@NOVADESIGNS

REACT

A popular **JavaScript** library for building **user interfaces** with a focus on **reusable components**, allowing developers to create complex UIs from simple, isolated pieces of code.

NOVADESIGNS

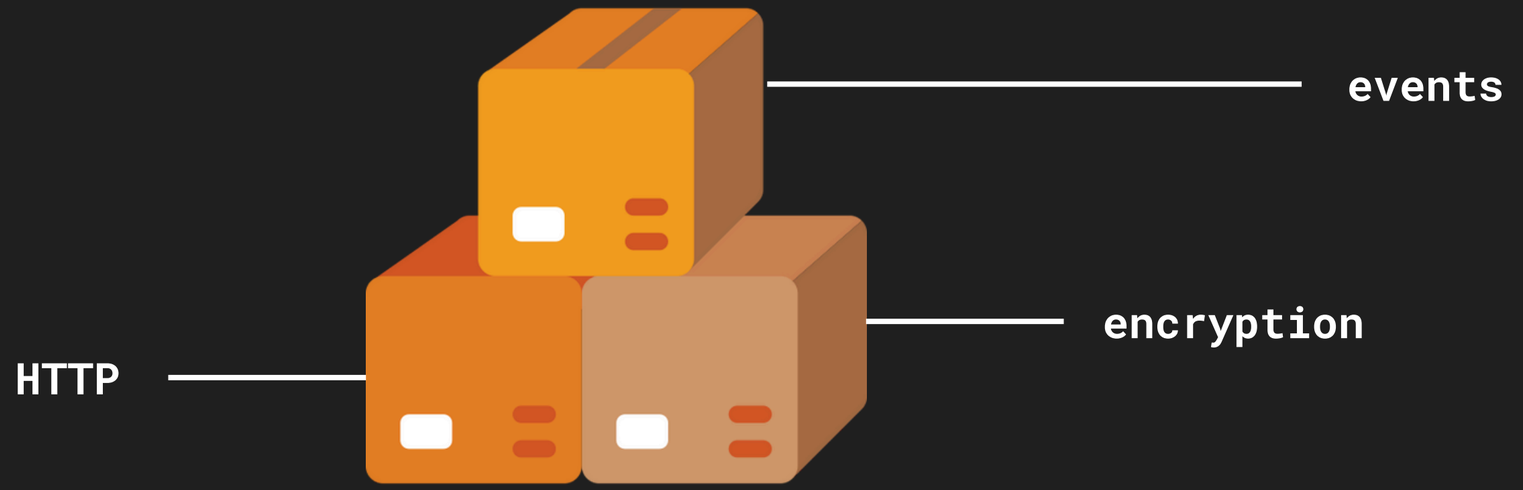
CREATE A REACT APP

VS CODE TERMINAL

```
> npm create-react-app appName
```

The command **npx create-react-app appname** is a tool that sets up a new React project with a default configuration.

NODE MODULES



Node modules are libraries or packages of **reusable code** in JavaScript that can be included in a Node.js project to provide specific functionality or features.

PUBLIC FOLDER

this is where you store static assets like images, fonts, and HTML files that are directly accessible to the client without any server-side processing.

manifest.json

defines **metadata** and settings for a web application, such as its name, icons, and start URL.

robots.txt

instructs **search engines** on which parts of a website they are allowed or disallowed to crawl and index.

favicon.ico

is an icon that appears in the browser tab or bookmark bar, representing a website.

SOURCE FOLDER

this is where you store the **source code of your application**, including JavaScript, CSS, and other files that are processed and bundled for the final build.

index.js

it is connected to the main HTML file to render the App.js

App.js

used for building and organizing the app's UI.

App.css

used for designin the app's UI.

index.css

used for designin the index's UI.

app.test.js

is a file where you write tests to check components

reportWebVitals.js

used to measure and report the performance metrics of your web application

JSX

is a syntax extension for JavaScript that allows you to write HTML-like code directly within JavaScript, making it easier to create and manage user interfaces in React.

```
const name = "John";
const element = <h1>Hello, {name}</h1>;
```

Use curly braces for embedding JavaScript expressions

Hello, John

FUNCTIONAL COMPONENT

is a simple way to create components in React using a JavaScript function, which returns JSX to define what the UI should look like.

Function Definition:
A JavaScript function that defines the component.

Return Statement:
where JSX is returned to render the component's UI.

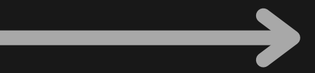
```
function ComponentName() {
  return (
    <div>You can write JSX here.</div>
  )
}

export default ComponentName
```

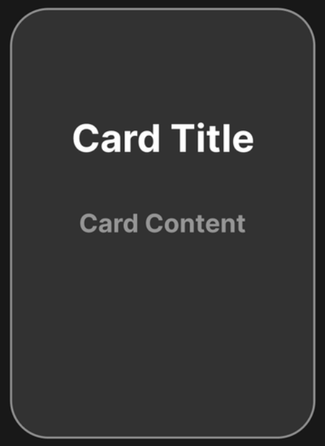
Function Name:
The name of the function, which should start with an uppercase letter and represents the component.

Export Default:
component is usually exported so it can be imported and used in other parts of the application.

PROPS

allowing you to customize and configure
how the component behaves and appears.

(see the next slide) 

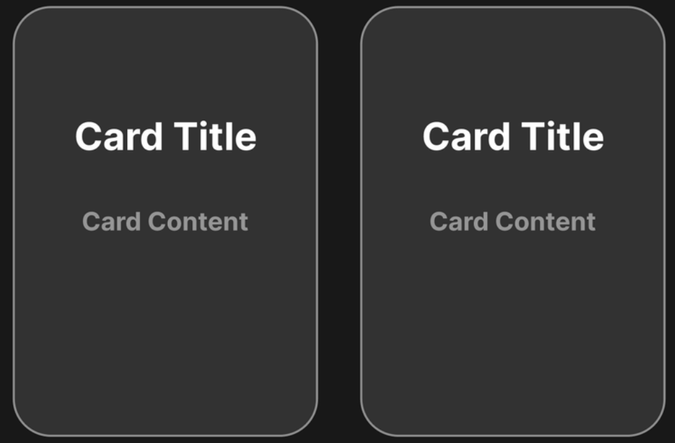



```jsx
function Card() {
  return (
    <div className="card">
      <h2>card title</h2>
      <p>card content</p>
    </div>
  );
}
export default Card;
```

for example, we have here a functional component called Card.

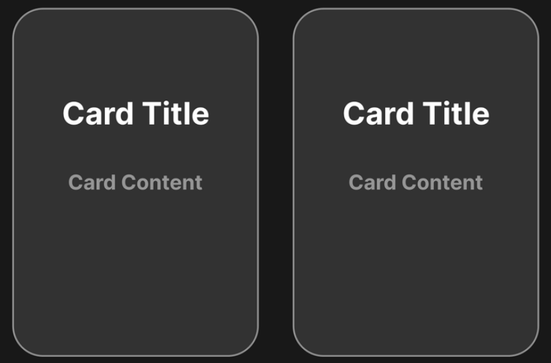
we imported two card component in app.js

```jsx
function App() {
  return (
    <div>
      <Card />
      <Card />
    </div>
  );
}
```

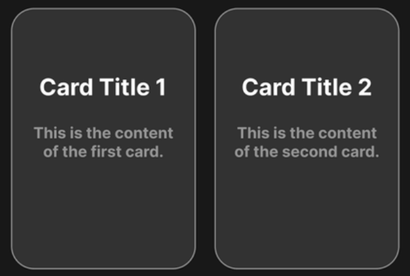


and then write the property we want that card to have.

```jsx
function App() {
  return (
    <div>
      <Card
      title="Card Title 1"
      content="This is card no. 1" />
      <Card
      title="Card Title 2"
      content="This is card no. 2" />
    </div>
  );
}
```




```jsx
function Card(props) {
  return (
    <div className="card">
      <h2>{props.title}</h2>
      <p>{props.content}</p>
    </div>
  );
}
```



and then write props in the parenthesis of the function and get the props.

SPREAD OPERATOR

The spread operator (...) in JavaScript allows you to expand or spread elements from an iterable (like an array) into individual elements, or to spread properties from an object into another object.

Arrays:

```
const numbers = [1, 2, 3];
const moreNumbers = [...numbers, 4, 5];
// [1, 2, 3, 4, 5]
```

Objects:

```
const person = { name: "Alice", age: 25 };
const updatedPerson = { ...person, age: 26 };
// { name: 'Alice', age: 26 }
```

In React, the spread operator is often used to pass props to components:

```
const buttonProps = { type: "button", className: "btn" };
<Button {...buttonProps} />
// This will render <Button type="button" className="btn" />
```

USESTATE

useState is a tool in React that lets you add and manage state in a functional component. It allows you to create a variable that can change over time and a function to update that variable.

```
const [count, setCount] = useState(0);
```

Current State — **Update the State** — **Current Value of State**

0 → `setCount(count + 1);` (0) → **1**

Current State → **Current State**

TERNARY OPERATOR

The ternary operator is a shorthand way to write simple if-else statements in JavaScript.

```
{condition ? "Expression 1" : "Expression 2"}
```

This is an expression that evaluates to true or false.

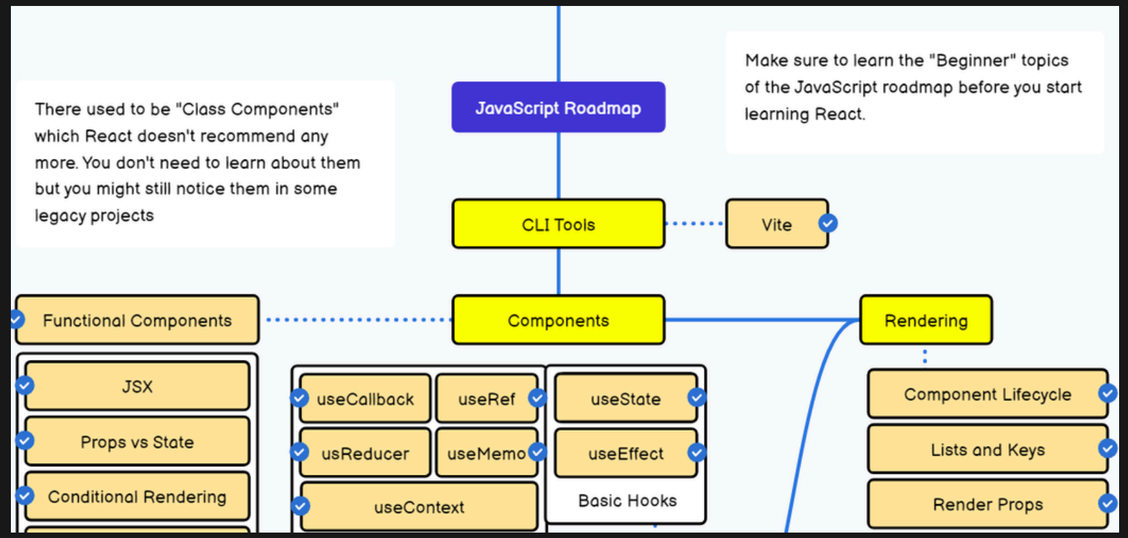
Executed when the condtion is true.

Executed when the condtion is false.

MORE INFO

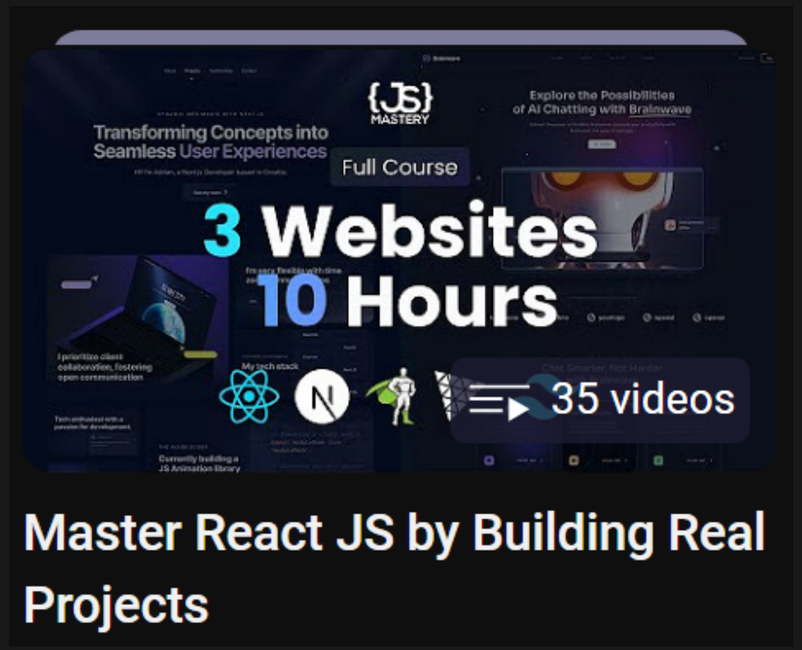
SUBSCRIBE TO NOVA DESIGNS FOR MORE INFO

REACT ROADMAP:



https://roadmap.sh/react

REACT PROJECTS:



JavaScript Mastery (YT)