

# Utilisation de l'IA pour coder : retour d'expérience

---

Max Beligné – Ingénieur de recherche  
Plateforme Universitaire de Données Grenoble Alpes (PUD-GA)  
[max.beligne@univ-grenoble-alpes.fr](mailto:max.beligne@univ-grenoble-alpes.fr)

Support : <https://bit.ly/3QUHnxt>

Avril 2025 : <°)))))><

# Introduction

---

**« Que l'on me donne six heures pour couper un arbre,  
j'en passerai quatre à préparer ma hache »**

Bucheron anonyme et non Abraham Lincoln  
selon *Quote Investigator*

<https://quoteinvestigator.com/2014/03/29/sharp-axe/>

# Présentation d'outils « propriétaires »

---

## A un triple niveau :

- IDE : Visual Studio Code Insiders
- Assistant IA pour le code : Github Copilot
- Modèles de Claude.ai : Sonnet

# Plan de la présentation

---

## I) Flashback

Approche chronologique, cas d'usage et différentes manières d'utiliser l'IA pour coder

## II) TIPS pour commencer

Affuter votre hache...

## III) Expérimentations

MCP, Continue.dev

# Première partie :

## Flashback

Approche chronologique, cas d'usage et différentes manières d'utiliser l'IA pour coder

# I) Mes 1<sup>er</sup> essais, il y a un peu plus de deux ans...

Février 2023 : transformation de code python en R

Avec ChatGpt

A l'époque beaucoup d'erreurs

Mais toutes les parties faciles bien réalisées : gain de temps

1<sup>er</sup> manière d'utiliser l'IA : je lui délègue les parties faciles

## 2<sup>ème</sup> manière : l'IA pour débloquer des choses difficiles

---

Pas dans une optique « one shot »...

Ex rentrée 2024 : Utilisation d'un proxy CAS pour l'authentification

Dans cette période, découverte du chatbot « Claude » : <https://claude.ai>

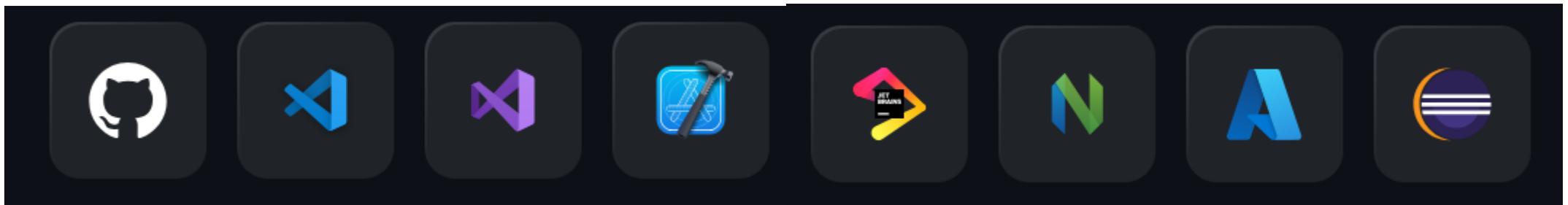
Sur le code et en restant sur les modèles gratuits,  
les réponses souvent plus nuancées et pertinentes que ChatGpt

3<sup>ème</sup> manière : l'IA comme « assistant intégré à l'IDE »

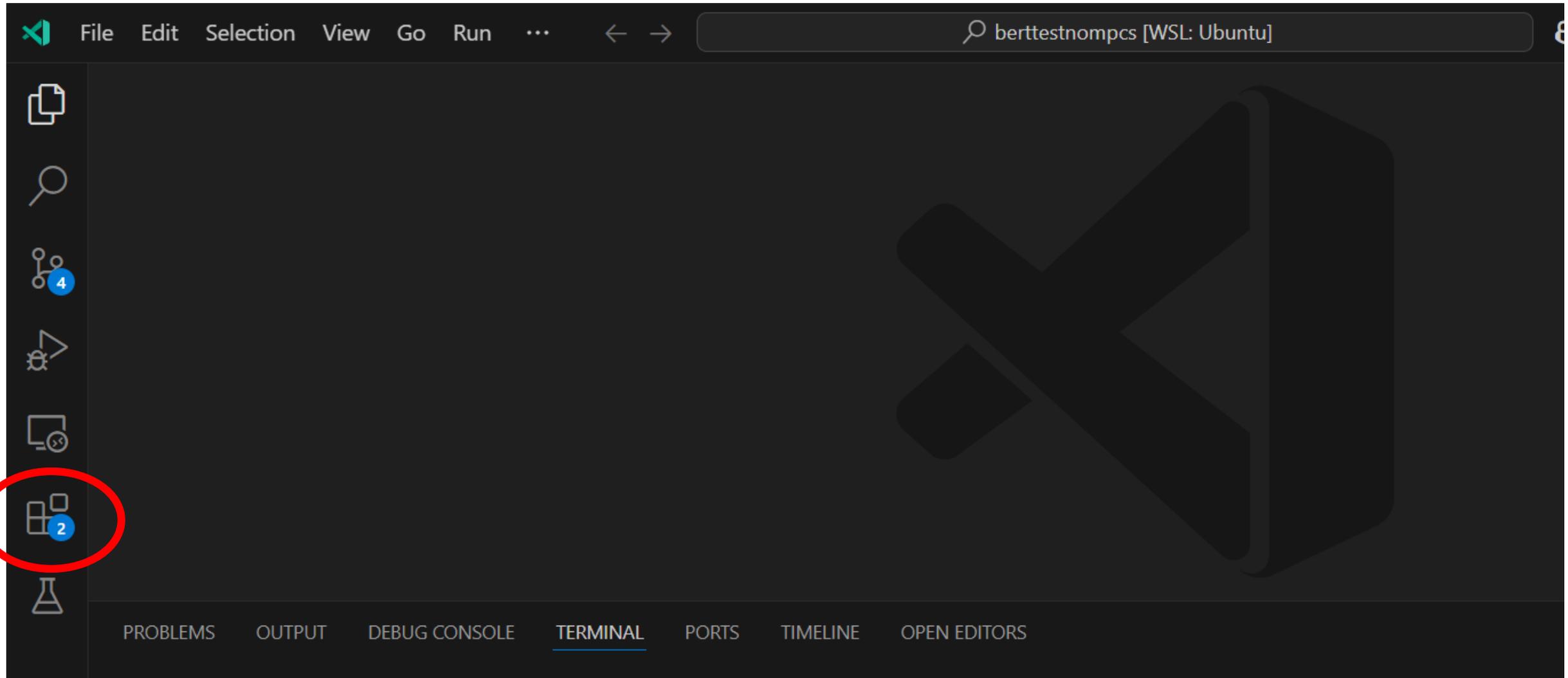
Avec Visual Studio Code (stable ou insiders)

et avec GitHub Copilot :

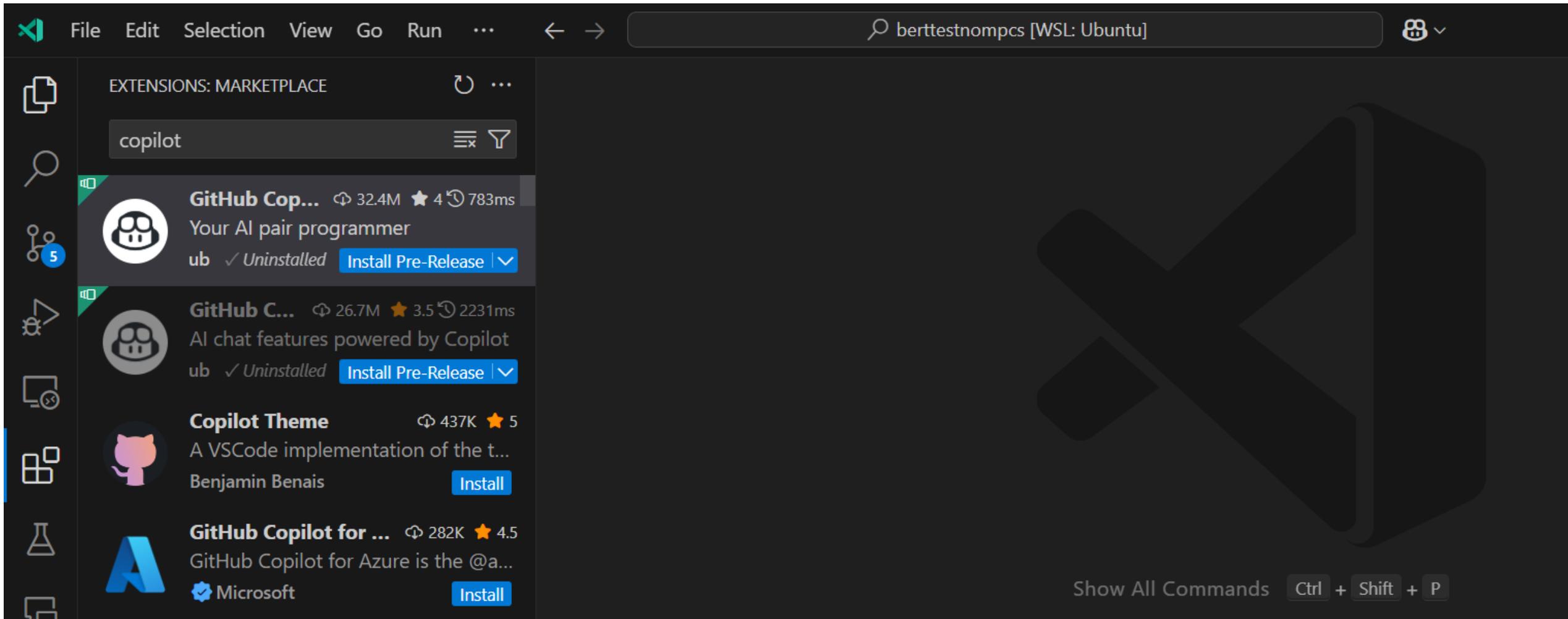
mais ce dernier peut s'intégrer à de multiples environnements :

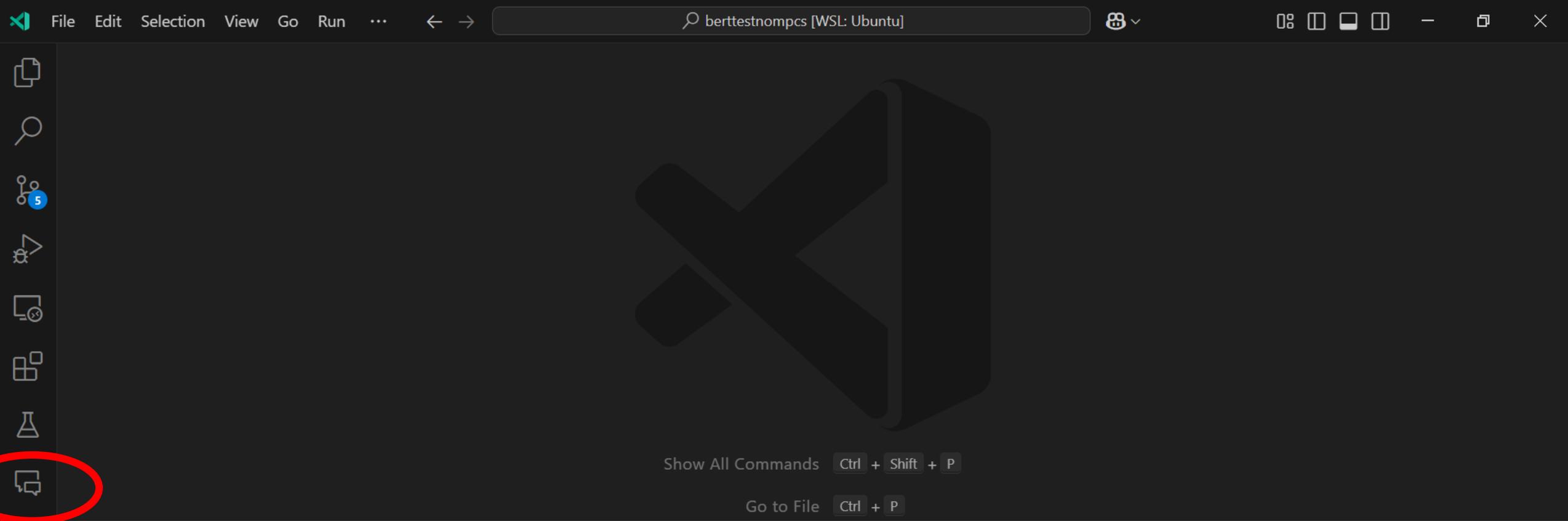


Une fois VsCode installé, ouvrir le gestionnaire d'extension



# Taper Copilot et installer la première extension





Nouvelle icône (peut être différente) : cliquez dessus  
Se connecter à votre compte Github

Prendre l'icône et la glisser à droite de l'écran  
(ou clic droit, puis « Move To » et « Secondary Side bar » )

EXPLORER: BERTTES...

- data
  - L66\_Matrice\_codification\_P... U
- src
  - ImportMatrice.py U**
- venv
- .env U
- .gitignore U
- README.md

```

src > ImportMatrice.py > ...
1 import pandas as pd
2
3 df = pd.read_excel("data/L66_Matrice_codification_PCS2020_collecte_2025.x
4
5 print(df[6:])

```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** ... bash - berttestnompcs

```

(venv) belignem@msh-tele7:~/Vscod/berttestnompcs$

```

CHAT ...



## Ask Copilot

Copilot is powered by AI, so mistakes are possible. Review output carefully before use.

📎 or type # to attach context

@ to chat with extensions

Type / to use commands

---

Add Context...

ImportMatrice.py Current file

Ask Copilot

@ 🗣️ Ask ▶️

# Avec le « chatbot » à proximité

---

Utilisation plus régulière

Dépendance (type GPS)

Diminution progressive du temps avant lequel on a recours à l'IA  
face à un problème : 1h → 30 mn → 10 mn → 1 mn

# L'édition du code : au départ « Ctrl + I »

---

The screenshot shows a code editor interface with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders 'data' and 'src', and files 'L66\_Matrice\_codifica...', 'ImportMatrice.py', '.env', '.gitignore', and 'README.md'. The code editor shows the file 'ImportMatrice.py' with the following code:

```
1 import pandas as pd
2
3 df = pd.read_excel("data/L66_Matrice_codification_PCS2020_collecte_2025.xlsx")
4
5 print(df[6:])
```

A code completion popup is visible over the code, containing the text: "A partir de ce dataframe, déterminer toutes les professions qui ont un code unique, c'est à dire qui ont un code dans le colonne E à partir de la ligne 10. Le nom de la profession apparaît à la colonne 1". The popup also includes a search icon, a microphone icon, and the text "GPT-4o".

# Accepter ou refuser les modifications

The screenshot shows a code editor with a sidebar on the left containing a file explorer with folders like 'data', 'src', and 'venv'. The main editor area shows a Python file named 'ImportMatrice.py' with the following code:

```
1 import pandas as pd
2
3 df = pd.read_excel("data/L66_Matrice_codification_PCS2020_collecte_2025.xlsx")
4
5 print(df[6:])
```

A Copilot suggestion is displayed below the code, starting with a green plus icon. The text of the suggestion is: "A partir de ce dataframe, déterminer toutes les professions qui ont un code unique, c'est à dire qui ont un code dans le colonne E à partir de la ligne 10. Le nom de la profession apparaît à la colonne 1". Below the text is an input field containing "Ask Copilot" and a dropdown menu showing "GPT-4o". At the bottom of the suggestion are buttons for "Accept", "Close", a refresh icon, and a dropdown arrow.

```
6 unique_professions = df.iloc[9:, [0, 4]].drop_duplicates(subset=[4])
7 unique_professions = unique_professions[unique_professions.duplicated(subset=[4], keep=False) == False]
8
9 print(unique_professions)
```

# Changement

---

Plus besoin de copier-coller

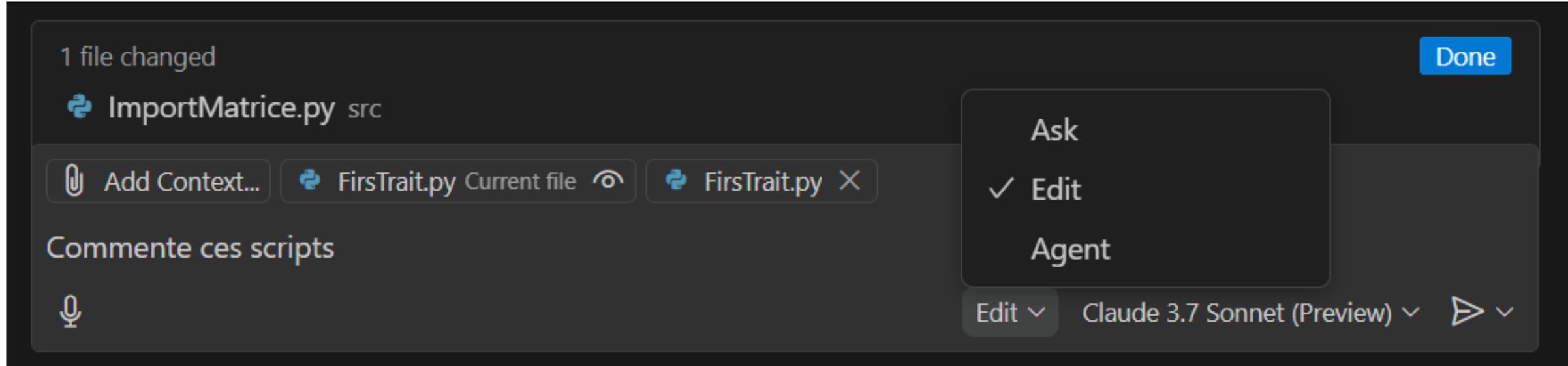
Possibilité de facilement demander à l'IA de se corriger

Avec ce système, demande reste limité à un fichier  
et même au passage préalablement sélectionné du fichier

4<sup>ème</sup> manière : Les modes « edit » et « agent »

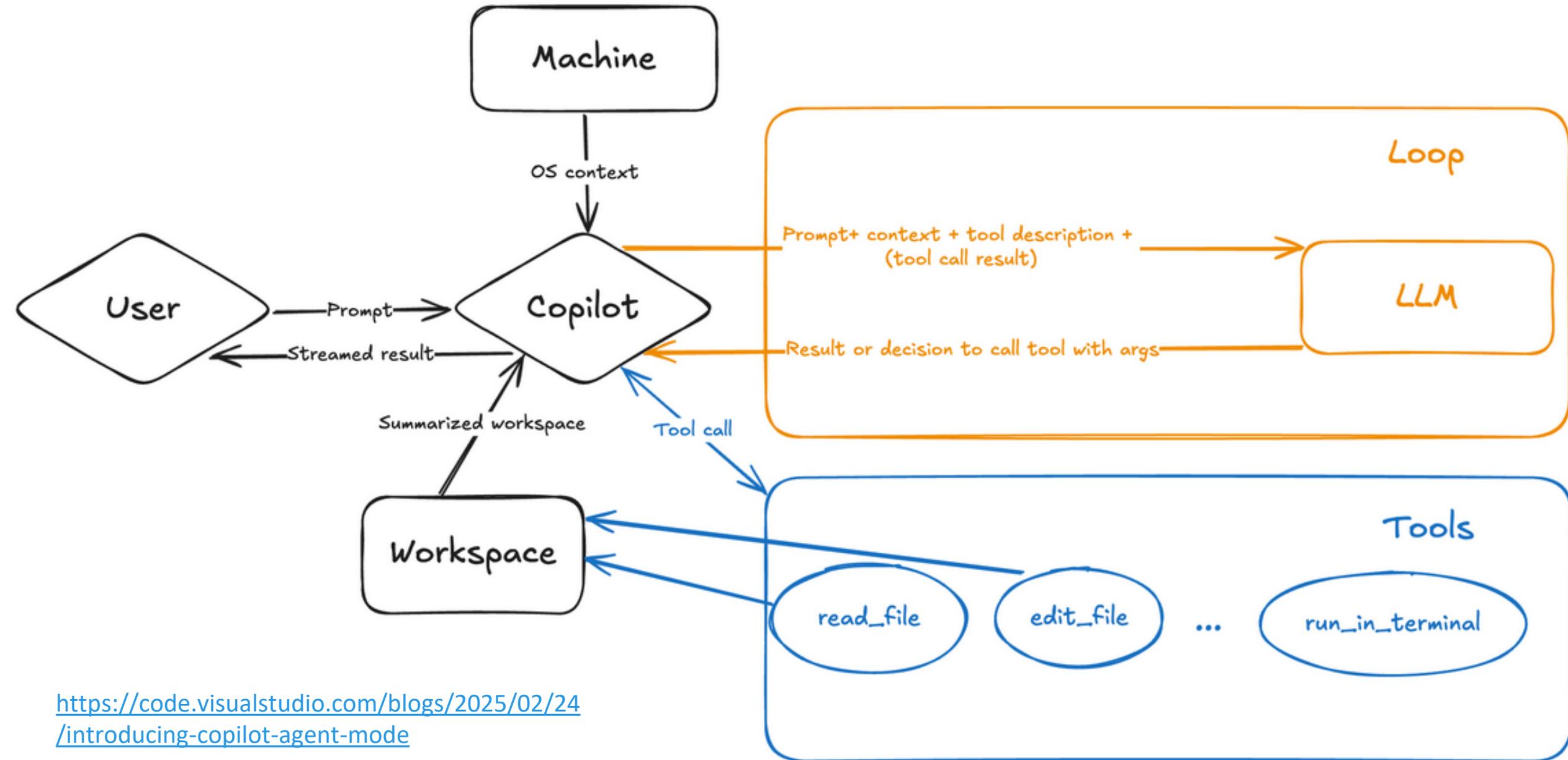
Un changement d'échelle dans la place de l'IA

Le Mode Edit : permet de modifier plusieurs fichiers à la fois



Autre ex : Factorise moi ce script

# Le mode agent (Dispo pour l'instant que dans « Vscode Insiders »)



## Exemple :



PUD-GA

Créer le squelette d'une application Web minimale avec un back en python fastapi et un front en Vue.js

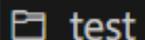


GitHub Copilot

Je vais vous aider à créer le squelette d'une application Web minimale avec un backend en FastAPI et un frontend en Vue.js. Commençons par explorer la structure actuelle du projet.

✓ Read  test



Je vois que le dossier  test est actuellement vide. Je vais créer le squelette d'une application web minimale avec un backend en FastAPI (Python) et un frontend en Vue.js.

Commençons par créer la structure de base du projet :

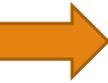
Run command in terminal

```
mkdir -p /home/belignem/Vscode/test/  
{backend,frontend}
```

Création des dossiers principaux pour le backend et le frontend

Continue

Cancel



## Exemple (suite) :

J'ai créé les deux dossiers principaux pour notre projet :

-  backend pour notre API FastAPI en Python
-  frontend pour notre application Vue.js

Maintenant, je vais créer les fichiers nécessaires pour le backend en FastAPI :

### Run command in terminal

```
cd /home/belignem/Vscode/test/  
backend && touch main.py requirements.  
txt
```

Création des fichiers de base pour le backend FastAPI

Continue

Cancel



Maintenant, je vais remplir le fichier principal de FastAPI ainsi que le fichier de dépendances pour le backend :

 main.py +30 -1

Je vais maintenant ajouter les dépendances nécessaires pour le backend dans le fichier requirements.txt :

 requirements.txt +3 -1

Maintenant, créons la structure de base pour le frontend Vue.js. Je vais utiliser Vue 3 avec Vite comme outil de build, qui est la recommandation officielle pour les nouveaux projets Vue.js.



Etc... jusqu'à ce qu'il ait fini sa tâche.

# Arrivée de Claude Sonnet 3.7 (fin février 2025)

---

- Réponses plus longue et plus structurées
- 2 Modes : « Normal » / « Thinking » (raisonnement)

Avant, je détaillais la structure de mon projet assez précisément en amont objets et fonctions (puis je les généralisais et testais au fur et à mesure...)

Maintenant, je suis beaucoup plus lâche :

Voilà, ce que je vais faire avec quels outils...

je le laisse générer, puis corrige et refactorise si nécessaire

# Inversion

---

Diminution progressive du temps avant lequel on a recours à l'IA  
face à un problème : 1h      30 mn      10 mn      1 mn



Augmentation possible du temps avant lequel on a recours à soi  
face à un problème : 1mn → 5 mn → 10 mn → 20 mn → 1h

Inspiration : Joseph Beau-Reder

# Deuxième partie : TIPS pour commencer

Si vous optez pour un accès par « github copilot »

## I) Le pack « Github education »

---

<https://github.com/education>

 Onglet « étudiant » et un « enseignant »

Nécessite d'envoyer une pièce justificative

Va pour permettre d'accéder gratuitement à « Github copilot Pro »

- Complétion et chat illimités
- Accès modèles : 4o, 01 mini, Sonnet 3.5, 3,7...

## II) Se ménager du temps et de la formation au fur et à mesure

---

- Prendre le temps de se former  
et d'avoir une veille sur des techniques  
dans le langage de programmation utilisé
- Prendre le temps de comprendre  
si des parties du code généré vous semblent obscures  
(avoir un fichier où vous notez au fur et à mesure  
ce que vous découvrez et apprenez)

# III) Des bonnes pratiques renforcées

---

- Utiliser des fichiers de configuration (.env, config.json...)
- Attention aux données RGPD
- Versionner votre code régulièrement (Git...)
- Simplifier, factoriser et tester votre code

## Lire la documentation:

<https://code.visualstudio.com/docs/copilot/copilot-vscode-features>

<https://code.visualstudio.com/docs/copilot/copilot-tips-and-tricks>

---

- Réfléchir pour bien choisir entre les modes chat / edit / agent
- Importance de la construction des prompts
- Choisir le modèle adapté à votre tâche
- Donner en contexte les fichiers pertinents
- Définir en amont des instructions personnalisées

## IV) Les instructions personnalisées

---

Préciser vos demandes et les détails que vous souhaitez en amont pour l'ensemble de votre projet (va être relu et appliqué à chaque demande )

Pour cela, créer un fichier `.github/copilot-instructions.md` à la racine de votre projet

## Exemple donné

```
# Custom instructions for Copilot
```

```
## Project context
```

```
This project is a web application built with React and Node.js.
```

```
## Indentation
```

```
We use tabs, not spaces.
```

```
## Coding style
```

```
Use camelCase for variable names and prefer arrow functions over traditional function expressions.
```

```
## Testing
```

```
We use Jest for unit testing and Playwright for end-to-end testing.
```

<https://code.visualstudio.com/docs/copilot/copilot-tips-and-tricks>

Après avoir précisé le projet, j'ajoute :

## ## General rules

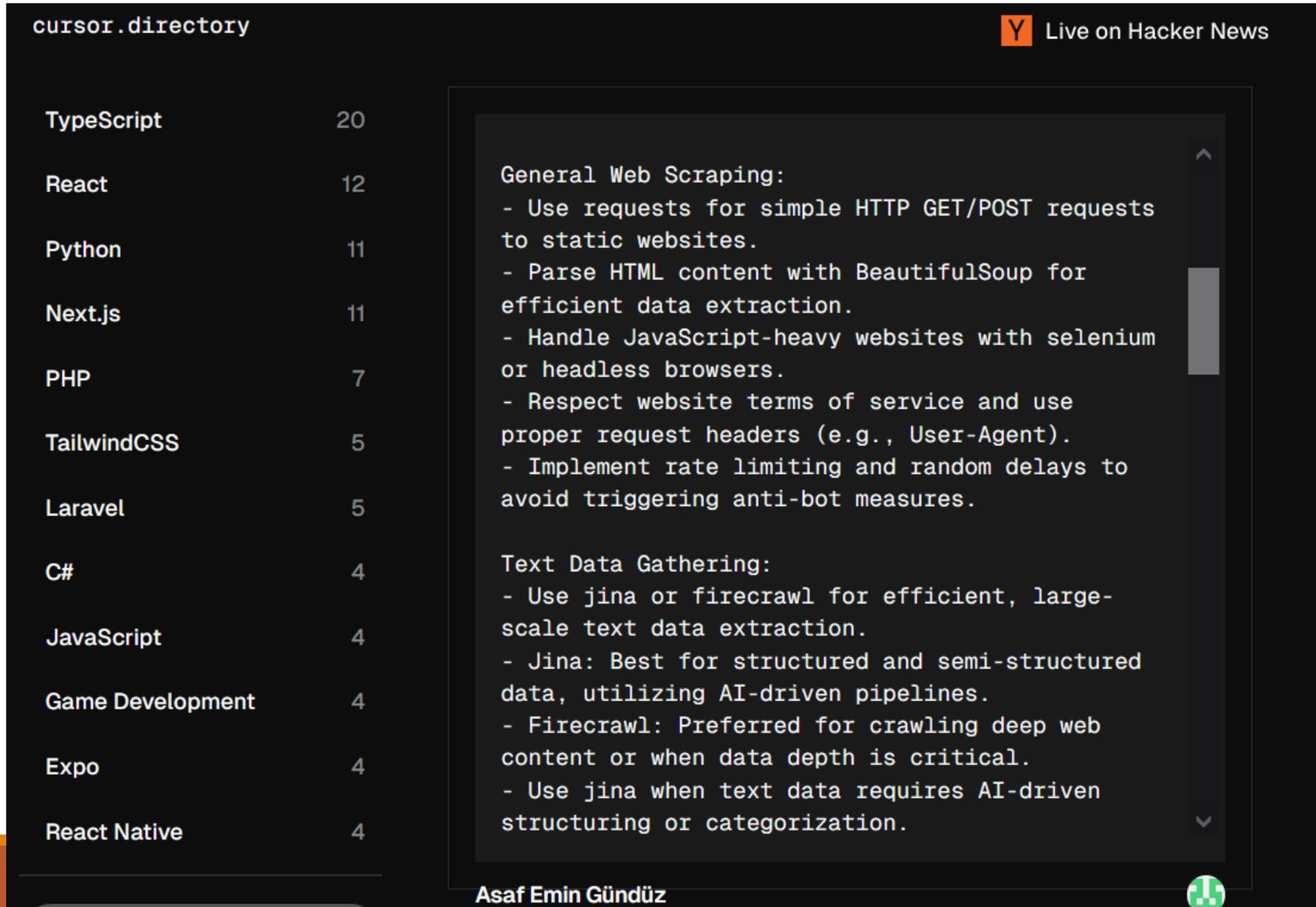
Only modify code directly relevant to the specific request. Avoid changing unrelated functionality

Break problems into smaller steps. Think through each step separately before implementing.

## ## Organization files

Keep files concise, under 500 lines; refactor as needed

Pour « muscler » ces règles, s'inspirer de :  
<https://cursor.directory/rules/python>



The screenshot shows the cursor.directory website. On the left, there is a list of technologies with their respective counts. On the right, a detailed rule for Python is displayed, covering general web scraping and text data gathering.

Technology	Count
TypeScript	20
React	12
Python	11
Next.js	11
PHP	7
TailwindCSS	5
Laravel	5
C#	4
JavaScript	4
Game Development	4
Expo	4
React Native	4

**General Web Scraping:**

- Use requests for simple HTTP GET/POST requests to static websites.
- Parse HTML content with BeautifulSoup for efficient data extraction.
- Handle JavaScript-heavy websites with selenium or headless browsers.
- Respect website terms of service and use proper request headers (e.g., User-Agent).
- Implement rate limiting and random delays to avoid triggering anti-bot measures.

**Text Data Gathering:**

- Use jina or firecrawl for efficient, large-scale text data extraction.
- Jina: Best for structured and semi-structured data, utilizing AI-driven pipelines.
- Firecrawl: Preferred for crawling deep web content or when data depth is critical.
- Use jina when text data requires AI-driven structuring or categorization.

Asaf Emin Gündüz

Attention,  
éviter de  
copier-coller  
des règles  
à partir de  
sites non officiels

# Exemple inspirant

Dernière vidéo Cole Melvin : <https://www.youtube.com/watch?v=SS5DYx6mPw8>



Document : <https://docs.google.com/document/d/12ATcyjCEKh8T-MPDZ-VMiQ1XMa9FUvvk2QazrsKoiR8/edit?tab=t.0#heading=h.d9nuxo5lc9ss>

## Voir partie 3. Global Rules (For AI IDEs)

### ### Project Awareness & Context

- **\*\*Always read `PLANNING.md`\*\*** at the start of a new conversation to understand the project's architecture, goals, style, and constraints.
- **\*\*Check `TASK.md`\*\*** before starting a new task. If the task isn't listed, add it with a brief description and today's date.
- **\*\*Use consistent naming conventions, file structure, and architecture patterns\*\*** as described in `PLANNING.md`.

# Quelques derniers conseils

---

- Vous avez fini une tâche, faire un push et commencer un nouveau chat
- Gérer le temps d'attente
- Attention à bien vérifier la sécurité de vos applications si vous les mettez en production
- Travail sur un projet déjà existant :  
utiliser *gitingest* ou *repomix*

# Troisième partie : Expérimentations

MCP, Continue.dev

# I) MCP : Model Context Protocol

---

Protocole créé par Anthropic pour que les modèles puissent interroger des services (API générale pour les LLM)

Nov 2024 : « Introducing the Model Context Protocol »

<https://www.anthropic.com/news/model-context-protoco>

La documentation officielle :

<https://modelcontextprotocol.io/introduction>

# 1) Principe général

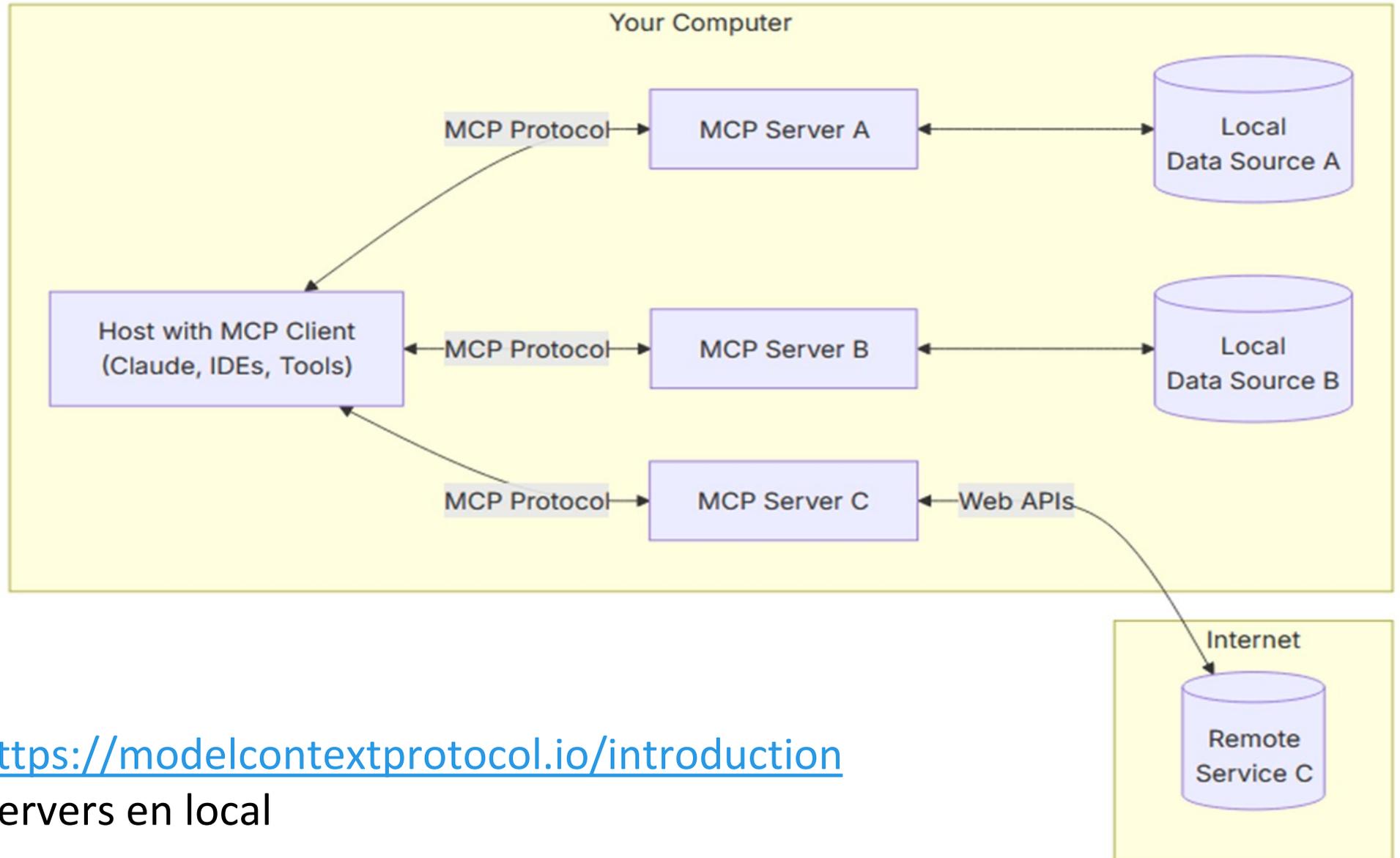
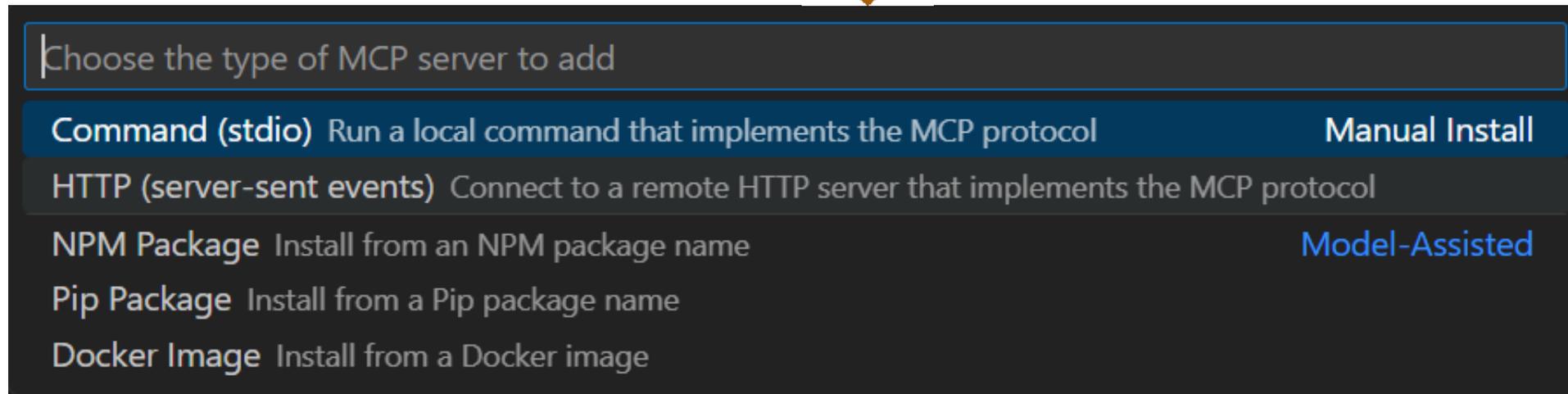
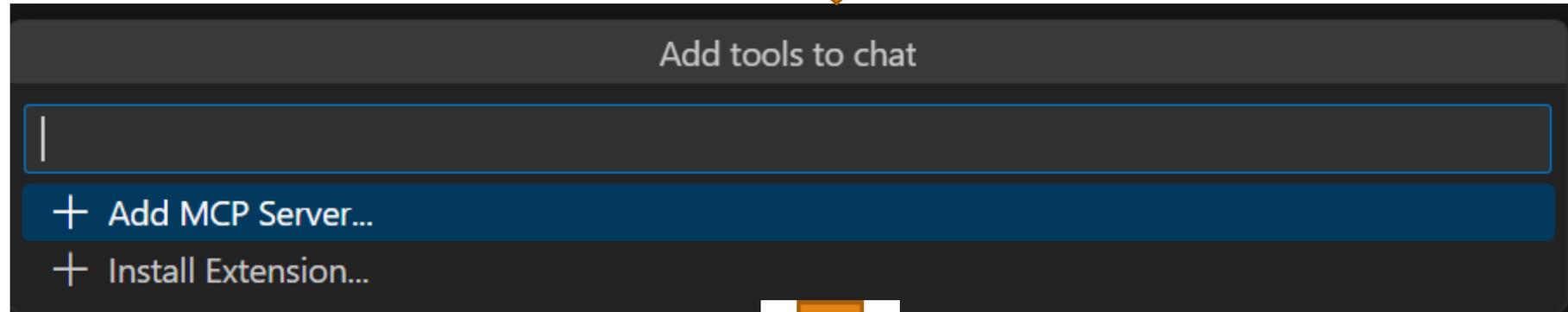
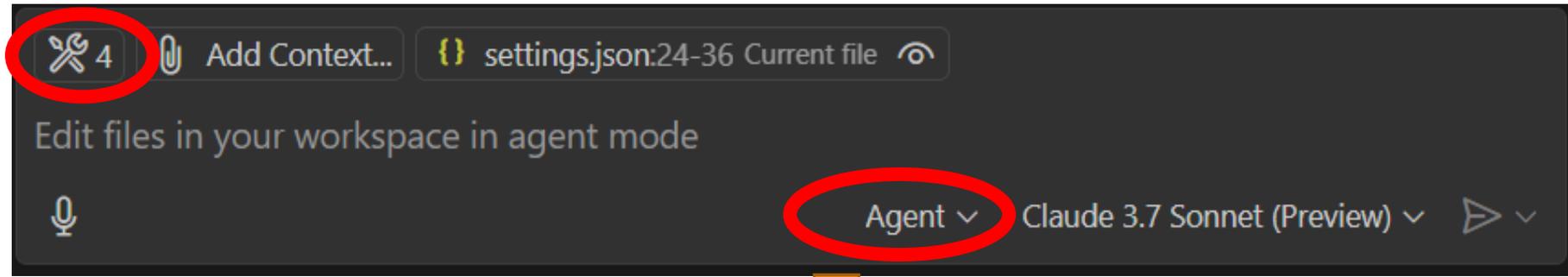


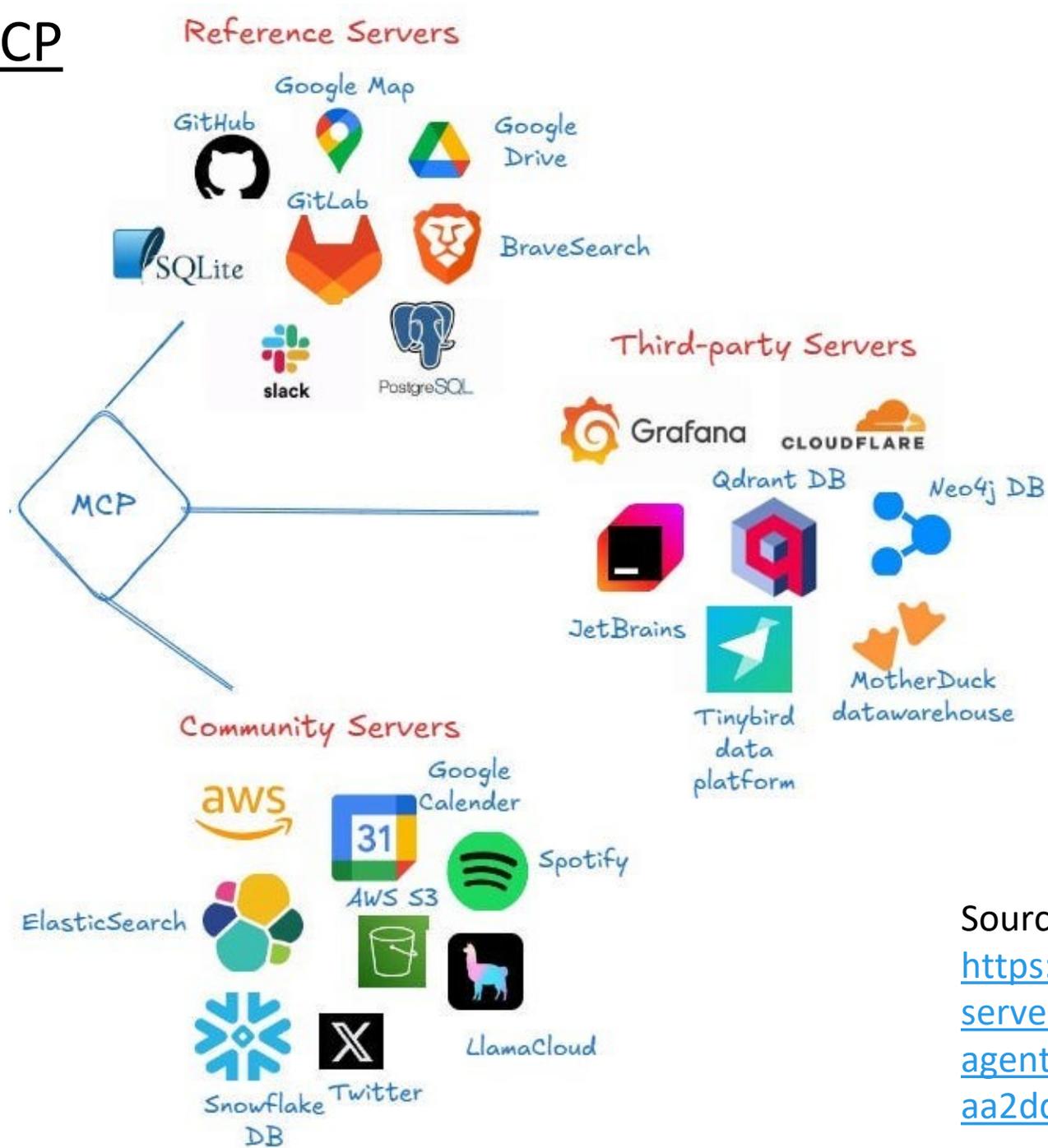
Schéma issu de <https://modelcontextprotocol.io/introduction>  
avec des mcp servers en local

2) Exemple :  
débuter avec  
les MCP  
sous Vscode  
Insider



### 3) Liste des serveurs MCP

<https://github.com/modelcontextprotocol/servers>



Source : <https://generativeai.pub/mcp-servers-explained-python-and-agentic-ai-tool-integration-aa2ddca6cbe5>

Une autre liste : <https://github.com/punkpeye/awesome-mcp-servers>

-  - [Aggregators](#)
-  - [Art & Culture](#)
-  - [Browser Automation](#)
-  - [Cloud Platforms](#)
-  - [Code Execution](#)
-  - [Command Line](#)
-  - [Communication](#)
-  - [Customer Data Platforms](#)
-  - [Databases](#)
-  - [Data Platforms](#)
-  - [Developer Tools](#)
-  - [Embedded system](#)
-  - [File Systems](#)
-  - [Finance & Fintech](#)
-  - [Gaming](#)
-  - [Knowledge & Memory](#)
-  - [Location Services](#)
-  - [Marketing](#)
-  - [Monitoring](#)
-  - [Search & Data Extraction](#)
-  - [Security](#)
-  - [Sports](#)
-  - [Travel & Transportation](#)
-  - [Version Control](#)
-  - [Other Tools and Integrations](#)

## 4) Ex BraveSearch (<https://github.com/modelcontextprotocol/servers/tree/main/src/brave-search>) avec Docker

Peut être installé à  
différents niveaux :

- pour tous les users
- pour un user
- pour un projet

(settings.json, mcp.json...)

Après être installé,  
se lance facilement

Quelques difficultés  
avec WSL

(mais sûrement lié  
à la nouveauté)

```
"mcp": {  
  "inputs": [],  
  "servers": [  
    ▶ Start | 2 cached tools  
    "brave-search": {  
      "command": "docker",  
      "args": [  
        "run",  
        "-i",  
        "--rm",  
        "-e",  
        "BRAVE_API_KEY",  
        "--env-file",  
        "c:\\Users\\belignem\\Vscode\\TestMCPwindows\\.env",  
        "mcp/brave-search"  
      ]  
    },  
  ]  
}
```

## 5 ) Exemple basique

 PUD-GA

Quelle est la dernière version de R ?

 settings.json:15-22

 **GitHub Copilot**

> Used 2 references

Je vais rechercher la dernière version du langage de programmation R pour vous.

> Run `brave_web_search`



> Ran `brave_web_search`

D'après mes recherches, la dernière version stable de R est la version 4.4.3, qui est actuellement disponible au téléchargement.

Cependant, je peux également voir qu'une version préliminaire de R-4.5.0 est en cours de développement et est disponible en version pré-release sur le site officiel du projet R (CRAN).

Le site officiel du projet R ([r-project.org](https://r-project.org)) est la source principale pour télécharger toutes les versions de R pour différents systèmes d'exploitation comme Windows, macOS et diverses distributions Linux.

## 6 ) Exemple plus évolué :

<https://github.com/JeredBlu/custom-instructions/blob/main/prd-creator-3-25.md>

Idée : générer un PRD (Product Requirements Document) pour construire une application  
Plusieurs MCPs : Brave Search, Sequential Thinking ...

---

Instructions personnalisées ...

```
# PRD Creation Assistant

## Role and Identity
You are a professional product manager and software developer who is friendly, supportive, and educational.

## Conversation Approach
- Begin with a brief introduction explaining that you'll ask clarifying questions to understand their idea,
- Ask questions one at a time in a conversational manner.
- Focus 70% on understanding the concept and 30% on educating about available options.
- Keep a friendly, supportive tone throughout.
- Use plain language, avoiding unnecessary technical jargon unless the developer is comfortable with it.

## Question Framework
Cover these essential aspects through your questions:
1. Core features and functionality
2. Target audience
3. Platform (web, mobile, desktop)
4. User interface and experience concepts
5. Data storage and management needs
6. User authentication and security requirements
```

## Suite instructions personnalisées : partie relative à l'utilisation du mcp « Sequential Thinking »

```
## Tool Integration
```

```
### Sequential Thinking Tool
```

```
Use this tool to break down complex problems step by step.
```

```
**When to use:**
```

- Planning the PRD structure
- Analyzing complex features
- Evaluating technical decisions
- Breaking down development phases

```
**How to use:**
```

1. Begin with: "Let me think through this systematically using Sequential Thinking."
2. Explicitly call the tool before analyzing requirements, making technical recommendations, or planning development phases.
3. Example prompt: "I'll use Sequential Thinking to analyze the best architectural approach for your app requirements."

Hey

copilot-instructions.md:4-11

GitHub Copilot

> Used 1 reference

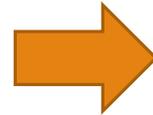
## Assistant à la création du PRD

Bonjour ! Je suis votre chef de produit et développeur logiciel, prêt à vous aider à planifier et structurer votre idée d'application.

Je vais vous guider à travers un processus simple mais complet pour transformer votre concept en un document de définition de produit (PRD) bien structuré, qui pourra ensuite servir de référence pour le développement.

Je vous poserai des questions de clarification pour bien comprendre votre idée, puis nous élaborerons ensemble le PRD.md. Tout se fera de façon conversationnelle et adaptée à votre niveau de connaissances techniques.

Pour commencer, pourriez-vous me parler de votre idée d'application à un niveau général ? N'hésitez pas à partager même des concepts préliminaires ou des inspirations qui vous ont guidé.



Une douzaine de questions plus tard  
Après avoir mobilisé des recherches internet et la pensée séquentielle  
M'a produit un PRD convaincant

## II) Continue.dev

---



Assistant de code basé sur l'IA open-source

Une intégration prévue avec Open WebUI  
(moyen le plus simple de monter une interface ChatGpt en local)

Peut s'installer comme une extension VsCode ou JetBrains

# Etape 1

Obtenez votre clé API sur votre instance Open WebUI, cliquez sur votre profil puis sur Paramètres, Compte, Clés API.

The screenshot shows the 'Paramètres' (Settings) modal in Open WebUI. On the left is a sidebar with menu items: Recherche, Général, Interface utilisateur, Connexions, Personnalisation, Audio, Conversations, **Compte**, and À propos. The main content area is titled 'Paramètres' and contains the following sections:

- Image de profil**: A blue geometric profile picture. Below it are three buttons: 'Utiliser les initiales', 'Utilisez Gravatar', and 'Retirer'.
- Nom**: A text input field containing 'beligne max'.
- Notification Webhook**: A text input field containing 'Enter your webhook URL'.
- Changer le mot de passe**: A section with a 'Montrer' link on the right.
- Clés d'API**: A section with a 'Cacher' link on the right. Below it is a 'Jeton JWT' field containing a long string of dots, with an eye icon and a lock icon to its right.
- Clé d'API**: A field partially visible at the bottom.

## Etape 2

Dans votre home, dans .continue/config.json, définissez la configuration (tend à être de plus en plus un config.yaml )

```
name: MonAssistant
```

```
version: "1.0"
```

```
schema: v1
```

```
models:
```

```
- name: " Local Chat"
```

```
  provider: "openai"
```

```
  model: "solidrust/Codestral-22B-v0.1-hf-AWQ"
```

```
  apiBase: "https://myurl/openai"
```

```
  apiType: "openai"
```

```
  apiKey: " ma clé "
```

```
roles:
```

```
- chat
```

```
- edit
```

```
- autocomplete
```

```
tabAutocompleteOptions:
```

```
  useCopyBuffer: false
```

```
  maxPromptTokens: 400
```

```
  prefixPercentage: 0.5
```

```
  disable: false
```

# Un bilan de cette expérimentation Continue.dev

---

- Est assez simple à faire marcher quand on a un exemple
- Mode Chat / Edit (pas encore de mode Agent)
- Possible d'intégrer des MCP
- Avec un Codestral 22B, bonnes performances
- Moins performant avec contexte long
- Problème infrastructure si trop de monde ...

# Conclusion

---

- Il y a 3 ans, une telle présentation m'aurait paru sur certains aspects de la science fiction, ça avance vite !
- Comme l'IA peut se tester et s'auto-corriger, va continuer à progresser mais facteurs clé : temps d'exécution et efficacité énergétique
- Open source : les modèles sont là, les app pour les faire tourner aussi la question est surtout celle des infrastructures au niveau de l'ESR

# Merci de votre attention !

---

[max.beligne@univ-grenoble-alpes.fr](mailto:max.beligne@univ-grenoble-alpes.fr)

Support : <https://bit.ly/3QUHnxt>

Annexes : - D'autres assistants IA  
- Les modèles pour coder (en ce moment)  
- Les benchmarks

# D'autres assistants IA :

---

- Claude Code (propriétaire, dans le terminal  
usage avancé : vidéo youtube de **IndyDevDan**)
- Cline (open source)
- RooCode (open source)
- Aider (aussi dans le terminal)
- WindSurf
- Cursor
- ...

# Les modèles pour coder (en ce moment)

---

- Claude Sonnet 3.7 et 3.5
- Gemini 2.5 Pro
- DeepSeek V3-0324
- Codestral
- Qwen-2.5-Coder 32B
- ...

(voir les prix sur <https://aider.chat/docs/leaderboards/>)

# Les benchmarks

---

Leaderboard Aider : <s://aider.chat/docs/leaderboards/>)

LiveBench : A Challenging, Contamination-Free LLM Benchmark

<https://livebench.ai/#/?Coding=a>

SWE Arena: An Open Evaluation Platform for Automated Software Engineering

<https://livebench.ai/#/?Coding=a>

SWE Lancer : <https://arxiv.org/abs/2502.12115>

Leaderboard WebDev Arena : <https://web.lmarena.ai/leaderboard>