npx parcel index.html - > instead of running the commend everytime with start point we can add it in package.json in script to run in npm

* In new application we don’t have idea of how to run the application we can go to package.json in script object we can see the command
* If we need to create the div element then we need to find the root element and then create a div element with createElement and render to the root
* The above is core concept of react which is very ugly or very clumcy to write when there is large application to overcome this we use **JSX**
* JSX -> is not part of react ..we can build react app without JSX with createElement
* JSX
  + Jsx try to merge the html and js together
  + **Jsx is not html it is different 🡪 it is not html inside javascript it is like html syntax**

JSX **looks like HTML**, but it's **not exactly HTML**—it's a **syntax extension for JavaScript** used in React.

**Here's the breakdown:**

* **JSX** stands for **JavaScript XML**.
* It allows you to **write HTML-like tags directly inside JavaScript code**.
* Under the hood, JSX is transformed into regular JavaScript using tools like **Babel**.

**✅ JSX Example:**

const element = <h1>Hello, world!</h1>;

This JSX code is **not HTML**, but it’s very similar. It gets compiled into:

const element = React.createElement('h1', null, 'Hello, world!');

**Differences between JSX and HTML:**

1. **className instead of class**

<div className="box"></div> // ✅ JSX

1. **Self-closing tags must be properly closed**

<img src="logo.png" /> // ✅

1. **JavaScript expressions go inside {}**

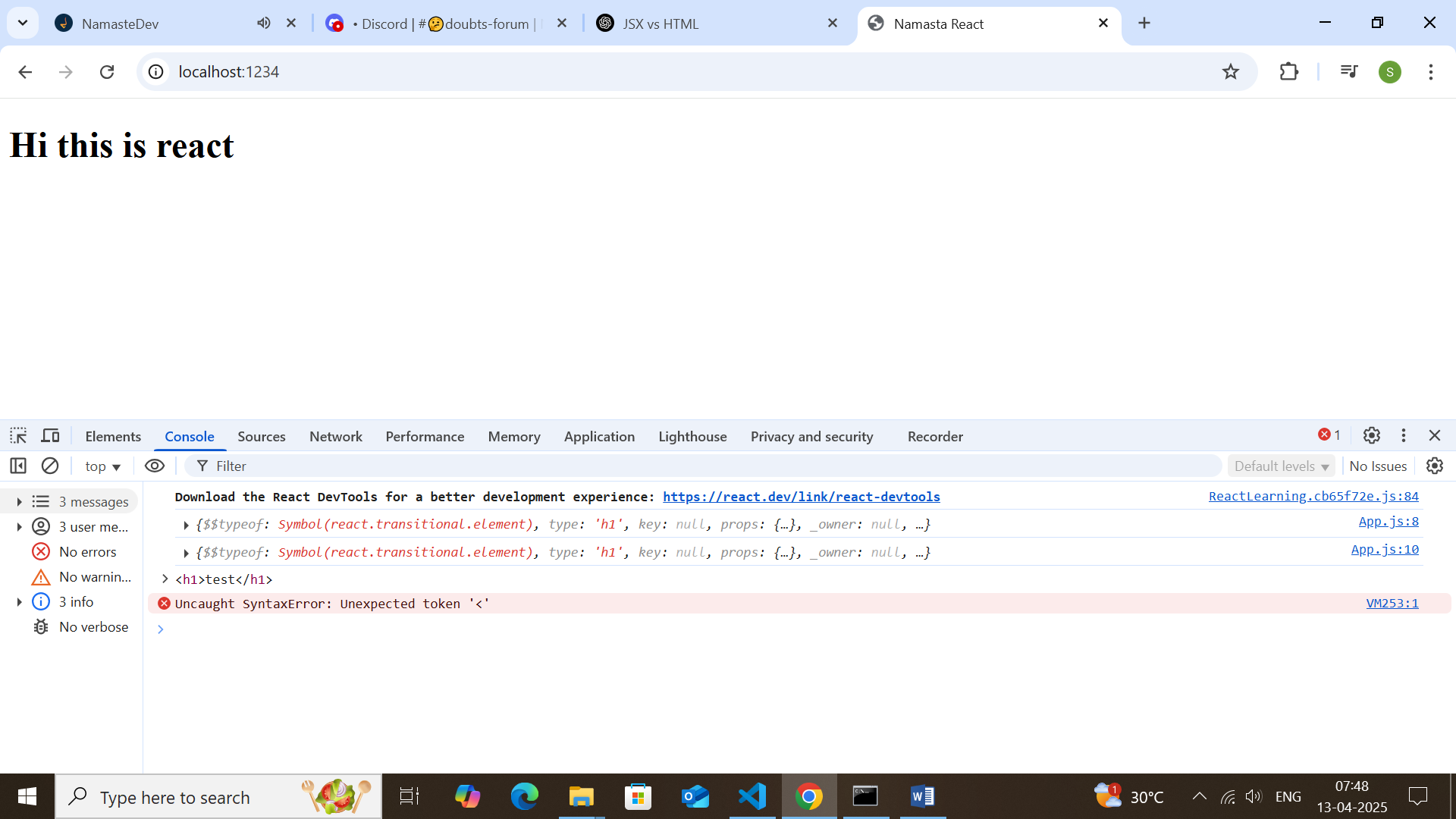
const name = "John";

<p>Hello, {name}</p>

**So, is JSX HTML inside JavaScript?**

👉 **No**, it’s **not HTML**—it’s **HTML-like syntax inside JavaScript**, which gets compiled into regular JavaScript code.

**JSX -> javascript xml**

* It is like html which is used with react to display dom elements and it is converted into object in backend
* Jsx -> this cannot be understood by js engine .. it is not ecmascript i.e language that js engine understand
* 
* But it is converted into createElement object by **parcel** .. the code is transplied into javascript code before it goes to js engine
* **PARCEL – alone is not doing it is done by babel present inside parcel**

**what is babel**

**Babel** is a **JavaScript compiler** that helps you write modern JavaScript (and JSX!) while making sure it works in older browsers.

### 🛠 What Babel does:

1. **Transforms modern JavaScript (ES6/ES7/ESNext)** into older JavaScript (like ES5).
2. **Converts JSX into regular JavaScript** (like React.createElement).
3. **Lets you use experimental features safely** before they’re officially in JavaScript.

### 💡 Why you need Babel in React:

React uses **JSX**, and browsers don’t understand JSX by default. Babel converts your JSX into browser-friendly JavaScript.

#### JSX with Babel:

const element = <h1>Hello</h1>;

⬇️ Babel compiles it into:

const element = React.createElement('h1', null, 'Hello');

### 📦 Babel is used with tools like:

* **Webpack** (or Vite)
* **Create React App** (already includes Babel)
* **Parcel**

JSX ==> react.createElemnet(through babel inside parcel) =>react object => HTML element(render do this)

**Babel -> It is js library which convert the jsx code to javascript code that is compatable with older browser too**

* We nee to wrap the jsx with () if we need to display jsx in multiple lines

**React components**

In react everything is component

There are 2 types of components in react

1.class based component 🡪 old based -uses class concept

2.functional based 🡪 latest code – uses functional

What is react functional component?

It is just normal js function .and It returns the jsx as output

Rendering the component in react

const HeadingComponent = () => {

  return <h2>namesta react app</h2>;

};

Root.render(<HeadingComponent/>

2.render one component inside another

const Header = () => (

  <div>

    <h1>namesta react app1</h1>

    <HeadingComponent />

  </div>

);

When we add component with small case intead of upper case

const component2 = () => {

  return (

    <div>

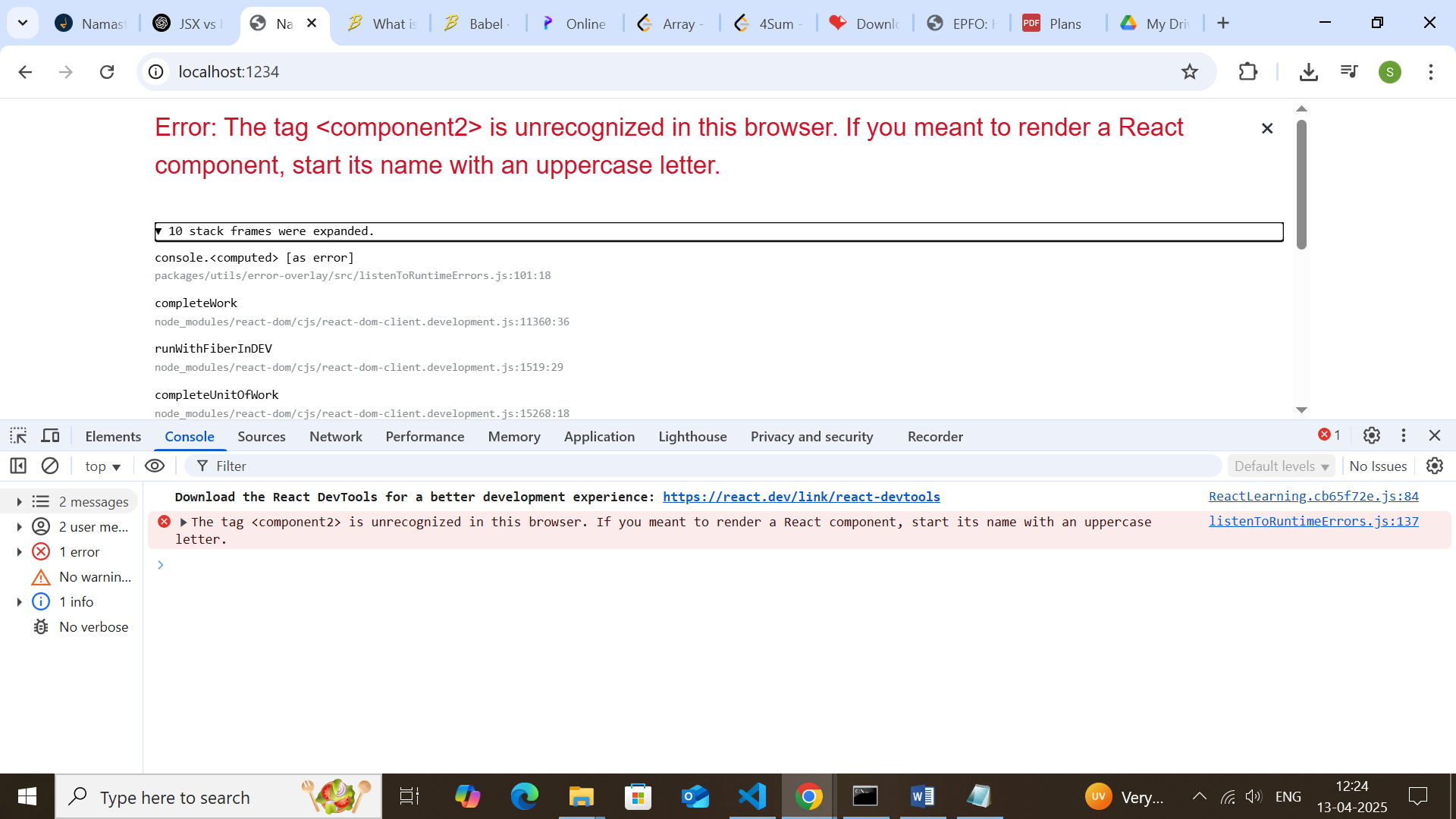
      <h2>component 2</h2>

      <component3 />

    </div>

  );

};



Put element inside component

1.we can run any js code within jsx inside {} of jsx

let number = 1000;

const Component2 = () => {

  return (

    <div>

      <h2>component 2 {number + 200}</h2>

      <Component3 />

    </div>

  );

};

Writing js inside jsx is very important and useful

2.how to put element inside component

const title = <h4>Hi this is title of element</h4>;

const Component2 = () => {

  return (

    <div>

      <h2>component 2 {number + 200}</h2>

      <h3>{console.log("test")}</h3>

      <div>{title}</div>

      <Component3 />

    </div>

  );

};

In case if data comes from the external variable have malicious data then attacker can get the record of user .. this is called cross browser attack..if attacker can run in user browser it will be a attack .. but the jsx takes care of it

**Jsx ->** prevent cross site attack .. it sanitizes all the data from javascript and handle it

We will be able to call the function i.e the component as function inside another component

<div>

      <h2>component 2 {number + 200}</h2>

      <h3>{console.log("test")}</h3>

      <div>{title}</div>

      <Component3 />

      {/\* can we call function inside component - yes \*/}

      {Component3()}

    </div>

There are 3 options through which we can call one component inside another

* As function
* As tag
* As variable if it is element like {}

Code is readable because of jsx which is making the application readable