WealCome
Expertise en conception logicielle
BDD / TDD / DDD / Clean Architecture

PROGRAMME DE FORMATION

Document généré le 08/11/2026

TDD et Clean Architecture dans le monde React/Typescript/State Manager

Type d'action : **Action de formation** Durée totale : **14h**

Informations de session

Lieu : Visio par Zoom

Formateur : Michaël AZERHAD - Président de S.A.S.U WealCome et architecte / expert technique depuis plus de 20 ans

Dates et horaires

Date	Début	Fin	Durée
21/12/2026	09:15	12:15	3h
21/12/2026	13:00	17:00	4h
22/12/2026	09:15	12:15	3h
22/12/2026	13:00	17:00	4h

Description

Les projets logiciels — backend comme frontend — souffrent souvent d'une complexité accidentelle qui apparaît quelques mois après leur démarrage.

Cette complexité non anticipée entraîne :

- un ralentissement massif du développement
- une peur de modifier le code existant
- des corrections par workarounds
- une perte de testabilité
- une perte d'intention métier dans le code
- une architecture rigide et difficile à faire évoluer

Cette formation montre comment prévenir cette dérive grâce à deux pratiques essentielles :

- Test-Driven Development (TDD)
- Clean / Hexagonal Architecture

Réalisation complète d'un FrontEnd React réaliste, en live coding avancé, basé sur un cas métier digne de ce que l'on rencontre en entreprise — loin des katas simplistes.

Le groupe peut choisir le taux d'utilisation des technologies IA comme Claude Code : 0% - 20% - 50% - 100%

Technologies :

- React 19+ / Vite.js
- Typescript, version courante
- Zustand ou Jotai (avec React-Query) si Redux-RTK n'est pas choisi, voire les deux mondes !
- Vitest

Ce sera interactif avec des exercices sur le chemin, des échanges de questions/réponses au tac au tac, et surtout une bonne ambiance, à la fois pro et détendue.

Objectifs de la formation

- Expliquer les principes fondamentaux du Test-Driven Development (TDD) et de la Clean / Hexagonal Architecture dans un environnement React
- Identifier les causes principales de la complexité accidentelle dans un projet logiciel et mettre en œuvre des pratiques permettant de la prévenir.
- Concevoir et développer un FrontEnd React from scratch en suivant une méthodologie professionnelle basée sur le TDD Outside-In.
- Structurer une application en Clean Architecture avec séparation claire entre cas d'usage, modèles métier, générateur de View Models, et adaptateurs.
- Comprendre l'intérêt d'un state manager tel que Redux ou équivalent
- Mettre en place une stratégie de tests complète combinant tests unitaires, intégration (tests d'adaptateurs secondaires) et end-to-end in-memory (tests de composants React).
- Analyser et comparer les différents types de tests afin de comprendre leur rôle dans la conception logicielle et la robustesse du système.
- Refactorer du code existant pour améliorer sa lisibilité, sa testabilité et son expressivité métier.
- Remettre en question les croyances courantes liées au TDD et la Clean Architecture à travers des exercices et échanges interactifs.
- Être capable de transformer un FrontEnd legacy en architecture hexagonale testable.
- Utiliser un assistant IA (Claude Code) pour produire du code conforme aux principes du Software Craftsmanship (TDD, Clean Architecture) et vérifier sa qualité par les tests.

Prérequis

- Bonne maîtrise de Javascript et/ou Typescript
- Bonne maîtrise d'un framework orienté composants comme React, Angular ou Vue.js
- Bonnes connaissances en OOP et/ou Functional Programming
- Notions de state management recommandées
- Capacité à écrire un simple test unitaire avec Jest ou Mocha

Public visé

- Technical Leader
- Développeur Frontend
- Développeur Full Stack
- Architecte technique

Méthodes pédagogiques

- Des apports théoriques sur le processus
- Des exemples concrets
- Des démonstrations complètes par le formateur en live coding
- Exercices réalisés en live par les participants afin de s'exercer sur cette application d'entreprise.
- Challenges proposés quant au TDD, Clean Archi et au refactoring de code (modification de structure du code)

Capacités développées

- Concevoir un use case métier en Typescript en appliquant une démarche TDD Outside-In sans adhérence aux composants React.
- Écrire des tests exprimant clairement des règles de gestion métier.
- Écrire des tests dirigeant la génération de View Models.
- Utiliser le refactoring avec le mindset "Mikado" pour éviter de se retrouver en "chantier" avec une application temporairement non fonctionnelle.
- Identifier les causes de la complexité accidentelle et appliquer des pratiques pour la prévenir.
- Structurer une application en Clean / Hexagonal Architecture avec séparation claire entre domaine, cas d'usage, View Models et adaptateurs.
- Développer un FrontEnd React réaliste avec complètement dirigé par les tests.
- Mettre en place une stratégie de tests combinant tests unitaires, tests d'intégration (adaptateurs secondaires), et end-to-end in-memory (depuis les composants React).
- Patterns de refactoring de tests de sorte à améliorer la lisibilité, la concision et donc la productivité.
- Analyser la qualité d'une architecture existante, code possiblement legacy, et proposer des améliorations concrètes.
- Argumenter des choix de conception logicielle auprès d'une équipe technique.
- Identifier et remettre en question les fausses croyances courantes liées au TDD et à la Clean Architecture.

Déroulé de la formation

Jour 1

- Tour de table initial pour partager les attentes, objectifs personnels et niveaux des participants
- Questions ciblées sur le TDD et la Clean Architecture afin d'ajuster la direction pédagogique au niveau réel du groupe
- Présentation du projet fil rouge : FrontEnd React / Typescript réaliste inspiré d'un cas métier rencontré en entreprise ambitieuse (pas de kata simpliste)
- Identification des règles de gestion, scénarios principaux et invariants métier
- Introduction au NoEstimates et à l'importance de travailler en itérations très courtes (« minimum learnable »)
- Explication de la bonne utilisation des mocks et distinction entre solitary unit tests et sociable unit tests afin de comprendre leur rôle dans la conception logicielle et la testabilité
- Illustration en live coding de l'impact des choix de tests sur l'émergence du design et la qualité de l'architecture
- Démonstration de Mutation Testing avec Stryker (pour JS) afin d'assurer la pertinence du TDD
- Intégration d'un State Manager comme Redux RTK pour comprendre son intérêt par rapport à une solution sans ce dernier
- Mise en évidence de l'émergence naturelle de la Clean / Hexagonal Architecture par la pratique rigoureuse du TDD et l'inversion de dépendances
- Approfondissement du projet avec des cas métier plus complexes et refactoring d'architectures naïves ou legacy vers des architectures robustes et évolutives
- Démonstration de Claude Code avec agents et skills conçus par Michaël AZERHAD afin d'utiliser l'IA tout en respectant les principes du Software Craftsmanship
- Phases de mob programming, questions ouvertes et revue de code pour ancrer les apprentissages
- Évaluation des acquis par échanges, revue de code et auto-évaluation des capacités développées

Jour 2

- Récapitulatif du travail accompli la veille pour garantir la compréhension de tous les stagiaires
- Démonstration concrète de l'inversion de dépendances avec React ou State Manager et intégration dans une architecture hexagonale
- Intégration d'une API tierce dirigée par une suite de tests d'intégration
- Continuité des exercices avec refactoring de code legacy simulé ou évolution avec du nouveau code, au choix
- Démonstration de Claude Code avec agents et skills conçus par Michaël AZERHAD afin d'utiliser l'IA pour l'infrastructure et les tests end-to-end, d'intégration et de journey tout en respectant les principes du Software Craftsmanship
- Phases de mob programming, questions ouvertes et revue de code pour ancrer les apprentissages
- Évaluation des acquis par échanges, revue de code et auto-évaluation des capacités développées
- Mise en place du canal Slack privé WealCome pour assurer un suivi post-formation de chaque stagiaire et accompagner la mise en pratique sur des projets réels

Modalité d'accès

Contactez-nous par mail : contact@wealcomecompany.com

Délais d'accès

Nous pouvons programmer les formations en fonction de vos contraintes et de nos disponibilités.

Moyens d'accompagnement et d'assistance (Formation à distance)

En cas de difficulté d'accès à la plateforme de visioconférence ou pour toute question technique ou pédagogique pendant la session, le stagiaire dispose d'un point de contact unique pour obtenir une assistance immédiate :

Contact d'assistance technique et pédagogique : WealCome

E-mail : contact@wealcomecompany.com

Accessibilité handicap

Nos formations sont accessibles aux personnes en situation de handicap.

Les aspects, l'accessibilité et le type de handicap au regard des modalités pédagogiques sont à évoquer impérativement au cours de l'entretien préalable à toute contractualisation afin de pouvoir orienter ou accompagner au mieux les personnes en situation de handicap.