

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

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

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

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

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

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

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

Class Name: SelectFood

Responsibilities:

- Lets the user select a specific food they want to add from the category they previously selected

Collaborators:

- RecordFood
- SelectFoodCategory

Class Name: RecordFood

Responsibilities:

- Displays the exercise details, calories, protein, etc.
- Allows the user to modify the details for the food
- Lets the user record the food saving it into their food log

Collaborators:

- SelectFood
- FoodLog

Class Name: BodyMetricLogView

Responsibilities:

- Gives the user the ability to select a body metric category they would like to track

Collaborators:

- BodyMetricRecorder

Class Name: BodyMetricRecorder

Responsibilities:

- Display current body metric to be tracked
- Display current body metric measurement if it exists for the category for that day
- Gives the user the ability to input their measurement for the selected body metric category and save / edit for that day

Collaborators:

- BodyMetricLogView

Class Name: MainPage

Responsibilities:

- Display information of the plans, progress, goals
- Allow users to track new metrics
- Retrieve context from UserContexts as key to search for value in the database for further usage (eg. show username, get fitness data)

Collaborators:

- Survey
- Login
- UserContexts

Class Name: LoginComponents

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

Class Name: UserContexts	
Parent Class (if any): UserContexts Subclasses (if any): SetContextComp, LogOutComp	
Responsibilities: UserContexts <ul style="list-style-type: none"> Create contexts for data's global usage and login status preserve Create providers to wrap all routes so that all components can get access to data preserved in the contexts SetContextComp <ul style="list-style-type: none"> Handling contexts updating for class components LogOutComp <ul style="list-style-type: none"> Handling user login status reset and notifying contexts class 	Collaborators: <ul style="list-style-type: none"> basically all other classes

Class Name: ProfileScreen	
Responsibilities:	Collaborators:

<ul style="list-style-type: none"> Fetch the current user's information using a GET request Display current user's information Allow users to navigate to edit their profile 	<ul style="list-style-type: none"> EditProfileScreen
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------

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

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

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

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

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

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

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

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

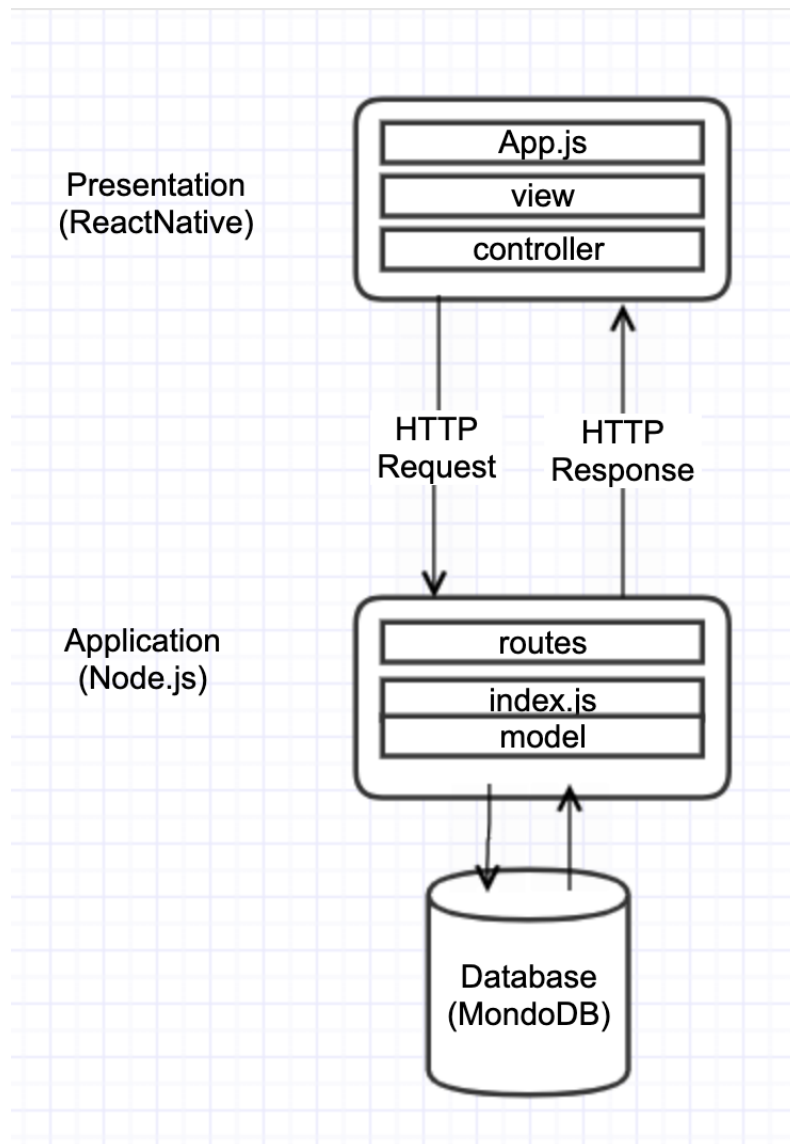
Class Name: postCustomizedExercises	
Responsibilities: <ul style="list-style-type: none"> controller post users' customized exercises to db 	Collaborators:

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

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"
  }
```