Code Inspection Checklist

1.Java & Typescript

General Review

Checklist

Reviewer: Juan Arguello
Reviewed File: UseUpload.ts , ProtectedRoute.tsx, useAuth.tsx

1. Variable Declaration

- ☑ Are variable names informative
  - Yes
- ☑ Are variable names unique (not confusing or similar)?
  - Yes
- ☑ Are variable names following chosen capitalization conventions (camel case)?
  - Yes
- ☑ Are variables properly initialized?
  - Yes
- ☑ Are variables labelled as private or public based on their use?
  - NA
- ☑ Is every declared variable used?
  - Yes
- ☑ Is there excessive use of unnecessary temporary variables?
  - No

2. Methods and method signatures including return and input types

- ☑ Do method names reflect method functionalities?
  - Yes
- ☑ Do method expected return values match the intended use of the return value?
  - NA
- ☑ Do methods have safeguards for problematic/unexpected input?

    - Yes (include try catch blocks)
- ☑ Is there a high cohesion between the methods within the same class?
    - NA

3. Class definitions and grouping into packages (Java)

- ☑ Do object classes reflect the required elements of the program?
    - NA
- ☑ Are classes placed in the appropriate packages reflecting the nature of their use?
    - NA
- ☑ Are classes in different packages loosely coupled?
    - NA

4. Control flow Defects

- ☑ Are Switch cases used instead of if/else blocks when appropriate?
    - NA
- ☑ Are While loops successfully terminated to avoid infinite loops?
    - NA
- ☑ Are control flows used efficiently in the handling of erroneous input?
    - NA

- ☑ Are loop variables declared properly so that their scopes are only as big as necessary?
    - NA
- ☑ Are there checks for edge cases (out of bounds) for For loops?
    - NA
- ☑ Are there else blocks used for every if condition to ensure no case goes unhandled?
    - Yes

5. Code style & practices

- ☑ Is code consistently indented, spaced, and formatted?
    - Yes (code is perfectly indented)
- ☑ Code is well documented using inline comments and docstrings.
    - No (code is self explanatory)
- ☑ Are Expensive operations minimized (shallow object copies replacing deep ones if possible)
    - NA
- ☑ Are generics used where possible to improve code readability & reduce complexity?
    - NA

Typescript Review Checklist

- ☑ Are type annotations and inference used?
    - NA
- ☑ Are strict Null checks in place?
    - NA

Use type inference, type annotation, and generics.

Rollbar Review Checklist

- ☑ Is Rollbar being used consistently?
    - NA (files are hooks)

2. Front End (TS & ReactJS)

- ☑ Is the single responsibility principle applied to react components?
    - NA
- ☑ Are container components used strictly for managing state and business logic?
    - NA
- ☑ Are presentational components used for UI rendering and logic strictly?
    - NA
- ☑ Are related components, styles, and assets grouped within same directory?
    - Yes
- ☑ Are functional components used instead of class components if possible?
    - NA
- ☑ Are React hooks used to manage & control state & effects in functional components?
    - Yes
- ☑ Are hooks called at the top level of functional components?
    - Yes
- ☑ Are unnecessary re-renders avoided?
    - NA
- ☑ Are local component states prioritized for UI-specific state matters?
    - NA

General Additional Notes:

UseUpload.ts:

The file is short however it complies with our coding standards and it serves its purpose correctly.

ProtectedRoute.ts:

The file may need to contain a try-catch block because of the use of useAuth() may throw an error.

UseAuth.tsx:

File is short but it is well written and clear.