Anforderungsdokument
„Basket Rolling Verwaltungssoftware“

Inhalt

1. Ziel der Software

Die Software dient zur Verwaltung eines Basketball-Vereins. Sie soll Informationen über Spieler, Mannschaften, Spiele, Trainings, Mitgliedsbeiträge und Statistiken strukturiert speichern und die tägliche Arbeit der Vereinsadministration und Trainer unterstützen.

1.1 Funktionen

- Login mit Berechtigungen (Admin: CRUD) / (User: Read-only)
- Spieler-verwaltung (Anlegen, Bearbeiten, Löschen von Spielern)
- Trainerverwaltung (Anlegen, Bearbeiten, Löschen von Trainern)
- Teamverwaltung (intern) (Anlegen, Bearbeiten, Löschen von Teams)
- Trainingsverwaltung (Anlegen, Bearbeiten, Löschen von Trainingsterminen)
- Statistikverwaltung von internen Spielern (Anlegen, Bearbeiten, Löschen von Statistiken)
- Hallenverwaltung (Anlegen, Bearbeiten, Löschen von Hallen)
- Ligenverwaltung, ohne Tabellensystem, nur zum Zuordnen von Mannschaften (Anlegen, Bearbeiten, Löschen von Ligen)
- Spiele-verwaltung (Anlegen, Bearbeiten, Löschen von Spielen)

2. Anforderungen

2.1 Funktionale Anforderungen

2.1.1 Login

- Der User soll die Möglichkeit haben sich mit einem Benutzernamen + Passwort anzumelden
- Das System muss Benutzerrollen unterstützen und differenzierte Zugriffsrechte abhängig von der Rolle umsetzen
- Rollen:
  - Insgesamt soll es zwei Rollen geben: **Admin** & **User**
  - **Admin**: kann *alle Daten verwalten* (CRUD: Create, Read, Update, Delete)
  - **User**: kann *Daten nur einsehen* (Read-only)

2.1.2 Spielerverwaltung

- Spieler anlegen, bearbeiten, löschen
- Spieler müssen einer Mannschaft zugeordnet werden
- Zuweisung von Elternkontakte
- Erfassung persönlicher Daten: Größe, Geburtsdatum/Alter, E-Mail

2.1.3 Elternkontakte

- Pro Spieler können ein oder mehrere Elternkontakte gespeichert werden
- Name, Telefonnummer, E-Mail soll hinterlegt werden
- Das Feld 'Elternkontakt' darf bei Spielern ab der Herrenliga 2 leer bleiben, ist jedoch bei U8–U16 Spielern verpflichtend

2.1.4 Mannschaftsverwaltung

- Anlegen interner Mannschaften mit Liga, Training, Trainer
- Anlegen externer Mannschaften (z. B. Gegner)

2.1.5 Spieleverwaltung

- Neue Spiele mit Datum, Halle, Liga, internem und externem Team anlegen
- Punkte-Ergebnis speichern
- Zuordnung zur Liga
- Ein Spiel darf nicht doppelt angelegt werden (gleiches Datum + gleiche Teams)

2.1.6 Statistiken

- Pro Spiel können individuelle Statistiken pro Spieler erfasst werden:
  - Punkte
  - Fouls
  - Starter (ja/nein)

2.1.7 Training

- Zuordnung von Trainings zu Mannschaften
- Angabe: Wochentag, Dauer, Halle

2.1.8 Mitgliedsbeiträge

- Beiträgen Spielern pro Jahr zuordnen
- Angabe: Betrag, Datum der Zahlung

2.2 Qualitätsanforderungen

Die Software soll neben der reinen Funktionalität auch bestimmte Qualitätsmerkmale erfüllen:

2.2.1 Benutzerfreundlichkeit

- Die Benutzeroberfläche (GUI) soll **intuitiv** und **einfach bedienbar** sein, auch für nicht-technische Benutzer (z. B. Trainer)
- Klare Menüführung und verständliche Eingabemasken sollen eine schnelle Orientierung ermöglichen

2.2.2 Zuverlässigkeit

- Die Anwendung soll **stabil laufen** und darf unter normalen Bedingungen **nicht abstürzen**
- Bei fehlerhaften Eingaben sollen Fehlermeldungen angezeigt werden

2.2.3 Performance

- Datenbankabfragen (z. B. Spielerlisten) sollen auch bei größeren Datenmengen effizient ausgeführt werden

2.2.4 Sicherheit

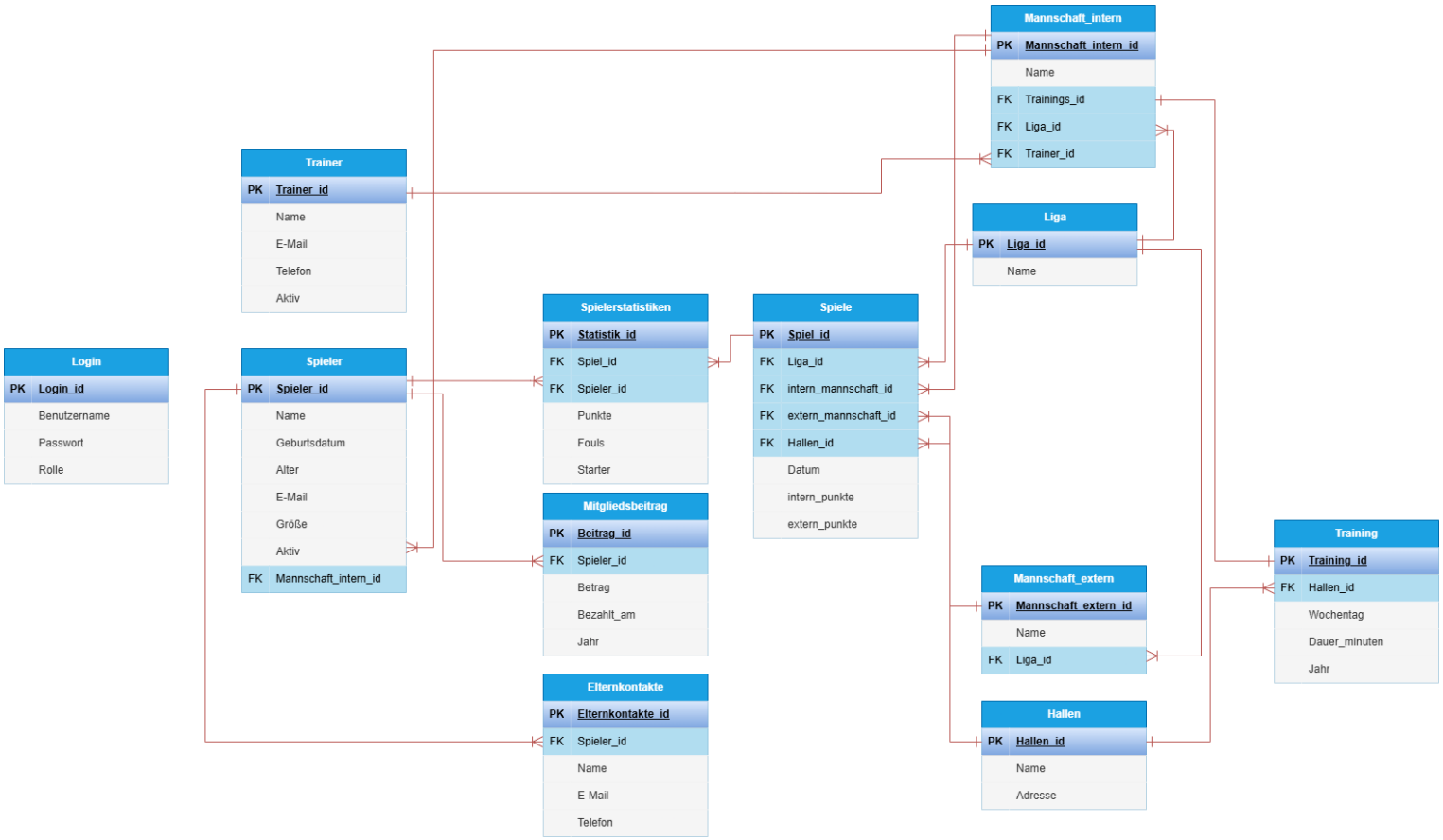
- Passwörter werden gehasht gespeichert
- Nur authentifizierte Benutzer haben Zugriff auf die Daten
- Rollenkonzept schützt vor unbefugten Änderungen (Admin & User)

2.2.5 Wartbarkeit & Erweiterbarkeit

- Der Code folgt dem Prinzip der **3-Schichten-Architektur** (GUI, Service, DAO)
- Der Quellcode ist dokumentiert und so aufgebaut, dass Erweiterungen (z. B. neue Entitäten oder Features) mit geringem Aufwand möglich sind

3. Tabellenstruktur

3.1 Beziehungen



4. Benutzeroberfläche

4.1 GUI-Anforderungen

Die Benutzeroberfläche der Verwaltungssoftware soll übersichtlich, intuitiv und rollenbasiert aufgebaut sein. Ziel ist es, sowohl Administratoren als auch Usern eine einfache Navigation und effiziente Bedienung zu ermöglichen.

Allgemeine Anforderungen:

- Klare Trennung zwischen Admin- und User-Ansicht
- Navigation über ein Menü oder Dashboard

Es soll ein zentrales **Hauptmenü** vorhanden sein, über das der Benutzer auf die Hauptfunktionen der Software zugreifen kann. Die Menüpunkte umfassen unter anderem:

- **Spieler verwalten**
- **Trainer verwalten**
- **Interne/externe Teams verwalten**
- **Spiele verwalten**
- **Ligen verwalten**
- **Trainings verwalten**
- **Mitgliedsbeiträge verwalten**
- **Statistiken verwalten**

Wählt man z.b. den Punkt **„Spieler verwalten“**, wird eine tabellarische Übersicht aller vorhandenen Spieler angezeigt. Innerhalb dieser Ansicht gibt es – **je nach Benutzerrolle** – unterschiedliche Möglichkeiten:

- **Administratoren** können Spieler bearbeiten, neue Spieler anlegen und bestehende löschen.
- **User** haben lediglich Leserechte und können die Spielerdaten einsehen, jedoch nicht bearbeiten oder neue anlegen.

Diese Struktur wiederholt sich sinngemäß für alle anderen Punkte wie **Teams**, **Spiele** usw., sodass Benutzer in jeder Übersicht Daten einsehen und – im Fall des Admins – auch verwalten können.

Die Bedienung soll konsistent aufgebaut sein, um Wiedererkennung und eine einfache Erlernbarkeit der Software zu gewährleisten

4.2 GUI – Elemente

Das Hauptmenü / Dashboard soll in einer *BorderPane* aufgebaut werden. Diese soll aus einer *MenuBar*, *Label*, *ImageView* und aus *Buttons* bestehen.

Hier ein Beispiel wie das Hauptmenü am Ende ausschauen könnte (Beta-Version):

Die GUI-Menüs für Spieler, Trainer und die anderen Objekte sollen auf einem ähnlichen Prinzip aufgebaut werden.

Die Objekte werden mit einer *ListView* ausgestattet sein damit der User/Admin alle Spieler, Trainer, etc... sehen kann. Buttons, TextField, ComboBox werden ebenfalls verwendet bei den Objekt Menüs.

5. Entwicklungsumgebung und Plattformen

Für die Entwicklung der „Basket Rolling Verwaltungssoftware“ werden folgende Plattformen und Tools verwendet:

- **Entwicklungsumgebung:** NetBeans IDE
- **Projekttyp:** Maven-Projekt
- **Programmiersprache:** Java
- **Datenbank:** PostgreSQL, verwaltet über **pgAdmin**
- **GUI-Framework:** JavaFX (für Benutzeroberflächen)
- **Versionskontrolle:** Git, in einem **privaten GitHub-Repository**
- **Zielplattform:** Desktop-Anwendung (Java)