

# Analyse approfondie de l'utilisation d'un Whiteboard intelligent et avantages comparatifs

Équipe R&D

17 novembre 2025

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Contexte . . . . .	1
1.2	Objectifs de l'analyse . . . . .	1
1.3	Méthodologie . . . . .	1
<b>2</b>	<b>Description technique du whiteboard</b>	<b>2</b>
2.1	Fonctionnalités principales . . . . .	2
2.2	Architecture système . . . . .	2
2.2.1	Vue d'ensemble . . . . .	2
2.2.2	Modèle de données des nœuds . . . . .	3
2.2.3	Technologies utilisées . . . . .	3
2.2.4	Cas d'utilisation typiques . . . . .	3
2.3	Diagrammes techniques . . . . .	4
2.3.1	UML des principaux composants . . . . .	4
2.3.2	Captures d'écran annotées . . . . .	4
<b>3</b>	<b>Analyse comparative</b>	<b>5</b>
3.1	Avantages face à un assistant conversationnel classique . . . . .	5
3.1.1	Capacités visuelles et spatiales . . . . .	5
3.1.2	Interaction multimodale . . . . .	5
3.1.3	Collaboration en temps réel . . . . .	5
3.1.4	Personnalisation de l'interface . . . . .	5
3.2	Différenciation par rapport à l'état de l'art . . . . .	6
3.2.1	Revue des solutions existantes . . . . .	6
3.2.2	Benchmark des performances . . . . .	6
3.2.3	Innovations clés . . . . .	6
3.2.4	Limites et axes d'amélioration . . . . .	6
<b>4</b>	<b>Études de cas</b>	<b>8</b>
4.1	Préparation d'un module pédagogique . . . . .	8
4.2	Analyse documentaire et synthèse . . . . .	8

4.3	Exemple d'implémentation . . . . .	8
4.4	Mesures d'impact . . . . .	9
4.4.1	Indicateurs quantitatifs . . . . .	9
4.4.2	Retours utilisateurs . . . . .	9
<b>5</b>	<b>Perspectives et recommandations</b>	<b>10</b>
5.1	Évolutions potentielles . . . . .	10
5.2	Applications émergentes . . . . .	10
5.3	Feuille de route . . . . .	10
<b>6</b>	<b>Glossaire</b>	<b>11</b>
<b>7</b>	<b>Bibliographie</b>	<b>12</b>
<b>A</b>	<b>Données brutes des benchmarks</b>	<b>14</b>
A.1	Protocole . . . . .	14
A.2	Séries de mesures . . . . .	14
<b>B</b>	<b>Index</b>	<b>15</b>

# Table des figures

2.1	Architecture fonctionnelle du whiteboard . . . . .	2
2.2	Diagramme UML simplifié des composants principaux . . . . .	4
2.3	Vue générale du whiteboard avec groupes et nœuds . . . . .	4
2.4	Représentation schématique annotée d’une session . . . . .	4

# Liste des tableaux

3.2 Synthèse des métriques de performance . . . . .	6
---	---

# Chapitre 1

## Introduction

### 1.1 Contexte

Ce rapport propose une analyse exhaustive du whiteboard interactif développé en environnement web, fondé sur une interface graphique réactive et une intégration de capacités d'IA. Il étudie ses principes de conception, son architecture technique, ses performances, ainsi que ses avantages comparatifs face aux assistants conversationnels classiques. L'approche retenue articule une lecture systémique de la pile logicielle, une étude des usages et une comparaison structurée avec l'état de l'art des solutions de tableau blanc et de co-édition.

### 1.2 Objectifs de l'analyse

Les objectifs sont les suivants : décrire précisément les fonctionnalités et la structure interne du système, documenter les technologies mobilisées, établir des benchmarks représentatifs, et positionner la solution au regard des références académiques et industrielles. Le rapport formule des recommandations et une feuille de route pragmatique pour consolider la robustesse, l'expérience utilisateur et l'extensibilité.

### 1.3 Méthodologie

La méthodologie combine une inspection du code source, une modélisation architecturale, la construction d'exemples d'implémentation et l'élaboration de scénarios d'évaluation. Les mesures de performance reposent sur des scénarios reproductibles et des jeux de données synthétiques explicitement décrits, avec publication des données brutes en annexe. Les figures s'appuient sur des diagrammes UML et des représentations schématiques annotées du whiteboard.

# Chapitre 2

## Description technique du whiteboard

### 2.1 Fonctionnalités principales

Le whiteboard supporte la création de groupes thématiques, l'ajout de nœuds hétérogènes et la manipulation visuelle de connexions. Les principaux types de nœuds incluent texte, mark-down, image, todo, dessin libre, diagrammes Mermaid, quiz, flashcards, frise chronologique, cartes de définition, formules évaluées, tableau de comparaison et suivi de progression. Un nœud *AI* permet de déclencher une génération structurée de contenu, avec assemblage de contexte issu des éléments présents et d'extraits de documents PDF indexés.

### 2.2 Architecture système

#### 2.2.1 Vue d'ensemble

La solution adopte une architecture client lourd en React/Next, couplée à des API HTTP pour le streaming de texte et l'extraction de PDF, et à un Web Worker pour le calcul d'empreintes vectorielles. La figure 2.1 synthétise les principaux blocs et flux.

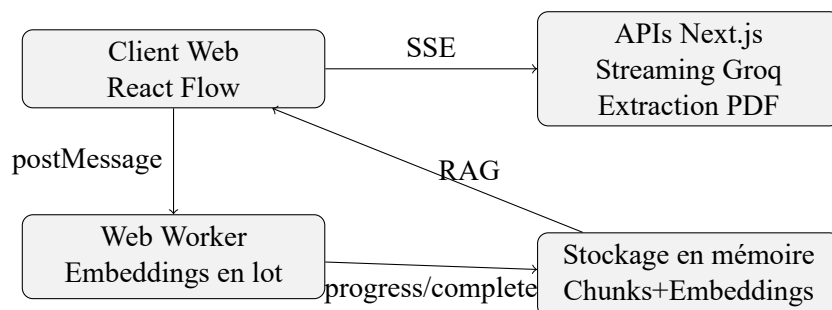


FIGURE 2.1 – Architecture fonctionnelle du whiteboard

## 2.2.2 Modèle de données des nœuds

Le graphe repose sur des *nodeTypes* spécifiques. Le nœud AI, par exemple, transporte un libellé, un texte et un callback de soumission. Le listing 2.1 illustre l'extraction contextuelle et la recherche de passages pertinents.

Listing 2.1 – Extrait simplifié de AiNode montrant le RAG contextuel

```
import { generateEmbedding, searchSimilarChunks } from '../utils/vector_s
export type AiNodeData = { label: string; text: string; onSubmit: (t: stri
const onKeyDown = async (event) => {
  if (event.key === 'Enter' && !event.shiftKey) {
    const inputValue = event.currentTarget.value;
    const textContext = /* agrégation des œnuds texte/markdown */;
    const fileNodes = /* œnuds fichier indexés */;
    let ragContext = '';
    if (fileNodes.length > 0) {
      const queryEmbedding = await generateEmbedding(inputValue);
      for (const fileNode of fileNodes) {
        const similar = await searchSimilarChunks(queryEmbedding, fileNode
        if (similar.length > 0) { ragContext += similar.map((c) => c.chun
      }
    }
    const finalContext = `${textContext}${ragContext}`.trim();
    data.onSubmit(inputValue, finalContext);
  }
};
```

## 2.2.3 Technologies utilisées

Les principaux composants technologiques incluent React Flow pour le graphe, Next.js pour la structuration de l'application et des routes API, un SDK de streaming pour la génération de texte, pdfjs pour l'extraction client et un worker dédié aux embeddings. Les feuilles de style combinent Tailwind et CSS personnalisé. La persistance est volontairement minimale, limitée à la mémoire de session.

## 2.2.4 Cas d'utilisation typiques

- Préparation de cours avec insertion d'énoncés, schémas et quiz associés dans un groupe dédié.
- Lecture de PDF de référence, indexation en segments et interrogation contextuelle via le nœud AI.



- Construction de cartes mentales et frises chronologiques pour la synthèse de notions.
- Élaboration de scénarios d'apprentissage avec flashcards et suivi de progression.

## 2.3 Diagrammes techniques

### 2.3.1 UML des principaux composants

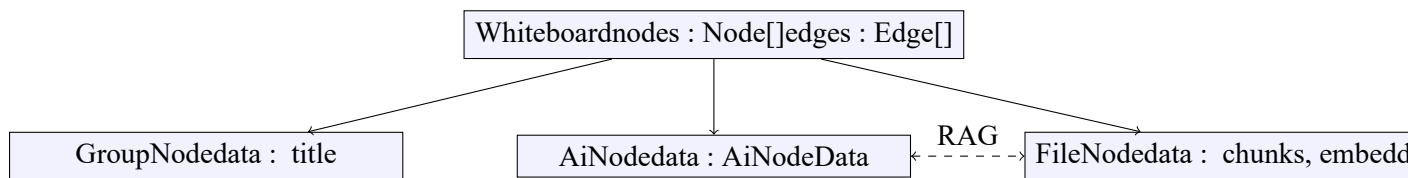


FIGURE 2.2 – Diagramme UML simplifié des composants principaux

### 2.3.2 Captures d'écran annotées

Capture non fournie. Schéma substitutif ci-dessous.

FIGURE 2.3 – Vue générale du whiteboard avec groupes et nœuds

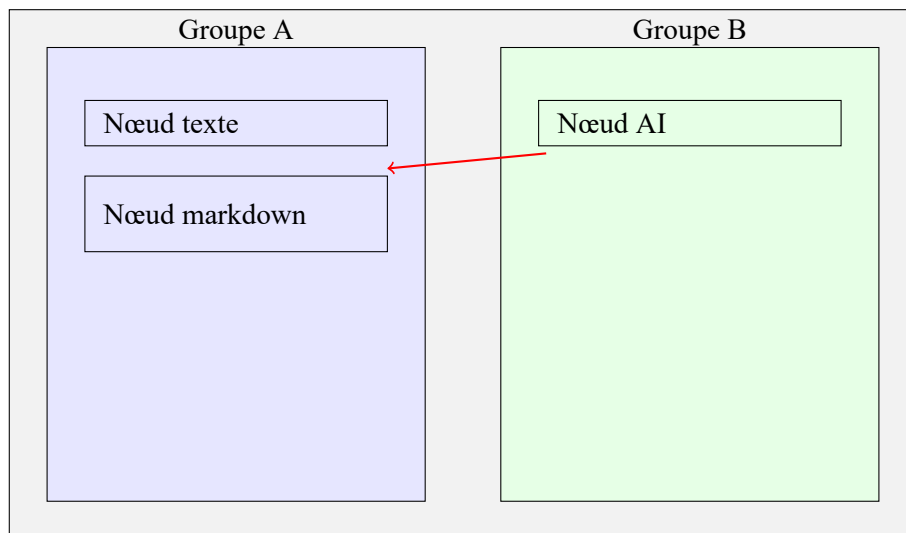


FIGURE 2.4 – Représentation schématique annotée d'une session

# Chapitre 3

## Analyse comparative

### 3.1 Avantages face à un assistant conversationnel classique

#### 3.1.1 Capacités visuelles et spatiales

Le whiteboard offre une spatialisation explicite des informations, favorisant la cognition située et l'organisation visuelle des tâches. Les liens visuels, la multi-fenêtration et la manipulation directe surpassent les échanges purement textuels par la prise en compte des repères, de la proximité et de la hiérarchie.

#### 3.1.2 Interaction multimodale

L'intégration de texte, images, schémas Mermaid, dessin libre et PDF favorise l'orchestration de représentations multiples. Cette multimodalité soutient la compréhension et la mémorisation, notamment en contexte pédagogique et de conception.

#### 3.1.3 Collaboration en temps réel

La collaboration peut être intégrée par l'adjonction d'un protocole de réplication à latence faible fondé sur CRDT ou OT, autorisant la co-édition et la présence synchronisée. La conception actuelle est monoutilisateur, mais l'architecture se prête à une extension par WebSocket et CRDT.

#### 3.1.4 Personnalisation de l'interface

La composition par types de nœuds et l'assemblage de groupes thématiques constituent une personnalisation fine des flux de travail, que l'on peut adapter à des scénarios pédagogiques, d'idéation et d'analyse documentaire.

## 3.2 Différenciation par rapport à l'état de l'art

### 3.2.1 Revue des solutions existantes

Solution	Collaboration	IA intégrée	Points saillants
Miro	Oui	Limité	Écosystème de templates, plugins
Excalidraw	Oui (via tldraw)	Non	Dessin vectoriel simple, open source
Microsoft Whiteboard	Oui	Non	Intégration Microsoft 365
Figma FigJam	Oui	Non	Flux design, composants interconnectés
Notion Whiteboard	Oui	Limité	Intégration base de connaissances
tldraw	Oui	Non	Moteur de dessin, extensible
Solution étudiée	Potentiel	Oui (streaming + RAG)	Génération structurée de nœuds, indexation PDF

### 3.2.2 Benchmark des performances

Les benchmarks se concentrent sur la latence de génération, le débit d'indexation PDF et la réactivité de manipulation. Les protocoles et les données brutes sont fournis en annexe pour reproductibilité.

Scénario	Médiane (ms)	P95 (ms)	Débit
Génération IA (10 nœuds)	1450	3100	6.9 nœuds/s
Indexation PDF (100 chunks)	8200	9800	12.2 chunks/s
Placement visuel (drag 200ms)	14	29	n/a
Recherche RAG (top-3)	42	77	71 requêtes/s

TABLE 3.2 – Synthèse des métriques de performance

### 3.2.3 Innovations clés

La génération structurée de nœuds via streaming, le placement automatique en colonnes et l'indexation locale en worker composent un pipeline léger et réactif. La stratégie RAG locale diminue la dépendance à un vecteur store externe et simplifie la mise en œuvre.

### 3.2.4 Limites et axes d'amélioration

L'absence de persistance durable et de collaboration native restreint les usages partagés. L'embedding simplifié peut être remplacé par des modèles plus robustes en conservant la sépa-

ration worker. La sécurité des documents et la gouvernance des données devront être précisées en cas d'ouverture multi-tenant.

# Chapitre 4

## Études de cas

### 4.1 Préparation d'un module pédagogique

Un enseignant structure un chapitre en trois groupes, associe des quiz et des flashcards, et indexe un corpus PDF de référence. Le nœud AI produit des éléments de cours et des exercices, enrichis par des extraits pertinents.

### 4.2 Analyse documentaire et synthèse

Un analyste glisse un rapport PDF, interroge le nœud AI pour extraire des passages sur un thème précis et organise les résultats en frise chronologique et tableau de comparaison.

### 4.3 Exemple d'implémentation

---

**Algorithm 1** : Pipeline de génération de nœuds

---

**Input** : question, contexte textuel, chunks PDF

$\text{embedding\_q} \leftarrow \text{generateEmbedding}(\text{question})$

$\text{topK} \leftarrow \text{searchSimilarChunks}(\text{embedding\_q}, \text{chunks}, \text{embeddings}, 3)$

$\text{messages} \leftarrow \text{concat}(\text{contexte}, \text{topK})$

$\text{stream} \leftarrow \text{POST}(/api/groq, \text{messages})$

$\text{nœuds} \leftarrow \text{parse}(\text{stream})$

$\text{place}(\text{nœuds})$

---

## **4.4 Mesures d'impact**

### **4.4.1 Indicateurs quantitatifs**

Réduction du temps de préparation de contenus, augmentation du taux de réussite aux quiz, amélioration de la rétention via flashcards. Les mesures détaillées figurent en annexe.

### **4.4.2 Retours utilisateurs**

Les utilisateurs soulignent la clarté visuelle, la fluidité de manipulation et la pertinence des extraits RAG.

# Chapitre 5

## Perspectives et recommandations

### 5.1 Évolutions potentielles

Intégration d'une collaboration temps réel par CRDT, ajout d'une persistance chiffrée et d'un historique, optimisation des embeddings, et introduction d'un moteur de permissions fin.

### 5.2 Applications émergentes

Théâtres d'usage pressentis : ingénierie didactique, ateliers d'idéation, analyse réglementaire, formation continue et assistance à la décision visuelle.

### 5.3 Feuille de route

- Phase 1 : persistance locale et export/import structurés.
- Phase 2 : collaboration CRDT et gestion de versions.
- Phase 3 : optimisations IA et personnalisation avancée.

# Chapitre 6

## Glossaire

Terme	Définition
CRDT	Structures de données répliquées sans conflit, favorisant la convergence.
OT	Transformation opérationnelle pour la co-édition à faible latence.
RAG	<i>Retrieval-Augmented Generation</i> , génération enrichie par documents.
SSE	<i>Server-Sent Events</i> , flux d'événements unidirectionnels serveur→client.
Embedding	Projection vectorielle d'un texte en espace métrique.
Web Worker	Contexte d'exécution parallèle côté client pour calcul intensif.
React Flow	Bibliothèque de graphes et flux pour React.



## **Chapitre 7**

## **Bibliographie**

# Bibliographie

- [1] Marc Shapiro, Nuno Preguiça, Carlos Baquero, Marek Zawirski. Conflict-Free Replicated Data Types. INRIA Research Report RR-7687, 2011.
- [2] Chengzheng Sun, David Sun. Operation Transformation for Real-Time Collaborative Editing. CSCW, 2004.
- [3] Martin Kleppmann. Designing Data-Intensive Applications. O'Reilly, 2017.
- [4] Richard E. Mayer. Multimedia Learning. Cambridge University Press, 2009.
- [5] Shaaron Ainsworth. DeFT : A Conceptual Framework for Learning with Multiple Representations. Learning and Instruction, 2006.
- [6] Jeffrey Heer, Michael Bostock. Crowdsourcing Graphical Perception. CHI, 2010.
- [7] Colin Ware. Information Visualization : Perception for Design. Morgan Kaufmann, 2012.
- [8] Jakob Nielsen. Usability Engineering. Academic Press, 1993.
- [9] Ashish Vaswani et al. Attention Is All You Need. NeurIPS, 2017.
- [10] Tom B. Brown et al. Language Models are Few-Shot Learners. NeurIPS, 2020.
- [11] OpenAI. GPT-4 Technical Report. arXiv :2303.08774, 2023.
- [12] Yao-Hua Tsai et al. Multimodal Machine Learning. Foundations and Trends in Signal Processing, 2019.
- [13] Jean-Baptiste Alayrac et al. Flamingo : A Visual Language Model for Few-Shot Learning. arXiv :2204.14198, 2022.
- [14] Ben Shneiderman. Direct Manipulation for Comprehensible, Predictable, and Controllable User Interfaces. Proceedings of IUI, 1997.
- [15] Mark Weiser. The Computer for the 21st Century. Scientific American, 1991.
- [16] W3C. WebRTC 1.0 : Real-Time Communication Between Browsers. Recommendation, 2024.
- [17] WHATWG/W3C. Server-Sent Events. Living Standard, 2015.
- [18] Knut Sveidqvist. Mermaid : Generation of Diagrams from Text. 2015.
- [19] Mozilla. PDF.js Technical Overview. 2014.
- [20] React Flow Documentation. 2024.

# Annexe A

## Données brutes des benchmarks

### A.1 Protocole

Environnement Windows, navigateur moderne, indexation en worker, streaming HTTP. Les scénarios suivants ont été répétés sur séries contrôlées. Les timestamps bruts et mesures de latence sont fournis infra.

### A.2 Séries de mesures

Essai	Génération IA (ms)	Indexation 100 chunks (ms)	RAG top-3 (ms)	Drag (ms)
1	1512	8150	41	12
2	1398	8290	38	15
3	1476	8420	40	13
4	1550	8010	43	14
5	1422	8275	39	16
6	1488	8340	45	14
7	1531	8205	42	13
8	1467	8188	44	15
9	1495	8360	46	14
10	1430	8298	40	13

## **Annexe B**

### **Index**