Checklist Items:

Programming:

The program is separated into four main files, the app, the user router, the workout router, and the data file. The app sets up all of the basic functionality and error handling that the server needs in order to interact with incoming clients and directs them to their desire paths. The user router handles all of the login and data manipulation that goes along with being an user in the program. It also handles authentication of the user so that we know whether or not the user is logged into the program and allowed to access various parts of the program. The workout router handles all of the data manipulation and delivery of the workout information. Between editing deleting and adding new workouts or changing parts of workouts these methods start to get complex! Lastly, the data file defines our schema's explicitly and clearly and sets up databases for the information of those schema's to be entered as a collection.

Building the data model and running unit tests were definitely difficult tasks for us. It took us quite a while to understand how to implement our data model correctly and additionally learn how to implement the QUnit tests. While both of these items seem fairly trivial now, but had some unfortunately steep learning curves. Testing wasn't terrible to implement, but it took forever to learn how to get it to the tests to run correctly, actually calling the file correctly. The API itself is organized well into clear methods with specs that allow for an easy understanding of what is going on inside of the API as a whole.

Design Challenges:

We had several issues and challenges to overcome in the design of this program; definitely in terms of implementation, but also in terms of designing the architecture of the program.

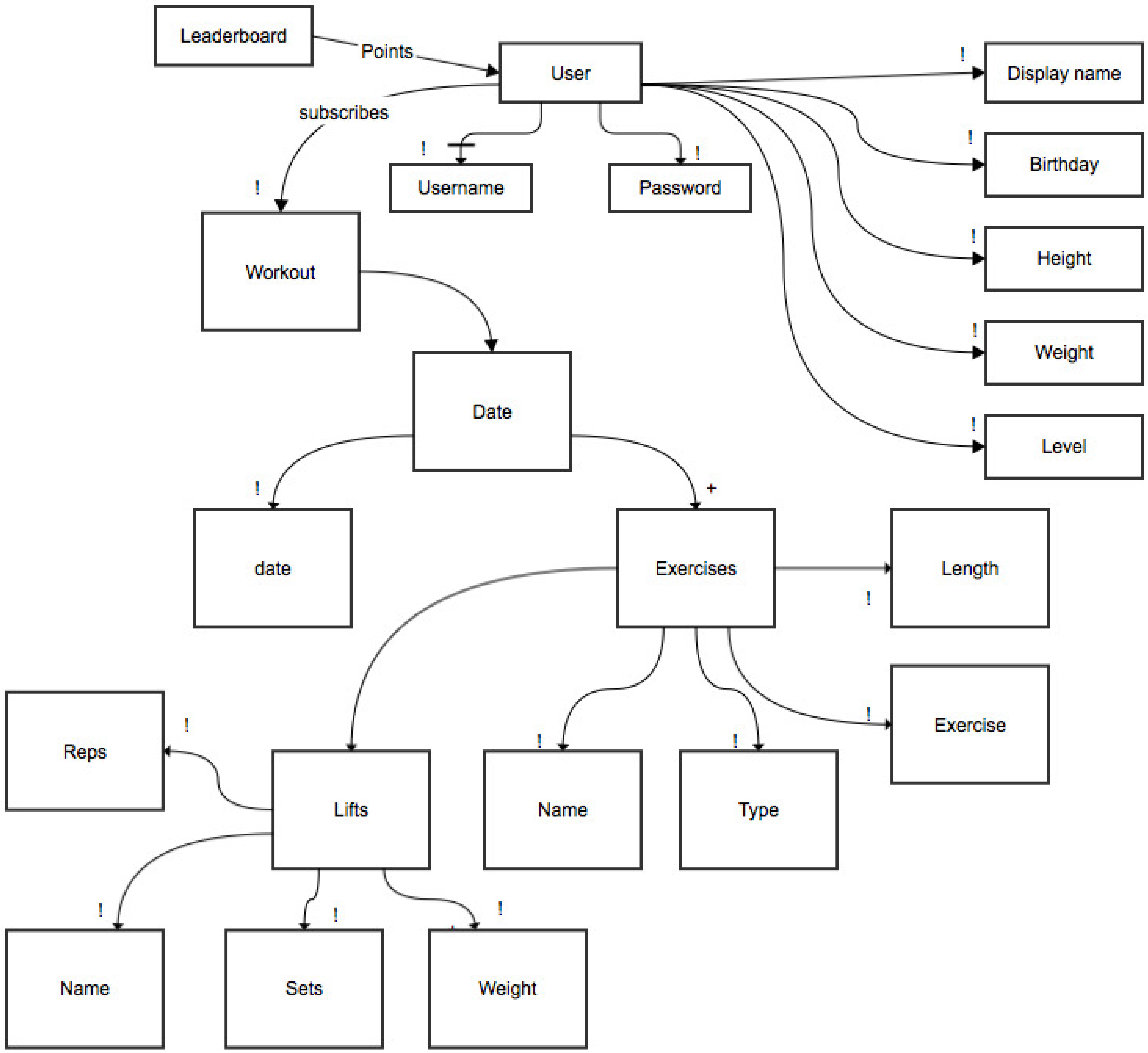
1. Error handling was a big one to handle in terms of designing this program. First, in figuring out what errors to handle and what to write up explicitly in the spec's. You can't handle everything but there definitely should be some error handling to make it easier on the user. One case of this was in updating the user's information and handling the cases where the user may not be explicitly entering information for every field of their user object. By updating the object in the database, without all of the parameters the corresponding fields in the database would go null. We had a couple of different options in doing this, one was to check to make sure we had a non null value for every parameter, but we decided to go with an easier solution, which was to push that choice to the client and have the inputs of the forms be generated dynamically upon the user wishing to edit.
2. Designing our data structure was definitely a big challenge for us. Because of all of the information we wanted to display if became increasingly difficult to put together a data structure that was easy to understand, build and operate on. We had a couple of different choices in how to organize our data structure and ended up going with a relational model for a couple of reasons. First, having a document structure became difficult to operate on, store information with, and we decided was subservient to a

relational structure. Because of the highly referential structure of each workout we had to be build each part separately and then populated into its "parent" object, but that was better than having to move down a nested structure to access any data that we wanted. Even still the structure that we have now is still cumbersome, but necessary for the amount of information that we are trying to display. Our design definitely changed a lot from when we first started.

3. Lastly, I think testing was a big design challenge for us. Maybe it wasn't intentional, but we found testing to be very difficult at first. Figuring out how to do an http request was increasingly difficult the more we tried frankly. Once we did figure out how to get an ajax request to work properly testing started to move along, but there are a million ideas on the internet about this and none of them are very clear. It would have been nice to have a heads up about how to do this at the start of the project.

The location of our app is at:
http://nickugunnisonstahdirkliftmhttp://nickugunnisonstahdirkliftmate-dirkstahlecker.rhcloud.com/ate-dirkstahlecker.rhcloud.com/

If you want to run the Qunit tests go to the url:
http://nickugunnisonstahdirkliftmate-dirkstahlecker.rhcloud.com/Testing.html


Leaderboard
Points
User
subscribes
Display name
Birthday
Height
Weight
Level
Username
Password
Workout
Date
date
Exercises
Length
Exercise
Lifts
Name
Type
Reps
Name
Sets
Weight
!
+