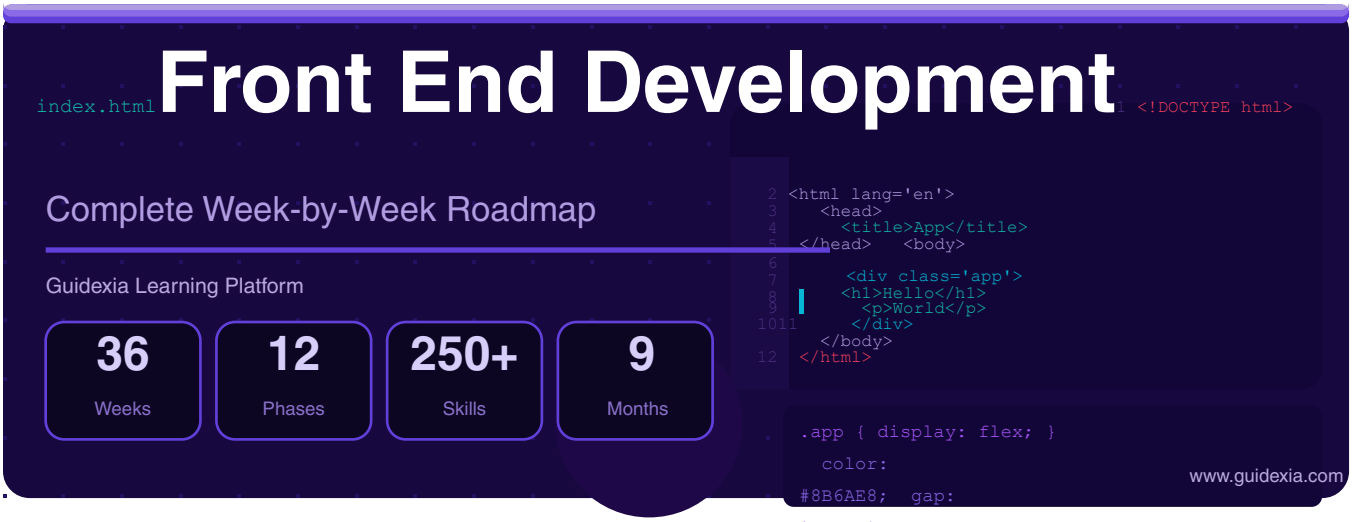

This is your complete, week-by-week Front End Development roadmap — 12 phases across 36 weeks (9 months). Starting from how the web works, you progress through HTML5, CSS3, JavaScript, TypeScript, React with hooks and state management, Next.js full-stack, Tailwind CSS, testing with Vitest and Playwright, performance optimisation, and a full portfolio and career launch strategy.

ROADMAP OVERVIEW — 12 PHASES | 36 WEEKS | 9 MONTHS

#	Phase	Timeline	Key Skills
01	Internet, Web & Developer Setup	Month 1	HTTP, DNS, Git, VS Code, Terminal
02	HTML5 — Structure of the Web	Month 1-2	Semantic HTML, Forms, ARIA, SVG
03	CSS3 — Styling & Layouts	Month 2-3	Flexbox, Grid, Media Queries, CSS Variables
04	JavaScript Fundamentals	Month 3-4	Closures, async/await, Fetch, Promises
05	DOM Manipulation & Browser APIs	Month 4	DOM, Events, localStorage, Web APIs
06	TypeScript	Month 5	TypeScript, Generics, Utility Types
07	React — The Industry Standard	Month 5-6	React, Hooks, React Query, Router
08	State Management & Advanced React	Month 6-7	Zustand, Redux Toolkit, React Hook Form
09	Next.js — Full-Stack React Framework	Month 7	Next.js, SSR/SSG, Server Components
10	CSS Frameworks, Design Systems & Tooling	Month 7-8	Tailwind, shadcn/ui, Vite, Storybook
11	Testing, Performance & Web Vitals	Month 8	Vitest, Testing Library, Playwright, Lighthouse

12	Portfolio, Projects & Career Launch	Month 8-9	Portfolio, GitHub, Interviews, Career

CORE TECH STACK HTML5 | CSS3 | JavaScript (ES2024) | TypeScript | React | Next.js | Tailwind CSS | Vitest | Playwright

MONTH 01 Internet + HTML Foundations

1 Internet, Web & Developer Setup

Weeks 1-2 (Month 1)

PHASE 01

Before writing a single line of code, you need to understand how the web works and set up your environment correctly. This foundation shapes everything that follows — from debugging to deployment.

WEEK Weeks 1-2 — How the Web Works & Dev Environment

Internet & Web Fundamentals

- How browsers render pages
- HTTP/HTTPS & request-response cycle
- DNS, IP addresses & hosting
- Domains & web servers

Developer Setup

- VS Code setup & essential extensions
- Chrome DevTools orientation
- Git & GitHub workflow
- Command line basics

SKILLS GAINED

Browser Rendering	**HTTP/HTTPS**	**DNS**	**Git**	**GitHub**
VS Code	**Chrome DevTools**	**Terminal**	**Command Line**	**Web Servers**

Outcome:
You understand how the web works, have a professional dev environment configured, and can use Git for version control from day one.

2 HTML5 — Structure of the Web

Weeks 3-5 (Month 1-2)

PHASE 02

HTML is the skeleton of every web page. This phase goes deep — not just tags, but semantic HTML, forms, accessibility, and the browser's built-in capabilities that most developers overlook.

WEEK Week 3 — HTML Fundamentals

Core HTML

- Document structure (DOCTYPE, head, body)
- Text: h1-h6, p, span, strong, em
- Links, images & media
- Lists: ul, ol, dl

HTML Attributes

- id, class, data-* attributes
- href, src, alt attributes
- Boolean attributes
- Custom data attributes

WEEK Week 4 — Semantic HTML & Accessibility

Semantic Elements

- header, nav, main, article, section
- aside, footer, figure, figcaption
- time, address, mark
- Why semantics matter for SEO

Accessibility (a11y)

- ARIA roles, labels, descriptions
- Tab order & keyboard navigation
- Screen reader compatibility
- Alt text best practices

WEEK Week 5 — HTML Forms & Media

Forms

- input types (text, email, number, date, file)
- select, textarea, checkbox, radio
- Form validation (required, pattern, min/max)
- fieldset & legend

Media & Embedding

- video & audio elements
- picture & srcset for responsive images
- iframe & embedding
- SVG inline & as img

SKILLS GAINED

HTML5	Semantic HTML	header/nav/main	Accessibility	ARIA
Alt Text	Forms	Input Types	Form Validation	SVG
Responsive Images	srcset	Video/Audio	Keyboard Navigation	

Outcome:
You can write clean semantic HTML5, build accessible forms, embed media correctly, and understand ARIA for screen reader support.

MONTH 02 CSS Layouts + Responsive Design

3 CSS3 — Styling & Layouts

Weeks 6-9 (Month 2-3)

PHASE 03

CSS transforms HTML into beautiful interfaces. This phase covers the full CSS power stack: selectors, box model, Flexbox, Grid, animations, and responsive design — everything needed to build any layout ever designed.

WEEK Week 6 — CSS Fundamentals

CSS Core

- Selectors: element, class, id, attribute
- Specificity & cascade
- Box model: margin, border, padding, content
- Display: block, inline, inline-block

Visual Styling

- Colors: hex, rgb, hsl, oklch
- Typography: font-family, size, weight, line-height
- Backgrounds: color, image, gradient
- Shadows: box-shadow, text-shadow

WEEK Week 7 — CSS Flexbox

Flexbox Container

- display: flex, flex-direction
- justify-content & align-items
- flex-wrap & align-content
- gap property

Flexbox Items

- flex-grow, flex-shrink, flex-basis
- align-self & order
- Shorthand: flex: 1
- Common Flexbox patterns

WEEK Week 8 — CSS Grid

Grid Container

- display: grid, grid-template-columns/rows
- fr unit & repeat()
- grid-gap / gap
- Named grid areas

Grid Items

- grid-column & grid-row span
- grid-area
- Auto-placement algorithm
- Grid vs Flexbox — choosing

WEEK Week 9 — Responsive Design & CSS Advanced

Responsive Design

- Media queries (@media)
- Mobile-first approach
- Viewport meta tag
- Fluid typography (clamp)

CSS Advanced

- Custom properties (--var)
- CSS functions: calc(), min(), max()
- Transitions & animations
- Pseudo-classes & pseudo-elements

SKILLS GAINED

CSS3	**Selectors**	**Specificity**	**Box Model**	**Flexbox**
CSS Grid	**fr Unit**	**Named Areas**	**Media Queries**	**Mobile-First**
Viewport	**CSS Variables**	**clamp()**	**calc()**	**Transitions**
Animations	**Pseudo-classes**	**Responsive Design**		

Outcome:
You can build any layout using Flexbox and Grid, implement responsive designs with media queries, and use CSS custom properties and animations.

MONTH 03 JavaScript Fundamentals

4 JavaScript Fundamentals

Weeks 10-13 (Month 3-4)

PHASE 04

JavaScript is the programming language of the web — the only language that runs natively in every browser. This phase builds deep JS fundamentals: not just syntax, but closures, the event loop, prototypes, and async patterns.

WEEK Week 10 — JS Core

Variables & Types

- let, const, var — differences
- Primitive types & typeof
- Type coercion & truthy/falsy
- Template literals

Operators & Control Flow

- Comparison: == vs ===
- Logical: &&, ||, ??
- Ternary operator
- switch & for...of / for...in

WEEK Week 11 — Functions & Scope

Functions

- Function declarations vs expressions
- Arrow functions
- Default parameters
- Rest parameters & spread

Scope & Closures

- var/let/const scoping
- Lexical scope
- Closures — practical uses
- IIFE pattern

WEEK Week 12 — Arrays, Objects & Destructuring

Arrays

- map, filter, reduce, find, some
- flat, flatMap, Array.from
- Spread with arrays
- Immutable array patterns

Objects & Destructuring

- Object shorthand & methods
- Destructuring: array & object
- Rest in destructuring
- Object.keys/values/entries

WEEK Week 13 — Async JavaScript

Callbacks & Promises

- Callback pattern & callback hell
- Promise: then/catch/finally
- Promise.all, Promise.race
- Error handling in promises

Async/Await & Fetch

- async/await syntax
- try/catch with async
- Fetch API for HTTP requests
- Working with JSON responses

SKILLS GAINED

JavaScript	let/const	Template Literals	Arrow Functions	Closures
Scope	map/filter/reduce	Destructuring	Spread/Rest	Promises
async/await	Fetch API	JSON	Error Handling	Event Loop
Type Coercion				

Outcome:
You are proficient in modern JavaScript — closures, async patterns, array methods, destructuring — and can fetch and handle API data confidently.

MONTH 04 DOM + Browser APIs

5 DOM Manipulation & Browser APIs

Weeks 14-15 (Month 4)

PHASE 05

JavaScript's power comes from controlling the browser. This phase teaches DOM manipulation, event handling, browser storage, and the Web APIs that make modern applications feel native.

WEEK Week 14 — DOM & Events

DOM Manipulation

- querySelector & querySelectorAll
- createElement, append, remove
- classList: add, remove, toggle
- innerHTML vs textContent vs innerText

Event Handling

- addEventListener & removeEventListener
- Event object & currentTarget
- Event delegation
- Event propagation: bubbling & capturing

WEEK Week 15 — Browser APIs

Storage & History

- localStorage & sessionStorage
- IndexedDB concept
- History API (pushState)
- URL & URLSearchParams

Modern Web APIs

- Intersection Observer
- MutationObserver
- Web Workers intro
- requestAnimationFrame

SKILLS GAINED

DOM	querySelector	createElement	classList	Event Handling
Event Delegation	Event Bubbling	localStorage	sessionStorage	History API
Intersection Observer	MutationObserver	requestAnimationFrame	Web Workers	

Outcome:
You can manipulate the DOM dynamically, handle any browser event pattern, use browser storage, and work with modern Web APIs.

MONTH 05 TypeScript + React Core

6 TypeScript

Weeks 16-18 (Month 5)

PHASE 06

TypeScript is now the default language of professional front-end development. Every major framework — React, Vue, Angular — is written in TypeScript. This phase takes you from zero to production-ready TypeScript.

WEEK Week 16 — TypeScript Basics

Type System

- Basic types: string, number, boolean, null
- Arrays & tuple types
- Union & intersection types
- Type aliases & interfaces

Functions & Objects

- Function parameter & return types
- Optional & default parameters
- Object type shapes
- Readonly & partial

WEEK Week 17 — TypeScript Advanced

Generics

- Generic functions & interfaces
- Constraining generics
- Built-in generics: Partial, Required, Pick, Omit
- Generic classes

Advanced Types

- Type narrowing & type guards
- Discriminated unions
- Template literal types
- Conditional types

WEEK Week 18 — TypeScript in Practice

TypeScript Config

- tsconfig.json setup
- Strict mode options
- Module resolution
- Declaration files (.d.ts)

TypeScript Patterns

- Type assertions vs type predicates
- Enums vs const objects
- Mapped types
- Utility types deep dive

SKILLS GAINED

TypeScript	Type Aliases	Interfaces	Union Types	Intersection Types
Generics	Partial/Required/Pick/Omit	Type Guards	Discriminated Unions	Conditional Types
tsconfig	Strict Mode	Mapped Types	Declaration Files	Template Literal Types

Outcome:
You can write fully typed TypeScript applications, use generics, utility types, and configure TypeScript for any project.

7

React — The Industry Standard

Weeks 19-22 (Month 5-6)

PHASE 07

React powers Facebook, Airbnb, and thousands of top companies. It is the most in-demand front-end skill in 2024-2025. This phase takes you from React basics through hooks, state management, and real API integration.

WEEK Week 19 — React Core

React Fundamentals

- Vite + React + TypeScript setup
- JSX syntax & expressions
- Functional components
- Props & prop types

React State

- useState hook
- Controlled vs uncontrolled inputs
- Conditional rendering
- Lists & keys

WEEK Week 20 — React Hooks

Core Hooks

- useEffect & cleanup
- useRef for DOM access
- useMemo & performance
- useCallback for stable references

Custom Hooks

- Extracting logic into custom hooks
- useLocalStorage hook
- useFetch hook
- useDebounce hook

WEEK Week 21 — React Router & Data Fetching

React Router v6

- BrowserRouter & Routes
- Link, NavLink, useNavigate
- useParams & useSearchParams
- Protected routes

Data Fetching

- Fetching in useEffect
- React Query (TanStack Query)
- Loading & error states
- Caching & refetching

WEEK Week 22 — React Patterns & Performance

Component Patterns

- Compound components
- Render props
- Higher-Order Components (HOC)
- Context API for global state

React Performance

- React.memo
- Code splitting & lazy loading
- Suspense boundaries
- Profiler DevTools

SKILLS GAINED

React	JSX	useState	useEffect	useRef
useMemo	useCallback	Custom Hooks	React Router	useParams
React Query	Context API	React.memo	Code Splitting	Suspense
Compound Components	HOC	TypeScript + React		

Outcome:
You can build complete React applications with hooks, routing, data fetching with React Query, and performance optimisation.

MONTH 06 — React Advanced + State

8 State Management & Advanced React

Weeks 23-24 (Month 6-7)

PHASE 08

Complex apps need more than local state. This phase covers the state management ecosystem — Zustand, Redux Toolkit, and Jotai — and advanced React patterns used at scale in production applications.

WEEK Week 23 — State Management

Zustand

- Store creation
- Selectors & subscriptions
- Middleware (persist, devtools)
- Zustand with TypeScript

Redux Toolkit (RTK)

- createSlice & reducers
- createAsyncThunk for async
- RTK Query for data fetching
- Redux DevTools

WEEK Week 24 — Advanced React Patterns

Forms

- React Hook Form
- Zod schema validation
- Field arrays
- Form performance

Advanced Patterns

- Portals for modals
- Error boundaries
- useReducer for complex state
- Optimistic updates

SKILLS GAINED

Zustand	Redux Toolkit	createSlice	createAsyncThunk	RTK Query
Redux DevTools	React Hook Form	Zod Validation	Portals	Error Boundaries
useReducer	Optimistic Updates	State Selectors	Persist Middleware	

Outcome:
You can implement scalable state management with Zustand or Redux Toolkit, build complex validated forms, and handle error boundaries professionally.

MONTH 07 Next.js + CSS Frameworks

9 Next.js — Full-Stack React Framework

Weeks 25-27 (Month 7)

PHASE 09

Next.js is the most popular React framework — used by Vercel, Shopify, TikTok, and thousands of companies. It adds SSR, SSG, file-based routing, and API routes, taking React from a UI library to a full-stack solution.

WEEK Week 25 — Next.js App Router

App Router Basics

- App Router vs Pages Router
- Server vs Client components
- Loading & error UI
- Layout nesting

Routing

- File-based routing
- Dynamic routes ([id])
- Route groups (folder)
- Parallel & intercepting routes

WEEK Week 26 — Data Fetching & Rendering

Rendering Strategies

- Server-Side Rendering (SSR)
- Static Site Generation (SSG)
- Incremental Static Regeneration (ISR)
- Streaming with Suspense

Data Fetching

- fetch in Server Components
- React Server Components
- Server Actions for mutations
- Caching & revalidation

WEEK Week 27 — Next.js Advanced

API Routes & Middleware

- Route Handlers (API routes)
- Middleware for auth
- Edge runtime
- Next.js Image & Font optimisation

Deployment

- Deploying to Vercel (free)
- Environment variables
- next.config.js
- Performance: Core Web Vitals

SKILLS GAINED

Next.js	App Router	Server Components	Client Components	Dynamic Routes
SSR	SSG	ISR	Streaming	Suspense
Server Actions	Route Handlers	Middleware	Vercel Deploy	Core Web Vitals
Image Optimisation	Edge Runtime			

Outcome:
You can build full-stack Next.js applications with SSR/SSG, Server Components, Server Actions, deploy to Vercel, and optimise Core Web Vitals.

10 CSS Frameworks, Design Systems & Tooling

Weeks 28-29 (Month 7-8)

PHASE 10

Professional front-end work uses battle-tested CSS frameworks and build tooling. This phase covers Tailwind CSS, component libraries, Storybook, and the modern build pipeline that every senior FE developer commands.

WEEK Week 28 — Tailwind CSS & Component Libraries

Tailwind CSS

- Utility-first philosophy
- Responsive prefixes (sm:, md:, lg:)
- Dark mode
- Custom theme (tailwind.config)

Component Libraries

- shadcn/ui components
- Radix UI primitives
- Headless UI
- Combining Tailwind + component libs

WEEK Week 29 — Build Tools & Storybook

Modern Tooling

- Vite: config, plugins, environment
- ESLint + Prettier config
- Husky & lint-staged
- Module aliases & path mapping

Storybook

- Storybook setup
- Writing stories
- Args & controls
- Visual regression testing

SKILLS GAINED

Tailwind CSS	Responsive Prefixes	Dark Mode	tailwind.config	shadcn/ui
Radix UI	Headless UI	Vite	ESLint	Prettier
Husky	lint-staged	Module Aliases	Storybook	Stories
Visual Regression				

Outcome:
You can build UIs with Tailwind CSS and shadcn/ui, configure Vite and ESLint, and document components in Storybook for team use.

MONTH 08 Testing + Performance + Portfolio

11 Testing, Performance & Web Vitals

Weeks 30-32 (Month 8)

PHASE 11

Professional front-end engineering requires tested, fast code. This phase teaches you to write tests with Vitest and Testing Library, measure and optimise Core Web Vitals, and implement accessibility testing.

WEEK Week 30 — Testing Front-End Code

Unit Testing

- Vitest setup & syntax
- Testing pure functions & hooks
- vi.mock for mocking
- Coverage reports

Component Testing

- React Testing Library
- getBy, findBy, queryBy queries
- userEvent interactions
- Testing with MSW (Mock Service Worker)

WEEK Week 31 — E2E Testing

Playwright

- Playwright setup & browser drivers
- Writing E2E test scripts
- Page Object Model pattern
- CI integration

Cypress

- Cypress setup & commands
- Intercept & stub network
- Component testing with Cypress
- Cypress Cloud

WEEK Week 32 — Performance & Accessibility

Core Web Vitals

- LCP, CLS, INP explained
- Lighthouse audit
- Web.dev measure tool
- Performance budgets

Accessibility Testing

- axe-core & Deque axe
- WAVE browser extension
- Manual keyboard testing
- Colour contrast checking

SKILLS GAINED

Vitest	React Testing Library	userEvent	MSW	Playwright
Cypress	Page Object Model	LCP	CLS	INP
Lighthouse	Performance Budget	axe-core	WAVE	Keyboard Testing
Colour Contrast	CI Testing			

Outcome:
You can write unit, component, and E2E tests, audit and improve Core Web Vitals, and test accessibility systematically.

12 Portfolio, Projects & Career Launch

Weeks 33-36 (Month 8-9)

PHASE 12

Your skills are invisible without evidence. This final phase builds a portfolio of deployed projects, polishes your GitHub, and prepares you for front-end developer interviews at top companies.

WEEK Weeks 33-34 — Capstone Projects & Portfolio

Capstone Projects

- Full-stack Next.js app with auth + API
- React + TypeScript SPA with state management
- Portfolio website (Next.js + Tailwind)
- Each project: deployed + GitHub + README

Portfolio Quality

- 3-5 projects with live links
- Code quality & comments
- README with tech stack & screenshots
- Personal brand on GitHub & LinkedIn

WEEK Weeks 35-36 — Interviews & Career

Technical Interviews

- JS fundamentals questions (closures, prototypes)
- React concepts (reconciliation, virtual DOM)
- CSS layout challenges
- Algorithm basics (arrays, strings)

Career Launch

- FE developer resume format
- Quantify impact: 'improved LCP by 40%'
- Take-home project strategy
- Negotiating front-end salaries

SKILLS GAINED

Capstone Projects	Next.js Portfolio	GitHub	README	Deployed Apps
Vercel	FE Resume	LinkedIn	JS Interview Questions	React Interview
CSS Interview	Algorithm Basics	Take-Home Projects	Salary Negotiation	Career Strategy

Outcome:
You have 3-5 deployed portfolio projects, a polished GitHub, and complete interview preparation to land front-end developer roles.

DEVELOPER SUCCESS TIPS & FREE RESOURCE DIRECTORY

Build Something Every Single Week	From week 3 onwards, build a small project with everything you learn. Imperfect projects teach more than perfect tutorials.
Read MDN, Not Random Blogs	The MDN Web Docs are the gold standard for HTML, CSS and JavaScript. Develop the habit of checking MDN before Stack Overflow.
Master Vanilla JS Before React	Do not skip Phases 4-5 to rush to React. Developers who skip JS fundamentals hit a wall in every React interview they attempt.
TypeScript Is Non-Negotiable in 2025	Every serious front-end team uses TypeScript. Invest time in Phase 6 — it will make you a more employable and better engineer.
Understand React, Not Just Copy Patterns	Know why React re-renders, what the reconciliation algorithm does, and how the virtual DOM works. These are interview questions at every level.
Deploy Every Project — Never Skip This	Use Vercel free tier. Every project must be live. A GitHub repo without a live link is invisible to hiring managers and clients.
Contribute to Open Source Early	Fix a typo in React docs, improve a README, add a Tailwind example. It builds GitHub history and genuine community credibility.

FREE RESOURCES

Resource	URL	Best For
MDN Web Docs	developer.mozilla.org	HTML, CSS, JS — the definitive reference
The Odin Project	theodinproject.com	Free full-stack curriculum, HTML → Node
javascript.info	javascript.info	Best free modern JavaScript guide
TypeScript Handbook	typescriptlang.org/docs	Official TypeScript docs & handbook
React Docs	react.dev	Official React documentation
Next.js Docs	nextjs.org/docs	Official Next.js documentation
Tailwind Docs	tailwindcss.com/docs	Official Tailwind CSS documentation
CSS Tricks	css-tricks.com	Flexbox, Grid, CSS guides & almanac
Josh W Comeau	joshwcomeau.com	Best CSS and React deep-dive articles
Kent C. Dodds Blog	kentcdodds.com	React patterns, testing, career advice
Fireship YouTube	youtube.com/@Fireship	Quick modern web tech explainers
Guidexia	www.guidexia.com	Structured roadmaps, mentorship & community

In 9 months, you will build the web. Every pixel, every interaction, every experience.

www.guidexia.com