MERN Stack Training
Company: Sensation Software Solutions
Student Name: Gurvinder Singh
Training Duration: 6 Months
Days: 21

Objective of the Day

The main objective of Day 21 was to understand the core concepts of **React State, Props, and Hooks (`useState`)**. The focus was on making React components dynamic, passing data between components, and handling user interactions efficiently.

Work Done on Day 21

The training session started with a **revision of React JS basics**. The trainer briefly revised important concepts such as **component-based architecture, JSX, and the virtual DOM** to build a strong foundation.

After the revision, the trainer explained the concepts of **Props and State** in detail. It was explained that:

- **Props** are used to pass data from a parent component to a child component.
- **State** is used to manage dynamic data inside a component.

It was also explained that whenever the state changes, React **re-renders the component automatically**, which updates the user interface.

Concept of Props

To understand props clearly, a simple example was discussed where data was passed from a parent component to a child component.

Example:

```
function Welcome(props) {
  return <h2>Welcome {props.name}</h2>;
}

function App() {
  return <Welcome name="Gurvinder" />;
}
```

From this example, it was understood that props are **read-only** and cannot be modified inside the child component.

Concept of State

After that, the concept of **State** was explained. State is mainly used to handle **dynamic and changeable data** in React applications.

The trainer introduced the `useState` hook, which is used in functional components to manage state.

Example:

```
import { useState } from "react";

function Counter() {
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>Count: {count}</p>
      <button onClick={() => setCount(count + 1)}>
        Increment
      </button>
    </div>
  );
}
```

This example helped in understanding:

- `count` represents the current state value
- `setCount` is the function used to update the state
- The component re-renders when the state value changes

Hands-on Practice

During the practical session, we created small React components such as:

- Counter application
- Show/Hide text using state
- Handling button click events

I personally implemented a **Show/Hide Message component** where the message appears or disappears on button click.

Learning Outcomes

By the end of Day 21, I learned:

- The difference between **Props and State**
- Practical usage of the `useState` hook
- How React handles component re-rendering
- How to manage user interactions in React

Conclusion

Day 21 was an important day because it covered the core concept of **state management in React**. These concepts form the foundation for building advanced features such as **forms, To-Do list applications, and full React projects**. The session was productive and provided both theoretical and practical understanding.