**GitHub Username**: Nils27

# WFA - Workouts For All

## Description

A fitness app, taking the user through a choice of 3 workouts.
Often the average user is not sure of what exercises to do, how many reps, sets, etc. This app will help guide them through a complete workout. The app will also solve the issue of users not remembering how which set they are on, due fatigue and concentrating on form. Users also tend to over rest, meaning their rest times between sets are not optimal for getting the most of their workouts.

The PRO version will not have any ads, and it will have an extra workout from the free version.

This app will be entirely written solely in the Java programming language. All strings will be stored in the strings.xml to enable RTL switching on all layouts. Accessibilty will be suuported as per the guidelines in https://developer.android.com/guide/topics/ui/accessibility/. From useful and descriptive labelling of both static and dynamic elements, smart grouping of content, etc.

## Intended User

User can be anyone who would like to perform some exercises/workouts. Suitable for all ages.

## Features

1. Guides users through a well structured workout.
2. Ability to update the exercise MIs, reps & weights, etc.
3. Provides automatic countdowns for rest times post exercise to ensure time is fully utilised.
4. Full tracking of where in the workout the user is (which exercise and which set, etc)
5. Sending a portable version to Wear OS devices to allow users to leave phones in lockers where they will not risk being damaged in the gym.
6. Saves user settings.
7. Saves users workout MI data.
8. Viewing of previous workout history.
9. Incorporates Google Fit.
10. Ads both banners and interstitual in the free version.

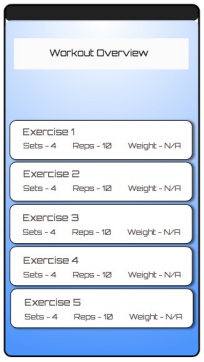
# User Interface Mocks

Note - Colour schemes, styles, overall look and feel may change.

## Screen 1


Main screen – provides user the choice to perform a workout, see previous workout MI or app settings.

## Screen 2

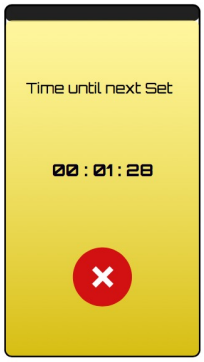



Workout Overview screen – recyclerview will be used to display all exercises in the workout along with the exercise details.

## Screen 3

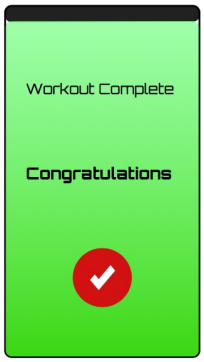
Workout Screen – Shows the current exercise and exercise details (reps, weight, etc). Will also show a picture to help identify the exercise or a clickable YouTube clip to show users how to perfimr the exercise. Also an option to send the workout to the Wear device (currently displayed as a star, but this will change to an icon that is more appropriate).

## Screen 4




Rest Screen – Automatically shows the user the recommended rest time before the next exercise. Can be dismissed early if the user choose to do so.

## Screen 5

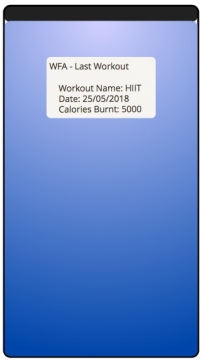



Workout Finished Screen – Informs the user the workout has been completed.

## Screen 6


Workout History Screen – Shows the user the historical workout data.

## Screen 7 - Widget




Widget – Shows the users last workout data.

# Key Considerations

### How will your app handle data persistence?

- SQLite db for data
- Content Provider / Room

An AsyncTask will be used to start the retrieval of data if it is retrieved from a web service.

### Describe any edge or corner cases in the UX.

Possibly it will keep the workout/app active much like a music playing app so users can get back to the workout easily from whatever distraction took them away from the workout.

### Describe any libraries you'll be using and share your reasoning for including them.

All libraries will be the latest versions as at 5th June 2018

- Android Studio *v3.1.2*

- Gradle – *services.gradle.org/distributions/gradle-4.4-all.zip*
- Glide – *(com.github.bumptech.glide:glide:4.7.1)* – for handling the loading and caching of images.
- Retrofit – *(com.squareup.retrofit2:retrofit:2.4.0)* – handle network calls for data loading.
- Gson – *(com.squareup.retrofit2:converter-gson)* – for JSON parsing to POJO
- DataBinding – *(will need Android 4.0 -API level 14)* - to avoid findViewByIds.
- Exoplayer – *com.google.android.exoplayer:exoplayer:r2.2.0* – possibly use of the player for youtube clips to show how exercises are performed.
- Design Support Library – *all support libraries will be 27.1.1 (com.android.support:support-)* – for material design aspects.

### Describe how you will implement Google Play Services or other external services.

Google play services will be needed for the Wear OS aspects.

## Required Tasks

### Task 1: Project Setup

Libraries:

- Glide
- Retrofit
- Design library
- Google Fit Data Model

### Task 2: Implement UI for Each Activity and Fragment

Main screen (3 options)

- Select workout
    - Workout Overview (Wear OS equivalent)
        - Workout Screen (Wear OS equivalent)
        - Rest/Recovery Screen (Wear OS equivalent)
- Previous Workout MI
    - Workout History
- Settings
    - View Activity (calendar view)

SubTasks

- Build UIs above
- Get Uis to communicate with each other

### Task 3: Enter Workouts

- Create the SQLite & Content Provider needed.
- Set up all the data needed for the exercises/workouts.

Have 3 stored workouts – Bodyweight Workout 1, Bodyweight Workout 2, HIIT.

## Task 4: Rest timer functionality

- Create the rest timer functionality

Create a rest timer class (maybe use a 3rd party lib if one exists).

## Task 5: Handle pictures and youtube videos

- If network images are used ensure they are retrieved correctly.
- Implement youtube functionality within the app.

To help users understand how to perform the exercise with the correct form.

## Task 6: Material Design

- Implement Material Design
- Animations
- Widget touches
- Finshing touches to make apps seem intuitive

Try to implement as much of material design as possible. Use shared element transitions, nice animations to show button touches, etc.

## Task 7: Wear OS

- Implement Wear OS

Implement the Wear OS aspects.

## Task 8: App Variants

- Build Free variant
- Build Paid variant

Interstitual ads maybe used in the free variant.

## Task 9: Prepare app for release

- App icons
- Store listings
- Promotional video for listing