CAHIER DES CHARGES COMPLET

PLATEFORME DE GESTION DES PROJETS DE FIN D'ÉTUDES (PFE)

ESPRIT/ESPRIM - Version 2.0

Document de référence : CDC-PFE-2025-v2.0
Date d'établissement : 19 Novembre 2025
Validité : 12 mois (jusqu'au 19 Novembre 2026)
Statut : Document contractuel

TABLE DES MATIÈRES

1. PRÉSENTATION DU PROJET

1.1 Contexte Général

ESPRIT (École Supérieure Privée d'Ingénierie et de Technologies) et ESPRIM (ESPRIT Monastir) gèrent annuellement plusieurs centaines de Projets de Fin d'Études (PFE) impliquant étudiants, encadrants académiques, encadrants entreprise, évaluateurs et administrateurs. Le processus actuel, largement manuel, génère des inefficacités, des risques d'erreurs et une charge administrative considérable.[7][11][^12]

1.2 Problématiques Identifiées

Gestion Administrative

- Suivi fragmenté des propositions et affectations PFE
- Processus de validation et d'évaluation non standardisés
- Communication inefficace entre parties prenantes
- Absence de traçabilité et d'audit trail
- Risques de perte de documents et d'informations

Sécurité et Confidentialité

- Accès non contrôlés aux données sensibles
- Absence de chiffrement des rapports PFE
- Pas de système d'authentification robuste
- Non-conformité RGPD potentielle[^13]
- Risques de plagiat non détectés

Expérience Utilisateur

- Interfaces obsolètes ou inexistantes
- Processus longs et complexes
- Manque de visibilité sur l'avancement
- Communication asynchrone et fragmentée

1.3 Solution Proposée

Développement d'une **plateforme web intégrée et sécurisée** permettant la gestion complète du cycle de vie des PFE, de la proposition initiale à la soutenance finale, avec :

- Authentification multi-facteurs et contrôle d'accès granulaire[14][15]
- Interface moderne et intuitive inspirée de l'identité ESPRIT/ESPRIM[11][12]
- Modules IA pour détection de plagiat et analyse automatique
- Conformité RGPD et chiffrement des données sensibles[16][17][^13]
- Dashboard temps réel et analytics avancés
- Architecture évolutive et scalable

1.4 Bénéfices Attendus

Pour les Étudiants

- Processus simplifié de soumission et suivi PFE
- Visibilité temps réel sur l'avancement
- Communication facilitée avec encadrants
- Accès sécurisé à leurs documents

Pour les Encadrants

- Gestion centralisée de tous les PFE encadrés
- Outils d'évaluation standardisés
- Notifications automatiques des deadlines
- Réduction de la charge administrative

Pour l'Administration

- Automatisation des tâches répétitives
- Vision globale et analytics détaillés
- Traçabilité complète des opérations
- Conformité réglementaire assurée
- Réduction des coûts opérationnels (estimation : -40 à 75%)[18][19]

2. CONTEXTE ET OBJECTIFS

2.1 Objectifs Stratégiques

Objectif Principal
Moderniser et digitaliser intégralement le processus de gestion des PFE pour améliorer l'efficacité opérationnelle, garantir la sécurité des données et offrir une expérience utilisateur de qualité supérieure.

Objectifs Spécifiques

1. **Automatisation** : Réduire de 60% le temps de traitement administratif[4][18]
2. **Sécurité** : Atteindre 0 incident de sécurité majeur durant la première année[^20]
3. **Adoption** : 85%+ des utilisateurs actifs dans les 3 mois post-lancement
4. **Performance** : Temps de chargement <3s pour 95% des pages
5. **Conformité** : 100% conforme RGPD et normes académiques tunisiennes[^13]

2.2 Périmètre du Projet

Inclus dans le Périmètre

- Gestion complète du cycle de vie PFE (proposition → soutenance → archivage)
- Système d'authentification et autorisation (JWT + 2FA)[15][14]
- Modules pour tous les rôles utilisateurs (étudiants, encadrants, admin, jury)
- Upload, versioning et stockage sécurisé de documents
- Module IA de détection de plagiat et analyse automatique
- Dashboard analytics et reporting
- Notifications push et email
- Interface web responsive (desktop, tablet, mobile)
- API RESTful documentée (OpenAPI/Swagger)
- Conformité RGPD complète[^13]

Exclus du Périmètre (Phase 1)

- Application mobile native (iOS/Android) → Phase 4
- Intégration SSO avec systèmes universitaires existants → Phase 2
- Module de gestion des stages (hors PFE) → Extension future
- Signature électronique qualifiée → Phase 3
- Visioconférence intégrée pour soutenances → Phase 4
- Support multilingue (arabe/anglais) → Phase 3

2.3 Parties Prenantes

Rôle	Responsabilités	Niveau d'Implication
Direction ESPRIT/ESPRIM	Validation stratégique, budget	Comité de pilotage mensuel
Service des Stages	Expression besoins, recette fonctionnelle	Quotidien
Service Informatique	Infrastructure, déploiement, maintenance	Quotidien
Encadrants Académiques	Tests utilisateurs, feedback	Hebdomadaire
Étudiants (panel)	Tests UX, validation interface	Bi-mensuel
RSSI (si existant)	Validation sécurité, audit	Revues sécurité
DPO	Conformité RGPD	Validation DPIA

3. SPÉCIFICATIONS FONCTIONNELLES

3.1 Module Gestion des Utilisateurs

Fonctionnalités Principales

Authentification et Profils

- Inscription avec validation email obligatoire
- Login sécurisé (email/password + 2FA TOTP)[14][15]
- Gestion profil utilisateur (photo, coordonnées, CV)
- Modification mot de passe avec historique (empêcher réutilisation des 5 derniers)
- Récupération mot de passe par lien temporaire (validité 30 min)
- Support clés de sécurité matérielles (U2F/FIDO2) pour admins[^15]

Gestion des Rôles et Permissions (RBAC)[19][18]

Hiérarchie des rôles :

```
SUPER_ADMIN (Direction/DSI)
├── ADMIN_STAGES (Chef service stages)
│   ├── GESTIONNAIRE_STAGES (Personnel service)
│   └── COORDINATEUR_SPECIALITE (Par département)
├── ENCADRANT_ACADEMIQUE
│   └── ENCADRANT_ASSISTANT (Co-encadrant)
├── EVALUATEUR
│   ├── EVALUATEUR_RAPPORT
│   ├── EVALUATEUR_MI_PARCOURS
│   └── MEMBRE_JURY
├── ENCADRANT_ENTREPRISE
├── ETUDIANT
│   ├── ETUDIANT_PFE (en cours)
│   └── ETUDIANT_DIPLOME (terminé)
└── INVITE (consultation limitée)
```

Matrice des Permissions Détaillée

Gestion des Propositions PFE

Action	Étudiant	Enc. Entreprise	Enc. Académique	Évaluateur	Gestionnaire	Admin
Soumettre proposition	✓ (propre)	-	-	-	-	✓
Modifier proposition	✓ (avant valid.)	✓ (propre étudiant)	-	-	-	✓
Valider/Rejeter	-	-	✓ (assigné)	-	✓	✓
Voir toutes	-	-	✓ (département)	-	✓	✓

Action	Étudiant	Enc. Entreprise	Enc. Académique	Évaluateur	Gestionnaire	Admin
Supprimer	-	-	-	-	-	✓

Gestion des Rapports

Action	Étudiant	Encadrant	Évaluateur	Gestionnaire	Admin
Déposer rapport	✓ (propre)	-	-	-	✓
Télécharger	✓ (propre)	✓ (encadré)	✓ (assigné)	✓	✓
Évaluer	-	✓ (encadré)	✓ (assigné)	-	✓
Voir notes	✓ (propre)	✓ (encadré)	✓ (évalué)	✓	✓
Modifier notes	-	-	✓ (24h après)	-	✓

Administration

Action	Gestionnaire	Coordinateur	Admin	Super Admin
Gérer utilisateurs	✓ (étudiants)	✓ (département)	✓	✓
Affecter encadrants	✓	✓	✓	✓
Config. grilles éval.	-	✓	✓	✓
Paramètres système	-	-	✓	✓
Logs sécurité	-	-	✓	✓
Gérer rôles/perm.	-	-	-	✓

3.2 Module Gestion des Propositions PFE

Soumission de Proposition (Étudiant)

Formulaire Multi-étapes

1. **Informations Générales** (Étape 1/4)
   - Titre du PFE (min 10 caractères, max 200)
   - Type : Stage PFE / Projet académique / Recherche
   - Spécialité : Liste déroulante (GL, IA, Réseaux, etc.)
   - Date début souhaitée / Date fin prévue
2. **Description Détaillée** (Étape 2/4)
   - Contexte et problématique (min 200 caractères)
   - Objectifs du projet (liste à puces, min 3 objectifs)
   - Technologies envisagées (tags auto-complétion)
   - Livrables attendus

3. **Informations Entreprise/Organisme** (Étape 3/4)
    - Nom entreprise
    - Secteur d'activité
    - Adresse complète
    - Encadrant entreprise (nom, email, téléphone, fonction)
    - Convention déjà signée ? (Oui/Non) → Upload si oui
4. **Documents Justificatifs** (Étape 4/4)
    - Lettre de motivation (PDF, max 5MB)
    - CV étudiant (PDF, max 5MB)
    - Attestation entreprise (PDF, max 5MB, optionnel)
    - Cahier des charges préliminaire (PDF, max 10MB, optionnel)

Validation et Workflow

```
[Soumission Étudiant]
    ↓
[Vérification Automatique] (format, complétude)
    ↓
[En Attente Validation] → Notification Coordinateur
    ↓
[Révision Coordinateur] (3 jours max)
    ↓
    ├── [APPROUVÉ] → Notification étudiant + Affectation encadrant
    ├── [REJETÉ] → Notification + Commentaires obligatoires
    └── [DEMANDE MODIFICATION] → Retour à étudiant
```

Affectation Encadrant (Coordinateur/Admin)

- Liste propositions validées sans encadrant
- Filtres : Spécialité, Technologies, Date soumission
- Suggestion automatique encadrants (IA matching) basée sur :
    - Domaine d'expertise
    - Charge actuelle (max 5 PFE simultanés, configurable)
    - Historique encadrements
    - Disponibilité déclarée
- Affectation manuelle avec confirmation
- Notification automatique encadrant + étudiant

3.3 Module Gestion des Rapports

Dépôt de Rapport (Étudiant)

Types de Rapports

- Rapport bibliographique (optionnel, 1er mois)
- Rapport mi-parcours (obligatoire, après 50% durée)
- Rapport final (obligatoire, avant soutenance)
- Rapport stage entreprise (si applicable)

Interface de Dépôt

- Drag & drop ou sélection fichier
- Format accepté : PDF uniquement
- Taille max : 50MB
- Vérifications automatiques :
  - Scan antivirus (ClamAV)[^20]
  - Détection format/corruption
  - Extraction métadonnées (auteur, date création)
  - Watermarking automatique avec identité étudiant
- Champs obligatoires :
  - Titre exact du rapport
  - Type de rapport
  - Résumé (max 500 caractères)
  - Mots-clés (min 3, max 10)
- Versioning automatique (v1.0, v1.1, etc.)
- Historique complet des dépôts conservé

Analyse IA Automatique (après upload)

1. **Détection de Plagiat**
   - Comparaison avec base interne (tous rapports précédents)
   - Comparaison web (APIs externes optionnelles)
   - Score de similarité (0-100%)
   - Rapport détaillé avec passages suspects surlignés
   - Seuil alerte : >30% similarité → Notification encadrant + admin
2. **Évaluation Préliminaire**
   - Analyse structure (présence chapitres attendus)
   - Qualité rédactionnelle (fautes orthographe via LanguageTool)

- Longueur adéquate (pages, mots)
- Présence bibliographie/références
- Score qualité global (A/B/C/D)
- Suggestions d'amélioration automatiques

Évaluation par Encadrant/Jury

Grille d'Évaluation Personnalisable

Rapport Final - Exemple Spécialité Génie Logiciel

Critère	Poids	Note /20	Commentaire
Qualité rédaction	10%	[___]	[Champ texte]
État de l'art	15%	[___]	[Champ texte]
Analyse/Conception	25%	[___]	[Champ texte]
Implémentation	30%	[___]	[Champ texte]
Tests et validation	15%	[___]	[Champ texte]
Respect cahier charges	5%	[___]	[Champ texte]
Total	**100%**	**[Calc auto]**	

Fonctionnalités Évaluation

- Grilles par type rapport et spécialité (configurables admin)
- Calcul automatique note finale pondérée
- Sauvegarde brouillon (évaluation partielle)
- Validation finale (verrouillage après 24h modifiable uniquement par admin)
- Export PDF évaluation complète
- Historique modifications avec audit trail

3.4 Module Gestion des Soutenances

Planification (Gestionnaire/Coordinateur)

Interface Calendrier

- Vue calendrier mensuel/hebdomadaire/quotidien
- Créneaux configurables (ex: 9h-12h, 14h-18h)
- Durée soutenance paramétrable (défaut : 1h)
- Salles disponibles (gestion ressources)
- Détection automatique conflits (jury/salle/heure)

Composition Jury

- Minimum : Président + Rapporteur + Encadrant (3 membres)
- Maximum : 5 membres
- Rôles : Président, Rapporteur, Examinateur, Encadrant, Invité entreprise
- Contraintes :
  - Encadrant ne peut être président
  - Au moins 1 membre externe au département
  - Disponibilité vérifiée automatiquement
- Affectation drag & drop depuis pool encadrants
- Génération automatique convocations (PDF) avec QR code

Saisie Notes Soutenance

Grille Jury - Exemple

Critère	Poids	Note /20	Évaluateur
Présentation orale	20%	[___]	Tous
Maîtrise technique	30%	[___]	Président
Réponses aux questions	25%	[___]	Tous
Qualité support (slides)	10%	[___]	Rapporteur
Respect timing	5%	[___]	Président
Professionnalisme	10%	[___]	Tous
Total Soutenance	**100%**	**[Auto]**	

Calcul Note Finale PFE

```
Note Finale = (Note Rapport × 40%) + (Note Soutenance × 40%) + (Note Encadrant Entreprise
```

Pondérations configurables par spécialité.

Délibération

- Mention : Très Bien / Bien / Assez Bien / Passable / Ajourné
- Seuils configurables (ex: TB ≥ 16, B ≥ 14, AB ≥ 12, P ≥ 10)
- Commentaires jury (optionnel)
- Validation collégiale (signature électronique simple)
- Génération automatique PV soutenance

3.5 Module Notifications et Communications

Notifications Push (In-app)

- Badge compteur notifications non lues (navbar)
- Dropdown liste notifications récentes (10 dernières)
- Page complète notifications avec filtres
- Marquage lu/non lu
- Actions rapides depuis notification

Types de Notifications

Événement	Destinataire(s)	Priorité
Nouvelle proposition soumise	Coordinateur	Normale
Proposition validée/rejetée	Étudiant	Haute
Encadrant affecté	Encadrant + Étudiant	Haute
Rapport déposé	Encadrant	Normale
Évaluation terminée	Étudiant	Haute
Plagiat détecté (>30%)	Encadrant + Admin	Critique
Soutenance planifiée	Tous membres jury + Étudiant	Haute
Rappel deadline (J-7, J-3, J-1)	Étudiant	Haute
Note finale publiée	Étudiant	Haute

Notifications Email

- Même logique que push notifications
- Template HTML responsive avec logo ESPRIT
- Option désactivation par type (paramètres utilisateur)
- Digest quotidien optionnel (résumé journalier)

Messagerie Interne

- Chat 1-to-1 Étudiant ↔ Encadrant académique
- Chat 1-to-1 Étudiant ↔ Encadrant entreprise
- Pas de chat groupe (éviter dispersion)
- Historique messages conservé et searchable
- Indicateur "en ligne" / "dernière activité"
- Upload fichiers légers (<5MB) dans discussions

3.6 Module Analytics et Reporting (Admin/Coordinateur)

Dashboard Principal

KPIs Temps Réel

- Nombre total PFE en cours / terminés / en attente
- Taux validation propositions (%)
- Taux soumission rapports mi-parcours/final à date
- Nombre soutenances planifiées ce mois
- Moyenne générale notes PFE (par spécialité)
- Taux détection plagiat (nombre cas / total rapports)

Graphiques Interactifs

- Évolution nombre PFE par mois (ligne chart)
- Distribution notes finales (histogramme)
- Top 10 technologies utilisées (bar chart)
- Charge encadrants (nombre PFE par encadrant, bubble chart)
- Taux réussite par spécialité (pie chart)
- Timeline soutenances (Gantt chart)

Rapports Exportables

Rapport	Description	Format Export
État global PFE	Vue complète tous PFE actifs	PDF, Excel
Performance encadrants	Nombre encadrements, moyennes notes	Excel
Bilan annuel par spécialité	Stats complètes année académique	PDF
Détection plagiat	Liste tous cas détectés avec scores	Excel, PDF
Planning soutenances	Calendrier complet avec jurys	PDF, iCal
Export RGPD utilisateur	Toutes données personnelles	JSON, PDF

Filtres Avancés

- Période (année académique, semestre, mois, personnalisé)
- Spécialité (GL, IA, Réseaux, Cybersécurité, etc.)
- Statut (en cours, terminé, ajourné)
- Encadrant
- Entreprise
- Note (plage)

3.7 Module Administration Système

Gestion Utilisateurs (Admin)

- Création manuelle utilisateur (tous rôles)
- Import CSV massif (template fourni)
- Modification profils, rôles, permissions
- Activation/Désactivation comptes
- Réinitialisation mot de passe
- Historique activité utilisateur (logs)
- Export liste utilisateurs (Excel, CSV)

Configuration Paramètres Globaux (Super Admin)

Paramètres Académiques

- Années académiques (création, activation, archivage)
- Spécialités et départements
- Grilles d'évaluation par type rapport/spécialité
- Pondérations calcul notes finales
- Seuils mentions (TB, B, AB, P)
- Durées maximales PFE par type

Paramètres Sécurité

- Politique mots de passe (longueur min, complexité, expiration)
- Durée tokens JWT (access: 15-30 min, refresh: 7-30 jours)[^14]
- Obligation 2FA par rôle
- Limites sessions simultanées par utilisateur
- Rate limiting par endpoint
- Liste IPs bloquées/autorisées (whitelist/blacklist)

Paramètres Système

- Taille max upload fichiers (par type)
- Formats fichiers acceptés
- Durée conservation logs (jours)
- Fréquence backup automatique
- Seuil détection plagiat (%)
- Templates emails notifications

Gestion Logs et Audit[21][22]

- Consultation logs sécurité (connexions, échecs, modifications permissions)

- Logs applicatifs (erreurs, warnings, infos)
- Filtres : Date, utilisateur, type événement, niveau gravité
- Recherche full-text dans logs
- Export logs (JSON, CSV, TXT)
- Statistiques événements (graphiques)
- Alertes automatiques événements critiques

4. ARCHITECTURE TECHNIQUE

4.1 Stack Technologique

Frontend - Application Web

Framework et Librairies

- **React 18.3+** avec TypeScript 5.x
  - Typage fort pour réduction bugs
  - Composants fonctionnels + Hooks
  - Strict mode activé
- **State Management**
  - Redux Toolkit (état global application)
  - React Query v5 (cache serveur, synchronisation)
  - Zustand (états UI légers)
- **Routing** : React Router v6
- **UI Library** : Material-UI (MUI) v5 ou Ant Design v5
  - Personnalisation complète thème ESPRIT
  - Composants accessibles WCAG 2.1 AA
- **Formulaires** : React Hook Form + Zod (validation TypeScript-first)
- **Charts** : Recharts ou Chart.js v4
- **HTTP Client** : Axios avec interceptors (JWT auto-refresh)
- **Build Tool** : Vite 5.x (HMR ultra-rapide, optimisation production)
- **Testing**
  - Jest + React Testing Library (tests unitaires/intégration)
  - Cypress v13 (tests E2E)
  - Coverage target : >80%

Backend - API RESTful

Serveur Principal (Node.js)

- **Runtime** : Node.js 20.x LTS
- **Framework** : Express.js 4.x ou Fastify 4.x (performances supérieures)
- **Langage** : TypeScript 5.x
- **Base de données** : SQLite 3.x avec **SQLCipher** (chiffrement AES-256)[17][16]
- **ORM** : Prisma 5.x (type-safe, migrations automatiques)
- **Authentification**
  - JWT (jsonwebtoken) : Access + Refresh tokens[^14]
  - Bcrypt (hachage mots de passe, cost factor 12)
  - Speakeasy (2FA TOTP)[^15]
  - @simplewebauthn (U2F/FIDO2 hardware keys)[^15]
- **Validation** : Zod (schémas TypeScript partagés avec frontend)
- **Upload Fichiers** : Multer + ClamAV integration
- **Email** : Nodemailer + templates MJML
- **Logs** : Winston (multi-transports : console, fichier, base)[^21]
- **Testing**
  - Jest (unitaires)
  - Supertest (tests API)
  - Coverage target : >75%

Service IA/ML (Python)

- **Framework** : FastAPI 0.110+
- **Runtime** : Python 3.11+
- **ML/NLP**
  - Scikit-learn (clustering, classification)
  - SpaCy 3.7 (NLP, analyse texte)
  - Sentence-Transformers (embeddings sémantiques)
  - NLTK (preprocessing texte)
- **Détection Plagiat**
  - TF-IDF + Cosine Similarity (base)
  - Doc2Vec (analyse sémantique profonde)
  - Levenshtein distance (similarité chaînes)
- **Queue Asynchrone** : Celery + Redis (tâches longues)
- **Validation** : Pydantic v2 (modèles données)
- **Testing** : PyTest (coverage >70%)

Infrastructure et DevOps

Conteneurisation

- **Docker** : Images multi-stage (optimisation taille)
  - Frontend : nginx alpine + fichiers statiques
  - Backend Node : node:20-alpine
  - Backend Python : python:3.11-slim
  - Database : custom SQLCipher alpine
- **Docker Compose** : Orchestration environnements dev/staging

Base de Données

- **SQLite avec SQLCipher**[23][16][^17]
  - Chiffrement AES-256-CBC transparent
  - Clé de chiffrement 256 bits stockée dans Vault
  - Page size : 4096 bytes
  - PRAGMA cipher_page_size = 4096
  - PRAGMA kdf_iter = 256000 (PBKDF2 iterations)
  - Backup incrémental via SQLite backup API
  - Migration vers PostgreSQL facile si scaling nécessaire

Stockage Fichiers

- **Phase 1** : Local filesystem avec structure organisée

```
/storage
  /reports
    /{year}/{specialty}/{student_id}/
  /proposals
    /{year}/{student_id}/
  /avatars
    /{user_id}/
```

- **Phase 2+** : Migration S3-compatible (AWS S3, MinIO, Backblaze B2)
  - Versioning activé
  - Lifecycle policies (archivage automatique après 2 ans)

Cache et Sessions

- **Redis 7.x**
  - Cache application (requêtes fréquentes)
  - Sessions utilisateurs
  - Rate limiting (compteurs)
  - Queue jobs (Celery broker)
  - Pub/Sub (notifications temps réel optionnel)

Reverse Proxy et Load Balancing

- **Nginx**
  - SSL/TLS termination (Let's Encrypt)
  - Compression gzip/brotli
  - Caching statique
  - Rate limiting applicatif
  - Header sécurité (CSP, HSTS, etc.)

CI/CD

- **GitHub Actions** ou GitLab CI
  - Pipelines :
    - Lint (ESLint, Prettier, Pylint, Black)
    - Tests (Jest, PyTest, Cypress)
    - Build (Docker images)
    - Scan sécurité (Snyk, Trivy)
    - Deploy (staging auto, production manuel)
  - Branches :
    - `main` → Production
    - `develop` → Staging
    - `feature/*` → Feature branches

Monitoring et Observabilité

- **Application Monitoring**
  - Prometheus + Grafana (métriques infrastructure)
  - Sentry (error tracking, performance monitoring)
  - Datadog ou New Relic (APM, alternative premium)
- **Logs Centralisés**
  - ELK Stack (Elasticsearch, Logstash, Kibana) ou Loki
  - Rétention : 90 jours opérationnels, archivage long terme
- **Uptime Monitoring**
  - UptimeRobot ou StatusCake (checks HTTP externes)
  - Healthchecks internes (/health, /ready endpoints)

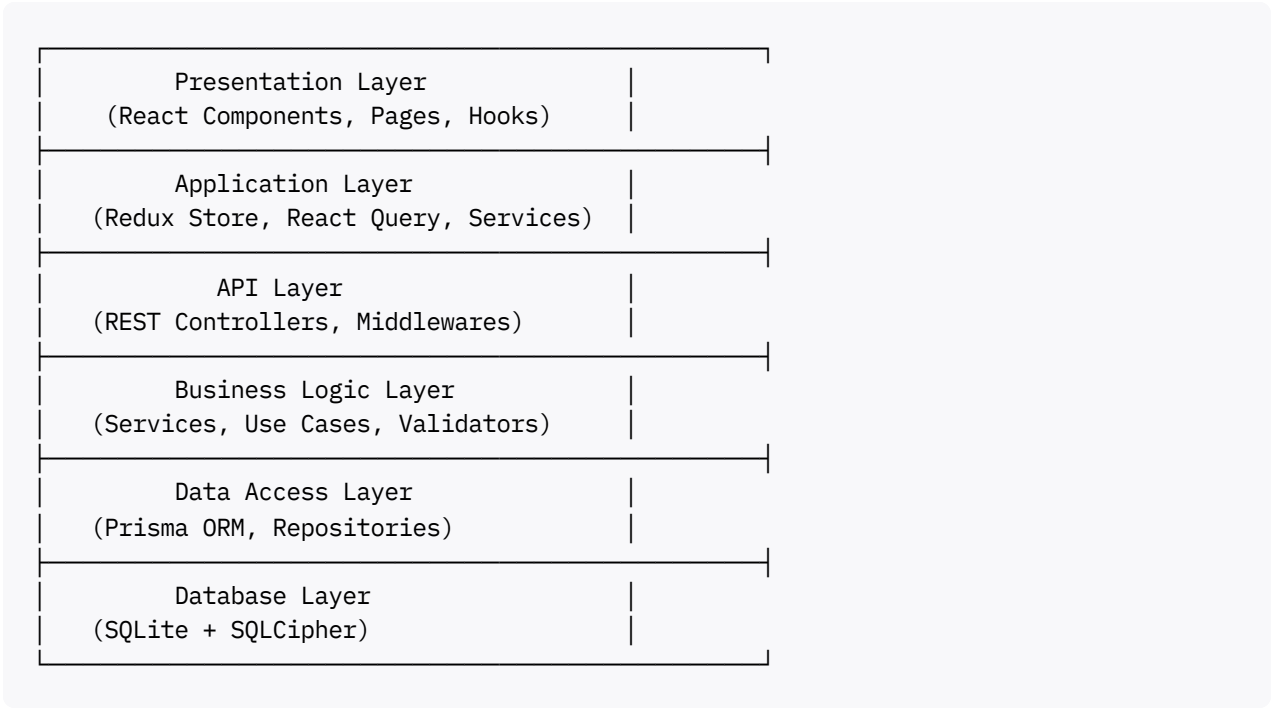
Backup et Disaster Recovery

- **Stratégie 3-2-1**
  - 3 copies données
  - 2 types médias différents

    - 1 copie offsite
- **Fréquence**
    - Base données : Backup complet quotidien (3h du matin) + incrémental 6h
    - Fichiers : Backup quotidien + sync temps réel vers stockage secondaire
- **Rétention**
    - Quotidiens : 7 jours
    - Hebdomadaires : 4 semaines
    - Mensuels : 12 mois
    - Annuels : 5 ans (compliance)
- **Tests restauration** : Mensuels (procédure documentée)
- **RPO** : <6 heures (Recovery Point Objective)
- **RTO** : <4 heures (Recovery Time Objective)

4.2 Architecture Logicielle

Pattern Architectural : Layered Architecture



Modèle de Données Principal (Schéma Prisma)

```
// schema.prisma (simplifié)

model User {
  id            String   @id @default(uuid())
  email         String   @unique
  passwordHash  String
  firstName     String
  lastName      String
```

```
  role          Role
  twoFactorSecret String?
  twoFactorEnabled Boolean @default(false)
  isActive      Boolean  @default(true)
  createdAt     DateTime @default(now())
  updatedAt     DateTime @updatedAt

  // Relations
  studentProfile    StudentProfile?
  supervisorProfile SupervisorProfile?
  sessions          Session[]
  auditLogs         AuditLog[]
  notifications     Notification[]
}

model PFEProposal {
  id              String   @id @default(uuid())
  title           String
  description     String
  status          ProposalStatus
  specialty       String
  technologies    String[] // Array
  studentId       String
  academicSupervisorId String?
  companySupervisorId  String?
  companyName     String?
  startDate       DateTime?
  endDate         DateTime?
  createdAt       DateTime @default(now())
  updatedAt       DateTime @updatedAt

  // Relations
  student         User @relation(fields: [studentId])
  academicSupervisor User? @relation(fields: [academicSupervisorId])
  documents       Document[]
  reports         Report[]
  evaluations     Evaluation[]
  defense         Defense?
}

model Report {
  id            String   @id @default(uuid())
  pfeId         String
  type          ReportType // BIBLIO, MID_TERM, FINAL
  filePath      String
  fileSize      Int
  version       String
  plagiarismScore Float?
  qualityScore    String? // A/B/C/D
  uploadedAt    DateTime @default(now())

  // Relations
  pfe           PFEProposal @relation(fields: [pfeId])
  evaluations   Evaluation[]
  plagiarismReport PlagiarismReport?
}
```

```
model Defense {
  id          String   @id @default(uuid())
  pfeId       String   @unique
  scheduledAt DateTime
  room        String
  duration    Int // minutes
  status      DefenseStatus

  // Relations
  pfe         PFEProposal @relation(fields: [pfeId])
  juryMembers JuryMember[]
  finalGrade  FinalGrade?
}

model Evaluation {
  id          String   @id @default(uuid())
  reportId    String?
  defenseId   String?
  evaluatorId String
  criteria    Json // {criterion: {weight, note, comment}}
  totalScore  Float
  comments    String?
  submittedAt DateTime @default(now())

  // Relations
  report      Report? @relation(fields: [reportId])
  defense     Defense? @relation(fields: [defenseId])
  evaluator   User @relation(fields: [evaluatorId])
}

model AuditLog {
  id          String   @id @default(uuid())
  userId      String?
  action      String
  entity      String
  entityId    String?
  changes     Json?
  ipAddress   String
  userAgent   String
  timestamp   DateTime @default(now())

  // Relations
  user        User? @relation(fields: [userId])
}

enum Role {
  SUPER_ADMIN
  ADMIN_STAGES
  GESTIONNAIRE_STAGES
  COORDINATEUR_SPECIALITE
  ENCADRANT_ACADEMIQUE
  ENCADRANT_ASSISTANT
  EVALUATEUR_RAPPORT
  EVALUATEUR_MI_PARCOURS
  MEMBRE_JURY
```

```
  ENCADRANT_ENTREPRISE
  ETUDIANT_PFE
  ETUDIANT_DIPLOME
  INVITE
}

enum ProposalStatus {
  DRAFT
  SUBMITTED
  UNDER_REVIEW
  MODIFICATION_REQUESTED
  APPROVED
  REJECTED
}

enum ReportType {
  BIBLIOGRAPHY
  MID_TERM
  FINAL
  COMPANY
}

enum DefenseStatus {
  SCHEDULED
  COMPLETED
  CANCELLED
  POSTPONED
}
```

API RESTful - Endpoints Principaux

```
# Authentification
POST   /api/auth/register
POST   /api/auth/login
POST   /api/auth/logout
POST   /api/auth/refresh-token
POST   /api/auth/forgot-password
POST   /api/auth/reset-password
POST   /api/auth/setup-2fa
POST   /api/auth/verify-2fa
POST   /api/auth/verify-u2f

# Utilisateurs
GET    /api/users              # Liste (admin only)
GET    /api/users/:id          # Détail
PUT    /api/users/:id          # Modification
DELETE /api/users/:id          # Suppression (soft delete)
POST   /api/users/import       # Import CSV
GET    /api/users/export       # Export Excel

# Propositions PFE
GET    /api/proposals          # Liste (filtres: status, specialty, year)
GET    /api/proposals/:id      # Détail
POST   /api/proposals          # Création
PUT    /api/proposals/:id      # Modification
```

```
DELETE /api/proposals/:id      # Suppression
POST   /api/proposals/:id/submit       # Soumission validation
PUT    /api/proposals/:id/validate     # Validation (coordinateur)
PUT    /api/proposals/:id/reject       # Rejet
PUT    /api/proposals/:id/assign       # Affectation encadrant

# Rapports
GET    /api/reports            # Liste
GET    /api/reports/:id        # Détail
POST   /api/reports            # Upload
GET    /api/reports/:id/download # Téléchargement (URL signée)
DELETE /api/reports/:id        # Suppression
GET    /api/reports/:id/plagiarism # Résultat plagiat
GET    /api/reports/:id/analysis   # Analyse IA

# Évaluations
GET    /api/evaluations        # Liste
GET    /api/evaluations/:id    # Détail
POST   /api/evaluations        # Création
PUT    /api/evaluations/:id    # Modification
DELETE /api/evaluations/:id    # Suppression

# Soutenances
GET    /api/defenses           # Liste/Calendrier
GET    /api/defenses/:id       # Détail
POST   /api/defenses           # Planification
PUT    /api/defenses/:id       # Modification
DELETE /api/defenses/:id       # Annulation
POST   /api/defenses/:id/jury  # Affectation jury
POST   /api/defenses/:id/grade # Saisie note

# Notifications
GET    /api/notifications      # Liste utilisateur connecté
PUT    /api/notifications/:id/read # Marquage lu
PUT    /api/notifications/read-all # Tout marquer lu
DELETE /api/notifications/:id  # Suppression

# Analytics (admin)
GET    /api/analytics/dashboard    # KPIs principaux
GET    /api/analytics/proposals    # Stats propositions
GET    /api/analytics/reports      # Stats rapports
GET    /api/analytics/defenses     # Stats soutenances
GET    /api/analytics/supervisors  # Performance encadrants
POST   /api/analytics/export       # Export rapport personnalisé

# Administration
GET    /api/admin/settings         # Paramètres globaux
PUT    /api/admin/settings         # Modification paramètres
GET    /api/admin/logs             # Consultation logs
GET    /api/admin/audit            # Audit trail
POST   /api/admin/backup           # Déclenchement backup manuel
GET    /api/admin/health           # Health check système

# IA (Service Python via proxy)
POST   /api/ai/plagiarism          # Détection plagiat
```

```
POST   /api/ai/analyze-report      # Analyse qualité
POST   /api/ai/suggest-supervisors # Matching encadrants
```

Documentation API : OpenAPI 3.1 (Swagger UI automatique sur `/api/docs`)

5. ARCHITECTURE DE SÉCURITÉ

5.1 Système d'Authentification Multi-niveaux

Authentification Principale - JWT[^14]

Configuration Tokens

- **Access Token**
  - Durée : 30 minutes
  - Stockage : Memory (variable JavaScript, pas localStorage)
  - Claims : userId, role, permissions, exp, iat
  - Algorithme : RS256 (asymétrique, clés RSA 2048 bits)
- **Refresh Token**
  - Durée : 14 jours
  - Stockage : HttpOnly Cookie (secure, sameSite: strict)
  - Rotation automatique à chaque utilisation (token family)
  - Révocation possible (stockage database avec expiration)

Workflow Authentification

```
1. User login (email + password)
2. Backend vérifie credentials (bcrypt compare)
3. Si 2FA activé → Demande code TOTP
4. Génération Access Token (30min) + Refresh Token (14j)
5. Access Token → Response body
6. Refresh Token → HttpOnly Cookie
7. Frontend stocke Access Token en mémoire
8. Expiration Access Token → Auto-refresh via endpoint /auth/refresh
9. Refresh Token expiré → Logout forcé
```

Authentification à Deux Facteurs (2FA/MFA)[^15]

TOTP (Time-based One-Time Password)

- Librairie : Speakeasy (Node.js)
- Algorithme : SHA-1 (standard TOTP)
- Période : 30 secondes
- Window : ±1 période (tolérance 30s décalage)

- Apps compatibles : Google Authenticator, Authy, Microsoft Authenticator
- Setup :
  1. Utilisateur active 2FA dans paramètres
  2. Backend génère secret (32 caractères base32)
  3. QR code affiché (librairie qrcode)
  4. Utilisateur scanne avec app authenticator
  5. Vérification code initial obligatoire
  6. Codes backup générés (10 codes à usage unique, stockés hachés)

U2F/FIDO2 (Hardware Security Keys)[^15]

- Pour rôles : SUPER_ADMIN, ADMIN_STAGES obligatoirement
- Librairie : @simplewebauthn/server + @simplewebauthn/browser
- Clés compatibles : YubiKey, Google Titan, Windows Hello
- Enregistrement :
  1. Challenge cryptographique généré (serveur)
  2. Clé matérielle signe challenge
  3. Attestation stockée (public key, counter)
- Authentification :
  1. Challenge généré
  2. Clé signe challenge
  3. Vérification signature + anti-replay (counter)

Réauthentification

- Actions sensibles requièrent réauthentification :
  - Modification paramètres 2FA
  - Changement email/password
  - Suppression compte
  - Téléchargement données personnelles (RGPD)
- Fenêtre validité : 5 minutes

5.2 Contrôle d'Accès RBAC Granulaire[18][19]

Implémentation Middleware

```
// middleware/authorize.ts
import { Request, Response, NextFunction } from 'express';

interface AuthRequest extends Request {
  user?: {
    id: string;
```

```
    role: Role;
    permissions: string[];
  };
}

export const authorize = (
  requiredRoles: Role[],
  requiredPermissions?: string[]
) => {
  return (req: AuthRequest, res: Response, next: NextFunction) => {
    const user = req.user;

    if (!user) {
      return res.status(401).json({ error: 'Non authentifié' });
    }

    // Vérification rôle
    if (!requiredRoles.includes(user.role)) {
      return res.status(403).json({ error: 'Accès refusé - Rôle insuffisant' });
    }

    // Vérification permissions spécifiques
    if (requiredPermissions) {
      const hasPermissions = requiredPermissions.every(perm =>
        user.permissions.includes(perm)
      );

      if (!hasPermissions) {
        return res.status(403).json({ error: 'Accès refusé - Permissions insuffisantes' }
      }
    }

    next();
  };
};

// Utilisation
router.post(
  '/proposals/:id/validate',
  authenticate, // Vérifie JWT
  authorize(
    [Role.COORDINATEUR_SPECIALITE, Role.ADMIN_STAGES],
    ['proposal:validate']
  ),
  validateProposal
);
```

Permissions Granulaires

Format : `{entity}:{action}` (ex: `proposal:create`, `report:evaluate`)

Rôle	Permissions
SUPER_ADMIN	`*:*` (toutes)

Rôle	Permissions
ADMIN_STAGES	`user:*, proposal:*, report:*, defense:*, evaluation:*, settings:read`
GESTIONNAIRE_STAGES	`user:create, user:read, user:update, proposal:*, report:read, defense:*`
COORDINATEUR_SPECIALITE	`proposal:validate, proposal:assign, defense:schedule, evaluation:configure`
ENCADRANT_ACADEMIQUE	`proposal:read, report:read, report:evaluate, evaluation:create`
EVALUATEUR	`report:read, report:evaluate, evaluation:create`
ENCADRANT_ENTREPRISE	`report:read, evaluation:create (company only)`
ETUDIANT_PFE	`proposal:create, proposal:update (own), report:create (own), report:read (own)`

5.3 Sécurité des Données

Chiffrement en Transit - TLS 1.3[^20]

Configuration Nginx

```
server {
    listen 443 ssl http2;
    server_name pfe.esprit.tn;

    # Certificat Let's Encrypt
    ssl_certificate /etc/letsencrypt/live/pfe.esprit.tn/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/pfe.esprit.tn/privkey.pem;

    # TLS 1.3 uniquement
    ssl_protocols TLSv1.3;
    ssl_prefer_server_ciphers off;

    # HSTS (1 an)
    add_header Strict-Transport-Security "max-age=31536000; includeSubDomains; preload" a

    # Autres headers sécurité
    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header X-Content-Type-Options "nosniff" always;
    add_header X-XSS-Protection "1; mode=block" always;
    add_header Referrer-Policy "strict-origin-when-cross-origin" always;

    # CSP
    add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inl
}
```

Chiffrement au Repos - SQLCipher[16][17][^23]

Configuration

```
// db/connection.ts
import Database from 'better-sqlite3';
import { getEncryptionKey } from './vault'; // HashiCorp Vault ou AWS Secrets Manager

const encryptionKey = getEncryptionKey(); // Récupération sécurisée

const db = new Database('pfe.db', {
  verbose: console.log
});

// Activation SQLCipher
db.pragma(`cipher = 'aes-256-cbc'`);
db.pragma(`kdf_iter = 256000`); // PBKDF2 iterations (sécurité élevée)
db.pragma(`cipher_page_size = 4096`);
db.pragma(`cipher_hmac_algorithm = HMAC_SHA512`);
db.pragma(`cipher_kdf_algorithm = PBKDF2_HMAC_SHA512`);

// Déchiffrement base avec clé
db.pragma(`key = '${encryptionKey}'`);

// Test connexion
try {
  db.prepare('SELECT count(*) FROM sqlite_master').get();
  console.log('✅ SQLCipher initialized successfully');
} catch (error) {
  console.error('❌ Failed to decrypt database - Invalid key');
  process.exit(1);
}

export default db;
```

Gestion des Clés de Chiffrement

- **Production** : HashiCorp Vault ou AWS Secrets Manager
- **Développement** : Variables d'environnement (.env.local, gitignored)
- **Rotation** : Annuelle (procédure migration vers nouvelle base chiffrée)
- **Backup clés** : Stockage offline sécurisé (coffre physique)

Chiffrement Champs Sensibles

```
// Champs additionnellement chiffrés en base
// (double couche : SQLCipher + chiffrement applicatif)
import crypto from 'crypto';

const algorithm = 'aes-256-gcm';
const FIELD_ENCRYPTION_KEY = process.env.FIELD_ENCRYPTION_KEY; // 32 bytes hex

export function encryptField(text: string): string {
  const iv = crypto.randomBytes(16);
  const cipher = crypto.createCipheriv(algorithm, Buffer.from(FIELD_ENCRYPTION_KEY, 'hex'

  let encrypted = cipher.update(text, 'utf8', 'hex');
  encrypted += cipher.final('hex');
```

```
  const authTag = cipher.getAuthTag().toString('hex');

  return `${iv.toString('hex')}:${authTag}:${encrypted}`;
}

export function decryptField(encrypted: string): string {
  const parts = encrypted.split(':');
  const iv = Buffer.from(parts[^0], 'hex');
  const authTag = Buffer.from(parts[^1], 'hex');
  const encryptedText = parts[^2];

  const decipher = crypto.createDecipheriv(algorithm, Buffer.from(FIELD_ENCRYPTION_KEY, '
  decipher.setAuthTag(authTag);

  let decrypted = decipher.update(encryptedText, 'hex', 'utf8');
  decrypted += decipher.final('utf8');

  return decrypted;
}

// Utilisation sur champs ultra-sensibles (optionnel)
// - Numéros téléphone personnels
// - Adresses domicile
// - Informations médicales (si applicable)
```

Protection des Documents

Watermarking Automatique

```
// services/watermark.ts
import PDFDocument from 'pdfkit';
import fs from 'fs';

export async function addWatermark(
  inputPath: string,
  outputPath: string,
  userId: string,
  userName: string
): Promise<void> {
  const watermarkText = `${userName} (ID: ${userId}) - ${new Date().toISOString()}`;

  // Utilisation PDFKit ou pdf-lib pour overlay watermark
  // - Texte diagonal transparent sur chaque page
  // - Footer avec métadonnées utilisateur
  // - QR code traçabilité (optionnel)

  // Implémentation simplifiée
  const doc = new PDFDocument();
  const stream = fs.createWriteStream(outputPath);

  doc.pipe(stream);

  // Charge PDF original et ajoute watermark
  // (nécessite pdf-lib pour manipulation PDF existant)
```

```
  doc.fontSize(10)
     .fillColor('#999999', 0.3)
     .text(watermarkText, 50, 800, {
       align: 'center',
       rotate: -45
     });

  doc.end();
}
```

URLs Signées (Téléchargement Sécurisé)

```
// services/signed-url.ts
import jwt from 'jsonwebtoken';

const URL_SIGNING_SECRET = process.env.URL_SIGNING_SECRET;

export function generateSignedUrl(
  resourceId: string,
  userId: string,
  expiresIn: number = 1800 // 30 minutes
): string {
  const token = jwt.sign(
    {
      resourceId,
      userId,
      action: 'download'
    },
    URL_SIGNING_SECRET,
    { expiresIn }
  );

  return `/api/files/download/${resourceId}?token=${token}`;
}

export function verifySignedUrl(
  token: string,
  resourceId: string,
  userId: string
): boolean {
  try {
    const decoded = jwt.verify(token, URL_SIGNING_SECRET) as any;

    return (
      decoded.resourceId === resourceId &&
      decoded.userId === userId &&
      decoded.action === 'download'
    );
  } catch (error) {
    return false;
  }
}

// Middleware route téléchargement
```

```
router.get('/files/download/:id', authenticate, async (req, res) => {
  const { id } = req.params;
  const { token } = req.query;

  if (!verifySignedUrl(token, id, req.user.id)) {
    return res.status(403).json({ error: 'URL invalide ou expirée' });
  }

  // Vérification permissions additionnelles (RBAC)
  const hasAccess = await checkFileAccess(req.user, id);
  if (!hasAccess) {
    return res.status(403).json({ error: 'Accès refusé' });
  }

  // Watermark si PDF
  const file = await getFile(id);
  if (file.mimeType === 'application/pdf') {
    const watermarkedPath = await addWatermark(
      file.path,
      `/tmp/${file.id}_watermarked.pdf`,
      req.user.id,
      req.user.fullName
    );

    res.download(watermarkedPath);
  } else {
    res.download(file.path);
  }

  // Log téléchargement (audit)
  await logAudit({
    userId: req.user.id,
    action: 'FILE_DOWNLOAD',
    entityId: id,
    ipAddress: req.ip
  });
});
```

5.4 Protection contre les Attaques Web[^20]

Cross-Site Request Forgery (CSRF)

```
// middleware/csrf.ts
import csrf from 'csurf';

// Protection CSRF pour routes modifiantes
const csrfProtection = csrf({
  cookie: {
    httpOnly: true,
    secure: process.env.NODE_ENV === 'production',
    sameSite: 'strict'
  }
});

// Route génération token CSRF (GET initial page)
```

```
app.get('/api/csrf-token', csrfProtection, (req, res) => {
  res.json({ csrfToken: req.csrfToken() });
});

// Application sur toutes routes POST/PUT/DELETE
app.use('/api', (req, res, next) => {
  if (['POST', 'PUT', 'DELETE'].includes(req.method)) {
    return csrfProtection(req, res, next);
  }
  next();
});
```

Cross-Site Scripting (XSS)

```
// Sanitisation entrées utilisateur
import DOMPurify from 'isomorphic-dompurify';

export function sanitizeInput(input: string): string {
  return DOMPurify.sanitize(input, {
    ALLOWED_TAGS: [], // Aucune balise HTML autorisée
    ALLOWED_ATTR: []
  });
}

// Validation schéma Zod avec transform
const proposalSchema = z.object({
  title: z.string()
    .min(10, 'Titre trop court')
    .max(200, 'Titre trop long')
    .transform(sanitizeInput),
  description: z.string()
    .min(200, 'Description insuffisante')
    .transform(sanitizeInput),
  // ...
});

// Frontend React : échappement automatique JSX
// Dangereux : <div dangerouslySetInnerHTML={{__html: userInput}} />
// Sûr : <div>{userInput}</div> (échappement auto)
```

SQL Injection

```
// ORM Prisma : requêtes préparées automatiques (safe)
const user = await prisma.user.findUnique({
  where: { email: userProvidedEmail } // Paramétrisé automatiquement
});

// JAMAIS de requêtes SQL brutes avec concaténation
// ✖ DANGEREUX
const query = `SELECT * FROM users WHERE email = '${email}'`;

// ✓ SI SQL brut nécessaire : paramètres bindés
```

```
const query = `SELECT * FROM users WHERE email = ?`;
const result = db.prepare(query).get(email);
```

Path Traversal

```
// Validation noms fichiers
import path from 'path';

export function sanitizeFilename(filename: string): string {
  // Retire caractères dangereux
  const sanitized = filename.replace(/[^a-zA-Z0-9_\-\.]/g, '_');

  // Empêche path traversal (../, ../../, etc.)
  const parsed = path.parse(sanitized);

  if (parsed.dir !== '' || sanitized.includes('..')) {
    throw new Error('Nom de fichier invalide');
  }

  return sanitized;
}

// Utilisation upload
app.post('/upload', upload.single('file'), (req, res) => {
  const originalName = req.file.originalname;
  const safeName = sanitizeFilename(originalName);
  const finalPath = path.join(UPLOAD_DIR, req.user.id, safeName);

  // Vérification finale : chemin absolu dans répertoire autorisé
  const resolvedPath = path.resolve(finalPath);
  const allowedDir = path.resolve(UPLOAD_DIR);

  if (!resolvedPath.startsWith(allowedDir)) {
    throw new Error('Path traversal détecté');
  }

  // Déplacement fichier sécurisé
  fs.renameSync(req.file.path, resolvedPath);
});
```

Rate Limiting & DDoS Protection[^20]

```
// middleware/rate-limit.ts
import rateLimit from 'express-rate-limit';
import RedisStore from 'rate-limit-redis';
import Redis from 'ioredis';

const redis = new Redis(process.env.REDIS_URL);

// Rate limiters différenciés par endpoint

// Authentification : strict
export const authLimiter = rateLimit({
  store: new RedisStore({
```

```
    client: redis,
    prefix: 'rl:auth:'
  }),
  windowMs: 15 * 60 * 1000, // 15 minutes
  max: 5, // 5 tentatives
  message: 'Trop de tentatives de connexion. Réessayez dans 15 minutes.',
  standardHeaders: true,
  legacyHeaders: false,
  skipSuccessfulRequests: false, // Compter même succès (anti-brute-force)
  keyGenerator: (req) => {
    // Rate limit par IP + email combinés
    return `${req.ip}:${req.body.email || 'unknown'}`;
  }
});

// API générale
export const apiLimiter = rateLimit({
  store: new RedisStore({
    client: redis,
    prefix: 'rl:api:'
  }),
  windowMs: 60 * 1000, // 1 minute
  max: 100, // 100 requêtes/min
  message: 'Trop de requêtes. Réessayez dans 1 minute.',
  skip: (req) => {
    // Whitelist IPs internes/admin
    const whitelistIPs = process.env.WHITELIST_IPS?.split(',') || [];
    return whitelistIPs.includes(req.ip);
  }
});

// Upload fichiers : très restrictif
export const uploadLimiter = rateLimit({
  store: new RedisStore({
    client: redis,
    prefix: 'rl:upload:'
  }),
  windowMs: 60 * 60 * 1000, // 1 heure
  max: 20, // 20 uploads/heure
  message: 'Limite uploads atteinte. Réessayez dans 1 heure.'
});

// Application
app.post('/api/auth/login', authLimiter, login);
app.use('/api', apiLimiter);
app.post('/api/reports/upload', uploadLimiter, uploadReport);
```

CAPTCHA Adaptatif

```
// Intégration hCaptcha ou Google reCAPTCHA v3

// Déclenchement si :
// - 3+ tentatives login échouées
// - Pattern suspect (trop rapide, user-agent bot)
// - IP flaggée
```

```
import axios from 'axios';

export async function verifyCaptcha(token: string, ip: string): Promise<boolean> {
  try {
    const response = await axios.post(
      'https://hcaptcha.com/siteverify',
      {
        secret: process.env.HCAPTCHA_SECRET,
        response: token,
        remoteip: ip
      }
    );

    return response.data.success;
  } catch (error) {
    return false;
  }
}

// Middleware
export const requireCaptcha = async (req, res, next) => {
  // Vérifier si captcha nécessaire (Redis flag)
  const requiresCaptcha = await redis.get(`captcha_required:${req.ip}`);

  if (requiresCaptcha) {
    const captchaToken = req.body.captchaToken;

    if (!captchaToken) {
      return res.status(400).json({
        error: 'Captcha requis',
        requiresCaptcha: true
      });
    }

    const valid = await verifyCaptcha(captchaToken, req.ip);

    if (!valid) {
      return res.status(400).json({ error: 'Captcha invalide' });
    }

    // Reset flag après succès
    await redis.del(`captcha_required:${req.ip}`);
  }

  next();
};
```

5.5 Audit et Monitoring[22][21]

Logs de Sécurité Complets

```
// services/audit.ts
import winston from 'winston';
```

```
const auditLogger = winston.createLogger({
  level: 'info',
  format: winston.format.combine(
    winston.format.timestamp(),
    winston.format.json()
  ),
  transports: [
    // Fichier logs sécurité
    new winston.transports.File({
      filename: 'logs/security.log',
      maxsize: 10485760, // 10MB
      maxFiles: 30,
      tailable: true
    }),
    // Console en dev
    ...(process.env.NODE_ENV === 'development'
      ? [new winston.transports.Console()]
      : [])
  ]
});

export async function logAudit(event: {
  userId?: string;
  action: string;
  entity?: string;
  entityId?: string;
  changes?: any;
  ipAddress: string;
  userAgent: string;
  success: boolean;
  errorMessage?: string;
}) {
  // Log Winston
  auditLogger.info({
    timestamp: new Date().toISOString(),
    ...event
  });

  // Stockage base de données (requêtes ultérieures)
  await prisma.auditLog.create({
    data: {
      userId: event.userId,
      action: event.action,
      entity: event.entity,
      entityId: event.entityId,
      changes: event.changes,
      ipAddress: event.ipAddress,
      userAgent: event.userAgent,
      success: event.success,
      errorMessage: event.errorMessage
    }
  });

  // Alerte temps réel si événement critique
  if (CRITICAL_ACTIONS.includes(event.action)) {
    await sendSecurityAlert(event);
```

```
  }
}

// Middleware automatique
app.use((req, res, next) => {
  // Intercepter réponse pour logging
  const originalSend = res.send;

  res.send = function(data) {
    if (['POST', 'PUT', 'DELETE'].includes(req.method)) {
      logAudit({
        userId: req.user?.id,
        action: `${req.method} ${req.path}`,
        ipAddress: req.ip,
        userAgent: req.get('user-agent'),
        success: res.statusCode < 400
      });
    }

    return originalSend.call(this, data);
  };

  next();
});
```

Événements Loggés Systématiquement[22][21]

Événement	Détails Enregistrés
Connexion réussie	userId, IP, user-agent, timestamp, 2FA utilisé
Connexion échouée	email tenté, IP, raison échec, timestamp
Logout	userId, IP, durée session
Modification permissions	userId auteur, userId modifié, anciennes/nouvelles permissions
Affectation rôle	userId auteur, userId affecté, ancien/nouveau rôle
Accès données sensibles	userId, type ressource, resourceId, action
Téléchargement document	userId, documentId, IP, timestamp
Modification paramètres système	userId auteur, paramètre modifié, ancienne/nouvelle valeur
Détection plagiat >50%	reportId, score, studentId, timestamp
Tentative accès non autorisé	userId, ressource tentée, permissions manquantes
Upload fichier suspect	userId, filename, raison rejet (malware/format)
Modification base données	Table, operation (INSERT/UPDATE/DELETE), recordId, changes

Détection d'Anomalies

```
// services/anomaly-detection.ts

// Règles basiques (Phase 1)
```

```
export async function detectAnomalies(userId: string): Promise<Alert[]> {
  const alerts: Alert[] = [];

  // 1. Connexions depuis nouveaux pays/villes
  const recentLogins = await getRecentLogins(userId, 7); // 7 derniers jours
  const currentLogin = recentLogins[^0];
  const previousLocations = recentLogins.slice(1).map(l => l.location);

  if (!previous
<span style="display:none">[^10][^3][^5][^6][^8][^9]</span>

<div align="center">⁂</div>

[^1]: https://fr.scribd.com/document/638262201/Cahier-de-Charge-1
[^2]: https://www.sagemcom.com/sites/default/files/TN-Ind/Catalogue_PFE_2024_Ezzahra.pdf
[^3]: https://fss.rnu.tn/useruploads/files/Service%20Stages/2024-2025/STAGES-PFE-BOOK-202
[^4]: https://blog-gestion-de-projet.com/cahier-des-charges-projet/
[^5]: https://fr.scribd.com/document/868567501/PFE-2025
[^6]: https://portail.isikef.tn/OpenFiles/Catalogue_PFE_Licence_2025.pdf
[^7]: https://www.studocu.com/row/document/the-institut-superieur-des-etudes-technologiqu
[^8]: https://www.manager-go.com/gestion-de-projet/dossiers-methodes/elaborer-un-cdc
[^9]: http://www.anpr.tn/cahier-des-charges-de-la-plateforme-de-gestion-mobidoc/
[^10]: https://esen.rnu.tn/portail/avis-esen/au-20242025-l3m2-lancement-de-la-plateforme-
[^11]: https://www.esprit.tn
[^12]: https://www.esprit.tn/groupe-esprit/esprit-monastir-esprim/
[^13]: https://www.gdpr-advisor.com/gdpr-compliance-for-educational-institutions/
[^14]: https://www.linkedin.com/pulse/implementing-secure-authentication-mechanisms-best-
[^15]: https://dev.to/wesleyisr4/how-to-create-two-factor-authentication-2fa-and-best-pra
[^16]: https://www.tencentcloud.com/techpedia/102689
[^17]: https://www.sqliteforum.com/p/securing-your-sqlite-database-best
[^18]: https://www.ijrti.org/papers/IJRTI2505044.pdf
[^19]: https://ijrehc.com/uploads2025/ijrehc06_147.pdf
[^20]: https://tvm.offensoacademy.com/best-practices-for-cybersecurity-2025-guide/
[^21]: http://www.aacc.edu/policies/technology-monitoring-and-audit-logging/
[^22]: https://security.georgetown.edu/technology-audit-logging-guidelines/
[^23]: https://dev.to/stephenc222/basic-security-practices-for-sqlite-safeguarding-your-d
```