Rapport d'Audit de Sécurité – Application Planify

1. Introduction

Ce rapport présente l'audit de sécurité de l'application Planify (frontend Vue.js, backend Node.js/Express, MongoDB). L'objectif est d'identifier les forces, faiblesses et recommandations pour garantir la sécurité des données et des utilisateurs.

2. Architecture et technologies utilisées

- **Frontend** : Vue.js
- **Backend** : Node.js, Express
- **Base de données** : MongoDB
- **Hébergement** : Plesk (SSL Let's Encrypt, HSTS)

3. Protections en place

Backend

- Mots de passe hachés (bcrypt, 12 rounds)
- Validation stricte des données (express-validator)
- Rate limiting (express-rate-limit)
- Headers de sécurité (Helmet, CSP, HSTS)
- JWT sécurisé (secret fort, validation issuer/audience)
- Gestion centralisée des erreurs
- Limitation de la taille des requêtes
- CORS géré par Plesk
- Logs de sécurité

Frontend

- Headers de sécurité pour les requêtes API
- Gestion automatique des tokens expirés
- Validation côté client
- Protection XSS native de Vue.js

Base de données

- Connexion sécurisée (timeouts, pas d'exposition de mot de passe)
- Validation des schémas
- Pas d'exposition des mots de passe

4. Résultats des audits de dépendances

- **Backend** : 0 vulnérabilité (`npm audit`)
- **Frontend** : 1 vulnérabilité faible (brace-expansion, DoS). Correction possible avec `npm audit fix`.

5. Checklist de sécurité

- [x] Authentification sécurisée (bcrypt, JWT fort)
- [x] Validation des données partout
- [x] Rate limiting actif
- [x] Headers de sécurité (CSP, HSTS, etc.)
- [x] CORS géré par le serveur web (Plesk)
- [x] Limitation de la taille des requêtes
- [x] Gestion des erreurs centralisée
- [x] Pas de mot de passe en clair
- [x] Variables d'environnement sensibles (JWT_SECRET, etc.)
- [x] HTTPS activé (SSL Let's Encrypt)
- [x] HSTS activé
- [ ] OCSP Stapling (à vérifier sur Plesk)
- [x] Logs de sécurité
- [x] Sauvegardes MongoDB recommandées

6. Recommandations complémentaires

- Corriger la vulnérabilité faible du frontend :
    - Dans le dossier `frontend`, lancer : `npm audit fix`
- Vérifier que le JWT_SECRET est bien fort et unique en production.
- Vérifier que la base MongoDB n'est pas exposée publiquement (firewall, accès restreint).
- Mettre à jour régulièrement les dépendances (`npm update`).
- Effectuer un test de pénétration manuel (injection, XSS, CSRF, brute force).
- Surveiller les logs d'accès et d'erreur.
- Mettre en place un monitoring (fail2ban, alertes mail, etc.).
- Vérifier la configuration des sauvegardes automatiques de la base.

7. Tests réalisés et à réaliser

- Test de connexion avec de mauvais identifiants (rate limiting)
- Test de validation des formulaires (données invalides)
- Vérification des headers de sécurité (`curl -I https://votre-domaine/`)
- Test de la réinitialisation de mot de passe (questions secrètes)
- Test de l'expiration du token JWT

8. Conclusion

L'application Planify présente un niveau de sécurité élevé sur les points essentiels. Il reste à corriger la vulnérabilité faible du frontend et à vérifier OCSP Stapling sur Plesk. Il est recommandé de maintenir à jour les dépendances et de surveiller régulièrement les logs et alertes de sécurité.

Rapport généré automatiquement le {{date du jour}}.