ReactJS

WHAT IS REACT?

React is a JavaScript framework

Used for front end web development

Created and used by Facebook

Famous for implementing a virtual dom

WHAT IS REACT?

- React is a JavaScript library for building user interfaces.
- React was created by Jordan Walke, a software engineer at Facebook (2012). React was publicly released at the JS Conference in May 2013.
- **Virtual DOM:** One of React's key innovations is the use of a virtual DOM. Instead of directly manipulating the browser's DOM, React creates a lightweight representation of the DOM in memory, making updates faster and more efficient.
- **Component-Based Architecture:** React introduced a component-based architecture, where UIs are broken down into reusable components. This approach promotes code reusability, maintainability, and scalability.

Applications

Who is using React?

Facebook

Prime video

Netflix

Instagram

Common tasks in front-end development

App state	Data definition, organization, and storage
User actions	Event handlers respond to user actions
Templates	Design and render HTML templates
Routing	Resolve URLs
Data fetching	Interact with server(s) through APIs and AJAX

ReactJS - Everything is a Component:

React has no…

- controllers
- directives
- templates
- global event listeners
- models
- no view models

Just Components

Fundamentals of React

JavaScript and HTML in the same file (JSX)

Embrace functional programming

Components everywhere

JavaScript and HTML *in the same file*

Traditional approach

React approach

Installation

Method 1: Using Create React App (CRA)

Step 1: Install Node.js

Download and install Node.js from the official website: Node.js

Step 2: Install Create React App (CRA) globally

Open a command window and run the following command to install Create React App globally:

```
npm install -g create-react-app
```

Step 3: Create a React App

Navigate to the desired directory where you want to create your React application.

Then, execute the following commands:

```
cd Documents
npx create-react-app demo
cd demo
code .
```

This (`code .`) will open the project into VS Code editor Window

Step 4: Run the React App

In the VS Code Terminal, within the project directory, run the following command to start the development server:

```
npm start
```

This will start the React application, and it will be accessible in your web browser at `http://localhost:3000`.

Method 2: Running React with HTML Page

Step 1:

- Include React Libraries in HTML File
- Create an HTML file and include the following CDN links in the <head> section of your HTML file:

```
<!DOCTYPE html>
<html lang="en">
<head> <title>React App</title>
      <script src="https://unpkg.com/react@18/umd/react.development.js"></script>
      <script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
      <script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
</head>
<body>
<div id="root"></div> <!-- Your React components will be rendered here -->
      <script type="text/babel">
            const App = () =>
                  {
                     return ( <div><h1>Hello, React!</h1></div>);
                  };
            ReactDOM.render(<App />, document.getElementById('root'));
</script>
</body>
</html>
```

Step 2: Write React Components

Write your React components inside `<script type="text/babel">` tag in the HTML file as shown in the example above.

Step 3: Open HTML File in Browser

Open the HTML file in a `web browser` or start the `live server` in VS Code (https://marketplace.visualstudio.com/items?itemName=ritwickdey.LiveServer), and you will see your React application running.

ReactJS Components

- In React.js, a component is a reusable, self-contained building block for creating user interfaces.
- Components are the core abstraction in React, encapsulating both the UI and the behavior of a part of the application.
- They can represent anything from a simple button or input field to more complex elements like forms, tables, or entire sections of a web page.
- **React Functional Components**, are a fundamental aspect of modern React applications. These components are essentially JavaScript functions that accept data in the form of props(properties) and return React elements.

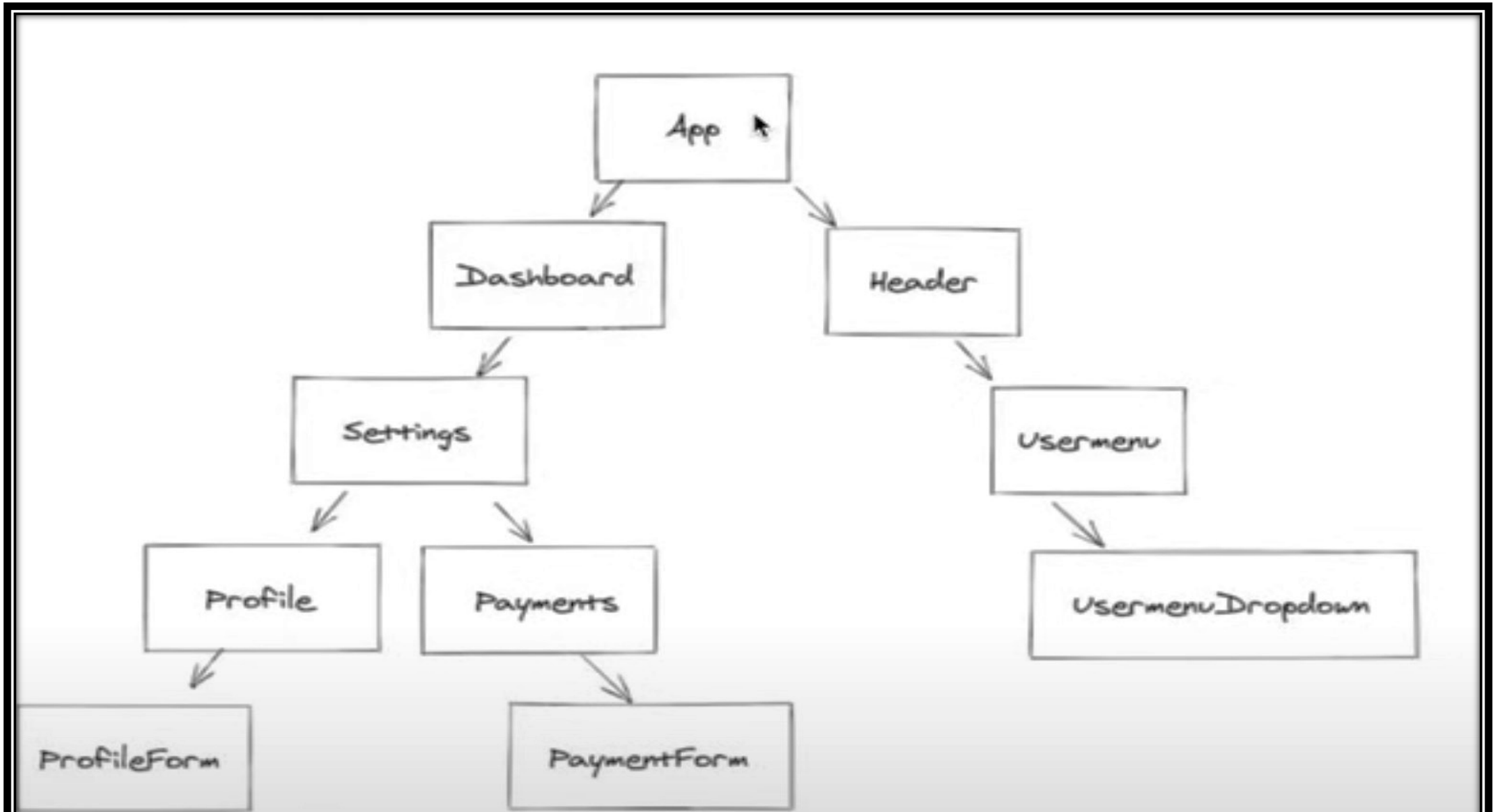


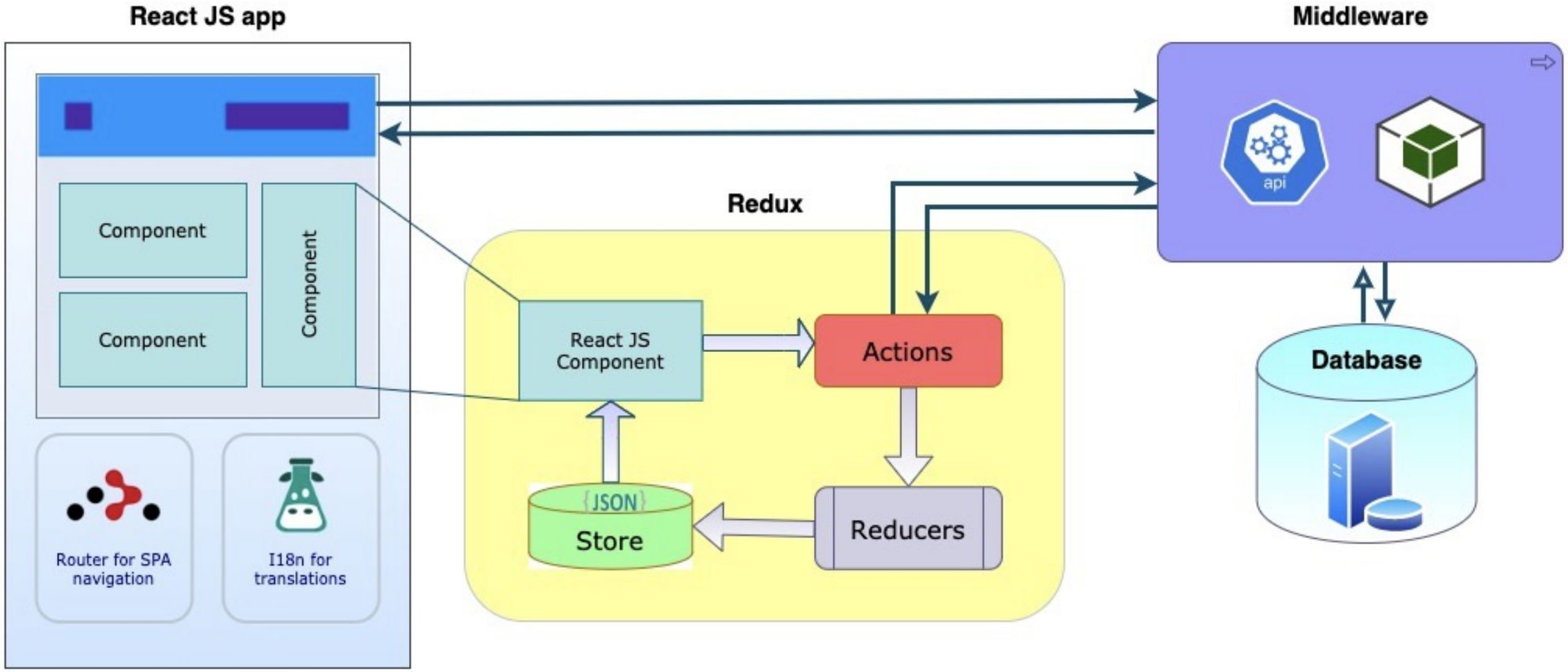
Diagram of a basic React component tree.

React Component Lifecycle

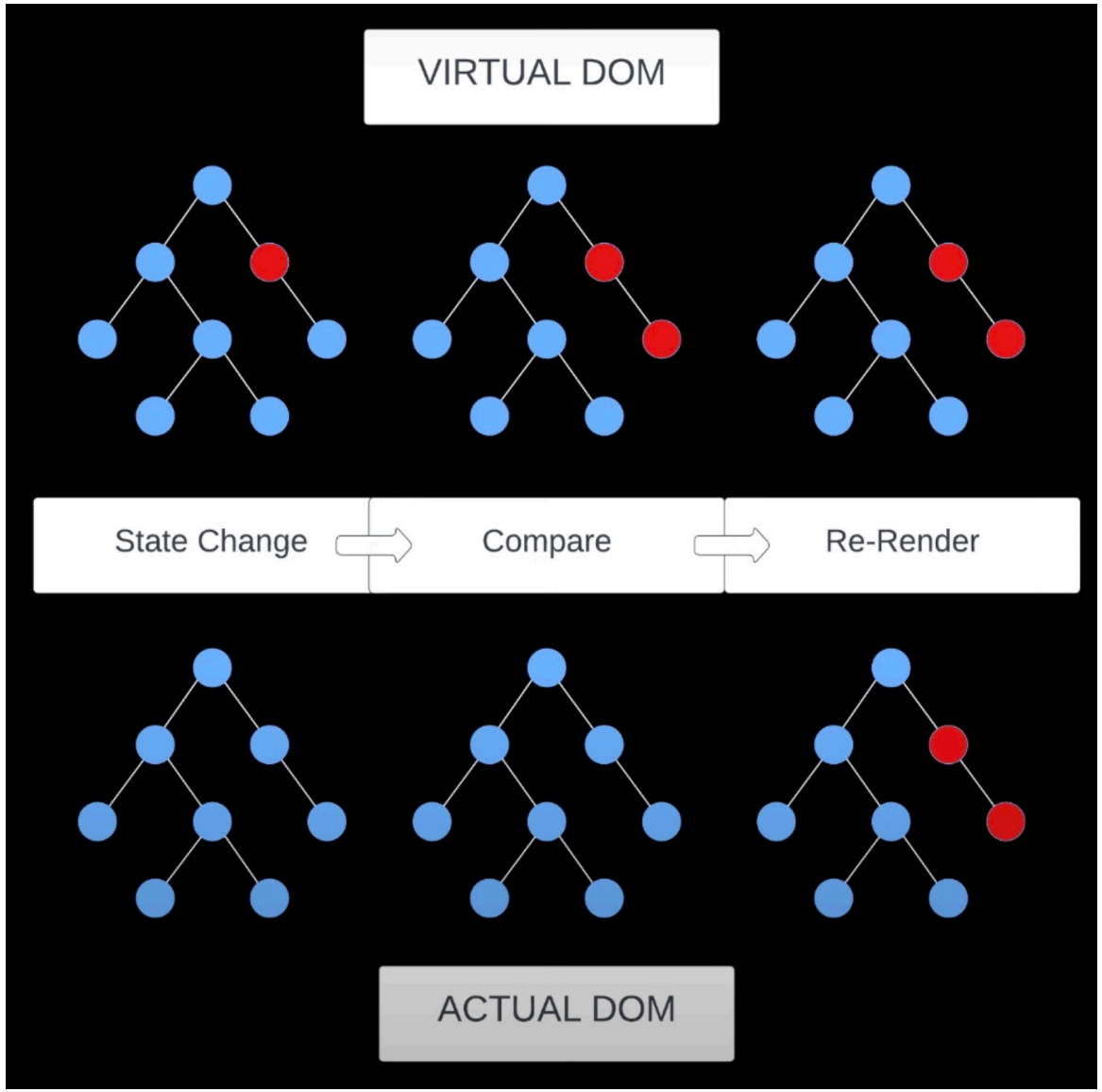
- The React component lifecycle consists of three main phases:
  - Mounting
    - During the Mounting phase, a component is created and inserted into the DOM.
  - Updating
    - The Updating phase occurs when a component is re-rendered due to changes in props or state.
  - Unmounting
    - The Unmounting phase happens when a component is removed from the DOM, and it involves the componentWillUnmount method.
- React components can have lifecycle methods that allow developers to run code at specific times during these phases, ensuring efficient management of component behavior and state throughout their lifecycle.

ReactNative vs ReactJS

FEATURE	REACT NATIVE	REACTJS
Platform	Mobile applications	Web applications
UI Components	Native UI components	Web-based UI components
Development	Separate for iOS and Android	Focused on browser platforms
Performance	Slower due to bridging	Faster as it runs in the browser
Device APIs	Access to device APIs	Limited access to device APIs
Navigation	Mobile-specific navigation	Web-based routing
Deployment	Requires separate binaries	Simpler for web applications


React JS app
Component
Component
Component
Router for SPA navigation
I18n for translations
Redux
React JS Component
Actions
{JSON}
Store
Reducers
Middleware
api
Database

Virtual vs Actual DOM



Files in the react.js

- **Index.html** is the first file gets run
- **App** is the default parent component in react
- **Index.js** says what are the components to be rendered
- **App.js** says what content to be rendered in app component

Files in the react.js

Files	Purpose
node_modules	All the node module dependencies are created in this folder
public	This folder contains the public static assets of the application
public/index.html	This is the first page that gets loaded when we run the application
src	All application related files/folders are created in this folder
src/index.js	This is the entry point of the application
package.json	Contains the dependencies of the React application

JSX(JavaScript XML)

- JSX stands for JavaScript XML.
- JSX is syntax extension to javascript and is used to define the react components
- It allows us to define react elements using syntax that looks similar to HTML
- JSX can easily convert HTML tags to react elements.
- It is faster than regular JavaScript.
- JSX follows XML rule.

useState

- useState is a built-in React hook that allows functional components to manage state.
- It enables functional components to have stateful logic.
- **Syntax:** `const [state, setState] = useState(initialState);`
  - **state:** Represents the current state value.
  - **setState:** Function to update the state value. When called, it triggers a re-render of the component with the updated state.
- **Initialization**:
  - You provide **useState** with an initial state value as its argument. This initial state is only used during the first render.
- **Usage**:
  - You can access the current state value through the state variable.
  - To update the state, you call the **setState** function with the new value you want to assign to the state.