Maîtriser la clean architecture avec React.js

Date et durée

Code formation : DEV002-FR
Durée : 2 jours
Nombre d'heures : 7 heures

Description

La clean architecture est une approche révolutionnaire qui vous permet de **concevoir des applications indépendantes des technologies**. En plaçant les règles métier au cœur de votre architecture, vous gagnez en flexibilité, en testabilité et en maintenabilité.
Cette formation intensive de 2 jours vous plongera dans **l'univers de React.js, de la Clean Architecture et du TDD**. Vous apprendrez à structurer vos applications react, à écrire des tests unitaires et d'intégration, puis à **modéliser votre domaine métier grâce au Domain-Driven Design**. En combinant ces approches, vous serez en mesure de créer des applications adaptées aux besoins de votre entreprise.
Grâce à des démonstrations de live coding et des **exercices pratiques en live**, vous acquerrez les compétences nécessaires pour concevoir et développer des applications performantes et évolutives. Vous serez en mesure de **prendre des décisions architecturales éclairées** et de mener à bien des projets complexes.
Important : *pour garantir une expérience d'apprentissage optimale, cette formation est limitée à un groupe de 4 à 10 participants.*

Objectifs

A l'issue de cette formation clean architecture, vous atteindrez les objectifs suivants :

- maîtriser parfaitement le concept de la clean architecture et l'inversion des dépendances ;
- développer des applications de haute qualité en appliquant la méthode du développement piloté par les tests (Test-Driven Development - TDD) ;
- concevoir des structures de dossiers organisées et scalables pour tout type de projet ;
- renforcer la fiabilité de votre code grâce à une couverture de tests complète pour garantir un haut niveau de confiance et effectuer des déploiements régulièrement ;
- mettre en œuvre différents types de tests pour garantir la qualité à tous les niveaux ;
- améliorer continuellement votre code grâce à des refactorisations sécurisées.

Points forts

Un formateur expert et consultant en ingénierie logicielle, des démonstrations de live coding, des exercices pratiques en live, des exemples de clean architecture avec react.js tirés de projets réels et un code source disponible sur GitHub.

Modalités d'évaluation

Travaux Pratiques

Pré-requis

Suivre cette formation nécessite les prérequis suivants :

- **Matériel :** un ordinateur équipé d'une caméra, d'un micro et d'une connexion internet stable.
- **Logiciels :**
  - la dernière version stable de Node.js ;
  - la dernière version stable de Docker ;
  - Un environnement de développement tel que Visual Studio Code ou WebStorm.
- **Compétences :**
  - de solides connaissances en JavaScript et TypeScript (compréhension des types, interfaces, etc.) ;
  - une expérience dans l'écriture de tests unitaires ;
  - des notions de base sur les concepts de programmation orientée objet.

Info Stack Technique : React (voir NextJS), TypeScript et Vitest.

Public

Cette formation s'adresse aux publics suivants :

- tous professionnels ayant au moins 2 ans d'expérience en développement back-end comme :
  - les développeurs back-end souhaitant approfondir leurs connaissances en architecture logicielle et passer au niveau supérieur ;
  - les tech leads désireux d'optimiser la conception et la maintenance des applications de leur équipe ;
  - les architectes logiciels en quête de nouvelles approches pour concevoir des systèmes plus robustes et évolutifs.

Programme

Jour 1 : introduction au TDD et aux fondamentaux de la clean architecture

- Accueil et tour de table :
  - vos attentes et présentation du programme prévu.
- TDD en action :
  - découvrez comment le développement piloté par les tests vous guide naturellement vers une architecture propre et maintenable (démonstration avec un petit projet concret).
- Les fondamentaux de la clean architecture :
  - plongez dans les principes de l'architecture de code propre et comprenez comment ils s'articulent avec le TDD.
- Démonstration d'un projet pilote :
  - découpage collaboratif des des Use Cases (cas d'utilisation) ;
  - application de la conception pilotée par le domaine (Domain-Driven Design) ;
  - isolation et ajustement des fonctionnalités de base ;
  - développement piloté par les tests depuis l'extérieur.

Jour 2 : Approfondissement et montée en compétences

- Introduction à la refactorisation :
  - pourquoi refactoriser ?
  - les différents types de refactorisation ;
  - les outils et les bonnes pratiques.
- Intégration d'API externes :
  - les concepts de base des appels d'API ;
  - le choix de la bonne API.
- Introduction aux mocks :
  - qu'est-ce qu'un mock ?

    - à quoi servent les mocks ?
    - présentation de Mock Service Worker (MSW).
- Utilisation avancée des mocks avec MSW :
    - créer des mocks réalistes ;
    - isoler les tests ;
    - optimiser les tests.
- Discussion sur les trades off :
    - les avantages et les inconvénients des mocks ;
    - les autres techniques de test ;
    - quand utiliser quelle technique ?