Final Paper for CSC 391: Athlete App Independent Study

Alexander Donald, Maurilio Saddoud, & Andrew Denny

Spring 2021

Introduction

As the end of the Fall 2020 semester was wrapping up and we were looking at classes for the Spring 2021 semester, we began talking about the possibility of doing an independent study. We wanted to gain both interesting and practical experience that we were struggling to find in the courses offered. The idea of learning how to build mobile applications was a hit amongst ourselves, so we began thinking about the formalities of receiving an independent study such as getting a faculty adviser. Having all worked closely with Professor Sakib Miazi in his Data Structures class, we decided to reach out to him. We presented the idea to Professor Miazi who excitedly accepted his role as our potential faculty adviser. From here, we went through the necessary steps to apply for the independent study such as creating a detailed syllabus outlining the course content, assigned readings, and evaluation procedures and submitting this course overview to Dr. Carl Yerger, the math and computer science department chair, for review and eventually acceptance.

Our independent study, named the Athlete App Independent study, is built around the goal of learning how to develop hybrid mobile applications and ultimately building an app that Davidson coaches and athletes can use to track specific metrics such as hours of sleep, stress/anxiety, soreness/fatigue, etc. The purpose for this application is to close the gap between coaches and athletes, giving the coaches a better understanding of what their athletes need and when they need it.

To be able to develop this app, we needed to learn the basics of HTML and JavaScript which are the framework of React Native, the mobile application development software that we would be using. We used Code Academy, an interactive online coding platform, to build small projects using HTML and JavaScript that enhanced our skills and understanding of these languages. Once we had a solid foundation in these two languages, we began our journey of learning React Native with an online Udemy course, *The Complete React Native + Hooks Course*. This course allowed us to follow along and create different React Native apps to learn the key skills needed to successfully develop mobile apps and more specifically our athlete app. Along the way, we built several very cool applications such as a restaurant search app that incorporated the Yelp API, allowing us to search for restaurants nearby categorized by price and quality. With our mobile application development tool kit full, we embarked on our journey of building the athlete app, Kumo Metrics (kumo is cloud in Japanese).

Motivation

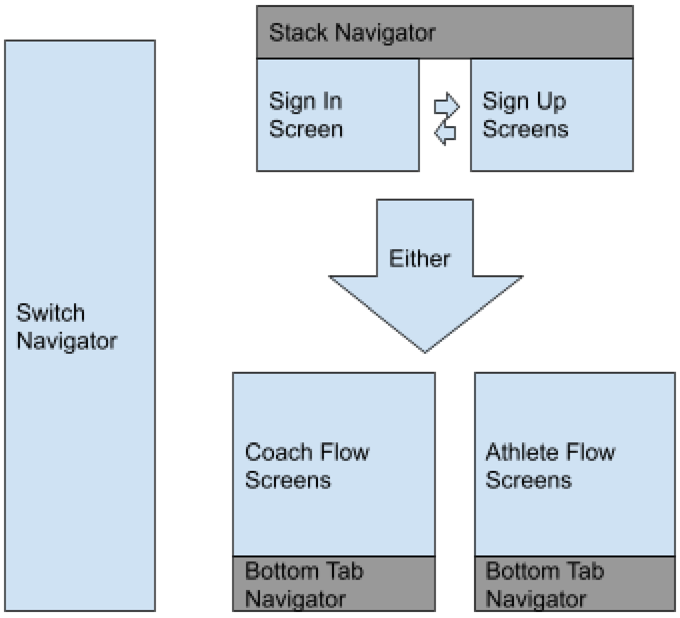
Our app, one that collects and records an athlete’s metrics over the course of a year or season, was born from John Young, the Davidson College swimming head coach, and his love for data. As the head coach of the swim team, and being someone who loves tracking metrics, John often complained about the valuable data that went uncollected as each season passed. He pondered the idea of subscribing to WHOOP, a fitness tracking service that utilizes the WHOOP Strap 3.0, a band which collects physiological data to display daily analytics and ultimately optimizes training, sleep, and recovery. This service would have easily solved John's problem, however, it is very costly, and additionally, the WHOOP strap collects more data than John is concerned with. Now we have arrived at a stalemate: John wants to collect data on his athletes, however, the only option does not seem to be worth the money—this is where the team behind Kumo Metrics comes in. Akin to WHOOP, we are interested in optimizing training, sleep, and recovery. Unlike WHOOP, our goal was to bring this technology and service to everyone, not just to those who can afford it.

Development

We began the development of the app by designing a rough outline of what we wanted the app to look like on the software Justinmind. We designed two different login flows where a user can either sign up as a coach and create a new team for his athletes to join or sign up as an athlete and join a pre-existing team. Once a user signs up we want to have two different interfaces depending on whether the user is a coach or an athlete. On the coach side we want there to be three different screens that the coach can flip through using bottom tabs. The first screen is the Questions Screen. This is the screen where coaches can add new questions to prompt their athletes with or delete pre-existing questions if they are no longer relevant. The second screen will be the Team Data Screen where the coach will have the ability to see all of his athletes' responses and see the average response to each question presented to him in a graph. The final screen is the Settings Screen where the coach has the ability to generate a join code for his athletes to use to join the team and a coach code for assistant coaches to use. On the athlete side, there are also three screens that the user can flip between using bottom tabs. The first screen is the Questions Screen where the athletes answer the questions that the coach has pushed to them. The second screen is the Athlete Data Screen where athletes can see all their previous answers to questions. Finally the last screen is the Setting Screen where athletes are able to log out and delete their account after graduation.

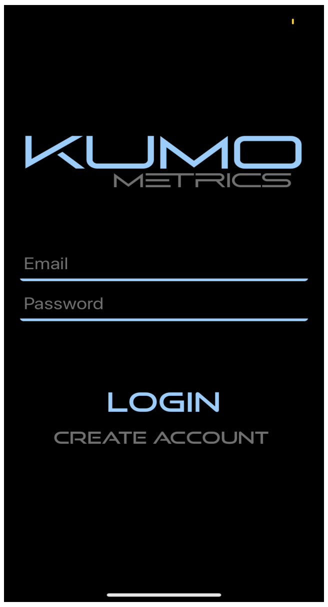
Navigation

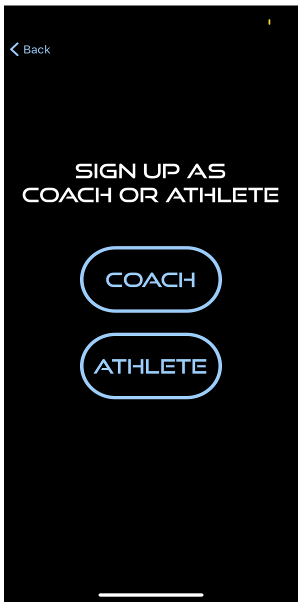
The first aspect of the application we developed was how we would navigate users through the application. We knew we wanted the users to be able to navigate back and forth between the different sign-up and sign-in screens. However, after the user logged in we didn't want them to have the ability to return back to the sign-in/sign-up. In addition we didn't want coaches or athletes to have access to the other's screens. As a result, we implemented an overarching Switch Navigator that once a user would enter a new navigation flow would prevent them from accessing any of the other flows. Within this Switch Navigator, the sign-in flow was implemented with a Stack Navigator that would allow the users to easily navigate back and forth between sign-in/sign-up screens. Finally, both the coach navigation flow and the athlete flow were implemented using a Bottom Tab Navigator which would present the different screens as tabs that the users could navigate back and forth between.

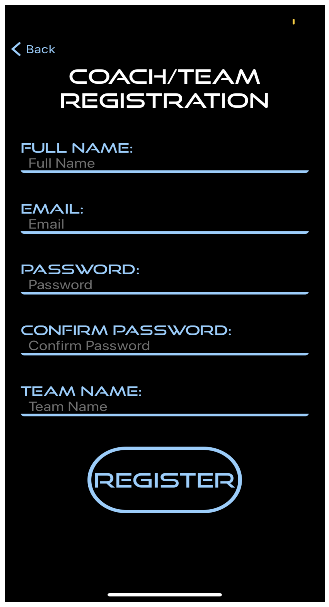


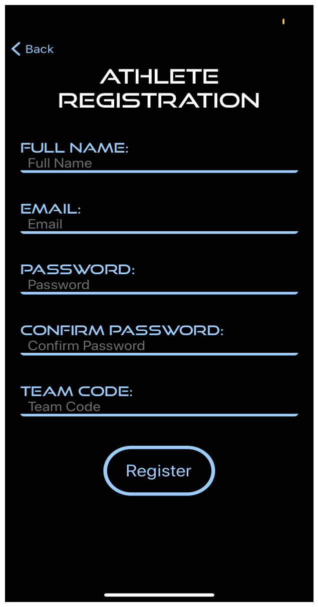
Sign-Up/Sign-In Screens

After the navigation, the next aspect of the application that we moved on to was the overall design of the sign-up and sign-in screens because they would be the first things that users would see when opening the app. To get the design we were looking for on the logo we used the font family "Abnes". The blue we used throughout the app was the color code: "#8ecfff". For the rest of our screens headers we used the font family "goodTimes". We ended up with the following screens:









Database

The design of the sign-in/sign-out screens was finished, but we were confronted with the issue of how to actually sign-up a user. Given that we only had one semester to complete most of this application we decided to use Google's Firebase to help us with user authentication and to store data that users would enter into our app. Once we decided that we were going to use Firebase, we still had to figure out how to structure our data. We decided that we were going to have 2 large overarching collections: one for the different teams that would be using our app and one for the users. The user collection would contain a document for every user that has signed up to use the application. Each document would contain the following fields:

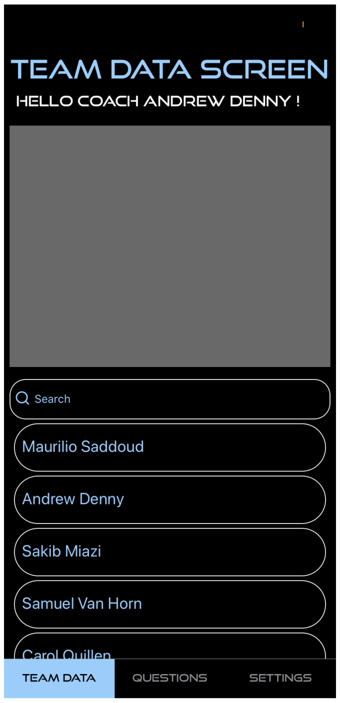
- 'teamId' - the team they are a member of
- 'id' - the randomly generated firebase id for this user's documents
- 'type' - athlete or coach
- 'email' - the email they signed up with

For the teams collections, it would contain a separate document for each team. Each team document contains the fields:

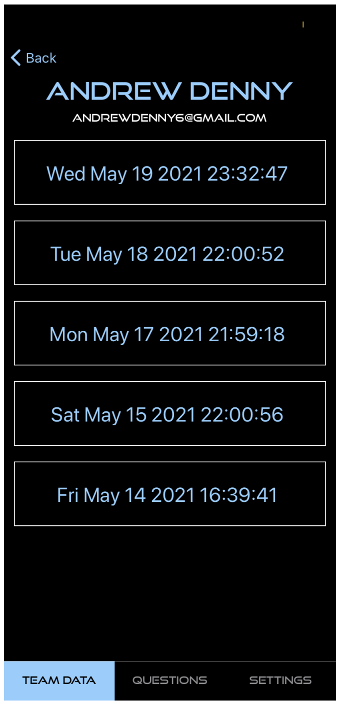
- 'mainCoachID' - the id of the coach that created the team
- 'coachCode' - the current coach join code
- 'signUpCode' - the current athlete join code
- 'teamName' - Name of the team

Within the team document it would contain three new collections of athletes, associateCoaches, and questions. The athlete collection contains the documents of all the athletes of the team, the question collection has the documents of all current team questions, and associateCoaches collection contains all documents of team coaches. Within each athlete document there was another collection called responses. Responses had a document for each time an athlete submitted their answers to the team's question. In each response document, the field's key is the question asked by the coach and the athlete's answer is the value.

<u>Coach Flow</u>

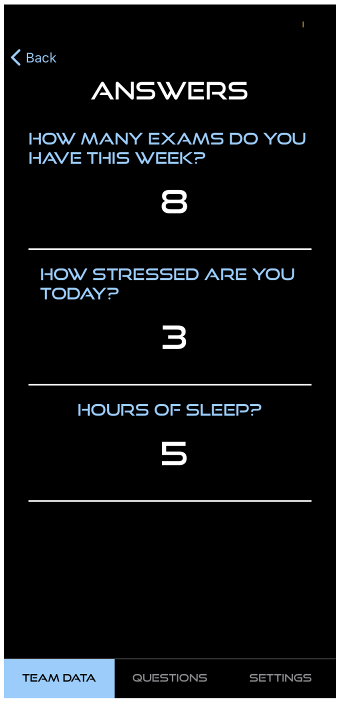

TEAM DATA SCREEN
HELLO COACH ANDREW DENNY !
Search
Maurilio Saddoud
Andrew Denny
Sakib Miazi
Samuel Van Horn
Carol Quillen

Team Data



ANDREW DENNY
ANDREWDENNY6@GMAIL.COM
Wed May 19 2021 23:32:47
Tue May 18 2021 22:00:52
Mon May 17 2021 21:59:18
Sat May 15 2021 22:00:56
Fri May 14 2021 16:39:41

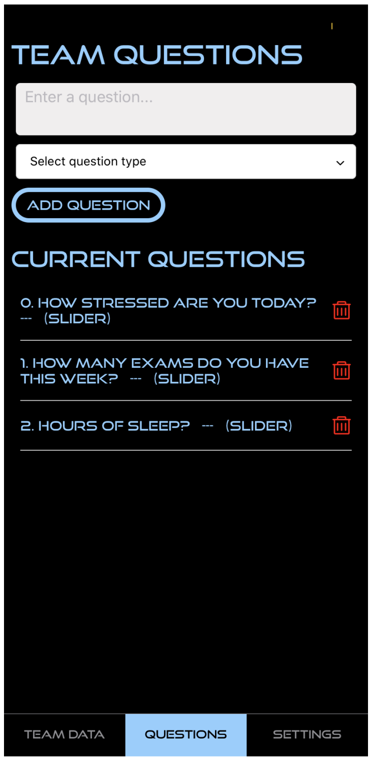
Athlete Info



ANSWERS
HOW MANY EXAMS DO YOU HAVE THIS WEEK?
8
HOW STRESSED ARE YOU TODAY?
3
HOURS OF SLEEP?
5

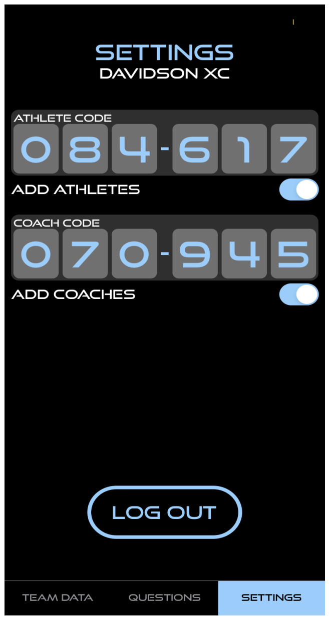
Response Info

Inside the coach flow the first screen the coach is presented with is the Team Data Screen. Inside this screen, we wanted the coach to see a list of all of his athletes. To implement this list we had to access all the athletes' documents from the team that the current coach created and get that information. In order to do this, we used firebase to find the team that had a mainCoachID field that matched the current logged in coaches ID. Once that team was found, we could loop through that team's athlete collection, saving each athlete's information to a state array. In addition, the coach can use the search bar to find a particular athlete on their team. The search bar searches through the list of athletes in real time so the coach doesn't have to type out the athletes full name in order to find who they are looking for. The coach can also click on the athlete he wants to see more information about them, and it will take him to the second screen above. This screen shows the coach all the days that this athlete has submitted responses and from there the coach can click on the day he wants to take a closer look at. This was implemented by using the athlete's name to navigate to their document inside the athlete collection. From the athlete's document, all we had to do was loop through all the documents inside the responses collections and use the time field to organize them into a list, and the field data gave us the information for the Response Info Screen.This screen is also supposed to have a graph so the coach can see the overall trends with his team's answer, however, we ran into issues implementing graphs so currently it is just a gray box.



0. HOW STRESSED ARE YOU TODAY? --- (SLIDER)
1. HOW MANY EXAMS DO YOU HAVE THIS WEEK? --- (SLIDER)
2. HOURS OF SLEEP? --- (SLIDER)

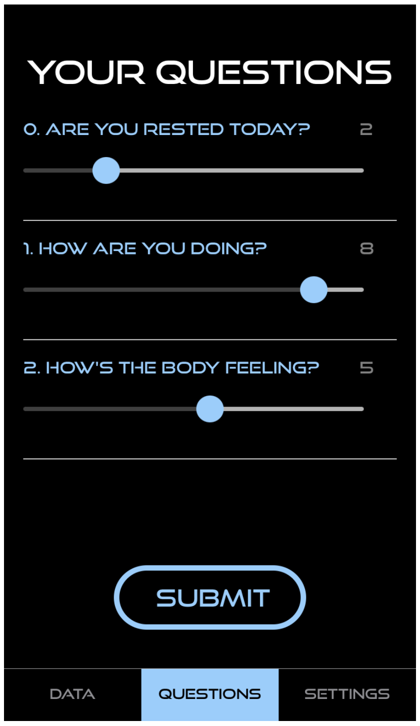
The next screen is the Team Questions Screen. This screen allows the coach to add and delete questions that will be presented to their players on the athlete flow. We use state to keep track of what is in the text input and to make sure that a question is typed before a question can be added. Once the add question button is clicked with a valid question and question type, the new question is added to a collection on Firebase associated with the coach's team. All current questions are displayed via a FlatList under the current questions header. To retrieve all the coach's current questions, we query Firebase to retrieve all documents within the questions collection, and display the questions field of each document. For each question, we have included a trash icon that, when clicked, allows the coach to delete a question. In short, this screen is where coaches modify what questions will be displayed to their athletes.



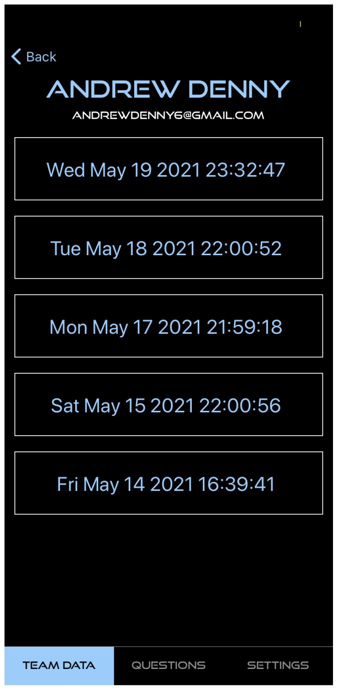
ATHLETE CODE 084-617
ADD ATHLETES
COACH CODE 070-945
ADD COACHES

The final coach flow screen is the Settings Screen. Although there aren't many options here yet, this screen enables the team's coaches to open or close their team to add new athletes, generating a unique code that athletes may use to join their respective teams. Additionally, coaches also have the ability to log out from this screen.

Athlete Flow



Inside the athlete flow, the first screen is the Athlete Question Screen. This screen is where athletes respond to questions assigned by their coach. Currently, all questions are presented as a list with a slider input for each question. Athletes are able to express their feelings toward each question with a number from 0 (low) to 10 (high). Although we will likely improve this screen in the future by adding better, more specific ways for athletes to record their answers, the basic method that is currently being used achieves the fundamental goal of our app: athletes being able to respond to questions posed by their coach, and the coach being able to access these responses (as seen on the Team Data Screen). Once the submit button is clicked, all questions are recorded to a collection of responses on Firebase and the athlete is notified of their submission.


Back

ANDREW DENNY

ANDREWDENNY6@GMAIL.COM

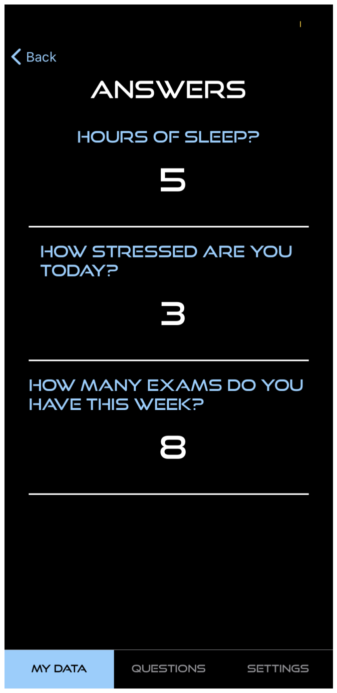
Wed May 19 2021 23:32:47

Tue May 18 2021 22:00:52

Mon May 17 2021 21:59:18

Sat May 15 2021 22:00:56

Fri May 14 2021 16:39:41


Back

ANSWERS

HOURS OF SLEEP?

5

HOW STRESSED ARE YOU TODAY?

3

HOW MANY EXAMS DO YOU HAVE THIS WEEK?

8

The next screen is the Athlete Data Screen. Here, we wanted the athlete to see all of their previous responses. To implement this idea, we wanted to show the athlete a list of all the days they have submitted responses. From this list, the athlete can click on the date they want to see the responses for. The implementation for this flow is the same as it was on the coach side.

The final screen is the Athlete Settings Screen (not pictured). This screen allows the athlete to log out by using a call to firebase authentication. In addition, on this screen the athlete can delete his account. When this button is pressed, any reference to this athlete's id is deleted from the Firebase database. Finally, the athlete can update their name just in case they misspelled it on initial registration.

Future Plans

Although the spring semester is finished and a rough draft of Kumo Metrics is complete, we plan to continue developing the app throughout the summer. Because we set out with the goal of getting Davidson sports teams to incorporate the app into their training routine, we feel that building a polished app is a great opportunity for us to continue to develop our skills and make our mark on the Davidson athletic community.

We will now discuss several areas of improvement that we will complete this summer. First, we will learn how to build graphs in React Native to visualize data collected from athletes. We will include graphs on the coach's team data screen to summarize their team's responses and on the athlete's data screen to display the trend of their responses over time. Second, we will update the athlete questionnaire to provide a better experience for them as they provide answers. Rather than having a single page with a slider for each question, we will most likely create a new screen for each question, turning the athlete question screen into a “slide deck.” Also, it is important that we incorporate more than just slider inputs for athletes to indicate their feeling toward a question. Some questions may need the athlete to elaborate with words, meaning a text box may be a better way to record their response. After the screens on Kumo Metrics are fully operational, we think it will be important to add notifications to remind athletes to fill out their questions daily. For the app to be effective and meaningful to coaches, their athletes must provide consistent responses.

Besides any other small changes that we decide to make along the way, Kumo Metrics should be polished and ready to be launched at this point. During this stage, it will be critical that we gather feedback from coaches and athletes concerning the usability and effectiveness of the app. This will allow us to engage in a feedback cycle to adapt to the needs of users. After talking with coaches and athletes, we should know whether Davidson sports teams are interested in the app or if we need to pivot our idea to something that appeals more to coaches and athletes. Nevertheless, the goal is to have Kumo Metrics fully functional by Fall, allowing sports teams to be smarter about their training without sacrificing performance on or off the field.