

Project Fit  
System Design Documentation

Shuhong Duan, Collin Hei Lok Chan,  
Vincent Liang, Matthew Melchior, Ziyue Gong

## Content

1. System Design CRC cards
2. System Architecture diagram (Three tiers-MERN Stack)
  - 2.1 Diagram
  - 2.2 Explanation
  - 2.3 System Decomposition/Project structure

## 1 System Design CRC cards

<b>Class Name:</b> Calendar	
<b>Responsibilities:</b> <ul style="list-style-type: none"><li>• A calendar page that displays which days have had activity recorded on them.</li><li>• Allow users to click on specific dates to track activities on those dates.</li></ul>	<b>Collaborators:</b> <ul style="list-style-type: none"><li>• TrackActivitySelect</li><li>• MainPage</li></ul>
<b>Class Name:</b> TrackActivitySelect	
<b>Responsibilities:</b> <ul style="list-style-type: none"><li>• An intermediary page that allows user to select which type of metric they wish to log (Exercise log, food log, body metrics, etc.)</li></ul>	<b>Collaborators:</b> <ul style="list-style-type: none"><li>• Calendar</li><li>• ExerciseLog</li></ul>
<b>Class Name:</b> ExerciseLogView	
<b>Responsibilities:</b> <ul style="list-style-type: none"><li>• Display all of a user's previously recorded sets in a scrollable list.</li><li>• Present options allowing users to add, update, or delete recorded sets.</li></ul>	<b>Collaborators:</b> <ul style="list-style-type: none"><li>• ExerciseRecorder</li><li>• ExerciseGroupSelect</li></ul>
<b>Class Name:</b> ExerciseGroupSelectView	
<b>Responsibilities:</b> <ul style="list-style-type: none"><li>• Provide users the option to search for exercises by clicking on a muscle group, or querying via a search bar.</li></ul>	<b>Collaborators:</b> <ul style="list-style-type: none"><li>• ExerciseLog</li><li>• ExerciseSelect</li></ul>
<b>Class Name:</b> ExerciseSelectView	
<b>Responsibilities:</b> <ul style="list-style-type: none"><li>• Display list of all found exercises given query from previous page (ExerciseGroupSelect)</li><li>• Allow users to select any displayed exercises, moving them to a screen where they can log the exercise.</li></ul>	<b>Collaborators:</b> <ul style="list-style-type: none"><li>• ExerciseGroupSelect</li><li>• ExerciseRecorder</li></ul>

<b>Class Name:</b> ExerciseRecorderView	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Display exercise in terms of Weight &amp; Reps or Time &amp; Distance depending on exercise type</li> <li>• Allow user to modify exercise metrics</li> <li>• Allow users to log exercise, saving the exercise metrics in their history</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• ExerciseLog</li> <li>• ExerciseSelect</li> </ul>

<b>Class Name:</b> Survey	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Display survey questions and answer options</li> <li>• Upon click, directs user to next question or the result page</li> <li>• Store user's survey results and send it to backend</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• Plan</li> </ul>

<b>Class Name:</b> Plan	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Match user's survey results with a plan</li> <li>• send the plan to frontend</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• Survey</li> </ul>

<b>Class Name:</b> FoodLog	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Display food the user has previously added in the form of a list</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• RecordFood</li> <li>• SelectFoodCategory</li> </ul>

<b>Class Name:</b> SelectFoodCategory	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Gives the user the ability to select a food category for the kind of food they want to record</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• FoodLog</li> <li>• SelectFood</li> </ul>

<b>Class Name:</b> SelectFood	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Lets the user select a specific food they want to add from the category they previously selected</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• RecordFood</li> <li>• SelectFoodCategory</li> </ul>

<b>Class Name:</b> RecordFood	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Displays the exercise details, calories, protein, etc.</li> <li>• Allows the user to modify the details for the food</li> <li>• Lets the user record the food saving it into their food log</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• SelectFood</li> <li>• FoodLog</li> </ul>

<b>Class Name:</b> BodyMetricLogView	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Gives the user the ability to select a body metric category they would like to track</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• BodyMetricRecorder</li> </ul>

<b>Class Name:</b> BodyMetricRecorder	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Display current body metric to be tracked</li> <li>• Display current body metric measurement if it exists for the category for that day</li> <li>• Gives the user the ability to input their measurement for the selected body metric category and save / edit for that day</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• BodyMetricLogView</li> </ul>

<b>Class Name:</b> MainPage	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Retrieve context from UserContexts as key to search for value in the database for further usage (eg. show username, get fitness data)</li> <li>• Display user's progress/goals information based on data retrieved from database by using UserContext as a key</li> <li>• Direct the user to the activities adding page via a process/goal</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• Survey</li> <li>• Login</li> <li>• UserContexts</li> <li>• CreateExercisePlan</li> </ul>

tracking button	
-----------------	--

<b>Class Name: ShortTermGoalComp</b>	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Allow users to add/delete/select from a certain amount of goals</li> <li>• Display user's goals' information on main screen and setting page</li> <li>• Communicate with database to modify/retrieve data</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• MainPage</li> </ul>

<b>Class Name: LoginComponents</b>	
<b>Parent Class (if any): LoginComponents</b> <b>Subclasses (if any): Login, Register</b>	
<b>Responsibilities:</b>  <b>Login:</b> <ul style="list-style-type: none"> <li>• Interact with the database to authenticate the users' login information (Email address, password)</li> <li>• Redirect the Users to the register page if they do not have an account</li> <li>• Redirect the users to the information dashboard (Main page)</li> </ul> <b>Register</b> <ul style="list-style-type: none"> <li>• Handle users' identity information's format</li> <li>• Interact with the database to upload the users' registration information</li> <li>• Redirect the Users to the login page if they already have an account</li> <li>• Redirect the users to survey page</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• Survey</li> <li>• MainPage</li> <li>• UserContexts</li> </ul>

<b>Class Name: UserContexts</b>	
<b>Parent Class (if any): UserContexts</b> <b>Subclasses (if any): SetContextComp, LogOutComp</b>	
<b>Responsibilities:</b>  <b>UserContexts</b> <ul style="list-style-type: none"> <li>• Create contexts for data's global usage and login status preserve</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• basically all other classes</li> </ul>

<ul style="list-style-type: none"> <li>• Create providers to wrap all routes so that all components can get access to data preserved in the contexts</li> </ul> <p><b>SetContextComp</b></p> <ul style="list-style-type: none"> <li>• Handling contexts updating for class components</li> </ul> <p><b>LogOutComp</b></p> <ul style="list-style-type: none"> <li>• Handling user login status reset and notifying contexts class</li> </ul>	
---	--

Class Name: ProfileScreen	
<p><b>Responsibilities:</b></p> <ul style="list-style-type: none"> <li>• Fetch the current user's information using a GET request</li> <li>• Display current user's information</li> <li>• Allow users to navigate to edit their profile</li> </ul>	<p><b>Collaborators:</b></p> <ul style="list-style-type: none"> <li>• EditProfileScreen</li> </ul>

Class Name: EditProfileScreen	
<p><b>Responsibilities:</b></p> <ul style="list-style-type: none"> <li>• Allow user to edit their profile</li> <li>• Handle the updated user information back to the database with a PUT request</li> <li>• Allow user to go back to the profile and not save his changes</li> </ul>	<p><b>Collaborators:</b></p> <ul style="list-style-type: none"> <li>• ProfileScreen</li> </ul>

Class Name: ColorTheme	
<p><b>Responsibilities:</b></p> <ul style="list-style-type: none"> <li>• Allow user to select background color from a dropdown list</li> <li>• Set all screens(e.g. survey, plan) background color as user selected</li> </ul>	<p><b>Collaborators:</b></p> <ul style="list-style-type: none"> <li>• Survey</li> <li>• Plan</li> </ul>

Class Name: CreateExercisePlan	
<p><b>Responsibilities:</b></p> <ul style="list-style-type: none"> <li>• Give users an interface to build and save an exercise plan.</li> <li>• Allow users to modify duration of exercise plan (total weeks)</li> <li>• Allow users to modify frequency of exercise plan (workouts/week)</li> </ul>	<p><b>Collaborators:</b></p> <ul style="list-style-type: none"> <li>• Settings</li> <li>• CreateWorkout</li> </ul>

<ul style="list-style-type: none"> <li>allow users to modify exercises for any given workouts</li> </ul>	
--	--

<b>Class Name:</b> CreateWorkout	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>Allow users to add/delete exercises for the selected workout</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>CreateExercisePlan</li> <li>WorkotExerciseSearch</li> </ul>

<b>Class Name:</b> WorkoutExerciseSearch	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>Allow users to search for exercises to add for the selected workout</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>WorkoutExerciseSearch</li> </ul>

<b>Class Name:</b> WorkoutExerciseSearch	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>Allow users to search for exercises to add for the selected workout</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>WorkoutExerciseSearch</li> </ul>

<b>Class Name:</b> Setting	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>Allow users to navigation to page color theme, set exercise plan, set calories budget</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>color theme</li> <li>set exercise plan</li> <li>set calories budget</li> </ul>

<b>Class Name:</b> exerciseCustomized	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>Allow users to create customized exercises</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>postCustomizedExercises</li> <li>exerciseCustomizedSelectController</li> </ul>

<b>Class Name:</b> postCustomizedExercises	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>controller</li> <li>post users' customized exercises to db</li> </ul>	<b>Collaborators:</b>



<b>Class Name:</b> exerciseCustomizedSelect	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• controller</li> <li>• display a user's customized exercises to the specific user when the user selecting exercise to log</li> </ul>	<b>Collaborators:</b>

<b>Class Name:</b> selectGoal	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• a menu to direct users to setting a specific long term goal</li> </ul>	<b>Collaborators:</b>

<b>Class Name:</b> weightGoal	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Allow users to create a weightGoal</li> <li>• Checks if weight measure available, if not, prompts user for starting weight input</li> <li>• Checks user input valid</li> <li>• Checks one user has only one goal associated to his/her account</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• bodyMetricLogController</li> <li>• longTermGoalController</li> </ul>

<b>Class Name:</b> longTermGoalController	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• create goal</li> <li>• get goal given userid</li> <li>• patch user with goalId</li> </ul>	<b>Collaborators:</b>

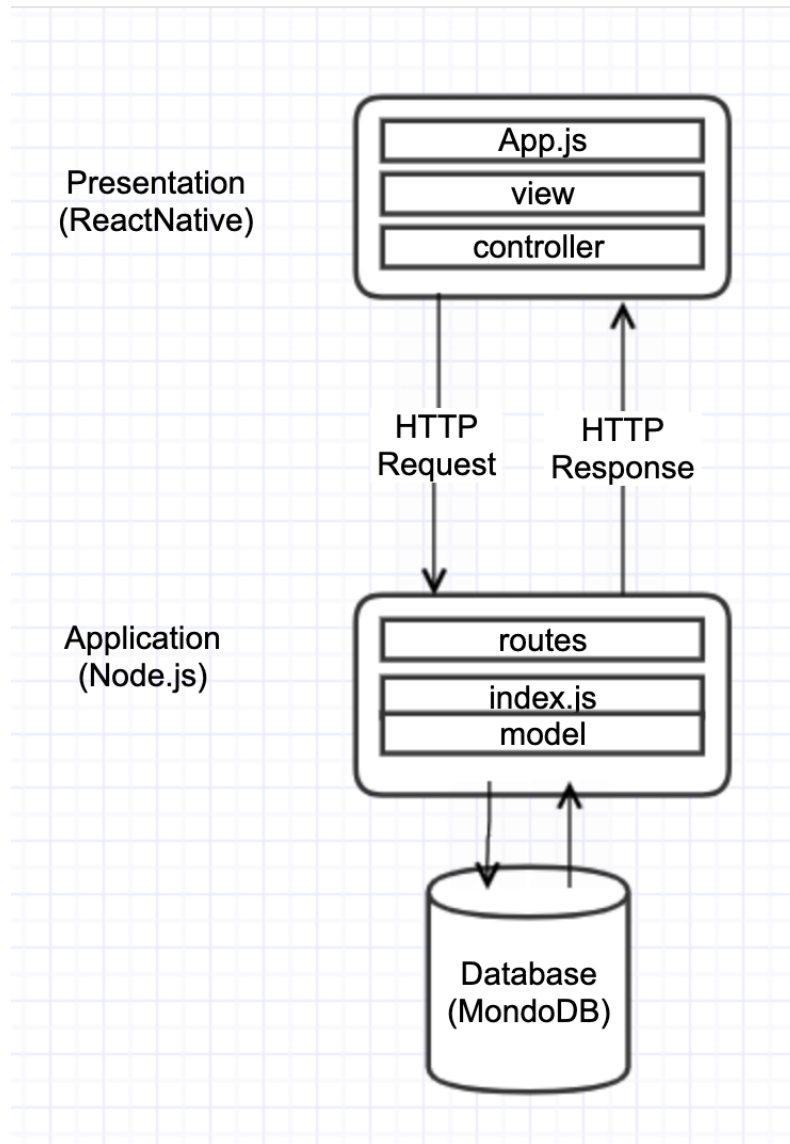
<b>Class Name:</b> SelectExercisePlan	
<b>Responsibilities:</b> <ul style="list-style-type: none"> <li>• Give users an interface to view and select default exercise plans</li> </ul>	<b>Collaborators:</b> <ul style="list-style-type: none"> <li>• Settings</li> <li>• ViewExercisePlan</li> <li>• Plan</li> </ul>

<b>Class Name:</b> ViewExercisePlan	
<b>Responsibilities:</b> <ul style="list-style-type: none"><li>• Allow users to view the default exercise plans more in depth, including the specific workouts and exercises.</li></ul>	<b>Collaborators:</b> <ul style="list-style-type: none"><li>• SelectExercisePlan</li></ul>

## 2. System Architecture diagram (Three tiers-MERN Stack)

reference: <https://www.mongodb.com/mern-stack>

2.1 Diagram: <https://www.bezkoder.com/react-node-express-mongodb-mern-stack/>



### 2.2 Explanation

- the front-end display tier (React Native), application tier (Express.js & Node.js), and database tier (MongoDB Atlas).
- React Native: build up interfaces through components, connect them to data on backend server, and render them as HTML.
- Express.js: has models for URL routing (matching an incoming URL with a server function), and handles HTTP requests & responses.
- MongoDB Atlas: data storage
- Interactions among the tiers: Making HTTP Requests or GETs or POSTs from React.js, we can connect to Express.js functions, which in turn use MongoDB's

Node.js drivers, either via callbacks for using Promises, to access and update data in your MongoDB database.

### 2.3.System Decomposition/Project Structure

Backend:

- models: directory that contains the data structure to model objects from response e.g. Users.js, Exercise.js
- routes: directory that contains files that define server endpoints e.g. users.js, Exercise.js
- config.js: config file that holds the credentials to connect to the MongoDB Atlas database
- index.js: entry point of backend server
- package.json: dependencies:

```
"dependencies": {
  "body-parser": "^1.20.0",
  "cors": "^2.8.5",
  "express": "^4.18.1",
  "mongoose": "^5.13.14",
  "morgan": "^1.10.0"
}
```

Frontend:

- view: directory that contains all js files of the main screens of the mobile app
- App.js: entry point of the react native frontend (contains tab bar navigation)
- package.json: dependencies

```
"dependencies": {
  "@react-navigation/native": "^6.0.10",
  "@react-navigation/native-stack": "^6.6.2",
  "expo": "~45.0.0",
  "expo-splash-screen": "~0.15.1",
  "expo-status-bar": "~1.3.0",
  "react": "17.0.2",
  "react-dom": "17.0.2",
  "react-native": "0.68.2",
  "react-native-web": "0.17.7"
}
```