**Data Analyzation of Fitness Technologies - Team Analytics**

Luke Colussi

Dylan DeAlmeida

Aidan Shwom

Adam Slavny

Andres Wolfe

April 21, 2023

1. [Introduction](#_bookmark0) 1
   1. [Purpose](#_bookmark1) 1
   2. [Scope](#_bookmark2) 1
   3. [Definitions, Acronyms, Abbreviations](#_bookmark3) 1
      1. [Software](#_bookmark4) Components 1
      2. [User](#_bookmark5) Interface 1
2. [Design Overview](#_bookmark7) 2
   1. [Description of Problem](#_bookmark8) 2
   2. [Technologies Used](#_bookmark9) 2
      1. React ……………………………………………………………………………………………………………………… 2
      2. Node.js …………………………………………………………………………………………………………………… 3
      3. MongoDB ……………………………………………………………………………………………………………….. 3
      4. Netlify ……………………………………………………………………………………………………………………. 3
   3. [System Architecture](#_bookmark10) 4
   4. [System Operation](#_bookmark12) 5
3. [User Interface](#_bookmark19) 5
   1. Separated View……………………………………………………………………………………………………………………………….. 5
   2. Combined View ……………………………………………………………………………………………………………………………… 5
4. [Database](#_bookmark24) 6
5. [References](#_bookmark40) 15

# Introduction

## Purpose

## The purpose of this document is to describe the implementation of the Team Analytics Software described in the Senior Project requirements. The Team Analytics Software is designed to compliment other connected fitness apps to provide data visualization.

## Scope

## This document describes the implementation details of the Team Analytics Software. The software will consist of a two major functions. Data visualization in the forms of different charts, both line and pie chart, with advanced analyzation tools. These included data customization within the chart as well as queries to assist with data analyzation.

## Definitions, Acronyms, Abbreviations

### Software Components

### React A popular open-source JavaScript library for building user interfaces. It was developed by Facebook and is now maintained by both Facebook and a community of individual developers and companies. React allows developers to build reusable UI components that can be composed to create complex user interfaces for web and mobile applications.

### Node.js A JavaScript runtime built on the Chrome V8 JavaScript engine. It allows developers to run JavaScript code outside of a web browser, making it possible to build server-side applications with JavaScript. Node.js is popular for building scalable network applications, including web servers, APIs, and microservices. It is also often used for building tooling and automation scripts. Node.js is open-source and maintained by the Node.js Foundation and a large community of contributors.

### npm (short for "Node Package Manager") is a package manager for the JavaScript programming language. It is used primarily for installing, sharing, and managing packages or modules of reusable code that can be used in Node.js projects or front-end web applications. npm allows developers to easily install and manage third-party libraries and tools, as well as to create and publish their own packages to the npm registry. It is included with Node.js and can be accessed through the command line interface.

### JSON (JavaScript Object Notation) is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. It is a text format that represents data objects as key-value pairs, enclosed in curly braces. JSON is commonly used in web applications to transmit data between a server and a client, as it is language-independent and supported by many programming languages.

### API stands for Application Programming Interface. It is a set of rules and protocols that allows different software applications to communicate with each other. APIs are commonly used to access web-based services, allowing developers to integrate the functionality of one application with another. They define the types of requests that can be made, how to make them, and the data formats that should be used for the exchange of information.

### User Interface

### Line graph A type of chart that displays data as a series of points connected by straight lines. It is commonly used to show trends over time.

### Pie chart A circular chart divided into sectors, each representing a proportion of the whole. It is commonly used to show the distribution of data.

### Legend A key to the colors, symbols, or patterns used in a chart to represent different data sets. It is used to help the viewer understand the information displayed in the chart.

### Data Analytics The process of examining and interpreting data to uncover patterns, relationships, and insights that can inform decision-making.

### Fitness JSON file A file containing data about a person's physical activity, such as steps taken, distance traveled, and calories burned, typically recorded by a fitness tracker or mobile app.

### Distribution The way in which something is spread out or arranged over an area or among a group.

### Data Sets A collection of related data points that are grouped together for analysis. Viewer: The person who is looking at or interacting with the user interface.

# Design Overview

## Description of Problem

This project works in conjunction with three other groups in the development of a fitness application that allows a user and their partner to keep track of each other’s workout metrics and status.  The problem that this project intends to solve is the lack of analyzation tools to illustrate the effectiveness and overall usage of each feature and workout metric incorporated in each fitness application. In other words, if a feature or metric is under-utilized by a user or if users have issues with the application such as failure to connect to their partner or simply user-interface navigation, there is no current method to log user activity and also their workout progress to retroactively analyze the application’s usage as both a debugging tool and a development tool. This arises further problems as time and resources will be wasted on intended features that are not being used by users and real estate of the user-interface will take from other features that are more important.

## Description of Solution

Our proposed solution involves a standardized logging system using JSON files that other groups will incorporate into their applications. This logging system will solve the issue of not being able to retroactively view user activity in the application as each event a user can do, whether it is saving a setting or pushing a button, is stored in the log files along with the timestamp and other metadata. This further allows for any debugging as developers can see where the user was and what they were doing at any time and before an issue occurred. These JSON files are inserted into a database, which our web application takes from in order to display the data visualizations. This will allow researchers who use the web application to gauge the effectiveness of the user interface and its features, as a pie chart shows the usage of each user-interface page as a percentage. Thus if there was an issue connecting to the partner, both charts will demonstrate how frequently and how long the user was trying to connect to their partner during the workout. Also, if a partner ends up dropping a connection, this will be represented visually as well. In addition, the issue of viewing the data shown to the user throughout the entire workout is solved through the chart, which displays the various metrics along with elevation. Furthermore, specific queries are used to highlight regions where the heart rate, for example, has a specific distance between the user and their partner. This allows the developers and researches to know the conditions of the user and their partner, as well as the current user-interface page as a tooltip.

## Technologies Used

## React

## A popular open-source JavaScript library for building user interfaces. It was developed by Facebook and is now maintained by both Facebook and a community of individual developers and companies. React allows developers to build reusable UI components that can be composed to create complex user interfaces for web and mobile applications.

## Node.js

## A JavaScript runtime built on the Chrome V8 JavaScript engine. It allows developers to run JavaScript code outside of a web browser, making it possible to build server-side applications with JavaScript. Node.js is popular for building scalable network applications, including web servers, APIs, and microservices. It is also often used for building tooling and automation scripts. Node.js is open-source and maintained by the Node.js Foundation and a large community of contributors.

## MongoDB

## MongoDB is a popular NoSQL document-oriented database that stores data in a flexible, JSON-like format called BSON (Binary JSON). It is designed to be scalable, high-performance, and easy to use, making it a popular choice for many modern applications. MongoDB supports various data models, including key-value, document, graph, and column-family, and provides features such as ad hoc queries, indexing, and aggregation to handle complex data processing. It is widely used in web development, mobile applications, and big data analytics.

## Netlify

### Netlify is a cloud-based platform that provides developers with a variety of tools for deploying and managing modern web applications, including React applications. With Netlify, developers can quickly and easily deploy their applications to a global content delivery network (CDN), allowing for fast and reliable access from anywhere in the world. Netlify provides a range of features to support the deployment process, including continuous integration and delivery (CI/CD), automatic SSL certificate generation, and built-in caching and asset optimization. Additionally, Netlify offers a variety of deployment options, including automatic deploys from Git repositories, drag-and-drop deployments, and integration with popular build tools like Webpack and Gatsby. With its ease of use and powerful features, Netlify is a popular choice for developers looking to deploy and manage their React applications in a production environment.

## System Architecture

## Figure 1 depicts the high-level system architecture. The system will be constructed from multiple distinct components depicted below:

## 

## Figure 1: System Architecture

## System Operation

## We are in the process of deploying our React app to a live server, however it can also be run on a local host.

## Live deployment

## To deploy our React application, we used Netlify, a cloud-based hosting and deployment platform. We created a Netlify account and connected it to our Git repository, specifying our build settings such as the build command and build directory. Whenever changes were pushed to our connected Git repository, Netlify automatically built and deployed our application. We monitored the status of our build and deployment process through the Netlify dashboard, where we were able to see any build errors or issues that arose. Once our application was successfully built and deployed, we accessed it through the unique URL that Netlify generated for our application. Overall, Netlify provided an easy and efficient way for us to live deploy our React application to a global audience.

## Netlify link: <https://magnificent-treacle-5aaa81.netlify.app/>

## Local deployment

## To run the application through your local host you will need the following:

## System Requirements

## To run our React application you will require a package manager such a npm, which is usually downloaded alongside Node.js. Go to the Node.js website: <https://nodejs.org/>. Click on the "Download" button on the homepage. Choose the appropriate installer for your operating system (e.g. Windows, macOS, Linux). Run the installer and follow the installation prompts. Once the installation is complete, open your command prompt or terminal and run the following command “node -v” to check if Node.js and npm are installed. From there you can clone the GitHub repository to your local machine. To do so go to the repository you want to clone on GitHub. Click on the green "Code" button on the right-hand side of the page. Select "HTTPS" if it is not already selected. Copy the URL provided. Open your command prompt or terminal. Navigate to the directory where you want to clone the repository by using the "cd" command. For example, “cd Documents/GitHub/”. Once you are in the desired directory, run the following command “git clone [paste the URL here]”. Press Enter to run the command. This will create a local copy of the repository on your machine. Once the cloning is complete, navigate into the repository by running “cd repository-name”. Replace "repository-name" with the actual name of the repository. You can now work with the repository on your local machine, make changes, and push them back to the GitHub repository. To make the repository function properly you must now install all the required packages using the command “npm i” and the code will begin to function as intended.

# User Interface

# One of the key features of the front-end user interface is the data visualizations in the form of graphs and pie charts. The pie chart displays the usage of each user-interface page as a percentage, which allows researchers to understand the user's interaction with the system. It also helps developers to identify areas that need improvement or optimization. The graph shows various metrics along with elevation, which provides users with a visual representation of the data shown throughout the entire workout. It also allows developers and researchers to monitor the user and their partner's conditions, such as heart rate, by highlighting regions where there is a specific distance between them. The graph and pie chart also help users and developers identify any issues that may occur during the workout, such as a connection drop. In summary, the front-end user interface is designed to provide users and researchers with meaningful visual representations of the data collected during a workout, making it easy to monitor the user's performance and identify areas that need improvement or optimization. The user interface accomplishes this by using recharts to display the data in a separated and combined format. The graph is customizable, allowing the user to choose which metric they want to appear on the graph, using the options menu. The custom legend will adapt to gray out any non-selected data as well. To change how that data is presented there is a combine and separate button in the top center. Additionally, users can create queries to identify certain trends and analyze data. These queries will highlight all sections of the graph where the query applies.

# Separated View

# Figure 2 shows the default separated view

# 

# Figure 2: Separate view

# Combined View

# Figure 3 shows the combined view with custom queries

# 

# Figure 3: Combined view

# Database

# The success of this project depends heavily on the use of standardized log files that incorporate user, partner, and workout metadata, as well as computational values like heart rate, steps, calories, and elevation, all with timestamps for data visualization. The logging system captures event data, enabling analysis of user activity within the application. The JSON log files are created and modified upon application start and are sent to the database when the first workout ends, allowing the React application to retroactively link user and partner JSONs using metadata and timestamps.

# Figure 4-5 shows the standardized JSON file format and logging document

# 

# Figure 4: Standard JSON File Figure 5: Logging document

# As of now, the database has received 3.64k total requests, including those from fitness applications and the React application. A total of 282 log files have been generated and stored in the database. The database also provides export functionality and input validation.

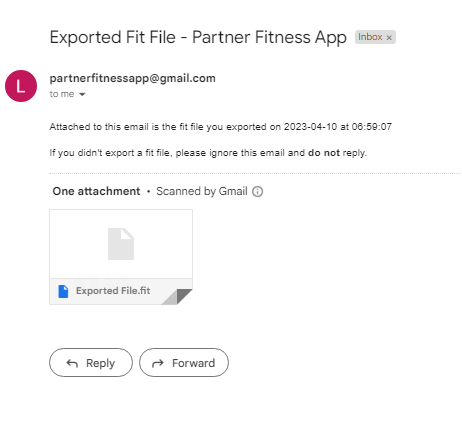
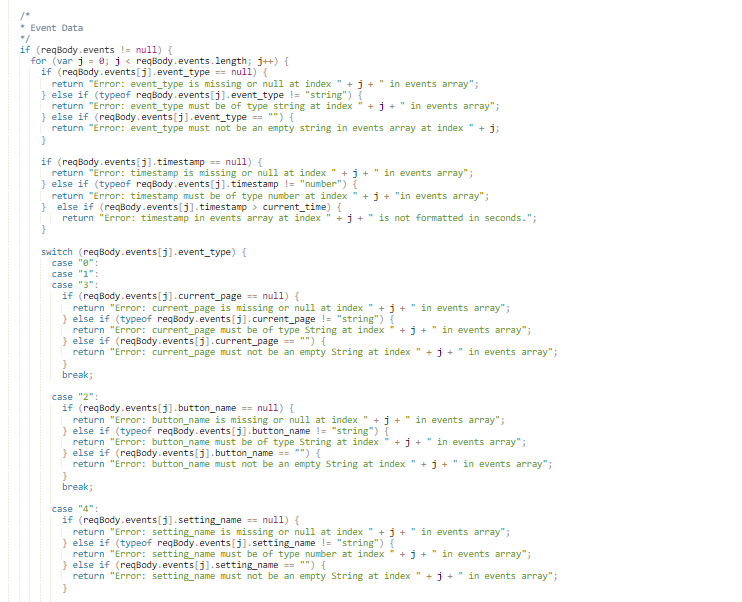
# Figure 6 shows the statistics of our MongoDB database

# 

# Figure 6: Database Statistics

# It features an HTTPS endpoint that accepts an email and FIT file as bytes, allowing users of other applications to export their generated FIT files. Input validation is enforced at the endpoints for insertion, checking for correct formatting and object placement within the log file, as well as pre-existing log files and valid type and value for any extra values, reducing the testing load on the React application.

# Figures 6-7 depict the exporting functionality and input validation implemented in our database



# Figure 6: Endpoint code Figure 7: HTTPS endpoint email