IDRIS

Assistants de Développement

Déploiement

- Antoine REGNIER

→ 01/04/2025

Sommaire

Objectifs

Dans cette présentation nous verrons quels sont les critères pertinents à considérer lors du déploiement d'un d'assistant de développement.

Deux cas d'usages différents:

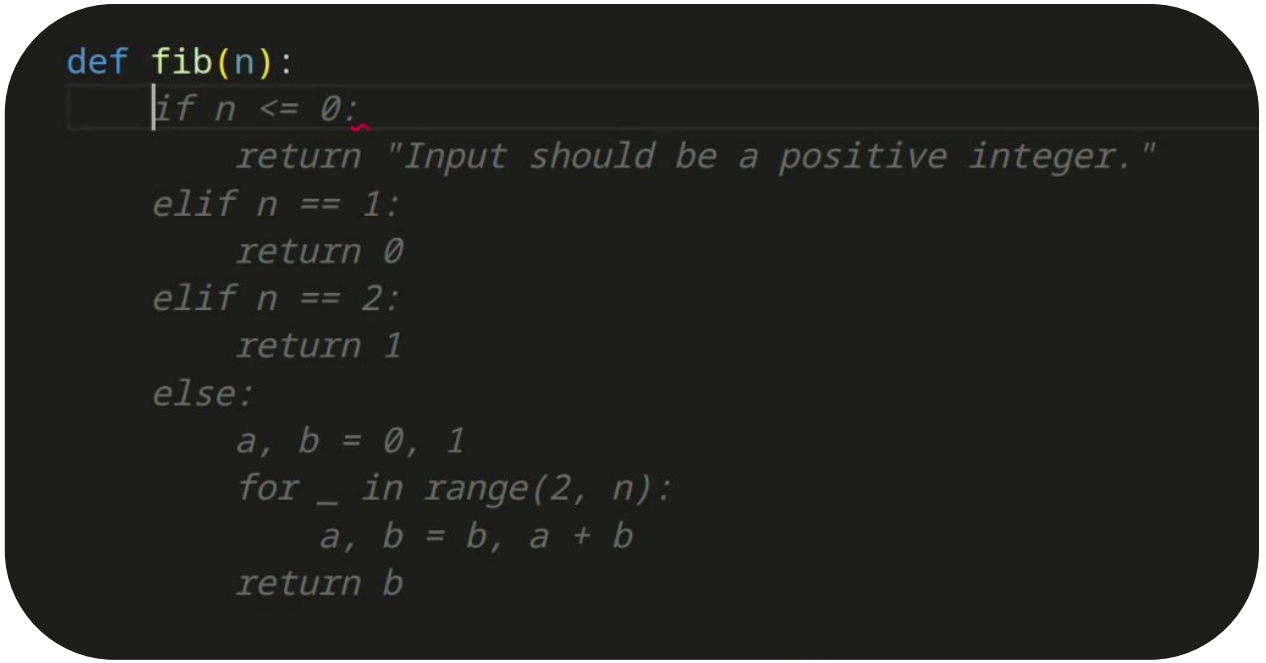
- en complétion
- chatbot/génération

Complétion

Lors de l'édition de code l'assistant propose une suite logique au code que l'utilisateur est en train de taper

Dans ce type d'utilisation on privilégie :
la latence de la suggestion
la cohérence avec l'existant

Ce type d'utilisation demande généralement de petit contexte (<1000 tokens).
On va généralement y privilégier des petits modèles (3B-32B de paramètres)



Génération / Chat

Dans le mode génération on va donner des ressources de travail ainsi que des instructions à l'assistant.

Ce type de tâche demande généralement un plus grand contexte de travail afin de pouvoir traiter des fichiers tel que des bibliothèques logicielle dans leurs intégralité.
Comme la tâche n'est pas dans l'interaction une plus grande latence est tolérable

02 Auto Hébergé vs SaaS

Déploiement d’assistant de code : auto-hebergé vs Saas

Auto Hébergé

Pour
Contrôle sur le matériel.
Contrôle sur la localité des données.
Prévisibilité des coûts d’hébergement
Sécurité par l’isolation réseau

Contre
Coût RH (MCO, MCS)
Investissement initial
Performance dans le temps

Hébergé chez fournisseur

Pour
Contrôle sur la localité des données (si hébergeur français).

Contre
Coût RH (MCO, MCS)
Imprévisibilité des coûts à moyen terme (bande passante)
Expertise nécessaire pour la sécurisation

Software as a service

Pour
Facilité d’utilisation.
Performance.
Maintien en condition opérationnelle

Contre
Imprévisibilité des coûts à moyen terme si facturé aux token

02.A SaaS

Déploiement d'assistant de code : Saas

Liste des principaux acteurs :

Nom		Entreprise	Nationalité	Traitement en EU
Saleforces Enstein AI		Saleforces	USA	???
Claude		Anthropic	USA	Non
Cursor		Anysphere Inc	USA	Non
Gemini 2.0		Google	USA	Non
Copilot		Microsoft	USA	??
ChatGPT		OpenAI	USA	??
DeepSeek-R1		Deepseek	Chine	Non
Mistral-large		Mistral	France	Oui

02.B

Auto Hébergé & Hébergé (IaaS)

Déploiement d’assistant de code

Individuel

Mutualisé

Pour
Permet le travail Hors ligne.
Latence.

Contre
Qualité de génération moindre.
Coûts et consommation de l’équipement personnel.

Pour
Meilleurs modèles disponibles.
Mutualisation des coûts.

Contre
Accès internet requis.
Latence.

03

Les Briques Logicielles et Matérielles

03 Les Briques Logicielles et Matérielles

A **Matériels**

B Modèles

C Moteurs d’inférence

D Les routeurs de requêtes

E Les interfaces d’utilisation

Matériels : Les critères principaux

Dans l'achat de matériel les critères principaux sont les suivants :

Taille et vitesse de la VRAM

- Il y a de grosse pénalité à devoir décharger/recharger le modèle lors de l'inférence

Puissance de calcul

- La puissance de calcul permet à plus d'utilisateurs d'être servis simultanément

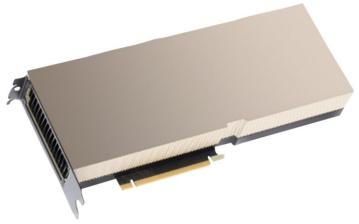
Présence d'unité de calcul spécifique

Consommation

Prix

Support

Matériels : Les GPUs et NPUs



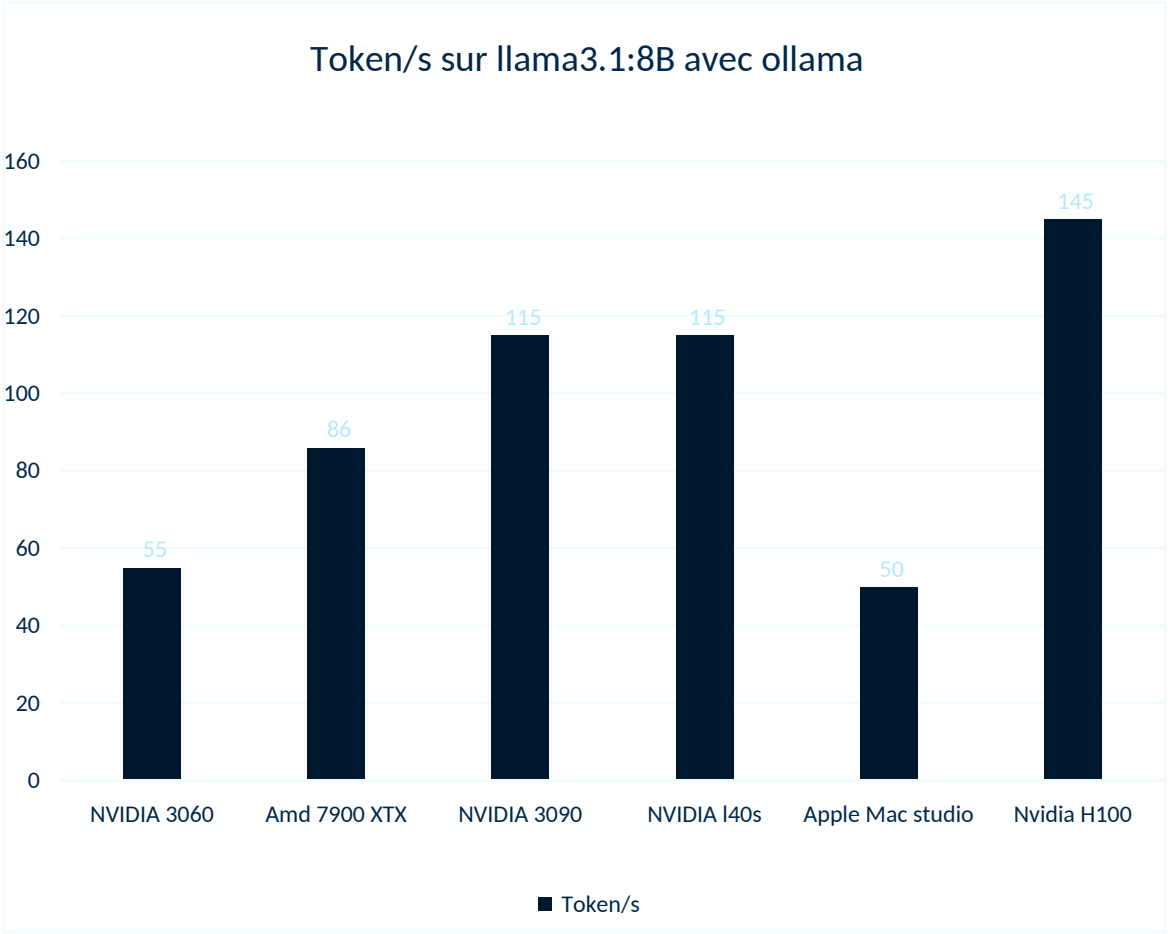
330 €	1 100 €	2 000 €	6 000 €	6 000 €	30 000 €
NVIDIA 3060	AMD 7900 XTX	NVIDIA 3090	NVIDIA l40s	APPLE Mac studio	NVIDIA H100
12 Go	24 Go	24 Go	48 Go	256 Go	80 Go
170W	370W	420W	350W	77W	700W

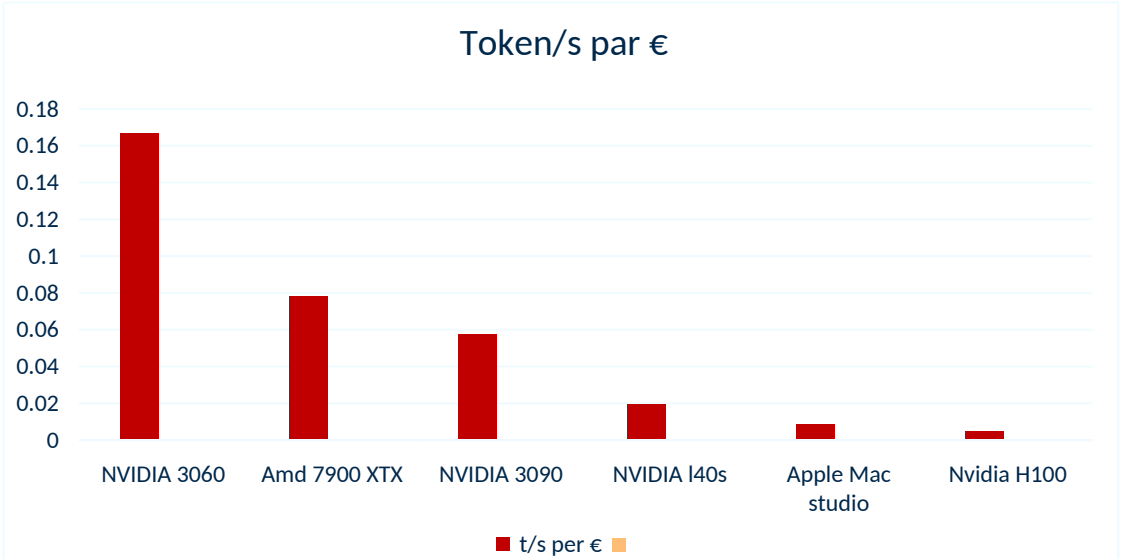
Pour éviter une grosse perte en performance il est important d'avoir suffisamment de VRAM pour faire entrer le modèle

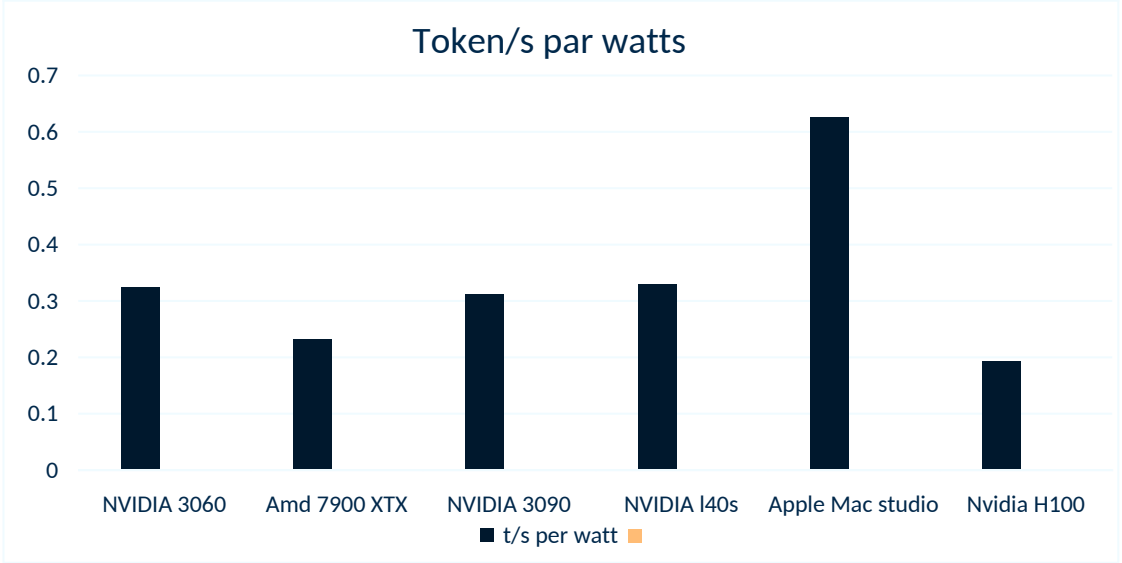
N.B : Sur les appareils présenté, l'Apple Mac Studio a une unité de calcul dédiée à L'IA intégré (NPU)

Matériels : Les résultats pour un utilisateur

Token/s sur llama3.1:8B avec ollama







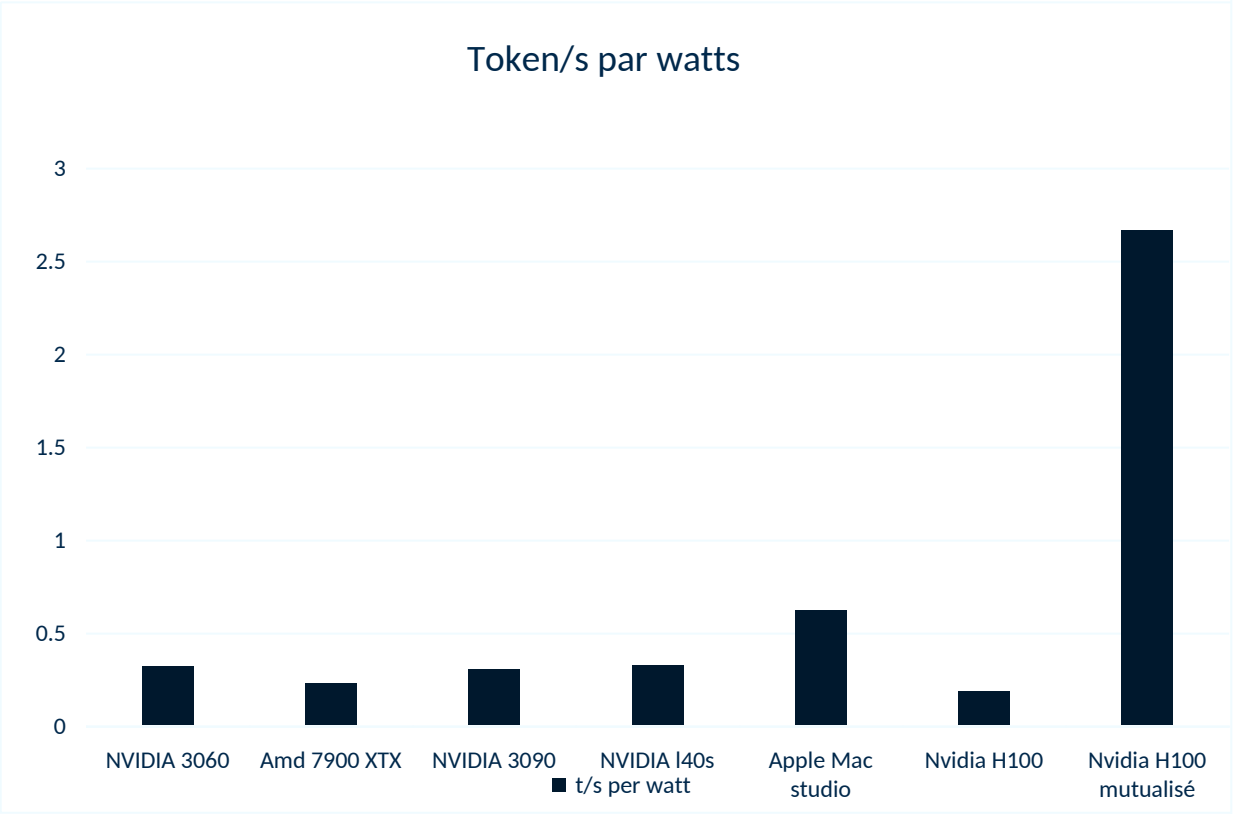
Matériels : Les effets d’échelles de la mutualisation

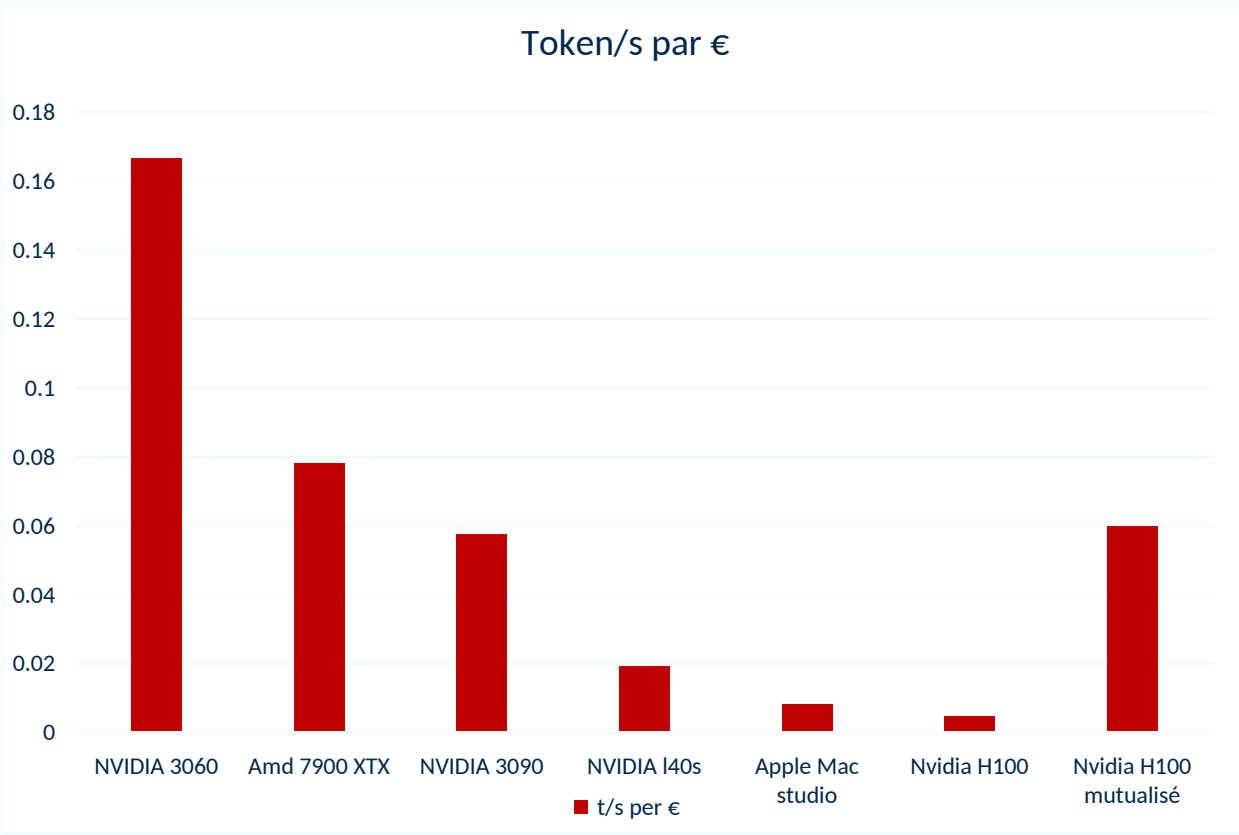
Vitesse de lecture Humaine ~ 10 token/s

	Price	Conso(W)	Token per second	T/s per Watt	T/s per euros
Nvidia H100 70 utilisateurs	30000,00	700,00	1866,30	2,67	0,06
Nvidia H100 Par User	428,57	10,00	26,66	2,67	0,06

Matériels : Les effets d'échelles

Token/s sur llama3.1:8B mix Ollama pour les mesures 1 utilisateurs et vLLM pour le mutualisé





x
.

Matériel : quelques fournisseurs de services français

Fournisseur	Matériel	VRAM	Prix € ht/mois
OVH	Nvidia A10	24	554,80
OVH	L40S	48	1008
OVH	2* L40S	96	2016
OVH	2* H100	160	3880
Scaleway	2* L40S	96	2044
Scaleway	2* H100	160	3985.8

Matériel :

Il est important de choisir son matériel en fonction du nombre d’utilisateurs cible de l’assistant de code.

Le matériel n’est pas cher par rapport au coût de l’hébergement.

03 Les Briques Logicielles et Matérielles

A **Matériels**

B **Modèles**

C **Moteurs d’inférence**

D **Les routeurs de requêtes**

E **Les interfaces d’utilisation**

Modèle : Les critères principaux

Le choix du modèle se repose sur plusieurs critères :

Le dataset d'entraînement :

Est ce que le modèle à été entraîné sur votre langage de programmation ?

La taille du modèle :

Est ce qu'elle est proportionnée au matériel ?

Le modèle est-il compressible ?*

La licence :

Le modèle est-il juridiquement compatible avec votre cadre d'usage ?

La performance :

Le modèles est-il performant à suivre les instructions demandées ?

Cas d'usage :

Génération ou complétion.

Modèle : les LeaderBoards de performances

Plusieurs leaderboards sont disponibles sur huggingface permettant une comparaisons des modèles :

https://huggingface.co/spaces/bigcode/bigcode-models-leaderboard

https://aider.chat/docs/leaderboards/

aider

Modèle : Adéquation au matériel et compression

Estimation de la quantité de VRAM nécessaire à l'exécution du modèle (hors caches)

Taille du modèle	FP32	FP16	FP8	INT4
3B	11,7 Go	5,8 Go	2,9 Go	1,5 Go
7B	26,9 Go	13,5 Go	6,7 Go	3,4 Go
14B	53,8 Go	26,9 Go	13,5 Go	6,7 Go
32B	123,5 Go	61,9 Go	30,9 Go	15,4 Go

Attention : la quantité de calcul par propagation ne changent selon la compression.
Cependant si des unités de calculs spécialisées (tensor cores de NVIDIA) pour la précision choisie sont présentent, les calculs peuvent êtres accélérés

On constate des dégradations de suivi de consigne en cas de trop forte compression. Il faut tester et/ou trouver des comparaison par rapport au modèle de base

Modèles : Quelques noms

Famille et origine	3B	6B-8B	14B	32B	Licence
Qwen2.5-Coder (Alibaba Chine)	Oui	Oui	Oui	Oui	Apache License
Starcoder-v2 (bigCode International)	Oui	Oui	Oui	Non	Open RAIL-M v1 License
Deepseek-coder-v2 (Deepseek Chine)	Non	Non	Non	Oui	DEEPSEEK LICENSE AGREEMENT
Codellama (Meta USA)	Non	Oui	Oui	Oui	LLAMA 2 COMMUNITY LICENSE
OpenCodeInterpreter (Chine)	Oui	Oui	Oui	Oui	DEEPSEEK LICENSE AGREEMENT

Source : Huggingface

Modèles

Choisir son modèle en fonction du **matériel**, De l'**usage désiré** et de la **complexité des problèmes** auxquels l'assistant aura à répondre.

Un plus gros modèle compressé n'est pas forcement mieux qu'un plus petit modèle

03 Les Briques Logicielles et Matérielles

A **Matériels**

B **Modèles**

C **Moteurs d’inférence**

D **Les routeurs de requêtes**

E **Les interfaces d’utilisation**

Moteur d’inférence

Le choix du moteur d’inférence repose sur plusieurs critères :

Le support du matériel :

Est ce que le matériel cible est compatible avec votre moteur?

Le support du modèle :

Est ce que le type de modèle désiré est supporté?

La licence :

La licence du moteur est-elle juridiquement compatible avec votre cadre d’usage ?

La performance :

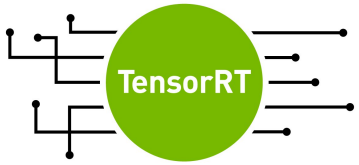
Le moteur parallélise bien les requêtes ? est-il performant ? as-t’il des fonctionnalité dont vous avez besoins ?

Cas d’usage :

Individuel ou mutualisé

Moteur d’inférence

Plusieurs moteurs d’inférences sont disponibles selon le cas d’usage que vous souhaitez

TensorRT nvidia	TGI HuggingFace	vLLM UC Berkeley	LLaMA.cpp ggml.ai
			
Closed Sources	Open Source	Open Sources	Open Sources

Moteur d'inférence : LLaMA.cpp

github : https://github.com/ggml-org/llama.cpp

	Cas d'usage	Performance	Parallélisme	Support architecture	Support quantization	Packaging
LLaMA.cpp (Ollama)	Déploiement Individuels	lent	Pipeline	CUDA, ROCm, XPU, X86, ARM,Apple Silicon,	Int4	Docker, Bare , Python, c++

LLaMA.cpp est simple à utiliser quand il est utilisé avec ollama.
Il supporte le chargement/déchargement dynamique de modèles.

Moteur d'inférence : vLLM

github : https://github.com/vllm-project/vllm

	Cas d'usage	Performance	Parallélisme	Support architecture	Support quantization	Packaging
vLLM	Déploiement Mutualisé	Rapide et hautement parallèle	Pipeline et Tensor	CUDA, ROCm, XPU, X86, ARM, Silicon, TPU, Gaudi, Neuron	AWQ, BnB, GGUF, GPTQ, Int4, Int8, FP8,	Docker, Bare , Python

Dédié à la production, Il offre moins de souplesse que llama.cpp.
De très bonnes performances

03 Les Briques Logicielles et Matérielles

A Matériels

B Modèles

C Moteurs d’inférence

D Les routeurs de requêtes

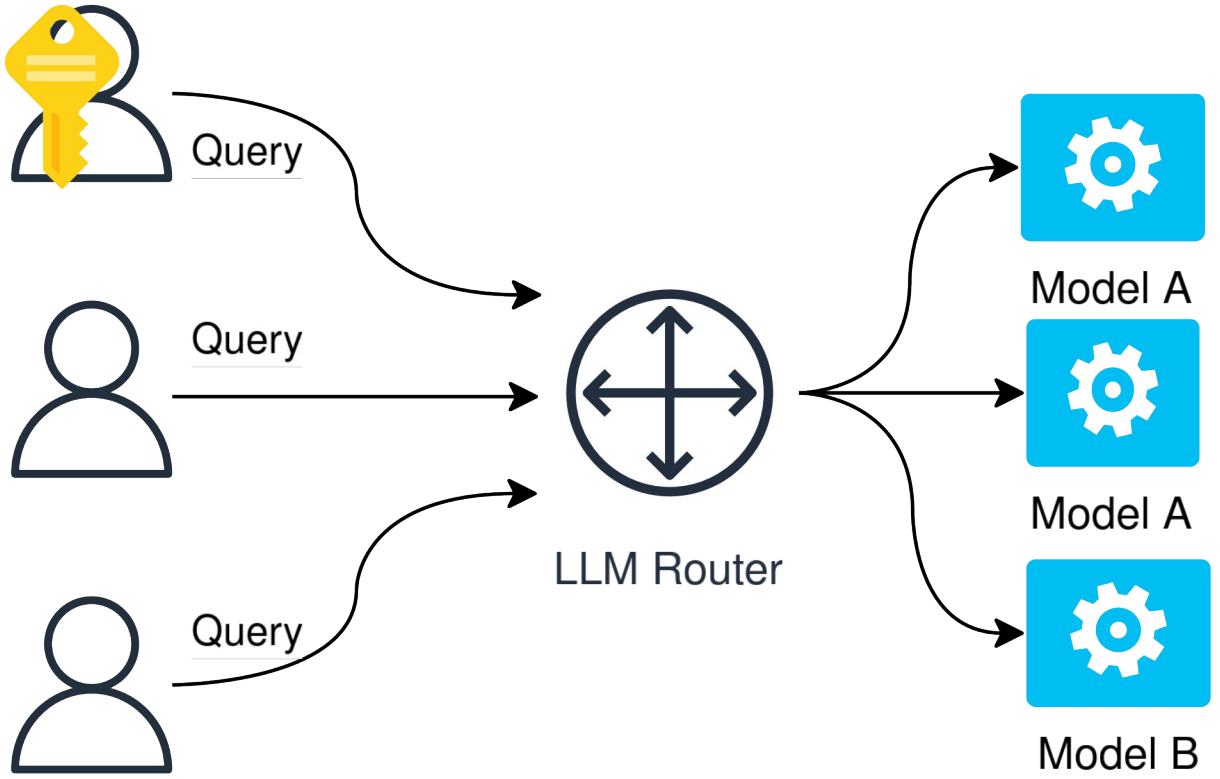
E Les interfaces d’utilisation

Routeur de requêtes

Les routeurs de requêtes sont nécessaires en cas de moteurs d'inférences multiples. Il assure :

- Répartition de charges
- L'authentification et les accès
- le routage vers les moteurs d'inférence.

Le routage peut être intelligents (router automatiquement vers le modèle approprié) ou juste suivre les indications contenues dans la requête



Routeur de requêtes

Quelques noms :

Aristote Dispatcher de centrale supéléec
github : https://github.com/CentraleSupelec/aristote-dispatcher
licence MIT

liteLLM de Berri AI
github https://github.com/BerriAI/litellm
licence MIT

03 Les Briques Logicielles et Matérielles

A Matériels

B Modèles

C Moteurs d’inférence

D Les routeurs de requêtes

E Les interfaces d’utilisation

Interfaces d’utilisation

Une fois que l’assistant est instancié il peut être joint par API : (openAI, message ou ollama).
Il existe plusieurs interfaces utilisateurs pouvant s’interfacer avec ces API

IDE : **Continue.dev** sur vsCode ou Jetbrain

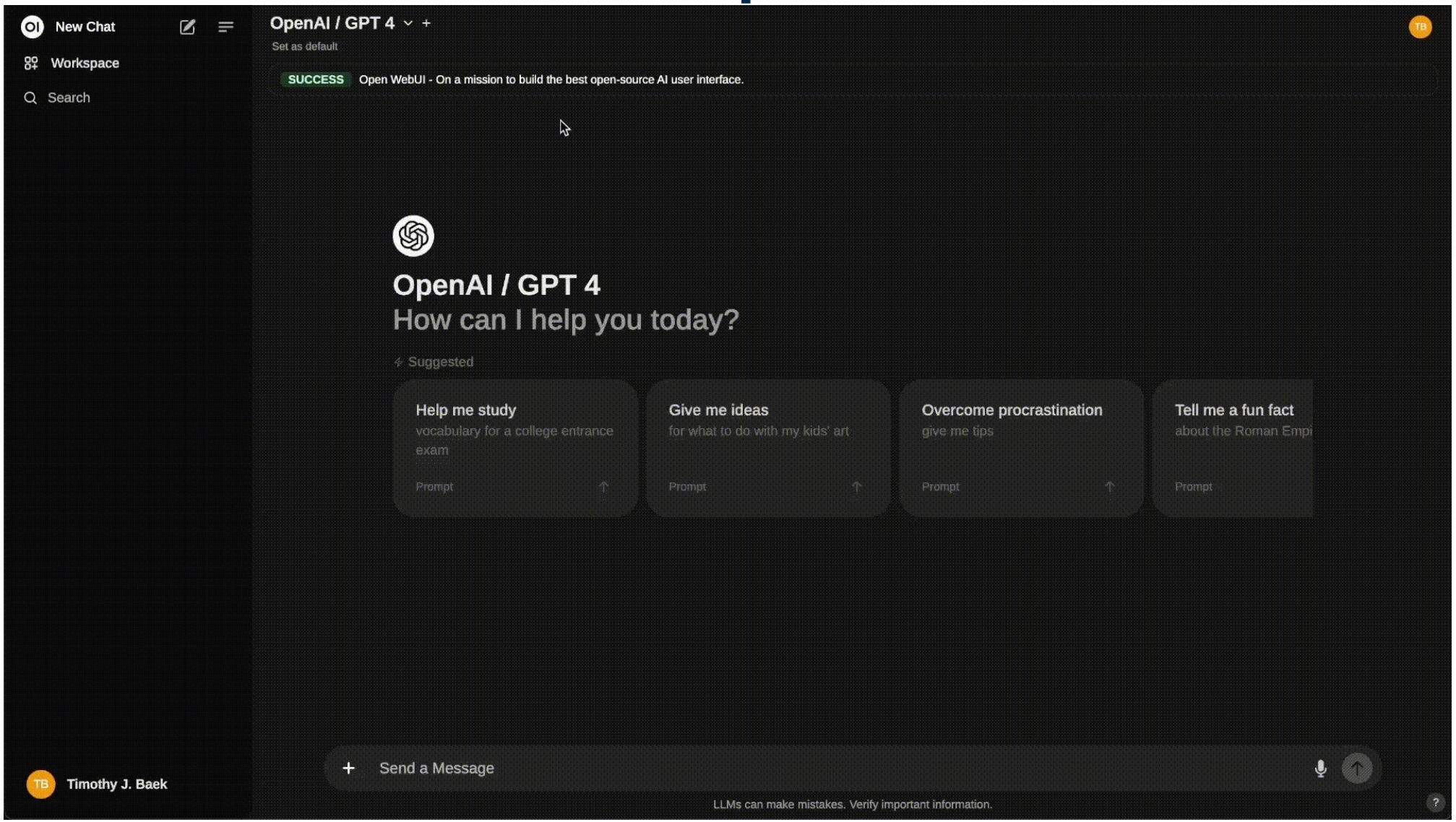
Chat interface : **Open-Webui**

Cli : **ollama run**

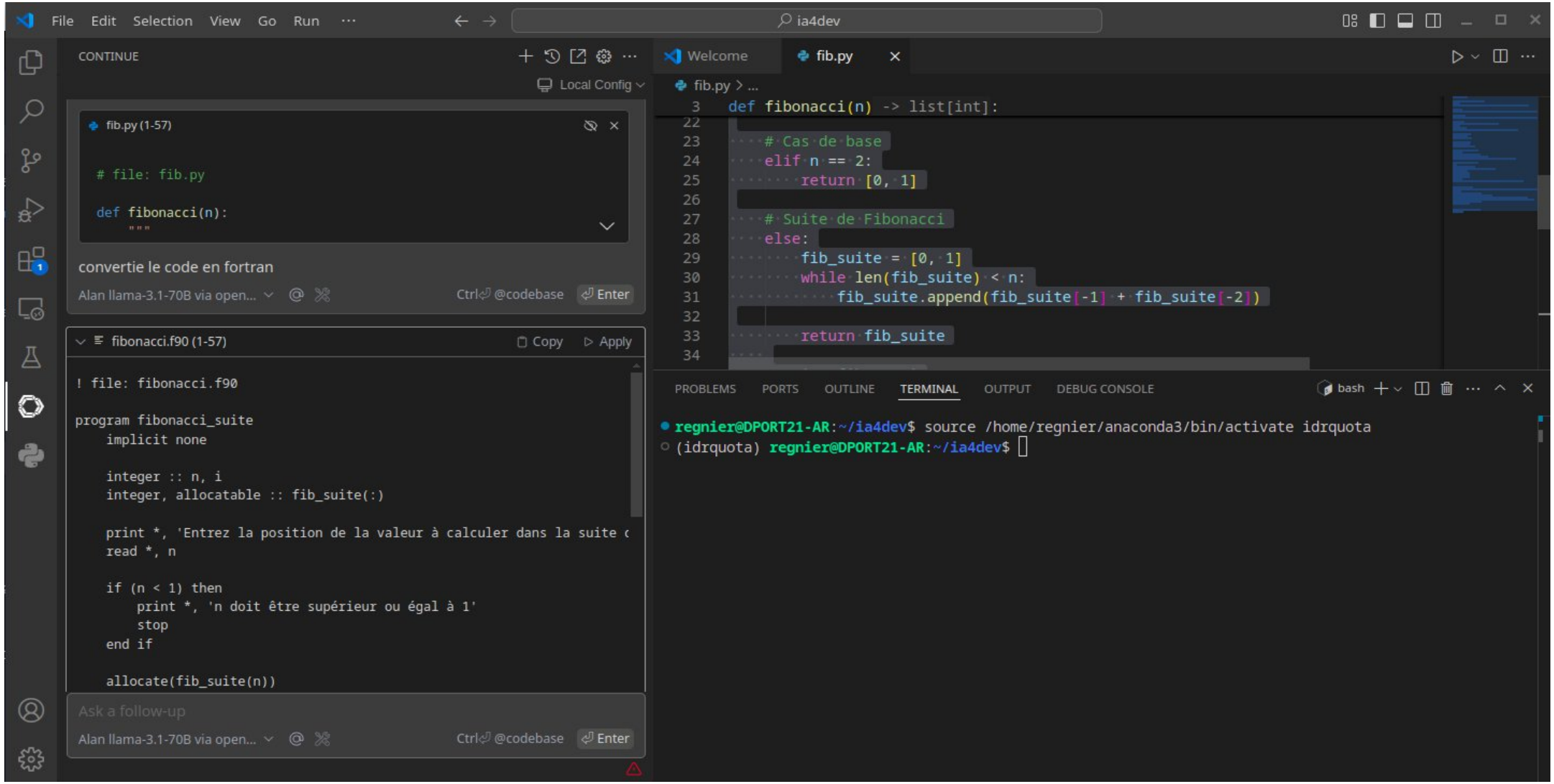
aider.chat

aider

Interfaces d’utilisation : Open Webui

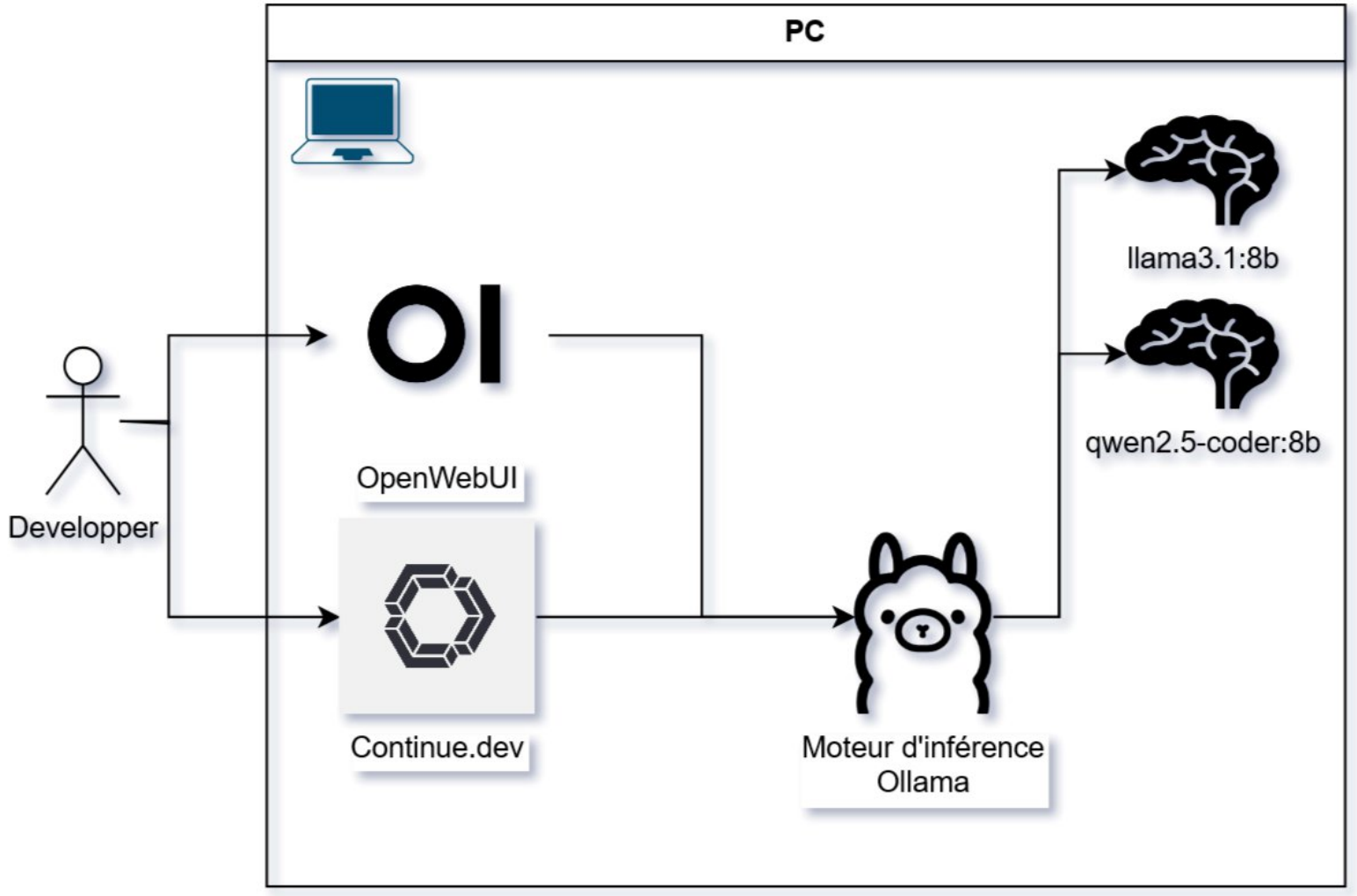


Interfaces d’utilisation : Continue.dev



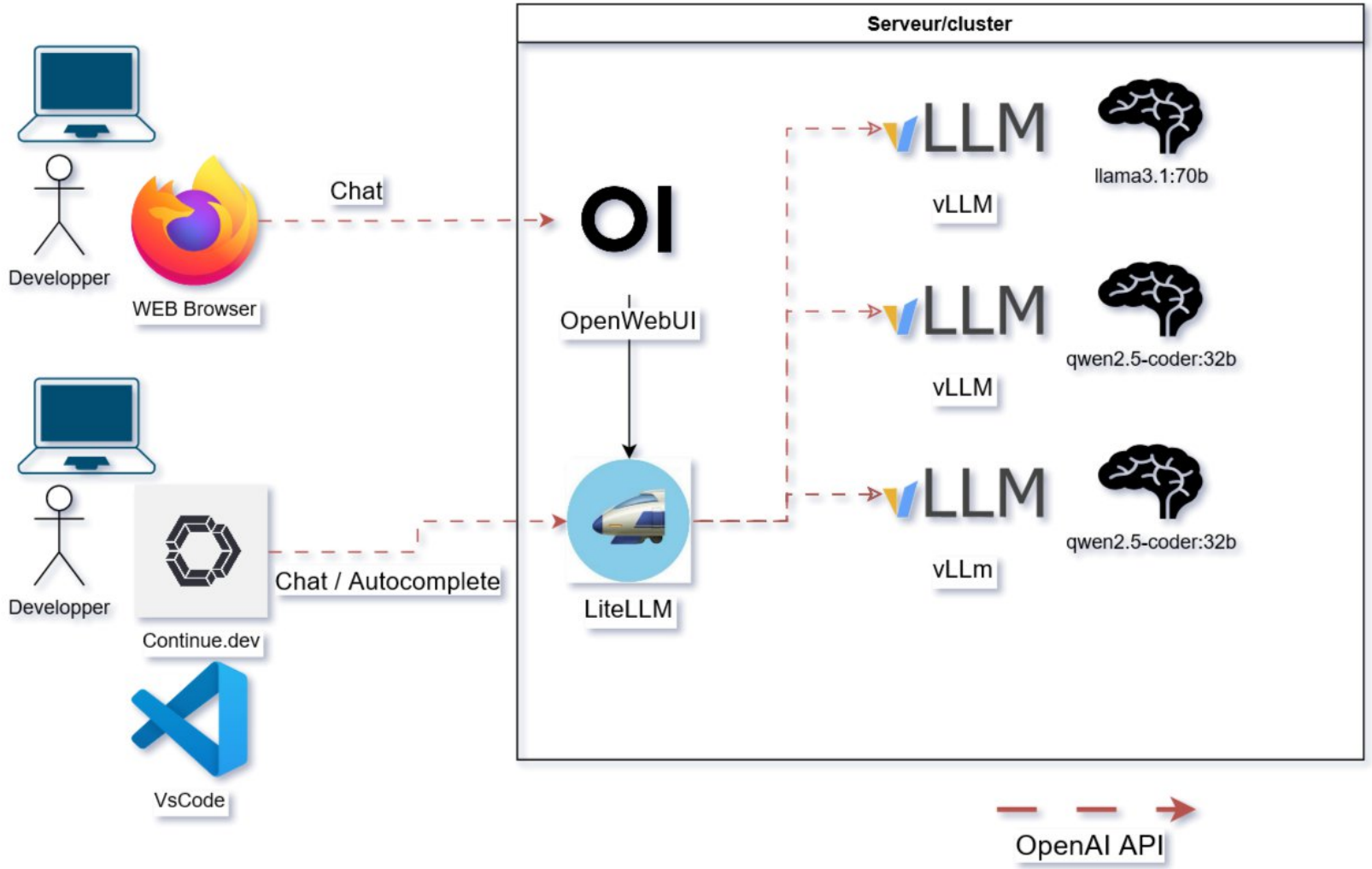
01 Déploiement individuel

Déploiement individuel



01 Déploiement mutualisé

Déploiement Mutualisé



Architectures types recommandée lo

Nombre d'utilisateurs	Matériel recommandé	Moteur d'inférence	Modèle IA	Hébergement
1-3 utilisateurs	RTX 3060, Mac Studio	LLaMA.cpp (Ollama)	7B-13B	Local
20 utilisateurs	L40S ou 2* 3090	vLLM	13B-34B	Serveur interne/cloud privé
100 utilisateurs	4 * H100	vLLM	34B-65B	Serveur interne/cloud privé

Remerciements

Merci à l'IDRIS et au PNRIA

Merci à

- **Léo Nguyen**
- **Léo Hunout**
- **Nathan Caserreau**
- **Pascal Huynh et Dinh-Tuan Tran**

Questions ???

Contact

mail : antoine.regnier@idris.fr

