

Référentiel de la Stack Technique et de la Gouvernance du Projet HRComplianceTech

1. Introduction et Objectifs du Document	1
1.1 Rôle du Référentiel Technique	2
1.2 Périmètre et Alignement avec le Cahier des Charges (CdC).....	2
1.3 Règles de Mise à Jour du Document	3
2. Architecture de la Solution et Flux de Données	3
2.1 Schéma Architectural Global (Modules 1, 2, 3).....	4
2.2 Zones de Sécurité et Confidentialité.....	4
2.3 Flux d'Information entre les Modules (API)	5
Flux 1 : Dépôt d'un Signalement Anonyme (F.1.4)	5
Flux 2 : Consultation/Modification (RH/Juriste).....	5
Flux 3 : Gestion Offline (F.2.7)	6
3. Stack de Développement 	6
3.1 Module 1 : Application de Dépôt (Webapp)	7
3.2 Module 2 : Application de Gestion (Client Lourd - Windev)	8
3.3 Module 3 : API Core et Services de Sécurité	8
4. Stack de Données et Services Critiques 	10
4.1 Base de Données Relationnelle (SGBD)	11
4.2 Stockage de Fichiers Chiffrés (Pièces Jointes)	11
5. Outils et Méthodologie 	12
5.1 Gestion de Version (Git)	13
5.3 Environnement de Développement	13
6. Récapitulatif des Versions et des Licences	14
6.1 Tableau Récapitulatif des Technologies et Versions Stables.....	15
6.2 Contrainte de Licences (C.8.3.5).....	17
6.3 Synthèse des Outils Open Source	17

1. Introduction et Objectifs du Document

1.1 Rôle du Référentiel Technique

Ce document a pour objectif principal de **servir de guide normatif** pour l'ensemble des développeurs de la **Maitrise d'Œuvre (MOE)** (Baptiste et Maxime) tout au long du cycle de vie du projet.

Son rôle est triple :

- **Assurer l'Homogénéité** : Définir une **Stack Technique (ensemble d'outils et de technologies)** unique et validée. Ceci garantit que toutes les parties du projet sont développées avec les mêmes standards et spécifications.
- **Contrôler la Conformité des Versions** : Éviter les **conflits de versions** et les problèmes d'incompatibilité entre les dépendances. Il sert de **base claire** pour l'installation des paquets et l'environnement de développement.
- **Faciliter l'Intégration** : Fournir les bases documentaires nécessaires pour les phases d'intégration et pour les développeurs qui rejoindraient le projet ultérieurement.

1.2 Périmètre et Alignement avec le Cahier des Charges (CdC)

Ce référentiel technique couvre la totalité des composants de la solution, conformément aux exigences du CdC :

1. **Module 1 : Application de Dépôt** (Client Lourd - Windev)
2. **Module 2 : Application de Gestion** (Webapp - React)
3. **Module 3 : API Core et Services de Sécurité** (Python/Flask)
4. **Base de Données Centrale** (MySQL)

Alignement Critique avec le CdC :

Le choix des outils et de leurs versions, détaillé dans ce document, est directement motivé par la nécessité d'honorer les contraintes légales et techniques du Cahier des Charges. Ce référentiel sert de preuve de la capacité à respecter notamment :

Référence CdC	Exigence	Impact de la Stack Technique
---------------	----------	------------------------------

C.8.3.1	Chiffrement Fort (AES-256)	L'API Python/Flask doit intégrer des librairies cryptographiques spécifiques pour le chiffrement des données sensibles.
C.8.3.4	Conformité Juridique (Sapin 2/RGPD)	Le choix de MySQL doit supporter les fonctions d'audit et de traçabilité nécessaires à la conformité (gestion des logs inaltérables).
C.8.3.2	Exigence OS Client Lourd	Le choix de Windev est lié à l'environnement d'exécution Windows 10 Pro ou supérieur exigé.

Les technologies sélectionnées et non négociables pour ce projet sont : **Python/Flask** (API), **MySQL** (Base de données), **React** (Frontend) et **Windev** (Client Lourd).

1.3 Règles de Mise à Jour du Document

Le maintien de ce référentiel est crucial pour la qualité du produit final. Les règles de gouvernance sont les suivantes :

- Responsables de la Validation** : La validation finale du document et de toute modification est la responsabilité de **Baptiste et Maxime** (l'équipe MOE).
- Déclenchement de la Mise à Jour** : Une révision et une mise à jour du document seront obligatoires **lors de la rencontre de conflits de versions** ou après la gestion d'un incident technique majeur lié à l'incompatibilité d'un outil.
- Procédure en Cas de Déviation** : Si un outil ou une version spécifiée s'avère impossible à utiliser (problème technique, arrêt de support, etc.), la procédure est d'abord la **recherche d'une solution alternative** immédiate. Cette solution doit ensuite faire l'objet d'une **mise à jour formelle et argumentée de ce document** avant d'être déployée sur l'ensemble de l'environnement de développement.

2. Architecture de la Solution et Flux de Données

2.1 Schéma Architectural Global (Modules 1, 2, 3)

L'architecture est structurée selon un modèle **3-Tiers (ou Client-Serveur étendu)** pour garantir la centralisation des règles métier et, surtout, des mécanismes de sécurité critique.

Module	Rôle Architectural	Technologie(s)
Tiers 1 : Clients (Modules 1 & 2)	Interfaces Utilisateur et gestion locale (Offline Windev).	React (Webapp) et Windev (Client Lourd)
Tiers 2 : API Core (Module 3)	Unique DB Layer et Moteur de Sécurité. Gère l'authentification, le chiffrement/déchiffrement, les règles métier, la traçabilité et l'intégration de la Pseudo-IA.	Python / Flask (avec ORM ou connecteur Python vers MySQL)
Tiers 3 : Données	Stockage persistant et sécurisé.	MySQL (Données) et Stockage Objet/S3 (Fichiers)

Principe Fondamental : Les Tiers 1 (Clients) n'interagissent **JAMAIS** directement avec le Tiers 3 (Données). **Le Module 3 (API Python/Flask) est l'unique couche d'accès aux données (DB Layer)**, éliminant toute exposition directe de MySQL aux clients.

Rôle de la Pseudo-IA (F.3.3) : La fonction de Pseudo-IA est intégrée au sein du **Module 3**. Elle est déclenchée par l'API après l'enregistrement d'un nouveau signalement pour proposer une `categorie_detectee` et enregistrer le processus dans la table `ia_classifications`.

2.2 Zones de Sécurité et Confidentialité

La solution est construite autour de la **centralisation de la confiance** dans le Module 3, essentielle pour satisfaire le **Chiffrement Fort (C.8.3.1)** et la **Conformité Juridique (C.8.3.4)**.

- **API Core (Module 3) - La Zone de Confiance :**
 - Contient les **clés secrètes** (stockées dans l'environnement local sécurisé) nécessaires au chiffrement/déchiffrement **AES-256**.
 - Détient les identifiants de connexion **MySQL** et au **Stockage Objet/S3**.
- **Base de Données (MySQL) - Zone de Stockage Chiffré :**
 - Les données sensibles (description, lieu, etc.) sont stockées exclusivement en format **chiffré au repos** (BLOB).
 - L'API est le seul utilisateur autorisé.
- **Authentification (C.8.3.3) :**
 - Gérée de bout en bout par l'API Core, par vérification des mots de passe hachés dans la table users.

2.3 Flux d'Information entre les Modules (API)

Toutes les communications entre Tiers 1 et Tiers 2 s'effectuent via des appels **HTTPS/REST** pour garantir l'intégrité des messages.

Flux 1 : Dépôt d'un Signalement Anonyme (F.1.4)

1. **Client (React/Windev)** : Saisie des données non chiffrées par le déposant.
2. **Client → API** : Envoi des données (claires) au Module 3 via HTTPS.
3. **API (Module 3) :**
 - a. **Chiffrement (AES-256)** des données sensibles (Corps du signalement, adresse e-mail si non anonyme).
 - b. Enregistrement des données chiffrées dans MySQL.
 - c. Enregistrement des pièces jointes chiffrées vers le Stockage Objet (S3).
 - d. Génération et enregistrement du `tracking_code` et du `tracking_password_hash`.
 - e. Déclenchement du service de **Pseudo-IA** (classification).
4. **API → Client** : Retour du `tracking_code` au déposant.

Flux 2 : Consultation/Modification (RH/Juriste)

1. **Client (Windev/React)** : Authentification de l'utilisateur.

2. **Client → API** : Demande de consultation d'un dossier.

3. **API (Module 3)** :

- a. Vérification de l'Authentification et des Permissions (RBAC).
- b. Récupération des données chiffrées de MySQL.
- c. **Déchiffrement** des données dans l'environnement Python.
- d. **Enregistrement d'Audit** : L'action de lecture est loguée immédiatement dans audit_logs par l'API, avec le chaînage cryptographique (previous_hash).

4. **API → Client** : Retour des données **en clair** au client pour affichage.

Flux 3 : Gestion Offline (F.2.7)

1. **Client Lourd (Windev)** : Hors ligne, le client enregistre les modifications (statut, notes, etc.) dans une base de données locale (buffer).

2. **Client Lourd → API** : Dès la reconnexion, le Client Lourd envoie l'intégralité de son buffer à l'API Core.

3. **API (Module 3)** :

- a. Traitement séquentiel des actions contenues dans le buffer.
- b. Mise à jour des tables MySQL.
- c. **Traçabilité** : Enregistrement de chaque action de synchronisation dans audit_logs par l'API, garantissant l'intégrité de l'historique malgré le mode hors ligne.

3. Stack de Développement

Cette section liste l'ensemble des technologies retenues pour le développement des trois modules, garantissant l'homogénéité technique de la **Maitrise d'Œuvre (MOE)**.

3.1 Module 1 : Application de Dépôt (Webapp)

Ce module correspond à l'interface de gestion utilisée par les RH/Juristes pour consulter et traiter les signalements.

Domaine	Technologie	Version Cible	Rôle et Justification
3.1.1 Frontend Framework	React	18.x	Framework leader, assurant l'évolutivité.
3.1.2 Langage de Base	TypeScript	5.x	Typage fort pour la réduction des erreurs et la maintenance du code.
Design / CSS	Tailwind CSS	Dernière version	Framework utilitaire CSS pour un développement rapide et la garantie d'une interface responsive et homogène.
Validation des Données	Zod	Dernière version	Librairie de validation de schéma. Permet de valider la conformité des données des formulaires côté client avant l'envoi à l'API.
Requêtes HTTP	Axios	Dernière version	Client HTTP performant. Nécessaire pour la gestion simple des erreurs HTTP et la mise en place des intercepteurs d'authentification (tokens JWT).

Communication	HTTPS/REST	N/A	Interagit exclusivement avec le Module 3 (API Core).
----------------------	-------------------	-----	---

3.2 Module 2 : Application de Gestion (Client Lourd - Windev)

Ce module correspond au client lourd utilisé par les employés (Salariés) pour déposer un signalement, avec la capacité de fonctionner hors ligne.

Domaine	Technologie	Version Cible	Rôle et Justification
3.2.1 Environnement de Développement	Windev	Version 28 ou 29	Choix imposé par les exigences client pour la distribution d'une application de bureau.
3.2.2 Langage de Programmation	WLangage	N/A	Langage propriétaire de l'environnement Windev, utilisé pour la logique métier spécifique au client lourd.
Justification Clé	Windows 10 Pro	N/A	Le choix Windev répond directement à la Contrainte C.8.3.2 (Exigence OS Client Lourd) et permet la gestion des actions en Mode Offline (F.2.7) via une base de données locale (SQLite ou HyperFileSQL).

3.3 Module 3 : API Core et Services de Sécurité

Ce module est le **Cœur de la Solution et Unique DB Layer** (Tiers 2), responsable de la sécurité, de la traçabilité et de la logique métier.

Domaine	Technologie	Version Cible	Rôle et Justification
3.3.1 Langage de Programmation	Python	3.10 ou supérieur	Langage polyvalent, excellent pour les backends, le traitement de données (Pseudo-IA) et possédant des bibliothèques cryptographiques robustes.
3.3.2 Framework API	Flask	2.x	Micro-framework léger et flexible. Il est idéal pour construire une API minimaliste et performante, centrée sur les fonctions de sécurité, d'audit, et l'exposition de points d'accès (endpoints) REST.
ORM / DB Connector	SQLAlchemy	2.x	ORM Python standard, choisi pour interagir avec la base de données MySQL de manière sécurisée (seul outil autorisé à se connecter au Tiers 3).
3.3.3 Librairie de Chiffrement	cryptography	Dernière version stable	Critique : Cette bibliothèque est le standard <i>de facto</i> pour la cryptographie en Python. Elle est requise pour implémenter l'algorithme AES-256 avec salage , répondant ainsi à l'exigence de Chiffrement Fort (C.8.3.1).
3.3.4 Logique IA	Scikit-learn / Librairies Python natives	N/A	Permet de construire le module de Pseudo-IA (F.3.3) pour la classification des signalements sans dépendre de services cloud externes payants.

4. Stack de Données et Services Critiques

Cette section définit l'infrastructure de stockage, qui est le cœur de la **Confidentialité (C.8.3.1)** et de la **Conformité Juridique (C.8.3.4)** du projet. L'architecture sépare clairement les données structurées (MySQL) des données non structurées (Pièces Jointes).

4.1 Base de Données Relationnelle (SGBD)

Domaine	Technologie	Justification
4.1.1 Moteur de Base de Données	MySQL	Moteur open-source, robuste, et mature. Choisi pour sa performance nécessaire aux opérations de lecture/écriture fréquentes (audit log) et sa large compatibilité avec l'API Python.
4.1.2 Fournisseur d'Hébergement	Aiven.io (Choix Arrêté)	Solution DBaaS (Database-as-a-Service) Managée. Ce choix est critique car il délègue la complexité de l'administration, des patchs de sécurité et des sauvegardes automatiques. Aiven.io est préféré pour : <ul style="list-style-type: none">• La simplicité de mise en œuvre de la haute disponibilité.• Le support des exigences de RTO & PRA (C.8.4.1).
4.1.3 Outil de Modélisation	dbdiagram.io	Outil simple et rapide pour générer le schéma de base de données à partir du langage DBML. Utilisé pour la documentation et le partage du modèle.

Rôle Sécuritaire du SGBD : La base de données est structurée pour supporter la traçabilité (**audit_logs avec chaînage de hashs**) et le stockage de l'information la plus critique en format chiffré (BLOB), garantissant le respect du **RGPD** et de la **Loi Sapin 2**.

4.2 Stockage de Fichiers Chiffrés (Pièces Jointes)

Les pièces jointes sont des données hautement sensibles. Un service de stockage objet sera utilisé pour la scalabilité, la sécurité et la séparation physique des données (C.8.3.1).

Domaine	Technologie	Justification
4.2.1 Service de Stockage Objet	Cloudflare R2 (Recommandé)	Service de stockage objet compatible S3 . Choisi pour son excellent plan gratuit et surtout l'absence de frais de sortie (<i>egress fees</i>), optimisant les coûts de développement. Il assure la séparation physique des fichiers chiffrés.
Alternative	AWS S3 (Free Tier)	Alternative standard du marché si R2 n'est pas accessible.
4.2.2 Librairie de Connexion	boto3 (Python)	Librairie officielle Python pour AWS/S3 . Elle sera utilisée par le Module 3 (API Python) pour gérer la connexion sécurisée, l'upload et le téléchargement des fichiers chiffrés.

Rôle Sécuritaire du Stockage Objet : Le **Module 3 (API Python)** est responsable de chiffrer le fichier avec AES-256 **avant** de l'envoyer au stockage objet, qui ne stocke alors que des données binaires chiffrées (NF.4.1.1)

5. Outils et Méthodologie

Cette section détaille les outils de support et la méthodologie de travail adoptée par la MOE pour garantir l'efficacité, la collaboration et la qualité du code.

5.1 Gestion de Version (Git)

Domaine	Technologie	Justification
5.1.1 Plateforme	GitHub	Plateforme leader pour l'hébergement de dépôts Git. Offre des fonctionnalités essentielles (revues de code, gestion des <i>issues</i> , Actions pour l'intégration continue) qui améliorent la qualité et la collaboration.
5.1.2 Stratégie de Branching	Gitflow Simplifié	Cette stratégie garantit la stabilité des versions livrables tout en facilitant le travail en parallèle : <ul style="list-style-type: none">• main : La branche stable, correspondant à la version actuellement en production.• develop : La branche d'intégration principale où toutes les fonctionnalités sont fusionnées et testées (pré-production).• feature/* : Création d'une branche spécifique pour chaque nouvelle fonctionnalité ou tâche (isolation du travail).

5.3 Environnement de Développement

Cette standardisation assure que les développeurs de la MOE travaillent dans un environnement de travail identique pour chaque module.

Domaine	Technologie	Module(s) Concerné(s)	Justification
5.3.1 IDE Principal	Visual Studio Code (VS Code)	Python (API) et React (Webapp)	IDE léger, performant et supportant nativement le

			développement en Python et TypeScript/JavaScript .
IDE Spécifique	Windev Environnement	Windev (Client Lourd)	Environnement de développement requis pour le WLanguage et la compilation du Client Lourd (Module 2).
5.3.2 Gestionnaire de Dépendances	npm	React (Webapp)	Gestionnaire de paquets standard pour l'écosystème Node.js/JavaScript (React, Tailwind, Axios, Zod).
Gestionnaire de Dépendances	pip	Python (API)	Gestionnaire de paquets standard pour l'environnement Python. Utilisé pour gérer les dépendances critiques de l'API (Flask, SQLAlchemy, cryptography).

6. Récapitulatif des Versions et des Licences

6.1 Tableau Récapitulatif des Technologies et Versions Stables

Ce tableau synthétise l'ensemble des technologies retenues, classées par module architectural, et sert de référence unique pour la MOE

Composant	Module(s)	Technologie	Version Cible	Rôle Critique / Justification
API Core (DB Layer)	Module 3	Python	3.11.x	Langage pour l'API Core et le moteur de chiffrement.
Framework API	Module 3	Flask	2.x	Micro-framework pour les endpoints REST et la sécurité.
ORM / DB Layer	Module 3	SQLAlchemy	2.x	ORM Python, unique outil d'accès à MySQL (Tiers 3).
Chiffrement Fort	Module 3	cryptography	Dernière stable	CRITIQUE (C.8.3.1) : Implémentation AES-256 avec salage.
Stockage Fichiers	Module 3	boto3	Dernière stable	Librairie Python pour la connexion sécurisée à Cloudflare R2 (API S3).
Pseudo-IA	Module 3	Scikit-learn	Dernière stable	Librairie Python pour la classification automatique (F.3.3).
Webapp Frontend	Module 1	React	18.x	Framework d'interface utilisateur pour l'application de gestion RH/Juriste.

Langage Client	Module 1	TypeScript	5.x	Typage fort pour la robustesse et la maintenance du code.
Requêtes HTTP	Module 1	Axios	Dernière stable	Client HTTP performant (gestion des Intercepteurs pour l'authentification).
Validation Forms	Module 1	Zod	Dernière stable	Validation des schémas de données côté client avant envoi à l'API.
Design/CSS	Module 1	Tailwind CSS	Dernière stable	Framework CSS utilitaire pour le design rapide et homogène.
Client Lourd	Module 2	Windev	29 (ou récente)	Environnement pour le client Lourd Windows (C.8.3.2) et la gestion Offline (F.2.7) .
Langage Client Lourd	Module 2	WLangage	N/A	Langage natif de l'environnement Windev.
Moteur DB	Tiers 3	MySQL	8.0.35	SGBD central pour les données chiffrées et les métadonnées.
Hébergement DB	Tiers 3	Aiven.io	N/A	Fournisseur DBaaS managé (RTO/PRA).
Stockage Fichiers	Tiers 3	Cloudflare R2	N/A	Service de stockage Objet compatible S3 (séparation

				physique des PJ chiffrées).
Gestion de Version	MOE	GitHub	N/A	Plateforme d'hébergement du code source et des revues.
Gestion Projet	MOE	Notion	N/A	Outil de suivi des tâches et de la documentation.
Maquettage	MOE	Figma	N/A	Prototypage des interfaces UI/UX.
IDE	MOE	VS Code	Dernière stable	IDE principal pour Python et React.
Modélisation DB	MOE	dbdiagram.io	N/A	Outil de génération du schéma de base de données (DBML).

6.2 Contrainte de Licences (C.8.3.5)

L'utilisation de technologies soumises à licence dans le projet impose une clarification des responsabilités, conformément au point **C.8.3.5** du Cahier des Charges.

- **Logiciels Concernés** : L'outil principal concerné par cette contrainte est l'environnement de développement **Windev (PC SOFT)**.
- **Responsabilité de la MOA** : La **Maitrise d'Ouvrage (HRComplianceTech Solutions)** est l'entité unique responsable de l'acquisition, de la validité et de la couverture de toutes les licences logicielles nécessaires à la distribution légale du produit final.
- **Périmètre de la MOE** : L'équipe MOE utilise les environnements de développement sous licence mis à disposition. Elle garantit que le code produit sera conforme aux exigences de licence logicielle pour la distribution, mais n'est pas responsable des défauts de couverture de licence de la part de la MOA.

6.3 Synthèse des Outils Open Source

Les outils suivants, considérés comme essentiels, sont Open Source, ce qui réduit le risque de licence directe et les coûts associés : Python, Flask, SQLAlchemy, cryptography, React, TypeScript, Zod, Axios, Tailwind CSS, MySQL, GitHub, Notion, Figma