POTLA MOHAN SAI **DATE:14-02-2022**

MERN Stack Assignment 1

1.What exactly do you mean when you say "prop drilling," and how do you avoid it?

Props are nothing but the data. Prop drilling is a process of in react app in which props are passed from one component to other component by passing through the other components which do not require the props. It makes the process lengthy.
We can avoid prop drilling by creating a data store in which props are stored and we can connect any component to the data store irrespective of the component position in the component position tree. All the components have access to the data store.
We can also avoid it by creating the global data store using the concept of context in react and the components in context will have access to it.

2.In React JS, how do you add validation to props?

To validate props React JS has an inbuilt component. This feature validate props to make sure that the values passed through props are valid or not.

The Syntax for validation of props in React JS:

```
class Component extends React.Component {
        render() {}
}
component.propTypes = {/*……..*/}
```

3. Is it possible to use classes in NodeJS?

Yes, it is possible to use classes in Node JS. Classes are templates for creating objects. We can create classes in Node JS by using JavaScript prototype and ECMA Script 6(ES 6).

4.What is the purpose of super(props)?

To initialise the state object we define the constructor() function . We have to use super() to implement constructor() function in React component. We call super(props) to make sure that the react component constructor function is getting called or not. super(props) is just a reference to the parent constructor function.

5.Why are the Express app and server separated?

Express app is a simple framework for web applications in Node JS and server is responsible for the initialisation of the middleware, setting up the engine and routes through which requests are made.
By making them separate, we will have the following advantages
1.faster testing execution
2.It allows testing the API in-process without having to perform the network calls
3.data abstraction and encapsulation
4.modularity
5.scalability
6.reusability