

**Introduction to React and Next.js**

**Course Number:** RCT-128  
**Duration:** 5 days

**Overview**

Accelebrate’s Introduction to React and Next.js training course teaches attendees how to leverage Next.js, a powerful React framework. Participants learn how to set up a development environment, create a Next.js project, and understand React component fundamentals, including React hooks and custom hooks. Students gain practical experience implementing routes with App Router, fetching data, and understanding isomorphic rendering. The course also covers advanced topics like Next.js caching, styling techniques, and unit testing for React components. Finally, attendees learn how to deploy Next.js apps, ensuring a well-rounded understanding of both the development and deployment processes in the Next.js ecosystem.

**Prerequisites**

All attendees must have experience with modern JavaScript or TypeScript, including new language features like classes, modules, arrow functions, and destructuring.

**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 Needed on Each Student PC**

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

**Objectives**

* Understand why Next.js is a React framework
* Configure a development environment for Next.js
* Create a Next.js project
* Understand React component fundamentals
* Use React hooks and custom hooks
* Implement routes with app router
* Fetch data with Next.js
* Work with isomorphic rendering
* Explore Next.js caching
* Apply styling to Next.js apps
* Employ unit testing to test React components
* Deploy Next.js apps

**Outline**

* Introduction
* React and Next.js Overview
  + Why React and Next.js?
  + What Problem Does React solve?
  + What Problem Does Next.js solve?
* Development Environment
  + Install Node.js
  + Configure Visual Studio code
  + Install React Developer Tools
  + Next.js CLI
* Next.js Project Setup
  + Create a new project
  + Folder Structure
  + Browser Support
  + Styles and Assets
  + Dependencies
* React Components
  + Creating an Element
  + Create a Function Component
  + Rendering a Component
  + Composing & Reuse
* React 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
* React Component Props
  + Immutable Props
  + String Literals vs. Expressions
  + Prop Types
  + Default Prop Values
  + Naming Patterns for Props
* React Component Events
  + What are Events?
  + Common Events in React: Click and Change
  + Event Handlers and Functional Component
  + Passing Event Handlers via Props
* React Component Hooks
  + What is Component State?
  + State Hook
  + Effect Hook
  + Callback Hook
  + Custom Hooks
* Capture Data with Forms
  + Controlled and Uncontrolled Components
  + Enable Change Logic across Multiple Form Controls
  + Wiring up Input, Text Area, and Select
  + Handling Different Types of Input
* React Component Architecture
  + Reusable Components
  + Component Communication
  + Design Patterns
  + Container and Presentation Components
  + Defining Prop Drilling
* App Router
  + Define Routes
  + Pages and Layouts
  + Linking and Navigating
  + Dynamic Routes
  + Error Handling
* Fetching Data
  + Server-Side Data Fetching
  + Client-Side Data Fetching
  + Data Fetching Patterns
  + Connecting Forms to Data Fetching
  + Server-Only Forms
  + Form Validation
  + Updating Cookies
* Isomorphic Rendering
  + Server Rendering
  + Client Rendering
  + Server and Client Composition Patterns
* Caching
* Styling
  + CSS Modules
  + Tailwind CSS
  + CSS-in-JS (Styled Components)
  + Sass
  + Slice and Dice a Graphic Design File
  + Responsive Design
* Unit Testing Overview
  + Jest and 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
* Deployment
* Conclusion