Fitnessapp-Anwendung

Architekturentwurf von **Felix Gerber**

Versionsnummer	Autor	Änderungsvermerk
1.0	Felix Gerber	Initialer Archetekturentwurf

Datum: 14.06.2020

Zielgruppe: Der Auftraggeber, Felix Gerber

Inhalt

2. Statische Sicht – zu persistierende Daten 4
3. Statische Sicht: Objekttypen zur Laufzeit 7
4. Architektur 8
5. Auswahl des Technologiestacks 11
6. Fazit 13

Abbildungsverzeichnis

Abbildung 1: ERM-Diagramm....4
Abbildung 2: UML-Diagramm....7

Tabellenverzeichnis

Tabelle 1: Datenbankschema....5
Tabelle 2: Beispieldaten Equipment....5
Tabelle 3: Beispieldaten Standort....6
Tabelle 4: Beispieldaten Workout....6
Tabelle 5: Beispieldaten Feedback....6
Tabelle 6: Beispieldaten Zusatzinformationen....6
Tabelle 7: Beispieldaten Übungen....7
Tabelle 8: Anforderungsdefinition....8
Tabelle 9: Entwicklungsumgebungswahl....11
Tabelle 10: Dateinspeicherungswahl....12

1. Architekturstrategie

Um die richtige Architektur auszuwählen, wird das Anforderungsdokument in Einheiten unterteilt, anhand dieser werden Ideen genannt und Abschätzungen zur optimalen Implementierung getroffen.
Wesentliche Alternativen beziehen sich auf die Technologiewahl und die Unterteilung der verschiedenen Anforderungen.

Dieses Architekturdokument wurde für die erste Implementierung geschrieben. Deswegen ist keine Garantie geleistet, dass dieses Dokument 100% authentisch bleibt.

2. Statische Sicht – zu persistierende Daten

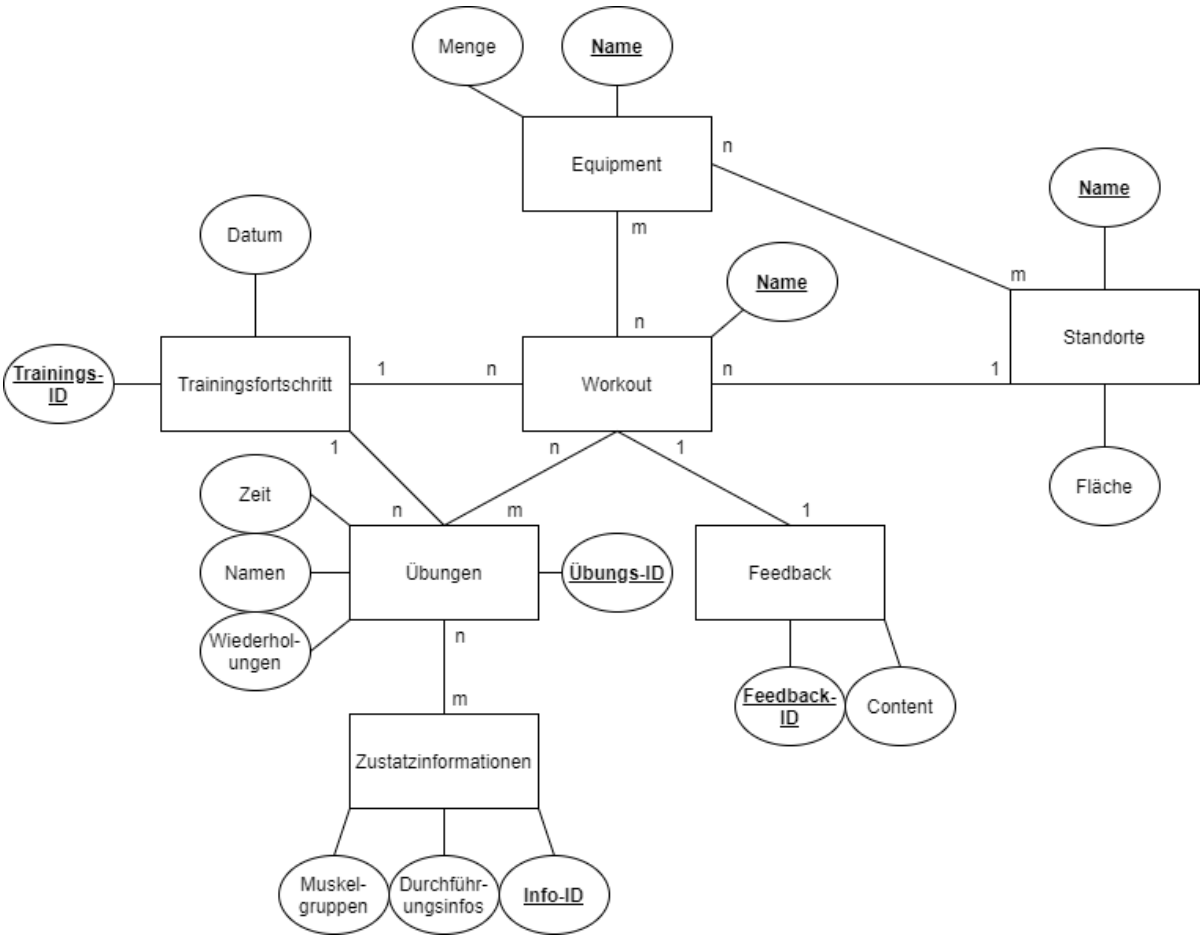

Abbildung 1: ERM-Diagramm

Für die App sind die Entitäten Workout, Trainingsfortschritt, Übungen, Feedback, Standorte und Equipment Ausschlaggebend. Um diese zu unterstützen wird eine Hilfsentität Zusatzinformationen gebildet.

Als Primärschlüssel werden Unique Identifiers gewählt, sodass jede Entität problemlos identifiziert werden kann.

Entität	**Attribut**	**Datentyp**
Equipment	Name	String
	Menge	Int

Standorte	Name	String
	Fläche	Int[]
Workout	Name	String
Feedback	Feedback-ID	String
	Content	Int[]
Übungen	Übungs-ID	String
	Zeit	Int
	Namen	String
	Wiederholungen	Int
Zusatzinformationen	Info-ID	String
	Durchführungsinfos	String
	Muskelgruppen	String
Trainingsfortschritt	Trainings-ID	String
	Datum	Date

Tabelle 1: Datenbankschema

Name	**Menge**
Langhantel 5KG	1

Tabelle 2: Beispieldaten Equipment

Name	Fläche
Zuhause	2, 2

Tabelle 3: Beispieldaten Standort

Name
Workout Nr. 1 – Oberkörper

Tabelle 4: Beispieldaten Workout

Feedback-ID	Content
ID-1	1, 3, 5, 6

Tabelle 5: Beispieldaten Feedback

Info-ID	Muskelgruppen	Durchführungsinfos
ID-1	Trizeps, Brust	Durch gleichzeitiges Durchdrücken beider Arme gelangt man in die Ausgangsposition für die Liegestütze. Dabei ruht der Körper dann auf beiden Händen sowie den Zehenspitzen. Die Finger beider Hände müssen dabei nach vorne zeigen, während die Daumen nach innen zeigen.

Tabelle 6: Beispieldaten Zusatzinformationen

Übungs-ID	Zeit	Namen	Wiederholungen
ID-1	600	Liegestütze	100

Tabelle 7: Beispieldaten Übungen

3. Statische Sicht: Objekttypen zur Laufzeit

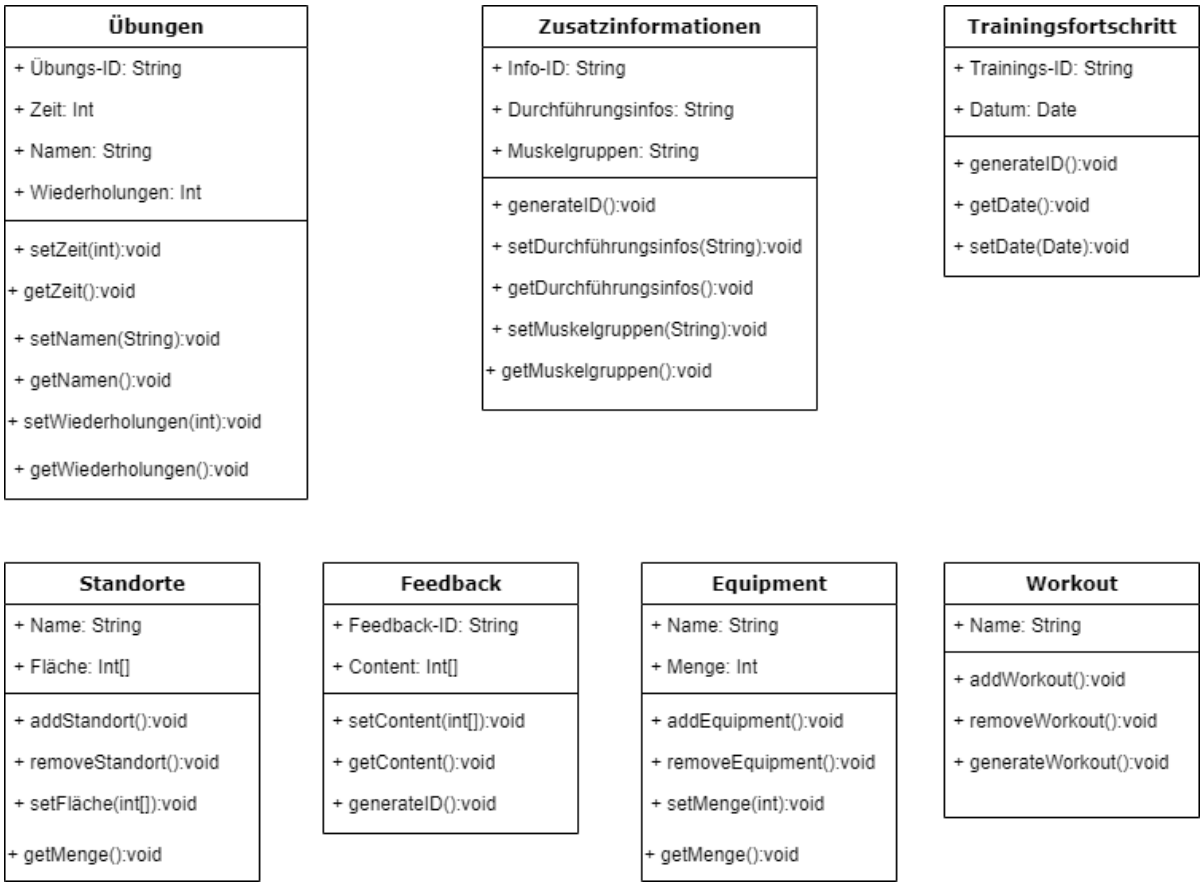

Abbildung 2: UML-Diagramm

Das UML-Diagramm ist momentan noch sehr basislastig, da beim Programmieren auftretende Problemstellungen noch nicht in diesem UML-Diagramm behandelt werden können. Es bietet jedoch einen groben Überblick auf die anzuwendende Struktur. Da die komplette App offline funktionell ist, kann auf eine komplizierte Struktur mit Serververbindung verzichtet werden.

4. Architektur

Für die Einheitsunterteilung wurde eine Auflistung der Use-Cases erstellt. Im Folgenden werden die Anforderungen und die resultierenden funktionellen Kausalitäten in absteigend wichtiger Reihenfolge genannt und vorgestellt.

1. Anforderungen

Tabelle 1 stellt die Anforderungen der App da. Dies gilt nur für die Anforderungen welche einen direkten Einfluss auf die App haben, diese werden mit einer Referenz zum Anforderungsdokument versehen.
Die Tabelle ist nach der Priorisierung der Einzelnen Anforderungen geordnet.

Nummer	Beschreibung der Anforderung	Bestandteil von
R3	Der User bekommt ein Workout erstellt	UC-3
R7/R8/R9	Informationen zur richtigen Durchführung	UC-2, UC-3
R1/R2	Trainingsfortschritt sehen/dokumentieren	UC-1
R12	Variationsmöglichkeiten einsehen	UC-2, UC-3
R5/R4	Workout variabel einstellen	UC-3
R6	Fortschritt vergleichen	UC-2
R13	Informationen Muskelgruppen	UC-2
R11/R10	Trainingsfeedback	UC-3, UC-4
R15/R16/R17	Standortfunktion	UC-5
R14	Feedback deaktivieren	UC-4

Tabelle 8: Anforderungsdefinition

2. Vorstellung der Einheiten

Im Folgenden werden alle Einheiten im Detail vorgestellt und deren Funktionalität in der App erläutert. Die Reihenfolge beschreibt wieder die Priorisierung, die bereits tabellarisch gezeigt ist.

i. R3 – User bekommt ein Workout
Dies ist ein K.O. Kriterium, da dies die essentielle Funktion der App ist.
Der User soll anhand der App ein Workout zugewiesen bekommen, dies soll mit Hilfe diverser im folgenden genannten Anforderungsmöglichkeiten variabel sein.
Außerdem werden die durchgeführten Workouts gespeichert, um sie später einsehen zu können.

ii. R7/R8/R9 – Informationen zur richtigen Durchführung
R7, R8 und R9 sind in einem Punkt zusammengefasst, da Sie jeweils die gleichen technischen Aufforderungen aufweisen.
Aus bereits vorgespeicherten Daten werden immer wieder die gleichen Durchführungen zu den jeweiligen Übungen dargestellt.
Voraussichtlich findet dies nur mit Infobildern und Text statt.

iii. R1/R2 - Trainingsfortschritt sehen/dokumentieren
Die Daten der gespeicherten, absolvierten Workouts werden visuell angezeigt, dies wird schriftlich und graphisch geschehen.
Die ist verknüpft mit R3 welches die Workouts selber sind und abspeichert.

iv. R-12 Variationsmöglichkeiten einsehen
Die Variationsmöglichkeiten werden Anhand eines Nutzerfortschrittsmessenden Algorithmus verschiedene Variationen von Übungen aus einer vorgespeicherten Datei entnehmen und diese dem User darstellen.

v. R5/R4 – Workout variabel einstellen
Dem User ist die Möglichkeit gegeben, sein Workout zu individualisieren, indem er bestimmte Parameter wie Zeit und momentane Fitness anpassen kann. Ein Algorithmus passt dann dementsprechend R3 an.

vi. R6 – Fortschritt vergleichen
Der User kann seine individuellen Statistiken (R4/R5) mit vorgespeicherten Daten vergleichen um seinen Fitnessstand besser beurteilen zu können.
Dies Unterscheidet sich von R4/R5 dadurch, dass Daten verglichen werden können.

vii. R13 – Informationen Muskelgruppen
Vorgespeicherte Daten und Infographiken werden abgerufen um dem User Wissen zu vermitteln.

viii. R11/R10 - Feedbackfunktion
Der User hat die Möglichkeit am Ende eines Workouts (R3) die Feedbackfunktion zu verwenden. Diese gibt dem Algorithmus Zusatzinformationen in Bezug auf das nächste zu erstellende Workout und passt dieses an den Nutzer an. Diese Daten werden auch gespeichert um sie in R1/R2 ansehen zu können

ix. R15/R16/R17 – Standortfunktion
Dies bezieht sich nicht auf den Geographischen Standort der von mobilen Endgeräten bezogen wird, sondern auf voreingestellte Orte welche modifiziert werden können auf diverse Attribute.
Diese können bei der Erstellung eines Workouts (R3) angegeben werden, damit der Algorithmus ein optimales Workout bereitstellen kann.

x. R14 – Feedback deaktivieren
Die Option soll dem Nutzer gegeben werden, die Feedbackfunktion (R11/R10) zu deaktivieren. Damit der Algorithmus weiterhin korrekt funktioniert, muss dies im Quellcode weitergeleitet werden.

5. Auswahl des Technologiestacks

Die Architektur zum Design der App wird im Folgenden genannt und Erläutert. Die verwendetet Architekturentscheidungen sind von oben nach unten priorisiert angegeben. Alternative Architekturen werden tabellarisch dargestellt und eine Begründung zur Entscheidungsfindung genannt.
Für ein besseres und übersichtlicheres visuelles Ansprechen sind die verwendeten Architekturen unterstrichen.

1. Entscheidung Entwicklungsplattform
   Das Anforderungsdokument beschreibt, dass die App für mobile Endgeräte entwickelt wird. Die App soll außerdem komplett offline funktionieren und keine externen Informationen benötigen.
   Es wird davon Ausgegangen, dass mobile Endgeräte Smartphones und Tablets sind und nicht Notebooks oder Laptops.
   Auf unterschiedliche Bildschirmgrößenskalierungen wird aufgrund der hohen Anzahl an Möglichkeiten keine Rücksicht auf jede Bildschirmgröße genommen, da dies sich im Aufwand als sehr schwierig darstellt.

	Android Studio	Unity 3D
Unterstützt Android	Ja	Ja
Unterstützt Apple	Ja	Ja
Einarbeitungsaufwand	Gering	Hoch
Programmiersprache	Java/Kotlin	.NET

Tabelle 9: Entwicklungsumgebungswahl

Da Apple und Android die Marktführenden Betriebssysteme sind ist mit dieser Möglichkeit sehr flächendeckend eine Lösung mit beiden Architekturen getroffen. Aufgrund des Unterschieds im Einarbeitungsaufwand wird die Entscheidung jedoch für Android Studio getroffen, da dies den Zeitaufwand drastisch reduziert.

2. Datenspeicherung
   Um die Anforderungen R1/R2/R4/R5/R10/R11/R14/R15/R16 umsetzen zu können müssen Zwischendaten gespeichert werden. Die Datenbasis muss dabei auf dem Gerät des Users sein, da die App offline funktionieren soll. Außerdem gilt es die Verwaltbarkeit, Verfügbarkeit und Geschwindigkeit zu überprüfen, da diese logischerweise eine wichtige Rolle spielen in der Entwicklung und Instandhaltung der App.

	SQL-Server	**Internes Datei-System**
Offline	Nein	Ja
Skalierbarkeit	Gut	Schlecht
Verwaltbarkeit	Gut	Gut
Verfügbarkeit	Gut	Gut
Geschwindigkeit	Gut	Gut

Tabelle 10: Dateinspeicherungswahl

Fazit:
Da die Grundanforderungen nicht geändert werden sollen, wird um diese herumgearbeitet und sie als Grundgerüst benutzt, auch wenn dies Teilweise schlechtere Optionen hervorruft, stellen diese den geringsten Aufwand mit dem größten Nutzen da.
Die Wahl der Entwicklungsplattform fällt auf Android Studio und die Wahl zur Datenspeicherung aus oben genannten Gründen auf ein Internes Datei-System.

6. Fazit

Die erste Entwicklung der Anwendung basiert auf diesem Dokument und dem Anforderungsdokument. Dem Entwickler sind jedoch Änderungen an Funktionen- und Variablennamen überlassen, sowie weitere Attribute und Entitäten zu ergänzen.

Auf ein Verteilungsdiagramm wurde für dieses Dokument verzichtet, da die App offline auf dem mobilen Endgerät des Nutzers laufen soll und somit keine Verteilung zu Servern oder Datenbanken nötig ist.