CSM Athletics
Final Report

Mason English, Hunter Arnell, Caleb Micho,
Kyle Voris, and Tyler Blount

June 23, 2017

Table of Contents

1.0 Introduction

1.1 Client Description

The Colorado School of Mines Athletic Department trains its athletes to be in peak physical condition. These athletes compete in varsity level sports representing the school. To help maintain the athlete's' peak performance, head strength training coach Trevor Florendo designs custom workouts for all athletes, tracks athlete lifting data, and maintains a safe environment for athletes to train.

1.2 Project Description

To track athlete progress, data sheets are hand entered into an Excel spreadsheet that keeps track of each athlete's workout and performance. This process is inefficient and time-consuming for staff members to maintain with a growing athletics department. In addition, because each sheet is in an athlete's possession for two weeks at a time before being returned to the coaches, there is a risk for data to go missing due to the sheet being damaged or lost by the athlete.

Our job this field session was to create a system that maintains these records with minimal upkeep for staff members. In addition, the site also allows coaches to create workouts for individual athletes, whole teams, and customizable subgroups. Workouts should be reusable and able to be assigned to multiple days at a time for the same group. The website also requires a leaderboard to promote competition between athletes in the same or different sports. Finally, the website should be user-friendly on both mobile and PC devices as well as fit the current IronDigger design currently used by the Athletic Department.

2.0 Requirements

2.1 Functional Requirements

Coach/Administrator View

- Coaches can create workouts with lifts of different sets and repetitions
- Coaches can schedule workouts to specific groups or sports teams over the desired date range filtered by day of week
- Coaches can create custom groups of athletes
- Coaches can download excel spreadsheets of sports teams lifting progress each month
- Coaches can download a formatted word document of lifts assigned to an Individual group for a date range desired
- Coaches can create custom lifts

Athlete View

- Records each athlete's lifting history
- Maintains records of workouts for all athletes
- Allows athletes to view their records on a profile page viewable only by that athlete
- Allows athletes to view their lifting progression

Both Views

- Has a leaderboard for each exercise filtered by sports team (all an option)
- Utilizes Mines Multipass login capabilities

2.2 Non-Functional Requirements

- Must be easy to view and interact with on mobile devices
- Has a well designed and easy to read web page layout
- Capable of using a Mines server to host the website
- Website name goes with existing IronDigger design scheme

3.0 System Architecture

A multitude of working parts needed to come together to construct the IronDigger website. Figure 1 is a high-level diagram of the website architecture.

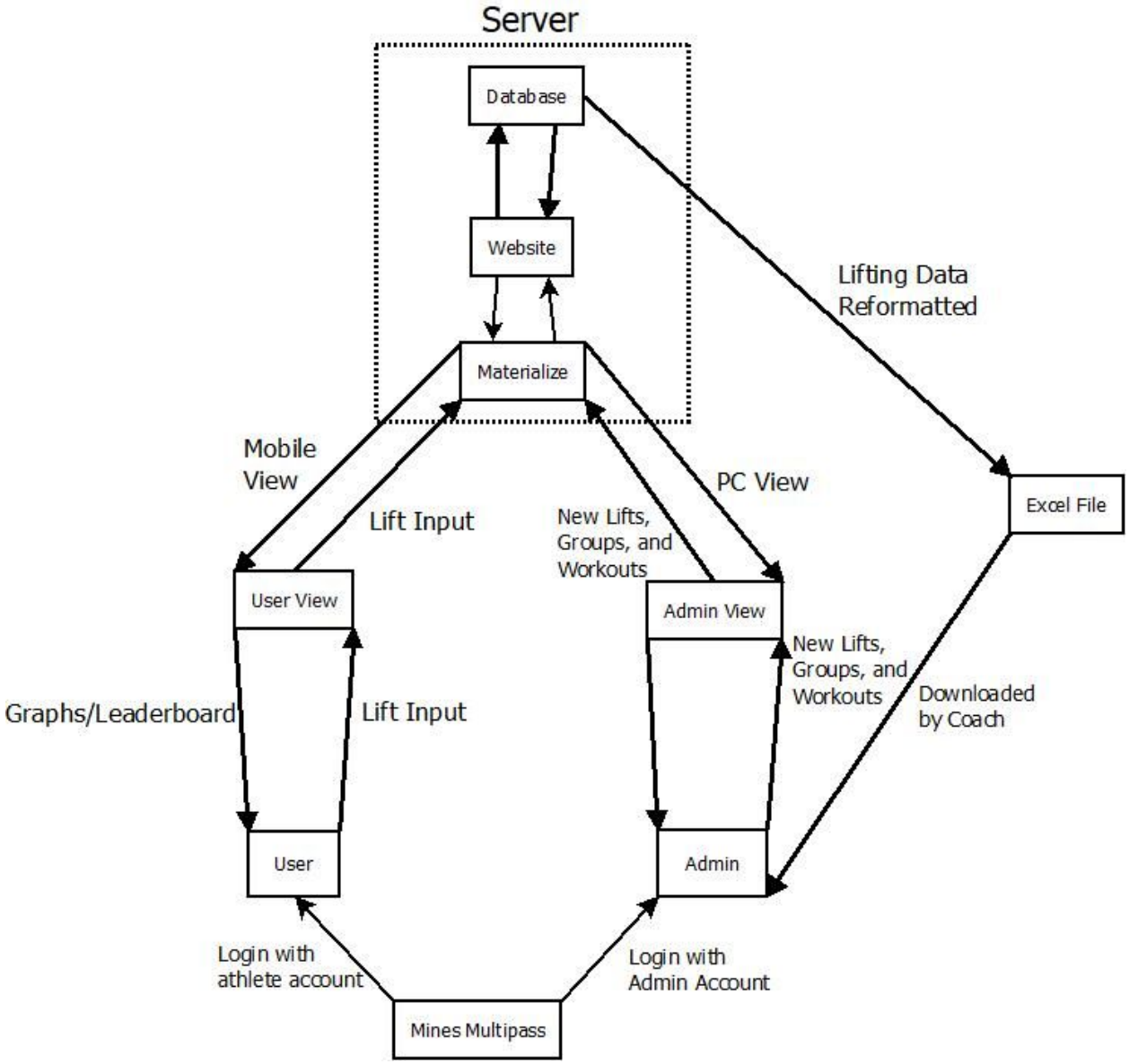

Figure 1: A graphic layout of the system architecture layout

3.1 Multipass

Multipass is a service run by Mines CCIT department to help authenticate users and guard against unwanted access. We utilized Mines Multipass system to authenticate users when they login. When a user logs in to the website the server will redirect the user to login with their Mines Multipass. Upon a successful login, they will

be redirected to create a profile if it is their first time, or to their workout page if they already have a profile.

3.2 Laravel Version 5.4

We used the Laravel 5.4 framework for building our website. Laravel is a free, open-source PHP web framework intended for the development of web applications. Laravel follows the model–view–controller architectural pattern and had an easy to implement user permissions feature. We used the authentication system in Laravel to control what admins and users can and cannot view, as shown in Figure 1 with the separation of admin and user. We used Laravel's Scheduler to create an automatic task that pulls each user's lift data from the database to a CSV for the admin to download and store on their computer.

3.3 Materialize

We used Materialize to design the website with both mobile friendly and desktop friendly viewing. Materialize is a responsive CSS framework based on Google's Material Design Language. Material Design focuses on clean, flat elements, with a simple set of colors and smooth animations on interactive elements. This makes it easy to use and navigate mobile pages but is also responsive enough to be used on the admin portion for desktop viewing. Thus meeting the requirement to create a website that would interact with the user in the most friendly way possible.

3.4 MySQL

We used MySQL for the database to store all the lift and workout data. MySQL is an open-source relational database management system that is currently owned by Oracle Corporation. MySQL is commonly used with mines servers and allows the website to seamlessly be served in the future. Furthermore, MySQL follows PostgreSQL syntax and allows for easy query command to CSV.

4.0 Technical Design

4.1 Database Schema/Design

Our project involved creating, coordinating, and displaying many different lifts for multiple groups. We determined creating a database to store this information would be the easiest and most maintainable option.

Early on, we decided on using MySQL as our database management system. The main influence in this decision was Mines servers historically having used MySQL for other sites and projects. Therefore, we felt using MySQL would make it easier to integrate with existing systems and more likely that our site would end up being served by the school. MySQL also worked well with the Laravel framework in sending and getting data.

We started with a rough idea of what we needed for the site by examining how the daily workouts go: athletes do lifts, which are grouped in workouts, which happen 3-5 times per week for 2-4 weeks at a time. There was, however, a lot of variance in a single workout, a single week, and a single workout period, which meant that we needed a flexible method for assigning lifts to athletes on a given day. We decided on having a `lift`, `workout`, and `schedule` table, with a `workout_to_lifts` relation (Figure 2).

A lift is comprised of description of the lift and the lift name itself, with the added attribute of whether it is being tracked. Workouts are frequently performed multiple days, but the number of sets and reps usually changes each day and each workout period. Thus, we decided to leave that out of the workout table and instead include it in a relational table, so that different workouts can contain the same lifts, and the same workout can contain lifts with different numbers of sets and reps than another instance of that workout. Workout instead included only a name and is related to a list of lifts that it contains. Schedule is used to relate a workout to the date that that workout occurs on, as well as to a group that will be performing the workout.

On the athlete side, the situation was much more static than with the lifts. Athletes are typically involved in one sport, and are in only one subgroup in that sport. The subgroups may change, but the changes will be very infrequent. Therefore, we decided upon a `user` and `group` table, with the `user_to_group` relation (Figure 2).

Users input their max for various lifts, which are stored and used later on to calculate lift weights, show trends in the graphs, and list their name on the leaderboard. Users are typically in a single sport, though they can be in multiple. They will also be a part of a sub-group in a sport, though this is not required. Because of the nature of this, we decided that rather than having a separate sport table, we could treat sports as groups. We would instead create a parent attribute in the group table, where sports would be blank and sub-groups would have their sport. This made it much easier to display the sports and groups together when the admin schedules a workout. Users and groups are related in another relation.

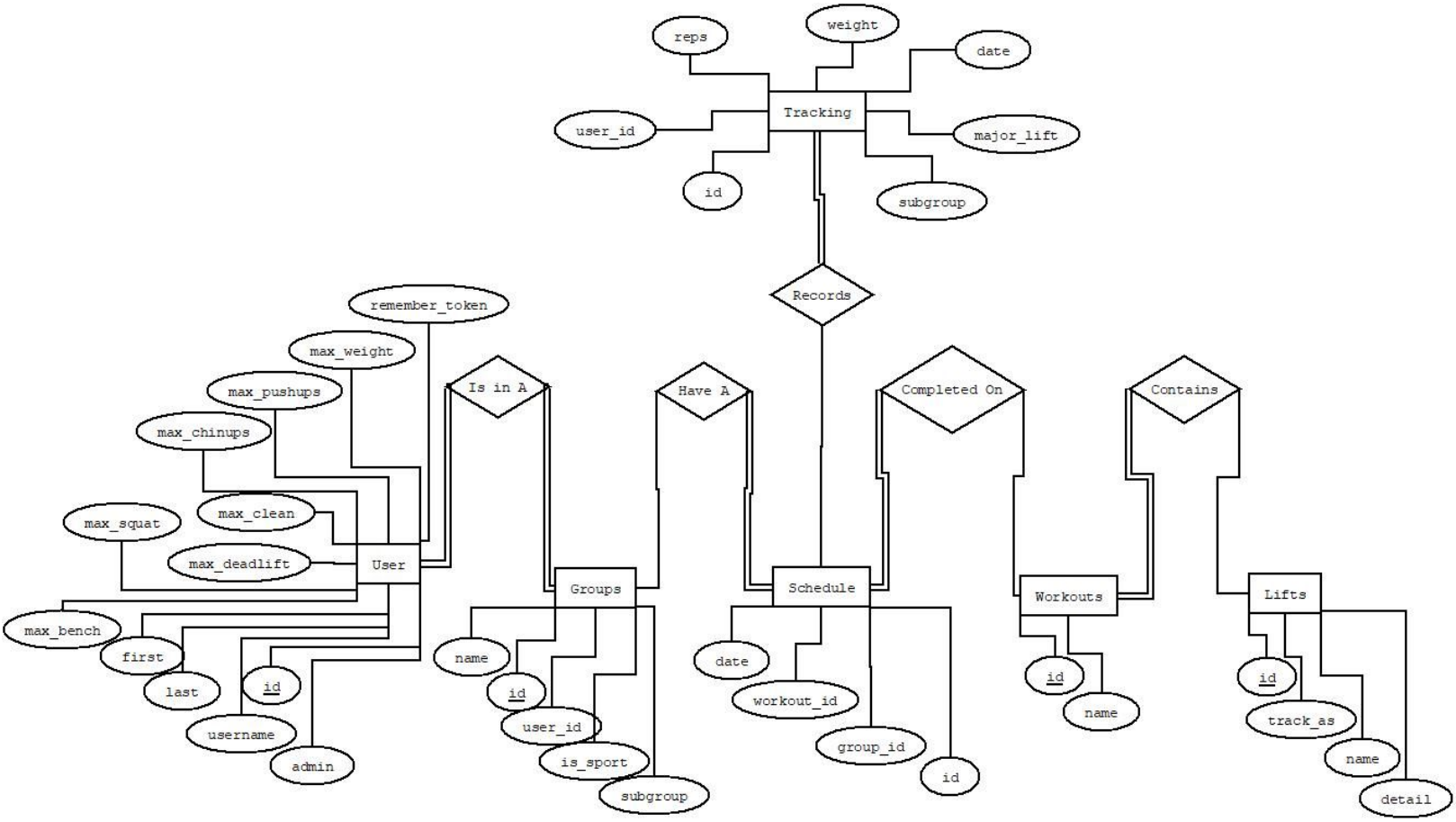

Figure 2: A graphical representation of the database schema used

4.2 Lift History Visualization

An athlete's lift progression is displayed in both a graph and a table (Figure 3) format to provide information on improvement over time and exact dates, weights, and

reps. This was achieved by using D3.js, a Javascript document object model framework for visualizing data.

D3.js contains methods to convert JSON formatted data into SVG, HTML, or CSS elements, allowing for the creation of data-driven documents. Using Laravel's query builder, a copy of the MySQL database entries for a single athlete are encoded into JSON, then fed into the D3 framework. Once there, an anonymous function is run on the data to visualize it.

date	weight	reps
Tue Jun 06 2017	165	8
Wed Jun 07 2017	145	13
Thu Jun 08 2017	155	3
Fri Jun 09 2017	118	1
Sat Jun 10 2017	179	6
Sun Jun 11 2017	101	4
Mon Jun 12 2017	151	6

Figure 3: An example of the graphs and tables created by D3.js for tracking athlete progress

5.0 Design Decisions

5.1 Split Views

Several design decisions made throughout the process are explained in this section. One of the design decision we faced, during the creation of our site, was the separation of the admin view and the student view. Our site was required to work on mobile, as athletes may be entering their data while still at the gym working out. However, we also had several features, such as group creation and workout creation, that would be difficult to implement in a visually appealing way on a mobile device. In addition to the visual aspect, some of these functionalities favored being on a PC over a mobile device for usability as well. To alleviate this problem we decided to make the administrator view less mobile friendly and assume that its functionalities are always accessed on a PC. The students can be reasonably assumed to always be using a mobile device because the environment where they will most likely be using the site is a gym. For any functionalities that overlapped, for example, the leaderboard, we chose to make it mobile friendly as that will be easier to read on a PC than a PC version of the page would be to read on a mobile device.

5.2 Lift History Page

To accommodate the client’s request to store all user’s lift data, and stay within our capabilities of data storage, we decided to create a page that displayed download links to CSV's of user’s lift. This page has a section for each sport and has a CSV download button for a month of lifts that corresponds to each sport. This decision was made based on the fact that Mine’s has over 500 athletes typing in at least two lifts nearly every day. This creates a very large database table, and furthermore, the client has no experience communicating with a database. Thus, we chose to store old lift data in a CSV, the client’s preferred file format. This way, the client can download all the lift data by user, create wanted visualizations, and keep for as long or short as desired.

5.3 Website

Originally, the client requested that we make a mobile app for the athletes to use to enter lift data and him to view it. We began to discuss with the client why this was needed and what functionality/features he wanted. It quickly came to light that an app would not be the best. Thus, we decided to create a mobile-friendly website. This way, the client can assign workouts and view athlete's progress on a desktop as well. This

will be much easier than if done on an app. We then can get the desired access from a phone for the athletes, by the mobile-friendly design.

6.0 Results

6.1 Summary

Our project has been thoroughly tested and is currently working beyond the base scope that was specified by the client. During the project, our testing was done manually. Since our project was a website, a large portion of our testing was trying to get the features to work properly on both a mobile view and a PC view. Beyond that, we made sure that there could be no entries into the database that would create the possibility of it breaking and that the login page worked properly to direct the users to the proper page and restrict anyone who didn't have access. In addition, algorithms were tested separately to make sure that they worked properly before they were implemented into the website.

6.2 Implemented Features

- Multipass login
- Profile for each user
- User ability to track their max in four lifts
- User ability to view graphs of lift progress
- User ability to view current lifts
- User ability to enter weights lifted during workout
- User ability to view a leaderboard for major lifts sorted by sports
- Admin ability to create and edit groups
- Admin ability to create and edit workouts
- Admin ability to create custom lifts
- Admin ability to create a schedule of workouts
- Admin ability to download workout data for each month
- Admin ability to create and download a formatted workout plan

6.3 Future Features

- Ability to delete users
- Admin ability to add admins
- Chat function between coaches and athletes
- Drop down bar to chose sport to download CSV's of lift data
- Drop down bar to chose lift to view graphs for that lift
- Calculate weight of lift based on percentage of user's max
- Schedule section for users to view workout times