RecyClique - Système RAG Intelligent : Dossier Complet Mis à Jour

Executive Summary

RecyClique va intégrer un système de base de connaissance intelligent permettant aux utilisateurs (admin, bénévoles, clients) d'interroger naturellement toutes les ressources documentaires via un chatbot omniprésent (web, Discord, etc.). Le système optimise automatiquement les coûts et la qualité des réponses grâce à des benchmarks continus, un routage intelligent des modèles LLM, et une gestion sophistiquée des providers et limites d'API.

Objectif principal: Interface ultra-simple en surface, optimisation sophistiquée en interne avec maîtrise totale des coûts et disponibilité.

1. Architecture Globale

1.1 Vue d'Ensemble

Le système se compose de 5 couches principales :

1. **Couche Données** : Sources documentaires (kDrive Infomaniak, Paheko, docs RecyClique)
2. **Couche Indexation** : LEANN pour la recherche sémantique
3. **Couche Intelligence** : Claude Agent SDK avec sub-agents spécialisés
4. **Couche Providers** : Gestion multi-fournisseurs LLM avec fallbacks intelligents
5. **Couche Interface** : Chatbot unique accessible partout + Admin Panel sophistiqué

1.2 Flux Simplifié

```
Utilisateur pose question naturelle
    ↓
Agent principal analyse et délègue
    ↓
Sub-agents spécialisés travaillent en parallèle
    ↓
LEANN recherche documents pertinents
    ↓
Router détecte complexité + vérifie disponibilité providers
    ↓
```

```
Sélectionne LLM optimal (coût/qualité/disponibilité)
    ↓
Gestion rate-limiting + fallback si quota dépassé
    ↓
Synthèse finale retournée à l'utilisateur
```

2. Composants Techniques Retenus

2.1 LEANN - Moteur de Recherche Sémantique

Pourquoi LEANN :

- Réduction stockage 97% (6GB vs 201GB pour alternatives)
- 100% local, zéro dépendance cloud
- Open-source gratuit
- Performance <2s pour recherche complexe
- Structure graphe HNSW pour rapidité

Rôle :

- Indexer tous les documents RecyClique
- Recherche sémantique rapide
- Retrieval pour RAG (Retrieval-Augmented Generation)

Déploiement :

- 2 containers Docker distincts :
  - Container 1 : Service FastAPI pour RecyClique interne
  - Container 2 : Serveur MCP pour accès Claude Desktop externe
- Partage du même index via volume Docker
- Embeddings via OpenAI API (pas de modèles locaux requis)

2.2 Claude Agent SDK - Orchestration Intelligente

Pourquoi Claude Agent SDK (vs alternatives) :

- Production-tested (utilisé par Claude Code, Cursor, JetBrains)
- Sub-agents natifs avec contextes isolés
- Gestion automatique tokens/contexte (évite bloat)
- Skills système (composants réutilisables)
- MCP intégré nativement
- Plus simple que LangGraph pour notre usage

Architecture Agent :

```
MainAgent (interface utilisateur)
├─ SearchAgent (recherche documents via LEANN)
├─ CreationAgent (génération contenu)
├─ AnalysisAgent (raisonnement multi-hop complexe)
└─ DiscordFormatterAgent (adaptation réponses Discord)
```

Principe Sub-agents :

- Chaque agent = fenêtre contexte propre
- Travail parallèle possible
- Délégation automatique selon intent utilisateur
- Pas de nesting infini (sub-agents ne créent pas d'autres sub-agents)

2.3 Synchronisation Documents - kDrive Infomaniak

Solution retenue : rclone + WebDAV

Pourquoi :

- kDrive utilise protocole WebDAV standard
- rclone mature, fiable, supporte sync incrémental
- Documents OnlyOffice = fichiers `.docx` standards (indexables directement)
- Pas de dépendance API custom Infomaniak

Workflow :

1. Cron job VPS sync kDrive → dossier local (/data/docs)
2. Détection changements (rclone optimisé)
3. Trigger rebuild index LEANN si modifications
4. Fréquence : toutes les 6h ou daily (configurable)

Avantages vs alternatives :

- Cache local = performance indexation
- Pas de latence réseau temps réel
- Respecte limitations compte gratuit (60 requêtes/min API)
- Simple à maintenir

3. Routage Intelligent LLM

3.1 Principe

Au lieu d'utiliser toujours le même modèle (coûteux), le système analyse automatiquement chaque requête et choisit le modèle optimal selon complexité, disponibilité provider, et contraintes budget.

Exemple concret :

- Question simple "Où trouve-t-on le règlement intérieur ?" → Haiku (rapide, pas cher)
- Question moyenne "Résume les procédures de tri textile" → Sonnet (équilibré)
- Question complexe "Compare nos pratiques upcycling avec nouvelles normes EU et recommande changements" → Opus (puissant)
- Si OpenAI down/quota dépassé → Fallback Groq ou Claude direct

3.2 Proxy Multi-Modèles + Gestion Providers

Base technique retenue : Adaptation de `fuergaosi233/claude-code-proxy`

Pourquoi ce repo :

- 1.6k stars, communauté active
- OpenAI-compatible natif
- Support Ollama (futurs modèles locaux)
- FastAPI async/streaming
- Bien documenté

Améliorations prévues :

- Router intelligence (classification complexité)
- Configuration UI (inspirée `zimplexing/claude-code-proxy-enhance`)
- Multi-provider (inspiré `ujisati/claude-code-provider-proxy`)
- **NOUVEAU : Gestion providers + rate-limiting + fallback chain**

3.3 Cascade LLM par Complexité + Disponibilité

Niveau Simple :

- Modèles : Haiku, Mistral 7B, GPT-4o-mini
- Latence : <1s
- Coût : $0.001-0.003/requête

Niveau Medium :

- Modèles : Sonnet, Claude, GPT-4o
- Latence : 1-2s
- Coût : $0.005-0.01/requête

Niveau Complex :

- Modèles : Opus, GPT-4, Claude Opus
- Latence : 2-5s
- Coût : $0.02-0.05/requête

Mécanisme fallback sophistiqué :

1. Essai provider principal (ex: OpenAI tier simple)
2. Vérifie quota/rate-limit disponible
3. Si quota insuffisant → provider alternate (ex: Groq)
4. Si confidence réponse < seuil → escalade tier modèle
5. Log complet tentatives + provider utilisé + fallbacks déclenché

4. Système Gestion des Providers LLM

4.1 Concept

Panneau centralisé permettant à un admin (non-technique) de gérer tous les providers LLM, leurs limitations, coûts, quotas, et conditions d'utilisation.

4.2 Gestion Credentials et Tiers

Interface Admin - Section Providers :

Configuration par Provider :

- **Provider Name** : OpenAI, Anthropic, OpenRouter, Groq, TogetherAI, etc.
- **API Key** : Champs sécurisés (encrypted storage)
- **Tier** : Free, Pro, Enterprise
- **Status** : Active/Inactive/Testing
- **Coût par 1k tokens** : Input/Output séparé
- **Rate Limits** :
  - Requêtes par minute (RPM)
  - Requêtes par jour (RPD)
  - Tokens par minute (TPM)
  - Tokens par jour (TPD)
- **Notes** : Conditions spéciales, contrat, date expiration clé

Exemple configuration Free Tier OpenRouter :

```
Provider: OpenRouter
Tier: Free (150 requêtes/jour, 200k tokens/jour max)
```

```
Cost: $0 (free)
Rate Limits: 150 RPD, 1000 TPM
Status: Active
Models Available: 300+
```

Exemple configuration PayAsYouGo OpenAI :

```
Provider: OpenAI
Tier: Pay-As-You-Go
Cost: Input $0.003/1k, Output $0.006/1k
Rate Limits: 10000 RPM, 200000 TPM
Status: Active
Budget Cap: $100/month (alert si dépassé)
Models: GPT-4o, Sonnet, Haiku
```

4.3 Base de Connaissance Live - Veille IA

Nouveau composant : "Provider Knowledge Hub"

Accès : Chatbot spécialisé sur page admin "Providers & Contracts"

Capacités du chatbot IA :

- Questions naturelles sur providers
- Recherche conditions utilisation actualisées
- Procédures obtention clés API
- Comparaison pricing real-time
- Alertes changements ToS
- Recommandations optimisation coûts
- Exemple questions :
  - "Comment augmenter quota OpenAI free tier ?"
  - "Quelle est la différence tarifaire Sonnet vs Haiku aujourd'hui ?"
  - "OpenRouter a-t-il des nouvelles restrictions free tier ?"
  - "Quel provider offre meilleures conditions pour gros volumes ?"

Source data :

- Scraping auto documentation officielle providers
- Alert système changements détectés
- Import manuel updates critiques
- Historique (facilite audit)

4.4 Temporisation et Estimation Tokens Live

Dashboard Provider Status (temps réel) :

```
Provider          Status    Quota Today   Used    Remaining   RPM Left  Next Reset
──────────────────────────────────────────────────────────────────────────────────
OpenAI            ✓ Active  200k tokens   47.2k   152.8k ↓    8/10      23:14
OpenRouter        ✓ Active  150 req/day   89      61 ↓        7/100     23:45
Groq              ✓ Active  Unlimited     5.3k    ∞           ∞         -
TogetherAI        ⚠ Warn    50k tokens    45.8k   4.2k ⚠      2/50      06:32
Anthropic         ✗ Testing                                   5/10      -
Local Ollama      ✓ Active  Unlimited     3.2k    ∞           ∞         -
```

Logique Temporisation :

1. Requête arrive → Router vérifie quota temps réel
2. Calcule tokens estimés
3. Détermine providers disponibles
4. Si multiple options : choisit optimal (coût/latence)
5. Si quota dépassé → attente (queue) ou fallback
6. Log : provider choisi, tokens consommés, fallbacks déclenché

Système Queue Intelligent :

- Requêtes en attente si quota dépassé
- Priorisation : simple avant complexe
- Auto-retry quand quota se libère (minuit, reset)
- Notification utilisateur si délai important

4.5 Fallback Automatique Intelligent

Chaîne Fallback Configurable par Skill :

Exemple SearchAgent :

```
Hiérarchie : OpenAI (Haiku) → OpenRouter (Mistral) → Groq → Local Ollama

Si OpenAI quota dépassé :
  → Try OpenRouter (check quota)
     → Si OK, utilise Mistral 7B
     → Qualité similaire, coût +0.1%
     → Si OpenRouter aussi saturé → Groq
```

Exemple CreationAgent :

```
Hiérarchie : OpenAI (Sonnet) → Anthropic (Claude) → OpenRouter (Mix models)

Si OpenAI down :
```

```
   → Try Anthropic direct
      → Si pas de clé Anthropic → OpenRouter
```

Fallback Branching per Requête :

```
Déterminer complexité estimée
    ↓
Provider 1 (optimal pour coût/perf)
  ├─ Check quota + rate limits
  ├─ Si OK → Call + monitor response time
  ├─ Si down/quota dépassé → Provider 2
  │  └─ Repeat check
  ├─ Timeout long (>10s) → Provider 2 parallel
  └─ Si échec total → Fallback local Ollama (mode dégradé)
```

Logging Fallback :

- Chaque tentative = log (provider, status, tokens estimés)
- Alertes si fallback fréquent (signe problème provider)
- Statistiques fallback par jour/semaine (insights optimization)

5. Benchmarking Avancé + Human Feedback Arena

5.1 Dimensions Benchmarking Étendues

Dimension 1 : Performance Standard (existant)

- Coût (input/output tokens)
- Latence TTFT/ITL
- Throughput tokens/sec

Dimension 2 : Qualité Réponse (LLM-as-Judge)

- Précision factuelles
- Cohérence logique
- Complétude réponse
- Pertinence query

Dimension 3 : NOUVEAU - Capacités Agentiques

- Autonomie décision (quand appeler tools, quand s'arrêter)
- Cohérence instructions multi-step
- Format sortie standard (XML/JSON/Markdown)
- Gestion erreurs gracieuse

Dimension 4 : NOUVEAU - Arena Human Feedback

- Humains évaluent directement (0-10 score)
- Comparaison pairwise (Modèle A vs B)
- Feedback contexte (tâche, complexité, domaine)
- Accumulation dataset fine-tuning futur

5.2 Capacités Agentiques par Complexité

Benchmarking Agentic = Évaluation selon niveau complexité tâche

Tâches Niveau 1 (Simple) :

- Single tool call (recherche LEANN)
- Pas décision logique
- Modèle attendu : Haiku, Mistral 7B
- Scores attendus : coût bas OK, latence <2s, format output simple

Tâches Niveau 2 (Medium) :

- 2-3 tool calls séquentiels
- Décisions basiques (if/then simples)
- Modèle attendu : Sonnet
- Scores attendus : output structuré (JSON valide), coherence chaîne logic, pas d'hallucinations

Tâches Niveau 3 (Complex) :

- 4+ tool calls, boucles, conditions imbriquées
- Raisonnement multi-hop
- Gestion erreurs (retry, fallback)
- Modèle attendu : Opus, GPT-4
- Scores attendus : planning sophistiqué, recovery errors élégant, output XML/JSON complex

5.3 Évaluation Format Sortie

Standards de Sortie Attendus :

Pour SearchAgent :

```
{
  "query_original": "...",
  "complexity_detected": "simple|medium|complex",
  "results": [
    {
      "source": "doc_id",
      "snippet": "...",
      "relevance_score": 0.95,
      "url": "..."
```

```
    }
  ],
  "metadata": {
    "timestamp": "ISO8601",
    "provider_used": "OpenAI",
    "tokens_used": 245,
    "fallback_triggered": false
  }
}
```

Benchmark métriques format :

- JSON schema compliance (score 0-100)
- Required fields présents (score 0-100)
- Nested structure validity (score 0-100)
- Field data types correctness (score 0-100)
- Overall format score = moyenne

Alertes si format dégradé :

- Modèle retourné texte brut au lieu JSON → rouge flag
- Parseable but non-standard → warning
- Missing fields → dégradation score
- Permet détecter régressions modèles

5.4 Arena Human Feedback

Interface Admin Section : "Benchmark Arena"

Workflow :

1. **Setup Arena Session :**
   - Choisir skill à benchmarker
   - Charger 10-50 test cases (dataset curated)
   - Sélectionner 3-5 modèles à comparer
   - Lancer exécution parallèle
2. **Live Comparison :**
   - Interface affiche réponses côte-à-côte
   - Humain évalue sur échelle 0-10 chaque réponse
   - Comparaisons pairwise (A vs B vs C)
   - Critères personnalisables par skill
3. **Feedback Capture :**
   - Score numérique + commentaires texte

- Temps évaluation (humain rapide ou lent ?)
- Confidence score évaluateur
- Tags issues détectées ("hallucination", "format error", "incomplete")

4. **Agrégation Results :**
   - Calcul moyenne scores par modèle
   - Pairwise victory matrix
   - Tags issues distribution
   - Tendances temporelles (si répétitions)
5. **Export Dataset :**
   - Export évals humains (format standard)
   - Seed pour fine-tuning models futurs
   - Historique complet (audit trail)

Exemple Arena Session :

```
Skill: document_generation
Test cases: 20 creation requests
Models: Sonnet vs Opus vs GPT-4o
Evaluators: 2 humains RecyClique

Résultats:
Sonnet: 7.2/10 avg (quick, OK quality)
Opus:   8.8/10 avg (excellent)
GPT-4o: 8.1/10 avg (good, pricey)

Issues detectées:
Sonnet: 2x hallucinations mineures
Opus:   Format XML perfect, best coherence
GPT-4o: 1x timeout (latency issue)

Conclusion: Opus worth extra cost for this task
```

5.5 Workflow Benchmarking Complet

Étape 1 : Collecte Automatique

- Scheduler lance benchmarks (quotidien, hebdo, on-demand)
- 50-100 cas test par skill
- Exécution parallèle 5-10 modèles
- Collection métriques auto (coût, latence, tokens, format validity)

Étape 2 : LLM-as-Judge Evaluation

- Petit modèle rapide évalue chaque réponse
- Rubriques scoring custom par skill

- Détecte hallucinations, erreurs format
- Score 0-100 qualité

Étape 3 : Human Arena (Optionnel)

- Admin invite human evaluators si important
- Comparative feedback collecté
- Valide/invalide conclusions LLM-as-judge
- Accumule dataset fine-tuning

Étape 4 : Análise + Recommandations

- Calcul Pareto frontier (coût vs qualité)
- Détection changements vs baseline
- Suggestions routing updates
- Estimations economie si changements

Étape 5 : Approval + Deployment

- Affiche changements suggérés admin
- Comparaison coûts (ancien vs nouveau routing)
- Humain approve/reject
- Commit vers routing DB versioned
- Déploiement automatic production

6. Gestion Coûts et Analyse Avancée

6.1 Tableau de Bord Coûts Complexe

Section Admin : "Cost Analytics & Forecasting"

Vue Globale :

```
Dépenses Septembre 2025: $487.23
├─ Comparé Août: +12% (+$51.50)
├─ Trend: ⬆ +2% semaine/semaine
└─ Forecast Octobre: $530-560 (if pattern continues)

Breakdown par Agent:
├─ SearchAgent: $120.45 (25%) ← Trend stable
├─ CreationAgent: $290.30 (60%) ← Trend ⬆ +15%
└─ AnalysisAgent: $76.48 (15%) ← Trend stable

Breakdown par Provider:
├─ OpenAI: $280 (57%) [10 millions tokens]
├─ Anthropic: $145 (30%) [5 millions tokens]
├─ OpenRouter Free: $0 (0%) [150 req/jour utilisées]
```

```
└─ Groq Free: $0 (0%) [Unlimited]

Breakdown par Modèle:
├─ Haiku: $12 (2.5%) [320k tokens]
├─ Sonnet: $180 (37%) [2.8M tokens]
├─ Opus: $220 (45%) [1.2M tokens]
└─ GPT-4o: $75 (15%) [1.1M tokens]
```

6.2 Scénarios "What-If" pour Optimisation

Interface Interactive "Cost Simulator"

Permet admin explorer alternatives sans risques :

Scénario 1 : "Si on remplace Opus par Sonnet pour AnalysisAgent ?"

```
Impact Estimé:
├─ Coût: $76.48 → $38 (-50%)
├─ Latence: +1.2s (acceptable ?)
├─ Qualité: -8% (vs reference)
└─ Verdict: Savings $38/mois, mais qualité concerns

Alternative: Hybrid
├─ Simple analysis → Sonnet
├─ Complex analysis → Opus (si > 3 sub-questions)
└─ Estimé: $55 (-28%), latence+0.3s, qualité -2%
```

Scénario 2 : "Si on achète plan Anthropic paid tier ?"

```
Option 1: Stay Pay-As-You-Go
├─ Coûts: $145/mois
└─ Commitement: None

Option 2: Buy $500 credit/mois Anthropic
├─ Coûts: $500
├─ Credits utilisés/mois: ~$145 current
├─ Surplus: $355 (expires fin mois)
├─ Effective rate: -20% si tokens consommés

Option 3: Upgrade OpenAI to Pro $20/mois
├─ Coûts: $20 (plan) + API usage ($280)
├─ Benefit: 10x higher rate limits
├─ Si current load saturating → latency benefit ✓

Recommendation: Option 3 (upgrade OpenAI)
├─ Reason: Rate limit relief needed (fallbacks triggered 8% requests)
├─ ROI: $20 pays itself via reduced latency + fallback overhead
└─ Approve: Y/N buttons
```

Scénario 3 : "Coûts par Use Case End-User"

```
Search query type A: avg $0.005/query
├─ Models: Haiku primary, Sonnet fallback
├─ Volume/day: 200
├─ Daily cost: $1
└─ Monthly: $30

Creation request type B: avg $0.18/request
├─ Models: Sonnet primary, Opus complex
├─ Volume/day: 15
├─ Daily cost: $2.70
└─ Monthly: $81

Analysis request type C: avg $0.35/request
├─ Models: Opus always
├─ Volume/day: 5
├─ Daily cost: $1.75
└─ Monthly: $52.50
```

6.3 Comparaisons Pricing Détaillées

Table Interactive : Coût par Provider/Modèle (updaté temps réel)

Model	Provider	Cost/1M Input	Cost/1M Output	Qual.	Latency	Notes
Haiku	OpenAI	$0.80	$4	**	<100ms	Low cost
Mistral 7B	OpenRouter	FREE	FREE	**	<300ms	Rate limite
Sonnet	Anthropic	$3	$15	***	500ms	Balanced
GPT-4o-mini	OpenAI	$0.15	$0.60	**	150ms	Good value
GPT-4o	OpenAI	$2.50	$10	****	800ms	Premium
Opus	Anthropic	$15	$75	****	1200ms	Best quality
Groq Llama	Groq	FREE	FREE	**	<50ms	Ultra fast,

Avec Simulation Cost Comparison :

```
Pour 1000 requêtes SearchAgent (Haiku):
├─ OpenAI: $0.80 (est coût réel si on avait volume)
├─ OpenRouter: FREE (150/jour limité)
├─ Groq: FREE (unlimited)
└─ Meilleur: Groq ou OpenRouter (free)

Pour 100 requêtes CreationAgent (Sonnet):
├─ Anthropic direct: $3.15
├─ OpenRouter: $3.10 (route à meilleur modèle)
├─ OpenAI GPT-4o: $2.65 (cheaper, less quality)
└─ Meilleur qualité/coût: OpenRouter ou Anthropic

Si 1M tokens/mois utilisés (scénario forte demande):
├─ OpenAI Pro plan: $350
├─ Anthropic $500 credit: $500
├─ OpenRouter: ~$280 à usage réel
├─ Groq Free: $0 (unlimited, latency <50ms ⚠)
└─ Stratégie: Groq pour non-critical, OpenRouter backup
```

7. Interface Admin - Panel de Gestion Complet

7.1 Vision Utilisateur Admin

Interface web unique centralisant toute la gestion du système intelligent, sans nécessiter connaissances techniques approfondies.

7.2 Sections Principales

Section 1 : Agents & Skills

- Liste hiérarchique agents → skills
- Statut (actif/inactif)
- Métriques utilisation (requêtes/jour)
- Configuration individuelle par skill

Section 2 : Profils Benchmark

- Historique benchmarks par skill
- Visualisation coût/latence/qualité par modèle
- Comparatifs temporels (graphiques tendances)
- Dernière exécution + prochaine scheduled
- **NOUVEAU : Arena Human Feedback button**

Section 3 : Configuration Routing

- Vue temps réel : quel modèle pour quelle complexité
- Fallback chains explicites
- Coût estimé par configuration
- Boutons : Auto-sync benchmarks ou Override manuel

Section 4 : Metrics & Monitoring

- Dashboard financier (coût total, par agent, tendances)
- Graphiques latence moyenne
- Token efficiency
- Quality trends
- Alertes dégradation

Section 5 : Automation Benchmarks

- Calendrier exécutions (schedule cron-like UI)
- Gestion datasets test (upload/edit)
- Choix modèles à tester

- Configuration rubrique qualité LLM-as-judge
- **NOUVEAU : Arena Human Feedback setup**

Section 6 : Provider Management (NOUVELLE)

- **Subsection A : Provider Configuration**
  - Liste providers : OpenAI, Anthropic, OpenRouter, Groq, TogetherAI
  - Par provider :
    - API key field (masked)
    - Tier status (Free/Pro/Enterprise)
    - Active/Inactive toggle
    - Coûts input/output
    - Rate limits (RPM/TPM/RPD/TPD)
    - Status indicator (green/yellow/red)
    - Last verified date
- **Subsection B : Knowledge Base Live**
  - Chatbot IA contextuel
  - Questions : conditions ToS, pricing, quotas, procédures clés
  - Réponses sourced documentation actuelle
  - Alertes auto si changements majeurs
- **Subsection C : Token Quota Monitor (Real-time)**
  - Table temps réel quota utilisé/restant
  - Graphiques consommation par provider
  - Alertes si quota bas (<20% remaining)
  - Estimations reset time
- **Subsection D : Fallback Configuration**
  - Par skill : chaîne fallback customizable
  - Test fallback chain
  - Monitor fallback triggers (stats)

Section 7 : Cost Analytics & Forecasting (NOUVELLE)

- **Vue Globale Dépenses :**
  - Coûts mois en cours
  - Comparaison mois précédent (% change)
  - Trend line (projection mois prochain)
  - Breakdown agent/provider/modèle
- **Cost Simulator (What-If) :**

    - Scénarios à tester (replace model X with Y, upgrade plan, etc.)
    - Impact estimé (coût, latence, qualité)
    - Approve button si gains intéressants
- **Detailed Pricing Comparison :**
    - Table interactive modèles
    - Cost per use case
    - Scenarios: low volume, medium volume, high volume
    - Recommendations

Section 8 : Pending Changes

- File notifications changements suggérés
- Impact prévu (coût, qualité, latence)
- Boutons Approve/Reject
- Historique décisions

7.3 Expérience Utilisateur Cible

Admin non-technique peut :

- Voir en temps réel performance système
- Comprendre où part le budget
- Tester scénarios avant committing
- Approuver optimisations en 1 clic
- Rollback si problème
- Lancer benchmarks ponctuels
- Ajouter/configurer providers
- Consulter knowledge base conditions ToS
- Monitorer quotas real-time
- Étudier alternatives coûts

Sans jamais :

- Toucher code
- Éditer fichiers config
- SSH sur serveur
- Comprendre architecture technique

8. Intégration Stack RecyClique

8.1 Stack Docker Complète

Services déployés :

```
PostgreSQL (données Paheko + routing config)
Redis (cache sessions + rate limiting)
Paheko (backend association)
Ollama (optionnel, embeddings locaux futurs)
LEANN Service FastAPI (recherche interne)
LEANN MCP Server (accès Claude externe)
RecyClique API (backend principal)
RecyClique Frontend (interface web)
Discord Bot (intégration serveur)
Knowledge Sync Service (kDrive → local)
Admin Panel (gestion benchmarks/routing/providers)
Benchmarking Engine (orchestration auto)
Provider Quota Monitor (real-time status)
Knowledge Base Chatbot (veille IA providers)
```

Réseau :

- Traefik reverse proxy (gestion sous-domaines)
- Isolation containers via network Docker
- Volumes partagés : index LEANN, documents sync

8.2 Sous-domaines Traefik

Routes proposées :

- `recyclic.jarvos.eu` → Frontend principal
- `api.recyclic.jarvos.eu` → Backend FastAPI
- `admin.recyclic.jarvos.eu` → Panel admin benchmarks
- `paheko.recyclic.jarvos.eu` → Paheko interface

Note : MCP server pas exposé web (stdio local uniquement pour Claude Desktop)

8.3 Intégration Discord

Bot Discord RecyClique :

- Container dédié avec discord.py
- Appelle RecyClique API (FastAPI)
- Commandes : `!ask`, `!search`, `!help`
- Formatting réponses Discord (embeds)
- Contexte channel/user transmis agents

Expérience utilisateur :

- Même intelligence que web
- Réponses formatées Discord
- Pas de limitation fonctionnelle

9. Workflow Complet Utilisateur Final

9.1 Scénario Simple

User : "Où se trouve la procédure de tri textile ?"

Système :

1. MainAgent reçoit question
2. Détecte intent : recherche document
3. Délègue SearchAgent
4. SearchAgent appelle LEANN search
5. Router détecte complexité simple
6. Vérifie disponibilité providers (OpenAI Haiku quota OK)
7. Selectionne Haiku (rapide, $0.001)
8. LEANN retourne 3 documents pertinents
9. Haiku génère réponse courte + liens
10. Log : provider=OpenAI, model=Haiku, tokens=120, cost=$0.0008
11. Total : <2s, coût $0.001

User voit : Réponse instantanée avec sources

9.2 Scénario Complexe

User : "Compare nos pratiques valorisation textile avec normes UE 2025 et propose plan action 6 mois"

Système :

1. MainAgent détecte complexité élevée
2. Délègue AnalysisAgent
3. AnalysisAgent décompose en sub-questions :
    - Quelles sont nos pratiques actuelles ?
    - Quelles sont normes UE 2025 ?
    - Quels gaps identifier ?
    - Quel plan action réaliste ?

4. Pour chaque sub-question :
    - LEANN search documents
    - Paheko context (données opérationnelles)
5. Router vérifie quotas :
    - Questions 1-2 → OpenAI Sonnet OK
    - Synthèse finale → Anthropic Opus quota limitée
    - Fallback : Si Anthropic saturé → OpenRouter route GPT-4o
6. AnalysisAgent agrège + génère plan structuré
7. Quality eval : LLM-as-judge score 8.7/10
8. Logs : Provider 1=OpenAI (Sonnet), Provider 2=OpenRouter (GPT-4o fallback)
9. Total : ~8s, coût $0.15

User voit : Plan action détaillé, sourcé, structuré

9.3 Scénario Création

User : "Crée un guide bénévole pour accueil nouveaux arrivants"

Système :

1. MainAgent délègue CreationAgent
2. CreationAgent :
    - LEANN search procédures existantes
    - Paheko fetch infos structure
3. Router tests disponibilité :
    - Anthropic Sonnet : quotas OK
    - OpenAI GPT-4o : pas cheaper option pour qualité
4. Route : Anthropic Sonnet (coût $0.10, quality 8.5/10)
5. Génération guide structuré JSON
6. Format validation : ✓ JSON valide
7. Option : stockage automatique Paheko docs
8. Total : ~5s, coût $0.10

User voit : Document prêt à l'emploi, modifiable

10. Avantages Clés du Système

10.1 Pour Utilisateurs Finaux

- Interface unique naturelle (chat)
- Pas besoin mémoriser outils/workflows
- Réponses instantanées documentées
- Accessible web + Discord
- Même expérience partout

10.2 Pour Admins RecyClique

- Optimisation coûts **automatique + simulable**
- Visibilité complète dépenses IA + prévisions
- **Gestion providers centralisée**
- **Quotas monitored real-time + fallbacks intelligents**
- Contrôle qualité via benchmarks (humans + AI)
- Configuration simple sans code
- Rollback sécurisé
- Audit trail complet

10.3 Pour Architecture Technique

- Scalable (ajout agents/skills facile)
- Modulaire (composants indépendants)
- Reproductible (Docker stack)
- Observabilité native
- Multi-instance possible
- Open-source maximal (réduction vendor lock-in)
- **Resilient vs provider downtime**

11. Timeline Déploiement

Phase 1 : Fondations (Semaines 1-3)

- Setup LEANN containers (FastAPI + MCP)
- Configuration OpenAI embeddings
- Sync kDrive → local (rclone)
- Index initial documents RecyClique
- Tests recherche basiques

Phase 2 : Intelligence (Semaines 4-6)

- Intégration Claude Agent SDK
- Création MainAgent + 3 sub-agents essentiels
- Tests délégation/routing
- Intégration LEANN dans agents

Phase 3 : Routing Basique (Semaines 7-8)

- Adaptation proxy `fuergaosi233/claude-code-proxy`
- Router heuristique simple (complexité)
- OpenRouter connection
- Tests cascade LLM basique

Phase 4 : Providers Management (Semaines 9-11)

- **NOUVEAU : Interface gestion providers**
- **Quota monitoring real-time**
- **Fallback chains configuration**
- **Rate limit enforcement**
- Tests multi-provider scenarios

Phase 5 : Benchmarking Avancé (Semaines 12-16)

- Framework DeepEval setup
- Test harness par agent/skill
- **NOUVEAU : Agentic capabilities eval**
- **NOUVEAU : Format output validation**
- **NOUVEAU : Arena human feedback interface**
- Baselines benchmarks initiaux
- Routing DB (PostgreSQL)
- Dashboard métriques MVP

Phase 6 : Automation + Analytics (Semaines 17-21)

- Scheduler benchmarks automatiques
- Routing decision engine
- **NOUVEAU : Cost analytics dashboard**
- **NOUVEAU : Provider knowledge base chatbot**
- Interface admin UI complète

- Workflow supervised approvals
- Slack/email notifications

Phase 7 : Intégrations (Semaines 22-24)

- Bot Discord
- Widget frontend chatbot
- Tests end-to-end
- Documentation utilisateur

Phase 8 : Production (Semaine 25+)

- Déploiement Traefik
- Monitoring alertes
- Fine-tuning routing
- Formation équipe

Total estimation : 6-7 mois pour système complet production-ready

MVP utilisable : 10-11 semaines (Phases 1-3, routing basique)

MVP + Providers Management : 14-15 semaines (Phases 1-4, gestion providers complète)

12. Risques et Mitigations

Risque 1 : Coûts LLM incontrôlés

Mitigation :

- Budgets plafonds par agent/skill
- Alertes dépassement temps réel
- Benchmarks continus optimisation
- Fallback modèles moins chers
- **Rate limiting enforced per provider**
- **Cost simulator for what-if analysis**

Risque 2 : Qualité réponses dégradée

Mitigation :

- LLM-as-judge monitoring continu
- **Human arena feedback validation**
- Thresholds qualité minimum

- Escalation automatique si low confidence
- **Agentic capability monitoring**
- Feedback utilisateurs tracking

Risque 3 : Complexité maintenance

Mitigation :

- UI admin non-technique
- Documentation exhaustive
- Rollback one-click
- **Provider knowledge base chatbot assist**
- Monitoring proactif

Risque 4 : Dépendance providers externes

Mitigation :

- Multi-provider (OpenRouter + OpenAI + Anthropic + Groq)
- Fallbacks configurés
- Option Ollama local (backup complet)
- Pas de vendor lock-in architecture
- **Real-time quota monitoring + alerts**

Risque 5 : Sync documents kDrive échoue

Mitigation :

- rclone retry automatique
- Alertes échec sync
- Logs détaillés
- Fallback index ancien (stale data OK temporairement)

Risque 6 : Arena Human Feedback bias/unreliability

Mitigation :

- Multiple evaluators consensus
- Confidence scoring per evaluation
- Historical reliability tracking
- Automatic flagging outliers
- Training guidelines evaluators

13. Évolutions Futures Envisageables

Court Terme (6-12 mois)

- Fine-tuning modèles custom RecyClique
- Embeddings locaux (Ollama) pour réduction coûts
- Multi-langues (FR/EN/DE/IT)
- Intégration Slack/Teams supplémentaire
- **Auto-escalation thresholds learning from benchmarks**

Moyen Terme (1-2 ans)

- Knowledge graph sémantique (relations explicites)
- Agents proactifs (suggestions non sollicitées)
- Génération automatique procédures
- A/B testing variants agents
- **Predictive scaling (forecast future costs/usage)**

Long Terme (2+ ans)

- Multi-tenancy (autres ressourceries)
- Marketplace skills RecyClique
- Fédération instances distribuées
- IA générative création visuels/vidéos

Annexes Techniques

A1. Stack Technologique Complète

Backend :

- Python 3.11+
- FastAPI (API framework)
- Claude Agent SDK (orchestration)
- LEANN (RAG/search)
- PostgreSQL (données + routing config)
- Redis (cache sessions + rate limiting)
- TimescaleDB (time-series benchmarks)

Frontend :

- React/Vue (interface web)

- Vite (bundler)
- TailwindCSS (styling)

Infrastructure :

- Docker + Docker Compose
- Traefik (reverse proxy)
- rclone (sync kDrive)
- Nginx (static files)

LLM/AI :

- OpenRouter (multi-model gateway)
- OpenAI API (embeddings + models)
- Anthropic Claude (agents)
- Groq API (low-latency)
- TogetherAI (open-source models)
- DeepEval (benchmarking)

Monitoring :

- Grafana (dashboards)
- Prometheus (metrics)
- Sentry (error tracking)
- Custom admin UI (benchmarks + providers + costs)

A2. Volumes Docker Critiques

- `leann_index/` : Index LEANN persistant (6-10GB)
- `postgres_data/` : Données Paheko + routing + benchmarks (variable)
- `docs_sync/` : Documents kDrive synchronisés (5-20GB)
- `redis_data/` : Cache sessions (<<1GB)
- `ollama/` : Modèles locaux optionnels (10-50GB si activé)
- `benchmark_results/` : Archives résultats benchmarks (1-5GB/année)

A3. Variables Environnement Sensibles

```
OPENAI_API_KEY (embeddings + models)
ANTHROPIC_API_KEY (Claude agents)
OPENROUTER_API_KEY (multi-models gateway)
GROQ_API_KEY (low-latency fallback)
TOGETHERAI_API_KEY (open-source models)
POSTGRES_PASSWORD (database)
DISCORD_BOT_TOKEN (Discord integration)
KDRIVE_WEBDAV_USER (rclone sync)
```

```
KDRIVE_WEBDAV_PASSWORD (rclone sync)
ADMIN_JWT_SECRET (admin panel auth)
SLACK_WEBHOOK_URL (notifications)
```

Gestion : Docker secrets + .env gitignored + HashiCorp Vault (production)

A4. Endpoints API Principaux

RecyClique API :

- `POST /api/chat/ask` : Chat principal
- `POST /api/search` : Recherche LEANN directe
- `GET /api/agents` : Liste agents disponibles
- `POST /api/benchmark/run` : Lancer benchmark manuel
- `GET /api/providers/status` : Status temps réel providers
- `POST /api/providers/config` : Update provider config
- `GET /api/costs/analysis` : Analytics dépenses
- `POST /api/costs/simulate` : What-if scenarios

LEANN Service :

- `POST /search` : Recherche sémantique
- `POST /index/rebuild` : Rebuild index
- `GET /health` : Health check

Admin Panel :

- `GET /admin/metrics` : Dashboard data
- `GET /admin/benchmarks/latest` : Derniers résultats
- `POST /admin/benchmarks/arena` : Setup arena session
- `GET /admin/benchmarks/arena/results` : Arena results
- `POST /admin/routing/update` : Approve changement routing
- `GET /admin/costs/analysis` : Analyse coûts
- `POST /admin/costs/simulate` : Simulator what-if
- `POST /admin/providers/add` : Ajouter provider
- `GET /admin/providers/list` : Liste providers
- `POST /admin/providers/update` : Update provider config
- `GET /admin/knowledge/base` : Veille providers

Knowledge Base Chatbot (Admin) :

- `POST /kb/ask` : Questions conditions ToS, pricing, etc.
- `GET /kb/history` : Historique interactions

A5. Configuration Traefik Singulière

Particularité MCP :
Le serveur MCP LEANN ne doit **pas** être exposé via Traefik. Communication stdio locale uniquement pour Claude Desktop. Si exposition nécessaire (admin externe), utiliser authentification forte + VPN/Tailscale.

Règles routing spéciales :

- Admin Panel : Authentification JWT avant proxy (données sensibles coûts/benchmarks)
- Pas d'accès public même temporaire
- IP whitelist recommandé production
- Rate limiting Traefik : 1000 req/min max admin

A6. Modèles Proxy Compatible OpenAI

Repos GitHub Recommandés :

Repo	Stars	Focus	Adapté
fuergaosi233/claude-code-proxy	1.6k	Model mapping, cascade	✅ Base
ujisati/claude-code-provider-proxy	800	Multi-provider routing	✅ Inspire
zimplexing/claude-code-proxy-enhance	600	Web UI configuration	✅ Inspire
1rgs/claude-code-proxy	400	Lightweight proxy	✓ Alternative
kiyo-e/claude-code-proxy	300	Simple wrapper	✓ Reference

Recommandation Fusion : Base fuergaosi233 + ajouter :

- UI zimplexing pour admin
- Multi-provider ujisati pour fallback chains
- Rate limiting custom pour providers
- Quota monitoring real-time

Conclusion

Ce système représente une approche innovante combinant simplicité utilisateur et sophistication technique. L'architecture modulaire permet déploiement progressif (MVP en 2-3 mois, complet en 6-7 mois) tout en garantissant scalabilité future.

Différenciations clés :

1. **Gestion sophistiquée providers** : Multi-providers avec fallbacks intelligents, quotas monitorés real-time
2. **Benchmarking avancé** : Agentic capabilities évaluation, human feedback arena, format output validation

3. **Analytics coûts** : What-if simulators, forecasting, breakdown granulaire, recommandations automation
4. **Interface admin non-technique** : Knowledge base IA live, provider management centralisé, approval workflows simples
5. **Resilience optimale** : Fallbacks automatiques, queue management, graceful degradation

RecyClique disposera ainsi d'un assistant intelligent évolutif, adapté spécifiquement aux besoins ressourcerie, tout en restant maître de ses coûts (multi-providers free/paid mix), qualité de service (benchmarks humains + IA), et résilience (fallbacks chaînes complètes).

Le système est conçu pour être :

- Accessible : UI admin sans code
- Économe : Optimisation coûts automatique
- Fiable : Multi-providers + fallbacks
- Évolutif : Architecture modulaire
- Transparent : Audit trail + analytics complets

Document Généré

Ce dossier peut être téléchargé en PDF/DOCX via le bouton ci-dessous.

Format exportable :

- PDF (formatage professionnel)
- DOCX (éditable Word)
- MD (version brute)