LE PROJET: SDLC

1. Identification des outils de sécurité nécessaires

A. Sécurité des dépendances :

- **npm audit** : Pour analyser les vulnérabilités des packages Node.js.
- **Snyk** : Pour effectuer des scans de vulnérabilités approfondis des dépendances.

B. Analyse de la sécurité des applications :

- **OWASP ZAP** : Scanner de sécurité pour les applications web permettant de détecter des failles courantes.

C. Sécurité des API :

- **Postman** : Plateforme pour tester et sécuriser les API.

D. Sécurité des images Docker :

- **Trivy** : Outil d'analyse des vulnérabilités dans les images Docker.

E. Modélisation des menaces :

- **Microsoft Threat Modeling Tool** : Outil pour identifier les menaces potentielles dans l'architecture de l'application.
- **OWASP Top 10** : Guide des risques de sécurité web les plus courants.

F. Bonnes pratiques de sécurité :

- **Node.js Security Best Practices** : Guide pour sécuriser les applications Node.js.
- **Vue.js Security Guide** : Guide pour sécuriser les applications développées avec Vue.js.

---

2. Dépôt Git et configuration des branches

Dépôt Git :
URL : Lien du dépôt

Stratégie de branches :

- **main** : Code stable prêt pour la production.
- **develop** : Branche pour les intégrations en cours.
- **feature/*** : Branches dédiées à chaque fonctionnalité.
- **hotfix/*** : Branches pour les corrections de bugs en urgence.

Règles Git :

1. Les contributions se font via des Pull Requests.
2. Les branches main et develop sont protégées, exigeant une revue de code avant fusion.
3. Les tests CI/CD sont déclenchés automatiquement pour valider les modifications.

3.Calendrier du projet

Phase	Activités	Durée
Phase 1 : Planification	- Définir les objectifs - Identifier les outils nécessaires - Créer le dépôt Git et configurer les branches	1 semaine
Phase 2 : Documentation	- Identifier les fonctionnalités sensibles - Établir une matrice RACI - Identifier les risques associés	2 semaines
Phase 3 : Conception	- Analyser l'architecture - Identifier les menaces - Proposer des contre-mesures	3 semaines

Phase 4 : Développement	- Implémenter les fonctionnalités - Ajouter les tests unitaires et d'intégration	4 semaines
Phase 5 : Tests	- Exécuter les tests de sécurité - Identifier les vulnérabilités - Créer un rapport	2 semaines
Phase 6 : Déploiement	- Configurer CI/CD - Lancer l'application en production	1 semaine
Phase 7 : Maintenance	- Mettre en place un plan de gestion des vulnérabilités - Planifier des mises à jour	Continu

---

4. La matrice RACI

Activité	Responsable (R)	Approbateur (A)	Consulté (C)	Informé (I)
Définir les exigences	Chef de projet	Directeur technique	Équipe Dev	Client
Développement des fonctionnalités	Développeur	Lead Dev	Équipe Dev, Équipe QA	Client
Tests de sécurité	Équipe QA	Responsable Sécurité	Développeur	Client
Déploiement	DevOps	Chef de projet	Équipe QA	Client
Gestion des permissions et rôles	Responsable Sécurité	Directeur technique	Équipe Dev, DevOps	Client

---

5.Risques identifiés

Processus	Risque	Impact	Probabilité	Mesures d'atténuation
Gestion des identifiants	Compromission des comptes utilisateur	Élevé	Moyen	- Appliquer une politique de mots de passe robustes via Snyk (audit des dépendances liées à l'authentification). - Activer l'authentification multi-facteurs (MFA) dans l’application.
Traitement des données	Fuite de données personnelles	Élevé	Moyen	- Chiffrer les données sensibles en transit (TLS) et au repos avec MongoDB TLS/SSL. - Restreindre l'accès aux données via RBAC avec MongoDB Compass et Snyk pour surveiller les vulnérabilités.
Notifications par e-mail	Phishing ou spam	Moyen	Faible	- Ajouter des en-têtes DKIM et SPF pour authentifier les e-mails. - Analyser les contenus sensibles avec Snyk Code pour détecter des vulnérabilités liées aux injections dans les e-mails.
Gestion des rôles et permissions	Permissions mal configurées entraînant un accès non autorisé	Élevé	Moyen	- Effectuer des revues régulières des permissions avec Snyk et MongoDB Compass. - Restreindre les actions sensibles grâce à RBAC avec MongoDB. - Mettre en place un audit d'accès via Audit Log et Trivy pour surveiller les anomalies.

6. RBAC aligné avec le RACI

Activité	Responsable (R)	Approbateur (A)	Consulté (C)	Informé (I)
Définir les exigences	**Chef de projet**		**Directeur technique**	**Équipe Dev, Client**
Développement des fonctionnalités	**Développeur**		**Lead Dev, Équipe QA**	**Client**
Tests de sécurité	**Équipe QA**		**Responsable Sécurité**	**Développeur, Client**
Déploiement	**DevOps**	**Chef de projet**	**Équipe QA**	**Client**
Gestion des permissions et rôles	**Responsable Sécurité**	**Directeur technique**	**Équipe Dev, DevOps**	**Client**

7. Intégration du Flux de Sécurité dans le Workflow

- **Étape 1 :** Authentification via une connexion sécurisée avec MFA.
- **Étape 2 :** Attribution des droits d'accès basée sur les rôles après authentification.
- **Étape 3 :** Gestion sécurisée des données lors de la soumission des formulaires.
- **Étape 4 :** Surveillance continue et alertes en cas d'anomalies.
- **Étape 5 :** Audits de sécurité réguliers et mises à jour.

8. Priorisation des Fonctionnalités en Fonction de Leur Criticité

Fonctionnalité	**Niveau de Priorité**
Authentification	Haute
Gestion des Utilisateurs	Haute
Formulaires	Moyenne

9. Identification et Justification des Fonctionnalités Sensibles

Fonctionnalités Sensibles :
1. Authentification

○Raison de l'Attention Particulière : Garantit que seuls les utilisateurs autorisés accèdent au système. Les vulnérabilités dans les mécanismes d'authentification peuvent conduire à des accès non autorisés, compromettant les données sensibles.
○ Mesures de Sécurité : Mise en œuvre de jetons JWT à courte durée de vie, authentification à plusieurs facteurs (MFA) et politiques de mots de passe robustes.

2. Gestion des Utilisateurs

○Raison de l'Attention Particulière : Permet le contrôle des rôles, des permissions et de l'accès aux données. Une mauvaise gestion peut entraîner une escalade des privilèges ou des fuites de données.
○ Mesures de Sécurité : Contrôle d'accès basé sur les rôles (RBAC), journalisation des sessions utilisateur et mécanismes de verrouillage de compte en cas d'activité suspecte.

3. Formulaires (Entrée et Soumission de Données)

○ Raison de l'Attention Particulière : Vecteurs communs d'attaques par injection (ex. : injections SQL, XSS). Une gestion incorrecte peut compromettre l'intégrité des données et la sécurité du système.

○ Mesures de Sécurité : Validation et assainissement des entrées, intégration de CAPTCHA et protocoles de transmission de données sécurisés.