-03-16 ", "id": 1, "remark": "Schneefall", "scoreTeamA": 56, "scoreTeamB": 0 },</pre>

Response

Status	Response
202	{"success": true}
202	{"success": false}
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Updaten erfolgreich war, oder nicht.

getGamesByDate

Request

Method	URL
GET	url/game/byDate/{date}

Param	Type	Description	Example Value
date	Path	Date for game	2017-03-16
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	{ "type": "gameResult", "success": true, "content": [{ "date": "2017-03-16", "id": 1, "remark": "Schneefall", "scoreTeamA": 56, "scoreTeamB": 0 }] }
202	{ "type": "gameResult", "error": { "errorCode": 0, "errorMessage": "Unparseable date: \"fds\"" }, "success": false }
401	{"type": "gameResult", "success": false}
500	{"error": "Something went wrong. Please try again later."}

getGamesByPlayerId (and between 2 Dates)

Request

Method	URL
GET	url/game/byPlayerId/{id}

Param	Type	Description	Example Value
id	Path	Mandatory; Id for Player	1
dateFrom	Query	Optional; Start Date	2017-03-15
dateTo	Query	Optional; End Date	2017-03-18
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

If no Date is specified, it will be ignored

Response

Status	Response
202	<pre>{ "type": "gameResult", "success": true, "content": [{ "date": "2017-03-16T00:00:00+01:00", "id": 1, "scoreTeamA": 56, "scoreTeamB": 0 }] }</pre>
401	<pre>{"type": "gameResult", "success": false}</pre>
500	<pre>{"error": "Something went wrong. Please try again later."}</pre>

insertGame

Request

Method	URL
POST	url/game

Param	Type	Description	Example Value
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre>{ "date": "2017-03-16", "scoreTeamA": 56, "scoreTeamB": 0, "remark": 20 }</pre>

Java.util.Date in Game was changed to Java.sql.Date, otherwise this wouldn't be possible

Response

Status	Response
202	<pre>{ "type": "singleGameResult", "success": true, "content": { "date": "2017-03-16", "id": 7, "remark": "Schnee", "scoreTeamA": 56, "scoreTeamB": 0 } }</pre>
401	{"type": "gameResult", "success": false}
500	{"error": "Something went wrong. Please try again later."}

deleteGame

Request

Method	URL
DELETE	url/game/{id}

Param	Type	Description	Example Value
id	Path	Id of Game that has to be deleted	1
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	{success: true}
202	{success: false}
401	{success: false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Löschen erfolgreich war, oder nicht.

PlayerResource

getAllPlayers

Request

Method	URL
GET	url/player

Param	Type	Description	Example Value
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	<pre>{ "type": "playerResult", "success": true, "content": [{ "admin": true, "id": 1, "name": "Admin\uD83E\uDD18", "positions": ["ATTACK", "GOAL", "MIDFIELD"], "statistics": { "avgGoalDifference": 0, "numDefeats": 0, "numDraws": 1, "numGamesPlayed": 1, "numGoalsGot": 0, "numGoalsShot": 0, "numGoalsShotHead": 0, "numGoalsShotHeadSnow": 0, "numGoalsShotPenalty": 0, "numNutmeg": 0, "numPosAttack": 0, "numPosDefense": 0, "numPosGoal": 1, "numPosMidfield": 0, "numWins": 0 }, "username": "admin" }, { "admin": true, "id": 2, "name": "\u2584\uFE3B\u0337\u033F\u253B\u033F\u2550\u2501\u4E00 Moser", "positions": [</pre>

	<pre> "ATTACK", "MIDFIELD"], "statistics": { "avgGoalDifference": -0.5, "numDefeats": 1, "numDraws": 1, "numGamesPlayed": 2, "numGoalsGot": 0, "numGoalsShot": 0, "numGoalsShotHead": 1, "numGoalsShotHeadSnow": 0, "numGoalsShotPenalty": 0, "numNutmeg": 0, "numPosAttack": 0, "numPosDefense": 0, "numPosGoal": 0, "numPosMidfield": 2, "numWins": 0 }, "username": "moe" } } </pre>
401	{"type": "playerResult", "success": false}
500	{"error": "Something went wrong. Please try again later."}

updatePlayer

Request

Method	URL
PUT	url/player

Param	Type	Description	Example Value
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre> { "admin": false, "id": 1, "name": "Martinii", "username": "martin" } </pre>

Response

Status	Response
202	{"success": true}
202	{"success": false}
401	{"type": "singlePlayerResult", "success": false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Updaten erfolgreich war, oder nicht.

setPositions

Request

Method	URL
PUT	url/player/positions/{playerid}

Param	Type	Description	Example Value
playerid	Query	Id of player to set positions	5
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre>{ "GOAL": false, "MIDFIELD": true, "ATTACK": false, "DEFENSE": true }</pre>

Response

Status	Response
202	{"success": true}
202	{"success": false}
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Updaten erfolgreich war, oder nicht.

getPlayerByUsername

Request

Method	URL
GET	url/player/{username}

Param	Type	Description	Example Value
username	Path	Username of the player you want to get	admin
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	<pre>{ "type": "singlePlayerResult", }</pre>

	<pre> "success": true, "content": { "admin": true, "id": 1, "name": "Admin\uD83E\uDD18", "positions": ["ATTACK", "GOAL", "MIDFIELD"], "statistics": { "avgGoalDifference": 0, "numDefeats": 0, "numDraws": 1, "numGamesPlayed": 1, "numGoalsGot": 0, "numGoalsShot": 0, "numGoalsShotHead": 0, "numGoalsShotHeadSnow": 0, "numGoalsShotPenalty": 0, "numNutmeg": 0, "numPosAttack": 0, "numPosDefense": 0, "numPosGoal": 1, "numPosMidfield": 0, "numWins": 0 }, "username": "admin" } </pre>
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

insertPlayer

Request

Method	URL
POST	url/game

Param	Type	Description	Example Value
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre>{ "admin": false, "name": "jerome", "username": "guina" }</pre>

Response

Status	Response
202	<pre>{ "type": "singlePlayerResult", "success": true, "content": { "admin": false, "goalDifference": 0, "id": 11, "name": "jerome", "numDefeats": 0, "numDraws": 0, "numWins": 0, "username": "guina" } }</pre>
202	<pre>{ "type": "singlePlayerResult", "error": { "errorCode": 0, "errorMessage": "com.mysql.jdbc.exceptions.jdbc4.MySQLIntegrityConstraintViolationEx ception: Duplicate entry 'guina' for key 'USERNAME'" }, "success": false, "content": { "admin": false, "goalDifference": 0, "id": 11, "name": "jerome", "numDefeats": 0, "numDraws": 0, "numWins": 0, "username": "guina" } }</pre>

	} }
401	{"type": "singlePlayerResult", "success": false}
500	{"error": "Something went wrong. Please try again later."}

The second case (MySQLIntegrityConstraintViolationException) happens when the username is not available anymore because it's already used by another player. In this case the existing player is returned.

deletePlayer

Request

Method	URL
DELETE	url/player/{id}

Param	Type	Description	Example Value
id	Path	Id of Player that has to be deleted	1
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	{success: true}
202	{success: false}
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Löschen erfolgreich war, oder nicht.

login

Method	URL
POST	url/player/security/login

MediaType	Sample Request Body
Application/json	<pre>{ "username": "admin", "password_enc": "21232f297a57a5a743894a0e4a801fc3" }</pre>

Response

Status	Response
202	a6444a246547facadfb56ed7a940b7f
204	
500	{"error": "Something went wrong. Please try again later."}

If LoginCredentials are valid, return a loginKey which is used for every further method call.
 If they are invalid, nothing (NO_CONTENT) is returned.

setPassword

Method	URL
PUT	url/player/security/{id}

Param	Type	Description	Example Value
id	Path	Id of the player you want to insert the password to	1
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre>{ "password_enc": "21232f297a57a5a743894a0e4a801fc3" }</pre>

Response

Status	Response
202	{"success": true}
202	{"success": false}
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Updaten erfolgreich war, oder nicht.

setGeoLocation

Method	URL
POST	url/player/geoloc/{id}

Param	Type	Description	Example Value
id	Path	Id of the player you want to set the geolocation	1
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre>{ "latitude": 46.601166, "longitude": 13.843841 }</pre>

Response

Status	Response
202	{"success": true}
202	{"success": false}
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

getGeoLocation

Method	URL
GET	url/player/geoloc/{id}

Param	Type	Description	Example Value
id	Path	Id of the player you want to set the geolocation	1
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	<pre>{ "type": "locationResult", "success": true, "content": { "latitude": 46.617955, "longitude": 13.848626 } }</pre>

	}
	}
202	{"type": "locationResult", "success": false}
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

getPlayersNearby

Method	URL
GET	url/player/geoloc/nearby/{id}

Param	Type	Description	Example Value
id	Path	Id of the player you want to set the geolocation	1
radius	Query	Radius (in km) which is considered "nearby"	2.5
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	{ "type": "playerIDResult", "success": true, "content": [1, 2, 3, 4] }
202	{ "type": "playerIDResult", "error": { "errorCode": 0, "errorMessage": "radius not set" }, "success": false }
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

ParticipationResource

getAllParticipations

Request

Method	URL
GET	url/participation

Param	Type	Description	Example Value
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	<pre>{ "type": "participationResult", "success": true, "content": [{ "numGoalsGot": 0, "numGoalsShotDefault": 0, "numGoalsShotHead": 0, "numGoalsShotHeadSnow": 0, "numGoalsShotPenalty": 0, "numNutmeg": 0, "player": { "admin": false, "goalDifference": 56, "id": 1, "name": "Martinii", "numDefeats": 0, "numDraws": 0, "numWins": 1, "username": "martin" }, "position": "ATTACK", "team": "TEAM1" },] }</pre>
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

getParticipationById

Request

Method	URL
GET	url/participation/{id}

Param	Type	Description	Example Value
id	Path	Id of the participation you want to get	5
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	<pre>{ "type": "participationResult", "success": true, "content": [{ "numGoalsGot": 0, "numGoalsShotDefault": 0, "numGoalsShotHead": 0, "numGoalsShotHeadSnow": 0, "numGoalsShotPenalty": 0, "numNutmeg": 0, "player": { "admin": false, "goalDifference": 56, "id": 1, "name": "Martinii", "numDefeats": 0, "numDraws": 0, "numWins": 1, "username": "martin" }, "position": "ATTACK", "team": "TEAM1" }] }</pre>
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

getParticipationById

Request

Method	URL
GET	url/participation/byGame/{id}

Param	Type	Description	Example Value
id	Path	Id of the game you want the participations for	5
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	<pre>{ "type": "participationResult", "success": true, "content": [{ "numGoalsGot": 0, "numGoalsShotDefault": 0, "numGoalsShotHead": 0, "numGoalsShotHeadSnow": 0, "numGoalsShotPenalty": 0, "numNutmeg": 0, "player": { "admin": false, "goalDifference": 56, "id": 1, "name": "Martinii", "numDefeats": 0, "numDraws": 0, "numWins": 1, "username": "martin" }, "position": "ATTACK", "team": "TEAM1" }] }</pre>
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

insertParticipation

Method	URL
POST	url/participation

Param	Type	Description	Example Value
idGame	Query	Id of the game	1
idPlayer	Query	Id of the player	2
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre>{ "numGoalsGot": 0, "numGoalsShotDefault": 0, "numGoalsShotHead": 0, "numGoalsShotHeadSnow": 0, "numGoalsShotPenalty": 0, "numNutmeg": 0, "position": "ATTACK", "team": "TEAM1" }</pre>

Response

Status	Response
202	{"success": true}
202	{"success": false}
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Erstellen erfolgreich war, oder nicht. Es ist möglich, dass bei success false diverse SQL Errors mitgeschickt werden (z.B. Constraint verletzt o.Ä.)

updateParticipation

Request

Method	URL
PUT	url/participation

Param	Type	Description	Example Value
idGame	Query	Id of the game	1
idPlayer	Query	Id of the player	2
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

MediaType	Sample Request Body
Application/json	<pre>{ "numGoalsGot": 0, "numGoalsShotDefault": 0, "numGoalsShotHead": 0, "numGoalsShotHeadSnow": 0, "numGoalsShotPenalty": 0, "numNutmeg": 0, "position": "ATTACK", "team": "TEAM1" }</pre>

Response

Status	Response
202	{"success": true}
202	{"success": false}
401	{"success": false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Löschen erfolgreich war, oder nicht.

deleteParticipation

Request

Method	URL
DELETE	url/participation/{id}

Param	Type	Description	Example Value
id	Path	Id of Participation that has to be deleted	1
loginKey	Query	Authentication Key	098f6bcd4621d373cade4e832627b4f6

Response

Status	Response
202	{success: true}
202	{success: false}
500	{"error": "Something went wrong. Please try again later."}

Success true oder false je nachdem ob das Löschen erfolgreich war, oder nicht.

Architektur – Programmierung

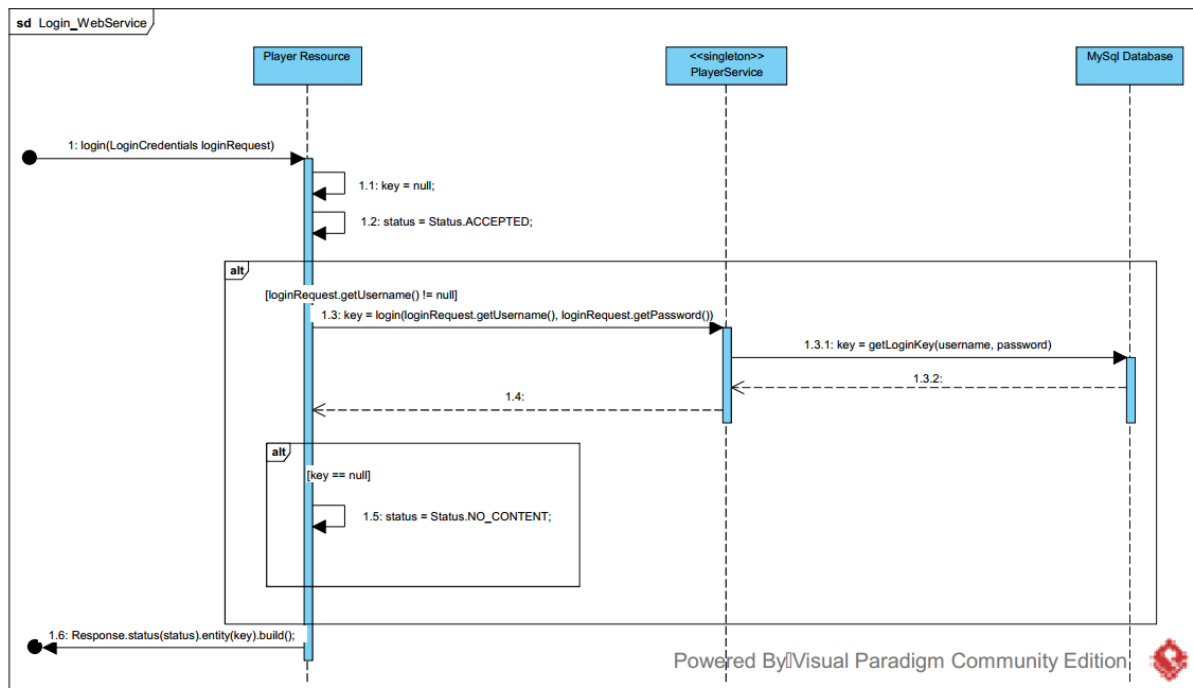
QR-Code Generator: <https://github.com/zxing/zxing>

QR-Code Scanner: <https://developers.google.com/vision/>

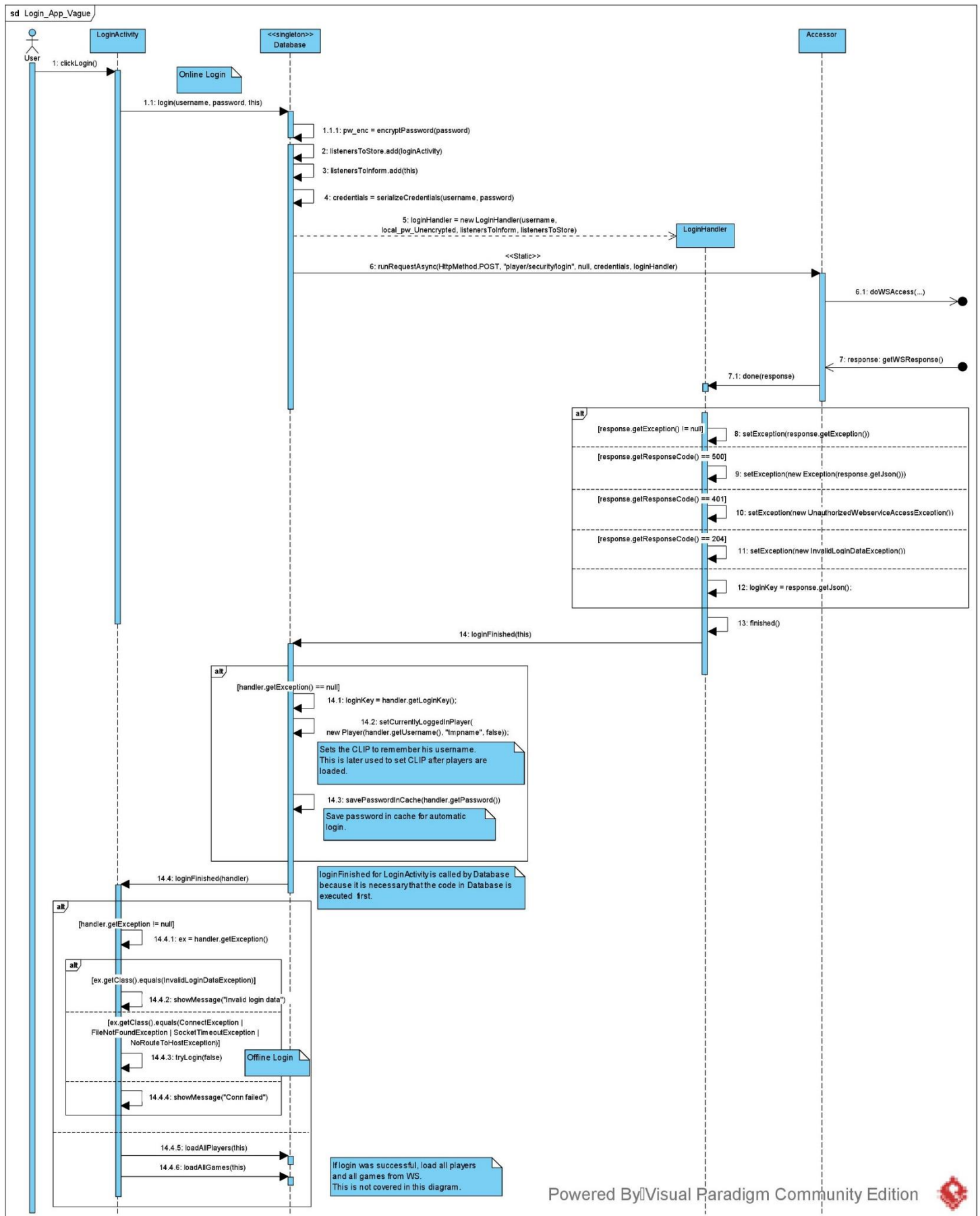
GSON: <https://github.com/google/gson>

Apache Common Lang: <https://commons.apache.org/proper/commons-lang/>

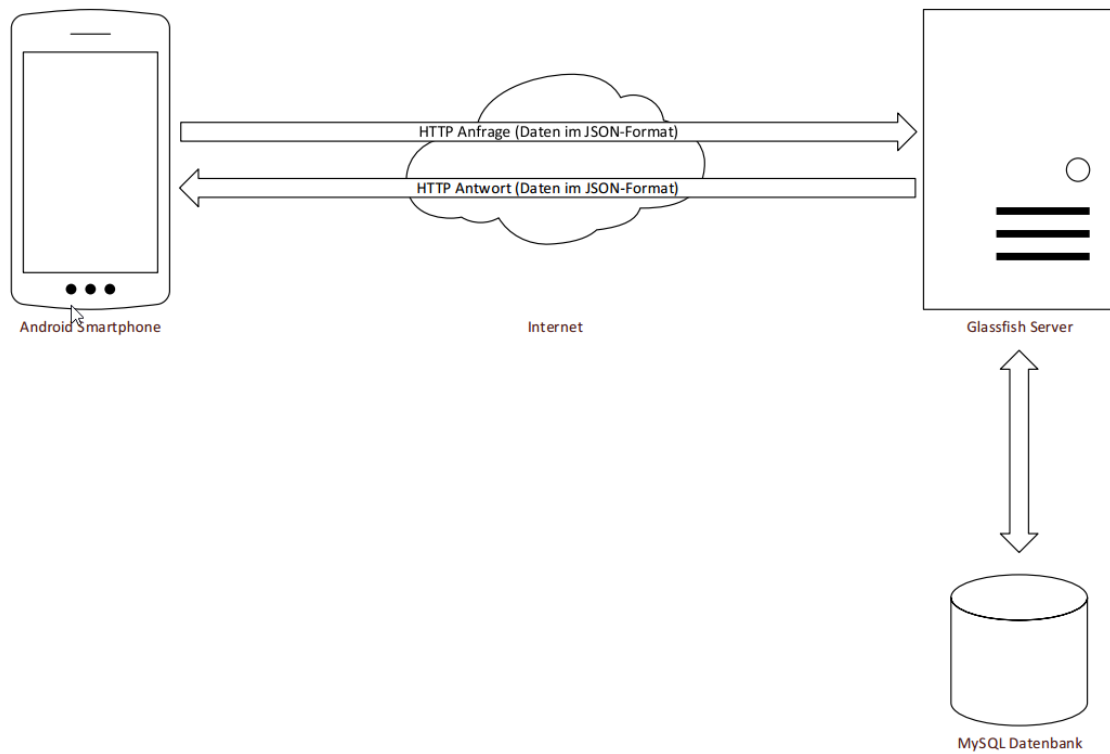
Login Webservice



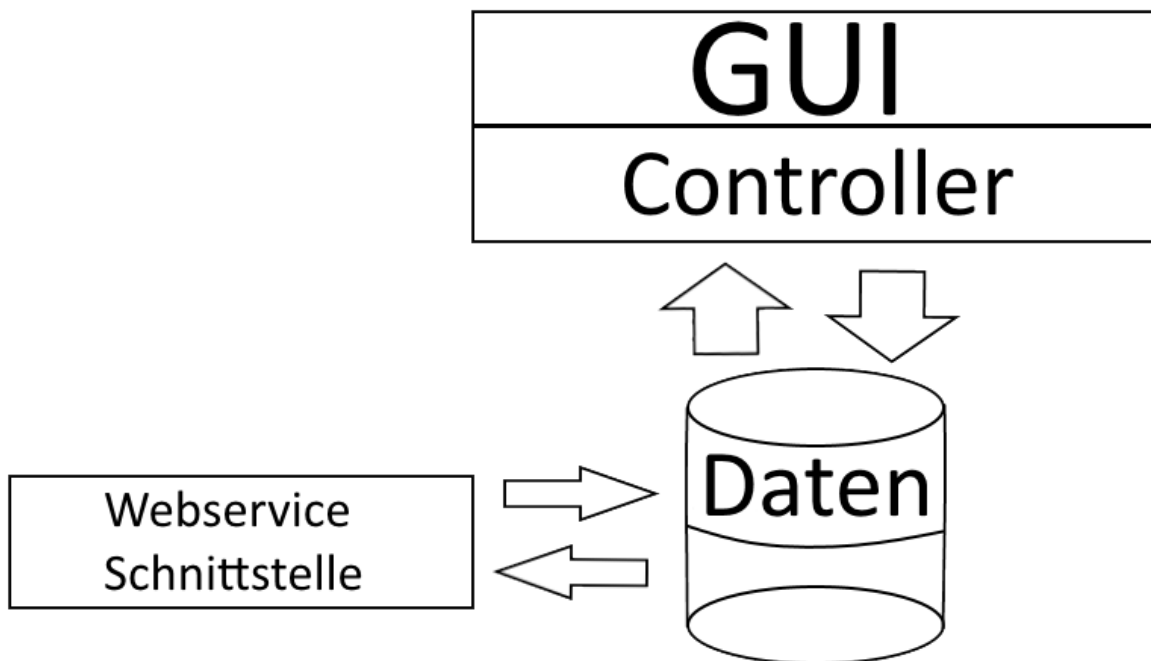
Login App



Kommunikation (App und Webservice)



Kommunikation in der App



Arbeitsaufteilung

Santner:

- Datenbank
- Spiel löschen
- Spiel ändern
- Spiel anlegen
- Spieler löschen
- Spieler anpassen
- Spieler Statistiken
- Offline Modus
- Webservice
- Klassendiagramme
- Storymap
- Tester

Wilscher:

- Webservice Schnittstelle
- Einstellungen
- Datenbank
- Datenbank(WebService)
- Klassendiagramme
- Impressum
- Tester

Moser:

- Spiel anlegen
- Spieler Statistiken (Scoreboard)
- Menü [Verantwortlich]
- Einstellungen
- Dokumentation
- Storymap
- Klassendiagramme
- Tester

Sunny:

- Layout
- Sicherheitsabfragen
- Spieler Auflistung
- Tester
- Spieler anlegen

Webservice:

- MAX HAIDER
- LUKAS KIRCHBAUMER
- DAVID WUCHERER

User-Guide Moskitos SoccerApp

Einleitung

Die App dient der Verwaltung und Analyse von Fußballspielen, die von dem Fußballclub Moskitos gespielt werden. Die in diesem Dokument enthaltenen Funktionen stehen den Usern zur Verfügung. Einige Funktionen können nur Admins nutzen, für diese sehen Sie bitte den Admin-Guide ein.

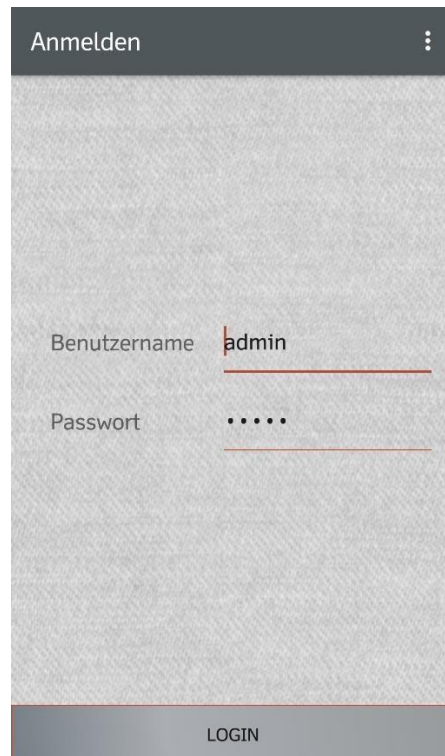
Zum erstmaligen Starten wird eine Internetverbindung benötigt.

Offline-Modus

Hier können nur die Daten eingesehen werden, die bei der letzten Ausführung mit Internetverbindung auf dem Gerät gespeichert wurden.

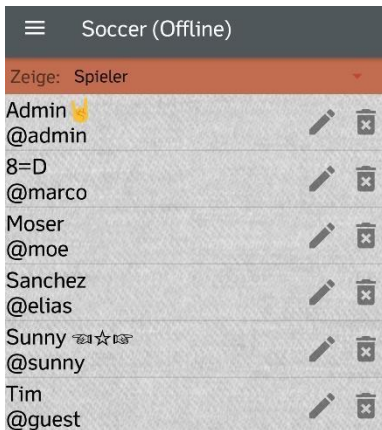
Login

Geben Sie hier Ihre Nutzerdaten ein. Sollten Sie noch keine Zugangsdaten erhalten haben, melden Sie sich bei einem Admin. Nach dem erstmaligen Anmelden wird keine Internetverbindung mehr benötigt, da nun eine Anmeldung im Offline-Modus möglich ist (bietet nur eingeschränkte Nutzung der App). Haben Sie Ihr Passwort vergessen, so melden Sie sich bitte bei einem Admin.



Hauptansicht

Hier wird eine Übersicht aller Spieler angezeigt.



Soccer (Offline)		
Zeige: Spieler		
Admin 🏆	@admin	
8=D	@marco	
Moser	@moe	
Sanchez	@elias	
Sunny 🌟🌟🌟	@sunny	
Tim	@guest	

Durch eine einfache Änderung der Zeigeoption kann hier zwischen den Spielern und Spielen gewechselt werden.



Durch Klick auf einen Spieler, wird dessen Statistik angezeigt.

Spieler Statistik	
Name	8=D @marco
Spielanzahl	2
Siege	1
Niederlagen	0
Unentschieden	1
Durchsch. Tordifferenz	0.5
Tore geschossen	2
Tore geschossen (Fuß)	0
Kopfballtore	2
Kopfballtore bei Schnee	0

Eigene Benutzerdaten bearbeiten

Hier können Sie jederzeit ihre eigenen Daten bearbeiten und Ihre Positionen setzen die Sie spielen wollen. In der Teamerstellung werden Sie nur für die von Ihnen eingetragenen Positionen berücksichtigt.

Benutzerdaten	
Name	Admin 🏆
Benutzername	admin
<input type="checkbox"/> Passwort erneuern	
Positionen	
<input checked="" type="checkbox"/> Sturm	
<input checked="" type="checkbox"/> Mittelfeld	
<input type="checkbox"/> Verteidigung	
<input checked="" type="checkbox"/> Torwart	
<div> <div>ABBRUCH</div> <div>SPEICHERN</div> </div>	

Bestenliste

Hier wird eine Liste aller Spieler angezeigt, diese sind sortiert nach Ihrer Leistung in der ausgewählten Kategorie.

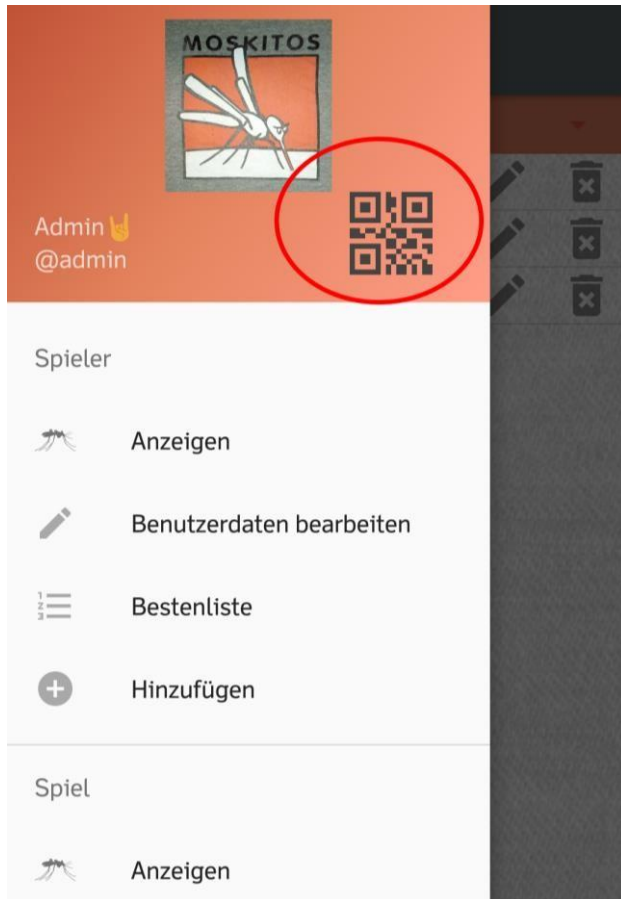
Bestenliste: Siege	
Zeige: Siege ▼	
8=D @marco	1
Sunny 🏆⭐🏆 @sunny	1
Admin 🙌 @admin	0
Moser @moe	0
Sanchez @elias	0
Tim @guest	0

Durch Klick auf einen Spieler, wird dessen Statistik angezeigt. Mithilfe der Zeigeoption, kann hier zwischen den einzelnen Kategorien gewechselt werden.



QR-Code anzeigen

Mit einem Klick auf den QR-Code im Menü, wird Ihnen ein QR-Code angezeigt. Mit diesem kann der Spielersteller schneller und komfortabler ein Spiel erstellen. Hierfür müssen Sie nur den erscheinenden Code dem Ersteller zeigen.



Einstellungen

Sprache wechseln

Hier kann zwischen Deutsch, Englisch sowie Spanisch gewechselt werden, die Sprache kann jederzeit angepasst werden.

Nachrichtenart Überblendung

Mit dieser Einstellung kann die Anzeige der Benachrichtigungen für Fehler und Warnungen geändert werden.

Impressum

In diesem Menüpunkt finden Sie die Ersteller dieser App, sowie verwendete Bibliotheken und 3rd-Party Anwendungen.

Admin-Guide Moskitos SoccerApp

Einleitung

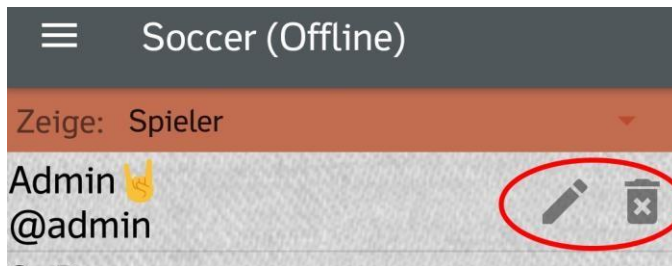
In diesem Dokument werden explizit die Zusatzfunktionen, die ein Admin erhält, näher ausgeführt. Sollten Sie die App als erster von Ihrem Verein in Betrieb nehmen, geben Sie für Nutzernamen, sowie für das Passwort „admin“ ein. Danach können Sie nach Belieben einen anderen Nutzer mit Admin-Rechten anlegen.

Offline-Modus

In diesem Modus kann man neue Spieler und neue Spiele hinzufügen. Diese können auch bearbeitet und gelöscht werden. Außerdem kann man die Daten einsehen, die bei der letzten Ausführung mit Internetverbindung auf dem Gerät gespeichert wurden.

Hauptansicht

In der Hauptansicht hat der Admin die Möglichkeit über einfaches klicken der Symbole, den betreffenden Spieler zu bearbeiten bzw. zu löschen.



Benutzerdaten bearbeiten

Hier können Anzeigename, Benutzername, Passwort und Wunschposition(en) geändert werden. Als Admin hat man die Möglichkeit die Daten jedes Users zu ändern oder zu löschen. Sie sind somit nicht nur auf Ihre eigenen Daten beschränkt. Außerdem können Sie einem normalen User jederzeit Adminrechte geben.



Spieler hinzufügen

Hier wird ein neuer Spieler mit den eingetragenen Daten hinzugefügt. Beim Anlegen können ihm bereits Admin-Rechte gegeben werden.

Spiel hinzufügen

Spielerauswahl

Das Datum und die teilnehmenden Spieler für das Spiel werden hier ausgewählt, dies kann entweder manuell oder mithilfe eines QR-Code Scanners erfolgen.

Spielerauswahl

06
Mai
2016

Datum

07
Juni
2017

08
Juli
2018

Wähle Spieler fürs Spiel aus

☒

Teilnehmer

<input checked="" type="checkbox"/>	8=D @marco
<input checked="" type="checkbox"/>	Admin 🙌 @admin
<input checked="" type="checkbox"/>	Moser @moe
<input checked="" type="checkbox"/>	Sanchez @elias
<input checked="" type="checkbox"/>	Sunny 🌞⭐🌈 @sunny
<input checked="" type="checkbox"/>	Tim

ABBRUCH

QRCODE

WEITER

Teamzuweisung

Hier wird eine Liste aller teilnehmenden Spieler angezeigt. Durch wechseln der Reiter kann entschieden werden, ob ein Spieler „Team 1“ oder „Team 2“ zugewiesen wird. Mit dem grünen Plus kann ein Spieler in das aktuell ausgewählte Team eingefügt werden. Die Position am Spielfeld kann aus den Wunschpositionen des Spielers gewählt werden. Durch Shuffle ist es möglich diesen Vorgang zu automatisieren, wobei danach die Korrektheit manuell überprüft werden sollte.

Team Division

TEAM1 → TEAM2

Name	Position	
Moser	Midfield	+
Sanchez	Midfield	+
Marco	Midfield	+
Sunny 🌟	Attacker	+
Admin 🙌	Midfield	-

Position dropdown menu:

- Midfield
- Attacker
- Goalkeeper

CANCEL SHUFFLE CONTINUE

Ergebniseingabe

Diese erfolgt über eine Tabelle. Über die Reiter kann zwischen „Team 1“ und „Team 2“ gewechselt werden. Durch Klick auf ein Icon (Bild) wird dessen Bedeutung angezeigt. Zusätzlich kann noch eine Anmerkung zum Spiel festgehalten werden.

Enter Data

TEAM1 → TEAM2

0:0

Remark

Name	P	🏠	👤	👤	👤	👤	👤
Admin 🙌	M	0	0	0	0	0	0
Moser	A	0	0	0	0	0	0
Sunny 🌟	G	0	0	0	0	0	0

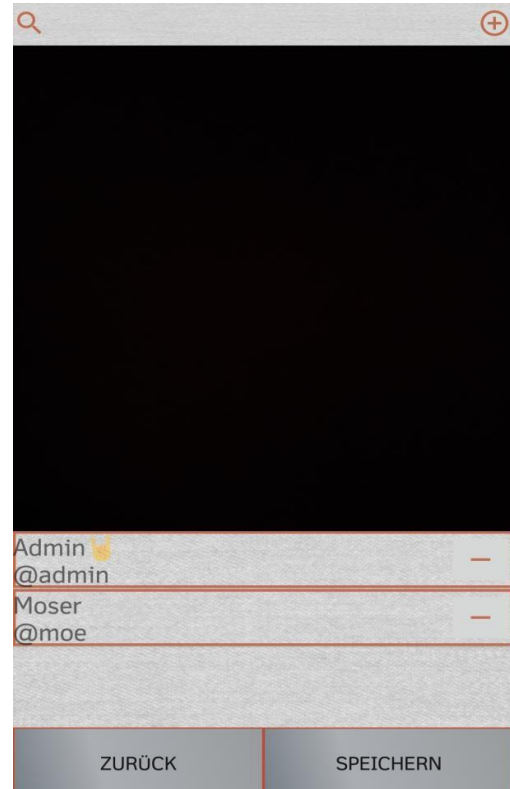
BACK SAVE

Spiel bearbeiten/löschen

Selbstverständlich können Sie im Nachhinein ein bereits erstelltes Spiel bearbeiten. Hierfür müssen Sie lediglich die Übersicht aufrufen und unter „Zeige“ Spiel auswählen. Mit einem Klick auf das dafür vorgesehene Icon können Sie die gewünschte Funktion nutzen.

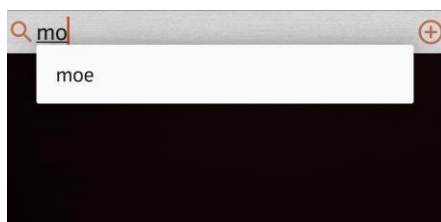
QR-Code Scanner

Aufrufbar in der Spielerauswahl (Spiel hinzufügen).



Die teilnehmenden Spieler müssen auf Ihrem Gerät in der App den QR-Code anzeigen, damit der Ersteller des Spieles, diese scannen kann.

Falls jemand sein Smartphone nicht zur Verfügung hat, kann dieser Spieler manuell über ein Textfeld mit Autovervollständigung hinzugefügt werden, dies kann wahlweise auch nachträglich in der Spielerauswahl erfolgen.



Einstellungen

Teamzuweisung

Mit dieser Einstellung erhalten Sie die Möglichkeit ihre präferierte Zuweisungs-Methode einzustellen.

Prozessfortschritt

1. Sprint

Zeitraum:		15.03. - 30.03.2017		Productowner: Sonnberger	
Ziele:					
Einarbeitung in Android					
Github einrichten					
Projekt erstellen					
Aufgabe ausarbeiten					
Webservice Anforderungen erstellen					
Review:					
Einarbeitung in Android		✓ <input type="checkbox"/>			
Github einrichten		✓ <input type="checkbox"/>			
Projekt erstellen		✓ <input type="checkbox"/>			
Aufgabe ausarbeiten		✓ <input type="checkbox"/>			
Webservice Anforderungen erstellen		✓ <input type="checkbox"/>			

2. Sprint

Zeitraum:	31.03. - 27.04.2017		Productowner:	Santner	
Ziele:					
Daten-Klassen erstellen					
Anmelden					
Daten ändern					
Menü erstellen					
Spieler anlegen					
Spiel anlegen					
Daily Scrums:					
Fr - 31.03.2017		Wilscher krank!			
03.04.2017 - 07.04.2017		Klassenfahrt nach Italien			
Mi - 19.04.2017		Sunny krank!; Menü erstellt			
Do - 20.04.2017		-			
Fr - 21.04.2017		Add Player implementiert			
Mo - 24.04.2017		Erfolgreich branch auf sprint2 umgestellt			
Di - 25.04.2017		Code bereinigt			
Mi - 26.04.2017		Clientseitiger Webservice-Zugriff teilweise fertig			
Do - 27.04.2017		Multi-language support (English/Deutsch)			
Review:					
Daten-Klassen erstellen		✓□			
Anmelden		✓□		Ohne Webservice	
Daten ändern		✓□		Ohne Webservice	
Menü erstellen		✓□			
Spieler anlegen		✓□		Ohne Webservice	
Spiel anlegen		✗□		Komplexer als vermutet	

3. Sprint	Zeitraum:		28.04. - 11.05.2017	Productowner:		Moser
	Ziele:					
	Spiel anlegen (von 2. Sprint)					
	Spiel bearbeiten					
	Spiel löschen					
	Shuffle für Spieler					
	Spieler bearbeiten					
	Spieler löschen					
	Umstellung auf Webservice					
	Daily Scrums:					
	Fr - 28.04.2017		Besprechung für Spielverwaltung, Sprint geplant			
	Mo - 01.05.2017		Frei!			
	Di - 02.05.2017		Besprechung Menü und Spielerstellungs GUIs			
	Mi - 03.05.2017		Neues Menü;			
	Do - 04.05.2017		Spieler für Spiel auswählen fertig			
	Fr - 05.05.2017		Daten für Spiel eintragen fertig			
	Mo - 08.05.2017		Layout überarbeitet			
	Di - 09.05.2017		Team Management fertig			
	Mi - 10.05.2017		Player verwalten fertig; Zusammenführung Spiel erstellen			
	Do - 11.05.2017		Sunny krank!; Shuffle funktionsfähig; Planung neuer Sprint			
	Review:					
	Spiel anlegen (von 2. Sprint)		✓ <input type="checkbox"/>			
	Spiel bearbeiten		✗ <input type="checkbox"/>		Zeitlich nicht mehr machbar	
	Spiel löschen		✗ <input type="checkbox"/>		Zeitlich nicht mehr machbar	
Shuffle für Spieler		✓ <input type="checkbox"/>				
Spieler bearbeiten		✓ <input type="checkbox"/>				
Spieler löschen		✓ <input type="checkbox"/>				
Umstellung auf Webservice		✓ <input type="checkbox"/>				

4. Sprint	Zeitraum:		12.05. - 24.05.2017	Productowner:		Moser
	Ziele:					
	Spiel bearbeiten (von 3. Sprint)					
	Spiel löschen (von 3. Sprint)					
	Spieler-Statistiken					
	Scoreboard					
	Rollen (User/Admin)					
	Präsentation vorbereiten					
	Daily Scrums:					
	Fr - 12.05.2017			Zuweisung der Aufgaben		
	Mo - 15.05.2017			Generelle Besprechung, Rollen implementiert, Layout verbessert		

Di - 16.05.2017	Lagebesprechung, Gameliste anzeigen	
Mi - 17.05.2017	Spiel bearbeiten & löschen fertig	
Do - 18.05.2017	Kleinere Bugfixes, Spieler-Statistiken	
Fr - 19.05.2017	Besprechung bzgl. Präsentation, Scoreboard fertig	
Mo - 22.05.2017	Layout für Tablets erstellt	
Di - 23.05.2017	Letzte Bugfixes für Präsentation	
Mi - 24.05.2017	Besprechung neu gewonnener Ideen aus Präsentationen	
Review:		
Spiel bearbeiten (von 3. Sprint)	✓ <input type="checkbox"/>	
Spiel löschen (von 3. Sprint)	✓ <input type="checkbox"/>	
Spieler-Statistiken	✓ <input type="checkbox"/>	
Scoreboard	✓ <input type="checkbox"/>	
Rollen (User/Admin)	✓ <input type="checkbox"/>	
Präsentation vorbereiten	✓ <input type="checkbox"/>	

5.Sprint

Zeitraum:		25.05. - 14.06.2017	Productowner:	Moser
Ziele:				
Dokumentation anfertigen				
Bug Fixes				
Zusatzfeatures (Anregungen von Präsentation)				
Daily Scrums:				
Do - 25.05.2017		Frei!		
Fr - 26.05.2017		Frei!		
Mo - 29.05.2017		Webservice überarbeitet, Asynchron, QR-Code Anzeige		
Di - 30.05.2017		Bestätigungsabfragen, 2. Teameinteilung GUI begonnen		
Mi - 31.05.2017		Unicodezeichen erlaubt		
Do - 01.06.2017		Doku begonnen		
Fr - 02.06.2017		Swipe GUI fertig		
Mo - 05.06.2017		Frei!		
Di - 06.06.2017		Frei!		
Mi - 07.06.2017		QR-Code Scanner fertig		
Do - 08.06.2017		-		
Fr - 09.06.2017		-		
Mo - 12.06.2017		Doku größtenteils fertig		
Di - 13.06.2017		Doku fertig		
Mi - 14.06.2017		Letzte Änderungen Doku		
Review:				
Dokumentation anfertigen	✓ <input type="checkbox"/>			
Bug Fixes	✓ <input type="checkbox"/>			
Zusatzfeatures (Anregungen von Präsentation)	✓ <input type="checkbox"/>			