**Name: SASHMITHA R**

**Superset ID: 6387562**

# 9.React.js – HOL

## 1.ES6 Features:

1. Arrow Functions: Concise syntax for defining functions.
2. let and const: Block-scoped variable declarations.
3. Template Literals: String interpolation with backticks.
4. Destructuring Assignment: Extracting data from arrays or objects.
5. Default Parameters: Setting default values for function parameters.
6. Rest and Spread Operators: For combining or expanding arrays and objects.
7. Classes and Inheritance: Object-oriented programming support.
8. Modules: import and export for modular code.
9. Promises: Handling asynchronous operations.
10. Set and Map: New data structures for managing collections.
11. Generators and Iterators: Custom iterable objects.

---

## 2.let in JavaScript:

- Block-scoped variable declaration.
- Cannot be redeclared in the same scope.
- Supports updating the value but not re-declaration.

**Example:**

```
let x = 10;

if (true) {

  let x = 20;

  console.log(x);
}
console.log(x);
```

### 3.var vs let:

| Feature | var | let |
|---|---|---|
| Scope | Function-scoped | Block-scoped |
| Hoisting | Hoisted and initialized as undefined | Hoisted but not initialized |
| Re-declaration | Allowed | Not allowed in the same scope |

### 4.const in JavaScript:

- Block-scoped.
- Cannot be reassigned.
- Prevents re-declaration.

**Example:**

```
const PI = 3.14;
```

### 5.ES6 Class Fundamentals:

- A blueprint for creating objects with methods and properties.

**Example:**

```
class Person {
  constructor(name) {
    this.name = name;
  }
  greet() {
    console.log(`Hello, ${this.name}`);
  }
}
const person1 = new Person('John');
person1.greet();
```

---

## 6.Class Inheritance:

- Allows one class to inherit properties and methods from another.

**Example:**

```
class Animal {
  constructor(name) {
    this.name = name;
  }
  speak() {
    console.log(`${this.name} makes a sound`);
  }
}
class Dog extends Animal {
  speak() {
    console.log(`${this.name} barks`);
  }
}
const dog = new Dog('Rex');
dog.speak();
```

---

### 7.Arrow Functions:

- Concise syntax with implicit this binding.

**Example:**

```
const add = (a, b) => a + b;

console.log(add(2, 3));
```

---

### 8.Set and Map:

1. **Set**: Stores unique values.

```
const set = new Set([1, 2, 3, 3]);

console.log(set);
```

2. **Map**: Stores key-value pairs.

```
const map = new Map();

map.set('name', 'John');

console.log(map.get('name'));
```

**Command :** npx create-react-app cricketapp

### ListofPlayers.js

```
import React from 'react';

const ListofPlayers = () => {

  const players = [

    { name: 'Virat', score: 95 },

    { name: 'Rohit', score: 80 },

    { name: 'Dhawan', score: 65 },

    { name: 'KL Rahul', score: 50 },
    { name: 'Hardik', score: 88 },

    { name: 'Jadeja', score: 72 },

    { name: 'Bumrah', score: 60 },

    { name: 'Shami', score: 55 },

    { name: 'Ashwin', score: 90 },

    { name: 'Pant', score: 40 },

    { name: 'Surya', score: 85 }

  ];

  const lowScorers = players.filter(player => player.score < 70); // ES6 arrow

  return (

    <div>

      <h2>All Players</h2>

      <ul>

        {players.map((player, index) => (

          <li key={index}>{player.name} - {player.score}</li>

        ))}

      </ul>

      <h2>Players with score below 70</h2>

      <ul>

        {lowScorers.map((player, index) => ( <li key={index}>{player.name} - {player.score}</li>))}

      </ul>

    </div>

  );

};

export default ListofPlayers;
```

**IndianPlayers.js**

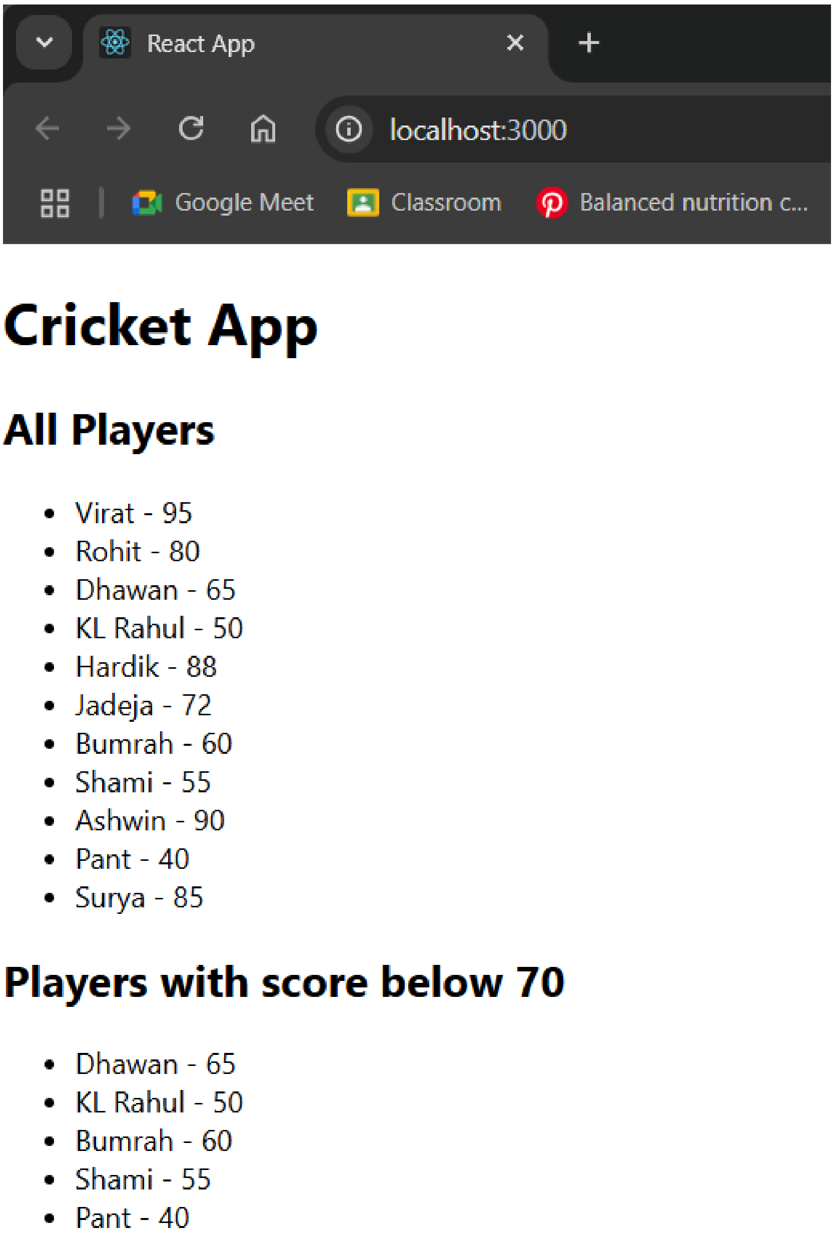
```
import React from 'react';
const IndianPlayers = () => {
  const team = ['Player1', 'Player2', 'Player3', 'Player4', 'Player5', 'Player6'];
  const [odd1, even1, odd2, even2, odd3, even3] = team;
  const T20players = ['Virat', 'Rohit', 'Hardik'];
  const RanjiPlayers = ['Pujara', 'Rahane', 'Karun'];
  const mergedPlayers = [...T20players, ...RanjiPlayers];
  return (
    <div>
      <h2>Odd Team Players</h2>
      <ul>
        <li>{odd1}</li>
        <li>{odd2}</li>
        <li>{odd3}</li>
      </ul>
      <h2>Even Team Players</h2>
      <ul>
        <li>{even1}</li>
        <li>{even2}</li>
        <li>{even3}</li>
      </ul>
      <h2>Merged T20 and Ranji Players</h2>
      <ul>
        {mergedPlayers.map((player, index) => (
     <li key={index}>{player}</li>
    ))}
   </ul>
  </div>
 );
};
export default IndianPlayers;
```

**App.js**

```
import React from 'react';
import ListofPlayers from './ListofPlayers';
import IndianPlayers from './IndianPlayers';
function App() {
 const flag = true;
 return (
  <div className="App">
   <h1>Cricket App</h1>
   {flag ? <ListofPlayers /> : <IndianPlayers />}
  </div>
 );
}
export default App;
```

**Command :** npm start

**Output :**



```
Cricket App

All Players

• Virat - 95
• Rohit - 80
• Dhawan - 65
• KL Rahul - 50
• Hardik - 88
• Jadeja - 72
• Bumrah - 60
• Shami - 55
• Ashwin - 90
• Pant - 40
• Surya - 85

Players with score below 70

• Dhawan - 65
• KL Rahul - 50
• Bumrah - 60
• Shami - 55
• Pant - 40
```