DAWAN

Plan de la formation

www.dawan.fr

Formation MERN Stack : MongoDB, Express.js, React et Node.js

Durée :	10 jours (70 heures)
Tarif inter-entreprises :	5 475,00 € HT (Présentiel) 4 380,00 € HT (Distanciel)
Public :	Développeurs Javascript
Pré-requis :	Bonnes connaissances de Node.js
Objectifs :	Construire une application web fullstack avec la MERN Stack : MongoDB, Express.js, React et Node.js
Modalités pédagogiques, techniques et d'encadrement :	• Formation synchrone en présentiel et distanciel. • Méthodologie basée sur l'Active Learning : 75 % de pratique minimum. • Un PC par participant en présentiel, possibilité de mettre à disposition en bureau à distance un PC et l'environnement adéquat. • Un formateur expert.
Modalité d'évaluation :	• Définition des besoins et attentes des apprenants en amont de la formation. • Auto-positionnement à l'entrée et la sortie de la formation. • Suivi continu par les formateurs durant les ateliers pratiques. • Évaluation à chaud de l'adéquation au besoin professionnel des apprenants le dernier jour de formation.
Sanction :	Attestation de fin de formation mentionnant le résultat des acquis
Référence :	SIT102244-F
Note de satisfaction des participants :	4,93 / 5
Contacts :	commercial@dawan.fr - 09 72 37 73 73

Modalités d'accès :	Possibilité de faire un devis en ligne (www.dawan.fr, moncompteformation.gouv.fr, maformation.fr, etc.) ou en appelant au standard.
Délais d'accès :	Variable selon le type de financement.
Accessibilité :	Si vous êtes en situation de handicap, nous sommes en mesure de vous accueillir, n'hésitez pas à nous contacter à referenthandicap@dawan.fr, nous étudierons ensemble vos besoins

Introduction

Panorama des librairies disponibles
Rappels HTML5 et JavaScript
React.js : présentation, historique des versions
Installer l'environnement de développement (NodeJS/npm/yarn)
Présentation de la commande create-react-app

Atelier : Créer et démarrer un projet ReactJS

Découvrir ECMAScript

Spécification ECMAScript et implémentations
La Syntaxe Objet :
Evolution de la programmation orientée prototype
Syntaxe de classes : membres d'instances et membres de classe
Agrégation d'objets (mixins)
Héritage
Les fonctions fléchées
Design patterns
Définition de modules
Système natif de gestion des modules, chargement
Gestion des dépendances

Créer un projet React

Organisation d'un projet React
Architecture du projet
Personnaliser la configuration du projet
Comprendre le rendu d'un projet
Le Mode Strict

Concevoir des Composants

Définir un composant
Les composants fonction et classe
Utiliser les propriétés
Etats des composants
Le style (CSS/SASS) et les assets
Gestion des évènements
Relation entre composants
Utiliser la composition

Atelier : Créer plusieurs composants

Mettre en forme du contenu avec JSX

Présentation de JSX
Les éléments React avec et sans JSX
Le DOM virtuel
Les expressions et les attributs
Se protéger des attaques par injection (XSS)
Les conditions et les boucles
Les fragments

Atelier : Afficher, boucler et conditionner des jeux de données

Comprendre le cycle de vie d'un composant

Comprendre les cycles
Monter un composant
Ajouter des méthodes à un composant et des propriétés par défaut
Fixer des types de propriété

Atelier : Charger des données depuis une API

Utiliser le cycle de vie dans un composant fonctionnel: Les Hooks

Présentation des Hooks
Liste des hooks disponibles
Les hooks d'états et d'effet
Les bonnes pratiques et pièges à éviter
Les hooks personnalisés

Atelier : Créer et manipuler une todolist

Naviguer dans un projet React

Mise en place du Routing (React-router)
Rendu d’une route
Liens via les Router
Organiser les routes
Passage de paramètres liens, redirections

Atelier : Mettre en place la navigation

Communiquer depuis une application: Les Formulaires

Rappel des comportements en HTML et JavaScript
Les formulaires avec React
Les composants contrôlés et non contrôlés
Récupérer et mettre à jour des données
Valider et filtrer les données
Présentation des frameworks disponibles (React Hook Form, Formik)

Atelier : Créer un formulaire d'inscription

Utiliser Redux dans un projet

Présentation de redux
Les avantages apportés par les Store
Créer et configurer un Store
Les Actions et Les Reducers
Utiliser les Action Creators
La fonction connect()
Les Hooks Redux (useStore, useSelector, useDispatch)

Atelier : Mettre en place une authentification

Comprendre l'architecture

Architecture en couches : du monolythe au microservices
Contraintes d'architecture des microservices
Gestion de l'authentification centralisée dans une architecture microservices
Intérêt d'une passerelle d'API
Gestion centralisée des traces

Développement d'API REST avec Express.js

Express.js : principe, fonctionnalités, pré-requis
Configuration du projet : packages et modules additionnels : Morgan, Helmet, Cors,...
Configuration de morgan pour la gestion des logs de requêtes HTTP
Organisation des couches du projet : routes, services, repositories
Intérêt d'une couche DTO, utilisation d'un mapper d'objets : automapper ou autre
Implémentation de contrôleurs REST : gestion des routes, traitement asynchrone
Gestion des paramètres de méthodes et du mapping
Gestion du download
Gestion de l'upload, configuration
Gestion des services et des transactions associées
Gestion du cross origin avec Cors et restriction des domaines appelants
Test de l'api REST avec Postman
Ecriture de tâches asynchrones, planification
Cache web

Atelier : Écriture de micro-services avec Express.js - Test des méthodes de services avec Postman ou autre

Documenter une API REST

Open API Specification (Swagger) : présentation, outil
Utilisation de express-openapi
Visualisation avec Swagger Editor
Documentation du code Java, génération de javadoc

Atelier : Documentation de l'api

Intercepter des requêtes et gérer les erreurs

Gestion globale des exceptions
Capture d'exceptions personnalisées
Intercepteurs de requêtes/réponses

Atelier : Gestion des exceptions et implémentation d'intercepteurs

Appeler d'autres API REST (écriture de clients)

http_request : méthodes et paramètres
Ecriture de requêtes GET, POST, PUT, DELETE
Gestion des paramètres et du corps de la requête

Gestion des headers
Gestion des réponses et utilisation de mappers

Atelier : Implémentation de clients pour un service REST

Sécuriser l'API REST

Configuration de Helmet
Gestion des données d’entête
Gestion des utilisateurs et des rôles : mise en place de JWT (passport-jwt)
Hachage des mots de passe avec bcryptjs

Atelier : Intégration de Helmet, jsonwebtoken, dotenv et bcryptjs

Tester une application Express.js avec Jest et SuperTest

Stratégies de tests, types supportés
Installation de jest, supertest et cross-env, configuration de l'application
Mocking des couches de l'application
Tests auto-configurés
Exécution et reporting

Atelier : implémentation et exécution de tests avec Jest et supertest

Configurer un projet Mongoose

Mongoose : Présentation, fonctionnalités
MongoDB : Installation, structure d'un schéma (collections, documents)
Configuration d'un projet Node.js utilisant Mongoose : pool de connexion, contraintes
Propriétés par défaut et paramétrage
Gestion des options de logging

Atelier : Intégration de Mongoose dans un projet Node.js, configuration de la Bdd NoSQL MongoDB et des traces

Construire un modèle et écrire des requêtes

Modèle de base
Documents et sous-dodcuments
Gestion des types
Requêter des documents
Mise à jour et suppression de documents

Gestion des validations
Mongoose migrations (ts-migrate-mongoose)

Atelier : Réalisation d'un schéma global de BDD MongoDB, gestion des migrations et écriture de requêtes

Maîtriser des concepts avancées

Cache : fonctionnement, mise en cache de requêtes (mongoose-cache)
Hooks
Mise en place d'une solution d'audit de tables (historique de modifications)

Atelier : Implémentation d'une couche complète de persistance - mise en place d'un cache