AI Project

Voici une **liste détaillée des technologies à utiliser pour le déploiement d'une IA puissante en localhost**, et une **explication du rôle de chaque composant** :

---

1. Système d'exploitation et drivers

a. Ubuntu 22.04 LTS

- **Rôle** : OS optimisé pour les environnements IA, stable, supporté par les libs NVIDIA, PyTorch, etc.

b. NVIDIA Drivers + CUDA + cuDNN

- **Rôle** : Interface entre les logiciels d'IA et les GPUs. CUDA permet les calculs massifs en parallèle.

---

2. Environnement IA local

a. Docker

- **Rôle** : Conteneuriser et isoler les environnements d'IA pour faciliter le déploiement multi-modèles et reproductible.

b. Python 3.10+ avec virtualenv / conda

- **Rôle** : Langage principal pour la majorité des frameworks IA (transformers, LLM, NLP, etc.)

---

3. Frameworks IA

a. PyTorch

- **Rôle** : Entraînement et inférence des modèles LLM sur GPU.

b. Transformers (Hugging Face)

- **Rôle** : Chargement, manipulation et exécution des modèles comme LLaMA, Mistral, Falcon, etc.

c. bitsandbytes / QLoRA

- **Rôle** : Accélération + quantification des modèles pour tourner sur moins de VRAM (4-bit, 8-bit)

d. Flash Attention / xFormers

- **Rôle** : Optimisation mémoire et vitesse pour modèles >7B

---

4. Modèles et formats

a. GGUF / GGML (llama.cpp)

- **Rôle** : Exécuter des LLM quantifiés (4-bit) sur CPU/GPU sans dépendance lourde

b. Mistral 7B / LLaMA 2 / Zephyr / Nous Hermes / Mixtral

- **Rôle** : LLM open-source spécialisés selon les cas d’usage

---

5. Serveur d’inférence local

a. Ollama

- **Rôle** : Lancement ultra simple de LLM locaux (mistral, llama2, codellama...) via une interface API REST/CLI

b. text-generation-webui

- **Rôle** : UI web complète pour lancer, tester, et paramétrer plusieurs modèles LLM

---

6. Routage et orchestration IA

a. LangChain / LlamaIndex

- **Rôle** : Construction de chaînes de raisonnement complexes, intégration avec des bases de données, fichiers, API

b. FastAPI

- **Rôle** : Exposer ton modèle LLM comme une API REST locale sécurisée

---

7. Base de connaissance (RAG)

a. ChromaDB / FAISS

- **Rôle** : Indexation sémantique de documents pour le Retrieval-Augmented Generation (RAG)

b. PDF, TXT, DOCX loader (via LangChain)

- **Rôle** : Extraction de connaissances métier à injecter dans le LLM

---

8. UI/UX local

a. Gradio / Streamlit

- **Rôle** : Interface utilisateur simple pour tester ton agent conversationnel, générateur de texte ou assistant personnel

---

9. Outils avancés pour usage local

a. LM Studio

- **Rôle** : Front-end pour interagir facilement avec les LLM (chat, import, test) sans CLI

b. KoboldAI / SillyTavern (RP & narration)

- **Rôle** : Cas d’usage IA narration, gaming, écriture créative

---

10. Sécurité et supervision

a. Prometheus + Grafana (optionnel)

- **Rôle** : Monitoring de la charge GPU, mémoire, performance des modèles

---

Exemple de workflow local :

```
Lancement modèle via Ollama (Mistral 7B quantifié)
→ Exposition via FastAPI
→ Accès via interface Gradio
→ Ajout de RAG via ChromaDB + LangChain
→ Surveillance GPU via Grafana
```

Avec cette stack complète et une machine adaptée, tu peux faire tourner une IA **puissante en local, sans connexion Internet**, avec des fonctionnalités proches de ChatGPT, **sous certaines conditions** :

Ce que tu pourras faire en local :

1. **Faire du chat avec un LLM** (type Mistral 7B, LLaMA 2, etc.) sans cloud

2. **Traiter des documents (PDF, Word, etc.)** et poser des questions dessus (via RAG)
3. **Lancer des agents IA autonomes**, assistants ou bots (via LangChain ou AutoGPT local)
4. **Coder, corriger ou compléter du texte** (CodeLLaMA, Deepseek, etc.)
5. **Créer des interfaces web personnalisées** avec Gradio ou Streamlit

---

Limites par rapport à ChatGPT :

Fonctionnalité	Local possible ?	Remarques
Chat conversationnel	**Oui**	Avec modèle 7B ou 13B max (selon ta VRAM)
Connaissances à jour	**Non**	Tu dois charger toi-même tes fichiers ou scraper
Compréhension d'image	**Partiel**	Possible avec `llava`, mais limité à certains modèles
Synthèse vocale	**Oui**	Avec `Coqui TTS` ou `Silero` en local
Génération audio ou vidéo	**Non ou expérimental**	Demande d'énormes ressources (diffusion, etc.)
Performance globale	**Bonne à très bonne**	Si 48 à 128 GB RAM et 1+ GPU 24GB VRAM

Conditions nécessaires pour un résultat optimal :

- **RAM ≥ 64 GB**
- **SSD rapide (NVMe)** pour les modèles quantifiés (gguf/ggml)
- **GPU ≥ 24 GB VRAM** (RTX 3090/4090 ou A100 pour >13B modèles)
- **Modèles quantifiés (4-bit)** pour économiser VRAM
- **Pas de dépendance Internet** une fois tout installé

Exemple de stack hors-ligne complète :

- **OS** : Ubuntu 22.04 LTS
- **Serveur LLM** : `Ollama` ou `text-generation-webui`
- **Modèle** : `mistral-7b-instruct.Q4_K_M.gguf` (quantifié 4-bit)
- **UI** : `Gradio` en local
- **RAG** : `LangChain` + `ChromaDB` + tes fichiers
- **Outils dev** : Python, CUDA, Docker
- **TTS** (facultatif) : `Silero` pour voix IA hors-ligne

Voici une **configuration complète et optimisée pour faire tourner une IA puissante en local (localhost)** avec jusqu'à **7 GPU**, adaptée à ton projet :

🧠 Objectif :

Faire tourner localement des modèles LLM 7B à 70B (chat, agents, RAG, etc.) en multi-GPU, de façon stable et extensible.

🧱 Liste des composants à acheter (2025 – meilleur rapport qualité/performance)

🖥️ Carte mère (Motherboard)

- **Modèle** : *ASRock WRX80 Creator* ou *Supermicro H12DSi*
- **Chipset** : Compatible AMD Threadripper Pro ou EPYC
- **Avantage** :
    - Jusqu'à 512 Go de RAM DDR4 ECC
    - 7+ slots PCIe 4.0 x16 pour 7 GPU
    - Support NVMe 4x

🧮 Processeur (CPU)

- **AMD Threadripper PRO 3955WX** *(16C/32T)*
- **Avantage** : Excellent support pour multitâche et bande passante PCIe élevée
- (Alternative : EPYC 7302P si tu choisis une carte mère serveur)

🧠 Mémoire RAM

- **128 Go DDR4 ECC Registered** (4×32 Go, extensible à 256 ou 512 Go)
- **Marque conseillée** : Crucial / Corsair / Kingston ECC
- **Pourquoi** :
    - Large capacité nécessaire pour les modèles >13B et l'usage de LangChain + vectordb

---

🎮 Cartes graphiques (GPU)

Objectif : Min 24 Go VRAM / carte, multi-GPU

- **7× NVIDIA RTX 3090 / RTX 4090**
  (ou **mix RTX 3090 et RTX A5000** pour réduire le coût)

- **Pourquoi NVIDIA** :
    - Support CUDA, cuBLAS, multi-GPU training
    - Très bon support pour quantized models (gguf, GPTQ, etc.)

❄️ Refroidissement (Cooling)

- **Watercooling CPU** : Corsair iCUE H150i / Arctic Liquid Freezer II
- **Ventilation rig** : Be Quiet! Silent Wings 4 / Noctua 140mm
- **Alimentation dédiée GPU** : Ventilation active sur GPU (blower style conseillé)

⚡ Alimentation (PSU)

- **2× PSU 1600W Platinum** (Corsair / EVGA)
- **Pourquoi** :
    - Chaque RTX 3090 tire jusqu'à 350W
    - Séparation CPU/GPU sur deux circuits

🧊 Boîtier (Châssis)

- **Rig open-air style mining** ou **boîtier custom rack 4U**
- **Exemples** : Veddha V3C, Hydra Rack Mining Frame
- **Pourquoi** : ventilation maximale et accessibilité aux slots GPU

💾 Stockage (Disques SSD)

- **1× SSD NVMe Samsung 990 Pro – 2To** *(Système + modèles)*
- **1× SSD NVMe Samsung 980 Pro – 2To** *(Données, bases vectorielles, cache)*
- **Option RAID-0** : pour performance supplémentaire (si tu fais de l'inférence lourde ou fine-tuning)

🧰 Divers

- Riser PCIe (x16 powered) pour montage des 7 GPU
- Ventilateurs additionnels pour frame open-air
- Multiprise avec protection surtension + monitoring wattmètre

---

📦 Budget approximatif (avril 2025)

Composant	Prix estimé (USD/€)
CPU Threadripper	~900 €
Carte mère WRX80	~600 €
RAM 128 Go ECC	~300 €
7× RTX 3090	~7 000 €
PSU 2×1600W	~500 €
SSD (2×2To)	~300 €
Watercooling	~150 €
Châssis open-air	~150 €
Divers (riser, câbles)	~100 €
Total	**~10 000–11 000 €**

Souhaites-tu que je t’envoie un **PDF complet avec les liens d’achat, alternatives low-cost, et conseils d’assemblage avancé** ?