Fiche du cours

60 h

Titre :

Web230 - JavaScript & React : Introduction

Description :

Découverte JavaScript moderne (ES6+) et React pour créer une SPA complète. On progresse des bases du langage vers composants, état & hooks, formulaires, appels API, routing et tests, avec une mise en place outillée (Vite, npm, linter). Vous apprendrez à structurer le projet (dossiers, variables d'environnement), gérer l'asynchrone (chargements/erreurs) et partager un état global simple (Context).

Objectifs :

- Maîtriser les bases JS ES6+ (modules, tableaux, objets, async/await).*
- Manipuler le DOM et les événements (notions) ; requêtes HTTP.*
- Démarrer un projet React (Vite), créer des composants avec JSX.*
- Gérer props, state et hooks de base (useState, useEffect).*
- Gérer formulaires, listes/keys, erreurs et chargement.*
- Mettre en place le routing (React Router) et le Context global.*
- Styliser (CSS Modules / Tailwind – au choix) et considérer accessibilité.*
- Écrire des tests (Jest/React Testing Library) et préparer un build.

Chapitres :

1. JavaScript moderne : syntaxe ES6+, tableaux/objets, fonctions, promesses, async/await*
2. Web & API : DOM, événements (aperçu), fetch, JSON, gestion d'erreurs*
3. Outillage : Node/npm, Vite, scripts, linter/formatter ; création d'un projet React*
4. Composants & JSX : composition, props, bonnes pratiques*
5. State & Hooks : useState, useEffect, cycle de vie, chargement/erreurs*
6. Listes & Formulaires : keys, composants contrôlés, validation simple*
7. Routing : React Router (routes, params, navigation, 404)*
8. Contexte & Data : Context API, patterns simples (Reducer – aperçu), appels API structurés*
9. UI & Qualité : CSS Modules/Tailwind, accessibilité, i18n (aperçu), debug, tests (Jest/RTL)*
10. Mini-projet : petite SPA (liste/détail + formulaire + API) avec README & build

À la fin :
Vous saurez créer une SPA basique en React : composants propres, state/hooks, routing, formulaires et appels API, avec un projet outillé (Vite, linter, tests) — prêt à montrer en portfolio. Vous saurez aussi structurer vos appels réseau (chargement/erreurs) et partager un état global simple via le Context API. Enfin, vous pourrez builder et publier un aperçu en ligne (Vite + hébergeur type Vercel/Netlify) avec un README clair.