Exams App

A school is managing its exams using a mobile application. The teachers are recording the exam details and the students are able to view them.

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

- Id - the internal exam id. Integer value greater than zero.
- Name - A string of characters representing the exam name.
- Group - A string of characters representing the group name.
- Details -A string of characters having the exam details.
- Status - A string of characters representing the status. Eg. “draft”, “pending”, “ready”, “canceled”, “done”, etc.
- Students - An integer value representing the number of expected students to take the exam.
- Type - the exam type. A string of characters. Eg. “license”, “masters”, etc.

The application should provide the following features (available without restarting the app):

- Teacher Section (separate activity)
  a. (1p) Register an exam. Using **POST /exam** call by specifying all the exam details. Available online and offline.
  b. (2p) View all the exams found in the system, in a list. Using **GET /exams** call, the teacher will retrieve all of them. The list should display at least the id, name, group and type. If offline, the app will display an offline message and a way to retry the connection and the call. Once the list is retrieved it should be available offline and online.
  c. (1p) By selecting an exam from the list, the teacher will be able to view all the exam details. To retrieve all the exam details **GET /exam** call will be used by specifying the exam id. Available online only.
- Student Section (separate activity) - Available online only.
  a. (1p) View all the exams that are having the status draft in the system in a list. Using **GET /draft** call, the student will retrieve all the exams having this status.(1p) By selecting an exam from the list, the student will be able to join the exam. Using **POST /join** by specifying the exam id.
- Stats Section (separate activity) - Available online only.
  a. (1p) View the exam details for a specified group. Using **GET /group** call, by using the specifying group name the user will be able to view all the exams associated with it. The list will present the exams in ascending order by type and descending by the number of expected students. If no such exams are available, the application will display a proper message. Also, note that the server is not ordering the list in any way.

(1p) On the server-side, once a new exam is added to the system, the server will send, using a WebSocket channel, a message to all the connected clients/applications with the new exam object. Each application, that is connected, will display the received exam details, in human form (not JSON text or toString) using an in-app “notification” (like a snack bar or toast or a dialog on the screen).

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

(0.5p) On all server/DB interactions, if an error message is generated, the app should display the error message using a toast or snack bar. On all interactions (server or DB calls), a log message should be recorded.