CS2005 Final Project Report for
Activity Tracker Application
Group 2

November 29, 2018

Tyler Strang
Kyle Au
Rebekah Pynn
Michael Adedokun

Introduction

For our group project, our goal was to create an application called Fitness Tracker that enabled users to keep track of their workouts and fitness goals. In Fitness Tracker, a user can create a profile, and upload their data from a file, which in a later version can be turned into uploading from a device. Once their data is uploaded and stored, a user can then view the statistics of all their past workouts. The application can display the total time of the run, the total distance covered, and the average altitude up, and down. Along with being able to view workouts, a user can also view their personal records across a range of dates such as longest distance and calories burned. A user can sort their records, choosing to view records for only certain dates or a certain month. The application has skeleton classes for easy implementation in the future of a friend feature.

Fitness Tracker Application

When deciding how to create Fitness Tracker, we decided to break the app's basic requirements into use cases, to make it easier to implement them as well as make sure Fitness Tracker did everything the client specified. Our use cases were Create Profile, View Data, Edit Data, Adding Friends, Import Data, and Add Devices. These covered the basic features we wanted to implement: create user profiles, import data from an external device, show data in a viewable format, store user's data, and add and store a user's devices. From there we did domain analysis to make sure the software we created was flexible as well as extendable. The domain analysis included a use case diagram, sequence diagrams, a UML domain model. The UML domain model provided the basic framework of the application, which we used to write the classes in our code. All of these documents were changed and updated as we moved through our implementation plans and our timeline and required features changed. In this final release, the Friends and editing data features are not implemented at all, while other use cases are different from the initial conception.

The majority of the data in the app was divided into two main classes: UserApp and Sessions. The user class took care of storing the user's data, including their username, password, list of sessions, and list of devices. The Sessions class handled the data from the user's workouts, and calculated statistics and records. The ActivityTracker class is where the Java Swing design is implemented, the main method is called, and the interactions between the other objects are written. There were two smaller classes, one for handling a user's devices and one handling a user's friends. All the data was stored in text files, with one containing all current user's usernames and passwords, and a text file for each individual user, storing a list of their devices, friends, and session info. In order to display all this data, we used Java Swing to create the interface for the application, and used JTable to be able to display workouts in a way the user could easily view.

Architecturally Significant Design

One of our main focuses when writing the code for our application was to make it flexible and extensible in the future, in case we ran into any issues or further requirements were added. The domain analysis helped in dividing responsibility among the different classes, and applying design priniciples allowed us to write code that was easily modifiable. When writing our code, we attempted to encapsulate what varied, keeping things that often changed away from the things that didn't. The User class holds device objects, friend objects, and session objects, but each of these

classes are modularized enough so that they can interact with each other, but changing the way one class behaves doesn't break another class.

GitHub/Point Distribution

https://github.com/Tylemaster/ActivityTracker.git

- Tyler Strang - 16
- Kyle Au - 8.5
- Rebekah Pynn - 8.5
- Michael Adedokun - 7