Synthèse du Projet de Refonte du Système de Rédaction Académique

Ce document compile l'ensemble des analyses, plans et spécifications techniques pour la refonte du système de rédaction académique, incluant l'intégration de Fileverse et des fonctionnalités Web3.

Table des matières

1. Introduction et contexte
2. Plan global de refonte
3. Architecture hybride académique-Fileverse-Web3
4. Authentification Wallet Web3
5. Intégration de Fileverse comme couche de stockage et collaboration
6. Adaptation des modules académiques
7. Fonctionnalités blockchain pour la recherche
8. Interface utilisateur
9. Tests et sécurité
10. Déploiement et documentation
11. Support et évolution

1. Introduction et contexte

Le système de rédaction académique actuel présente plusieurs faiblesses critiques identifiées lors d'une évaluation technique approfondie :

- Code majoritairement composé de stubs et démos non fonctionnels
- Absence d'authentification sécurisée (utilisation de localStorage)
- Gestion des données factice sans persistance réelle
- Fonctionnalités clés manquantes (collaboration, IA, export)
- Architecture déficiente et code de mauvaise qualité
- Problèmes d'interface utilisateur et d'accessibilité
- Absence de tests et de documentation
- Vulnérabilités de sécurité

La refonte proposée vise à transformer ce système en une plateforme de rédaction académique robuste, décentralisée et adaptée à la recherche en cryptomonnaie et Web3, en intégrant Fileverse comme couche documentaire principale.

2. Plan global de refonte

La refonte du système s'articule autour de 10 étapes clés :

1. **Audit et analyse de l'existant**
2. Analyse critique du code actuel
3. Cartographie des fonctionnalités académiques essentielles
4. Analyse des besoins spécifiques Web3

5. Étude technique de Fileverse

6. **Architecture hybride académique-Web3**

7. Conception de l'architecture système
8. Définition des modèles de données
9. Cartographie des flux de travail

10. Spécification des contrats intelligents

11. **Authentification Web3**

12. Intégration de connecteurs de wallets
13. Système de signature de messages
14. Gestion des profils utilisateurs

15. Gestion des permissions

16. **Intégration de Fileverse**

17. Configuration du SDK Fileverse
18. Adaptation de la couche de stockage
19. Système de métadonnées académiques

20. Gestion des versions et historique

21. **Adaptation des modules académiques**

22. Storyboard Engine
23. Éditeur de rédaction
24. Système de révision

25. Module de finalisation

26. **Fonctionnalités blockchain pour la recherche**

27. Certification des documents

28. Citations vérifiables
29. Attribution et droits d'auteur

30. Peer-review décentralisé

31. **Interface utilisateur**

32. Design system Web3
33. Vues de collaboration
34. Dashboard de recherche

35. Visualisation blockchain

36. **Tests et sécurité**

37. Tests unitaires et d'intégration
38. Audit de sécurité Web3
39. Tests de performance

40. Tests utilisateurs

41. **Déploiement et documentation**

42. Stratégie de déploiement progressif
43. Documentation technique
44. Documentation utilisateur

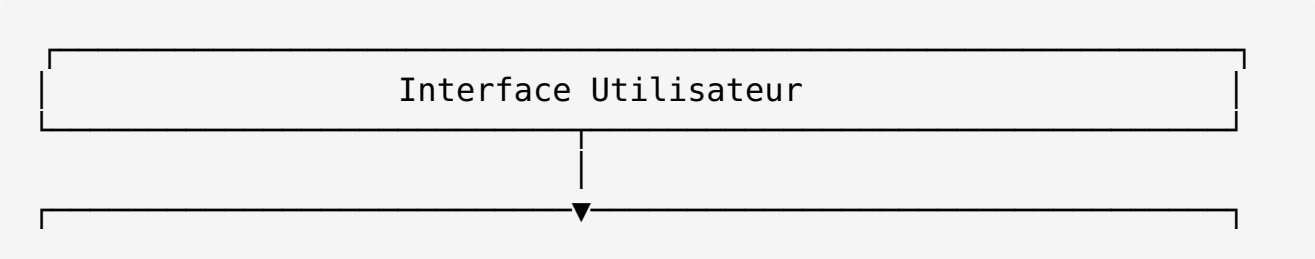
45. Formation

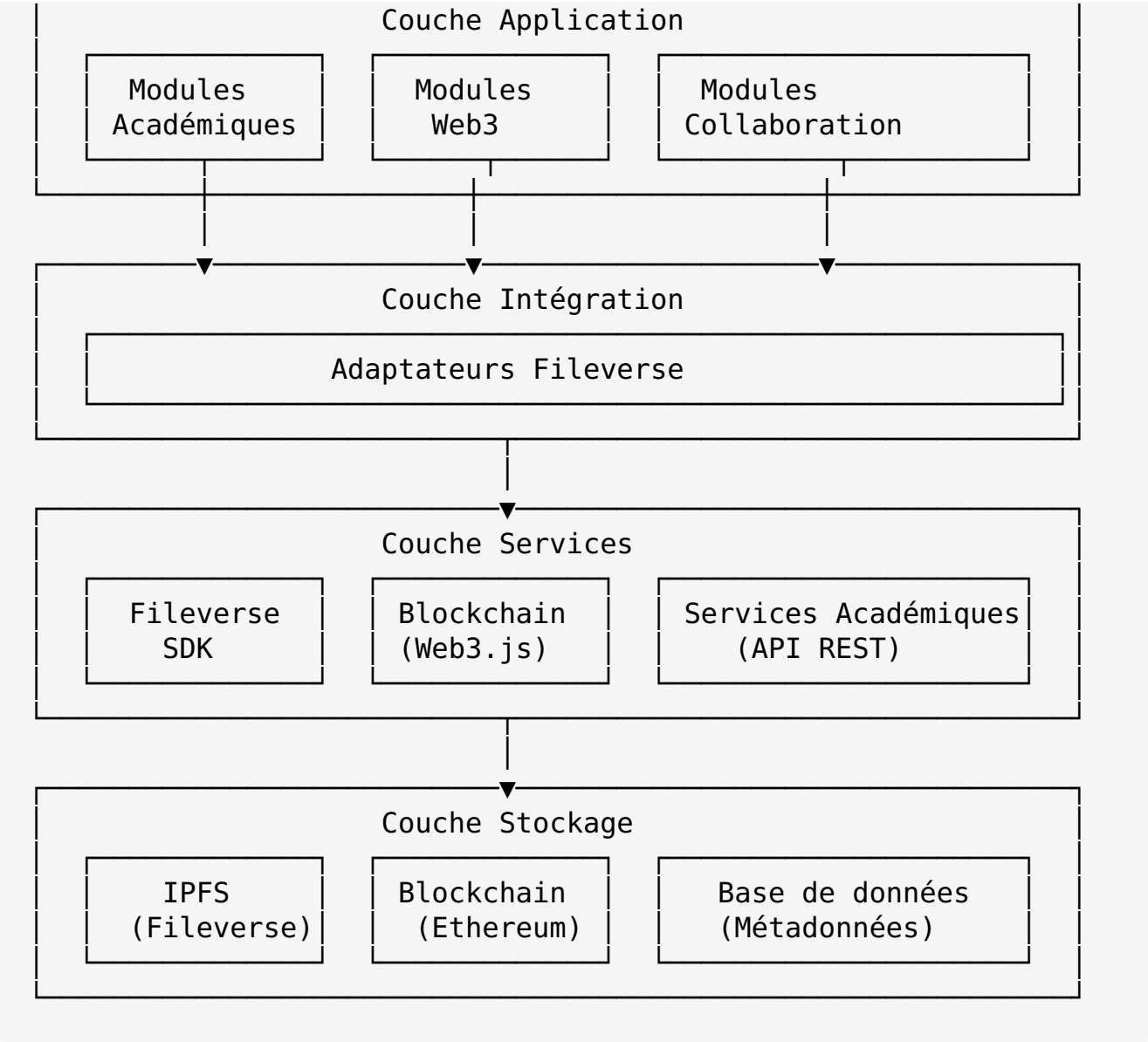
46. **Support et évolution**

    - Système de support technique
    - Plan d'évolution
    - Communauté
    - Métriques et analytics

3. Architecture hybride académique-Fileverse-Web3

Vue d'ensemble de l'architecture





Composants clés de l'architecture

Couche Interface Utilisateur

- **Dashboard académique** : Interface principale pour la gestion des projets de recherche
- **Éditeur de documents** : Intégration de l'éditeur Fileverse avec fonctionnalités académiques
- **Outils de collaboration** : Commentaires, révisions, et annotations en temps réel
- **Interface Web3** : Connexion wallet, visualisation des certifications et attributions

Couche Application

Modules Académiques - **Storyboard Engine** : Planification et structuration des documents académiques - **Module de Rédaction** : Gestion du contenu et des citations - **Module de Révision** : Analyse stylistique et grammaticale - **Module de Finalisation** : Préparation pour publication et export

Modules Web3 - **Gestionnaire d'Authentification** : Connexion et gestion des sessions via wallet - **Certification Manager** : Horodatage et certification des documents sur blockchain - **Citation Verifier** : Vérification des citations via blockchain - **Rights Manager** : Gestion des droits d'auteur et attributions

Modules de Collaboration - **Real-time Sync** : Synchronisation en temps réel des modifications - **Version Control** : Gestion des versions et historique - **Peer Review** : Système de révision par les pairs décentralisé - **Notification System** : Alertes et notifications pour les collaborateurs

Couche Intégration

- **Adaptateurs Fileverse** : Couche d'abstraction pour interagir avec l'API Fileverse
- **Event Bus** : Système de messagerie pour la communication entre modules
- **State Manager** : Gestion de l'état global de l'application
- **API Gateway** : Point d'entrée unifié pour les services externes

Couche Services

- **Fileverse SDK** : Interface avec les services Fileverse
- **Web3.js/ethers.js** : Interaction avec la blockchain
- **Services Académiques** : API pour les fonctionnalités académiques spécifiques
- **Identity Service** : Gestion des identités et profils utilisateurs

Couche Stockage

- **IPFS via Fileverse** : Stockage décentralisé des documents
- **Blockchain** : Stockage des certifications, attributions et métadonnées critiques
- **Base de données** : Stockage des métadonnées académiques et index de recherche

Flux de données et interactions

Création et édition de document

1. L'utilisateur se connecte via son wallet
2. Il crée un nouveau document académique dans l'interface
3. Le Storyboard Engine aide à structurer le document
4. Le contenu est édité dans l'éditeur intégré Fileverse
5. Les modifications sont synchronisées en temps réel via Fileverse
6. Les versions sont automatiquement enregistrées sur IPFS
7. Les métadonnées académiques sont stockées dans la base de données

Certification et publication

1. L'utilisateur finalise son document
2. Le module de Finalisation prépare le document pour publication
3. Le Certification Manager crée un hash du document
4. Ce hash est enregistré sur la blockchain avec horodatage
5. Un certificat de publication est généré avec preuve blockchain
6. Le document est publié sur IPFS avec métadonnées académiques
7. Les droits d'auteur sont enregistrés via smart contract

Collaboration et révision

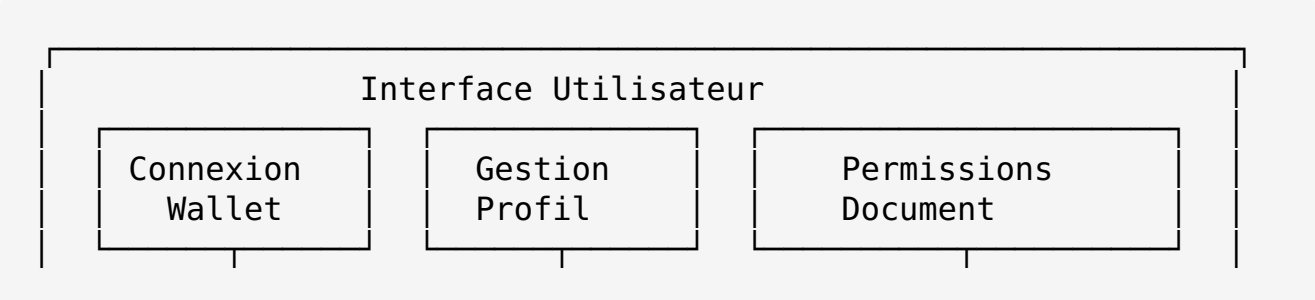
1. L'auteur invite des collaborateurs via leurs adresses wallet
2. Les collaborateurs accèdent au document via Fileverse
3. Ils ajoutent commentaires et suggestions en temps réel
4. Le module de Révision analyse la qualité académique
5. Les modifications sont tracées avec attribution blockchain
6. Le système de versions permet de comparer les changements
7. Le consensus sur les modifications est enregistré on-chain

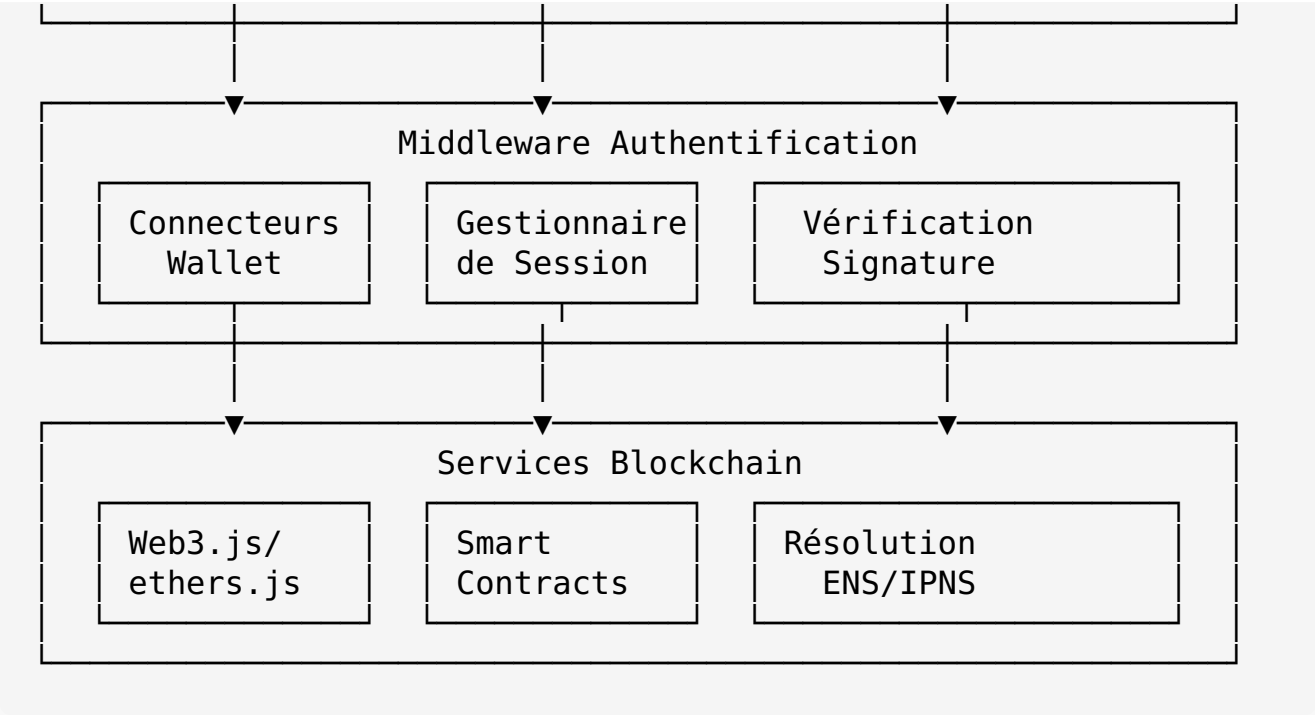
Avantages de cette architecture hybride

- **Décentralisation sélective** : Les données critiques sont sur blockchain/IPFS, les données volumineuses ou temporaires dans des bases traditionnelles
- **Flexibilité académique** : Conservation des fonctionnalités académiques spécifiques
- **Évolutivité** : Architecture modulaire permettant d'ajouter de nouvelles fonctionnalités
- **Résilience** : Stockage distribué via IPFS pour la persistance des documents
- **Transparence** : Traçabilité complète des contributions et modifications
- **Interopérabilité** : Possibilité d'intégration avec d'autres outils académiques

4. Authentification Wallet Web3

Vue d'ensemble du système d'authentification





Composants du système d'authentification

Interface Utilisateur

- **Bouton de connexion wallet** : Interface unifiée pour tous les types de wallets
- **Modal de connexion** : Affichage des options de connexion (Metamask, WalletConnect, etc.)
- **Profil utilisateur** : Interface de gestion du profil académique lié au wallet
- **Gestionnaire de permissions** : Interface pour gérer les droits d'accès aux documents

Middleware Authentification

Connecteurs Wallet - **Adaptateur Metamask** : Connexion via l'extension de navigateur Metamask - **Adaptateur WalletConnect** : Support des wallets mobiles via WalletConnect - **Adaptateur Coinbase Wallet** : Support du wallet Coinbase - **Adaptateur Rainbow** : Support du wallet Rainbow pour iOS/Android

Gestionnaire de Session - **Session Manager** : Gestion de l'état de connexion utilisateur - **Token Manager** : Gestion des jetons d'authentification basés sur signature - **Session Storage** : Stockage sécurisé des informations de session - **Session Expiry** : Gestion de l'expiration des sessions

Vérification Signature - **Message Signer** : Génération et signature de messages pour authentification - **Signature Verifier** : Vérification des signatures pour valider l'identité - **Challenge Generator** : Création de challenges uniques pour l'authentification - **Nonce Manager** : Gestion des nonces pour prévenir les attaques par rejeu

Services Blockchain

- **Provider Manager** : Gestion des connexions aux réseaux blockchain
- **Network Detector** : Détection et gestion des changements de réseau
- **ENS Resolver** : Résolution des noms ENS pour une meilleure expérience utilisateur
- **Contract Interactor** : Interface avec les smart contracts d'authentification

Flux d'authentification

Connexion initiale

1. L'utilisateur clique sur "Se connecter avec Wallet"
2. Le modal de connexion affiche les options disponibles
3. L'utilisateur sélectionne son wallet préféré
4. Le système génère un message unique à signer
5. L'utilisateur signe le message via son wallet
6. Le système vérifie la signature et l'adresse
7. Une session est créée avec un jeton d'authentification
8. L'utilisateur est redirigé vers le dashboard

Vérification de session

1. À chaque requête, le middleware vérifie le jeton d'authentification
2. Si le jeton est valide, la requête est autorisée
3. Si le jeton est expiré, une nouvelle signature est demandée
4. Si l'utilisateur change de compte dans son wallet, la session est invalidée

Déconnexion

1. L'utilisateur clique sur "Déconnexion"
2. La session est détruite côté client
3. Les jetons d'authentification sont invalidés
4. L'utilisateur est redirigé vers la page d'accueil

Gestion des profils académiques

Création de profil

1. Après la première connexion, l'utilisateur est invité à créer un profil académique
2. Il fournit ses informations académiques (nom, institution, domaine de recherche)
3. Ces informations sont liées à son adresse wallet
4. Un NFT de profil académique peut être créé pour représenter son identité

Liaison avec identifiants académiques

1. L'utilisateur peut lier son ORCID, ResearcherID ou autres identifiants
2. Un processus de vérification confirme la propriété de ces identifiants
3. Les identifiants vérifiés sont stockés dans le profil blockchain
4. Cette liaison renforce la crédibilité académique de l'adresse wallet

Système de permissions

Niveaux d'accès

- **Propriétaire** : Contrôle total du document et des permissions
- **Éditeur** : Peut modifier le contenu et inviter des réviseurs
- **Réviseur** : Peut ajouter des commentaires et suggestions
- **Lecteur** : Peut uniquement lire le document

Gestion des permissions

1. Le propriétaire du document définit les permissions par adresse wallet
2. Les permissions sont stockées dans un smart contract ou via Fileverse
3. Les modifications de permission sont enregistrées sur la blockchain
4. Les permissions peuvent être limitées dans le temps ou conditionnelles

Sécurité et considérations techniques

Protection contre les attaques

- **Signature unique** : Chaque demande d'authentification utilise un nonce unique
- **Vérification d'origine** : Protection contre les attaques de phishing
- **Rate limiting** : Limitation du nombre de tentatives d'authentification
- **Détection d'anomalies** : Alertes en cas de comportement suspect

Multi-chaîne et interopérabilité

- Support des principales blockchains (Ethereum, Polygon, Optimism, etc.)
- Résolution d'identité cross-chain via solutions comme CCIP
- Compatibilité avec les standards d'authentification Web3 (EIP-4361, CAIP-122)
- Support des solutions de scaling (L2, sidechains)

Expérience utilisateur

- Feedback clair à chaque étape du processus d'authentification
- Gestion des erreurs avec messages explicatifs
- Mode dégradé en cas d'indisponibilité du wallet

- Support des utilisateurs non-techniques avec guides détaillés

Implémentation technique

Bibliothèques recommandées

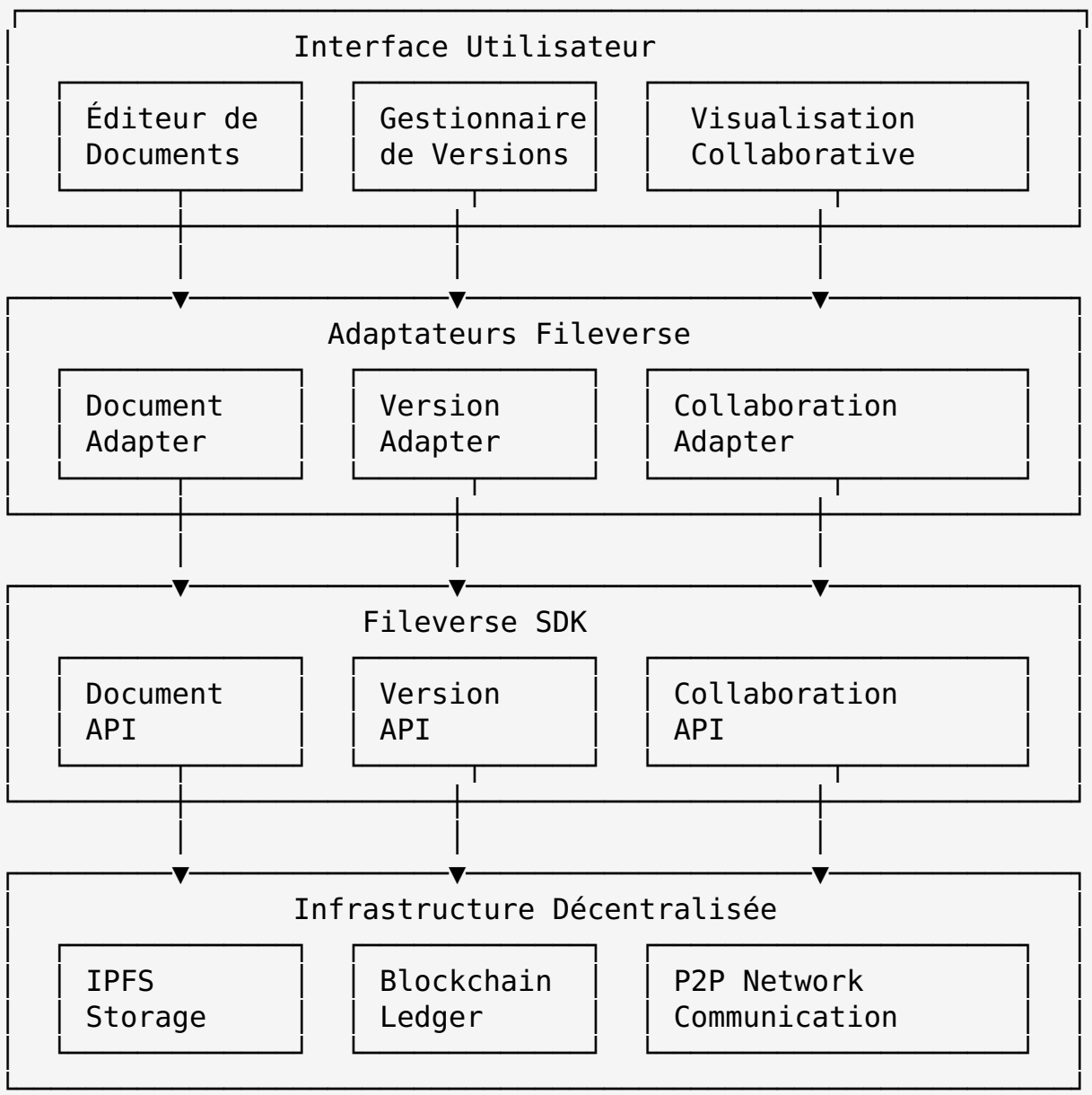
- **Web3-React** ou **RainbowKit** : Pour la gestion des connexions wallet
- **ethers.js** : Pour l'interaction avec la blockchain
- **SIWE (Sign-In with Ethereum)** : Pour l'authentification basée sur signature
- **React-Query** : Pour la gestion des états d'authentification côté client

Exemple de code pour l'authentification

```
// Exemple simplifié d'authentification par signature
async function authenticateWithWallet() {
  // 1. Connexion au wallet
  const provider = await connectWallet();
  const signer = provider.getSigner();
  const address = await signer.getAddress();

  // 2. Génération d'un message unique
  const nonce = generateNonce();
  const message = `Connexion au Système de Rédaction
Académique\n\nAdresse: ${address}\nNonce: ${nonce}\nDate: ${new
Date().toISOString()}`;

  // 3. Signature du message
  const signature = await signer.signMessage(message);

  // 4. Vérification côté serveur et création de session
  const authResult = await
verifySignatureAndCreateSession(address, message, signature);

  // 5. Stockage du jeton d'authentification
  if (authResult.success) {
    storeAuthToken(authResult.token);
    return true;
  }

  return false;
}
```

5. Intégration de Fileverse comme couche de stockage et collaboration

Vue d'ensemble de l'intégration Fileverse



Composants de l'intégration Fileverse

Interface Utilisateur

- **Éditeur de Documents** : Interface d'édition collaborative intégrée à Fileverse
- **Gestionnaire de Versions** : Interface pour naviguer et gérer l'historique des versions
- **Visualisation Collaborative** : Affichage des collaborateurs actifs et de leurs modifications

- **Panneau de Métadonnées** : Interface pour gérer les métadonnées académiques

Adaptateurs Fileverse

Document Adapter - **Document Mapper** : Conversion entre modèles académiques et documents Fileverse - **Metadata Enhancer** : Extension des métadonnées Fileverse avec données académiques - **Content Processor** : Traitement du contenu académique (citations, références, etc.) - **Schema Validator** : Validation des documents selon les schémas académiques

Version Adapter - **Version Controller** : Gestion des versions de documents académiques - **Diff Generator** : Génération des différences entre versions - **Snapshot Manager** : Création de snapshots pour les étapes clés (soumission, révision) - **History Tracker** : Suivi de l'évolution du document académique

Collaboration Adapter - **Collaboration Manager** : Gestion des droits et rôles de collaboration - **Comment Handler** : Traitement des commentaires et annotations académiques - **Review Processor** : Gestion des cycles de révision académique - **Change Tracker** : Suivi des modifications avec attribution

Fileverse SDK

- **Document API** : Création, lecture, mise à jour et suppression de documents
- **Version API** : Gestion des versions et de l'historique
- **Collaboration API** : Gestion des collaborateurs et des permissions
- **Storage API** : Interaction avec le stockage IPFS

Infrastructure Décentralisée

- **IPFS Storage** : Stockage décentralisé des documents et ressources
- **Blockchain Ledger** : Enregistrement des métadonnées critiques et certifications
- **P2P Network** : Communication en temps réel entre collaborateurs

Flux de données et processus clés

Création et sauvegarde de document

1. L'utilisateur crée un nouveau document académique
2. Le Document Adapter prépare la structure compatible Fileverse
3. Les métadonnées académiques sont ajoutées au document
4. Le document est envoyé à Fileverse via le SDK
5. Fileverse stocke le contenu sur IPFS
6. L'identifiant du document est retourné et stocké localement
7. Les métadonnées critiques sont enregistrées sur la blockchain

```
// Exemple de création de document avec Fileverse
async function createAcademicDocument(title, content, metadata)
{
  // Préparation des données académiques
  const academicMetadata = prepareAcademicMetadata(metadata);

  // Création du document via Fileverse SDK
  const document = await fileverse.documents.create({
    title,
    content,
    metadata: {
      ...academicMetadata,
      type: 'academic',
      createdAt: new Date().toISOString(),
      version: '1.0.0'
    }
  });

  // Enregistrement des métadonnées critiques sur blockchain
  await recordDocumentMetadataOnChain(document.id, {
    title,
    author: getCurrentUserAddress(),
    timestamp: Date.now(),
    contentHash: hashContent(content)
  });

  return document;
}
```

Gestion des versions

1. L'utilisateur modifie un document existant
2. Le Version Adapter détecte les changements significatifs
3. Une nouvelle version est créée avec description des changements
4. L'historique des versions est mis à jour
5. Les différences sont calculées et stockées
6. Les métadonnées de version sont enregistrées

```
// Exemple de gestion de version avec Fileverse
async function saveDocumentVersion(documentId, content,
versionNotes) {
  // Récupération de la version actuelle
  const currentVersion = await
fileverse.documents.getLatestVersion(documentId);

  // Calcul des différences
  const diff = generateDiff(currentVersion.content, content);
```

```
  // Création d'une nouvelle version
  const newVersion = await
fileverse.documents.createVersion(documentId, {
    content,
    metadata: {
      versionNotes,
      versionNumber: currentVersion.metadata.versionNumber + 1,
      timestamp: Date.now(),
      author: getCurrentUserAddress(),
      changes: summarizeChanges(diff)
    }
  });

  // Mise à jour de l'historique académique
  await updateAcademicHistory(documentId, newVersion.id, diff);

  return newVersion;
}
```

Collaboration en temps réel

1. L'utilisateur invite des collaborateurs via leurs adresses wallet
2. Le Collaboration Adapter définit les permissions appropriées
3. Les collaborateurs accèdent au document via Fileverse
4. Les modifications sont synchronisées en temps réel
5. Les conflits sont détectés et résolus automatiquement
6. L'attribution des modifications est maintenue

```
// Exemple d'invitation de collaborateurs avec Fileverse
async function inviteCollaborator(documentId, walletAddress,
role) {
  // Définition des permissions selon le rôle académique
  const permissions = mapAcademicRoleToPermissions(role);

  // Invitation via Fileverse SDK
  const invitation = await
fileverse.documents.invite(documentId, {
    address: walletAddress,
    permissions,
    metadata: {
      role,
      invitedBy: getCurrentUserAddress(),
      timestamp: Date.now()
    }
  });

  // Notification au collaborateur
  await notifyCollaborator(walletAddress, documentId, role);
```

```
  return invitation;
}
```

Extension des métadonnées académiques

Schéma de métadonnées académiques

```
{
  "academic": {
    "type": "article|thesis|report|book",
    "status": "draft|review|published",
    "discipline": "string",
    "keywords": ["string"],
    "abstract": "string",
    "references": [
      {
        "id": "string",
        "type": "article|book|website",
        "title": "string",
        "authors": ["string"],
        "year": "number",
        "doi": "string",
        "url": "string"
      }
    ],
    "citations": [
      {
        "id": "string",
        "referenceId": "string",
        "context": "string",
        "position": {
          "paragraph": "number",
          "offset": "number"
        }
      }
    ],
    "structure": {
      "sections": [
        {
          "id": "string",
          "title": "string",
          "level": "number",
          "content": "string"
        }
      ]
    }
  }
}
```

Synchronisation et résilience

Stratégie de synchronisation

- **Synchronisation en temps réel** : Pour les modifications collaboratives
- **Synchronisation différée** : Pour les métadonnées volumineuses
- **Synchronisation sélective** : Pour les ressources liées (images, tableaux)
- **Synchronisation incrémentielle** : Pour les grands documents

Gestion de la résilience

- **Mode hors ligne** : Fonctionnalité d'édition sans connexion
- **Récupération automatique** : En cas d'interruption de connexion
- **Sauvegarde redondante** : Copies de sécurité sur plusieurs nœuds IPFS
- **Vérification d'intégrité** : Validation des documents via hachage

Optimisations pour le contexte académique

Performance

- **Chargement progressif** : Pour les documents volumineux
- **Mise en cache intelligente** : Des sections fréquemment consultées
- **Compression sélective** : Pour les ressources volumineuses
- **Préchargement prédictif** : Basé sur les habitudes de navigation

Expérience utilisateur

- **Transitions fluides** : Entre modes d'édition et de visualisation
- **Feedback instantané** : Pour les actions de sauvegarde et synchronisation
- **Indicateurs de présence** : Montrant l'activité des collaborateurs
- **Notifications contextuelles** : Pour les événements importants

Implémentation technique

Dépendances requises

```
{
  "dependencies": {
    "fileverse-sdk": "^1.x.x",
    "ipfs-http-client": "^56.x.x",
    "ethers": "^5.x.x",
    "prosemirror": "^1.x.x",
    "yjs": "^13.x.x"
```

```
  }
}
```

Configuration initiale

```
// Configuration de Fileverse avec authentification wallet
const fileverse = new FileverseSdk({
  apiKey: process.env.FILEVERSE_API_KEY,
  wallet: connectedWallet,
  ipfsGateway: process.env.IPFS_GATEWAY,
  networkId: process.env.NETWORK_ID
});

// Initialisation des adaptateurs
const documentAdapter = new DocumentAdapter(fileverse);
const versionAdapter = new VersionAdapter(fileverse);
const collaborationAdapter = new
CollaborationAdapter(fileverse);

// Configuration de l'éditeur avec collaboration en temps réel
const editor = new AcademicEditor({
  document: documentAdapter,
  version: versionAdapter,
  collaboration: collaborationAdapter,
  realtime: true
});
```

6. Adaptation des modules académiques

Cette section sera développée dans une phase ultérieure du projet et détaillera l'adaptation des modules académiques spécifiques (storyboard, rédaction, révision, finalisation) pour qu'ils fonctionnent nativement avec Fileverse et l'environnement Web3.

7. Fonctionnalités blockchain pour la recherche

Cette section sera développée dans une phase ultérieure du projet et détaillera l'implémentation des fonctionnalités blockchain spécifiques pour la recherche académique (certification des documents, citations vérifiables, attribution et droits d'auteur, peer-review décentralisé).

8. Interface utilisateur

Cette section sera développée dans une phase ultérieure du projet et détaillera la refonte de l'interface utilisateur pour intégrer les fonctionnalités Web3 et Fileverse tout en maintenant une expérience utilisateur optimale pour le contexte académique.

9. Tests et sécurité

Cette section sera développée dans une phase ultérieure du projet et détaillera les stratégies de test et de sécurité pour garantir la robustesse et la fiabilité du système.

10. Déploiement et documentation

Cette section sera développée dans une phase ultérieure du projet et détaillera les stratégies de déploiement progressif et de documentation pour faciliter l'adoption du système.

11. Support et évolution

Cette section sera développée dans une phase ultérieure du projet et détaillera les stratégies de support technique et d'évolution du système pour répondre aux besoins futurs des utilisateurs.