

Mis à jour le 24/02/2026

S'inscrire

Formation NestJS

3 jours (21 heures)

Présentation

NestJS est un framework Node.JS qui permet de créer principalement des APIs efficaces et évolutives. La popularité de Nest.JS a énormément augmenté grâce à ses fonctionnalités d'excellences. En 2021, NestJS a explosé en popularité avec plus de 40 000 stars sur Github. Il est facile à utiliser et rapide à apprendre.

Ce framework utilise JavaScript et peut également utiliser TypeScript. Il combine des éléments de POO (programmation orientée objet), de PF (programmation fonctionnelle) et de PRF (programmation fonctionnelle réactive).

Ce dernier vous aide aussi à progresser en structurant correctement votre application. NestJS regroupe un ensemble de technologies et fonctionnalités nécessaires pour construire des serveurs HTTP fiables et durables utilisant Nodejs.

Nest.JS implémente le framework Express par défaut mais permet également l'utilisation de Fastify. Actuellement ce framework est en pleine croissance sur TypeScript dans l'univers Node.JS. Grâce à NestJS vous pourrez écrire des applications évolutives, testables et faiblement couplées.

Pour commencer un nouveau projet sur [Node.JS](#), NestJS est un excellent choix car basé sur une architecture modulaire. Cela permettant de définir contrôleur, services, middleware, pipes ou encore guards au sein de ceux-ci à l'image d'Angular duquel il s'inspire.

Comme toutes nos formations, celle-ci vous présentera **la dernière version stable** de la technologie et ses nouveautés.

Objectifs

- Apprenez à maîtriser la plateforme NestJS
- Savoir créer des applications sur NestJS
- Créer une architecture progressive pour de larges applications

Public visé

Développeurs web

Prérequis

- Maîtrise du langage JavaScript
- Connaissance en Node.JS
- Connaissance des bases de TypeScript sera en plus

Prérequis techniques

- NodeJS version 16+ installé
- Docker et Git installés
- **Visual Studio Code** ou un autre éditeur de texte

Programme de notre formation NestJS

[Jour 1 - Matin]

Fondamentaux et Révolution NestJS 12

- Comprendre la philosophie NestJS : modularité et injection de dépendances (DI)
- Découvrir les nouveautés de la v12 : moteur de build SWC et support ESM natif
- Installation et prise en main de la CLI NestJS
- Anatomie d'un projet : Modules, Controllers et Providers
- Cycle de vie d'une application NestJS (Request Lifecycle)
- Atelier pratique : Initialisation d'un projet et création d'une API modulaire.

[Jour 1 - Après-midi]

Maîtrise de la logique métier et Validation

- Utilisation des Middlewares et des Interceptors
- Validation de données avec Pipes et class-validator
- Gestion globale des erreurs via les Exception Filters
- Configuration dynamique avec @nestjs/config et validation de schéma (Zod/Joi)
- Optimisation des performances avec Fastify vs Express

- Atelier pratique : Mise en place d'un système de logging et de validation robuste.

[Jour 2 - Matin]

Persistence des données et Typage avancé

- Intégration d'ORMs modernes : Prisma ou TypeORM
- Patterns d'architecture : Repository et Data Mapper
- Gestion des migrations et du seeding de base de données
- Manipulation des relations complexes et gestion des transactions
- Optimisation des accès aux données (Lazy loading vs Eager loading)
- Atelier pratique : Connexion à PostgreSQL et modélisation d'une base relationnelle.

[Jour 2 - Après-midi]

Sécurité et Authentification (Modern Stack)

- Mise en œuvre de Guards et stratégies Passport.js
- Authentification JWT (JSON Web Tokens) et Refresh Tokens
- Gestion des autorisations : RBAC (Rôles) et ABAC (Attributs)
- Protection contre les vulnérabilités (CORS, Helmet, Rate-limiting)
- Sécurisation des secrets avec les variables d'environnement
- Atelier pratique : Implémentation d'un système d'Auth complet et sécurisé.

[Jour 3 - Matin]

Architectures Distribuées & Microservices

- Introduction aux Microservices avec NestJS
- Protocoles de transport : Redis, RabbitMQ ou gRPC
- Communication asynchrone et patterns d'événements
- Gestion des files d'attente avec BullMQ
- Architecture hybride : API Gateway et microservices
- Atelier pratique : Création d'un service de traitement asynchrone découplé.

[Jour 3 - Après-midi]

Industrialisation, Tests et Déploiement

- Tests unitaires et d'intégration avec Jest
- Tests de bout en bout (e2e) avec Supertest
- Observabilité : Monitoring avec Terminus et OpenTelemetry
- Conteneurisation avec Docker (optimisation multi-stage)
- Stratégies de déploiement CI/CD et checklist de production

- Atelier pratique : Projet final intégrant tests, Docker et monitoring.

Sociétés concernées

Cette formation s'adresse à la fois aux particuliers ainsi qu'aux entreprises, petites ou grandes, souhaitant former ses équipes à une nouvelle technologie informatique avancée ou bien à acquérir des connaissances métiers spécifiques ou des méthodes modernes.

Positionnement à l'entrée en formation

Le positionnement à l'entrée en formation respecte les critères qualité Qualiopi. Dès son inscription définitive, l'apprenant reçoit un questionnaire d'auto-évaluation nous permettant d'apprécier son niveau estimé sur différents types de technologies, ses attentes et objectifs personnels quant à la formation à venir, dans les limites imposées par le format sélectionné. Ce questionnaire nous permet également d'anticiper certaines difficultés de connexion ou de sécurité interne en entreprise (intraentreprise ou classe virtuelle) qui pourraient être problématiques pour le suivi et le bon déroulement de la session de formation.

Méthodes pédagogiques

Stage Pratique : 60% Pratique, 40% Théorie. Support de la formation distribué au format numérique à tous les participants.

Organisation

Le cours alterne les apports théoriques du formateur soutenus par des exemples et des séances de réflexions, et de travail en groupe.

Validation

À la fin de la session, un questionnaire à choix multiples permet de vérifier l'acquisition correcte des compétences.

Sanction

Une attestation sera remise à chaque stagiaire qui aura suivi la totalité de la formation.