Lazar STEVANOVIC

Guide d'Utilisation et Explication du Projet Next.js (Enregistrement + Connexion)

Partie 1 : Comment Lancer le Projet

Pour exécuter correctement ce projet Next.js, suivez les étapes suivantes :

1. Télécharger le projet

Clonez le projet depuis le dépôt GitHub ou téléchargez le dossier du projet.
« **https://github.com/Lazar1244/Auth** »

2. Installer les dépendances

Ouvrez un terminal dans le dossier du projet et exécutez :

npm install

3. Configurer la base de données

• Assurez-vous que MySQL via XAMPP est en cours d'exécution,
• Vérifiez que le fichier .env contient la bonne configuration pour la base de données.
• Appliquez les migrations Prisma (pour transferer la structure sur votre BD) en exécutant :

npx prisma migrate dev --name init

4. Lancer le projet

• Pour démarrer le serveur de développement :

npm run dev

• Accédez ensuite au site via **http://localhost:3000/** dans le navigateur.

Partie 2 : Explication du Fonctionnement du Projet

Ce projet est une application web sécurisée basée sur Next.js qui gère l'authentification des utilisateurs.

1. Processus d'Inscription et Connexion

• Un utilisateur peut s'inscrire en fournissant son nom, son email et un mot de passe.
• Le mot de passe est sécurisé avant d'être stocké en base de données.
• Lors de la connexion, l'utilisateur entre ses informations, qui sont vérifiées avant de lui accorder l'accès.

2. Protection des Pages

• La page principale (/) est protégée et redirige les utilisateurs non connectés vers /login.
• Une fois connecté, l'utilisateur voit ses informations personnelles.
• Il peut se déconnecter en un clic, ce qui supprime sa session.

Lazar STEVANOVIC

3. Utilisation de JWT et Sessions

• Une session est créée lorsqu'un utilisateur se connecte.
• Cette session est stockée dans un JWT (JSON Web Token) pour garantir une authentification sécurisée.

Partie 3 : Explication de Next.js et Sécurité du Projet xD

Librairies Utilisées et Leur Rôle

• NextAuth.js : Gestion de l'authentification sécurisée avec sessions et JWT
• Prisma : ORM permettant l'interaction sécurisée avec la base de données MySQL
• bcryptjs : Hashage sécurisé des mots de passe pour éviter toute fuite
• Zod : Validation des entrées utilisateur pour empêcher les injections SQL et XSS
• jsonwebtoken : Création de tokens JWT pour authentifier les utilisateurs

Pourquoi ce Projet est Sécurisé ?

1. Hashage des Mots de Passe
• Tous les mots de passe sont sécurisés avec bcryptjs avant d'être stockés.

2. Validation des Entrées
• Zod empêche toute tentative d'injection SQL ou de saisie invalide.

3. Sessions Sécurisées
• L'utilisation de JWT et NextAuth empêche le vol de sessions.

4. Pages Protégées
• Les utilisateurs non connectés sont immédiatement redirigés vers /login.

5. Déconnexion Sécurisée
• Lorsqu'un utilisateur se déconnecte, il ne peut plus accéder aux pages protégées.