Home work review checklist

1. **Coding**
    a. Use of ESlint to make sure code adheres to ES standards and rules
        i. https://eslint.org/docs/user-guide/getting-started
        ii. https://github.com/airbnb/javascript
    b. Well formatted code
        i. https://prettier.io/docs/en/install.html
    c. Make use ES6 feature for most of the cases
    d. Write comments to explain the logic
    e. Proper and meaningful naming convention for variables, functions, css classes, arguments .
    f. Proper use of framework and libraries in its recommended way
        i. React
            1. PropTypes for components (https://reactjs.org/docs/typechecking-with-proptypes.html)
            2. Using react hooks over class components (optional)
            3. State maintained in proper place (https://reactjs.org/docs/lifting-state-up.html)
            4. Passing key prop for looping within JSX and not to use index as value for it.
    g. Error handling - In react using error boundaries (https://reactjs.org/docs/error-boundaries.html)
2. **Folder structuring**
    a. components folder - place to maintain reusable code with few internal states not shared state.
    b. container folder - place to maintain states being used across re-usable components, if redux is used this will have the action, reducers, side effects (saga or thunk), constant and selectors files
    c. styles folder - place to maintain global css (themes, root css)
    d. utils folder - place to maintain helper functions, global constants, base API interceptors or configs
    e. assets folder - place to maintain images, icons, illustration
    f. pages folder - place to maintain different pages used mainly if we have different layouts (optional)
3. **UI/UX**
    a. Desktop responsive app
    b. Proper app layout done
    c. Typography - Good use of variants (font size and weight for different texts)
    d. Icons - Meaningful and proper size
    e. Color - Primary and secondary variants with good contrast ratio, hover state color changes
    f. No sharp corners - use of rounded borders or box shadows
    g. Loading indicator while fetching the data.
    h. Add empty states wherever needed
4. **Implementation**
    a. No hard coding
    b. Listing and filters is required at minimum
    c. App must not break while using

d. Not to use any libraries for transforming the JSON object, can use native ES6 features to do it.
e. Can use external libraries (lodash or any) for other utility functions except transforming the JSON response object (Not preferred)
f. Not to use any CSS framework or ready made UI components from external libraries (material-ui, ANT, semantic ui, bootstrap)
g. Good breakdown of components and state maintained at proper place.