DÉVELOPPEMENT FULLSTACK

# Fullstack React + Symfony : application web complète

Assemblez React et Symfony en une application découplée : API REST avec API Platform, authentification JWT de bout en bout, front React relié à l'API, tests et déploiement sur un serveur avec Docker.

| DURÉE | TARIF | NIVEAU | LANGUE | GROUPE | FORMAT |
|---|---|---|---|---|---|
| **5j** (35h) | **2 070 € / participant** Inter-entreprise. Intra : sur devis, au plus 2 070 € / participant | **Avancé** | **Français** | **3-10** | **Formation** Présentiel ou à distance |

## 1 PUBLIC VISÉ

- Développeurs qui connaissent React et Symfony et veulent les assembler en une application complète
- Développeurs front-end ou back-end qui veulent élargir leur profil au fullstack

## 2 PRÉREQUIS

- Construire une application React avec des hooks, React Router et des appels d'API
- Développer une application Symfony avec Doctrine, des formulaires et le composant Security
- Aucune connaissance d'API Platform n'est requise

## 3 OBJECTIFS PÉDAGOGIQUES

- Mettre en place une application découplée : projet Symfony avec API Platform, projet React avec Vite, CORS et variables d'environnement
- Exposer des ressources avec API Platform : opérations, groupes de sérialisation, relations et documentation OpenAPI
- Ajouter à l'API filtres, pagination et validation, avec des erreurs au format standard (RFC 9457)
- Sécuriser l'API par jetons JWT (LexikJWTAuthenticationBundle), avec rafraîchissement et contrôle d'accès par voter
- Gérer l'authentification dans React : stockage du jeton, contexte, routes protégées et affichage selon le rôle
- Développer le front React qui consomme l'API : couche service, hooks, pagination, filtres et CRUD avec mise à jour optimiste
- Envoyer des fichiers de React vers l'API et afficher dans les formulaires React les erreurs de validation renvoyées par Symfony
- Tester l'API avec ApiTestCase et le front avec Vitest, Testing Library et Mock Service Worker
- Déployer l'application sur un serveur avec Docker Compose, Nginx et HTTPS

## 4 PROGRAMME DÉTAILLÉ

**Jour 1 (7h) : Architecture et API Platform**

**Module 1 : Architecture découplée et environnement (3h30)**

Concevoir l'architecture (1h30)

- Application découplée ou monolithe : avantages et compromis

- Un seul dépôt ou deux ; le flux entre le front et l'API : HTTP, JSON, CORS
- Les outils : un client HTTP (Bruno ou Postman), le Profiler de Symfony, React DevTools
- Exercice d'entraînement : schéma d'architecture du projet fil rouge

Mettre en place les deux projets (2h)

- Symfony 8 et API Platform 5 ; base PostgreSQL et variables d'environnement
- React avec Vite et son proxy de développement
- CORS avec NelmioCorsBundle
- TP évalué : dépôt unique avec les deux projets, où React affiche une réponse de l'API

**Module 2 : API Platform : ressources et sérialisation (3h30)**

Premières ressources (1h30)

- Ce qu'API Platform apporte : génération de l'API, documentation
- #[ApiResource] ; opérations GetCollection, Get, Post, Patch, Delete
- Swagger UI généré automatiquement
- Exercice d'entraînement : ressource Catégorie avec ses opérations

Sérialisation et relations (2h)

- Groupes de sérialisation de lecture et d'écriture
- Champs calculés
- Relations : IRI ou objet imbriqué
- TP évalué : ressource Article avec des groupes de lecture et d'écriture distincts et une relation vers Catégorie

**Jour 2 (7h) : Filtres, validation et authentification côté API**

**Module 3 : Filtres, pagination, validation et erreurs (3h30)**

Filtres et pagination (1h30)

- Filtres déclarés avec #[QueryParameter] : ExactFilter, PartialSearchFilter, ComparisonFilter, DateFilter, SortFilter
- La pagination d'API Platform
- Exercice d'entraînement : filtrer les articles par catégorie et par date

Validation et erreurs (2h)

- Contraintes de validation sur les ressources
- Erreurs 422 au format Problem Details (RFC 9457, qui remplace la RFC 7807)
- State providers et state processors : le principe
- TP évalué : endpoint de recherche d'articles avec filtres, tri et pagination, et validation qui renvoie des erreurs 422 au format standard

**Module 4 : Authentification JWT côté Symfony (3h30)**

LexikJWTAuthenticationBundle (1h30)

- Installation et clés de signature
- Pare-feu sans état pour l'API ; /api/login
- Structure d'un JWT : en-tête, contenu, signature
- Exercice d'entraînement : obtenir un jeton depuis le client HTTP

Protéger les ressources (2h)

- Règles de sécurité sur les opérations ; voters par objet

- Exposer l'utilisateur courant : /api/me
- Jetons de rafraîchissement ; jeton dans un cookie httpOnly
- TP évalué : CRUD des articles réservé aux utilisateurs connectés, modification réservée à l'auteur par un voter, /api/me et rafraîchissement du jeton

**Jour 3 (7h) : Le front React**

**Module 5 : Authentification côté React (3h30)**

Le jeton dans React (1h30)

- Stocker le jeton : localStorage ou cookie httpOnly, les risques de chacun
- Contexte d'authentification et hook useAuth
- Ajouter le jeton à chaque requête ; une erreur 401 ramène à la connexion
- Exercice d'entraînement : connexion et déconnexion

Routes protégées et rôles (2h)

- Route protégée qui redirige
- Rôles lus depuis /api/me ; affichage selon le rôle
- Synchroniser la connexion entre les onglets
- TP évalué : interface avec parties publiques et parties réservées aux administrateurs, connexion persistante et déconnexion

**Module 6 : Front React : services, état et CRUD (3h30)**

Couche service et état (1h30)

- Organisation des dossiers : pages, composants, hooks, services, contextes
- Une couche service pour les appels à l'API ; typer les réponses avec TypeScript
- VITE_API_URL ; état local ou Context
- Exercice d'entraînement : service des articles et hook useArticles

CRUD complet (2h)

- Liste paginée par l'API ; formulaires de création et d'édition
- Suppression confirmée avec mise à jour optimiste
- TP évalué : interface de gestion des articles : liste filtrée et paginée, création, édition, suppression optimiste

**Jour 4 (7h) : Fichiers, erreurs et tests**

**Module 7 : Envoi de fichiers et erreurs dans React (3h30)**

Envoyer des fichiers (1h30)

- VichUploaderBundle côté Symfony ; endpoint d'envoi dans API Platform
- Envoyer un fichier depuis React avec FormData ; aperçu avant l'envoi
- Exercice d'entraînement : envoi d'un avatar

Erreurs et expérience utilisateur (2h)

- Les erreurs de validation de l'API placées sous les champs du formulaire
- Notifications de succès et d'erreur ; error boundaries ; page 404
- Squelettes de chargement
- TP évalué : image de couverture envoyée depuis le formulaire d'article, avec les erreurs de validation de l'API affichées sous chaque champ

**Module 8 : Tests de l'API et du front (3h30)**

Tester l’API (1h30)

- ApiTestCase d’API Platform
- Authentifier un utilisateur de test avec un JWT
- Vérifier statuts, structure du JSON et droits
- Exercice d’entraînement : tester la liste des articles

Tester le front (2h)

- Vitest et Testing Library
- Simuler l’API avec Mock Service Worker
- Tester le parcours de connexion et une page protégée
- TP évalué : tests des endpoints principaux de l’API, puis du formulaire de connexion et d’une page protégée côté React

**Jour 5 (7h) : Déploiement et projet de synthèse**

**Module 9 : Déployer sur un serveur (3h)**

Préparer la production (1h30)

- Secrets de Symfony ; construction de production de React
- Nginx sert le front et relaie l’API
- HTTPS avec Let’s Encrypt
- Exercice d’entraînement : configuration Nginx du projet

Docker Compose en production (1h30)

- Dockerfile de Symfony (PHP-FPM) ; Dockerfile de React en multi-stage avec Nginx
- Compose : Symfony, React, PostgreSQL, Nginx ; migrations au démarrage
- TP évalué : déployer l’application complète sur le serveur d’exercice en HTTPS

**Module 10 : Projet de synthèse, évalué (4h)**

Réalisation (2h30)

- Projet de synthèse, réalisé individuellement : plateforme de contenu avec espace public et administration : ressources API Platform, JWT et rôles, CRUD React, envoi de fichiers, tests, déploiement

Revue et retour (1h30)

- Revue de code collective : architecture, sécurité, cohérence entre l’API et le front
- Retour individuel du formateur sur le projet rendu

*Le contenu peut être ajusté selon le niveau, les prérequis et les besoins des participants.*

## 5 COMPÉTENCES VISÉES

- Concevoir une application découplée et son API REST
- Sécuriser une application de bout en bout avec JWT
- Développer un front React relié à l’API
- Fiabiliser l’API et l’application : validation, erreurs, tests
- Déployer une application fullstack

## 6 MODALITÉS PÉDAGOGIQUES

Formation délivrée en présentiel ou à distance, en classe virtuelle. Le formateur alterne entre méthode démonstrative, méthode interrogative (analyse des choix d'architecture entre le front et l'API) et méthode active (exercices d'entraînement, TP évalués et projet de synthèse).

## 7 MOYENS PÉDAGOGIQUES ET TECHNIQUES

- Support de cours numérique mis à disposition des apprenants
- Dépôt GitHub de démonstration avec le projet fil rouge versionné par étape
- Environnement de développement : VS Code + PHP 8.4 ou plus récent + Symfony CLI + Node.js en version LTS + Docker
- Un serveur Ubuntu (VPS) fourni par le formateur pour le déploiement
- Pour la classe virtuelle : visioconférence (Microsoft Teams ou équivalent), partage d'écran, chat en direct
- Supports remis par e-mail et sur la plateforme pédagogique LaPolaris (ressources, émargement)

En formation intra-entreprise sur le site du client, celui-ci s'assure et s'engage à disposer de toutes les ressources matérielles nécessaires à son bon déroulement (salle, postes de travail, logiciels, accès à Internet), conformément aux prérequis indiqués dans le programme et aux moyens prévus dans la convention.

## 8 MODALITÉS D'ÉVALUATION

- Questionnaire de positionnement individuel renseigné par chaque participant avant le démarrage de la formation, analysé par le formateur et donnant lieu, le cas échéant, à un ajustement du déroulé
- Exercices d'entraînement réalisés par chaque participant sur son poste, corrigés en séance
- Évaluation des acquis par les TP évalués, réalisés individuellement sur poste : un par module, puis le projet de synthèse décrit dans le dernier module du programme ; chaque objectif est évalué dans au moins un TP
- Grille de suivi renseignée par le formateur pour chaque participant et pour chacun des objectifs de l'action, au regard des critères d'évaluation ci-dessous
- Questionnaire d'appréciation renseigné par les participants en clôture de formation, et enquête à froid dans un délai de trois à six mois

## 9 CRITÈRES D'ÉVALUATION

- Deux projets qui démarrent et communiquent ; CORS limité à l'origine du front ; adresse de l'API lue depuis l'environnement
- Ressources exposées avec les bonnes opérations ; champs exposés selon les groupes ; relations en IRI ou en objet selon le besoin ; documentation OpenAPI à jour
- Filtres, tri et pagination utilisables depuis l'URL ; données invalides refusées avec une erreur 422 au format standard
- Connexion qui renvoie un JWT ; ressources protégées ; voter qui réserve la modification à l'auteur ; jeton rafraîchi sans nouvelle connexion
- Jeton stocké selon un choix justifié ; routes protégées ; actions masquées selon le rôle ; erreur 401 qui ramène à la connexion
- Appels regroupés dans une couche service ; liste paginée et filtrée ; création, édition et suppression qui mettent l'interface à jour

- Fichier envoyé, validé par l'API et affiché ; erreurs de validation de l'API affichées sous les bons champs
- Tests de l'API (statuts, JSON, droits) et tests du front avec une API simulée qui passent
- Application en ligne en HTTPS sur le serveur ; Nginx sert le front et relaie l'API ; migrations appliquées ; secrets hors du dépôt

## 10 INDICATEURS DE RÉSULTATS

Les indicateurs de résultats de cette formation (taux de satisfaction et taux de réussite) seront publiés sur sa fiche en ligne à l'issue des premières sessions réalisées.

## 11 MODALITÉS DE VALIDATION

**Une attestation de fin de formation** est remise à chaque participant à l'issue du parcours. Elle mentionne **l'intitulé** et la **nature de l'action**, sa **durée** en heures, sa **période de réalisation**, **les objectifs de l'action** et le **niveau d'atteinte** de chacun d'eux, exprimé en trois valeurs : **atteint**, **partiellement atteint**, **non atteint**. **Une attestation d'assiduité**, établie à partir des émargements, est délivrée dans les mêmes conditions.

**Formation non certifiante** : elle ne délivre pas de certification enregistrée au RNCP ou au Répertoire spécifique.

## 12 SUIVI ET ACCOMPAGNEMENT

- Émargement électronique signé par chaque participant et par le formateur à chaque demi-journée, en présentiel comme à distance
- Traçabilité des activités pédagogiques réalisées
- Attestation d'assiduité délivrée en fin de formation
- Suivi individuel via les QCM de fin de journée, les exercices corrigés et les TP évalués

## 13 CONDITIONS D'ACCÈS

Formation accessible sans prérequis administratif particulier. Cette formation n'est pas éligible aux financements publics ou mutualisés, notamment au compte personnel de formation et à une prise en charge par un opérateur de compétences.

## 14 DÉLAIS D'ACCÈS

Inscription possible jusqu'à 15 jours ouvrés avant le début de la session pour permettre la préparation logistique et, si besoin, l'adaptation du contenu au contexte client. Pour toute demande spécifique, nous contacter directement.

ACCESSIBILITÉ · HANDICAP

Nos formations peuvent être accessibles aux personnes en situation de handicap. La question d'un besoin d'aménagement (matériel, temporel ou pédagogique) est posée lors de l'analyse du besoin et du questionnaire de positionnement ; vous pouvez aussi nous le signaler à tout moment, y compris pendant la formation, afin d'étudier les adaptations possibles. Référent handicap : contact@lapolaris.fr

**LaPolaris**

TÉL. +33762584798

EMAIL contact@lapolaris.fr

WEB lapolaris.fr