PLAN DÉTAILLÉ DE FORMATION FULL-STACK WEB (70 jours)

Objectif global

Devenir développeur full-stack capable de créer des applications web modernes, dynamiques et sécurisées, en maîtrisant le front-end (JavaScript, Tailwind, React.js) et le back-end (Django ou Spring Boot + API REST + BDD).

Phase 1 : JavaScript 7 jours

Jour	***Objectif***	***Contenu à apprendre***	***Projet/Exercice***
1	*Comprendre les bases du JS*	*Variables (**let**, **const**), types, opérateurs, console.log*	*Mini-calculatrice en console*
2	*Maîtriser les conditions & boucles*	***if**, **else**, **switch**, **for**, **while**, **break**, continue*	*Générateur de table de multiplication*
3	*Manipuler les fonctions*	*Fonctions classiques, anonymes, fléchées, portée (scope)*	*Convertisseur de température*
4	*Utiliser tableaux & objets*	*Méthodes de tableau (push, map, filter, reduce), objets*	*Liste de produits avec filtre*
5	*Manipuler le DOM*	*Sélecteurs (**getElementById**, **getElementByTagName** **querySelector**), événements (**click**, **input**)*	*Liste interactive (ajout/suppression)*
6	*Travailler avec les APIs*	***JSON**, **fetch**, **promises**, **then**, **async/await***	*App météo avec API OpenWeather*
7	*Mini-projet*	*Révision + intégration DOM & API*	*ToDo List dynamique avec persistance locale*

Phase 2 : Tailwind CSS 8 jours

Jour	***Objectif***	***Contenu à apprendre***	***Projet/Exercice***
1	*Introduction à Tailwind*	*Installation (CDN / Vite / React), philosophie utility-first*	*Structure de base*
2	*Positionnement*	*container, flex, grid, gap, justify, items*	*Galerie d'images responsive*
3	*Typographie & couleurs*	*text-, bg-, font-, tracking, leading*	*Page de profil stylisée*
4	*Espacements & bordures*	*padding, margin, border, rounded, shadow*	*Carte utilisateur personnalisée*
5	*Responsive design*	*Breakpoints (sm, md, lg, xl, 2xl)*	*Navbar adaptative*
6	*États interactifs*	*hover:, focus:, active:*	*Formulaire stylisé interactif*
7	*Animations & transitions*	*transition, duration, ease, transform*	*Boutons animés*
8	*Mini-projet*	*Intégration complète*	*Reproduire une landing page (Figma / maquette)*

Phase 3 : React.js 20 jours

Jour(s)	***Objectif***	***Contenu à apprendre***	***Projet/Exercice***
1-2	*Comprendre React*	*JSX, composants, create-react-app, props*	*Page de profil dynamique*
3-4	*Gérer le state*	*useState, événements*	*Compteur, formulaire simple*
5-6	*Cycle de vie*	*useEffect, nettoyage, dépendances*	*Récupération API (ex. utilisateurs)*
7-8	*Routing*	*React Router, routes imbriquées, navigation*	*App multi-pages (Accueil, Contact, Blog)*
9-10	*Formulaires*	*Gestion d'entrée, validation simple*	*Formulaire d'inscription*
11-12	*Appels API*	*axios, affichage dynamique, chargement*	*Liste de posts depuis une API*
13	*Gestion globale du state*	*useContext, partage entre composants*	*Thème clair/sombre*
14	*Custom Hooks*	*Réutilisation logique*	*Hook de récupération de données*
15-16	*Intégration Tailwind*	*Styliser avec Tailwind dans React*	*Interface utilisateur réactive*
17	*Authentification*	*Formulaire de connexion, gestion de token*	*Auth mockée localement*
18-20	*Mini-projet React*	*Création d'une vraie app : dashboard, blog, gestion de tâches, quiz, etc.*	*Projet complet*

Phase 4 : Bases du développement Back-End (au choix : Django ou Spring Boot) 35 jours

Option A : Django (Python) 35 jours

Jour(s)	*Objectif*	*Contenu à apprendre*	*Projet/Exercice*
1-2	*Lancer un projet Django*	*Installation, architecture du projet*	*Création du projet + app*
3-5	*Routage et vues*	*URL, vues function/class-based*	*Afficher des pages simples*
6-8	*Modèles & ORM*	*Définir modèles, migrations, admin Django*	*App de gestion de livres ou tâches*
9-10	*Templates & forms*	*Templates, formulaire HTML, CSRF*	*Formulaire de contact*
11-13	*API avec DRF*	*Django REST Framework, serializers*	*API CRUD simple*
14-15	*Viewsets & Routers*	*DRF avancé*	*API RESTful propre*
16-18	*Authentification*	*Tokens, permissions, JWT (SimpleJWT)*	*Connexion + API sécurisée*
19-21	*Relations complexes*	*ForeignKey, ManyToMany*	*Blog avec auteurs & commentaires*
22-24	*Pagination, filtres, recherche*	*DRF filter_backends*	*API consultable*
25-27	*Tests & sécurité*	*Tests unitaires, gestion des erreurs*	*Test API & validation d'entrée*
28-30	*Déploiement local*	*SQLite → PostgreSQL, environnement .env*	*Projet prêt au déploiement*
31-35	*Mini-projet complet*	*API + admin + frontend (React ou template Django)*	*Blog / CRM / Dashboard API-first*

Option B : Spring Boot (Java) 35 jours

Jour(s)	*Objectif*	*Contenu à apprendre*	*Projet/Exercice*
1-2	*Lancer un projet Spring*	*Initializr, structure, @SpringBootApplication*	*Hello World*
3-5	*REST Controllers*	*@RestController, routing, DTOs*	*Exposer des endpoints*
6-8	*Modèles & JPA*	*Entités, relations, JPA Repository*	*CRUD sur une entité*
9-10	*Services & logique métier*	*Services @Service, injection de dépendances*	*Séparer logique métier*
11-13	*API REST complète*	*DTO, validations, status HTTP*	*API pour utilisateurs ou produits*
14-17	*Sécurité avec Spring Security*	*Auth JWT, roles, tokens*	*Authentification sécurisée*
18-21	*Gestion des erreurs*	*@ControllerAdvice, custom exceptions*	*API robuste*
22-25	*Tests & logs*	*JUnit, MockMvc, Slf4j*	*Tester les endpoints*
26-29	*Relations & jointures*	*@OneToMany, @ManyToOne, cascade*	*Blog / Auteur / Article*
30-32	*PostgreSQL & environnement*	*Configuration application.properties, .env*	*Connexion base distante*
33-35	*Mini-projet*	*API complète : Blog, gestion de tâches, CRM*	*Projet final complet*

Projets transversaux possibles

- *Application ToDo (React + API)*
- *Blog avec commentaires (React + Django/Spring)*
- *Tableau de bord admin (React + Tailwind + API)*
- *Application e-commerce simplifiée*
- *Portfolio dynamique (avec back-office)*

Outils & plateformes recommandés

Type	***Outils***
IDE	*VS Code, IntelliJ (Spring), PyCharm (Django)*
API Test	*Postman, Thunder Client*
DB	*SQLite (dev), PostgreSQL (prod), MySQL ou MongoDB*
Versionnage+ collabortion	*Git, GitHub, Trello*
Déploiement	*Vercel (React), Render / Railway / Heroku (Backend)*
