2026 Edition

```
import React from 'react';

function App() {
  return (
    <div className="app">
      <h1>Hello, React! 👋</h1>
    </div>
  );
}

export default App;
```

LearnStack

```
// Build • Learn • Ship
const future =
  <React.StrictMode>
    <App />
  </React.StrictMode>;
```

React.js

Complete Handbook 2026

From Zero to Full-Stack React Developer

JSX · Components · Props & State · Hooks · React Router · Context API · APIs · Testing · Next.js

LearnStack Tech Handbooks

Learn. Build. Grow.

₹99 | Gumroad Edition

15 Chapters + 3 Bonus + 6-Page Cheat Sheet

linktr.ee/LearnStack

Welcome to Your React Journey

How to Use This Book

This handbook is designed as a guided journey. Read the track that matches your level, write the examples yourself, and complete the activities before moving on.

BEGINNER Track

Chapters 1-5. Start here if React is new and you want strong fundamentals.

INTERMEDIATE Track

Chapters 6-10. Learn state, effects, events, lists, and forms.

ADVANCED Track

Chapters 11-15. Build routed, shared-state, optimized, API-driven apps.

Prerequisites Checklist

- ✓ Basic HTML tags and attributes
- ✓ Basic CSS selectors and box model
- ✓ JavaScript variables, functions, arrays
- ✓ ES6 features: arrow functions and destructuring
- ✓ You do not need previous React experience

Icons You Will See

💡 **PRO TIP**
Advanced insight that separates good from great developers.

⚠️ **COMMON MISTAKE**
Common beginner mistake and how to avoid it.

✅ **BEST PRACTICE**
Industry-standard way to write React in real projects.

Professional Practice

Build it small
Start with the smallest working component before adding abstractions.

Name it clearly
Good names reduce comments and make debugging faster.

Review it later
Return after the quiz and improve one piece of code with the chapter idea.

© 2026 LearnStack | React.js Handbook

REACT.JS COMPLETE HANDBOOK 2026

Contents

A clean learning path from your first component to production-ready React patterns.

15 Chapters
3 Bonus + Cheat Sheet

100+	75+	15	6
Examples	Quiz Questions	Hands-on Activities	Cheat Sheet Pages

Beginner — Intermediate — Advanced — Professional

BEGINNER TRACK — Chapters 01-05

INTERMEDIATE TRACK — Chapters 06-10

How to use the index: Finish every example page, then complete the activity before the quiz. The book is designed to remove empty reading and turn every concept into practice.

© 2026 LearnStack | React.js Handbook

ADVANCED TRACK + BONUS MODULES

Contents Continued

Move from app structure to professional patterns, performance, APIs, and interview-ready reference pages.

Advanced
React Workflow

ADVANCED TRACK

Chapters 11-15

BONUS + REFERENCE

Production confidence

01. Learn
Read the mental model first. Do not start with memorising syntax.

02. Build
Type every code example and change one thing before moving forward.

03. Grow
Use the quiz and activity to test whether you can apply the concept.

Designed for paid LearnStack buyers: clean reading, practical code, quizzes, activities, and no wasted blank pages.

Advanced build path: Routes · Shared State · Custom Hooks · Performance · APIs · Portfolio App

© 2026 LearnStack | React.js Handbook

< Chapter 01 />

What is React? Why React?

Think in components before you write components.

After this chapter you will be able to:

- Understand what problem React solves
- Explain library vs framework in plain language
- Describe Virtual DOM and diffing
- Recognise React as a component-based UI tool

Concept Map
Start with the mental model, then connect it to code.

Build Target
Every example leads into a hands-on activity for this chapter.

Quiz Focus
Expect practical questions on mistakes, outputs, and best practices.

© 2026 LearnStack | React.js Handbook

Core Concept

{/* ----- What is React? Why React? ----- */}

Mental Model

React is like a restaurant kitchen with recipe cards. When an order changes, the chef does not rebuild the entire restaurant - only the dish that changed is prepared again.

- React helps you build user interfaces by splitting screens into small reusable pieces called components.
- Instead of manually changing the DOM again and again, you describe what the UI should look like for each state.
- React compares the new UI description with the previous one and updates only what changed.
- This is why React feels simple for small apps and still works for large production apps.

🧠 **UNDERSTAND THIS DEEPLY**

React is like a restaurant kitchen with recipe cards. When an order changes, the chef does not rebuild the entire restaurant - only the dish that changed is prepared again.

✅ **BEST PRACTICE**

Write small components, keep data flow visible, and prefer clarity before cleverness.

Choice	Learning curve	Best use	Modern status
React	Medium	Component-based apps	✅ Industry standard
Vue	Easy	Progressive apps	✅ Strong ecosystem
Angular	Steep	Enterprise full framework	✅ Enterprise
Vanilla JS	Easy first, hard later	Small interactions	⚠️ Hard at scale

Chapter 1 - What is React? Why React?

Concept Expansion

Real project meaning

Use this concept to make components easier to change without touching unrelated files.

Interview angle

Explain the idea with a tiny UI example first, then mention the React term after the example.

Debug question

Ask: which value changed, who owns that value, and which component should re-render?

Best next step

Build the smallest working version first; then extract reusable parts only when repetition appears.

© 2026 LearnStack | React.js Handbook

Visual Map

Use this diagram as the shape of the idea before reading the code.

```
State changes
     |
     ▼
Virtual DOM tree
     | diff old vs new
     ▼
Minimal Real DOM update
     |
     ▼
Browser paints changed UI
```

What to Notice

React helps you build user interfaces by splitting screens into small reusable pieces called components.

Developer Habit

Pause before coding and name the moving parts: component, data, event, state, and output.

> 💡 **PRO TIP**
> A clean mental model saves more debugging time than memorising syntax.

Chapter 1 - What is React? Why React?

Trace It Before You Code

When stuck
Draw the flow with arrows before changing code.

When confident
Refactor names until the flow reads like plain English.

© 2026 LearnStack | React.js Handbook

Example 1: The DOM pain before React

Type this example yourself, then change one value and predict the result before saving.

counter-vanilla.js React

```
let count = 0;
const value = document.getElementById('value');
const button = document.getElementById('button');

button.addEventListener('click', function () {
  count = count + 1;
  value.textContent = count;
});
```

OUTPUT

🖥️ **What you will see in the browser:**

A counter works, but every UI change is wired manually through the DOM.

⚠️ **COMMON MISTAKE**

Do not copy code blindly. Read each line and ask what state, props, event, or effect it depends on.

Chapter 1 - What is React? Why React?

Code Deep-Dive Notes

Line-by-line focus

Identify what is imported, what data changes, what JSX is returned, and which user action triggers the update.

Try these edits

Rename one variable, change one piece of displayed data, and add a small conditional message.

Production habit

Keep the component small, predictable, and easy to test before adding styling or new libraries.

Read — Type — Change — Explain

© 2026 LearnStack | React.js Handbook

Example 2: The same counter as a React component

Type this example yourself, then change one value and predict the result before saving.

Counter.jsx React

```jsx
import { useState } from 'react';

function Counter() {
 const [count, setCount] = useState(0);

 return (
 <button onClick={() => setCount(count + 1)}>
      Count: {count}
 </button>
  );
}

export default Counter;
```

OUTPUT

What you will see in the browser:

A button displays Count: 0, then updates itself when clicked.

PRO TIP

Do not copy code blindly. Read each line and ask what state, props, event, or effect it depends on.

Chapter 1 - What is React? Why React?

Code Deep-Dive Notes

Line-by-line focus

Identify what is imported, what data changes, what JSX is returned, and which user action triggers the update.

Try these edits

Rename one variable, change one piece of displayed data, and add a small conditional message.

Production habit

Keep the component small, predictable, and easy to test before adding styling or new libraries.

Read — Type — Change — Explain

© 2026 LearnStack | React.js Handbook

Example 3: Everything can become a component

Type this example yourself, then change one value and predict the result before saving.

App.jsx React

```jsx
function Header() {
 return <h1>LearnStack Store</h1>;
}

function ProductCard() {
 return <article>React Handbook - ₹99</article>;
}

function App() {
 return <><Header /><ProductCard /></>;
}
```

OUTPUT

🖥️ **What you will see in the browser:**

A page made from Header and ProductCard building blocks.

✅ **BEST PRACTICE**

Do not copy code blindly. Read each line and ask what state, props, event, or effect it depends on.

Chapter 1 - What is React? Why React?

Code Deep-Dive Notes

Line-by-line focus

Identify what is imported, what data changes, what JSX is returned, and which user action triggers the update.

Try these edits

Rename one variable, change one piece of displayed data, and add a small conditional message.

Production habit

Keep the component small, predictable, and easy to test before adding styling or new libraries.

Read — Type — Change — Explain

© 2026 LearnStack | React.js Handbook

Hands-on Activity

⚡ TRY IT YOURSELF

Activity 1: Component Hunting

Time: 15 minutes | Level: Beginner

What you will build

Open any website and mark ten UI parts that could be components. Then sketch a component tree from page to smallest reusable part.

Steps

1. Create the component structure first.
2. Add state or props only after the layout is visible.
3. Type the starter code, then improve it with your own data.
4. Test one happy path and one empty/error path.

Challenge

Add one extra feature that uses a different concept from the same chapter.

Chapter Summary

- React helps you build user interfaces by splitting screens into small reusable pieces called components.
- Instead of manually changing the DOM again and again, you describe what the UI should look like for each state.
- React compares the new UI description with the previous one and updates only what changed.

Chapter 1 - What is React? Why React?

Activity Finish Line

Works
The app runs and the main action succeeds.

Handles edge cases
Empty, loading, invalid, or missing data has a visible UI.

Explained
You can explain the component tree and state flow without reading notes.

© 2026 LearnStack | React.js Handbook

Quick Quiz

Chapter 1 - 5 Questions

Q1 **MCQ - React is primarily used for what?**

A) A component stylesheet | B) Building user interfaces | C) A browser refresh rule | D) A database table

ANSWER
Answer: Building user interfaces
Because React renders UI from components and state.

Q2 **True/False - React is a full MVC framework.**

A) True | B) False | C) It depends | D) Not related

ANSWER
Answer: False
React is a UI library. Routing, data fetching and state libraries are added separately.

Q3 **MCQ - React was originally created at which company?**

A) A component stylesheet | B) Facebook / Meta | C) A browser refresh rule | D) A database table

ANSWER
Answer: Facebook / Meta
React was developed at Facebook and later open-sourced.

Q4 **Fill the Blank - React thinks in ___.**

A) True | B) False | C) It depends | D) Not related

ANSWER
Answer: components
Components are React's smallest reusable mental unit.

Q5 **MCQ - What does the Virtual DOM help React do?**

A) A component stylesheet | B) Update only changed UI parts | C) A browser refresh rule | D) A database table

ANSWER
Answer: Update only changed UI parts
React compares trees and patches the real DOM efficiently.

✓ 5 questions in this chapter | Total so far: 5

Chapter 1 - What is React? Why React?

After-Quiz Review

Score honestly
Mark every unsure answer. React skill grows when weak spots are named clearly.

Rebuild one example
Return to the code page for your weakest concept and rebuild it without copying.

Write one rule
Convert the chapter into one rule you can remember during an interview or project.

© 2026 LearnStack | React.js Handbook

LearnStack Free Preview

This was a free preview. Get the full book on LearnStack.

Visit: https://www.learnstack.co.in

Digital PDF delivery is handled through Gumroad email after purchase.