**9. ReactJS-HOL**

**Objectives**

**List the features of ES6**

ECMAScript 6 (ES6) introduced several powerful features:

* let and const for block-scoped variables
* Arrow functions () => {}
* Classes and inheritance using class and extends
* Template literals using backticks (`Hello ${name}`)
* Default parameters in functions
* Destructuring assignment
* Spread (...) and Rest (...) operators
* Promises for asynchronous programming
* Modules (import / export)
* Map and Set data structures

**Explain JavaScript let**

* let allows you to declare block-scoped variables.
* Unlike var, let is not hoisted to the top of the function or block.
* Example:

let x = 10;

{

let x = 20;

console.log(x); // 20

}

console.log(x); // 10

**Identify the differences between var and let**

| **Feature** | **var** | **let** |
| --- | --- | --- |
| Scope | Function-scoped | Block-scoped |
| Hoisting | Hoisted (initially undefined) | Hoisted (not initialized) |
| Redeclaration | Allowed | Not allowed in same scope |

**Explain JavaScript const**

* const is used to declare **constants** — variables that cannot be reassigned.
* Block-scoped like let.
* You can still **mutate** objects and arrays declared with const.
* Example:

const a = 5;

// a = 6; // Error

const arr = [1, 2];

arr.push(3); // Allowed

**Explain ES6 class fundamentals**

* ES6 introduced the class syntax to define object blueprints.
* Includes constructors, methods, and can be extended.

class Person {

constructor(name) {

this.name = name;

}

greet() {

console.log(`Hello, ${this.name}`);

}

}

**Explain ES6 class inheritance**

* Use extends keyword to inherit from another class.
* Use super() to call the parent constructor.

class Student extends Person {

constructor(name, id) {

super(name);

this.id = id;

}

}

**Define ES6 arrow functions**

* Shorter syntax for writing functions.
* Does **not** bind its own this, great for callbacks.

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

const greet = name => `Hello ${name}`;

**Identify set(), map()**

Set

* Stores unique values.
* No duplicate items allowed.

let s = new Set([1, 2, 2, 3]);

console.log(s); // Set {1, 2, 3}

Map

* Stores key-value pairs.
* Keys can be of any type (not just strings).

let m = new Map();

m.set('a', 1);

m.set(2, 'b');

console.log(m); // Map {"a" => 1, 2 => "b"}

**Creating a React Application named “cricketapp”**

1. **Create the Project**

npx create-react-app cricketapp

cd cricketapp

mkdir src/components

1. **ListOfPlayers.js**

* Declaring an array with 11 players and store details of their names and scores using the map feature of ES6
* Filtering the players with scores below 70 using arrow functions of ES6.

// src/components/ListOfPlayers.js

import React from 'react';

const players = [

  { name: "Jack", score: 50 },

  { name: "Michael", score: 70 },

  { name: "John", score: 40 },

  { name: "Ann", score: 61 },

  { name: "Elisabeth", score: 61 },

  { name: "Sachin", score: 95 },

  { name: "Dhoni", score: 100 },

  { name: "Virat", score: 84 },

  { name: "Jadeja", score: 64 },

  { name: "Raina", score: 75 },

  { name: "Rohit", score: 80 }

];

function ListOfPlayers() {

  const players70 = [];

  players.map((item) => {

    if (item.score <= 70) {

      players70.push(item);

    }

  });

  return (

    <div>

      <h1>List of Players</h1>

      <ul>

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

          <li key={index}>

            Mr. {item.name} <span>{item.score}</span>

          </li>

        ))}

      </ul>

      <hr />

      <h1>List of Players having Scores Less than 70</h1>

      <ul>

        {players70.map((item, index) => (

          <li key={index}>

            Mr. {item.name} <span>{item.score}</span>

          </li>

        ))}

      </ul>

    </div>

  );

}

export default ListOfPlayers;

1. **IndianPlayers.js**
   1. Display the Odd Team Player and Even Team players using the Destructuring features of ES6.
   2. Declare two arrays T20players and RanjiTrophy players and merge the two arrays and display them using the Merge feature of ES6

// src/components/IndianPlayers.js

import React from 'react';

export function OddPlayers({ players: [first, , third, , fifth] }) {

return (

<div>

<h2>Odd Players</h2>

<ul>

<li>First: {first}</li>

<li>Third: {third}</li>

<li>Fifth: {fifth}</li>

</ul>

</div>

);

}

export function EvenPlayers({ players: [, second, , fourth, , sixth] }) {

return (

<div>

<h2>Even Players</h2>

<ul>

<li>Second: {second}</li>

<li>Fourth: {fourth}</li>

<li>Sixth: {sixth}</li>

</ul>

</div>

);

}

const T20Players = ['First Player', 'Second Player', 'Third Player'];

const RanjiTrophyPlayers = ['Fourth Player', 'Fifth Player', 'Sixth Player'];

export const IndianPlayers = [...T20Players, ...RanjiTrophyPlayers];

function MergedPlayers() {

return (

<div>

<h2>List of Indian Players Merged:</h2>

<ul>

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

<li key={index}>Mr. {player}</li>

))}

</ul>

</div>

);

}

export default MergedPlayers;

1. **App.js**

import React from 'react';

import ListOfPlayers from './components/ListOfPlayers';

import MergedPlayers, {

OddPlayers,

EvenPlayers,

IndianPlayers,

} from './components/IndianPlayers';

const flag = true;

function App() {

return (

<div className="App">

{flag ? (

<>

<ListOfPlayers />

</>

) : (

<>

<OddPlayers players={IndianPlayers} />

<hr/>

<EvenPlayers players={IndianPlayers} />

<hr/>

<MergedPlayers />

</>

)}

</div>

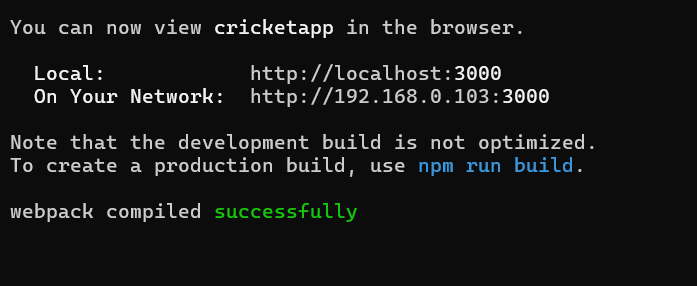
);

}

export default App;

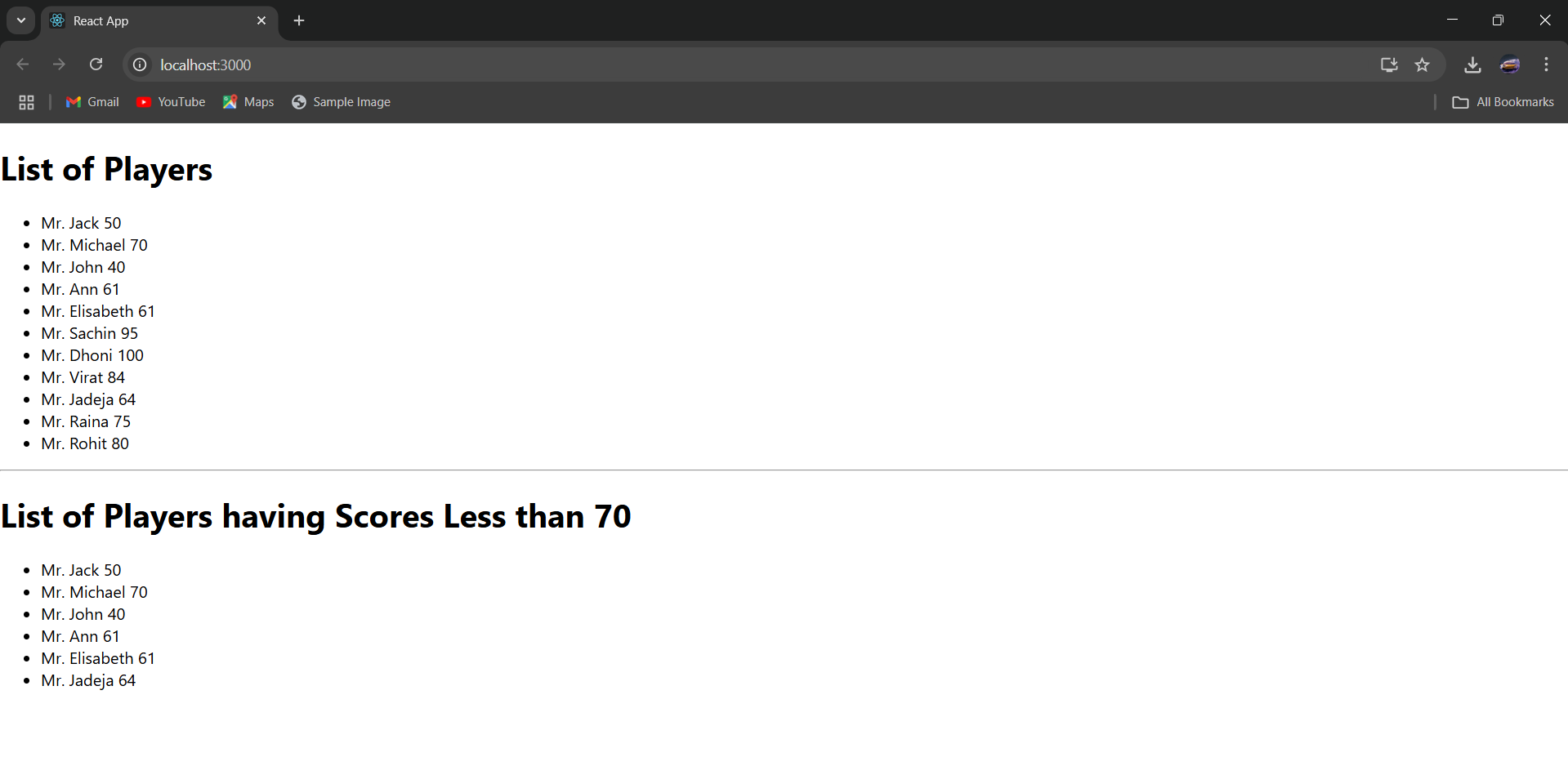
1. **To Run**

**Npm start**



<http://localhost:3000>

**When flag=true:**



**When flag=false:**

