Documents App

An office is managing their documents using a mobile app. Everybody is able to add a document, query documents or view reports.

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

- Id - the internal document identifier. Integer value greater than zero.
- Name - the document name. A string of characters.
- Status - document current status. A string of characters. Eg. “shared”, “open”, “draft”, “secret”, etc.
- Owner - the owner of the document. A string of characters.
- Size - An integer value representing the size in KB.
- Usage - the number of times the document was opened. An integer value.

The application should provide at least the following features:

- Owner Section (separate activity)
  a. (1p) Record the owner’s name in application settings. Persisted to survive app restarts.
  b. (1p) Record a document. Using **POST /document** call by specifying all the document details and the owner’s name persisted in the previous step. Available online and offline.
  c. (2p) View all the owner’s documents. Using **GET /documents** call, the owner will retrieve all his documents. If offline, the app will display an offline message and a way to retry the connection and the call. Once retrieved, the documents should always be available, no other server calls are needed.
- Manage Section (separate activity)
  a. (1p) View all the available documents in the system in a list. Using **GET /all** calls, the user will retrieve all the available documents. The list should contain the document name, size, and usage. Available only online.
  b. (1p) Delete a document, the user will be able to delete the selected document. Using **DELETE /document** call, by sending the document id. Available online only.
- Status Section (separate activity)
  a. (1p) View the top 10 documents, in a list containing the document name, status, usage, and owner. Using the same **GET /all** call. The list should present the result in descending order by their usage value. Note that the list received from the server is not ordered.

(1p) On the server-side, once a new document is added in the system, the server will send, using a WebSocket channel, a message to all the connected clients/applications with the new document object. Each application, that is connected, will display the received document name, size, and owner values, in human form (not JSON text or toString) using an in-app “notification” (like a snackbar 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 snackbar. On all interactions (server or DB calls), a log message should be recorded.