Fitness Tracker App

A group of users is tracking their fitness progress using a mobile application. Each user is able to manage their own fitness data.

On the server side, at least the following details are maintained:

- Id - the internal fitness data-id. Integer value greater than zero.
- Date - the date when the fitness data was recorded. A string in the format "YYYY-MM-DD".
- Type - the type of exercise performed. A string of characters.
- Duration - the duration of the exercise in minutes. An integer value greater than zero.
- Distance - the distance covered in the exercise, if applicable. An integer value.
- Calories - the number of calories burned during the exercise. An integer value.
- Rate - the average heart rate during the exercise, if applicable. An integer value.

The application should provide at least the following features:

- Main Section (separate activity)
  - A. (1p) View the list of recorded fitness data. Using the **GET /dates** call, the user will retrieve the list of all their recorded fitness data. If offline, the app will display an offline message and a way to retry the connection and the call. Once retrieved, the data should be available, even offline.
  - B. (2p) By selecting a date, the user can view the details of the fitness data recorded on that date. To retrieve the details of a specific date's fitness data, the **GET /entries** call can be used by specifying the date. Once retrieved, the data should be available, even offline.
  - C. (1p) Add fitness entry. Using **POST /entry** call by specifying all the fitness data details, the user will be able to create a new fitness record. Available online only.
  - D. (1p) Delete fitness entry. By selecting a date from the list, and using the **DELETE /entry** call, the user will be able to delete a fitness data record. Available online only.
- Progress Section (separate activity)
  - A. (1p) View the total calories burned for each week. The list will be retrieved using the **GET /all** call and will be based on the user's recorded fitness data. From the server, we will retrieve the raw data entries.
  - B. (1p) View the top 3 types. Using the same **GET /all** call compute the top 3 types by total distance . From the server, we will retrieve the raw data entries.
- (1p) On the server side, once new fitness data is added to the system, the server will send, using a WebSocket channel, a message to all the connected clients/applications with the new fitness data object. Each application, that is connected, will display the received fitness data details, in human form (not JSON text) using an in-app “notification” (like snackbar or toast or a dialog or a message on the screen).

(0.5p) On all server operations, a progress indicator will be displayed.

(0.5p) On all server interactions, if an error message is received, the app should display the error message using a toast or snackbar. A log message should be recorded on all interactions (server or DB calls).