**React.js Introduction**

Q1: What is React.js? How is it different from other JavaScript frameworks and libraries?

Ans: - React.js is a JavaScript library for building user interfaces, particularly single-page applications where performance and dynamic. React uses a UI development and virtual DOM to optimize rendering performance. because it's a lightweight UI library focused on building user interfaces using components and JSX syntax.

Q2: Explain the core principles of React such as the virtual DOM and component- based architecture.

Ans: - React works on two main ideas: Virtual DOM and Component-based architecture. The Virtual DOM is like a copy of the real webpage. Component-based architecture means the whole UI is made up of small, reusable pieces called components.

Q3: What are the advantages of using React.js in web development?

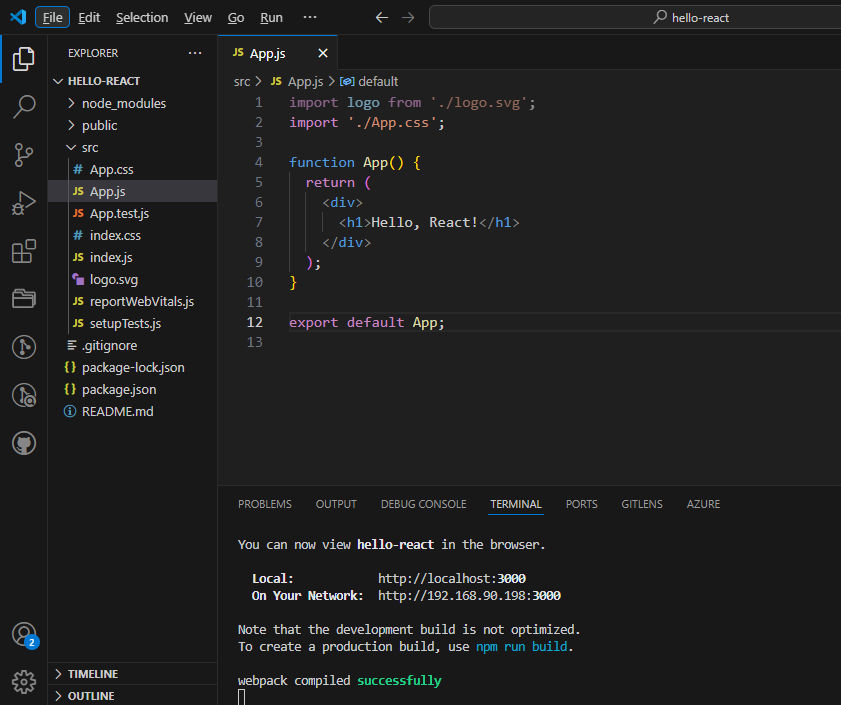
Ans: - React.js is popular in web development because it’s fast, easy to use, and lets developers build dynamic user interfaces using reusable components with efficient updates through the virtual DOM, making apps more responsive, maintainable, and scalable.

**Lab Tasks**

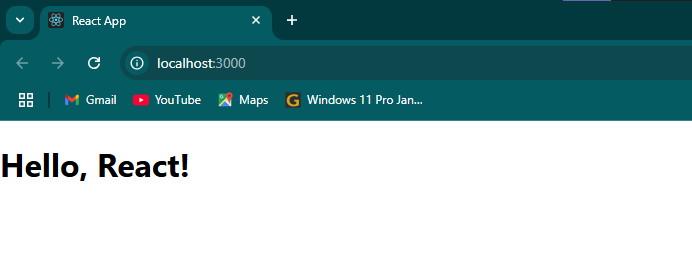
**Task 1 :**

* Set up a new React.js project using create-react-app.
* Create a basic component that displays "Hello, React!" on the web page.

Code:



Output:



**JSX (JavaScript XML)**

Q1: What is JSX in React.js? Why is it used?

Ans: - JSX (JavaScript XML) is a syntax extension for JavaScript used in React.js to describe what the UI should look like. It allows you to write HTML-like code inside JavaScript.

Q2: How is JSX different from regular JavaScript? Can you write JavaScript inside JSX?

Ans: - JSX is a syntax extension that looks like HTML but is used in JavaScript to describe UI in React. Unlike regular JavaScript, JSX allows HTML-like code within JavaScript. You can write JavaScript inside JSX by using curly braces “ {} “ to insert expressions, variables, or function calls.

Q3: Discuss the importance of using curly braces {} in JSX expressions.

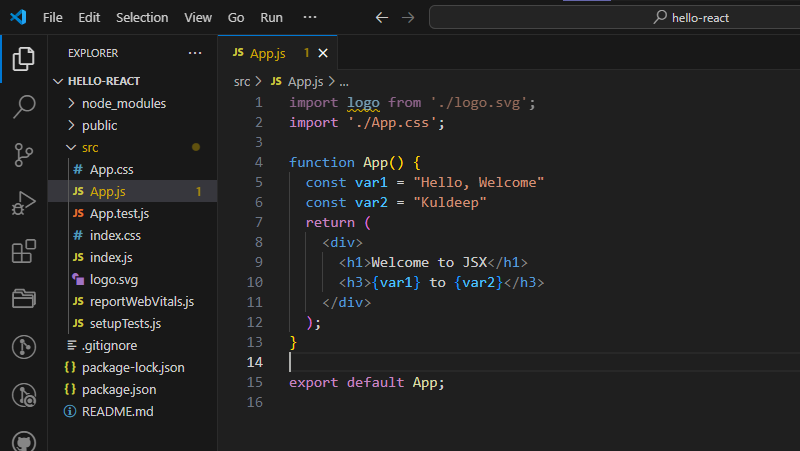
Ans: - Curly braces {} in JSX are important because they allow you to embed JavaScript expressions within JSX code. This enables dynamic content, such as variables, functions, or calculations, to be inserted directly into the rendered UI.

**Lab Tasks**

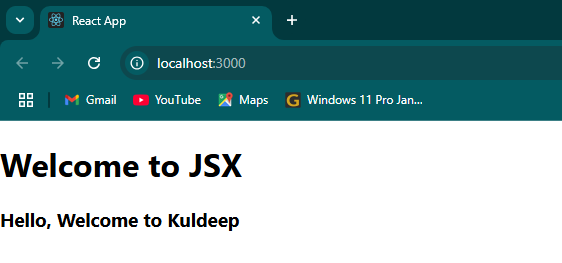
**Task 1 :** Create a React component that renders the following JSX elements:

* A heading with the text "Welcome to JSX".
* A paragraph explaining JSX with dynamic data (use curly braces to insert variables).

Code:



Output:



**Components (Functional & Class Components)**

Q1: What are components in React? Explain the difference between functional components and class components.

Ans: - components are the building blocks of a React application’s UI. A component is a reusable piece of code that defines how a certain part of your user interface.

1. **Functional Components :-** These are plain JavaScript functions that return JSX. Use React Hooks (like useState, useEffect) for state and lifecycle features.

Example:- function Welcome(props) {

return <h1>Hello, {props.name}</h1>;

}

1. **Class Components :-** These are ES6 classes that extend React.Component.

Example:- class Welcome extends React.Component {

render() {

return <h1>Hello, {this.props.name}</h1>;

}

}

Q2: How do you pass data to a component using props?

Ans: - React in pass data to a component using props. Props are like arguments you pass to a function. they allow you to customize components by supplying external data.

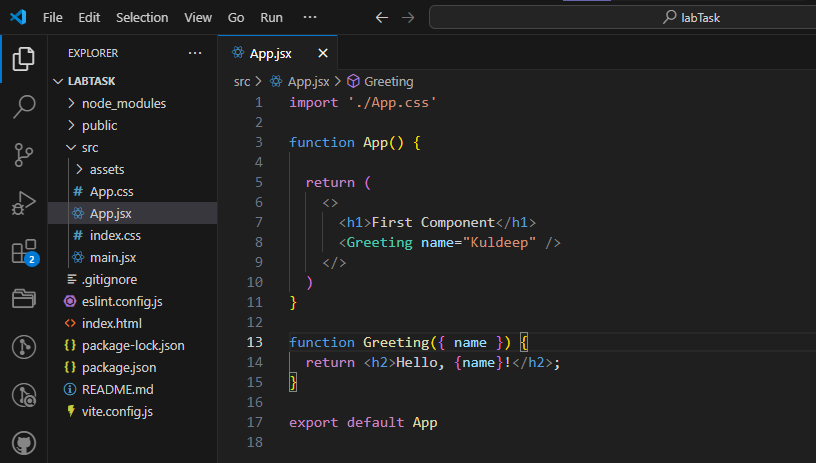
Q3: What is the role of render() in class components?

Ans: - The render() method in React class components is responsible for returning the JSX that defines the UI of the component. It is called automatically whenever the component's state or props change.

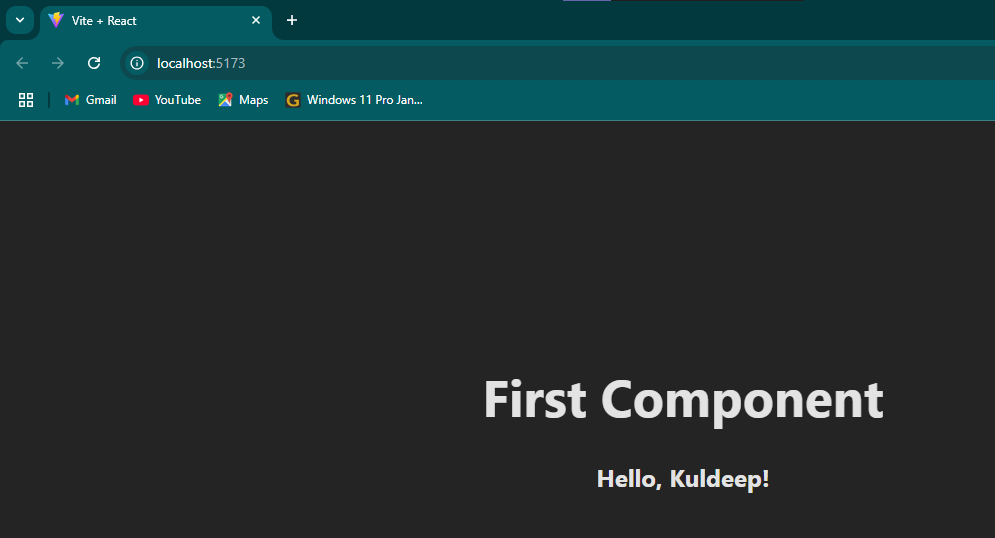
**Lab Tasks**

**Task 1 :** Create a functional component Greeting that accepts a name as a prop and displays "Hello, [name]!".

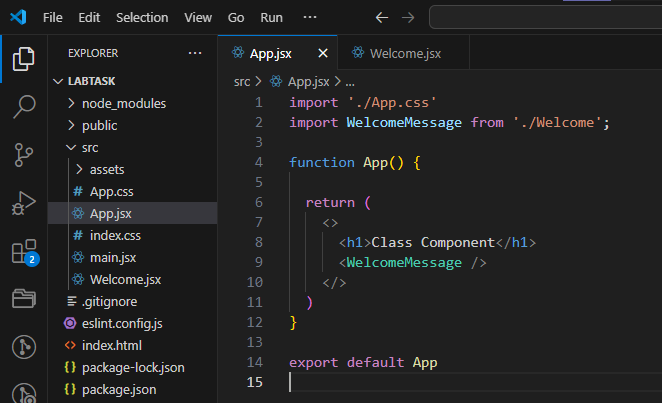
Code:

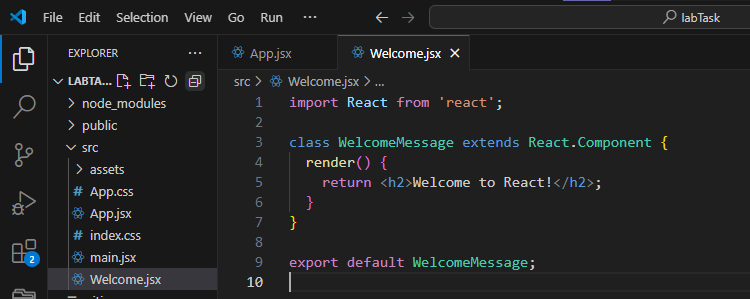


Output:

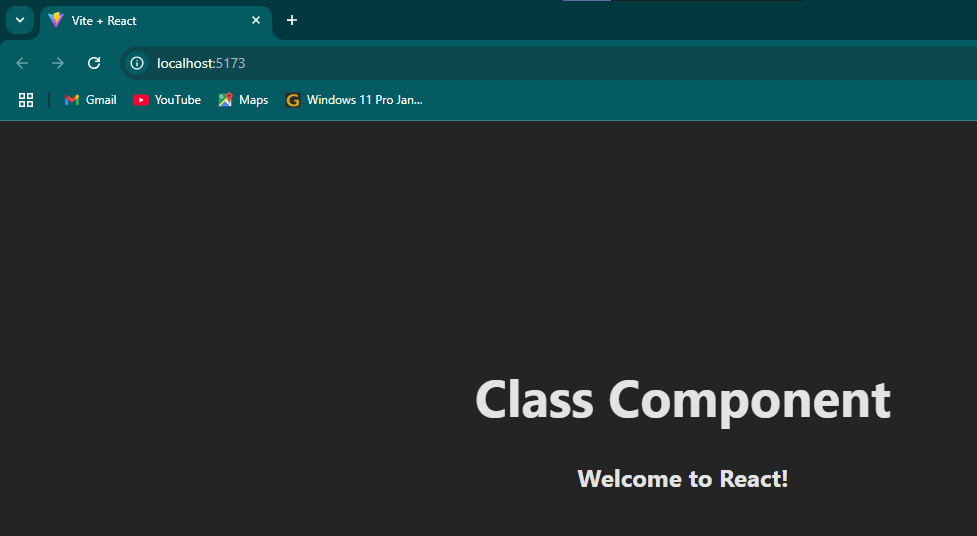


**Task 2 :** Create a class component WelcomeMessage that displays "Welcome to React!" and a render() method.

Code: 



Output:



**Props and State**

Q1: What are props in React.js? How are props different from state?

Ans: - props (short for properties) are read-only inputs passed from a parent to a child component, used to configure or customize components, while state is mutable, internal data managed within a component that can change over time and trigger re-renders props are external and fixed by the parent.

Q2: Explain the concept of state in React and how it is used to manage component data.

Ans: - State represents the local, mutable data of a component. Unlike props, which are passed from parent to child and are read-only, state is managed within the component. In functional components, React provides the useState hook.

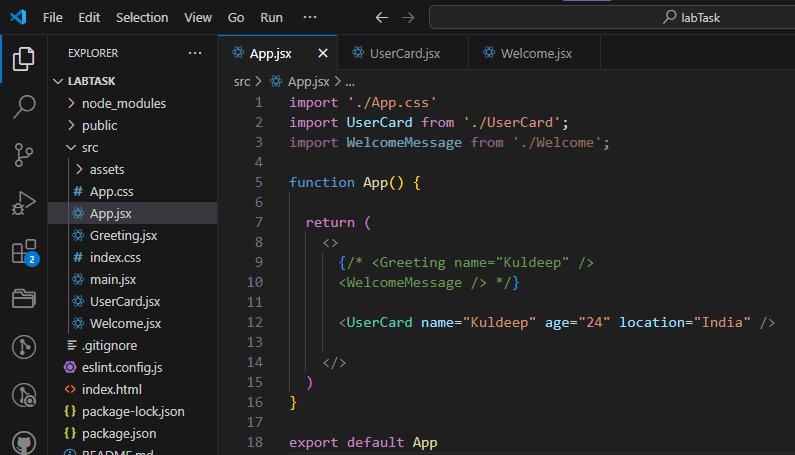
Q3: Why is this.setState() used in class components, and how does it work?

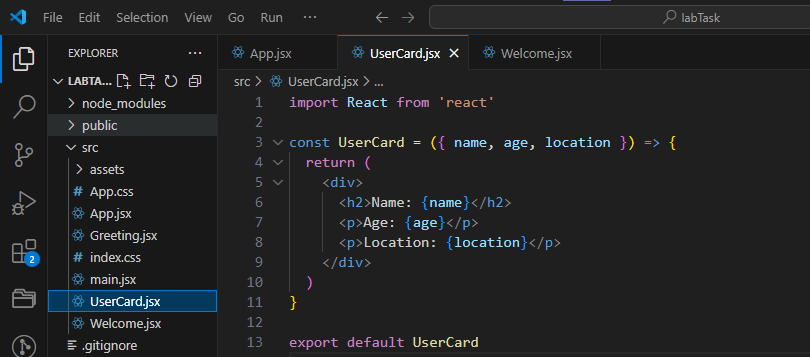
Ans: - class components, this.setState() is used to update the component's state and trigger a re-render with the new state values.

**Lab Tasks**

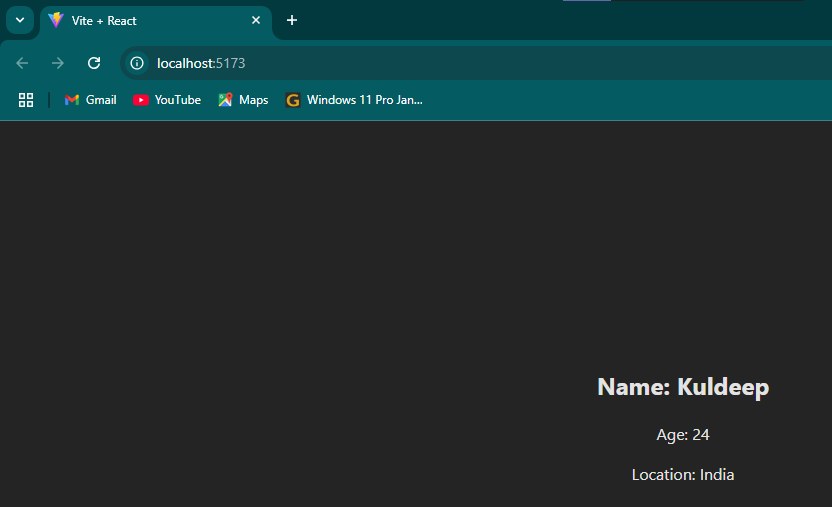
**Task 1 :** Create a React component UserCard that accepts name, age, and location as props and displays them in a card format.

Code:



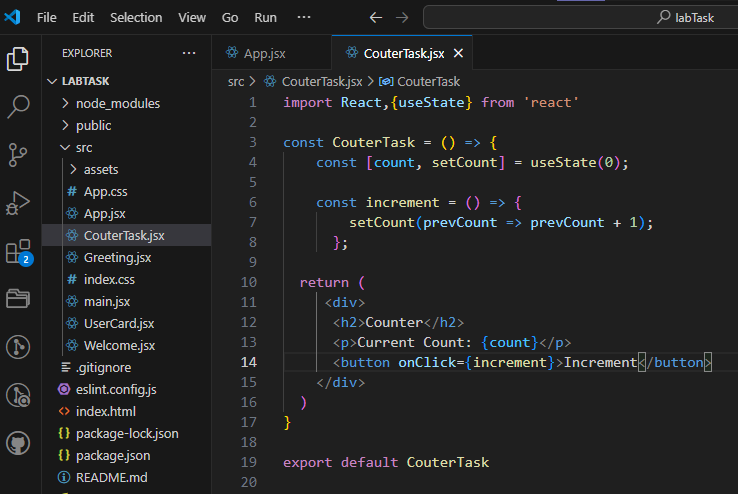


Output:



**Task 2 :** Create a Counter component with a button that increments a count value using React state. Display the current count on the screen.

Code:



Output:

