

Nick, Grant, Dirk

Project 3, Part 3

Updated from Part 1:

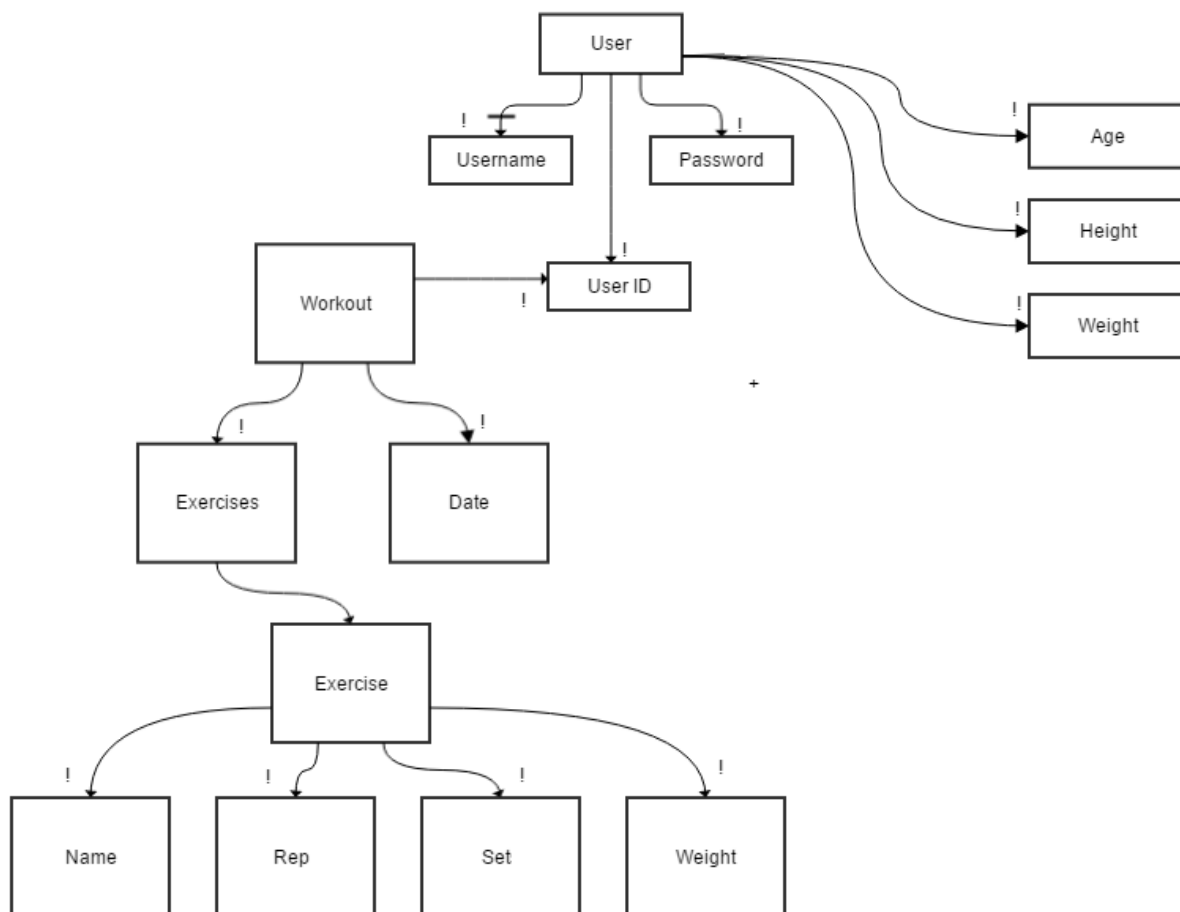
Purpose:

- Track workouts
- Display and compare workouts

Concepts:

- Workout – fulfills tracking and displaying workouts
- Exercise – fulfills comparing workouts

Data Model:



Part 3:

Challenges:

The biggest challenge we had was the one that confronted us immediately: our part 2 implementation of the API, and the data model that it implemented, was not sustainable. It was too complicated for our purposes within this app, and our understanding of http requests and the role they would ultimately play in the communication between the API and UI was lacking. Consequently, we ended part 2 with an API that functioned, but was very large and difficult to modify and test. Specifically, the data model was so complicated and the dependencies were so interconnected that it was nearly impossible to work with. We had to completely tear it down and start over at the beginning of this project to allow for the UI to effectively utilize it.

Relatedly, another challenge we ran into was determining what the relationship between users and workouts was going to be; the core problem here was deciding what a workout actually represented. We could have represented a workout as containing multiple days and multiple exercises per day, representing a specific “workout plan.” The other option was to have a workout be confined to a single day; it would still contain exercises inside, but a workout plan would no longer be a concept in the design. We decided upon the latter option because it enabled a simpler implementation and a workout plan was not a central concept to the design. Rather, the actual exercises completed were more the main idea of what we’re trying to accomplish with this app, according to our outlined purpose.

One of the challenges we faced was determining how to populate the actual pages with data from the database. We started out using embedded JavaScript, as we did with project 2, because of its simplicity and how straightforward it is. What we realized, though, is that the separation of client and server didn’t allow for this approach. We needed to use Handlebars, because this framework allows us to maintain the separation between the client and server and render the dynamically created html pages with data pulled from the database.

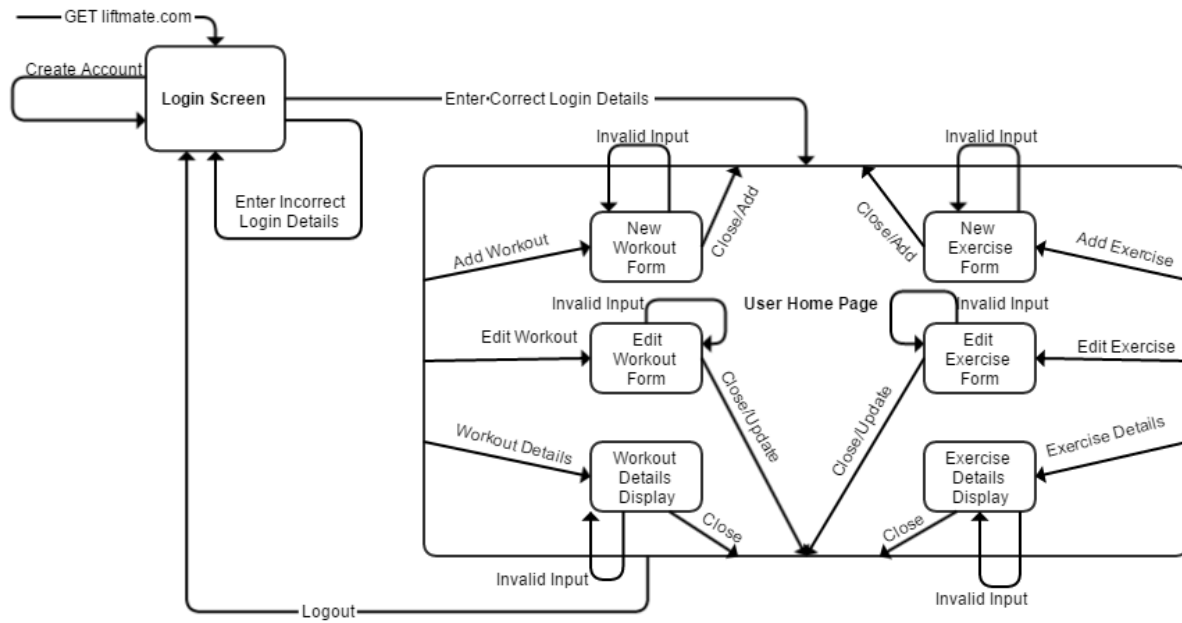
Lessons Learned:

The biggest lesson we learned is that of planning ahead. We created our data model and went through the design process as required by the project, but we didn’t realize the full implications of our design decisions until we were in the middle of the implementation. There’s so much complexity, deceptively much at times, and it’s difficult to wrap one’s head around the magnitude of problem when implementing an app like this. But thinking through all the specifics of the implementation will save so much time in the long run. For example, we didn’t consider how to actually link the API and the UI together until we had already completed the API and were in the process of coding the UI. It turned out that our API was returning incompatible data, and our implementation of the frontend was not capable of receiving the data we needed to receive. We will definitely remember this for the future and design each element so that it interacts nicely with the rest of the app.

We also ran into problems in the initial design of the data structure. We started out having too much of a nested hierarchy. I think we were coming from a very object-oriented perspective when we first started out, as we tried to insert hierarchy whenever possible. We initially had Users, Workouts, Dates, Exercises, and Lifts, each as separate models. The Workout was an encapsulation of Dates, Exercises, and Lifts, and each of these contained an internal nested structure. We realized how unnecessary this structure is in Mongo. When we updated the design, we ended with only three models and no hierarchy – Workouts contain a reference to a User and an array of Exercises, and this is the only linkage. But it fulfills the same requirements as the previous model while being vastly simpler. Next time, we will put more consideration into the data model to condense it down as much as possible and make it much simpler to code.

Overall, doing this project taught us the real depth and scope of a web application. We will utilize this knowledge and understanding of the full process to influence future projects from the very beginning now that we have a better understanding of how the whole thing comes together.

Wireframes:



Login page:

Browser window: Login Page





Address bar: http://

Existing User? Login!


New User? Create an Account!

User page:

User's Home Page



http://



My Workouts

Log Out

Add Workout

Workout1 Name (date)

EditDelete

Exercise1: infoEditDelete

Exercise2: infoEditDelete

Add Exercise





Workout2 Name (date)

EditDelete


Add Exercise

Add workout:

User's Home Page



http://



My Workouts

Log Out

Add Workout

Date

Workout1 Name (date)

Exercise1: info

Exercise2: info

Workout2 Name (date)

Edit workout:

← → × ↗

User's Home Page

http://

Q

My Workouts

Log Out

Add Workout

Workout1 Name (date)

Edit

Delete

Date

Exercise1: info

Edit

Delete

Exercise2: info

Edit

Delete

Add Exercise

Workout2 Name (date)

Edit

Delete

Add Exercise

Add exercise:

← → × ↗

User's Home Page

http://

Q

My Workouts

Log Out

Add Workout

Workout1 Name (date)

Edit

Delete

Exercise1 Name: info

Edit

Delete

Exercise2 Name: info

Edit

Delete

Add Exercise

Workout2 Name (date)

Edit

Delete

Add Exercise

Name

Reps

Sets

Weight

Submit

Edit exercise:

← → × ↗

User's Home Page

http://

Q

My Workouts

Log Out

Add Workout

Workout1 Name (date)

Edit

Delete

Exercise1 Name: info

Edit

Delete

Name

Old Name

Reps

Old Reps

Sets

Old Sets

Weight

Old Weight

Submit

Exercise2 Name: info

Edit

Delete

Add Exercise

Workout2 Name (date)

Edit

Delete

Add Exercise