Flanks lambdaloopers

Le guide de l'IA générative

# Concevoir l'IA générative pour les données patrimoniales

Comment transformer des portefeuilles complexes et fragmentés en une expérience conversationnelle sûre, traçable et réellement exploitable utile. Les choix techniques et stratégiques qui la sous-tendent.

Ce guide synthétise les préconisations techniques et stratégiques partagées par Héctor Borobia, Head of AI chez Flanks, dans la série d'entretiens **IA lo hicieron** de LambdaLoopers.

# 01 Du reporting manuel au dialogue direct avec la donnée financière

Flanks a bâti l'**infrastructure de données patrimoniales** qui alimente une nouvelle génération d'IA. Sa plateforme modulaire automatise l'agrégation et l'enrichissement de portefeuilles complexes et fragmentés pour les transformer en une **couche de données unique et de haute qualité**, une véritable vue à 360° sur chaque classe d'actifs. La mission : *démocratiser la gestion de patrimoine* en supprimant la friction manuelle, pour que family offices, banques privées, gérants d'actifs et fintechs se concentrent sur un conseil plus rapide, plus pertinent et plus personnalisé.

### Avant

- Consultations manuelles des portefeuilles
- Exports Excel et rapports ponctuels
- Dépendance à l'analyse technique manuelle
- Forte friction pour croiser portefeuilles, devises et indices de référence

### Après

- Questions en langage naturel sur les portefeuilles
- Analyse de l'exposition aux devises en temps réel
- Comparaison automatique avec des indices comme le S&P 500
- Préparation de rapports en quasi temps réel

**À retenir** L'**IA générative supprime la friction** autour de la donnée que Flanks fournissait déjà : une donnée financière de haute qualité.

# 02 Une IA générative productive, ce n'est pas que de l'IA : c'est du produit, de la donnée, du logiciel et du métier

Le plus difficile n'a jamais été d'appeler un modèle via une API. C'était de déterminer de **quelles données le conseiller a réellement besoin**, où se situe le risque, et comment lui rendre une réponse digne de confiance. Cinq éléments doivent avancer d'un seul mouvement.

### Cadrage fonctionnel

Définir précisément les questions auxquelles le système doit répondre, et celles qu'il doit écarter.

### Conception de la donnée

Savoir quelles données existent, où elles se trouvent et de quelle qualité avant de modéliser.

### Ingénierie logicielle

Intégrer API, outils, traçabilité et systèmes existants de façon robuste.

### Expertise financière

Interpréter portefeuilles, indices de référence, risque et reporting avec discernement.

### UX conversationnelle

Gérer l'ambiguïté et les clarifications, et instaurer la confiance dans la réponse.



Décision clé

Traiter l'IA générative comme un **projet produit et données**, pas comme une simple expérimentation d'IA.

# 03 Le modèle compte, mais moins que le système de données qui l'entoure

Choisir un modèle est la décision la plus facile. L'avantage durable réside dans ce que l'on construit autour : le contexte fourni, les outils connectés, et la finesse avec laquelle on gère l'ambiguïté.

### Choix du modèle

Commencer par un grand modèle pour isoler les variables. Ne pas choisir sur des benchmarks génériques : mesurer qualité, latence, coût et bon usage des outils, et employer différents modèles selon les tâches.

### Ingénierie du contexte

Le prompting décide **quel contexte entre, dans quel ordre**, et quels outils sont connectés. Plus de contexte n'est pas toujours mieux : cela ajoute du bruit, du coût et des erreurs.

### Frameworks

LangGraph, LlamaIndex, CrewAI et les SDK d'agents prennent en charge le code répétitif et les graphes d'état. Ce qu'ils ne feront *pas*, c'est concevoir votre architecture produit à votre place.

### UX conversationnelle

Face à une question ambiguë, le système doit demander, pas deviner. Si un titre est coté sur plusieurs places, la bonne réaction est : *« De quel marché parlez-vous ? »*

**Décision clé**

Ne pas courir après *« le meilleur modèle »*. Choisir le **bon modèle pour la bonne tâche**, au sein d'une architecture mesurable.

# 04 Les outils sont la frontière entre une IA probabiliste et un métier déterministe

Un outil de niveau production doit être conçu pour que le modèle **sache qu'il existe, sache quand y recourir**, reçoive des arguments propres et puisse se rétablir proprement en cas d'échec.

### Conception sémantique

Des entrées faciles à interpréter, des sorties structurées, des descriptions de fonctions claires, et des permissions définies par utilisateur et par domaine de données.

### Erreurs fonctionnelles

Pas de données pour une année ? L'outil ne renvoie pas « *erreur* », il répond « *Aucune valeur pour cette période ; essayez une autre plage* », pour que le modèle puisse simplement réessayer.

### Architecture déterministe

La logique critique réside dans des **outils contrôlés**. Le LLM interprète et rédige : les calculs, les permissions et les données réelles proviennent de systèmes déterministes.

### Préfiltrage et validation

Des validations programmatiques en amont du modèle. Un préfiltrage des champs pour éviter de transmettre du bruit inutile. Une latence mesurée par outil.

**À retenir** — Un outil mal conçu fait paraître tout l'agent maladroit. Un **outil bien conçu élimine discrètement l'essentiel du risque du système.**

# 05 Un protocole unique pour relier n'importe quel modèle à vos données et à vos outils

**Un bon MCP doit fonctionner avec plusieurs modèles. S'il ne marche qu'avec le plus puissant, c'est qu'il est probablement mal spécifié.**

Traiter le MCP comme un produit d'infrastructure

### + Ce qu'apporte le MCP

- Connexion standard entre modèles, outils et données
- Des capacités financières réutilisables, exposées
- Les clients connectent leurs propres agents aux données Flanks
- Séparation de l'interface finale et de la couche de données connectable
- Un nouveau modèle économique possible : le *« moteur de données »*

### Contrôles indispensables

- OAuth + jetons d'accès + identifiants client
- Listes d'autorisation et audit des appels d'outils
- Limites d'usage et observabilité de la latence
- Plus de MCP = plus de contexte, plus de coût, plus de bruit

**Décision clé** — Traiter le MCP comme un **produit d'infrastructure**, pas comme un simple connecteur technique.

# 06 La mémoire, ce n'est pas stocker davantage, c'est retrouver le bon contexte

**01 · Capter**

### Mémoire de travail

La conversation en cours, le contexte du fil de session en train de se dérouler.

**02 · Distiller**

### Résumé à la clôture

Un LLM capte les points clés et décide quels candidats méritent d'être conservés.

**03 · Stocker**

### Mémoire à long terme

Jalons et habitudes du conseiller : indices de référence, formats favoris, profil de risque.

**04 · Récupérer**

### Récupération contextuelle

Un outil réinjecte le contexte utile précisément au moment où l'utilisateur en a besoin.

*« L'IA apprend à mesure qu'on l'utilise »* est une idée séduisante, mais trompeuse. Un modèle figé n'apprend pas de lui-même : l'expérience s'améliore parce que le système **récupère et injecte le bon contexte** au bon moment, sans le moindre réentraînement.

**Décision clé**

Privilégier le **contexte comme mémoire** avant d'envisager un réentraînement, sauf si le ROI le justifie clairement.

# 07 Les agents IA ne sont pas l'unique solution : parfois, la meilleure IA est un simple workflow

L'orchestration commence par une question sans détour : *la tâche est-elle déterministe ?* Si les entrées et les étapes ne changent jamais, vous n'avez certainement pas besoin d'un système multi-agents.

**Workflow**

### Séquence figée

Entrées et étapes prévisibles. Aucune décision en temps réel requise.

**Agent unique**

### Décision en temps réel

Peu de spécialisation. ReAct ou des schémas simples suffisent généralement.

**Multi-agents**

### Sous-tâches spécialisées

Uniquement pour des sous-tâches parallélisables et agrégeables, avec une vraie spécialisation. Plus d'agents = plus de surface d'erreur.

### Conception modulaire

Flanks s'appuie sur des graphes acycliques et des composants découplés, ajoutant outils ou capacités sans redessiner tout le système.

### Le ML classique comme allié

Tout ne doit pas être résolu par des LLM. Des classifieurs de type BERT peuvent être préférables pour des tâches répétitives, rapides et déterministes.

**Décision clé** Laisser la **tâche** trancher entre workflow, agent unique ou multi-agents, pas la tendance du moment.

# 08 Une vraie mise en production, c'est : traçabilité, évaluations, garde-fous et coûts maîtrisés

L'AgentOps, c'est ce qui transforme une belle démo en un système réellement exploitable. L'observabilité a sa place dès le POC, **et non ajoutée à la fin.**



### Traçabilité

Instrumenter les entrée, appels d'outils, sortie, tokens, latence par étape, erreurs, relances, boucles et délais dépassés.

### Évaluations et golden set

Créer de 15 à 50 questions avec leurs réponses attendues dès le départ. Utiliser un LLM-juge, avec un modèle différent, pour évaluer à grande échelle.

### Garde-fous

En entrée : bloquer les questions hors domaine. En sortie : bloquer les réponses à risque. Sur le comportement : limiter les boucles et les actions non souhaitées.

### Feedback et FinOps

Le retour des utilisateurs alimente l'amélioration des prompts et du golden set, il n'entraîne pas le modèle. Mesurer tokens, appels et coût par client.

**À retenir** — Sans traçabilité, **pas de débuggage.** Sans évaluations, **pas d'amélioration.** Sans garde-fous, **pas de contrôle opérationnel.**

# Sept décisions pour porter l'IA générative en production dans la wealthtech

La leçon essentielle de Flanks : livrer de l'IA générative, ce n'est pas brancher un LLM. C'est **concevoir le système qui l'entoure.**

**01** **Valider d'abord le problème**

Partir d'une vraie douleur : réduire la friction d'accès à des données patrimoniales complexes.

**02** **Commencer petit pour pouvoir mesurer**

Un modèle, un outil, peu de MCP, un cas cadré. Si tout échoue d'un coup, impossible de savoir ce qui a cassé.

**03** **Séparer génération et logique**

Le LLM interprète et rédige. Les outils calculent, valident, interrogent et appliquent les permissions.

**04** **Concevoir outils et MCP comme un produit**

Exposer des API ne suffit pas. Les concevoir pour que les modèles les comprennent sans compromettre la sécurité.

**05** **Faire du contexte un avantage concurrentiel**

Le modèle se banalise. Le facteur différenciant, ce sont les données, le contexte, la mémoire et les permissions.

**06** **Ne pas sur-concevoir avec des agents**

Si un workflow suffit, utilisez un workflow. Le multi-agents seulement en cas de vraie spécialisation.

**07** **Instrumenter dès le POC**

Traçage, golden set, évaluations, feedback, garde-fous et coûts doivent exister avant de passer à l'échelle.

**Le modèle n'est pas le rempart. Le rempart, ce sont les données, les outils, l'orchestration, et la capacité à faire tourner l'IA générative sous contrôle.**

Flanks

# La gestion de patrimoine, en toute simplicité

Parlons-en !

Flanks.io