

Modern Frontend Frameworks and Accessible UI Engineering Training Course

21 – 25 December 2026
Amsterdam

AGILE LEADERS
Training Center

AGILE LEADERS
Training Center

Modern Frontend Frameworks and Accessible UI Engineering Training Course

Ref: 1732_230157 **Date:** 21 – 25 December 2026 **Location:** Amsterdam **Fees:** 5700 Euro

Overview

Modern Frontend Frameworks and Accessible UI Engineering Training Course is a one-week intermediate course for developers and UI engineers who leave with a Production Frontend Application Pack. Participants compare React, Angular, and Vue architecture while building component-based web interfaces, accessible UI engineering controls, routing, forms, state management, API integration, tests, performance checks, and deployment evidence. The course turns fragmented frontend practices into a traceable delivery workflow. Agile Leaders Training Center delivers modern frontend framework engineering.

Who Should Attend

- Teams responsible for developing browser-based product interfaces
- Functions responsible for reusable components and frontend architecture
- Teams responsible for integrating web interfaces with APIs
- Functions responsible for testing, accessibility, and performance evidence
- Technical leads responsible for frontend delivery standards and reviews

The course assumes participants already write HTML, CSS, and JavaScript for web applications and leaves out introductory coding, visual design software, UX research, mobile app development, and backend service implementation.

Departments and Industries

The course supports departments and industries that build and maintain production web interfaces.

- Software engineering, digital product, and frontend development teams
- Banking, insurance, and digital commerce platforms
- Healthcare portals and education technology services
- Government digital services and professional service applications
- Telecommunications, logistics, and customer self-service channels

Learning Objectives

By the end of this course, participants will be able to:

- Compare React, Angular, and Vue component architecture and delivery tradeoffs
- Build responsive component libraries with design tokens and typed interfaces
- Apply routing, forms, state management, and API integration patterns
- Evaluate web interfaces against WCAG 2.2 accessibility criteria
- Use automated tests, performance budgets, and frontend security checks
- Build and deploy a production-style single-page application

Course Agenda

Day 1: Framework Architecture and Project Setup

- React, Angular, and Vue Architecture Comparison Matrix
- Vite Project Scaffold and Module Boundary Plan
- TypeScript Interface and Data Contract Design
- Component Composition and Dependency Flow Diagram
- Frontend Repository and Git Branching Convention

Day 2: Components and Responsive Interfaces

- Design Token and CSS Custom Property System
- Reusable Component API and Prop Contract
- CSS Grid and Flexbox Responsive Layout Pattern
- Semantic HTML and ARIA Interaction Structure
- Storybook Component State and Documentation Set

Day 3: State, Routing, Forms, and APIs

- React Router Navigation and Route Guard Pattern
- Local, Shared, and Server State Decision Map
- Form Validation and Error-State Specification
- Fetch API Integration and Failure-Handling Flow
- Authentication Token and Browser Storage Risk Checklist

Day 4: Quality, Accessibility, and Delivery

- WCAG 2.2 Keyboard and Focus Review
- Testing Library Component and Interaction Test Suite
- Lighthouse Performance Budget and Rendering Audit
- Content Security Policy and Dependency Risk Review
- Build Pipeline and Static Hosting Deployment Checklist

Day 5: Frontend Application Practice

- Exercise: Framework Architecture Selection Review
- Exercise: Responsive Component Assembly and State Integration
- Exercise: Accessible Form and Keyboard Interaction Test
- Exercise: Performance, Security, and Deployment Readiness Review
- Capstone Exercise: Production Frontend Application Pack

Practical Exercises

The course uses suggested activities to convert interface requirements into tested implementation evidence.

- Suggested activity: compare React, Angular, and Vue for a financial service interface and document the architecture choice.
- Suggested activity: build a responsive healthcare portal component with states, validation, and API data.
- Suggested activity: test keyboard paths, focus behavior, rendering performance, and failure states.
- Suggested activity: assemble the production application pack with code, tests, audits, and deployment records.

FAQs

Who suits modern frontend frameworks training and what does it assume?

Developers, UI engineers, software engineers, and technical leads suit the course. It assumes working knowledge of HTML, CSS, and JavaScript and excludes introductory programming.

How does accessible UI engineering differ from UI design training?

Accessible UI engineering implements component behavior, semantic markup, keyboard operation, state, tests, performance, and deployment, while UI design training concentrates on visual systems, prototypes, and handoff specifications.

Should frontend developers learn React, Angular, and Vue together?

A comparative foundation helps developers recognize shared component, routing, state, and data patterns. The course uses React for implementation depth while comparing architectural choices in Angular and Vue.

How is WCAG 2.2 applied in frontend framework development?

WCAG 2.2 is applied through semantic structure, keyboard access, visible focus, labels, error handling, contrast, target behavior, and repeatable accessibility checks across responsive interface states.

AGILE LEADERS
Training Center

Conclusion

Participants take a Production Frontend Application Pack back to work. The pack connects architecture decisions, components, state, routes, forms, API behavior, accessibility evidence, automated tests, performance checks, security controls, and deployment records. It changes isolated interface code into a reviewable frontend delivery workflow that teams can maintain and extend.