**3. React-HOL**

**1. Explain React components**

React components are the building blocks of any React application. They encapsulate the logic and structure of UI elements. Each component can manage its own state and receive data via props.

**2. Identify the differences between components and JavaScript functions**

| **Aspect** | **React Components** | **JavaScript Functions** |
|---|---|---|
| Purpose | Used to render UI in React | Perform logic or computation |
| Returns | JSX/HTML (UI elements) | Any data (string, number, object) |
| Lifecycle Methods | Yes (in class components) | No |
| State Management | Yes (state/hooks) | No |
| React Dependency | Requires React to work | Standalone |

**3. Identify the types of components**

React supports:

1. **Class Components**: Based on ES6 classes, use lifecycle methods and state via this.state.
2. **Function Components**: Based on JavaScript functions. With Hooks, can now use state and side-effects.

**4. Explain Class Component**

A **Class Component** is created using the class keyword and extends React.Component. It must define a render() method and can maintain state and lifecycle methods.

```
class Welcome extends React.Component {
 render() {
  return <h1>Hello, {this.props.name}</h1>;
 }
}
```

**5. Explain Function Component**

A **Function Component** is a plain JavaScript function that returns JSX. It can now use **Hooks** (e.g., useState, useEffect) to manage logic and state.

```
function Welcome(props) {
 return <h1>Hello, {props.name}</h1>;
}
```

**6. Define Component Constructor**

In class components, a **constructor** is used to:

- Initialize state
- Bind methods to this
- Call super(props) to use parent constructor

```
constructor(props) {
  super(props);
  this.state = { name: "John" };
}
```

**7. Define render() function**

The render() method is required in class components and defines what UI elements to show:

```
render() {
  return <h1>Hello World</h1>;
}
```

Function components don’t need a render() method, they directly return JSX.

**HANDS-ON**

**CODE: CalculateScore.js**

```
import React from 'react';
import '../Stylesheets/mystyle.css';

function CalculateScore(props) {
  const { name, school, total, goal } = props;

  const average = total / goal;

  return (
    <div className="score-card">
      <h2>Student Score Report</h2>
      <p><strong>Name:</strong> {name}</p>
      <p><strong>School:</strong> {school}</p>
      <p><strong>Total Marks:</strong> {total}</p>
      <p><strong>Goal:</strong> {goal}</p>
      <p><strong>Average Score:</strong> {average.toFixed(2)}</p>
    </div>
  );
}
export default CalculateScore;
```

**mystyle.css**

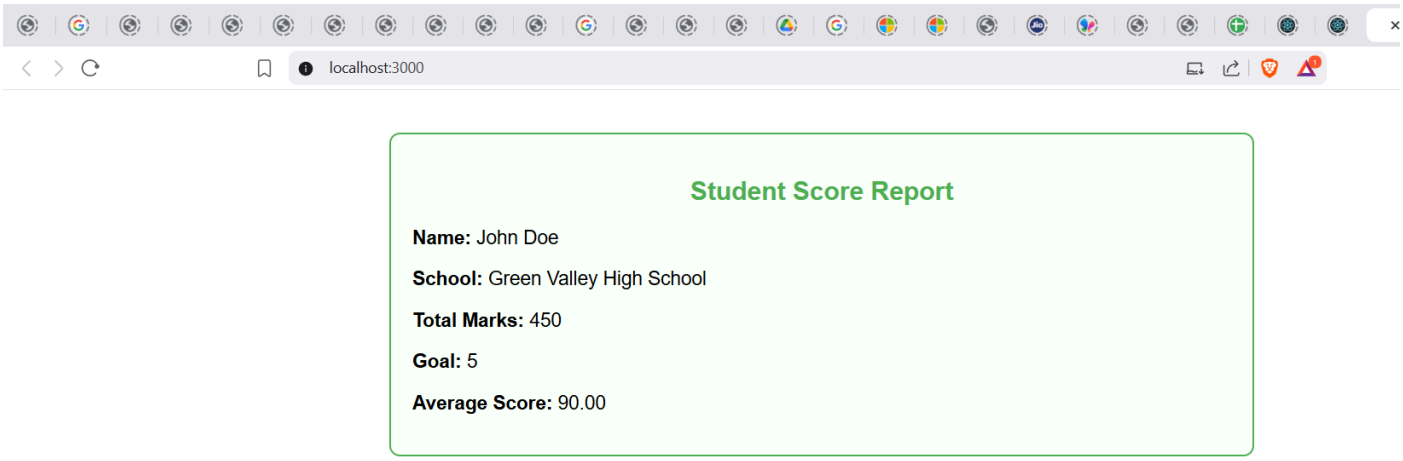
```
.score-card {
  margin: 40px auto;
  padding: 20px;
  width: 50%;
  border: 2px solid #4CAF50;
  border-radius: 10px;
  background-color: #f9fff9;
  font-family: Arial, sans-serif;
}
.score-card h2 {
  color: #4CAF50;
  text-align: center;
}
.score-card p {
  font-size: 18px;
}
```

**app.js**

```
import React from 'react';
import CalculateScore from './Components/CalculateScore';
function App() {
  return (
    <div className="App">
      <CalculateScore
        name="John Doe"
        school="Green Valley High School"
        total={450}
        goal={5}
      />
    </div>
  );
}

export default App;
```

**OUTPUT**


localhost:3000
Student Score Report
Name: John Doe
School: Green Valley High School
Total Marks: 450
Goal: 5
Average Score: 90.00