Formation **Nuxt**

Passez de simple “Vue-iste” à expert Nuxt ! En trois jours intensifs, cette formation Nuxt.js vous guide pas à pas de l’installation jusqu’au déploiement en production : routage automatique, SSR/SSG pour un SEO au top, Composition API et Pinia pour une gestion d’état moderne, TanStack Query pour des données toujours fraîches, tests Cypress/Vitest pour une qualité béton, puis optimisation Lighthouse avant de pousser votre app sur Vercel ou Netlify. Pensée pour les développeurs déjà aguerris à Vue.js, elle mêle ateliers pratiques et bonnes pratiques d’architecture pour vous permettre de livrer des sites ultra-performants, maintenables et prêts pour la scalabilité.

Durée
3 jours

Objectifs pédagogiques

- Comprendre les fondamentaux de Nuxt.js et les avantages de son utilisation
- Mettre en place un projet Nuxt.js avec une configuration personnalisée
- Créer des pages statiques et dynamiques en utilisant le système de routage automatique
- Optimiser les performances et le SEO avec SSR, SSG et lazy loading
- Déployer et maintenir une application Nuxt.js sur Vercel, Netlify, etc

Public
Développeurs

Prérequis
Bonne connaissance des technologies web (HTML, CSS, JavaScript).
Expérience avec Vue.js (Vue 2 ou Vue 3 selon la version de Nuxt utilisée).

Programme de formation

Phase d'inclusion

Accueil des participants, présentation des objectifs et contextes professionnels de chacun.

Introduction à Nuxt.js et son écosystème

Vue d'ensemble de l'écosystème Nuxt et ses avantages.
Outils de développement : Node.js, NPM, Vite.
Rappels des bases de Vue.js : directives, événements, composants.

Les bases de Nuxt.js

Structure d'un projet Nuxt.js (arborescence et conventions).
Fonctionnement du Virtual DOM et rendu des composants.
Création et réutilisation de composants Vue/Nuxt.
Communication entre composants : props, events, slots.

Options API vs Composition API

Découverte de l'Options API.
Introduction à la Composition API (setup, reactive, ref, computed).
Comparaison et cas d'usage selon le type de projet.

Gestion des formulaires et validation

Utilisation de v-model et binding des données.
Validation et gestion des erreurs.
Bonnes pratiques pour des formulaires robustes.

Composants avancés et gestion du cycle de vie

Hooks du cycle de vie (onMounted, onUpdated, etc.).
Composables (fonctions réutilisables avec la Composition API).
Téléportation, suspense et gestion des composants dynamiques.

Composants spécifiques à Nuxt.js

ClientOnly : rendre un composant uniquement côté client.
NuxtLayout et structuration globale d'un projet.
NuxtPage et gestion automatique des routes.
NuxtLink pour la navigation interne.
NuxtImg et NuxtPicture pour l'optimisation des images.

Gestion des données avec Pinia et API externe

Introduction à Pinia et différence avec Vuex.
Partage de données entre composants avec state, getters, actions.
Utilisation d'une API externe ou interne avec useFetch et useAsyncData.
Gestion des erreurs et des états de chargement.

La « magie » de Nuxt.js

Routage automatique basé sur l'arborescence des fichiers.
SSG (Static Site Generation) : génération de sites statiques.
SSR (Server Side Rendering) : rendu côté serveur.
Optimisations SEO : balises meta, gestion des réseaux sociaux.
Lazy loading, préchargement et optimisation des performances.

Tests et automatisation

Introduction aux tests end-to-end avec Cypress.
Configuration de base dans un projet Nuxt.js.

Écriture de tests (login, navigation, formulaires, etc.).
Tests unitaires avec Vitest ou Jest (selon préférence).
Montage des composants, assertions et mocks.
Couverture de code et bonnes pratiques.

Optimisation des données avec TanStack Query

Présentation de TanStack Query (gestion avancée du cache).
Intégration avec Nuxt.js et la Composition API.
Stratégies d'optimisation et invalidation intelligente des requêtes.

Bonnes pratiques et patterns avancés

Organisation des dossiers et fichiers pour un projet scalable.
Utilisation de plugins et modules Nuxt.js.
Sécurisation et gestion des permissions utilisateur.

Performance et déploiement

Techniques avancées d'optimisation (code splitting, pré-rendu).
Mesure des performances avec Lighthouse et Web Vitals.
Déploiement sur Vercel, Netlify ou plateformes PaaS.
Configuration pour la production et gestion des variables d'environnement.

Moyens et méthodes pédagogiques

- La formation alterne entre présentations des concepts théoriques et mises en application à travers d'ateliers et exercices pratiques (hors formation de type séminaire).
- Les participants bénéficient des retours d'expérience terrains du formateur ou de la formatrice
- Un support de cours numérique est fourni aux stagiaires

Modalités d'évaluation

- **En amont de la session de formation**, un questionnaire d'auto-positionnement est remis aux participants, afin qu'ils situent leurs connaissances et compétences déjà acquises par rapport au thème de la formation.

- **En cours de formation**, l'évaluation se fait sous forme d'ateliers, exercices et travaux pratiques de validation, de retour d'observation et/ou de partage d'expérience, en cohérence avec les objectifs pédagogiques visés.

- **En fin de session**, le formateur évalue les compétences et connaissances acquises par les apprenants grâce à un questionnaire reprenant les mêmes éléments que l'auto-positionnement, permettant ainsi une analyse détaillée de leur progression.