React Test

1) What are props drilling and how can we improve this mechanism for large scale applications?

Ans) Prop drilling is passing data to React component through many different components that doesn't want to it.

For example : let suppose we have 4 component **A**, **B**, **C** and **D**. that are sequentially connected parent to child with each other. component **A** have a state "**name**". That we need to use in the component **D**. how we achieve this property. using props drilling. Firstly component **A** pass "**name**" state as props to Component **B**. Component **B** pass "**name**" as props to component **C** and finally **C** pass "**name**" as props to the component D and then we use it.
In this process the component **B** and **C** have no need of state "**name**" but it pass unwanted to it.
Its ok for small applications. But in the case of large-scale application passing data through number of different components is difficult.
We can overcome this by using **Redux state management library** or **React context** which allow to keep data (application state) in a particular place and all application components can access it directly without passing through other components.

2) What are refs and when to use them ?

Ans) The **refs** are a special attribute is used to direct access and manipulate the DOM elements in react. such as focus of input fields , text selection, or reset value etc.

3) Why we use middleware in React like (Redux-Saga, Redux-Thunk etc) ?

Ans) Middleware allows for side effects (like API requests) to be run without blocking state updates.

Common use of middleware is to support asynchronous actions without using other library dependency.

4) What is the replacement of componentWillReceiveProps lifecycle method in React v17, what parameters do that method accepts?

Ans) useEffect hook is equivalent of the **componentWillReceiveProps**.

Its first argument is a function called effect. Second argument is array of dependencies, which is optional argument,

- **useEffect** run after every render cycle
- **useEffects** are always executed after render,
- **An Empty Array []:** the side-effect runs once after the initial rendering.
- Has state or props [props1, props2,……., state1, state2] the side-effect runs only when any dependency value changes.

Practical work

5)TASK:

Make a screen which shows the list of all dog breeds got from API.

The API end point will be:

https://dog.ceo/api/breeds/list/all

TYPE: GET

Clicking on any of the breed will open a MODAL popup, and it will show all the images of that particular breed. For instance, we click on hound, so the Modal will show images of hounds.
API endpoint to be used:

https://dog.ceo/api/breed/{breedNAME}/images
TYPE: GET

The images will be shown in form of a GRID. (Rows and Cols). Each row will have 5 images.

You can use any UI library for your ease.