RepQuest

Jaylen Cook, Dylan Hulon, Austin Jones, Benjamin Johnson

# Design Description

RepQuest is an Android native application that allows users to track their workouts while simultaneously incentivising the user’s commitment to their fitness goals. It does this by offering a multitude of features that allow the user to not only keep a detailed history of their fitness journey, but also have the opportunity to earn experience and level up their avatar by adhering to their fitness goals. Users will be able to create custom workout templates that they can use to record sets/reps/weight or distance/time. This allows users to keep a detailed, annotated workout history for later review and also allows users to track their workouts in real time with minimal taps for decreased distraction. The real time tracking of workouts will also include a persistent rest timer that stays active while users move through the app for effortless tracking of recovery periods. Experience points, personal records, and workout streaks will also be tracked automatically. The data from these workouts will be available to view either in the format of a filled-in template or organized into graphs to easily identify trends. This data will be available across multiple devices through the use of the Firebase Realtime Database.

The purpose of this design description is to explain how the system is organized and how its parts work together to deliver the behavior described above. It is intended to be used in tandem with the attached supporting diagrams in order to create a detailed illustration of how this system should be designed and implemented. The attached diagrams include a block diagram, illustrating the overall structure of the system, a component diagram which lists the classes that will be used and their relationships to each other, a UI storyboard to depict how the user will move through the app, and storage documentation which provides a model of how the stored data will be organized. The goal of this document and its appendices is to provide all information necessary to be used to implement every requirement correctly without any prior knowledge of this application.

Beginning with the overall structure of the system, the major components of the system will include the UI layer, data layer, local storage, a network connection, the Firebase server, and the Firebase Realtime Database. The UI layer will be responsible for rendering and displaying all of the components that the user interacts with, including the buttons, navigation bar, rest timer, and screens for workout, history, stats, goals/gamification, and settings. This layer will also consist of state holders that hold data to be made available to the UI and handle the logic behind user actions. This layer will not access storage or the network directly, rather it will interface with the data layer, which is responsible for the business logic of the application. The data layer will consist of classes that will expose data to the UI layer of the app and handle the core logic of how this data is manipulated. Each class within this layer will have the responsibility of working with only one source of data, being an entity within the local storage component. The UI and data layers will be discussed in more detail when discussing the component diagram later in this document. The local storage will consist of the data that is stored locally on the device, providing an offline-first experience. This data includes the workout template, exercise template and user data. This component will be discussed in more detail when discussing the entity relationship diagram later in this document. A small network connectivity component will monitor network status and inform data repositories when online to indicate when to send writes to the local database to the remote database immediately. When offline, writes to the local database are queued and sent to the remote database the next time network connectivity is established while the UI continues to operate against local storage. The backend component of the application consists of the Firebase server which authenticates users and stores user data in the Firebase Realtime Database. The backend is treated as a replica of the local storage that can be synced across multiple devices. There is no server-side business logic, only CRUD operations.

The Entity-Relationship Diagram illustrates the interaction between the three main components of the application. Each individual user can have one or more exercises and one or more workouts, and their primary key is their user id. Each individual exercise is included in one or more workouts, but belongs to only one user. Each exercise’s primary key is the exercise id, its foreign key is the user id, and its other attributes are the exercise name, unit of completion, number of sets, and rest period length. Each individual workout contains one or more exercises, but belongs to only one user. Each workout’s primary key is the workout id, its foreign keys are the user and exercise ids, and its other attributes are workout notes if any have been made and bodyweight during that workout.

# Revised Requirements

1. Navigation Bar
   1. On the bottom of the screen, persistent throughout the entire app unless the user is in an exercise
   2. Has a button to switch to Workout Screen
   3. Has a button to switch to History
   4. Has a button to switch to Stats
   5. Has a button for the Goals/Gamification Tab
   6. Has a button for Settings
2. Workout Screen
   1. “Create a Workout” Button
      1. “Add Exercise” Button
         1. Textbox for Exercise name
            1. If the user enters the same name as an existing exercise, a window will pop up notifying them of this and they will have to choose a different name
         2. “Unit of Completion” Radio Buttons (Reps, Distance, time, etc.)
            1. If Reps, Checkbox for “Weighted Equipment”

Numerical Input for weight of instrument

* + - 1. “Incremental” Checkbox
         1. If selected, a numerical input for how much to increment an exercise will appear
      2. Numerical Input for a finished rep/distance/time in current weight/length units for rep/distance or HH:MM:SS for time
         1. Weight/length units determined from settings, imperial by default (pounds, miles to 1 decimal place) or metric (kilograms, kilometers)
      3. Numerical Input for number of sets
      4. Checkbox for option to add a resting period
         1. If checked, Numerical Input for how long to rest for in HH:MM:SS format
      5. Confirm/Submit Button
      6. User can choose to save the Workout as a Workout Template locally
         1. A window will pop up informing the user if save was successful or displaying the error if it was unsuccessful
    1. “Remove Exercise” Button to the right of each exercise
       1. If the user has made any changes, two buttons will appear on the bottom of the screen forcing them to either “Undo” or “Save”
  1. Select a Workout Template
     1. User can select from a list of saved Workout Templates
     2. User can swipe left on an exercise to remove it from the list
     3. If the user has made any changes, two buttons will appear on the bottom of the screen forcing them to either “Undo” or “Save”

1. (Workout Screen) Screen During an Exercise
   1. Each exercise will be in a vertical list, and each exercise will have the following
      1. A label that displays the name of the exercise
      2. A textbox across from the exercise name
         1. The textbox only shows if the exercise consists of reps or a distance
         2. The textbox will be editable so the user can change these metrics on demand if need be.
      3. Circles underneath the exercise, in a single row, to measure completions
         1. The number of circles corresponds to the number of sets in the exercise
         2. Each circle can be tapped to perform a function
            1. For reps, tapping the button sets the number of reps completed to the goal number

The user can tap the circle again to decrement the number of reps completed by one each time

* + - * 1. For distance exercises, tapping the circle once will start a timer

The user must tap the circle again to mark the completion of a set and stop the timer

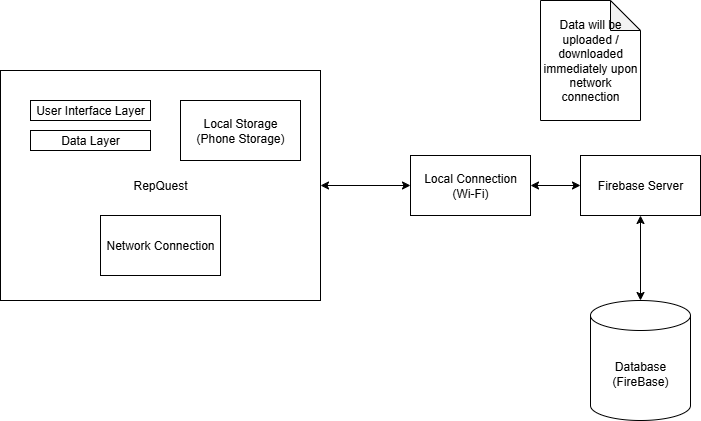
* + - * 1. For timed exercises, pressing a circle will start a countdown timer
        2. Once a set of an exercise has been marked as completed, the rest timer will start at the bottom of the screen
  1. A quit button/back arrow will appear in the top left
     1. A window will pop up asking the user to confirm they want to quit early
  2. Once a Workout has been completed, the results and notes will be saved and stored in the history tab
  3. Once a Workout has been completed, the data will be sent to the Stats and Goals Pages

1. (Workout Screen) Rest Timer
   1. After a set of an exercise has been completed, the rest timer will activate (if the user chose to have a rest period for that exercise), counting down to 00:00:00
      1. The rest timer can be paused/extended by tapping a pause button, or skipped with a skip button
      2. The timer will remain running in the background if the user switches apps.
      3. Upon completion of the timer, a chime and vibration will be made to notify the user to start the next set
         1. The chime and vibration can be individually turned off in settings
      4. Within the rest period, the user will have the option to make any notes about the prior workout
         1. Notes will be saved with the workout in the History tab
      5. There will be a button to stop an exercise early if the user feels they cannot continue the current exercise
2. History
   1. The user will be able to view workouts they have previously attempted in a calendar format
      1. The user can tap a day on the calendar to see the workout details of that day
      2. The user will see any notes they made for that workout in a textbox, otherwise no additional text will appear
3. Stats
   1. Stats will be shown in a vertical list
      1. Will show the most used goal for each exercise
      2. Will show the highest achieved goal for each exercise
   2. Each exercise can be tapped on to view the trends of each exercise
      1. These trends will also be displayed as a chart
4. Goals/Gamification
   1. The user will be able to set their own goals for each exercise that they aspire to reach
   2. Goals are logged upon achievement.
      1. The goal log can be viewed within the stats section.
      2. The goal log will show all achieved, attempted, and in progress goals.
   3. The user will see their experience and level on this screen
   4. The app is designed around a pixel art style game to enhance user experience.
      1. The gamification art style will present a colorful user experience
         1. The style presents a colorful “16-bit overhead” design.
         2. The art style will be similar to “Stardew Valley” or “Habatica”
         3. The avatar creation will have two radio buttons.
            1. Male
            2. Female
5. Settings
   1. Settings will be shown in a vertical list
      1. Toggle for vibration upon a completed rest timer
      2. Toggle for an audible chime upon a completed rest timer
      3. Option to set default unit of measurement for distance
6. Saving to a Database
   1. The app will save all data to a database once per day
      1. Certain actions may cause a save to happen immediately, which include, but is not limited to:
         1. Creating a Workout Template
         2. Completing a Workout
         3. Confirming a change in Settings

# 

# Appendix

## Block Diagram



## Component Diagrams

See NewComponent.drawio.png (GitHub)

See NewComponent.drawio.xml (GitHub)

## User Interface Storyboard

## Figure 3: User Interface Storyboard

For a better viewing experience, see UI Storyboard.png (GitHub: RepQuest/Documents/UI Storyboard/UI Storyboard.png)

## Message Documentation

See RepQuest Message Example.json

## Storage Documentation

