HISTIFIED

Rapport Technique Complet

Plateforme de vérification d'images et d'articles pour le journalisme et la recherche

Équipe :PixelPioneers25

Année : 2025

Table des matières

Table des figures

Liste des tableaux

Résumé

Ce document présente **Histified**, une plateforme full-stack de vérification destinée aux journalistes, chercheurs et plate-formes académiques. Histified combine :

— une analyse forensic d'images (métadonnées EXIF + analyses pixelaires),

— un pipeline d'extraction et d'analyse d'articles PDF (structure, citations, cohérence sémantique),

— une API REST (Express.js) et un front moderne (Next.js/React) pour l'upload et la visualisation,

— une méthodologie de scoring conservatrice (préférence à éviter les faux positifs),

— une interface utilisateur qui met en évidence verdicts, preuves et recommandations actionnables.

Le rapport couvre l'architecture, composants techniques, algorithmes, tests, jeux de données d'évaluation, exemples d'API, recommandations de déploiement et pistes d'amélioration.

1. Contexte et objectifs

1.1 Problématique

La circulation rapide d'images et d'articles erronés ou manipulés nécessite des outils automatiques permettant d'évaluer rapidement la crédibilité d'un média. Les journalistes ont besoin d'un verdict clair, argumenté et traçable — pas seulement d'une intuition.

1.2 Objectifs de Histified

— Fournir une évaluation automatisée et explicable de la véracité d'images et d'articles PDF.

— Offrir des scores combinant plusieurs critères (métadonnées, pixel analysis, qualité des sources, cohérence textuelle).

— Mettre à disposition une API et une interface réactive pour intégration dans des workflows journalistiques.

— Documenter les résultats et produire des rapports exportables (JSON, PDF résumé).

2. Résumé du système

2.1 Composants principaux

Histified se compose des modules suivants :

1. **Frontend (Next.js)** : Upload, visualisation des rapports, interface interactive.
2. **Backend (Express.js)** : API REST, orchestration des services d'analyse.
3. **Service Image Forensics** : extraction EXIF (ExifReader), analyses pixelaires (Sharp + algos custom).
4. **Service Article Verification** : extraction PDF (pdf-parse / pipeline OCR pour scanned PDFs), NLP (Natural.js / microservice spaCy), vérification des citations.
5. **Stockage (optionnel)** : MongoDB pour rapports, logs et historiques.

2.2 Diagramme d'architecture

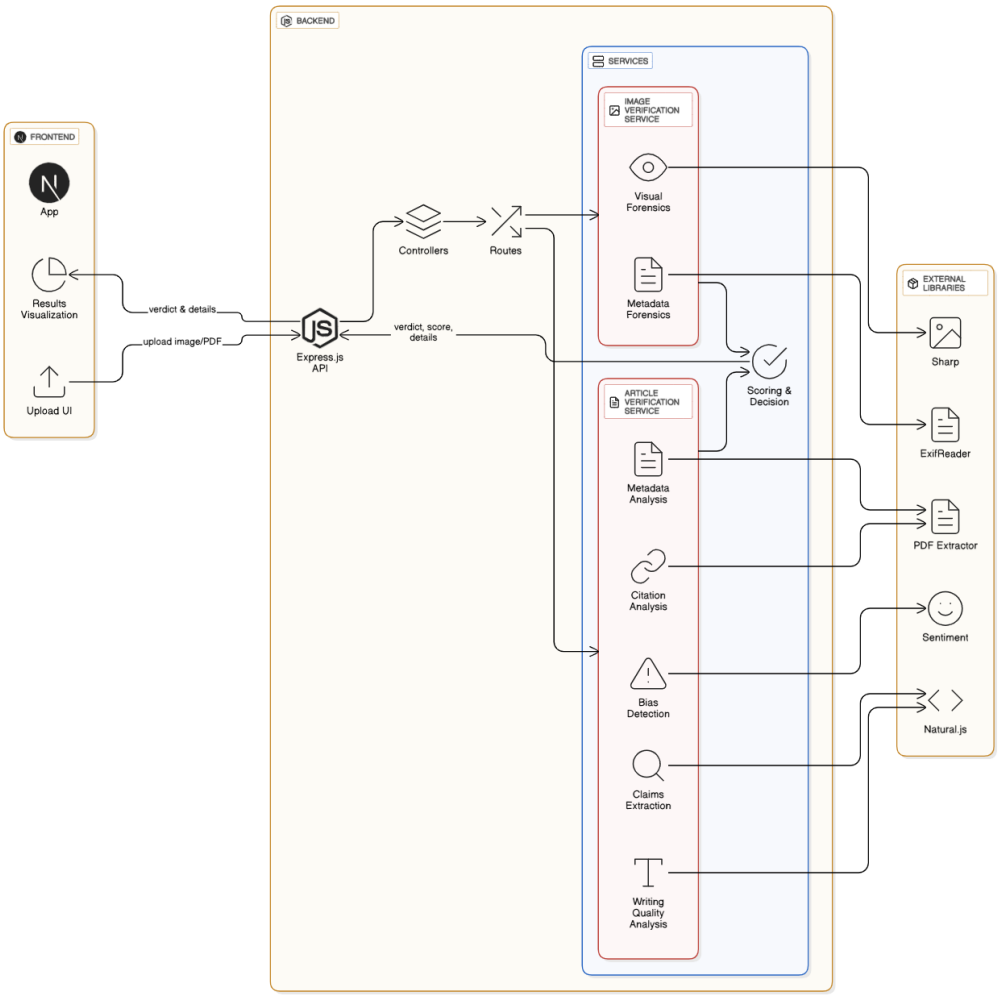

FIGURE 2.1 – Vue d'ensemble de l'architecture Histified (Frontend, API, Services d'analyse, BD).

3. Cas d'utilisation

3.1 Description générale

Historified est une plateforme full-stack de vérification de faits conçue pour les journalistes et les chercheurs. Elle permet d'évaluer rapidement :

— l'authenticité des images,

— la crédibilité des articles PDF.

Lorsqu'un utilisateur télécharge un fichier :

1. Pour une **image** :
   — Analyse des métadonnées via ExifReader pour détecter les manipulations (signatures de logiciels de retouche, incohérences de date ou d'appareil).
   — Analyse visuelle pixel par pixel via Sharp : cohérence de l'éclairage, régularité du bruit, artefacts de compression, cohérence physique (bords, netteté, échelle).
2. Pour un **article PDF** :
   — Extraction de texte et métadonnées.
   — Analyse NLP et sentiment : complétude des métadonnées, qualité des sources citées, métriques de rédaction, densité de faits, détection de biais ou langage sensationnaliste.

Chaque fichier reçoit un score pondéré sur 100, permettant de classer les résultats comme :

— **Authentique / Crédible** : score ≥ 75,

— **Suspect / Questionnable** : score 50–74,

— **Altéré / Non crédible** : score < 50.

3.2 Interface et interaction utilisateur

L'interface, développée en Next.js, offre :

— Téléversement simple et intuitif des fichiers.

— Communication en temps réel avec le backend.

— Visualisation claire des résultats : indicateurs circulaires colorés, sections détaillées, recommandations actionnables.

3.3 Diagramme de cas d'utilisation

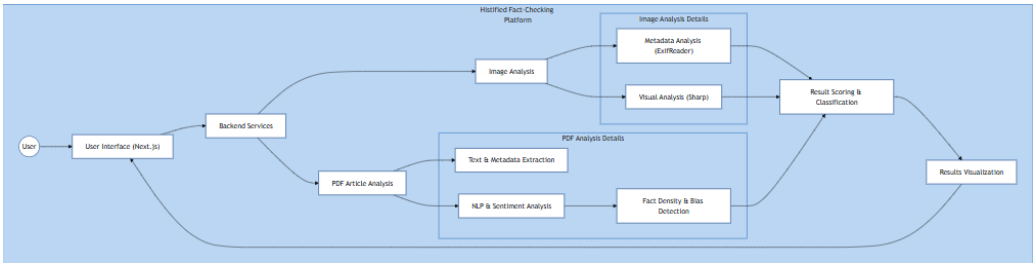

FIGURE 3.1 – Diagramme de cas d'utilisation de Historified.

Cette architecture modulaire et scalable permet une maintenance aisée, l'ajout futur de nouvelles méthodes d'analyse et l'intégration de différents types de fichiers, offrant aux professionnels des médias un outil fiable pour réduire la désinformation tout en gagnant du temps dans le processus de vérification.

4. Analyse d'images

4.1 Pipeline d'analyse d'images

L'analyse est divisée en deux couches :

1. **Forensique métadonnées** : extraction EXIF, détection d'outils d'édition, incohérences temporelles (dates de création vs date de modification), géolocalisation et présence de signatures d'éditeurs (ex. Photoshop).
2. **Forensique visuelle/pixel** : contrôles de cohérence d'éclairage, analyse de bruit, détection d'artéfacts de compression JPEG, vérification des contours/edges pour identifier composites, détection de recadrage et retouches locales (cloning).

4.1.1 Méthodes techniques

— **EXIF parsing** : ExifReader pour récupérer toutes les balises EXIF/ XMP.

— **Lighting consistency** : découpage en grille et analyse des histogrammes d'intensité pour trouver des ruptures anormales.

— **Noise pattern analysis** : estimation de la variance du bruit par région et comparaison (détecte différences de grain).

— **Compression artifacts** : détection de bordures de blocs et d'alignements chromatiques atypiques.

— **Physical coherence** : ratios d'aspect, profondeur de champ cohérente, correspondance des ombres (si possible).

4.1.2 Score et verdict

La combinaison suit un poids par composante :

Composante	Poids (points)
Métadonnées	40
Analyse visuelle (pixel-level)	60
Total	100

Verdict simplifié :

— ≥ 75 : **Probablement authentique**

— 50–74 : **Suspect / À vérifier**

— < 50 : **Altéré / Non fiable**

5. Analyse d'articles (PDF)

5.1 Pipeline d'extraction

1. Conversion texte brut via `pdf-parse` ou pipeline OCR (Tesseract) si PDF scanné.
2. Segmentation structurelle : détection des titres, sections, résumés (Abstract), figures, tables et références.
3. Extraction des métadonnées (auteurs, date, logiciel de création).
4. Extraction et vérification des références (présence de DOI / URL, domaines crédibles).
5. Analyse linguistique (tokenisation, longueur de phrase, lisibilité, détection répétitions).
6. Détection de réclamations factuelles (dates, chiffres, entités nommées) et vérification cross-source (lorsque possible).

5.2 Critères évalués et pondération

Le score article (0–100) est distribué ainsi :

Critère	Description	Poids
Métadonnées complètes	auteur, date, logiciel	20
Qualité des citations	DOI, sources académiques, domaines	25
Qualité d'écriture	longueur, structure, diversité lexicale	20
Densité de revendications factuelles	chiffres/dates/entités	20
Biais / Ton	sentiment, sensationalisme	15
Total		100

TABLE 5.1 – Répartition pondérée du scoring article

5.3 Exemples de règles détectées

— **Alerte :** Abstract trop court (moins de 80 mots).
— **Alerte :** Références sans DOI ni domaine académique.
— **Alerte :** Phrase-somme » trop longue (potentielle incohérence).
— **Alerte :** PDF scanné détecté → nécessite OCR.

5.4 Exemple de rapport JSON (extrait)

```
{
  "title": "Deep Learning for Vision",
  "source": "upload_2025-11-10-12-03.pdf",
  "structure": {
    "abstract": true,
    "introduction": true,
    "conclusion": true,
    "references": true
  },
  "format": { "style": "IEEE", "compliance": 92 },
  "semantic": { "consistencyScore": 0.87 },
  "warnings": [ "Abstract trop court", "R f rences sans DOI" ],
  "final_score": 86,
  "verdict": "Cr dible"
}
```

Listing 5.1 – Extrait JSON de rapport d'analyse

6. Backend – API et services

6.1 Routes principales

Endpoint	Description
`POST /api/v1/verify/image`	Upload image → analyse image + JSON report
`POST /api/v1/verify/article`	Upload PDF → extraction + analyse + JSON report
`GET /api/v1/report/:id`	Récupérer rapport précédemment généré
`POST /api/v1/health`	Santé (healthcheck)

6.2 Exemple d'appel curl

```
curl -X POST "http://localhost:3000/api/v1/verify/article" \
  -H "Accept: application/json" \
  -F "file=@/path/to/article.pdf"
```

Listing 6.1 – UPLOAD d'un PDF via curl

6.3 Architecture logicielle

Le backend suit des principes SOLID :

- Contrôleurs (routes only) : orchestration et validation des entrées.
- Services : logique d'analyse (ImageService, ArticleService).
- Utils : extraction EXIF, OCR wrapper, NLP wrapper.
- Workers (optionnel) : tâches longues -> queue (BullMQ / Redis).

7. Frontend – UX Visualisation

7.1 Principes UX

Interface orientée verdict clair et preuves :

— Score global circulaire et code couleur (vert/jaune/rouge).

— Sections détaillées développables : métadonnées, preuves visuelles, recommandations.

— Émojis et pastilles pour lecture rapide.

— Export PDF/JSON du rapport.

7.2 Composants clés

— FileUploader : gestion drag'n'drop, preview.

— ResultCard : score global + statut.

— EvidenceList : éléments de preuve triés par priorité.

— Timeline : historique des analyses.

7.3 Capture d'écran / Maquettes

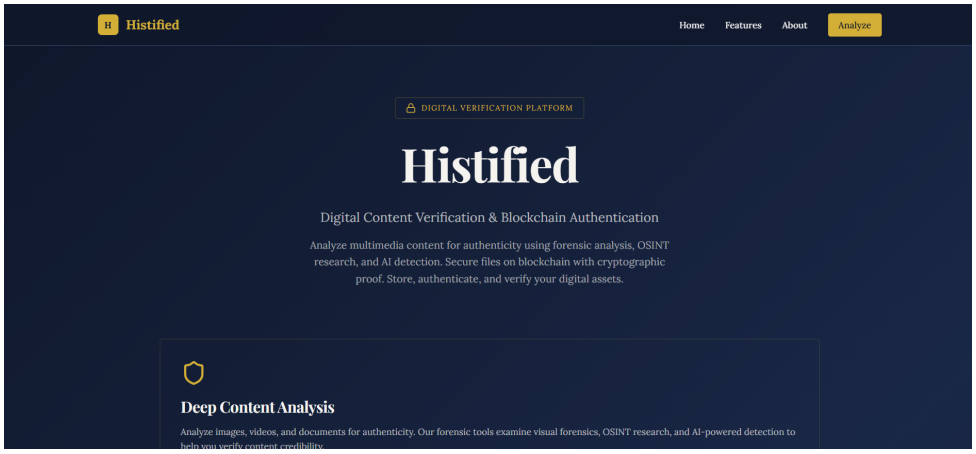

FIGURE 7.1 – Maquette de la page de résultat .

8. Tests et évaluation

8.1 Tests unitaires et d'intégration

Unitaires :

— Parser PDF : extraction des titres et figures.

— EXIF parser : récupération des champs date/logiciel.

— NoiseAnalyzer : test sur images synthétiques.

Intégration :

— Upload complet (frontend → backend → report).

— Scénarios : PDF valide (score attendu >80), PDF scanné, image manipulée (score <50).

8.2 Métriques d'évaluation

— Taux de détection d'altération (images manipulées) sur jeu de test : *à mesurer empirique*.

— Latence : objectif < 5s pour PDF simple (voir limites).

— Robustesse OCR pour PDF scannés.

8.3 Limitations connues

— Impossible de prouver l'authenticité absolue — Histified donne des indicateurs de confiance.

— Certaines manipulations sophistiquées (GANs de très haute qualité) restent difficiles à détecter automatiquement.

— Vérification cross-source dépend de l'accès en temps réel à des sources externes (peut nécessiter quotas / API payantes).

9. Blockchain et sécurité des fichiers

Pour renforcer la fiabilité et la traçabilité des fichiers vérifiés (images et articles PDF), Historified utilise la technologie blockchain via **Hedera Hashgraph** et le stockage décentralisé **IPFS** (via Pinata). Cette approche garantit que les informations sécurisées restent inviolables et accessibles pour de nombreuses années, indépendamment des modifications locales ou d'éventuelles altérations.

- **Stockage décentralisé :** Chaque fichier est uploadé sur IPFS, générant un identifiant unique (CID) qui permet de retrouver de manière permanente son contenu.
- **Intégrité des données :** Le hash SHA-256 du fichier est calculé et enregistré sur Hedera File Service. Toute modification du fichier entraîne un hash différent, permettant une vérification fiable.
- **Preuve immuable :** Les informations CID et SHA-256 sont encapsulées dans une transaction sur la blockchain Hedera, fournissant une preuve immuable de l'existence et de l'intégrité du fichier à un instant donné.
- **Vérification simple et rapide :** L'utilisateur peut soumettre un fichier pour vérification locale, et le système compare son hash avec les enregistrements Hedera pour confirmer si le fichier est authentique ou altéré.
- **Traçabilité longue durée :** L'utilisation de la blockchain garantit que les preuves de vérification restent disponibles et inviolables pour les années à venir, même en cas de suppression ou modification du fichier sur les serveurs locaux.

Cette architecture blockchain permet ainsi de combiner sécurité, transparence et durabilité, offrant aux journalistes et chercheurs un moyen fiable de certifier et tracer l'authenticité de leurs contenus.

10. Sécurité et déploiement

10.1 Sécurité

— Validation stricte des fichiers uploadés (type MIME, taille).
— Sandbox des moteurs d'analyse (éviter exécution de code provenant des métadonnées).
— Stockage chiffré des rapports sensibles si nécessaire.
— Limitation de taux et authentification sur les endpoints de production.

10.2 Déploiement visé

— Conteneurisation : Docker + Compose (services : api, worker, redis, mongodb).
— Orchestration : Kubernetes pour montée en charge.
— Monitoring : Prometheus + Grafana, logs centralisés (ELK).

11. README / Guide d'installation rapide

11.1 Pré-requis

— Node.js 18+, npm/yarn
— Docker
— MongoDB
— Redis (si Worker)

11.2 Installation rapide (local)

1. Cloner le dépôt : `git clone <repo>`
2. Installer le backend : `cd backend && npm install`
3. Lancer : `npm run dev` (ou via Docker Compose)
4. Frontend : `cd frontend && npm install && npm run dev`

11.3 Exemple d'appel

Voir la section *Exemple d'appel curl* au chapitre Backend.

12. Cas d'utilisation et démonstration

12.1 Scénarios de démo

1. Upload d'une image authentique (EXIF complet) → verdict *Probablement authentique*.
2. Upload d'une image composite (retouche locale) → preuve visuelle + score faible.
3. Upload d'un article PDF (format IEEE) → rapport détaillé et score élevé.
4. Upload d'un PDF scanné → message OCR requis + tentative automatique d'OCR.

12.2 Slides / Présentation

On inclut une courte démo vidéo et un set de slides explicatifs dans le dépôt ‘docs/-demo.mp4‘ et ‘docs/slides.pdf‘.

13. Roadmap et améliorations futures

— Intégration d'APIs de vérification tierces (CrossRef, Unpaywall) pour valider DOI et sources.

— Ajout de modèles ML spécialisés (detectors GAN / deepfake).

— Amélioration du moteur de vérification cross-source (recherche d'images inversée, recherche de phrases clés).

— Module d'annotations collaboratives pour journalistes (marquage manuel des preuves).

— Ajout d'un système de confiance utilisateur (notes des analystes humains) pour affiner les seuils.

14. Conclusion

Histified est une solution complète, modulaire et extensible pour la vérification d'images et d'articles. Son approche hybride (règles + analyses statistiques + composants ML optionnels) fournit des verdicts explicables et pratiques pour des workflows journalistiques. Le système n'a pas vocation à remplacer l'analyse humaine mais à alléger et objectiver la phase initiale de vérification.

Bibliographie et références

— Good practice in digital image forensics — articles et ressources académiques (ex. IEEE, ACM).

— Documentation pdf-parse, ExifReader, Sharp, Tesseract.

— Guides UX pour visualisation de données (Nielsen Norman Group).