BUILD · TAUGHT WITH REACT

Frontend Web

Everything a user sees and clicks. You write HTML, CSS and JavaScript by hand before React gets involved, then build typed, routed apps that stay fast and work from the keyboard.

Foundational | 5 phases | 23 sessions | 35 contact hours

Syllabus

5 phases, 23 sessions. Each phase ends in something you have built.

1 The Browser

Build and style a real page before any framework hides how it works.

4 sessions
$58 alone

1.1 How a Page Arrives

Requests, HTML, the DOM, and DevTools as your first debugger.

FREE PREVIEW

1.2 Semantic HTML & Forms

Landmarks, labels, and native controls - structure a form a screen reader can actually use.

1.3 The Cascade, Honestly

Box model, specificity, cascade layers, and custom properties - why your margin isn't working.

1.4 Layout & Your First Deploy

Flexbox, Grid, container queries, then Git and a live URL before you leave.

By the end: **A responsive, keyboard-navigable site deployed to a real URL.**

2 JavaScript, Honestly

Make a page respond to a human with no framework doing it for you.

5 sessions
$58 alone

2.1	**The Language First**	Values, control flow, functions, and reading a stack trace instead of guessing.
2.2	**Arrays & Objects**	The shapes every UI is made of, and copying instead of mutating.
2.3	**DOM & Events**	Render a list from data, handle a click, and keep state in one plain variable.
2.4	**Async & fetch**	Promises, await, and the three states every request has: loading, data, error.
2.5	**Modules, npm, and Vite 8**	Split code across files, install a dependency, and run a real dev server and build.

By the end: **A vanilla-JS app, built with Vite, that fetches live data and handles loading, empty, and error.**

3 Thinking in React

Move from pages to components and put each piece of state in the right place.

5 sessions
$58 alone

3.1	**Components, Props, JSX**	Compose a static UI out of small components before anything moves.
3.2	**Lists & Keys**	Render an array, and see why key={index} breaks the moment the list reorders.
3.3	**State & Events**	useState, immutable updates, and why your array change didn't re-render.
3.4	**Lifting State & Composition**	Move state to the nearest common parent and pass children instead of drilling props.
3.5	**Forms and the Effect You Don't Need**	Controlled inputs, useActionState, and deriving values instead of syncing them in an effect.

By the end: **An interactive multi-component React app with no data fetching and no behaviour you can't explain.**

4 Types, Data, Routes

Talk to a real API without hand-rolled fetching, and let the types catch the rest.

5 sessions
$58 alone

4.1	**TypeScript Where It Pays**	Typed props, state, and API responses - the useful ten percent of TypeScript 7.
4.2	**Routing with React Router v8**	Nested routes, URL params, and keeping shareable state in the URL.
4.3	**Server State with TanStack Query**	Caching, refetching, and retiring the useEffect fetch that caused your race condition.
4.4	**Suspense & Error Boundaries**	One loading fallback, one error boundary, and no more blank white screen.
4.5	**Client State, Sparingly**	Context for theme and session, and why most apps never need a store library.

By the end: **A typed, routed React app running against a real API with honest loading and error UI.**

5 Ship It

Get an app off your machine and prove it holds up for someone who isn't you.

4 sessions
$58 alone

5.1	**Tests You'll Actually Run**	Vitest browser mode for components, Playwright for the one flow that must never break.
5.2	**The Accessibility Audit**	A keyboard-only pass, a screen reader pass, and fixing what axe finds against WCAG 2.2 AA.
5.3	**Performance You Can Measure**	Core Web Vitals, code splitting, and React Compiler instead of hand-placed useMemo.
5.4	**Build, Deploy, Hand Over**	Env config, a CI check, a live URL, and a README a stranger can follow.

By the end: **A deployed app with a passing test suite, a WCAG 2.2 AA audit, and a measured performance budget.**

Tools you will use

React 19.2 · TypeScript 7 · Vite 8 · Tailwind CSS v4 · React Router v8 · TanStack Query · Vitest 4 · Playwright

What you will build

1. **Hand-Built Landing Page**
   No framework, keyboard-navigable, deployed to a real URL
2. **Vanilla DOM Data App**
   Fetch, render, filter, and handle every state with zero libraries
3. **Typed React Dashboard**
   Routes, cached server state, tests, and a WCAG 2.2 AA pass

Registration

Phase 1: The Browser	$58
Phase 2: JavaScript, Honestly	$58
Phase 3: Thinking in React	$58
Phase 4: Types, Data, Routes	$58
Phase 5: Ship It	$58
Whole track, all 5 phases	**$240**

Buying every phase separately costs $290, so the whole track saves $50.

No payment is taken online. Registering creates an order and payment instructions are sent with it. Access opens once payment is confirmed.