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 not required, but recommended

Target Audience

Designed for professional front-end and full-stack developers, tech leads, and QA engineers who build and maintain React single-page applications in enterprise environments. It suits teams modernizing legacy UIs, product groups scaling features across multiple domains, and organizations that need predictable delivery, maintainability, and testable code. Leaders in engineering management and L&D will find the curriculum aligned to measurable outcomes: faster feature throughput, lower defect rates, and a clearer React architecture that supports long-term growth. Delivered live online or in person.

Description

This live, hands-on course equips engineering teams to deliver production-ready React applications that drive business results. Participants learn modern component architecture, JSX best practices, hooks for stateful logic, and resilient event handling. You will capture complex user input with validated forms, integrate RESTful backends using fetch and the Backend-for-Frontend pattern, and navigate at scale with React Router including parameters, nesting, and 404s. We cover a brief look at server-side rendering and GraphQL/Apollo to frame technical choices. Quality is enforced with Jest and React Testing Library for components, events, and custom hooks. Optional segments demonstrate how AI agent tools can accelerate architecture exploration and test creation, improving delivery speed without sacrificing reliability.

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

Learning Outcomes

- Stand up a new React project with sensible folder structure, assets, dependencies, and browser support.
- Explain React's single-page architecture and where server-side rendering fits at a high level.
- Author performant UI with JSX, keys, fragments, conditional operators, and the virtual DOM/Fiber model.
- Build reusable function components with clear props, defaults, prop types, and naming patterns.
- Handle events and component communication patterns, including passing callbacks and avoiding prop drilling.
- Manage state with core hooks (state, effect, callback) and craft maintainable custom hooks.
- Implement robust forms (controlled/uncontrolled), validation, and submissions across diverse inputs.
- Integrate REST APIs (fetch, CRUD, BFF), add routing with React Router (params, nesting, 404), and write unit tests with Jest and React Testing Library, including components and 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
- 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
- Prop Types
- 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

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)