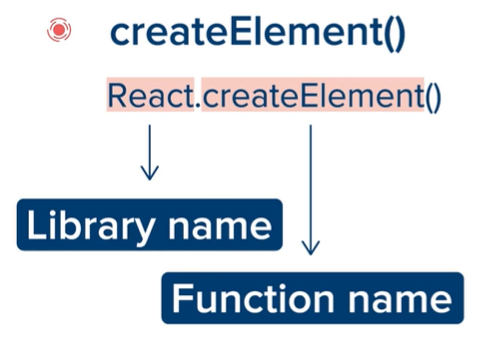


Links to add ReactJS in the body of HTML file: It consists of React and React-dom

<script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

**createElement() function:**



Render(): Render is a ReactDOM function.

CODES:

**App.js**

const heading = React.createElement(

    'h1',  *//type of HTML element we want to create*

    { id: 'abc' },  *//all attributes that we want HTML element to have*

    'This is a heading.'  *//content*

);

ReactDOM.render(

    heading,

    document.getElementById('react-container')

);

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

</head>

<body>

    <div id="react-container"></div>

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script type="text/javascript" src="js/app.js"></script>

</body>

</html>

**OUTPUT:**



**To create more than one element inside a single div element:**

**SEE CODES BELOW: Note that order of elements inside ‘box’ is important as it will be shown in that order only.**

**App.js :**

const heading = React.createElement(

    'h1',  *//type of HTML element we want to create*

    { id: 'abc' },  *//all attributes that we want HTML element to have*

    'This is a heading.',  *//content*

);

const paragraph = React.createElement(

    'p',

    null,

    'This is a text paragraph created by React.'

);

const box = React.createElement(

    'div',

    { className: 'box' }, *//can not use 'class' as it is already a reserved keyword.*

    heading,              *//first, heading will be shown*

    paragraph             *//second, paragraph will be shown*

);

ReactDOM.render(

    box,

    document.getElementById('react-container')

);

**Index.html :**

<html>

<head>

    <title>Using ReactJS</title>

</head>

<body>

    <div id="react-container"></div>

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

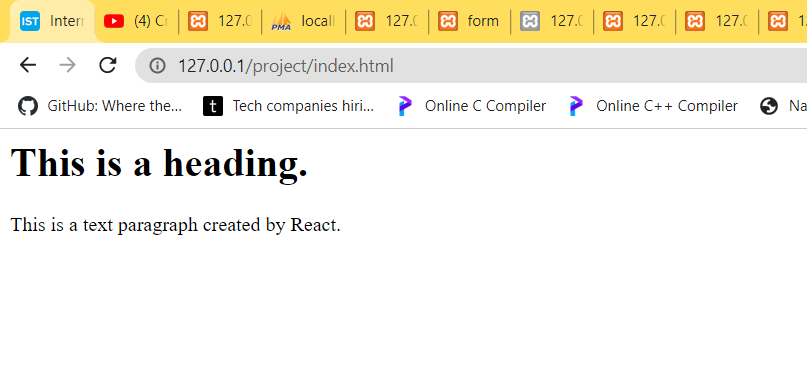
    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

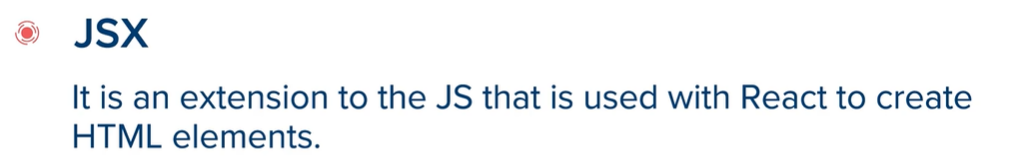
    <script type="text/javascript" src="js/app.js"></script>

</body>

</html>

**Output:**





**We need to take script tags of react, reactdom and babel and also, a script for adding js file to HTML file. Babel script will be added always after react and reactdom script tags.**

**Index.html :**

<html>

<head>

    <title>Using ReactJS</title>

</head>

<body>

    <div id="react-container"></div>

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**App.js :**

const box = (

    <div className='box'>

        <h1 id='abc'>This is a heading</h1>

        <p>This is a test paragraph which is created by React.

            This paragraph has no attribute and is contained inside a div which is also created by React.</p>

    </div>

);

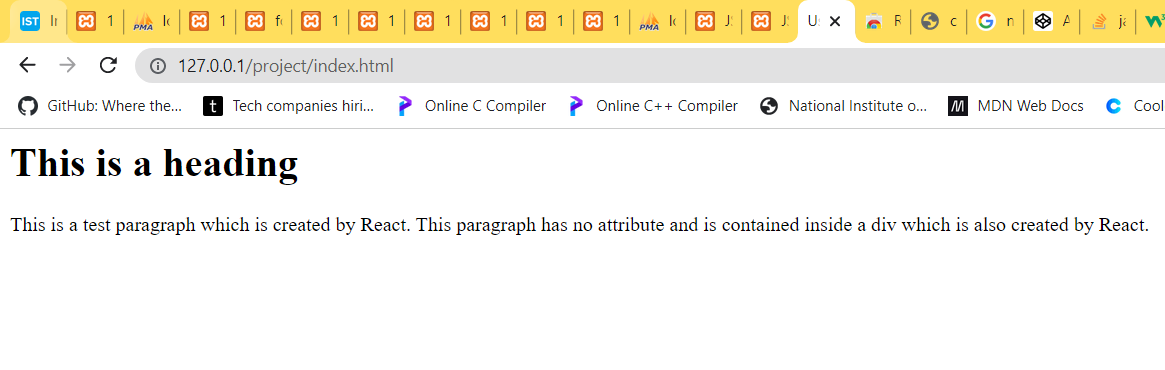
ReactDOM.render(

    box,

    document.getElementById('react-container')

);

**Output:**



**Another example:**

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

</head>

<body>

    <div id="react-container1"></div>

    <div id="react-container2"></div>

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**App.js**

*// const heading = <h1 id='abc'>This is a heading.</h1>*

*// const paragraph = <p>This is a paragraph by me.</p>*

const box = (

    <div className='box'>

        <h1 id='abc'>This is a heading.</h1>

        <p>This is a paragraph by me.</p>

    </div>

);

const NamePlate = (

    <div>

        <h1> ABOUT ME:</h1>

        <p><strong>Nation first, No Compromise</strong></p>

    </div>

)

ReactDOM.render(

    box,

    document.getElementById('react-container1'),

);

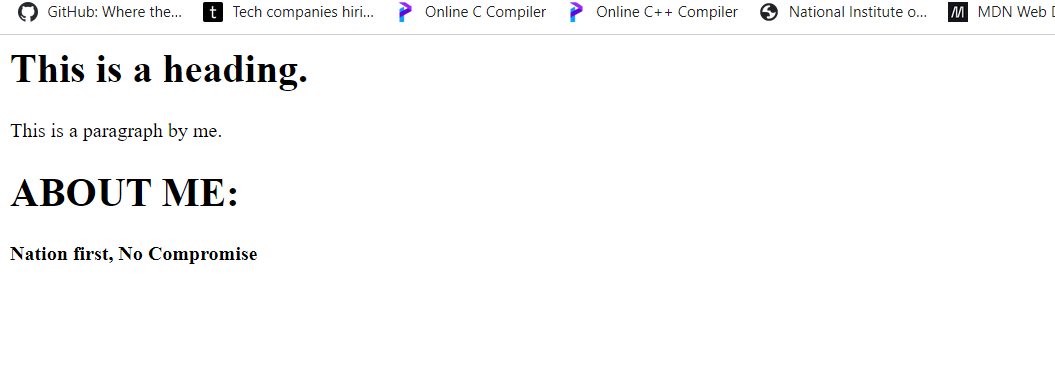
ReactDOM.render(

    NamePlate,

    document.getElementById('react-container2')

);

**OUTPUT:**



**JSX expression helps to let JSX know that we will use js instead of JSX, it is curly brackets {} by which we tell JSX.**

**It can be used in names(strings), mathematical expressions,attribute parsing etc.**

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

</head>

<body>

    <div id="react-container1"></div>

    <div id="react-container2"></div>

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**App.js**

*// const heading = <h1 id='abc'>This is a heading.</h1>*

*// const paragraph = <p>This is a paragraph by me.</p>*

const myattr = 'box';

const box = (

    <div className={myattr}>

        <h1 id='abc'>This is a heading.</h1>

        <p>This is a paragraph by me.</p>

    </div>

);

const name = "Apurv Anand"

const NamePlate = (

    <div>

        <h1> ABOUT ME:{name}</h1>

        <p>8+2 = {8 + 2}</p>

        <p>For bold text in HTML we can use {"<b>"} or {"<strong>"}</p>

        <p><strong>Nation first, No Compromise</strong></p>

    </div>

)

ReactDOM.render(

    box,

    document.getElementById('react-container1'),

);

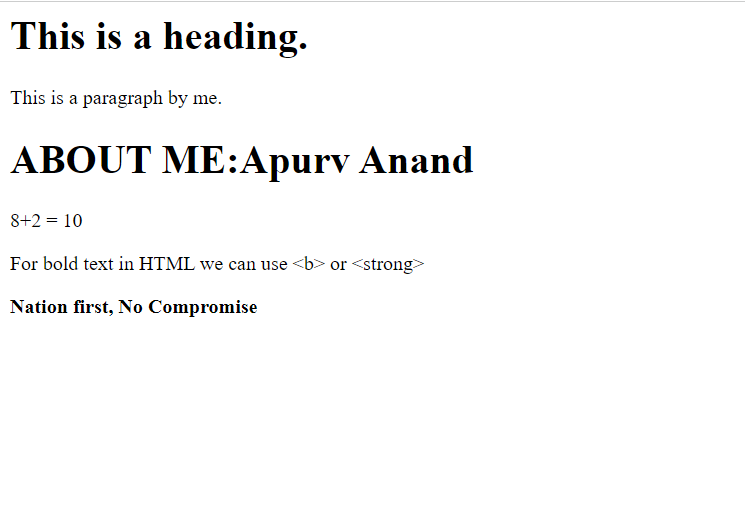
ReactDOM.render(

    NamePlate,

    document.getElementById('react-container2')

);

**Output:**



**Components: are like functions.**

**App.js**

function Box() {

*return* (

        <div className='box'>

            <h1 id='abc'>This is a heading.</h1>

            <p>This is a paragraph by me.</p>

        </div>

    );

}

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

const name = "Apurv Anand"

function NamePlate() {

*return* (<div>

        <h1> ABOUT ME:{name}</h1>

        <p>8+2 = {8 + 2}</p>

        <p>For bold text in HTML we can use {"<b>"} or {"<strong>"}</p>

        <p><strong>Nation first, No Compromise</strong></p>

    </div>);

}

*// const NamePlate = (*

*//     <div>*

*//         <h1> ABOUT ME:{name}</h1>*

*//         <p>8+2 = {8 + 2}</p>*

*//         <p>For bold text in HTML we can use {"<b>"} or {"<strong>"}</p>*

*//         <p><strong>Nation first, No Compromise</strong></p>*

*//     </div>*

*// )*

ReactDOM.render(

    <Box />,

    document.getElementById('react-container1'),

);

ReactDOM.render(

    <NamePlate />,

    document.getElementById('react-container2')

);

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

</head>

<body>

    <div id="react-container1"></div>

    <div id="react-container2"></div>

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**Output: same as just previous.**

**These components can be used to create cluster of components in a box or a small space. Also, CSS written in JSX can be styled in CSS file linked to HTML as shown below.**

**Components have first letter capital in their names. They are accessed as <Components /> and generally , take minimum possible responsibility as possible. These components are combined to form larger components.**

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

    <link rel="stylesheet" href="css/stylingInreact.css">

</head>

<body>

    <div id="react-container1"></div>

*<!-- <div id="react-container2"></div> -->*

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**App.js**

function Box() {

*return* (

        <div className='box'>

            <h1 id='abc'>This is a heading.</h1>

            <p>This is a paragraph by me.</p>

        </div>

    );

}

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

function App() {

*return* (

        <div className='row'>

            <div className='col'>

                <Box />

            </div>

            <div className='col'>

                <Box />

            </div>

        </div>

    );

}

ReactDOM.render(

    <App />,

    document.getElementById('react-container1')

);

*// ReactDOM.render(*

*//     <Box />,*

*//     document.getElementById('react-container1'),*

*// );*

**stylingInreact.css**

#react-container1 {

    background-color: cyan;

    border: solid 2px black;

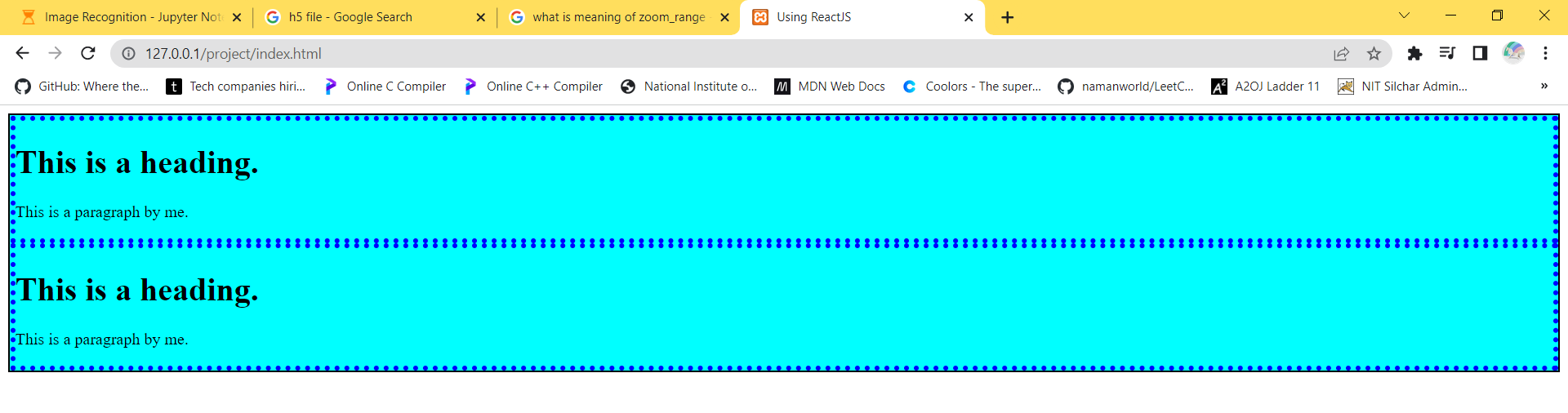
}

.box {

    border: dotted 5px blue;

}

**Output:**



**Above functions in form of arrow functions can be written as:**

**Name of function is thrown of left with ‘const’ and function attribute is converted to ‘=>’ .**

**App.js**

const Box = () => {

*return* (

        <div className='box'>

            <h1 id='abc'>This is a heading.</h1>

            <p>This is a paragraph by me.</p>

        </div>

    );

}

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

const App = () => {

*return* (

        <div className='row'>

            <div className='col'>

                <Box />

            </div>

            <div className='col'>

                <Box />

            </div>

        </div>

    );

}

ReactDOM.render(

    <App />,

    document.getElementById('react-container1')

);

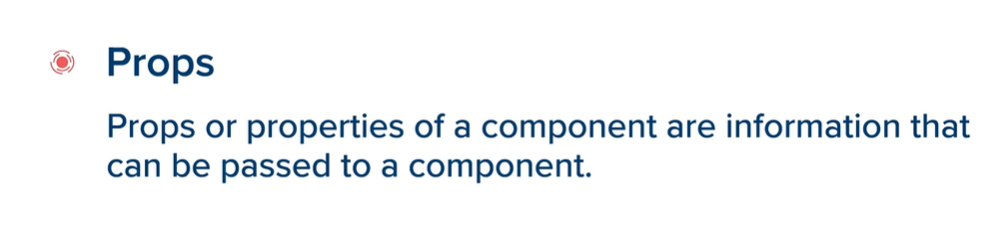
*// ReactDOM.render(*

*//     <Box />,*

*//     document.getElementById('react-container1'),*

*// );*

**Other things remain same.**



**Props refer to properties of components which can be accessed by ‘dot’ operator and ‘curly brackets’ can also be used for printing new strings , attributes etc.**

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

*<!-- <link rel="stylesheet" href="css/stylingInreact.css"> -->*

    <link rel="stylesheet" href="css/bootstrap.min.css">

</head>

<body>

    <div id="react-container1"></div>

*<!-- <div id="react-container2"></div> -->*

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**App.js**

const Box = (*props*) => {

*return* (

        <div className='box'>

            <h1 id='abc'>{*props*.heading}</h1>

            <p>This is a paragraph by {*props*.name}.</p>

        </div>

    );

};

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

const App = () => {

*return* (

        <div className='row'>

            <div className='col'>

                <Box heading="First Heading" name="Apurv Anand" />

            </div>

            <div className='col'>

                <Box heading="Second Heading" name="Apurv Anand\_2112052" />

            </div>

        </div>

    );

};

ReactDOM.render(

    <App />,

    document.getElementById('react-container1')

);

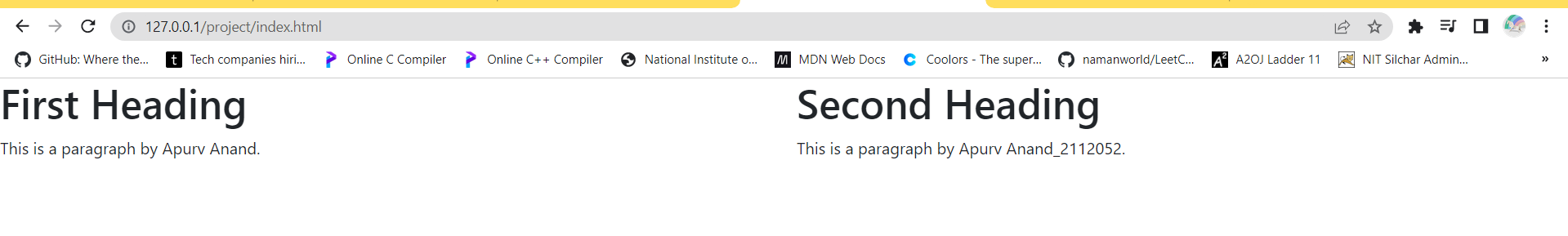
*// ReactDOM.render(*

*//     <Box />,*

*//     document.getElementById('react-container1'),*

*// );*

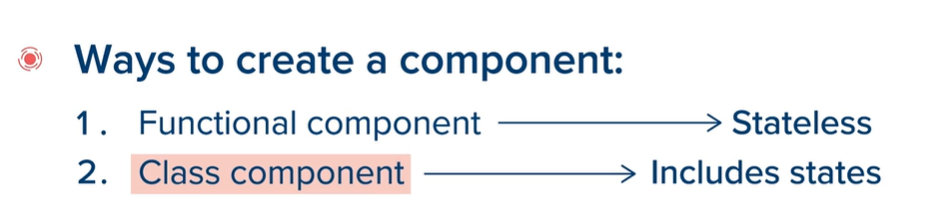
**Output:**





**As in props, for related information , the information always flows from parent component to child component. So, any prop related change can not be made in child component , but must always be made in parent component.**





**In above example as we gave props as input, so, it was a ‘functional component’ as it was not able to handle different states.**

**First we here create a ‘class’ instead of a ‘function’ ; which inherits all properties of React Component using ‘extend’ keyword. And all prop properties which we have input are already present inside the class, but as it is a class and not a function , we can access only using a ‘this’ keyword.**

**App.js**

class Box extends React.Component { *//Box class takes all properties as React component*

    render() {             *//return statement will be copied as it is inside render.*

*//prop can not be accessed directly, they are present inside class*

*//    and can be chosen using 'this' keyword.*

*return* (

            <div className='box' >

                <h1 id='abc'>{this.props.heading}</h1>

                <p>This is a paragraph by {this.props.name}.</p>

            </div>

        );

    }

};

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

const App = () => {

*return* (

        <div className='row'>

            <div className='col'>

                <Box heading="First Heading" name="Apurv Anand" />

            </div>

            <div className='col'>

                <Box heading="Second Heading" name="Apurv Anand\_2112052" />

            </div>

        </div>

    );

};

ReactDOM.render(

    <App />,

    document.getElementById('react-container1')

);

*// ReactDOM.render(*

*//     <Box />,*

*//     document.getElementById('react-container1'),*

*// );*

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

*<!-- <link rel="stylesheet" href="css/stylingInreact.css"> -->*

    <link rel="stylesheet" href="css/bootstrap.min.css">

</head>

<body>

    <div id="react-container1"></div>

*<!-- <div id="react-container2"></div> -->*

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

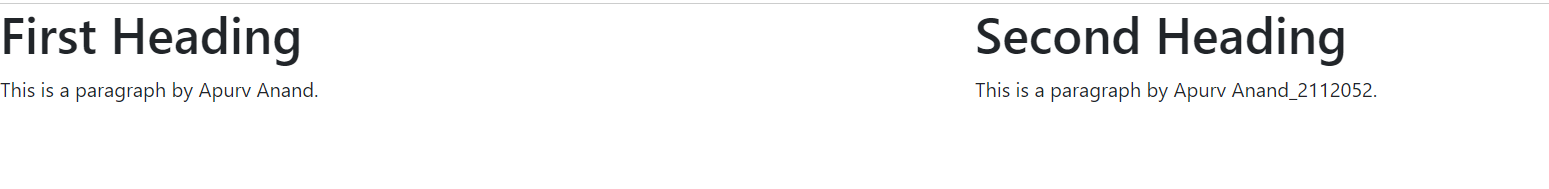
    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**Output:**



**States is also a property to a component but we need to explicitly assign it.**

**Ques: Change color of heading.**

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

    <link rel="stylesheet" href="css/stylingInreact.css">

*<!-- <link rel="stylesheet" href="css/bootstrap.min.css"> -->*

</head>

<body>

    <div id="react-container1"></div>

*<!-- <div id="react-container2"></div> -->*

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**App.js**

class Box extends React.Component { *//Box class takes all properties as React component*

    state = {

        color: "black"

    };

    changeColor(*color*) {

        this.setState({ color: *color* });

    }

    render() {             *//return statement will be copied as it is inside render.*

*//prop can not be accessed directly, they are present inside class*

*//    and can be chosen using 'this' keyword.*

*return* (

            <div className='box' >

                <h1 className={this.state.color}>{this.props.heading}</h1>

                <p>This is a paragraph by {this.props.name}.</p>

                <button onClick={() => this.changeColor("red")}>Red</button>

                <button onClick={() => this.changeColor("yellow")}>Yellow</button>

                <button onClick={() => this.changeColor("blue")}>Blue</button>

                <button onClick={() => this.changeColor("green")}>Green</button>

            </div>

        );

    }

};

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

const App = () => {

*return* (

        <div className='row'>

            <div className='col'>

                <Box heading="First Heading" name="Apurv Anand" />

            </div>

            <div className='col'>

                <Box heading="Second Heading" name="Apurv Anand" />

            </div>

            <div className='col'>

                <Box heading="Third Heading" name="Apurv Anand" />

            </div>

            <div className='col'>

                <Box heading="Fourth Heading" name="Apurv Anand" />

            </div>

        </div>

    );

};

ReactDOM.render(

    <App />,

    document.getElementById('react-container1')

);

*// ReactDOM.render(*

*//     <Box />,*

*//     document.getElementById('react-container1'),*

*// );*

**stylingInreact.css**

.row {

    display: flex;

    flex-wrap: wrap;

}

.col {

    flex-basis: 50%;

    box-sizing: border-box;

    border: solid 2px coral;

}

.box {

    padding: 10px;

}

button {

    margin: 8px;

}

.red {

    color: red;

}

.yellow {

    color: yellow;

}

.blue {

    color: blue;

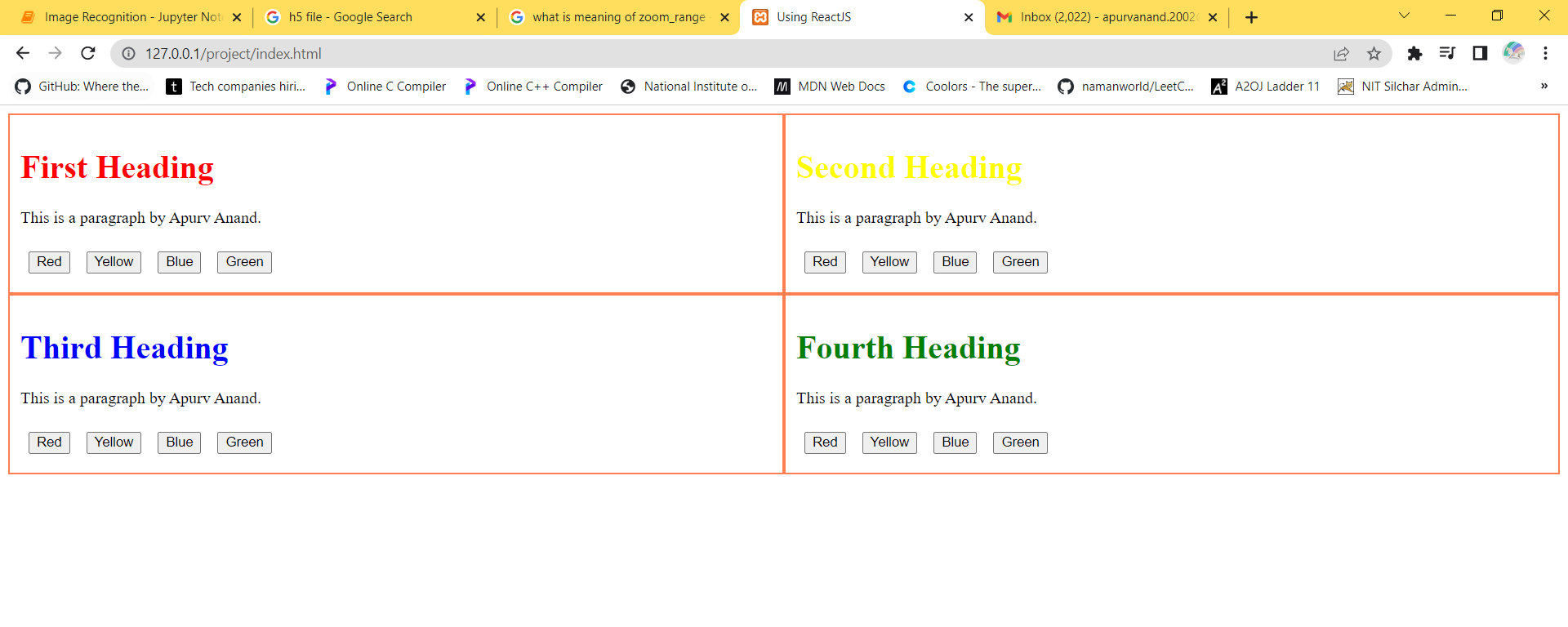
}

.green {

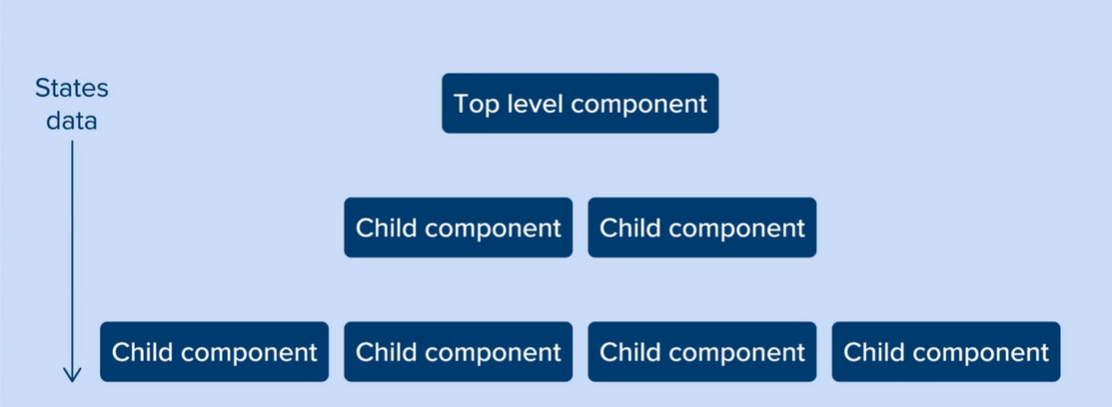
    color: green;

}

**OUTPUT:**







**It is uni-directional top-down data flow.If the data is stored on top-level component , its information can be sent to any lower child component.**

**It can be done by converting component state to application state on app component ,so, app component must first be converted to class and then component state must be converted to application state.**

**Application state is nothing but a state in which all information is contained in top level component.**

**Setting color is now a task in app class (after converting to class) and hence, set color must be declared in app class and also box is now not needed to be a class so ,it can be converted back to a function ,so , prop will be used, ‘this’ keyword is used in class so, it must be removed and in its place ‘props’ must be used if it is declared for props in the arrow function argument.**

**Important:**

**We need to bind ‘this’ as when we will access id , heading etc inside boxes , then ‘this’ which was earlier pointing at boxes object ; will point at states inside boxes i.e id, heading etc ; so, to keep it bound to point at boxes only and not inside it, we bindthis using following code:**

changeColor={this.changeColor.bind(this)}

**Important:**

**In return part, when we try to use more than 1 div then it throws an error as it always needs to have single parent element and all other things to be wrapped in it , so, to add more than 1 div, we must wrap all those divs inside another div declared outside all those divs as parent div.**

**ALL ABOVE THINGS CAN BE SEEN IN FOLLOWING CODES:**

**Index.html**

<html>

<head>

    <title>Using ReactJS</title>

    <link rel="stylesheet" href="css/stylingInreact.css">

*<!-- <link rel="stylesheet" href="css/bootstrap.min.css"> -->*

</head>

<body>

    <div id="react-container1"></div>

*<!-- <div id="react-container2"></div> -->*

    <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>

    <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

    <script src="https://unpkg.com/babel-standalone@6.26.0/babel.js"></script>

    <script type="text/babel" src="js/app.js"></script>

</body>

</html>

**stylingInreact.css**

.row {

    display: flex;

    flex-wrap: wrap;

}

.col {

    flex-basis: 50%;

    box-sizing: border-box;

    border: solid 2px coral;

}

.box {

    padding: 10px;

}

button {

    margin: 8px;

}

.red {

    color: red;

}

.yellow {

    color: yellow;

}

.blue {

    color: blue;

}

.green {

    color: green;

}

.stats {

    border: solid 2px cyan;

    margin: 5px;

}

**App.js**

const Box = (*props*) => {

*//return statement will be copied as it is inside render.*

*//prop can not be accessed directly, they are present inside class*

*//    and can be chosen using 'this' keyword.*

*return* (

        <div className='box' >

            <h1 className={*props*.color}>{*props*.heading}</h1>

            <button onClick={() => *props*.changeColor(*props*.id, "red")}>Red</button>

            <button onClick={() => *props*.changeColor(*props*.id, "yellow")}>Yellow</button>

            <button onClick={() => *props*.changeColor(*props*.id, "blue")}>Blue</button>

            <button onClick={() => *props*.changeColor(*props*.id, "green")}>Green</button>

        </div>

    );

};

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

*// Stats function which we have used inside App is also a function as Box.*

const Stats = (*props*) => {

    let boxes = *props*.boxes;

    let black\_count = 0, red\_count = 0, yellow\_count = 0, blue\_count = 0, green\_count = 0;

    boxes.forEach(*box* => {

*if* (*box*.color == "black") {

            black\_count++;

        }

*if* (*box*.color == "red") {

            red\_count++;

        }

*if* (*box*.color == "yellow") {

            yellow\_count++;

        }

*if* (*box*.color == "blue") {

            blue\_count++;

        }

*if* (*box*.color == "green") {

            green\_count++;

        }

    })

*return* (

        <div className='stats'>

            Total heading:color count

            <div>Black:{black\_count}</div>

            <div>Red:{red\_count}</div>

            <div>Yellow:{yellow\_count}</div>

            <div>Blue:{blue\_count}</div>

            <div>Green:{green\_count}</div>

        </div>

    )

}

class App extends React.Component {

    state = {

        boxes: [     *//an array named 'boxes' storing information of all boxes.*

            {

                id: 1,

                heading: "First Heading",

                color: "black"

            },

            {

                id: 2,

                heading: "Second Heading",

                color: "black"

            },

            {

                id: 3,

                heading: "Third Heading",

                color: "black"

            },

            {

                id: 4,

                heading: "Fourth Heading",

                color: "black"

            },

            {

                id: 5,

                heading: "Fifth Heading",

                color: "black"

            },

            {

                id: 6,

                heading: "Sixth Heading",

                color: "black"

            },

            {

                id: 7,

                heading: "Seventh Heading",

                color: "black"

            }

        ]

    }

    changeColor(*id*, *color*) {        *// this is a function.*

        let boxes = this.state.boxes;

        boxes[*id* - 1].color = *color*;

        this.setState({

            boxes: boxes

        });

    }

    render() {

*return* (

            <div>

                <div className='row'>

                    {this.state.boxes.map(*box* =>

                        <div className='col'>

                            <Box

                                id={*box*.id}

                                heading={*box*.heading}

                                color={*box*.color}

                                changeColor={this.changeColor.bind(this)}

                            />

                        </div>

                    )}

                </div>

                <div className='row'>

                    <Stats boxes={this.state.boxes} />

                </div>

            </div>

        );

    }

};

ReactDOM.render(

    <App />,

    document.getElementById('react-container1')

);

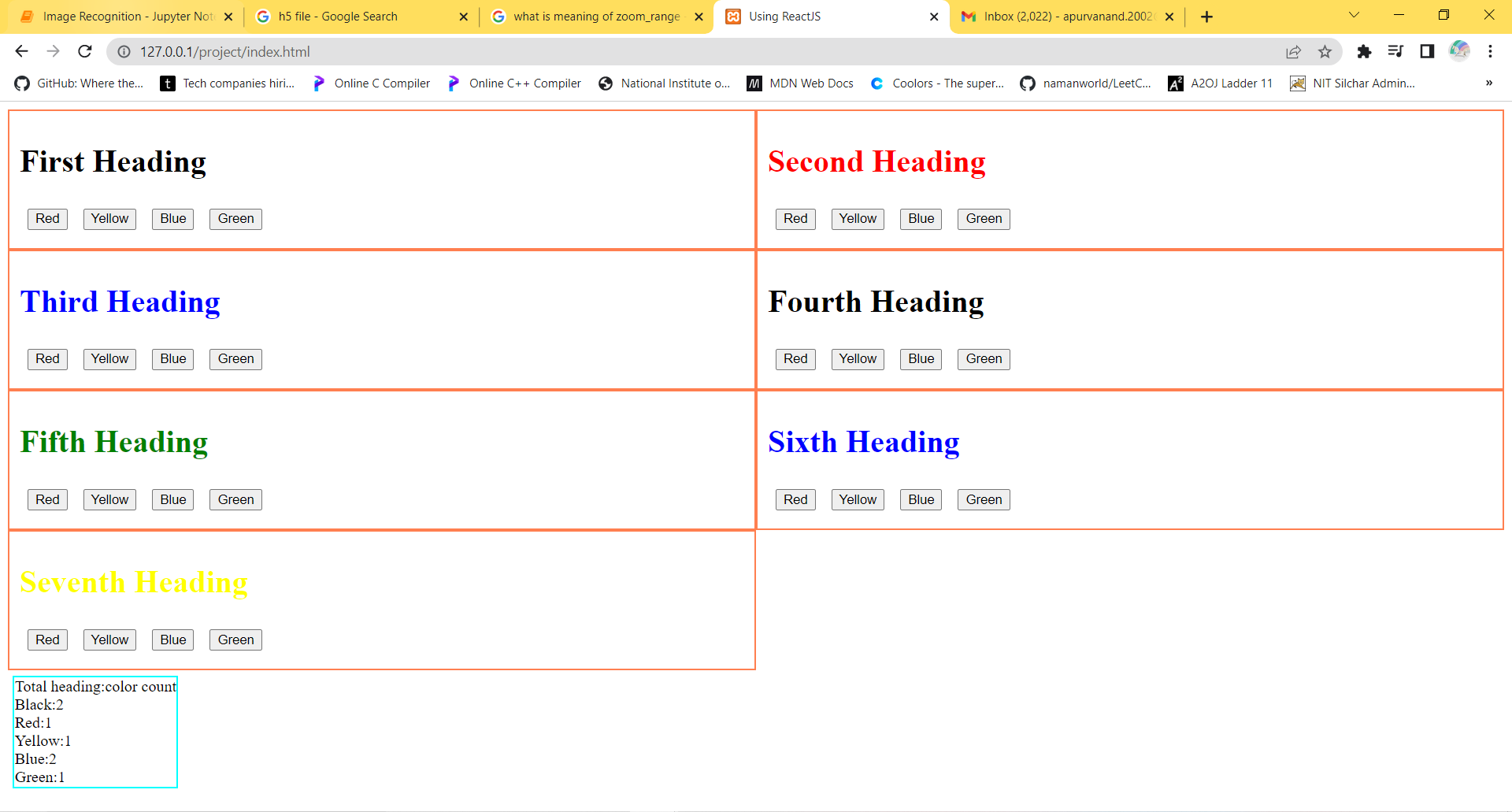
*// ReactDOM.render(*

*//     <Box />,*

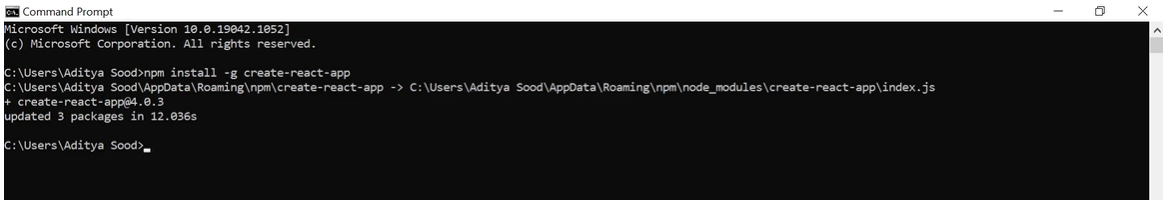
*//     document.getElementById('react-container1'),*

*// );*

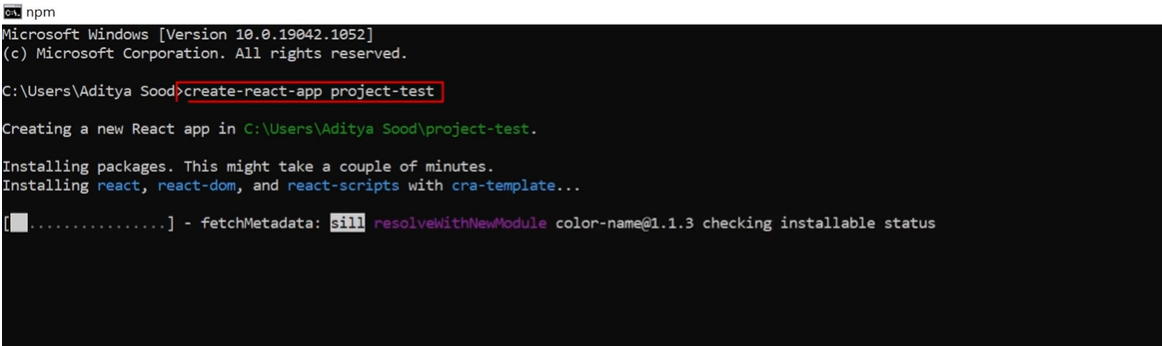
**OUTPUT:**



**To install react app:**

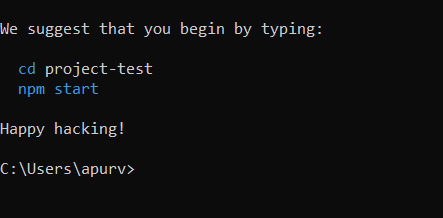


**To create react project:**



**‘project-test’ is to be replaced by folder name to be given.**

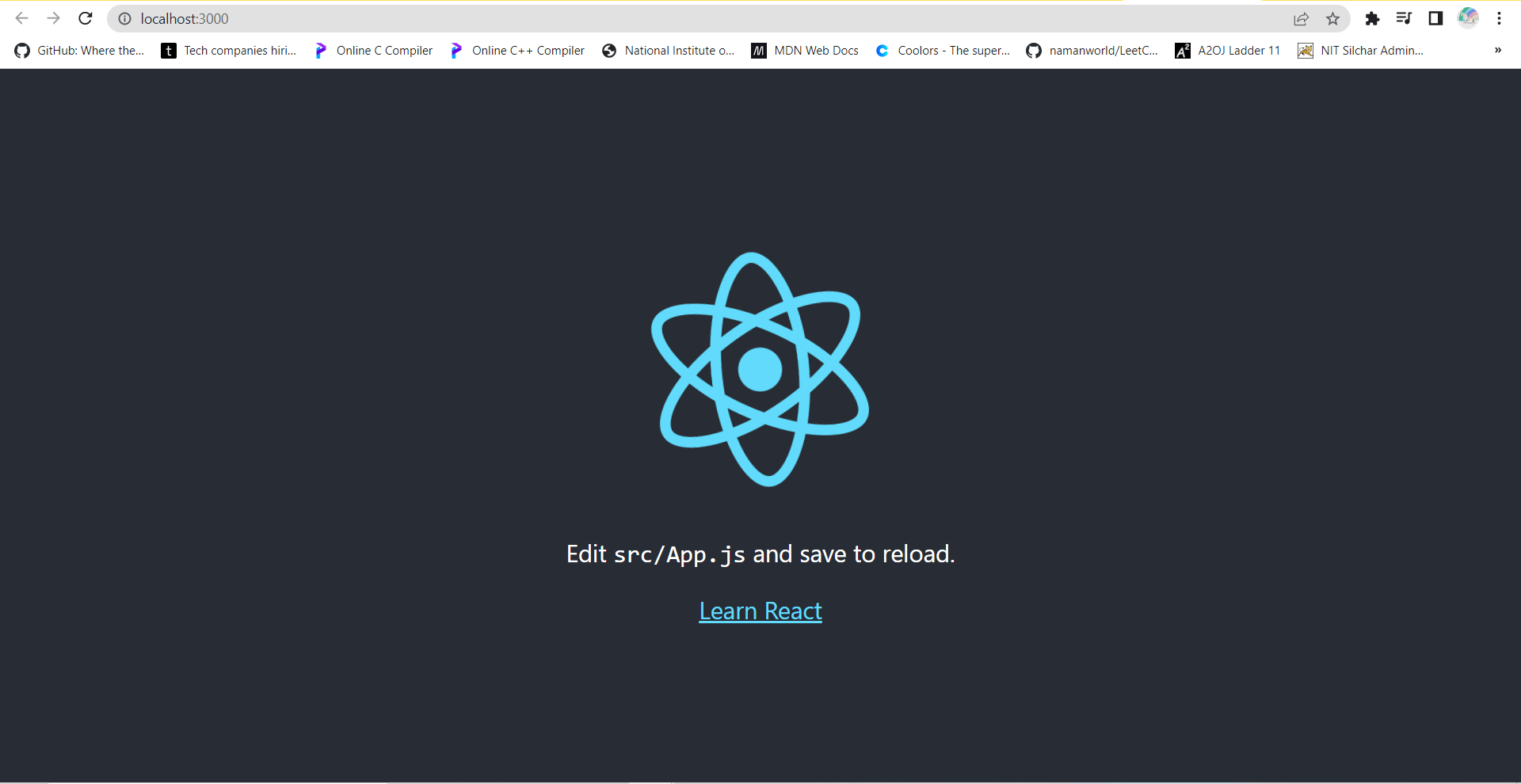
**After folder creation:**



**Cd : changes directory to our folder**

**Npm: node package manager uses Babel to convert JSX to JS.**

**After entering above codes and giving access.**



**This is photo of browser which will show our resulting webpage.**

**Now, enter folder name in search box on right side of windows sign. You will see various files in it.**

**We will open ‘App.js’ ;note the spelling and capital ‘A’ in name; and ‘index.css’ in our VS code compiler. Copy your js and css codes into App.js and index.css respectively.**

**Also, note that ReactDOM.render() part is not to be copied inside ‘App.js’**

**As it is assumed to be already present in App.js.**

**We need to add these lines in App.js :**

**First line of App.js must be:**

*import* React *from* 'react';

**Last line of App.js must be:**

*export* *default* App;

**then only App.js will run.**

**We can see our outputs on window which looks like previous image.**

**App.js**

*import* React *from* 'react';

const Box = (*props*) => {

*//return statement will be copied as it is inside render.*

*//prop can not be accessed directly, they are present inside class*

*//    and can be chosen using 'this' keyword.*

*return* (

    <div className='box' >

      <h1 className={*props*.color}>{*props*.heading}</h1>

      <button onClick={() => *props*.changeColor(*props*.id, "red")}>Red</button>

      <button onClick={() => *props*.changeColor(*props*.id, "yellow")}>Yellow</button>

      <button onClick={() => *props*.changeColor(*props*.id, "blue")}>Blue</button>

      <button onClick={() => *props*.changeColor(*props*.id, "green")}>Green</button>

    </div>

  );

};

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

*// Stats function which we have used inside App is also a function as Box.*

const Stats = (*props*) => {

  let boxes = *props*.boxes;

  let black\_count = 0, red\_count = 0, yellow\_count = 0, blue\_count = 0, green\_count = 0;

  boxes.forEach(*box* => {

*if* (*box*.color == "black") {

      black\_count++;

    }

*if* (*box*.color == "red") {

      red\_count++;

    }

*if* (*box*.color == "yellow") {

      yellow\_count++;

    }

*if* (*box*.color == "blue") {

      blue\_count++;

    }

*if* (*box*.color == "green") {

      green\_count++;

    }

  })

*return* (

    <div className='stats'>

      Total heading:color count

      <div>Black:{black\_count}</div>

      <div>Red:{red\_count}</div>

      <div>Yellow:{yellow\_count}</div>

      <div>Blue:{blue\_count}</div>

      <div>Green:{green\_count}</div>

    </div>

  )

}

class App extends React.Component {

  state = {

    boxes: [     *//an array named 'boxes' storing information of all boxes.*

      {

        id: 1,

        heading: "First Heading",

        color: "black"

      },

      {

        id: 2,

        heading: "Second Heading",

        color: "black"

      },

      {

        id: 3,

        heading: "Third Heading",

        color: "black"

      },

      {

        id: 4,

        heading: "Fourth Heading",

        color: "black"

      },

      {

        id: 5,

        heading: "Fifth Heading",

        color: "black"

      },

      {

        id: 6,

        heading: "Sixth Heading",

        color: "black"

      },

      {

        id: 7,

        heading: "Seventh Heading",

        color: "black"

      }

    ]

  }

  changeColor(*id*, *color*) {        *// this is a function.*

    let boxes = this.state.boxes;

    boxes[*id* - 1].color = *color*;

    this.setState({

      boxes: boxes

    });

  }

  render() {

*return* (

      <div>

        <div className='row'>

          {this.state.boxes.map(*box* =>

            <div className='col'>

              <Box

                id={*box*.id}

                heading={*box*.heading}

                color={*box*.color}

                changeColor={this.changeColor.bind(this)}

              />

            </div>

          )}

        </div>

        <div className='row'>

          <Stats boxes={this.state.boxes} />

        </div>

      </div>

    );

  }

};

*export* *default* App;

**index.css**

.row {

  display: flex;

  flex-wrap: wrap;

}

.col {

  flex-basis: 50%;

}

.box {

  padding: 10px;

}

button {

  margin: 8px;

}

.red {

  color: red;

}

.yellow {

  color: yellow;

}

.blue {

  color: blue;

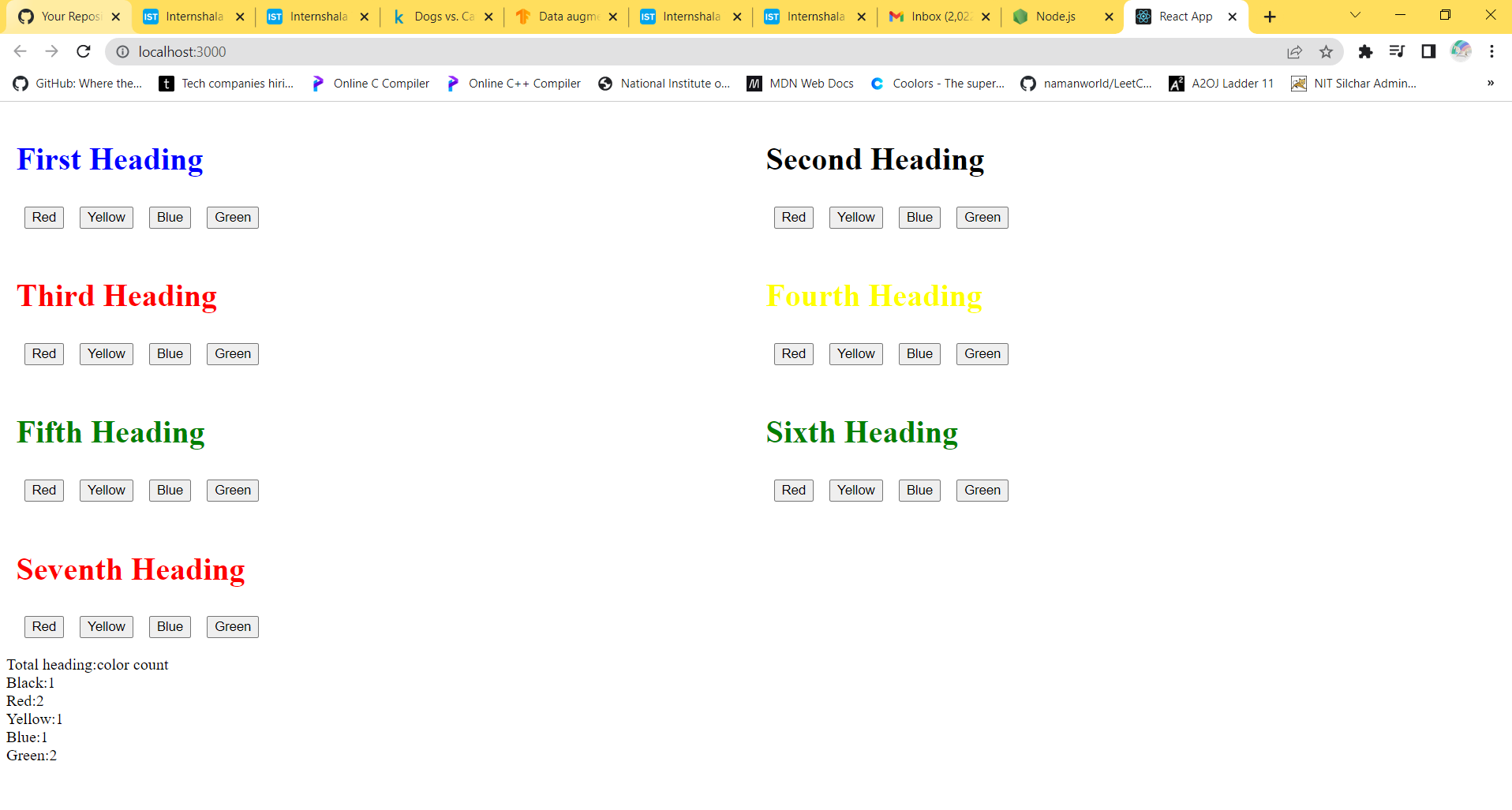
}

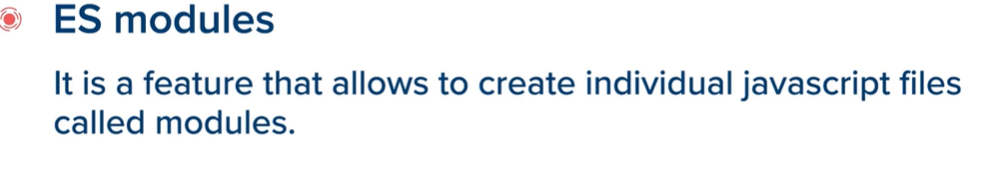
.green {

  color: green;

}

**Output on React App window:**





**It is feature of create react app.**

**We will break our long codes into small codes and store it in different files each called as ‘modules’. We can import each module so that it can be used as and when required.**

**Each module is to be made in src file of npm folder which we have created before.**

**Every file must have:**

**First line must be:**

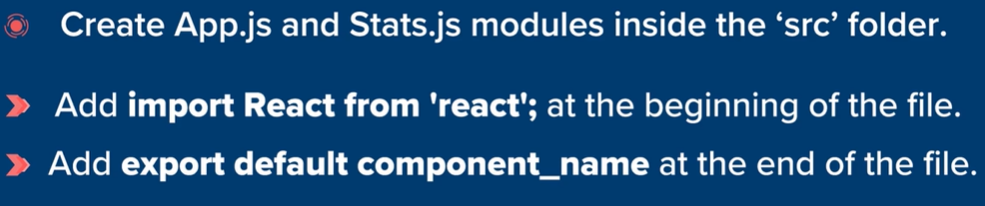
*import* React *from* 'react';

**Last line must be:**

*export* *default* App ; -> -> -> App will be replaced by name of file

**then only App.js will run.**

**For our example we will:**



**Box.js**

*import* React *from* 'react';

const Box = (*props*) => {

*//return statement will be copied as it is inside render.*

*//prop can not be accessed directly, they are present inside class*

*//    and can be chosen using 'this' keyword.*

*return* (

        <div className='box' >

            <h1 className={*props*.color}>{*props*.heading}</h1>

            <button onClick={() => *props*.changeColor(*props*.id, "red")}>Red</button>

            <button onClick={() => *props*.changeColor(*props*.id, "yellow")}>Yellow</button>

            <button onClick={() => *props*.changeColor(*props*.id, "blue")}>Blue</button>

            <button onClick={() => *props*.changeColor(*props*.id, "green")}>Green</button>

        </div>

    );

};

*export* *default* Box;

**Stats.js**

*import* React *from* 'react';

*// Stats function which we have used inside App is also a function as Box.*

const Stats = (*props*) => {

    let boxes = *props*.boxes;

    let black\_count = 0, red\_count = 0, yellow\_count = 0, blue\_count = 0, green\_count = 0;

    boxes.forEach(*box* => {

*if* (*box*.color == "black") {

            black\_count++;

        }

*if* (*box*.color == "red") {

            red\_count++;

        }

*if* (*box*.color == "yellow") {

            yellow\_count++;

        }

*if* (*box*.color == "blue") {

            blue\_count++;

        }

*if* (*box*.color == "green") {

            green\_count++;

        }

    })

*return* (

        <div className='stats'>

            Total heading:color count

            <div>Black:{black\_count}</div>

            <div>Red:{red\_count}</div>

            <div>Yellow:{yellow\_count}</div>

            <div>Blue:{blue\_count}</div>

            <div>Green:{green\_count}</div>

        </div>

    )

}

*export* *default* Stats;

**App.js (which is importing Box.js and Stats.js for use)**

*import* logo *from* './logo.svg';

*import* './index.css';

*import* React *from* 'react';

*import* Box *from* './Box';

*import* Stats *from* './Stats';

*// const box = (*

*//     <div className='box'>*

*//         <h1 id='abc'>This is a heading.</h1>*

*//         <p>This is a paragraph by me.</p>*

*//     </div>*

*// );*

class App extends React.Component {

  state = {

    boxes: [     *//an array named 'boxes' storing information of all boxes.*

      {

        id: 1,

        heading: "First Heading",

        color: "black"

      },

      {

        id: 2,

        heading: "Second Heading",

        color: "black"

      },

      {

        id: 3,

        heading: "Third Heading",

        color: "black"

      },

      {

        id: 4,

        heading: "Fourth Heading",

        color: "black"

      },

      {

        id: 5,

        heading: "Fifth Heading",

        color: "black"

      },

      {

        id: 6,

        heading: "Sixth Heading",

        color: "black"

      },

      {

        id: 7,

        heading: "Seventh Heading",

        color: "black"

      }

    ]

  }

  changeColor(*id*, *color*) {        *// this is a function.*

    let boxes = this.state.boxes;

    boxes[*id* - 1].color = *color*;

    this.setState({

      boxes: boxes

    });

  }

  render() {

*return* (

      <div>

        <div className='row'>

          {this.state.boxes.map(*box* =>

            <div className='col'>

              <Box

                id={*box*.id}

                heading={*box*.heading}

                color={*box*.color}

                changeColor={this.changeColor.bind(this)}

              />

            </div>

          )}

        </div>

        <div className='row'>

          <Stats boxes={this.state.boxes} />

        </div>

      </div>

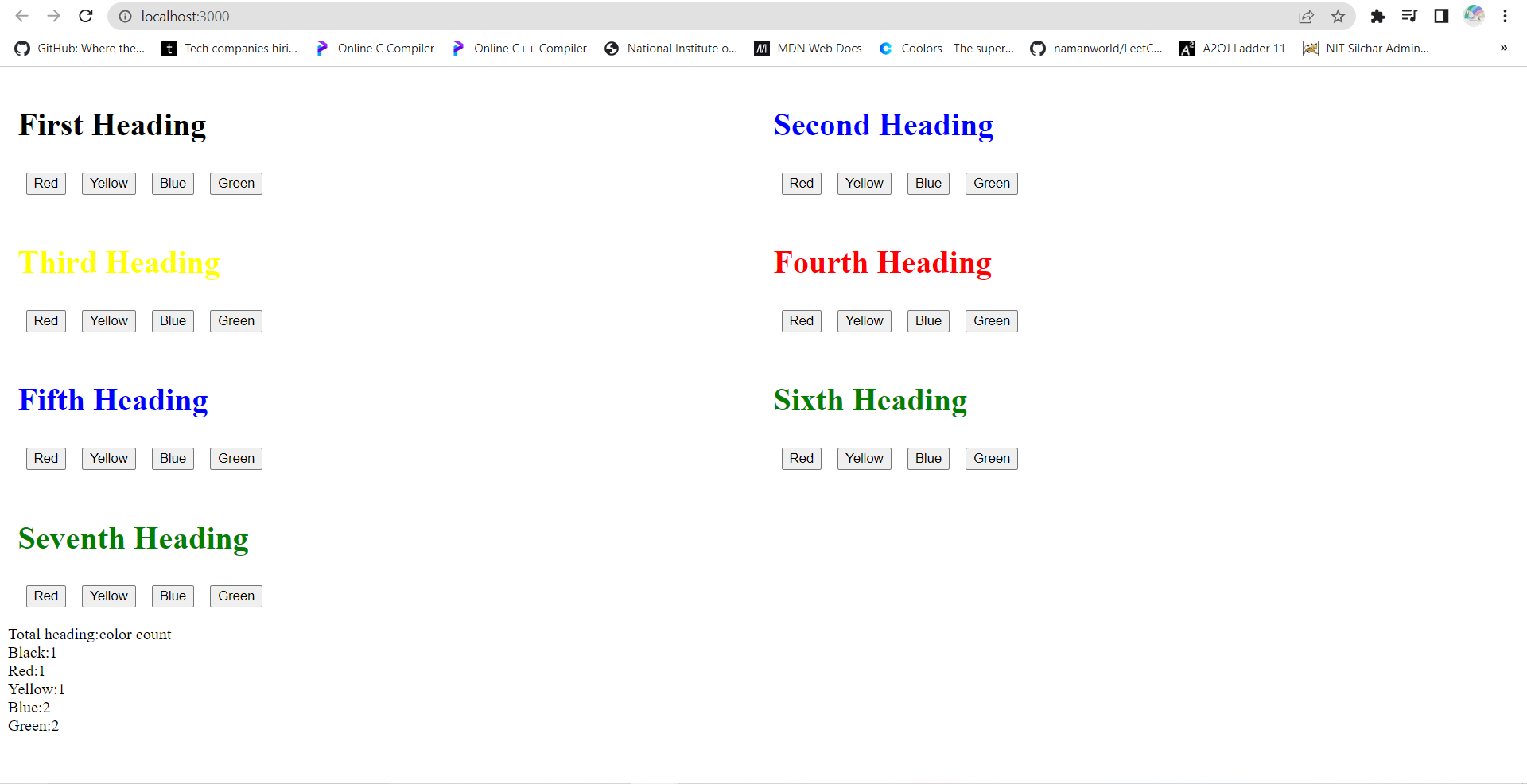
    );

  }

};

*export* *default* App;

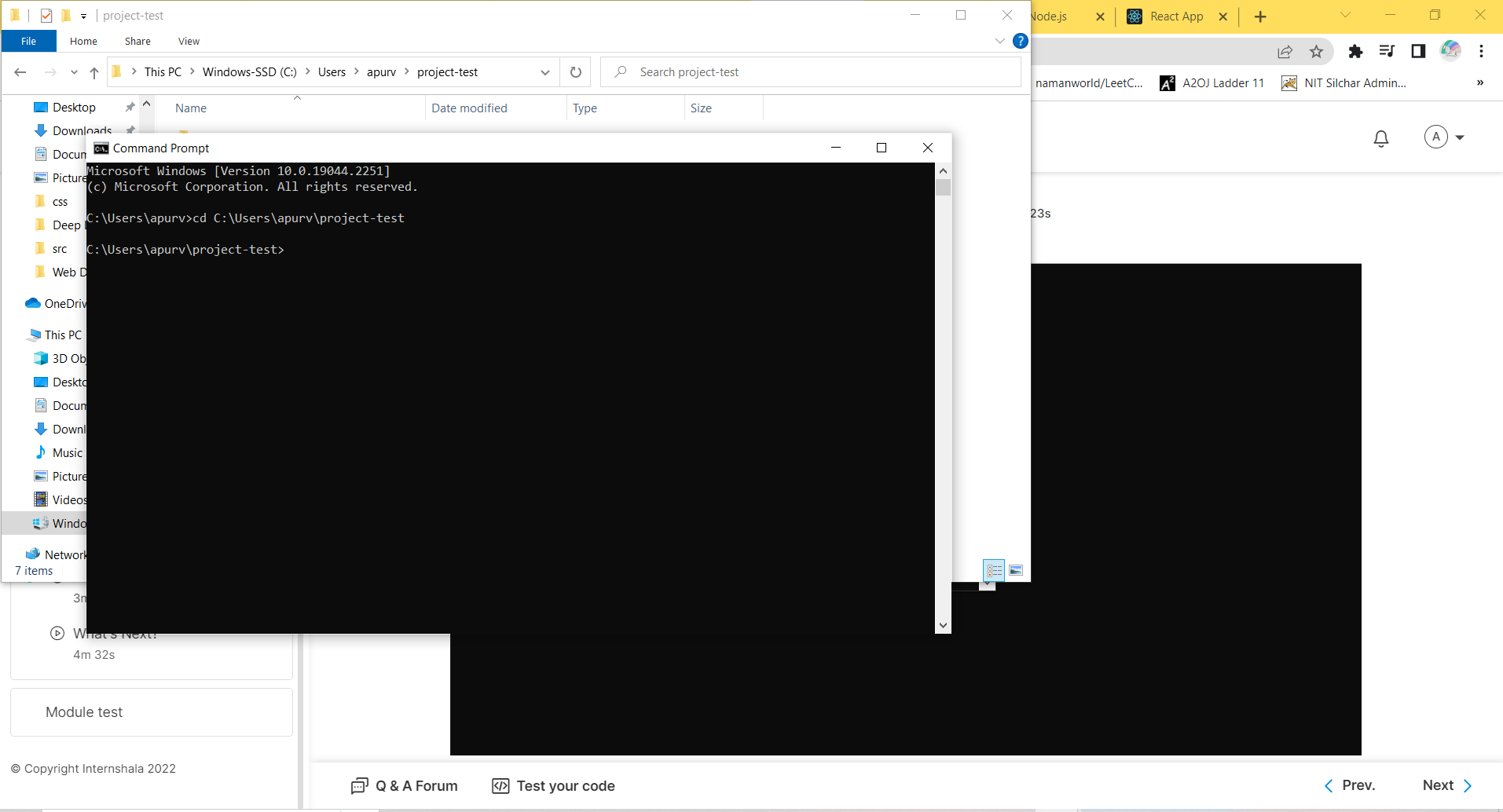
**OUTPUT:**



**Type npm start in command prompt if local host window of react app is not open, it wil show the output.**

**BUILDING THE REACT PROJECT:**

**We will first change our directory to that location in which project-test is present. As shown below:**



**Type cd and copy-paste location of This PC > Windows-SSD(C:) > Users >apurv > project-test as shown in photo by clicking on that line and paste in cmd prompt.**

**Then type**

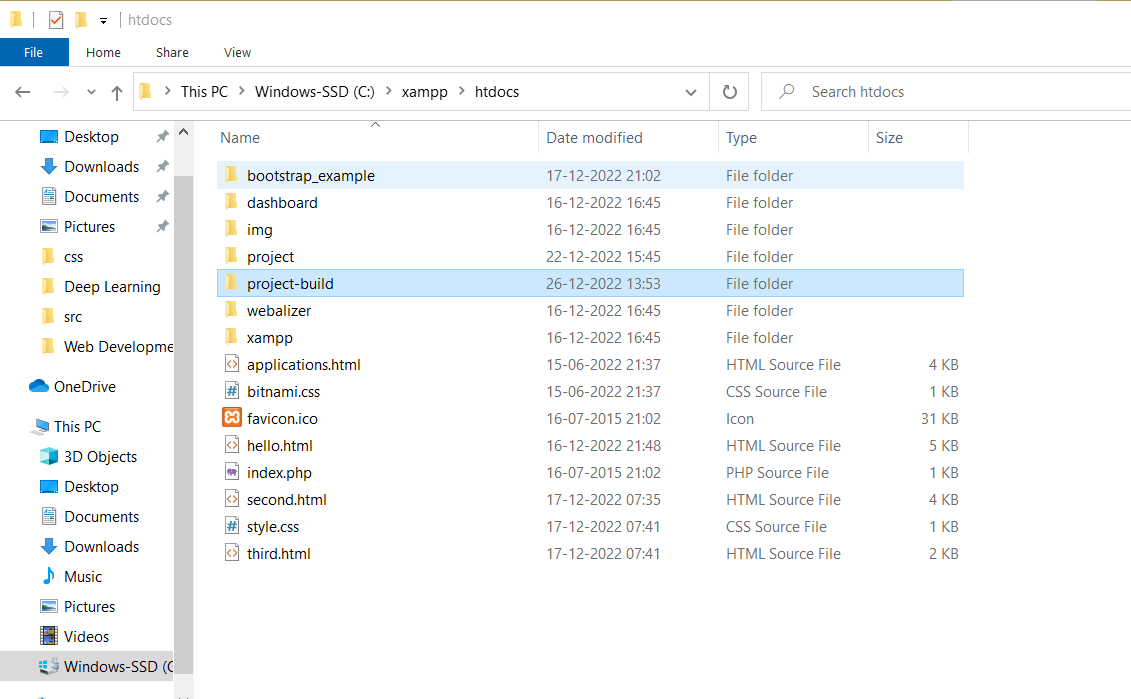


**To create a folder named ‘build’ in project-test.**

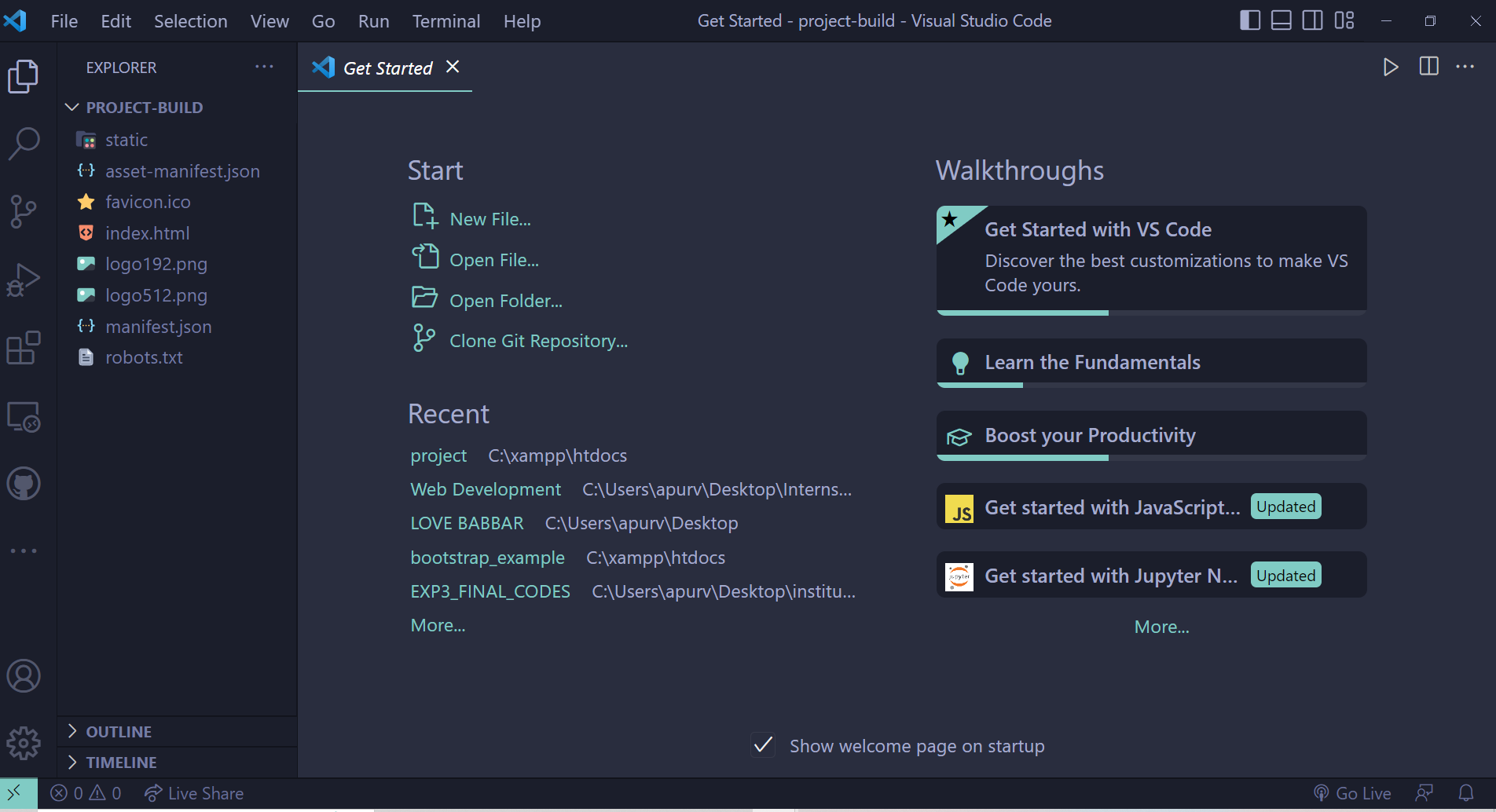
**Copy ‘build’ folder and paste it in C: > xampp> htdocs folder so, that it can be directly accessed by browser.**

**Why are we doing this? As react app creates more organised folders for company works and productions.**

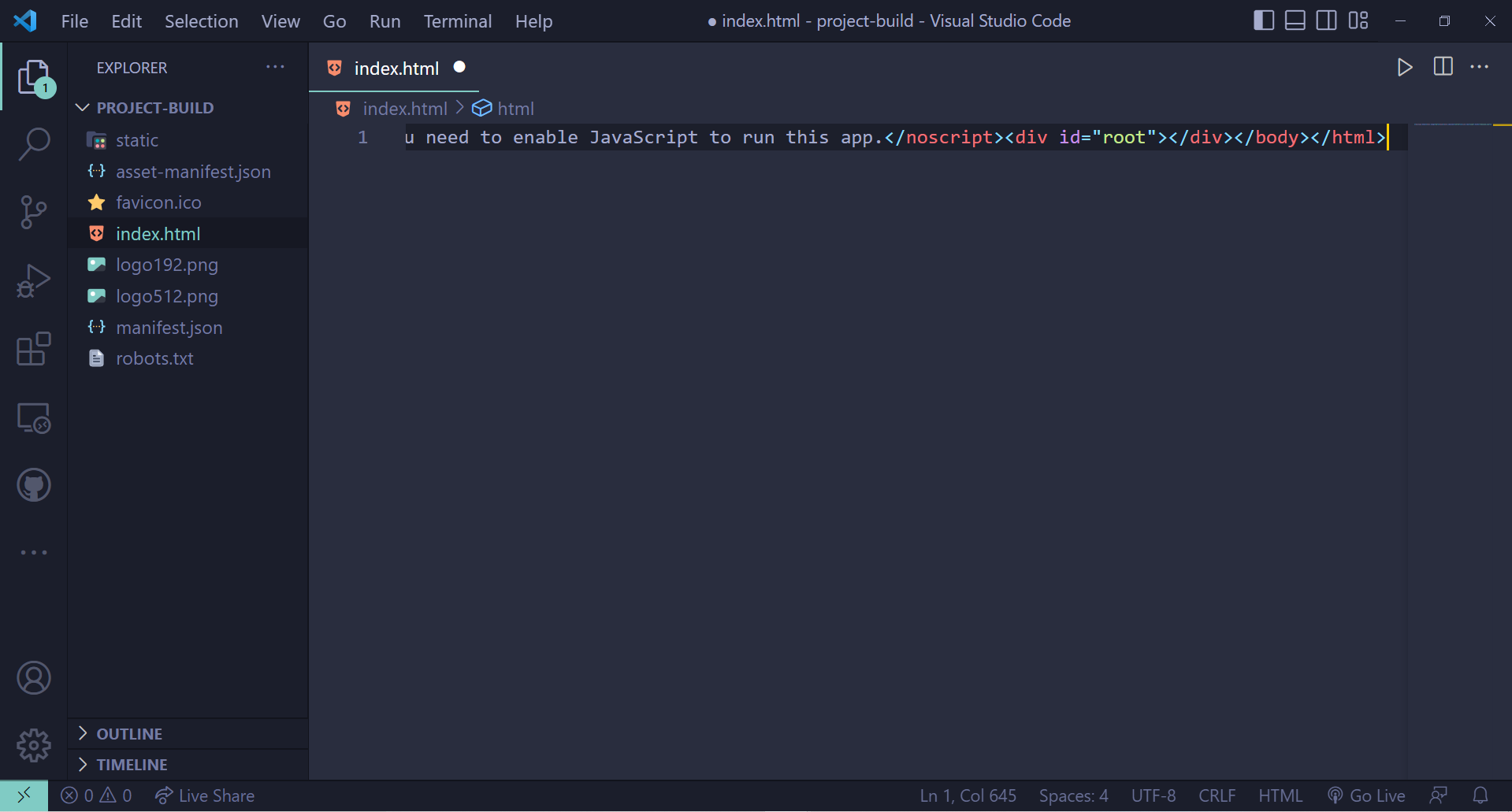
**We can rename the copied folder in htdocs to for eg, project-build.**



**Open folder using VSCode.**

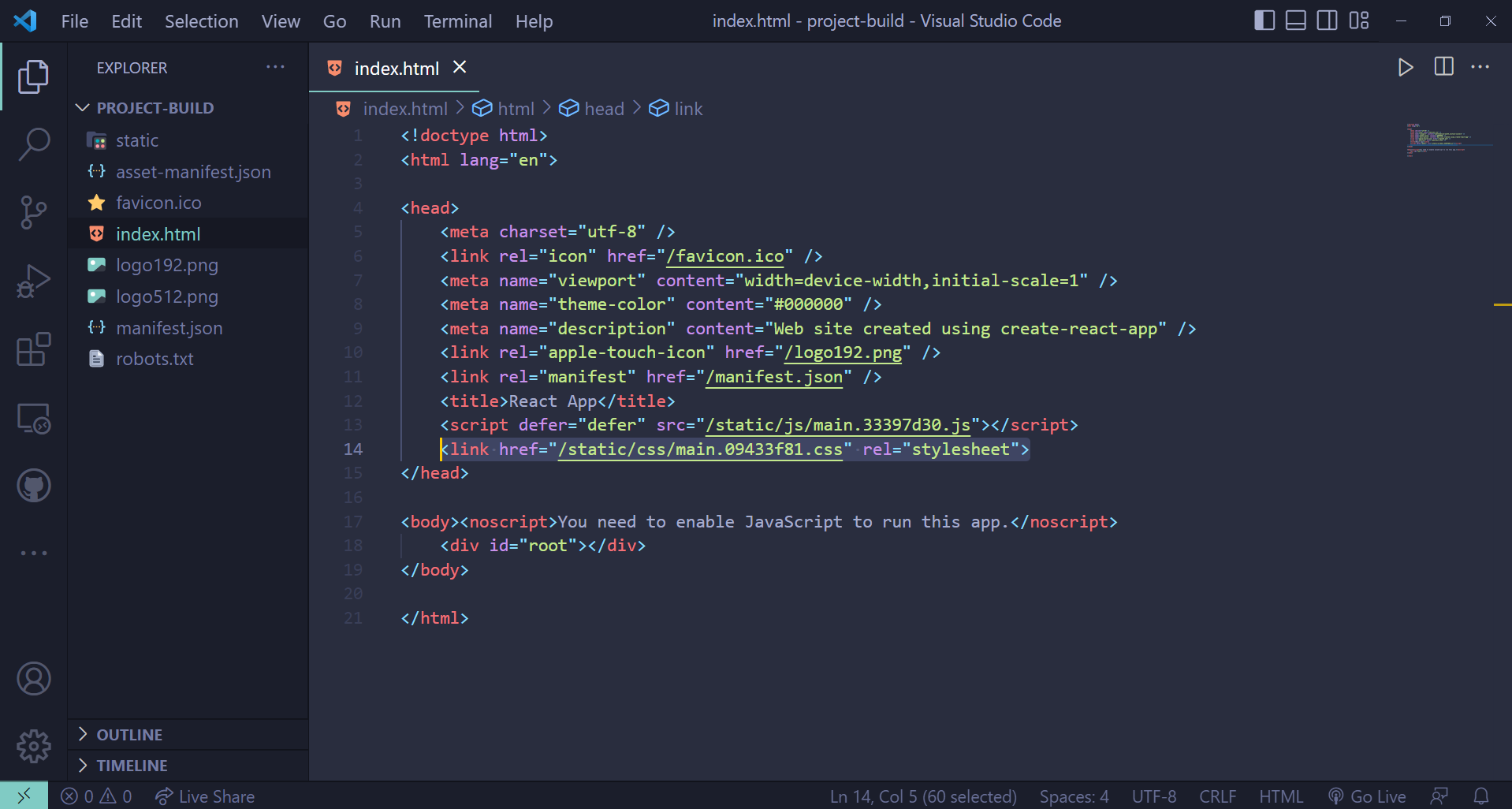


**Open index.html there.**



**To auto format it:**

**Ctrl+A > ctrl+K > ctrl+F**



**Now, remove first ‘/’ from this line so, that it is accessible by xampp local host after starting xampp local host service.**