Advanced Workout Planner: Technical Documentation

Table of Contents

1. Introduction

1.1. Purpose of this document

This document is a technical documentation of my project Advanced Workout Planner - mobile Flutter application. The purpose of this document is to keep track of the progress of the project and to serve as a source of technical information that will help me and other potential team members to recall important parts of the project in case they are forgotten. Technical documentation will be the primary source of information to any new members of the team so they can have all the necessary information available in one document.

1.2. Scope

The initial idea of application Advanced Workout planner is to help me with my everyday struggle – workout planning. After some years of going to the gym and working out, it becomes really hard to plan diverse and challenging workouts, so I have turned on technology to solve the problem for me. The goal is to create an application that produces challenging, uncommon and most importantly diverse workouts every time and consequently help experienced and non-experienced people, who struggle to come up with unique bodybuilding workouts or have a lack of experience doing so, to achieve the best workout experience every day.
The special emphasis is on user-friendliness. This means that the application has to be easy to operate and understandable to all users, even ones that do not yet have sufficient knowledge to

take advantage of all functionalities of the application. Because of the expertise needed to use the advanced parts, the application is inclined to help the user gather sufficient knowledge as easily as possible.

The whole project is separated into 3 thematically different parts: Viewing exercises, Manual workout creation, and Automatic workout creation.

Part 1 (*Viewing exercises*): The objective is to create a user-friendly educational application with an appealing and fluent design that can present a various number of bodybuilding exercises including their names, correct execution, target muscle group (with words and pictures) and other helpful information. Furthermore, it supports hyperlinks to an additional in-depth explanation for each exercise and muscle group. Additionally, it supports adding exercises under favorites and viewing them in a separate section.
It includes a powerful search engine for searching exercise names and filtering exercises with numerous filters (exercise properties, target muscle group).
It is also possible to search for exercises that target a specific muscle group or subgroup with so-called sub-searching. This is a method of my design where it is possible to display all subgroups of a muscle group or move a level lower where the user is presented with subgroups under the parent group, depending on a given command type (tap, long press). From here it is possible to view data (e.g. exercises) of the presented subgroup or move to even lower muscle group levels if possible.

Part 2 (*Manual workout creation*): The goal is to enable adding exercises to the "Workout section" where workout can be further edited and improved. In the workout section, the user can switch exercises with new ones, change their order and remove or add exercises. The application allows us to transform single exercise into a superset or tripleset and add different types of drop sets to exercises. The user is also able to save workouts and rate them. Switching exercises and creating supersets and triplesets is possible with an improved search engine that is similar to exercise viewing. The search engine can search for similar exercises and recommend exercises that would fit best in the superset/tripleset of the exercise.

Part 3 (*Automatic workout creation*): The objective is to develop a mechanism based on predefined exhaustive parameters, which can be fully customized, creates a unique and diverse workout that can be further saved and manually edited. Automatic workout creation will support various modes in which the parameters are already predefined but can be further edited.

To be able to complete each part of the project, the previous part has to be completely developed, otherwise, the next part is underdeveloped.

1.3. Definitions, acronyms and abbreviations

Class Diagram - Describes the structure of a system
Activity Diagram - Describes activities and actions taking place in a system

Exercise properties - In this project, I refer to additional exercise classification as exercise properties. These properties include difficulty, utility, mechanics, and force.

Difficulty - Represents how physically difficult is to perform the exercise (Hard, Medium, Easy).

Utility - Type of muscle intensity an exercise has on a muscle.

- **Basic** - Exercise that has greater absolute intensity on the muscle
- **Auxiliary** - Exercise that has greater relative intensity on the muscle

Mechanics - Type of movement used to perform an exercise.

- **Compound** - Exercise that involves two or more joint movements.
- **Isolated** - Exercise that involves just one discernible joint movement.

Force - Direction of the body movement while performing the exercise.

- **Pull** - Movement toward the center of the body during the contraction of the target muscle.
- **Push** - Movement away from the center of the body during the contraction of the target muscle.

Superset - A form of strength training in which you *move quickly* from one exercise to a separate exercise *without taking a break for rest* in between the two exercises.

Tripleset - A form of strength training in which you *move quickly* from one exercise to a second and third separate exercise *without taking a break for rest* in between the three exercises.

Dropset - A technique for *continuing an exercise with a lower weight* once muscle failure has been achieved at a higher weight.

Transformation into superset/tripleset - A process of finding an exercise to add to an existing exercise and consequently form a superset or tripleset.

Target muscle group - muscle group that is mostly activated while working certain exercises.

Level 0 subgroup (sg0) - The main muscle subgroup which does not have a parent muscle group (Eg.: Shoulders).

Level 1 subgroup (sg1) - The first muscle subgroup (Eg.: Deltoids).

Level 2 subgroup (sg2) - The second muscle subgroup - optional (Eg.: Front Deltoids).

1.4. Overview

Section 1 is the introduction and includes an idea and description of the project. It includes definitions of certain terms used in this document.

Section 2 explains the reason behind the development of a new fitness application even though the market is full of fitness and weight loss products.

Section 3 represents a guide to external resources on how to run and contribute to an existing application.

Section 4 explains an overview of the project architecture

Section 5 represents the current results of the application. It includes pictures on how the application looks like and what functionalities it provides.

2. Existing resources

Currently, there are a lot of fitness mobile applications on the market that can display exercises and create workouts. Most of them lack in ether number of exercises or ability to create workouts difficult enough to be used by professionals. A great number of them require a very high subscription fee which discourages a lot of potential customers. I have not found a single mobile application that would contain the amount of exercises and the extensive customization for a workout creation that I intend to implement.

3. Run/build the application

To run and build a Flutter application you need to have a working Flutter installed on your computer. To install Flutter follow these instructions.

When you have successfully installed Flutter on your computer and tested Flutter sample application you are ready to clone my project Advanced Workout Planner and start contributing. Request the project owner to add you to a list of collaborators to a private GitHub repository and follow this ultimate github collaboration guide.

4. Overview of the architecture

4.1. Muscle group and subgroup parameterisation

Every exercise has at least one target muscle group. Some exercises mostly exercises for lower back and legs, have 2 or even 3 target muscle groups.
Every exercise target a muscle group that is at least level 1 subgroup (level 0 subgroup is the main muscle group).
Only some exercises for the back do not have explicitly set the target muscle group. These exercises are classified as "Generic back" exercises.

1 - Shoulders
- **1.1** - Deltoid
  - **1.1.1** - Front Delts
  - **1.1.2** - Side Delts
  - **1.1.3** - Rear Delts
- **1.2** - Rotatory Cuff

2 - Arms
- **2.1** - Biceps
  - **2.1.1** - Biceps
  - **2.1.2** - Lower Biceps
- **2.2** - Triceps

3 - Forearms
- **3.1** - Upper-Outer Forearms
- **3.2** - Hand Flexor
- **3.3** - Hand Extensor

4 - Back
- **4.1** - Generic Back
- **4.2** - Upper Back
  - **4.2.1** - Traps
  - **4.2.2** - Levator Scapulae
  - **4.2.3** - Middle Trapezius
- **4.3** - Middle Back
  - **4.3.1** - Lower Trapezius
  - **4.3.2** - Rhomboids
  - **4.3.3** - Lats
  - **4.3.3** - Teres Major
- **4.4** - Rotatory Cuff
  - **4.4.1** - Infraspinatus
  - **4.4.1** - Teres minor
  - **4.4.2** - Subscapularis

5 - Chest
- **5.1** - Lower Chest
- **5.2** - Upper Chest

- **5.2** - Pectoralis Minor
- **5.2** - Serratus Anterior

6 - Core
- **6.1** - Abs
- **6.2** - Obliques
- **6.3** - Lower Back

7 - Legs
- **7.1** - Hips
  - **7.1.1** - Gluteus Maximus
  - **7.1.2** - Hip Abductors
  - **7.1.3** - Hip Flexors
  - **7.1.4** - Deep Hip External Rotators
- **7.2** - Thigs
  - **7.2.1** - Quadriceps
  - **7.2.2** - Hamstrings
  - **7.2.3** - Hip Adductors
- **7.3** - Calves
  - **7.3.1** - General or Gastrocnemius
  - **7.3.2** - Soleus
  - **7.3.3** - Tibialis Anterior

4.2. Exercise representation

4.2.1. In raw data

Exercises are represented in a JSON file format. Each exercise consists of the following data: (all parameters are mandatory, except for mg1 and mg2)

- **exID** - Represents the unique ID number of each exercise
- **mg, mg2, mg3** - ID of a target muscle group, combined of up to 3 numbers, separated with a dot (see [Muscle group and subgroup parameterization]). Mg1 and mg2 are optional and used for exercises that target more than one muscle group.
- **eqType** - Equipment type used to perform the exercise. Possible are the following equipment types:
  - Barbell
  - Dumbbell
  - Smith (smith machine)
  - Cable
  - Machine
  - Suspended (suspension handles)
  - Body Weight
- **name** - Represents the name of each exercise and has to be unique for each exercise. Exercise

names might differ depending on the information source.

- **diff** - Represents exercise difficulty. Exercise can have easy (diff = 1), medium (diff = 2) or hard (diff = 3) difficulty.
- **utility** - Type of muscle intensity an exercise has on a muscle (basic, auxiliary).
- **mechanics** - Type of movement used to perform an exercise (compound, isolated).
- **force** - Direction of the body movement while performing the exercise (push, pull).
- **link** - Hyperlink to the website containing an in-depth explanation of the exercise.

4.2.2. In a program

Exercises in a program are represented and used as an 'Exercise' objects in /lib/Logic/Exercise.dart file. These objects are generated/parsed at the start of the application.

Structure of the Exercise object:

Object **Exercise** has the following attributes:

```
String _exID;

// Supporting up to 3 target muscle groups for one exercise.
MgId _mg;
MgId _mg1;
MgId _mg2;

String _eqType;
String _name;
String _diff;
String _utility;
String _mechanics;
String _force;
String _link;
```

All attributes are represented as a String, except for attributes representing muscle groups. These attributes are represented as a new object MgId (Muscle group ID).
A new object representing muscle group is implemented because of the easier implementation of Exercise filtering with muscle groups and subgroups.

Structure of the MgId object:

```
// MgId represents muscle group id and consists of 3 levels of subgroups:
// sg0 is the main muscle group - Eg.: Shoulders
// sg1 is the first subgroup level - Eg.: Deltoid
// sg2 is the second subgroup level - Eg.: Front Deltoid
class MgId {
  // A string representing a target muscle group, combined from all subgroups
separated with a dot.
  // Mg attribute from raw data is passed as a fullMg parameter.
  String _fullMg;

  // Muscle subgroups
  String _sg0;
  String _sg1;
  String _sg2;

  List arr = new List<String>();

  // Parse full mg id to subgroups
  MgId(this._fullMg) {
    arr = _fullMg.split('.');

    if (arr.length == 1) {
      _sg0 = arr[0];
      _sg1 = "";
      _sg2 = "";
    } else if (arr.length == 2) {
      _sg0 = arr[0];
      _sg1 = arr[1];
      _sg2 = "";
    } else if (arr.length == 3) {
      _sg0 = arr[0];
      _sg1 = arr[1];
      _sg2 = arr[2];
    }
  }

  @override
  String toString() {
    if (this._sg1 == "") {
      return this._sg0;
    } else if (this._sg2 == "") {
      return this._sg0 + "." + this._sg1;
    } else {
      return this._sg0 + "." + this._sg1 + "." + this._sg2;
    }
  }

  String get sg0 => _sg0;

  String get sg2 => _sg2;

  String get sg1 => _sg1;
}
```

4.3. UML Diagrams

4.3.1. Class Diagram

Class diagram is represented as a .svg picture here.

5. Results

6. Conclusion