Parking App

A parking place is managing the available spaces using a mobile app. Each client is able to view the available spaces in the system and reserve and use one at the time. The owners are able to manage the place and view statistics.

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

- Id - the internal place id. Integer value greater than zero.
- Name - the place name. A string of characters representing the place name. Eg. L1N20 or L10N10.
- Type - the place type. A string of characters.
- Status - the place status. Eg. “available”, “taken”. A string type.
- Power - the electrical power capability. An integer number greater or equal to zero.

The application should provide at least the following features:

- Client Section (separate activity)
  a. (1p) View the available places in a list. Using **GET /places** call, the client will retrieve the list of available places in the system. If offline, the app will display an offline message and a way to retry the connection and the call. For each place the name, type and the power are displayed.
  b. (1p) Use a place. The client will be able to use a place, if available, using a **POST /take** call, by specifying the place id. Available online only.
  c. (1p) Once the client has a place, all the place details are presented to the user. Available offline too by persisting the details in a local database.
  d. (1p) Return the place. The user will have the option to free the place using **POST /free** call, by specifying the place id. Available online only.
- Owner Section (separate activity)
  a. (1p) The list of places descending by status and power. The list will be retrieved using the **GET /allPlaces** call, in this list along with the name, type and status, the app will display the power too. Note that from the server you are retrieving an unsorted list.
  b. (1p) Add a new place. Using a **POST /place** call, by sending the place object a new place will be added to the list, on success the server will return the place object with the id field set.
  c. (1p) Delete a place. Using **DELETE /place** call, by sending a valid place id, the server will remove the place. On success 200 OK status will be returned.

(1p) On the server side once a new place is added in the system, the server will send, using a websocket channel, a message to all the connected clients/applications with the new place object. Each application, that is connected, will display the received place details 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. On all interactions (server or db calls), a log message should be recorded.