WorkoutBuddy

University of Illinois at Urbana Champaign
CS428 : Software Engineering II
Prof. Darko Marinov
Spring 2014

Authors:
Rafael Angelucci
Tim Madigan
Alex Camargo
Joel Meyer
Weixi Liu
Daniel Wilsen

Table of Contents

Brief Description

WorkoutBuddy replaces current workout apps that are extremely one dimensional and are almost all geared towards a specific style of exercise. These applications often require users to use a set-in-stone list of exercises that cannot be used in a customizable way. Our application also expands on the functionality of the popular pen-and-paper choice that requires heavy personal management to maintain it.

WorkoutBuddy allows users to create their own exercises, workouts, and schedules. Whenever they are at the gym they can easily bring up a list of their predefined workouts and quickly get a layout for easily inputting in their workout data. Afterwards, they are free to share their workouts with their friends through email and track the progress they have made in a graphical interface.

These features provide users a personalized and flexible workout app that replaces other apps and all pen and notebook pads at the gym.

Process

Overview

We worked with a modified version of XP for our process. We liked several of the features in XP and our familiarity with the process from last semester made it a good choice for this semester's project.

Two of the main modifications that we made to XP were to remove test-first development and pair programming. We felt that we would be able to move faster with our progress if we allowed our team members to code independently. Also, because automated tests were required, we felt that since we would be writing them anyway, we could implement tests as we programmed and not before.

Partway through our project, we changed how we committed code to the master branch. We had all team members create their own branch where new features were developed and then required that at least one other team member review the code before committing changes. These changes allowed us to develop faster without the use of pair programming.

Issues

Iterative development posed a small issue because some user stories depended on the functionality of others. While user stories were divided into iterations to facilitate steady progress, this caused a problem when one user story was delayed and not completed in an iteration. The dependant user story was then not able to be started or completed in the next iteration because features it depended on were not functional.

Refactoring was not much of a problem during the lifetime of the project. Whenever a group member identified repetitive code, or code that could have been written in fewer lines, a refactoring was done.

Testing was mainly an issue because of the Android platform that we chose to develop on. When writing Android JUnit tests, you must choose between running the tests on Android emulator or a connected Android device. This was an issue because the emulator runs extremely slow on laptops and some of our team members did not own an android device.

Collaborative Development was an issue for us in the early iterations. At first, our team was only using one branch, which caused a lot of problems when committing code. Several features that one team member wrote would break as soon as another team member committed their code. Quickly afterwards, we moved to each team member having their own branch where they could develop new features and then merge. Although this was an improvement, a few problems arose when we committed directly to the master branch without peer reviewing the changes. Lastly, we moved to requiring pull requests or having at least one other team member look over branch merges.

Requirements & Specifications

Implemented User Stories

- Home Screen
  As a user I would like to be able to open the app to a clean intuitive home screen with a button to view my workouts and a button to create a new one.

- Login
  As a user I would to login to the app and have all of my workouts available on any android device.

- Action Bar
  As a user, I would like an Action Bar across the top of the various screens so I can easily access settings, home, and other important functionality.

- Database Connectivity
  As a user I would like to save all of my workouts, exercises, and progress to a database.

- Create New Workout
  As a user I would like to be able to create a custom workout.

- Create New Schedule
  As a user, I would like to be able to design a schedule for my workouts and have it saved.

- View My Workouts
  As a user, I would like to have a page where all my workouts are displayed and I can select which one I would like to perform.

- View My Exercises
  As a user, I would like to have a page where all my exercises are displayed and I

can select which one I would like to perform.

- Use Workout
  As a user, I would like to be able to assign the time/weight/reps for my selected exercise.

- Track Progress Graphically
  As a user, I would like to be able to track my progress (weight, goals) graphically.

- Share My Workout
  As a user, I would like to be to be able to share a specific workout with others via email.

- Set Up A Set Timer
  User should be able to start a timer that will let the user know when to start the next set.

- Create Exercise
  As a user I would like to be able to create a new exercise.

- Track Workout
  As a user, I would like to add/delete sets in my workout. I would also like to edit reps and weights used.

- Logout
  As a user I should be able to logout of the app and have my saved login info deleted.

- Edit Workout
  As a user, I would like to edit workouts and add or delete exercises to them at any point I wish.

- Delete Exercise
  As a user, I would like to be able to delete exercises I no longer want saved in my exercises.

- Delete Workout
  As a user I would like to be able to delete workouts I no longer want to use.

- Convenient Login
  As a user I should be able to login without typing username and passwords if I have recently logged in.

Implemented Use Cases

- Add existing exercise to a workout
  The user should be able to add an exercise that is already in the database to workout. When the user is creating a workout he will be able to specify what exercises will be a part of that workout.

- Add new exercise to the database
  The user should be able to add a new exercise to the database. When creating an exercise the user will specify what type of exercise it is, the name, and a brief description.

- Add a workout template to the database
  The user should be able to add a workout template to the database. In the workout template the user will be able to specify what types of exercises are in a workout, as well as how many sets and reps will be performed with each exercise.

- Add or delete a set from a workout
  When a user is in an active workout, they will be able to add or delete the set from the workout with the click of a button.

- View results graphically
  On the tracking page, the user can select one exercise from a list of all exercises from a pulldown menu. After an exercise is selected, its data will be plotted on a graph. The y-axis will be set depending on the type of exercise is it (weight, time, or distance based) and the x-axis will be set as time. The x-axis will go from when the exercise was first performed until the present time.

- Allow user to share workout with friends
  A user will be able to share workouts with their friends via email. The workout will be shared as text so that they can share the skeleton of their workout.

- Set a schedule for a workout
  A user will be able to schedule their workouts in advance. They will be able to specify the date to schedule a workout and how many times they want to repeat that workout.

- Set a timer to go off for rest periods between sets
  When the user is in between sets they will be able to pull a timer to time their rest period. The user will input the amount of time they want to rest and the timer will ring once that time has passed.

- Delete unused workout
  When the user views their list of workouts, by pressing and holding down for about a second (long click), an action bar will appear with the option to delete the selected workout. It is deleted from their list and the database.

- Allow user to enter in weight and reps for a set
  When the user is in an active workout, they will be able to click on an exercise and bring up its list of sets. They will be able to input weight and reps for each of their sets

Use case diagrams of system

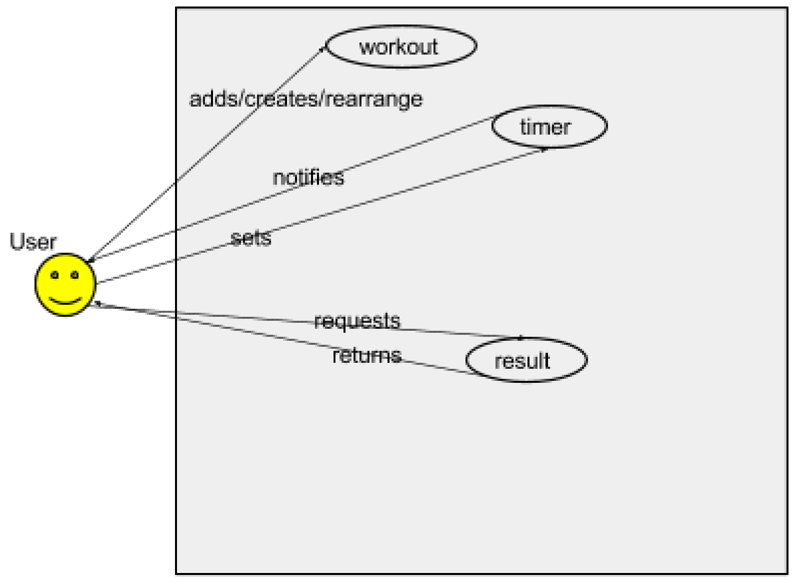

Figure J0

Use Cases Briefs

Actor	Goal	Brief
Athlete	Add existing exercise to a set/workout	Athlete chooses existing workout or set for which to add. Athlete chooses an existing activity name. Athlete enter the static information about the exercise that doesn't change each day and labels one or zero of the parameters variable.
Athlete	Create new exercise to a set/workout	Athlete chooses existing existing workout or set for which to add. Athlete types in the activity name. Athlete enters the static information about the exercise that doesn't change each day and labels one or zero of the parameters variable.

Architecture & Design

We started our design based on a wireframe, which guided the rest of the system. The user's view is dictated on XML files in the res folder. These views are linked to Activity subclasses which respond to user inputs. The Activity classes can manipulate the database via the `AsyncHttpPostWrapper` class inside the `httpRequests` package. The `AsyncHttpPostWrapper` class sends HTTP Post requests with data collected from the user to different PHP files inside the "http://workoutbuddy.web.engr.illinois.edu/PhpFiles/" directory. These PHP Files use the passed in data to execute SQL Command. The PHP Files send a HTTP Response with a correlate SQL responses.

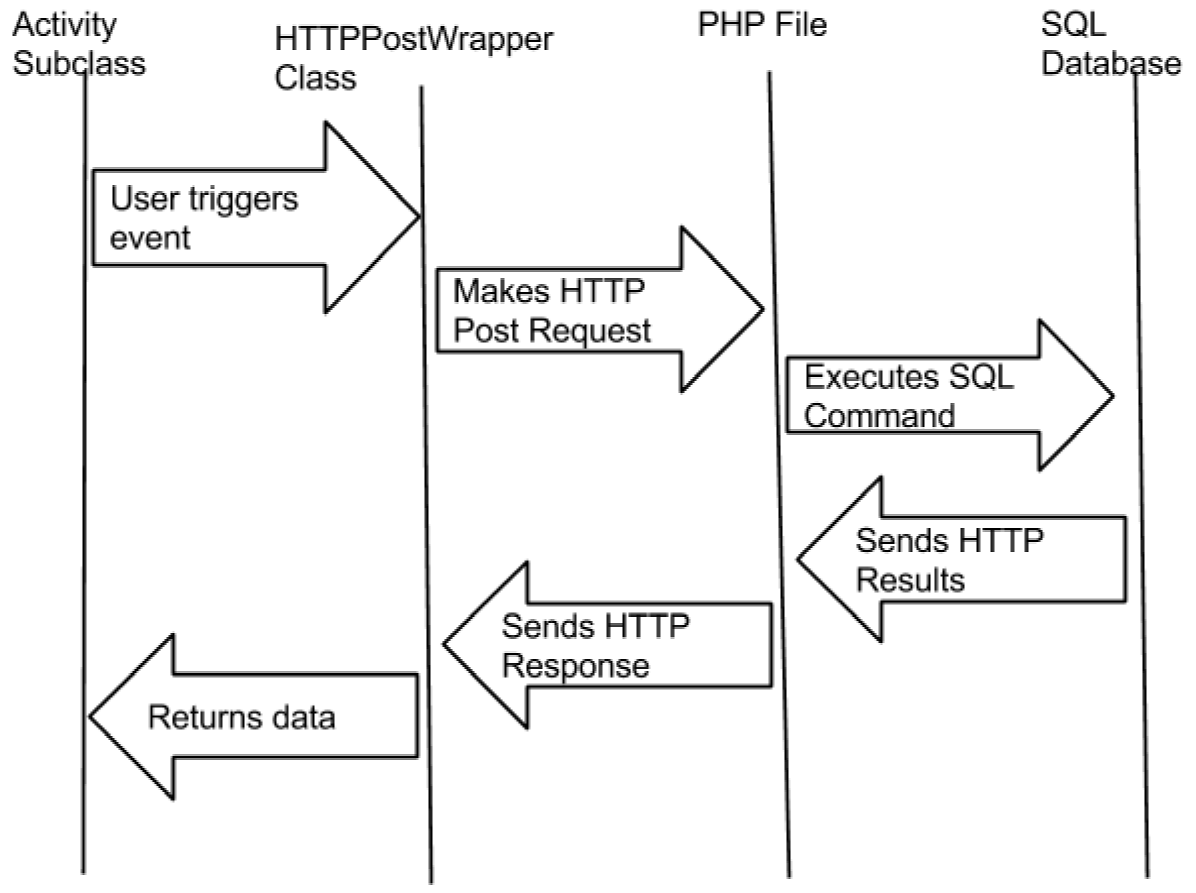
Activity Subclass
HTTPPostWrapper Class
PHP File
SQL Database
User triggers event
Makes HTTP Post Request
Executes SQL Command
Sends HTTP Results
Sends HTTP Response
Returns data

Figure J2

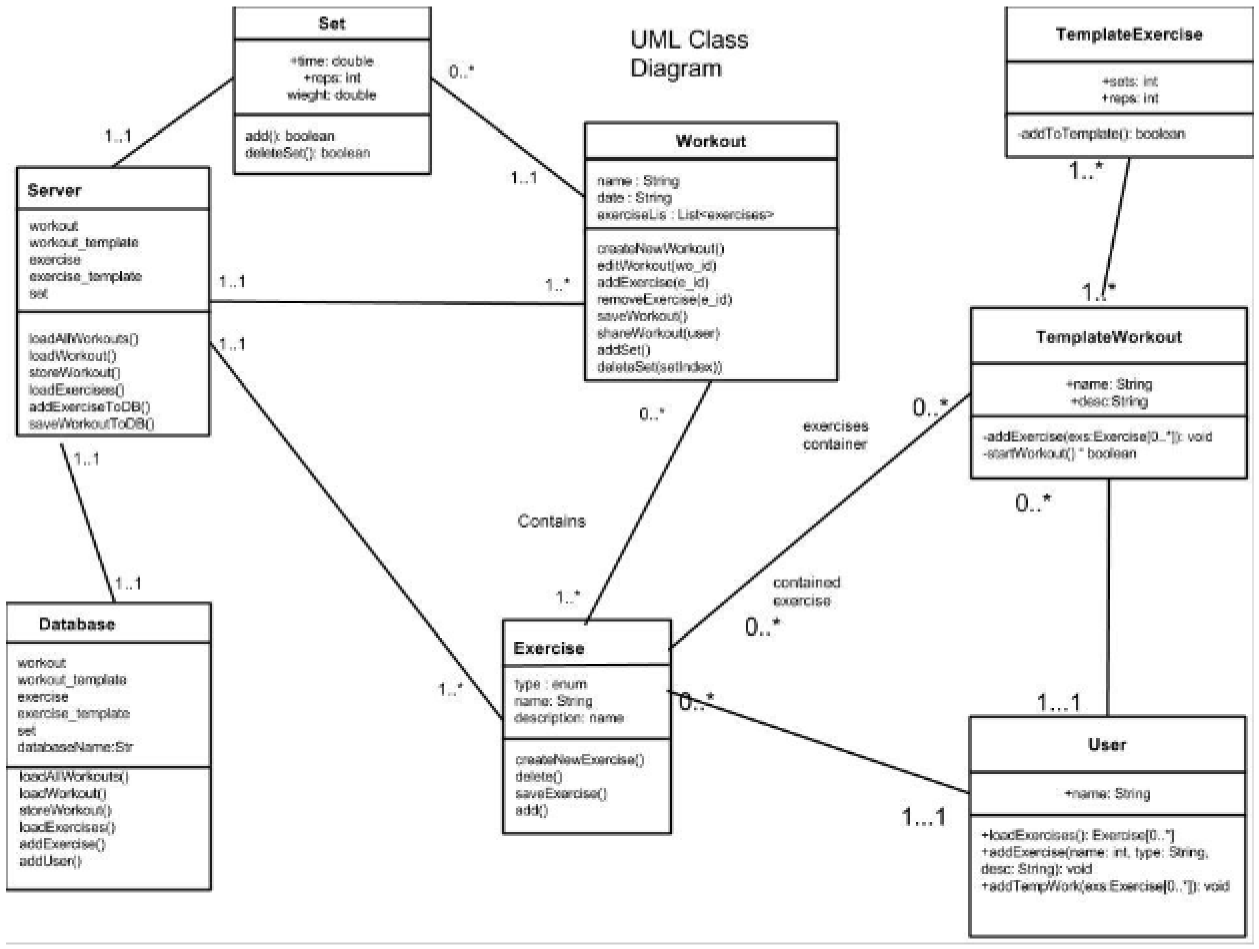

Figure J1

Database Design

Our design for the database contains 6 entities: User, Workout, Template Workout, Set, Exercise and Template Exercise.

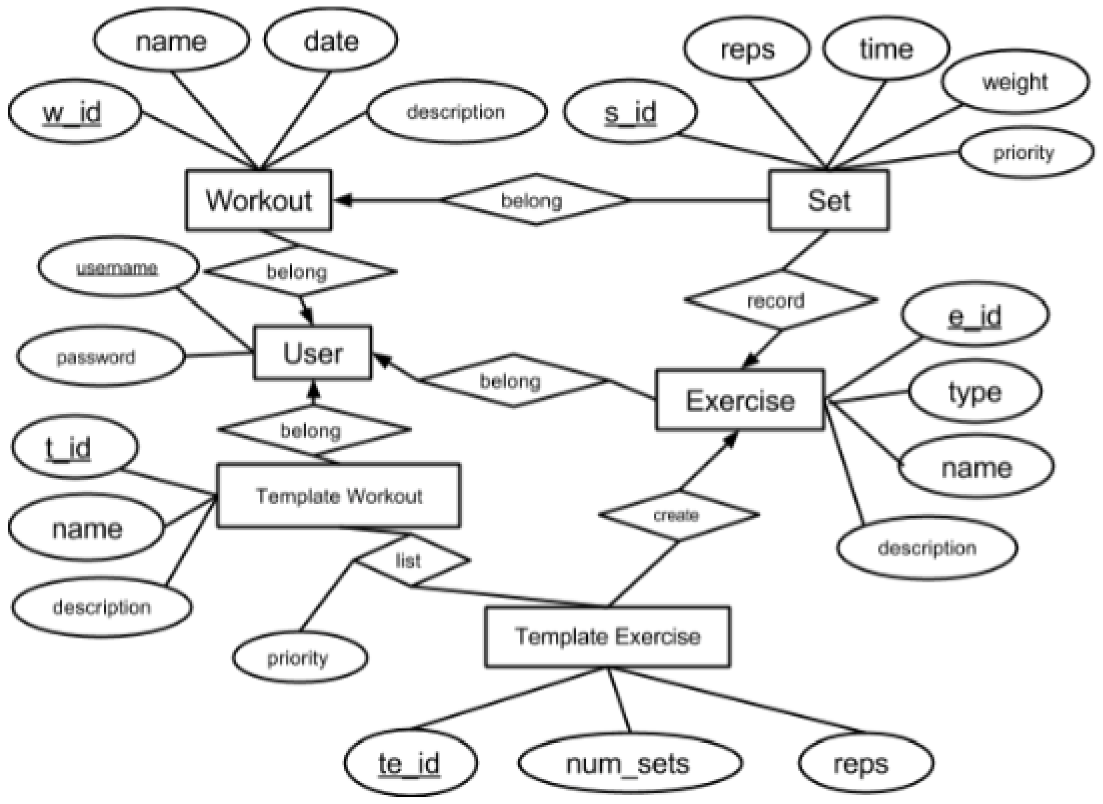

Figure J11

Framework Choice

Developing on Android heavily influenced how we designed our application. Mainly, Android has unique ways that the user interface must be implemented as well as how features are implemented in Android activities.

Most features in Android are implemented through classes called Activities. Android's activities were convenient because we could easily split all of our user stories into a separate activity. This created a convenient way to separate most functionality in a clear way.

Android requires Java as it programming language and this influenced our method of connecting to the internet. Our application required database connectivity and Android did not provide a direct way to connect to our database. Because of this, we created several http post wrapper classes that allowed us to call PHP code in our Android Java environment.

Future Plans

Joel - I have learned Android Development, Fragments, Activities, Android Testing and process. I plan to use this code later as a reference and guide for future Android apps.

Alex - Working on this project helped to teach me about the importance of communication when developing software in a group. With several people all working on the same project code base as the same time, it is important to communicate clearly what you are working on, what you will change and where you are running into problems. Early and frequent communication can help to prevent others from having the same problems and help to keep everyone updated on changes that will affect their code. Moving forward I think the project can benefit from implementing a few user stories that we did not get to. Mainly, allowing users to be able to view and add their friend's workouts directly within the application would expand the benefits of WorkoutBuddy. Users could find ways to improve their own routines by seeing what their friends are doing and also get motivation by seeing their friend's progress.

Tim - I am planning to continue working on this app with Rafael after school has finished. One high level desire I have is to create a more intuitive GUI throughout the whole app. Along with this, I want to add the ability to edit a workout at a date after its creation, update the create work functionality and combine it with the my exercises list, add the ability to delete an exercise, and finish the scheduling of workouts. I want to use this app to schedule, plan, and track my own workouts, so I am planning on creating a finished product of the said functionality with Rafael, as he focused mainly on the backend and I mainly worked on the front end. Once finished, I would also like to make it available on the Android market.

Rafael - I am planning on continuing work on this app after school has finished. My main concern is changing how things are stored on the database because as of right now it takes too much processing for simple tasks. I would also like make sure that this app can be used online and offline with synchronization between the MySQL server and android's built in SQLite.

Daniel - No plans given

Weixi - The overall security of our app can be improved in various ways. For example, currently we are sending along with user credentials each time when we are querying the server. Although the password is hashed, any node along the path where the data packet goes through can see it and modify the query to access user information. We can add a secured layer, namely SSL, and make use of sessions to reduce the risks.