GitHub Username: aitchiss

Workout Pal

Description

Plan your workouts and track your progress with Workout Pal. Create custom exercise plans, or get inspired by pre-loaded workout templates. Workout Pal guides you through your workout, including planned rests and tracks your progress when you complete a session. View your activity in a handy graph, and see your progress in each activity type in your personalised workout log.

Intended User

This app is for people who go to the gym and engage in workout activities, e.g. weightlifting, HIIT, etc.

Features

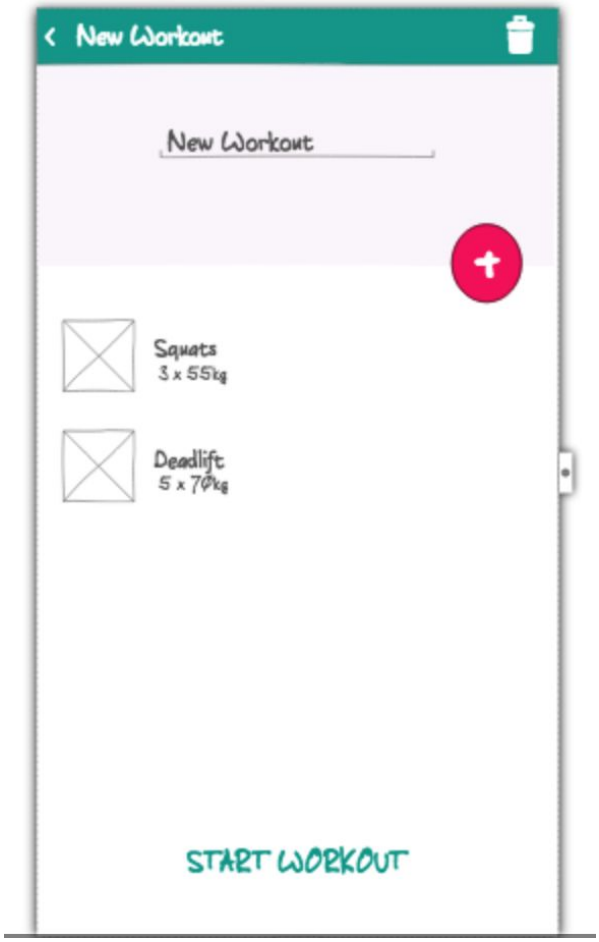
- Build customised workout plans or choose from a range of pre-loaded templates
- Follow the workout on the app while at the gym
- The app saves the users activity and allows them to see in handy graphs:
    - How many sessions they have logged each week
    - How their weight lifting progress has improved over time

User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

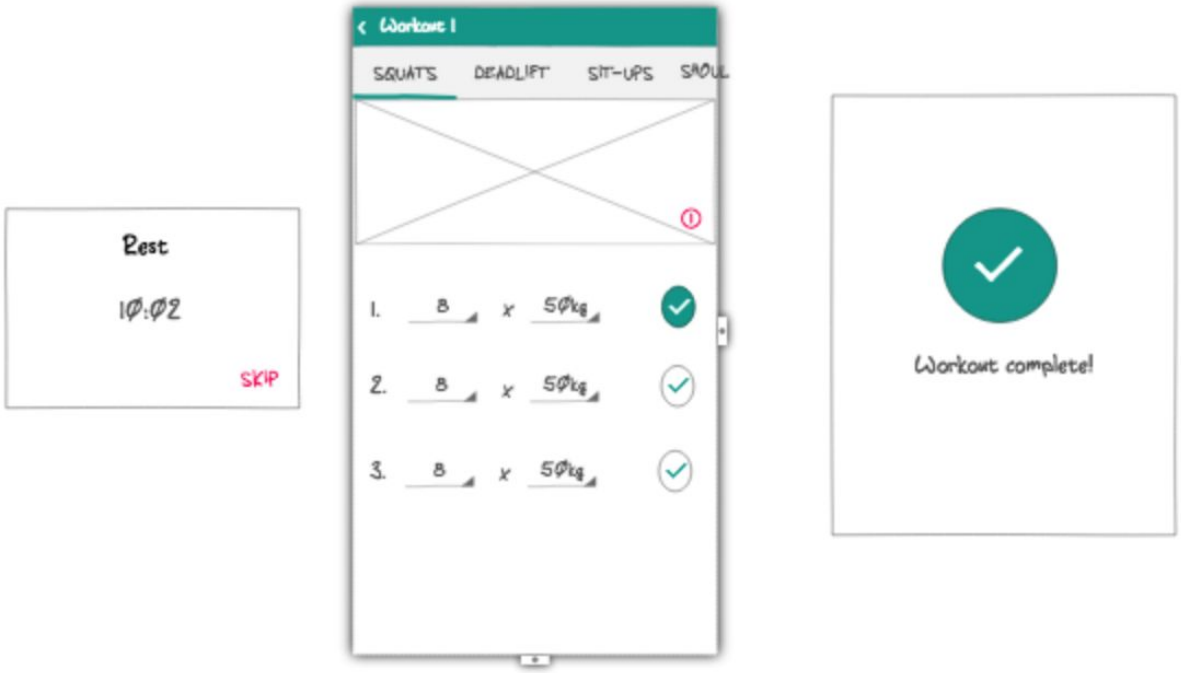
Home Screen & Menu:

Workout detail screen:



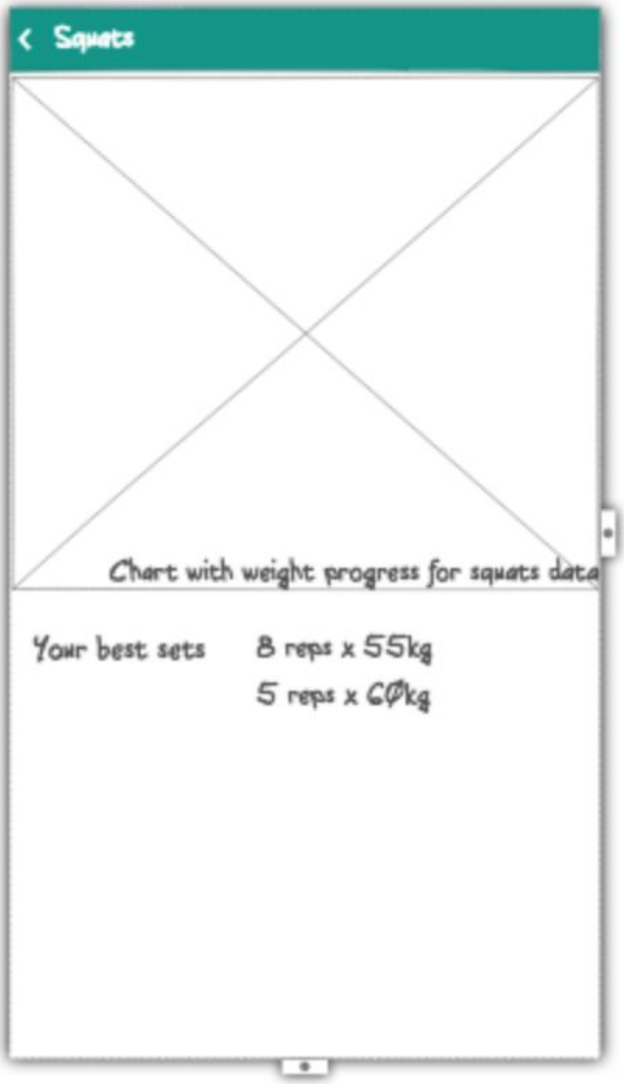
Add to workout screen & ‘ADD’ dialog:

Play workout screen, rest dialog and ‘complete’ dialog:

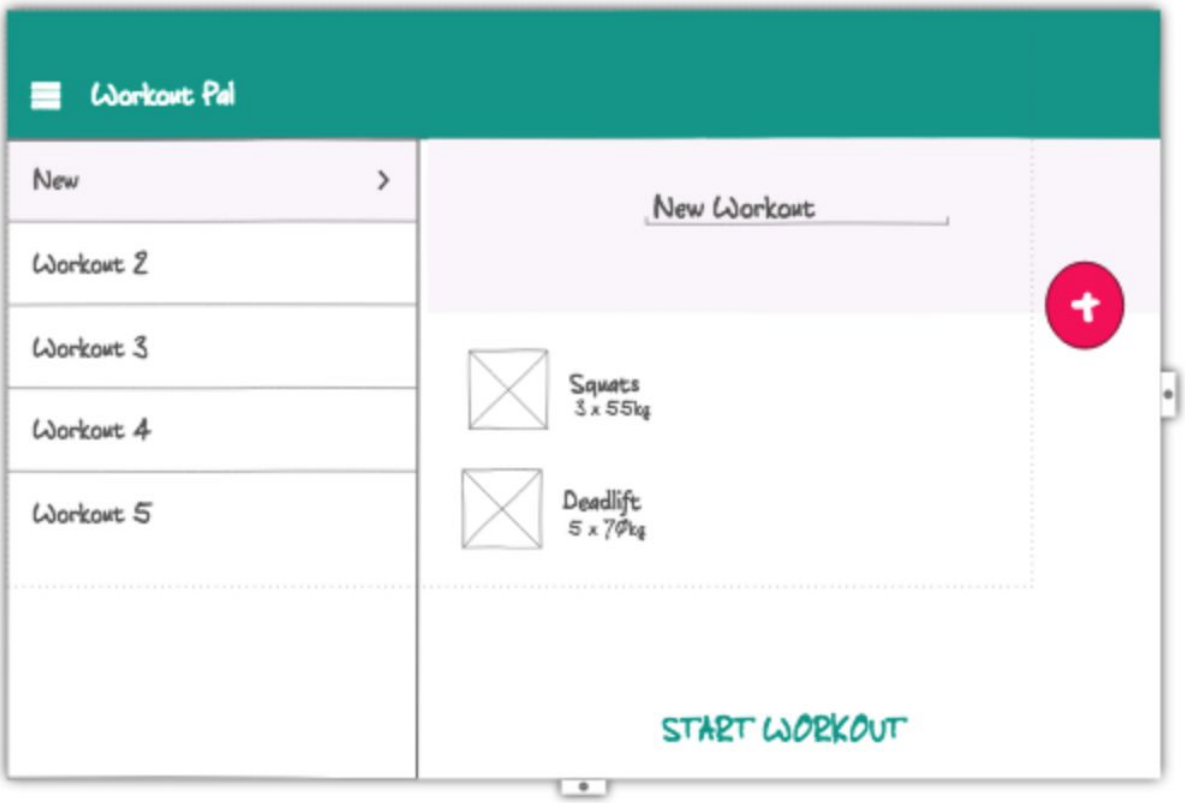

Rest
10:02
SKIP
Workout 1
SQUATS
DEADLIFT
SIT-UPS
1. 8 x 50kg
2. 8 x 50kg
3. 8 x 50kg
Workout complete!

Log screen:

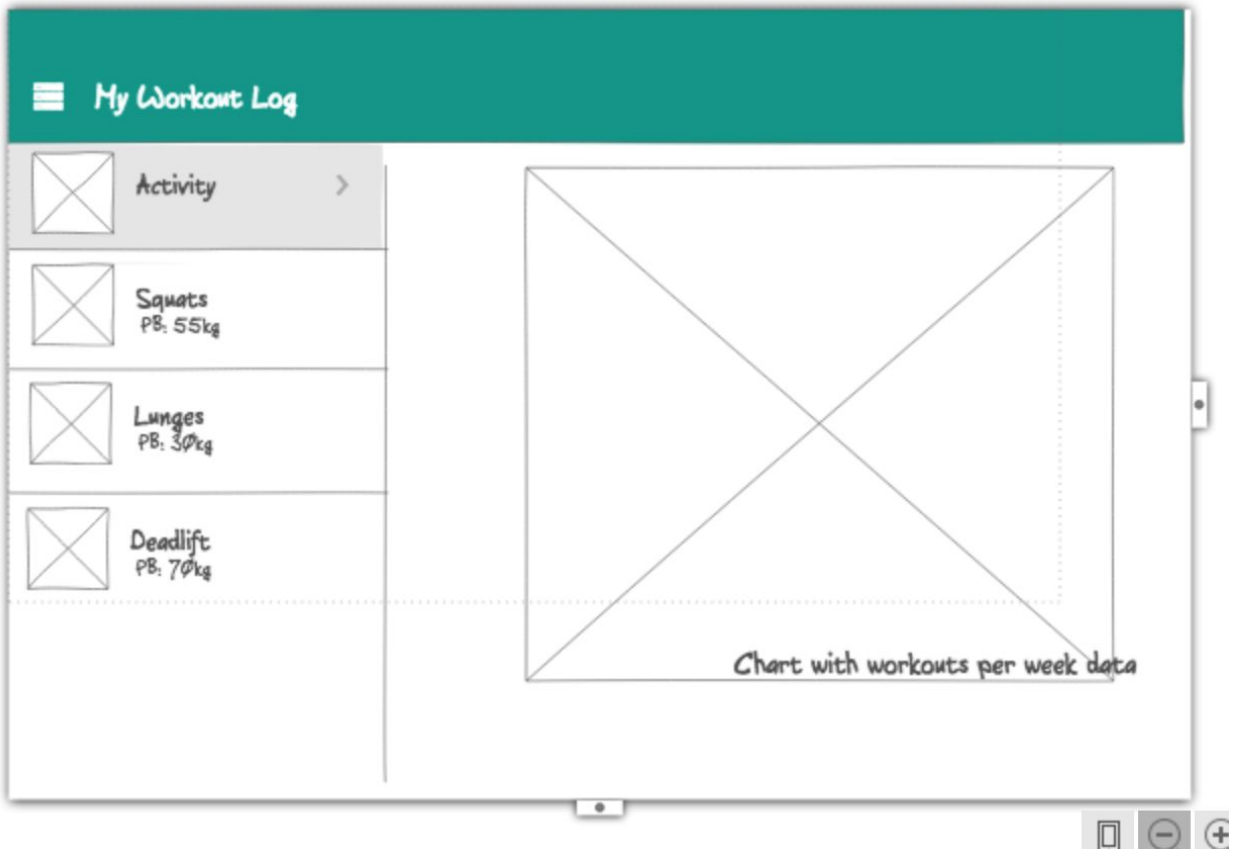
Workout log activity detail screen:



Home screen tablet layout:



Workout log: tablet layout:



Key Considerations

How will your app handle data persistence?

- Firebase realtime database for user-entered data, with access controlled by Firebase Authentication
- Firebase realtime database for storing list of activities from API (these will be synced on a regular basis by a JobDispatcher.
- Shared preferences for preference settings (e.g. kg/lbs)

Describe any edge or corner cases in the UX.

- Empty states will need to be included for ‘new’ workouts with no activities added yet, and graphs where no data is available yet
- Placeholder images will need to be available for activities which have no photo, or photo fails to load
- If the user taps ‘back’ mid-way through a workout, the app needs to handle storing the progress they have made so far

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

- Okhttp for making a network call to fitness API to retrieve work out activity types at scheduled intervals
- AndroidPlot for graphs in workout log (https://github.com/halfhp/androidplot)

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

- Firebase authentication to manage initial log in
- Firebase realtime database for data persistence of user info

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

- Set up Firebase Authentication
- Set up the Firebase Realtime Database for user activity
- Set up the JobDispatcher for syncing training activity types into the database at regular intervals

Task 2: Implement UI for Each Activity and Fragment

- Implement the Firebase Authentication login page
    - Customise to app style
    - Create the empty Dashboard UI as a placeholder for now to confirm successful login

Task 3: Create the 'Home' view

- Create a fragment for the Home/Workouts List UI

- Create a main activity that loads the fragment
- Inc. recycler view and adapter

Task 4: Create the 'Workout Detail' view

- Create a fragment for the workout detail
- Create a WorkoutDetail activity that shows the fragment
- Populate the UI with the selected workout details OR, with placeholder UI for a new workout (depending how the activity was launched)

Task 5: Create the 'Add to workout' view

- Create a fragment for the add to workout view
- Create an activity to display the fragment
- Populate the UI with the activities retrieved from the fitness API
- Hook up the UI elements to change in line with the current spinner selection
- Create the 'ADD' dialog with entry fields for sets/reps/weight/etc
- On 'DONE', update the related workout with the new activity and ensure saved in DB.

Task 6: Create the 'Play workout' view

- Populate with the activities associated with the workout
- Save user's progress/history as each element is marked done
- Show dialog rest timers if appropriate
- When the last activity is completed, show the 'completed' dialog notification before returning to the workout detail page

Task 7: Create the 'Log' view

- Implement the AndroidPlot library
- Show the user's activity on an AndroidPlot graph
- Populate the UI with the user's Personal Best in each activity they have completed, making the recycler view and associated adapter as needed

Task 8: Create the 'Log detail' view

- Show the user's progress for the individual activity on an AndroidPlot graph

Task 9: Create the tablet layouts

- Create master/detail tablet layouts for the home screen and the workout log
- Create an optimised UI for all screens in tablet mode

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"