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

PROGRAMME DE FORMATION

Document généré le 27/01/2026

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

Dates et horaires

Date	Début	Fin	Durée
18/02/2026	09:15	12:15	3h
18/02/2026	13:00	17:00	4h
19/02/2026	09:15	12:15	3h
19/02/2026	13:00	17:00	4h

Objectifs de la formation

- Sensibiliser les participants à la discipline du Test-Driven Development (TDD) et de la Clean Architecture dans le monde Web afin de prévenir la complexité accidentelle.
- Développer des compétences en TDD et en Clean Architecture dans l'écosystème React/TypeScript (et un state manager) à travers des sessions de live coding professionnelles et approfondies.
- Réaliser une application Front from scratch, conforme aux standards d'une entreprise.
- Mettre en œuvre et comparer les différents types de tests (unitaires, d'acceptation, end-to-end, d'intégration) pour garantir la qualité du code.
- Clarifier et démontrer les concepts clés et annexes à travers des séances interactives de questions-réponses et des démonstrations concrètes.

Description

Les frontends comme les backends souffrent d'une complexité accidentelle dans la plupart des projets. Une complexité accidentelle est une complexité additionnelle malvenue qui aurait pu être évitée. Quand s'observe t'elle réellement ? Quelques mois après le début du projet. Pourquoi donc le terme "accidentelle" et non le terme "additionnelle" ? Car bien souvent, elle survient par surprise, sans anticipation aucune. Surtout lorsque le développeur n'est pas aguerri sur les concepts théoriques relatifs à la conception logiciel. Quelles en sont ses conséquences :

- Un temps de développement considérablement augmenté pour la moindre fonctionnalité normalement simple.
- Une peur extrême de changer le code existant, de peur d'y engendrer des impacts non perçus ni maîtrisés.
- Par cette peur de changer/casser, chaque bug remonté est corrigé avec des workarounds ; autrement dit des pansements indirects proches de la supercherie.
- Une expressivité du code réduite à néant, par à-coups de ces workarounds et du stress engendré par la soumission face au code existant.
- Un code qui devient de moins en moins testable, par des prises de raccourcis contraires aux bonnes pratiques d'architectures et de code design souvent passées inaperçues, sans prise de conscience.
- Des sessions fatiguantes de débugging à outrance devant ce code jugé farfelu, non expressif et souvent bancal.
- Une application devenue très rapidement inflexible, dont même la moindre évolution technologique telle une mise à jour de frameworks devient le signe d'un besoin de refonte globale ...

Est-ce une fatalité ? Devant le nombre de projets dans ce cas-là, il faut croire que oui. Mais il existe heureusement des pratiques logicielles qui redonnent de l'espoir et qui atténuent drastiquement cette complexité accidentelle si bien comprises et bien menées. Parmi ces pratiques, je mettrai l'accent sur deux d'entre-elles qui me paraissent totalement cruciales et qui ont changé mon quotidien dès 2011 y compris dans le monde du Front :

- Le Test-Driven Development alias TDD
- La Clean Architecture (cf Hexagonal Architecture ou Ports/Adapters architecture).

Dans cette formation, je vous démontrerai comment réaliser un frontend from scratch avec ces pratiques dans un live coding avancé et fluide, sans manquer de vous exposer une bonne stratégie de tests combinant tests d'acceptation, tests unitaires, tests d'intégration et tests de composants orientés end-to-end_inmemory. Mon premier souhait : que vous compreniez bien que le TDD n'est pas une technique de test mais de codage permettant une amélioration notable de vos algorithmes et de vos designs. Stop les console.log intempestifs et les lancements de browsers fréquents, viens dans un monde où tu ne débuggueras quasiment plus jamais ! Le sujet sera un sujet React/Typescript/Redux (ou Zustand, ou encore Jotaï, au choix) digne de ce qu'on attend de nous en entreprise, bien loin d'un vulgaire Kata de tri de nombres. Technologies :

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

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. Ayant l'habitude d'enseigner sur les sujets Craft, j'ai acquis une pédagogie qui vous plaira et qui ne laissera personne sur le carreau. Aussi, j'assurerai un suivi sous forme de réponses à vos nouvelles questions post-formation de sorte à ce que chacun d'entre-vous évoluent et progressent sans blocage et dans la bonne

direction au quotidien. Pour finir, vous constaterez pour beaucoup que quasiment tout ce que vous pensiez au sujet de ces pratiques sont en réalité de fausses croyances … Je n'ai plus qu'à vous souhaiter la bienvenue dans ce noble monde du développement logiciel professionnel. Les sessions regroupent entre 3 et 11 personnes, afin de garder une haute qualité d'intéraction.

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)

Déroulé de la formation

Jour 1

- Rapide tour de table, présentation de chacun et exposition des attentes
- Introduction et cours théorique sur le TDD cassant les énormes quiproquos à son sujet
- Si le groupe est vraiment novice en TDD, Kata éventuel et judicieusement choisi de mises en pratique avec Node.js, Typescript et Jest
- Introduction et cours théorique sur la Clean Architecture
- Démarrage d'écriture d'une application "from scratch" digne d'un cas réel d'entreprise en TDD (User Story avec plusieurs règles de gestion) tout en respectant la Clean Architecture dans le monde Web avec React/Typescript/State Manager.
- Approche agile avec le mindset NoEstimates initiée par un mini atelier BDD judicieusement mené.
- Si option Redux choisie, introduction à celui-ci avec cassure des fausses croyances à son sujet
- Séances de questions / réponses tout au long

Jour 2

- Suite du live coding de l'application "from scratch" digne d'un cas réel d'entreprise
- Séances de refactoring au fil de l'eau du code ET des tests de l'application exemple, rendues plaisantes et sans crainte grâce au TDD
- Clarification de concepts subtils relatifs au TDD et à la Clean Architecture
- Démonstration de la pratique efficace de TDD sur des composants React sous la forme de test "end-to-end InMemory"
- Compréhension du concept de "test d'intégration"
- Séance de questions / réponses tout au long

WealCome

192 avenue de la Division Leclerc, 95160 Montmorency, France