Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

# React Essentials

## Class Duration

21 hours of live training delivered over 3-5 days to accommodate your scheduling needs

## Student Prerequisites

- Experience with HTML, CSS, and JavaScript
- Experience with client-side web application programming and TypeScript not required, but recommended

## Target Audience

Software engineers, tech leads, and front-end developers building revenue-impacting web applications who want a rigorous, pragmatic approach to React. Ideal for teams standardizing on React, TypeScript, and REST/BFF patterns, and for organizations prioritizing maintainability, testability, and delivery speed. This course assumes professional programming experience and benefits engineering managers and L&D partners seeking a live, expert-guided workshop to accelerate team capability and reduce time-to-value.

## Description

This live, hands-on course equips experienced developers to ship production-grade React applications that drive business outcomes. Participants start with modern project setup, browser support, and asset strategy, then move through React's rendering model and JSX to create reusable components with robust prop contracts with TypeScript. You will master state, effects, callbacks, and custom hooks; build dependable forms; and integrate RESTful APIs using Fetch and a Backend for Frontend pattern, with a brief look at GraphQL/Apollo. The course covers client routing with React Router, including parameters, 404 handling, and nesting. Throughout, we emphasize component architecture, communication patterns, and testability, finishing with unit testing using Jest and React Testing Library. Optional segments demonstrate how AI tools can accelerate architecture exploration and test generation.

Copyright 2025 Cloud Contraptions LLC – All Rights Reserved

## Learning Outcomes

- Stand up a clean, scalable React project with a sensible folder structure, assets pipeline, and dependency strategy.
- Explain React's rendering model, JSX, the virtual DOM/Fiber, and apply keys and operators to optimize list rendering.
- Design reusable function components with clear prop contracts, defaults, naming patterns, and static typing with TypeScript.
- Wire component events and lift handlers via props to enable predictable, testable UI behavior.
- Manage state, effects, callbacks, and custom hooks—including generics—to encapsulate logic for reuse.
- Build reliable forms using controlled/uncontrolled inputs, validation, and submission flows; optionally leverage form libraries.
- Integrate data using Fetch, REST, and a BFF pattern; handle initial loads and mutations, with brief exposure to GraphQL/Apollo.
- Implement routing with React Router (404s, params, nesting) and write unit tests with Jest/RTL for components and custom hooks.

## Training Materials

All students receive comprehensive courseware covering all topics in the course. Courseware is distributed via GitHub in the form of documentation and extensive code samples. Students practice the topics covered through challenging hands-on lab exercises.

## Software Requirements

Students will need a free, personal GitHub account to access the courseware. Students will need permission to install Node.js and Visual Studio Code on their computers. Also, students will need permission to install NPM Packages and Visual Studio Code Extensions. If students are unable to configure a local environment, a cloud-based environment can be provided.

## Training Topics

### React Project Setup

- Create a new project
- Folder Structure
- Browser Support
- Styles and Assets

- Dependencies
- JavaScript vs TypeScript
- Setup AI Agent Tools (optional)

## React Overview

- React Overview
- Why React?
- Single-page web application architecture
- Server-side rendering (brief overview only)

## Component Rendering and JSX

- What problem does JSX solve?
- Embedding Expressions
- Specifying Attributes
- Using Fragments
- Virtual DOM and Fiber Nodes
- Ternary Operator (?)
- Logical (&& and ||) Operators
- Rendering a list of data
- Optimizing rendering with keys

## Components

- Creating an Element
- Create a Function Component
- Rendering a Component
- Composing & Reuse

## Component Props

- Immutable Props
- String Literals vs. Expressions
- Statically-Typed Components
- Default Prop Values
- Naming Patterns for Props

## Component Events

- What are Events?
- Common Events in React: Click and Change
- Event Handlers and Functional Component

- Passing Event Handlers via Props

## Component Hooks

- What is Component State?
- State Hook
- Effect Hook
- Callback Hook
- Custom Hooks
- TypeScript Generic Types and Reusable Hooks

## Component Architecture

- Reusable Components
- Component Communication
- Design Patterns
- Container and Presentation Components
- Defining Prop Drilling
- Explore/Define Architecture with AI Tools (optional)

## Capture Data with Forms

- Controlled and Uncontrolled Components
- Enable Change Logic across Multiple Form Controls
- Wiring up Input, Textarea, and Select
- Handling different types of Input
- Form Validation and Submission
- React Form Hooks Library (optional)

## AJAX and RESTful APIs

- Fetch API
- Get Data from a REST API for use with a React
- Loading Data on the Initial Render with the Effect Hook
- Add, Replace, and Remove RESTful API Data Calls
- Backend for Frontend (BFF) pattern
- GraphQL + Apollo (brief overview only)

## Routing with React Router

- What problem does React Router solve?
- Installation
- Basics

- Handling Not Found (404)
- Parameters (Url & Query)
- Nesting
- React Router & Remix (brief overview only)

### Unit Testing Overview

- Jest and React Testing Library
- What are React components tested for?
- Tests, Test Suites, Assertions, and Mocking
- Test DOM rendering
- Test Event Handlers with Spies
- Test Custom Hooks
- Mocking Components
- Mocking Hooks
- Using AI Agent Tool to Create Tests (optional)