WEEK 7 HANDSON 1

{

"name": "cricketapp",

"version": "0.1.0",

"lockfileVersion": 3,

"requires": true,

"packages": {

"": {

"name": "cricketapp",

"version": "0.1.0",

"dependencies": {

"@testing-library/dom": "^10.4.1",

"@testing-library/jest-dom": "^6.6.4",

"@testing-library/react": "^16.3.0",

"@testing-library/user-event": "^13.5.0",

"react": "^19.1.1",

"react-dom": "^19.1.1",

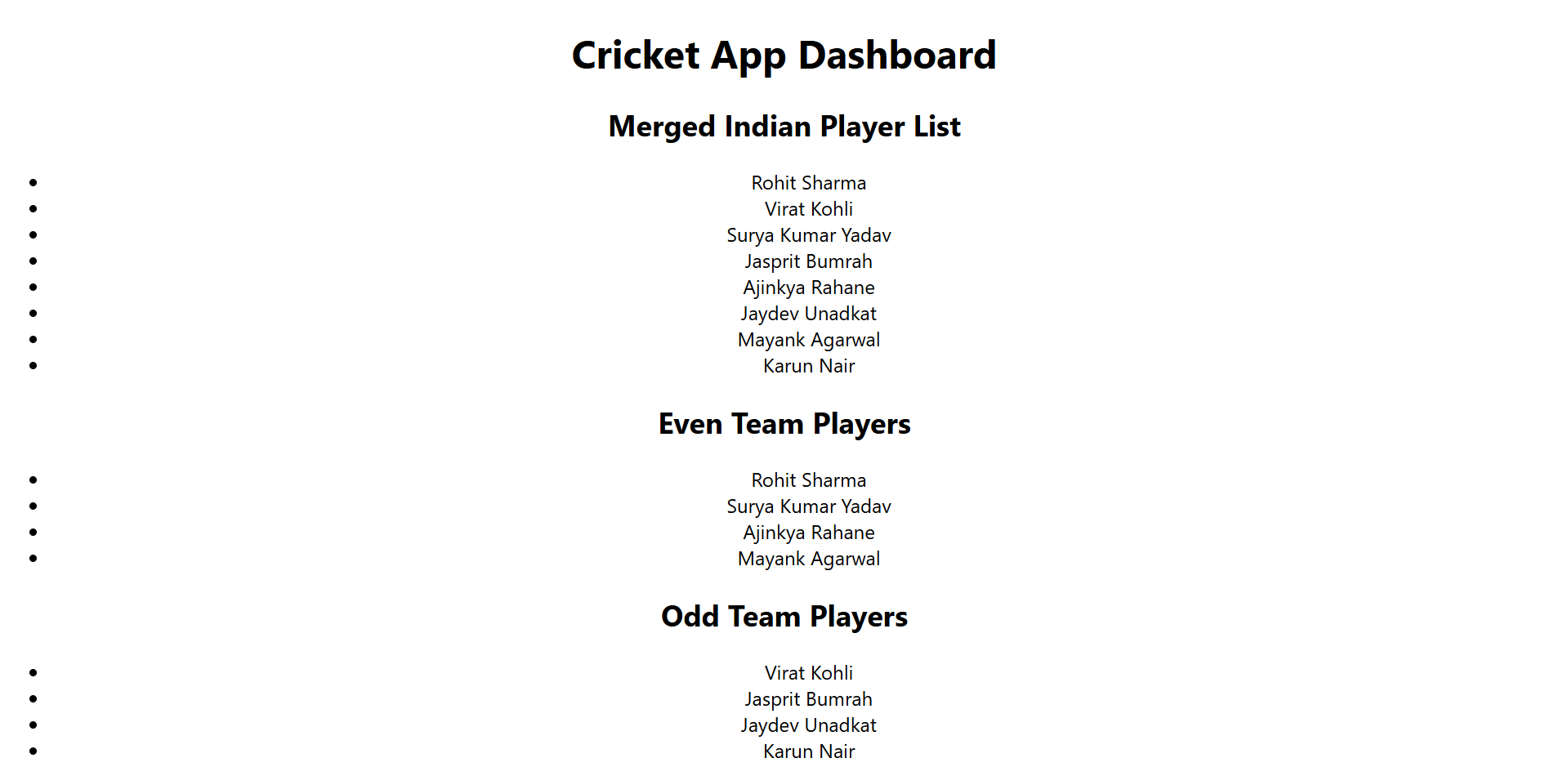
"react-scripts": "5.0.1",

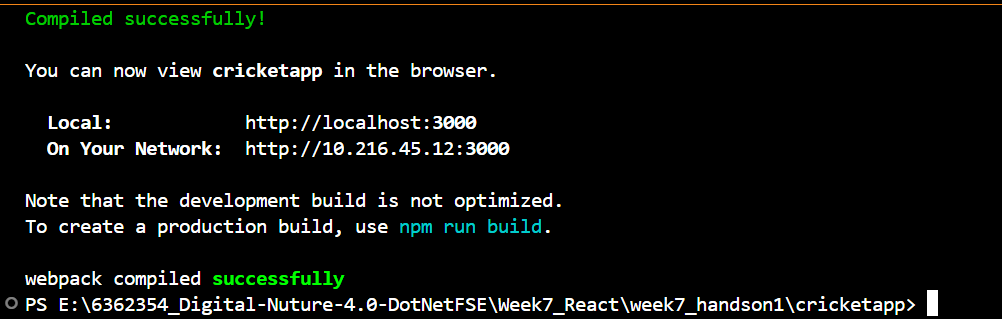
"web-vitals": "^2.1.4"

}

},

**OUTPUT**

****

****

****

**OBJECTIVE ANSWERS**

### **1. List the Features of ES6**

**ES6, also known as ECMAScript 2015, introduced major improvements to JavaScript. Key features include let and const for block-scoped variable declarations, arrow functions for concise syntax, template literals for easier string handling, default parameters in functions, destructuring assignment for arrays and objects, and enhanced object literals. ES6 also introduced important data structures like Map and Set, class-based OOP syntax, Promises for async operations, and modules for better code organization. These features significantly improve code readability, maintainability, and performance.**

### **2. Explain JavaScript let**

**The let keyword is used to declare variables in ES6 and offers block-level scoping, meaning it is only accessible within the nearest enclosing {} block. This is a major improvement over var, which is function-scoped. let allows better control over variable visibility and avoids unintended behavior like variable hoisting. Variables declared with let can be reassigned but not redeclared within the same scope. It is now the preferred way to declare variables that may change during execution.**

### **3. Identify the Differences Between var and let**

**The main difference between var and let is scope. var is function-scoped, while let is block-scoped. This means let restricts access to the variable within the {} in which it is declared, reducing bugs due to accidental reuse of variables. Additionally, variables declared with var are hoisted to the top of their scope and initialized with undefined, while let declarations are hoisted but not initialized, resulting in a ReferenceError if accessed before declaration. Also, let does not allow redeclaration in the same scope, making code more robust and predictable.**

### **4. Explain JavaScript const**

**The const keyword is used to declare variables whose value cannot be reassigned. Like let, const is also block-scoped. It is particularly useful for defining constants or values that should not change, such as configuration values or references to functions. However, const only prevents reassignment of the variable identifier—it does not make the contents immutable. For example, arrays and objects declared with const can still have their contents modified. Using const improves code safety by making the intention of immutability clear to developers.**

### **5. Explain ES6 Class Fundamentals**

**ES6 introduced the class syntax as syntactic sugar over JavaScript's prototype-based inheritance. A class in ES6 is declared using the class keyword and can include a constructor method, class properties, and other methods. The constructor is a special function used for initializing objects. Class methods are defined inside the class body and are automatically added to the prototype of the class. This provides a cleaner and more readable way to implement object-oriented programming in JavaScript, making it easier for developers with backgrounds in other languages like Java or C++ to adopt JavaScript.**

### **6. Explain ES6 Class Inheritance**

**ES6 supports class inheritance through the extends keyword, which allows one class (child) to inherit the properties and methods of another class (parent). The child class can call the parent constructor using the super() keyword. This promotes code reusability and logical hierarchy in applications. Inheritance enables developers to create base classes with shared functionality and then extend them for more specific use cases. ES6 class inheritance simplifies the traditionally complex prototype chain and offers a clearer and more consistent syntax.**

### **7. Define ES6 Arrow Functions**

**Arrow functions, introduced in ES6, provide a shorter syntax for writing function expressions. They are declared using the => syntax and are especially useful for inline functions and callbacks. Unlike regular functions, arrow functions do not bind their own this context—they inherit this from the surrounding lexical scope. This makes them ideal for use inside class methods, React components, and array methods like map(), filter(), and forEach(). Their concise syntax and predictable behavior have made arrow functions a common standard in modern JavaScript.**

### **8. Identify Set() and Map() in ES6**

**Set and Map are new data structures introduced in ES6. A Set is a collection of unique values, meaning it automatically removes duplicates. It supports operations like add(), delete(), and has(). Set is useful when you need to store a list of items with no repetitions.**

**A Map is a collection of key-value pairs where keys can be of any type (not just strings like in plain objects). It provides methods like set(key, value), get(key), delete(key), and has(key). Unlike plain objects, Map maintains the insertion order of keys and is more efficient for frequent additions and deletions.**