Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

React Apps with React Router (formerly Remix)

Class Duration

35 hours of live training delivered over 5 days.

Student Prerequisites

- Experience with modern JavaScript or TypeScript
- Working knowledge of modern language features such as classes, modules, arrow functions, destructuring, and async/await
- Experience building React components with hooks

Target Audience

Web developers building full-stack React apps who want a web-standards-first alternative to Next.js, including teams upgrading React Router 7 framework-mode apps. The course assumes the client-side React covered in Advanced React.

Description

This course teaches full-stack React with React Router in framework mode. Students build route modules with server loaders and actions, client loaders, and typed route data. Middleware is always on, so authentication, sessions, and request-scoped data run through server and client middleware. Progressive enhancement gets real attention: forms that work before JavaScript loads, fetchers, pending and optimistic UI, and view transitions. Later modules cover rendering strategies, testing, deployment, and upgrading a React Router 7 app.

Learning Outcomes

- Choose between framework, data, and declarative modes.
- Set up a framework-mode project with the Vite plugin and generated route types.
- Define nested and dynamic routes with `routes.ts` or file-system conventions.

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

- Load and mutate data with loaders, actions, client loaders, and fetchers.
- Compose server and client middleware with a typed router context.
- Build progressively enhanced forms with pending and optimistic UI.
- Choose between server rendering, prerendering, and SPA mode.
- Test route modules with Vitest, React Testing Library, and Playwright.
- Deploy to Node.js, Cloudflare, Vercel, or Netlify.
- Upgrade a React Router 7 app to React Router 8.

Training Materials

Comprehensive courseware is distributed online at the start of class. All students receive a downloadable MP4 recording of the training.

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.

Versions covered: React Router 8 with React 19.3 or later, Vite 7 or later, and Node.js 22.22 or later.

Training Topics

React and React Router Overview

- What problem does React solve?
- What problem does framework mode solve?
- React Router vs. Next.js and TanStack Start
- Framework, data, and declarative modes
- How Remix became React Router

Framework Architecture

- The `react-router` and `@react-router/*` packages
- ESM-only packages
- The Vite plugin and dev server
- The Vite Environment API
- `Request`, `Response`, and the Fetch API

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

- Route module splitting

Development Environment

- Node.js LTS and Visual Studio Code
- React Developer Tools
- `react-router dev` and `react-router build`
- Route types with `react-router typegen`
- TypeScript for framework mode

Project Setup

- `create-react-router` and starter templates
- `root.tsx`, entry files, and route modules
- Configuring `react-router.config.ts`
- Styles, assets, and the `links` export
- Environment variables
- Server-only modules

React Components and JSX Review

- Function components and composition
- Expressions, attributes, and fragments
- Conditional rendering and lists
- Typing props with TypeScript
- Event handlers and callback props

React Hooks and Actions

- `useState` and `useEffect`
- What the React Compiler replaces
- `useActionState` and `useFormStatus`
- `useOptimistic` for optimistic UI
- `use` for promises and context
- Custom hooks

React Component Architecture

- Reusable components and communication
- Container and presentation components
- Avoiding prop drilling
- Colocating data needs with routes

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Routing

- `route`, `index`, `layout`, and `prefix`
- File-system routes with `@react-router/fs-routes`
- Dynamic, optional, and splat segments
- Nested routes and `<Outlet>`
- `<Link>`, `<NavLink>`, and `useNavigate`
- Redirects
- Route module exports

Data Loading and Mutations

- `loader` functions
- `action` functions
- `clientLoader` and `clientAction`
- Typed `loaderData` and `actionData`
- Revalidation and `shouldRevalidate`
- Streaming with `<Await>` and Suspense
- `meta` and `headers` from `loaderData`

Middleware and Router Context

- Server and client middleware chains
- `RouterContextProvider` and `createContext`
- Authentication, sessions, and cookies
- Logging and request timing
- Errors and redirects from middleware

Rendering Strategies

- Server rendering and hydration
- Static prerendering
- SPA mode
- `.server` and `.client` modules
- React Server Components status

Progressive Enhancement

- What is progressive enhancement?
- Forms that work before JavaScript loads
- Performance, resilience, and accessibility
- Building on the platform’s form model

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Pending and Optimistic UI

- Pending states with `useNavigation`
- Busy indicators and skeletons
- Optimistic UI with fetchers
- View transitions with `<ViewTransition>`

Form vs. Fetcher

- `<Form>` for navigating submissions
- `useActionData()` for action results
- `useFetcher()` for non-navigating mutations
- When the URL should change

State Management

- Server state vs. client state
- URL state with `useSearchParams`
- Loaders and actions instead of component state
- Avoiding copies of server data in `useState`

Network Concurrency Management

- Multiple in-flight submissions
- Interruption and cancellation
- Race conditions and revalidation order

Styling

- CSS Modules
- Tailwind CSS
- Sass
- Route-level stylesheets with the `links` export

Testing

- Vitest and React Testing Library
- Route stubs with `createRoutesStub`
- Testing loaders, actions, and middleware
- Mocking network boundaries
- Playwright for end-to-end tests

Upgrading from React Router 7

- New Node.js, React, and Vite minimums
- Clearing 7.x deprecation warnings

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

- Enabling `v8_middleware` on 7.x
- `v8_viteEnvironmentApi` and `v8_splitRouteModules`
- `v8_passThroughRequests`
- Trailing-slash-aware data requests
- Replacing `react-router-dom` imports
- ESM-only fallout in configs and tests
- `meta` reading `loaderData`
- `getLoadContext` returning `RouterContextProvider`
- Cloudflare development with `@cloudflare/vite-plugin`

Deployment

- Node.js with the built-in server or Express
- Cloudflare Workers
- Vercel and Netlify
- Containers and self-hosting
- Static hosting for prerendered and SPA builds