Sharing Cars

At a law firm the employees are all sharing the cars available and they are tracking them using a mobile application. The application has a section for the office clerk, the person that manages the cars, and a section for the employee, the person who is using the car. Note that even an clerk is an employee.

Considering the above scenario the application should provide at least the following features:

- Employee Section
    a. (1p) The list of available cars, at least the name and the type of the car should be displayed. The list will be retrieved from the server side using a GET /cars call.
    b. (2p) When offline, once the employee has taken a car, instead of presenting the list of cars, the app will display the car details and a button to release the car (the car is marked available again). Once a car is confirmed to be released (POST /release with car id), the list of available cars is displayed once again on the main screen (if online, otherwise it will display a message that we are in the offline mode and offer the choice to retry).
    c. (1p) Get a car. Using a POST /rent call, by sending the car id on success it will return the car object with the status set to taken.
- Clerk Section
    a. (2p) By making a POST /register call with at least the following information: name and type, the clerk will be allowed to register a car. On success it will receive a car object that has at least the id field set. This object will be maintained on the device storage to survive between restarts.
    b. (1p) Available offline. Present details about the last car (we assume that one clerk has only one car, no full access yet :)). It will display details like:
        - Id - the id of the car, received from the server once we register a new car.
        - Name - the name of the car.
        - Type - the type of the car. Eg. convertible, sports, truck, van, ...
        - Status
            - Available - if the car is not taken.
            - Taken - if the car was rented by an user.

(1p) On the server side once a car is registered the server will send, using a websocket channel, a message to all connected applications with the new car object. The application will add the new object in the list of available cars. Alternatively the app should check at 5 second intervals if there are new cars (making the same GET /cars call) and if it's detecting one it will add the car to the main list.

(1p) On all server operations a progress indicator will be displayed. Also on all interactions, a log message should be recorded. If error messages are received from the server the app should display them as toast or snackbar messages.