LET'S | LIFT

by Mark DeRose

Table of Contents:

Overview:

A simple application used to track progress in workout routines.

Requirements:

Milestone 1:

My initial scope would be one that can provide three points of functionality:

1. Users can create, add, and manage their own exercises (different types of lifting or cardio routines) to their account.
2. Users can create, add, and manage their own workouts which consist of several different exercises.
3. Users can track their workouts and see how much they accomplished over time.

If I'm feeling more ambitious and time permits, I would like to add some additional functionality:

- Add a BMI calculator for users to calculate their body mass index.
- Create default routines that users can make use of rather than having to create their own from the application.
- Populate a chart that breaks down amount of workouts completed over a period of time (per week/month/etc).

Milestone 2: I am still intending on using the scope above. But while I am getting my program off the ground I'm only using lifting exercises at the moment (instead of lifting and cardio, as having to split the class will make my database more complicated, and as I'm just starting to learn the framework I want to keep things as simple as I can starting out).

Milestone 3: I am setting aside the additional functionality for my submission and only hoping to complete 1-3 above. Hopefully I will be able to implement those additional features in the future, but given that I'm dealing with some newer technologies I want to keep the program simple until its working with basic functionality.

Design - Table Layout:

Initially I created the following schema (using a simple tool at https://erdplus.com/) to ensure that my database maintains third normal form.
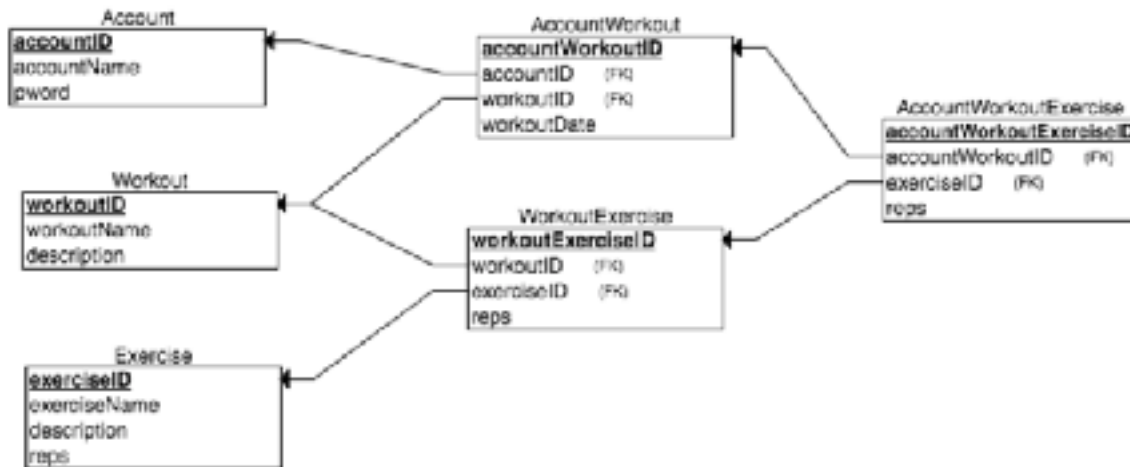
However, after reviewing the lecture, notes, and additional information about JPA/Hibernate/Spring, I think it may be easier to implement these tables using Entity mapping rather than a one-table-per-class structure. We'll see how it goes.

I am going to use MongoDB hosted at mlab.com to handle my user account table, and table per class set up via Oracle SQL on DePaul's server to handle all of the workout and exercise information.

Design - Code:

      I laid out my package structure as shown on the right so that I can have a clear separation between classes/interfaces/repositories and my service. I know we're going to be covering MVC at one point so hopefully it won't be too much of a hassle to refactor when the time comes.

      As far as actually writing the code, it wasn't too difficult until I hit having to communicate with the Oracle DB. The MongoDB was very simple and painless. Oracle however, had issues with Maven dependencies followed by several regarding connectivity to DePaul's server, along with tables being populated with null values. I'm still having some problems with the relationship between my Workout and Exercise tables but those will need to be worked out as I keep moving.

      During Milestone 3, I have added a "workoutDate" variable to my Workout class (as we need users to be able to track the actual date of their workouts). Embarrassingly enough, it took a while to debug an error with Spring/Oracle connecting because I had named my variable "date", which is already the name of an SQL data type.

Design - Mockup:
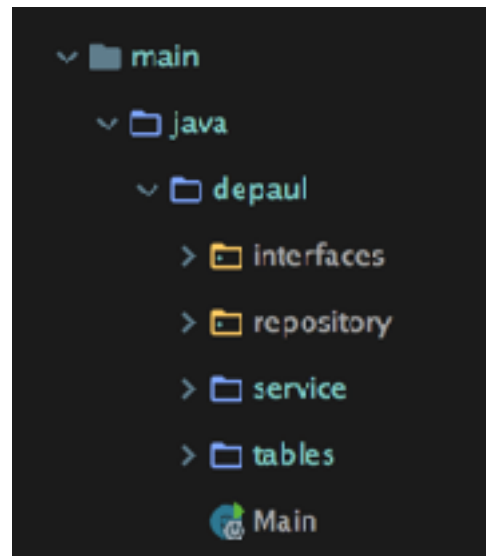
Link: https://app.moqups.com/mderose25@gmail.com/XHOTG49VIM/view/page/aa9df7b72 (please let me know if you have any issues with accessing the moqup).

    I created my GUI mockup using Google Moqups. My main concern with creating the mockup was that I understand the flow and potential issues I can run into with interaction between my GUI and my database. It has a very simple layout with a navigation bar along the top.

Each workouts and exercises page has the ability to view, create, or delete the respective type. A user must have exercises created before being able to add them to a workout (though if I have time, I may add some generic lifts so that users don't need to create them on their own). Additionally, I have simple pages for "about" and a landing homepage merely for accessibility.

Design - HTML:

I began setting up templates for the actual HTML implementation of my GUI. The layout of my templates folder is shown on the right.

I used HTML5, CSS, and Bootstrap.css to help style the page. So far, the only pages that have more than a navigation bar and background are home and login (pictured below). I obtained a picture from https://unsplash.com/ (which hosts free high-res images) to use as a background for my application.
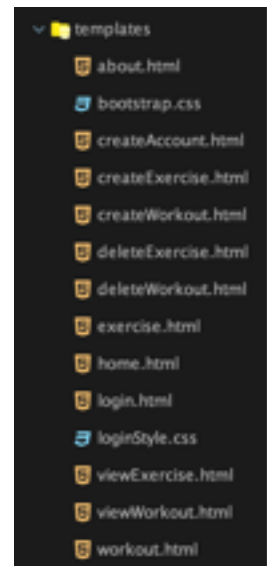
Home:
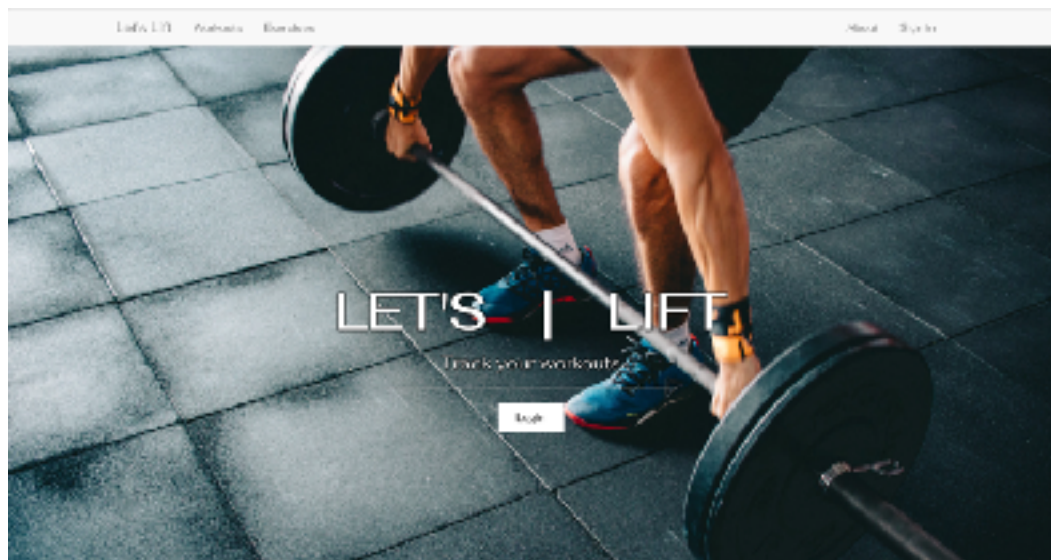
Login:
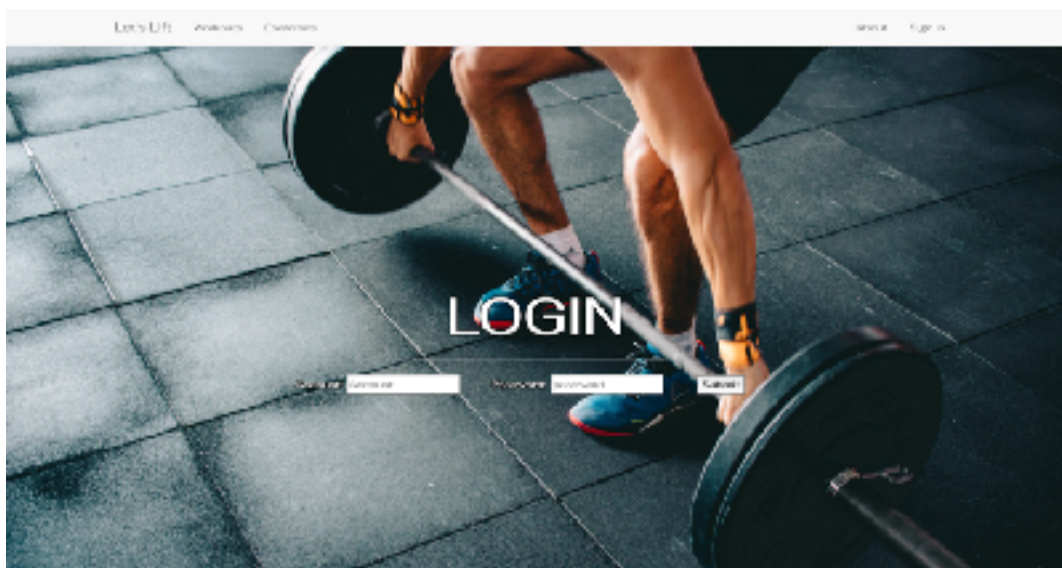
Milestone Feedback and Changes:

Milestone 1: I accidentally included the two JAR files for JUnit testing. I was setting up IntelliJ for the first time and trying to add a path just to make sure testing was set properly and forgot to remove them before submission. I changed the title of my project from "Workout Tracker" to "Let's Lift!" simply because "Workout Tracker" is very bland and boring.

Milestone 2: No feedback given other than good progress. However, I have made some small changes to the layout of my workout and exercise classes.

Decision Matrix:

Problem	Decision	Alternatives	Rationale
IDE	IntelliJ	Eclipse, NetBeans	IntelliJ is very popular and I have wanted to get more experience with it.
Code Repo	Github	None	I'm still getting the hang of Git and the only way to get better is to keep practicing.
Database	Oracle/Mongo	HSQL	I have experience with Oracle and Mongo seems very simple, I will use Oracle for my user workouts/exercises and Mongo for user personal data.

Appendix:

Setup instructions: src/lib/com/oracle contains the ojdbc.jar that we discussed on the forums. Must be added externally to run.

Milestone 2:

Code run-through:

Maven build:

Milestone 3:
Code run-through (minimal updates to Java code in MS3):

Maven build: