

## **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>	
<b>Parent Class (if any):</b> <b>Subclasses (if any):</b>	
<b>Responsibilities:</b>	<b>Collaborators:</b>

<b>Class Name:</b> ExerciseLog	
<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> ExerciseGroupSelect	
<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> ExerciseSelect	
<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> ExerciseRecorder	
<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>



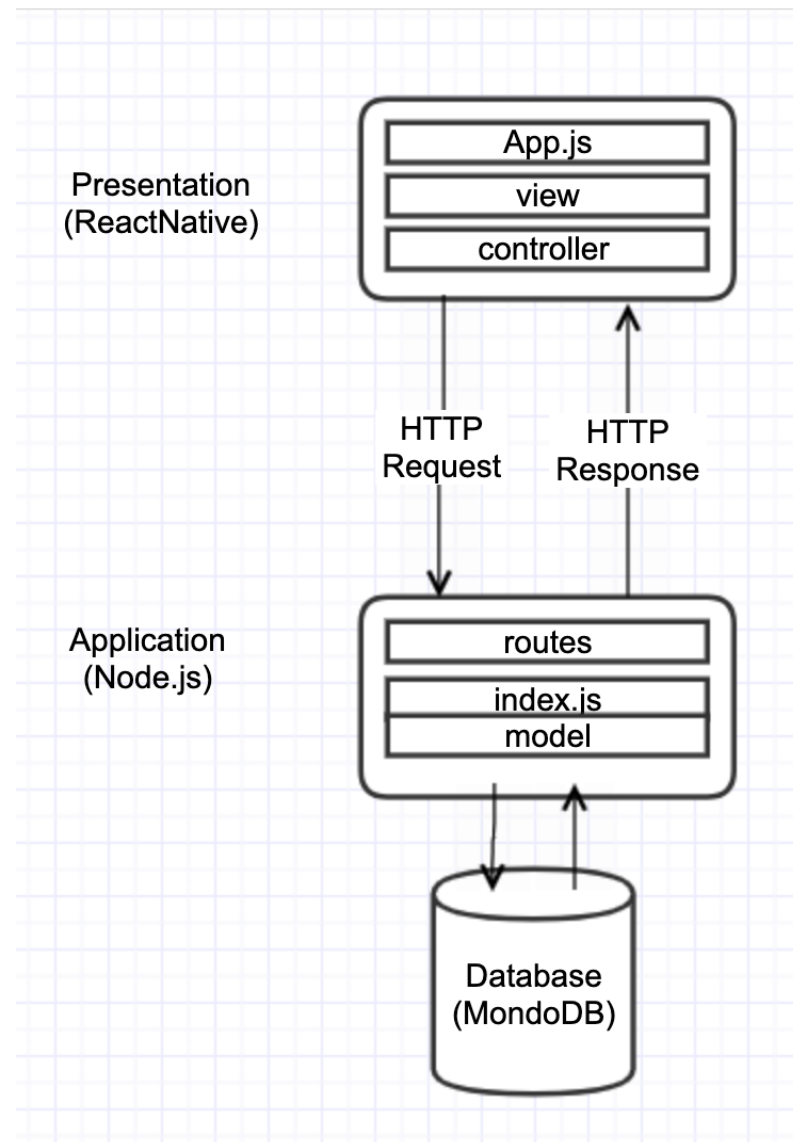



<b>Class Name:</b> Plan	
<b>Responsibilities:</b> <ul style="list-style-type: none"><li>• Match user's survey results with a plan</li><li>• send the plan to frontend</li></ul>	<b>Collaborators:</b> <ul style="list-style-type: none"><li>• Survey</li></ul>

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

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

2.1



### 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"  
}
```