



story.

# L'IA dans le développement logiciel

Outils, opportunités et défis





Your sound. Your story.

# Plan de la présentation

1

Introduction & fondamentaux des LLM

2

Panorama des outils & cas concrets

- Low code
- Assistant dev
- Agents IA

3

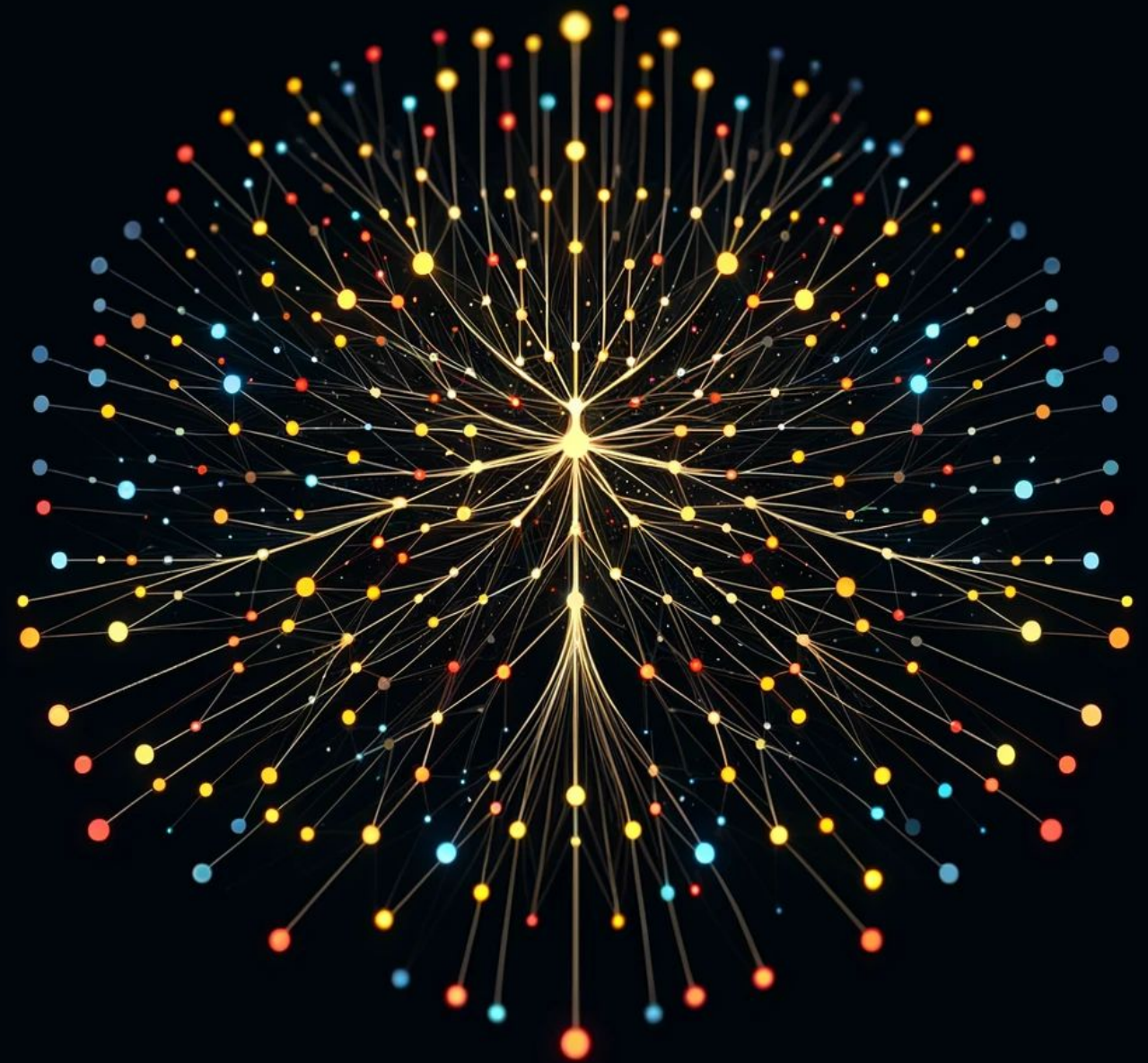
Opportunités & défis



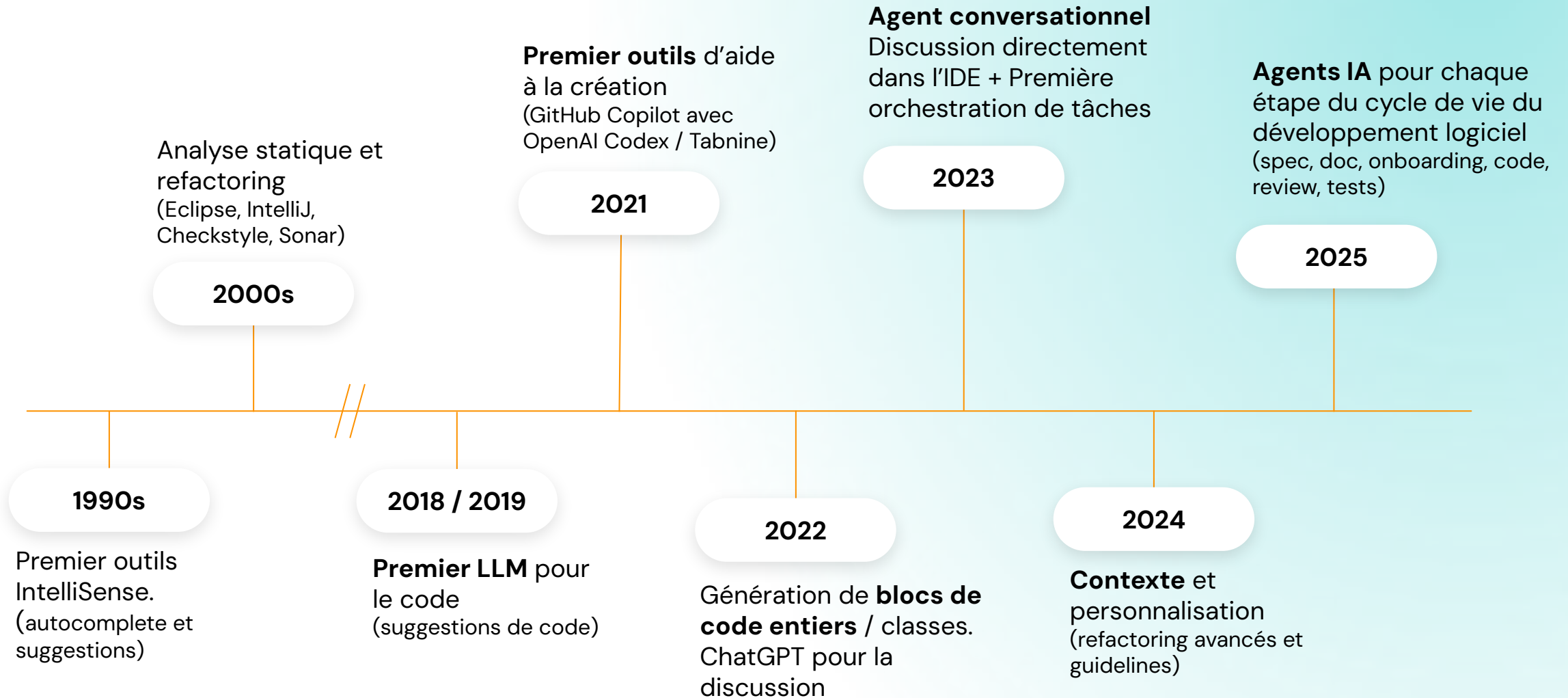
1

# Introduction & fondamentaux des LLM

---



# IA: un accélérateur inévitable pour le code ?

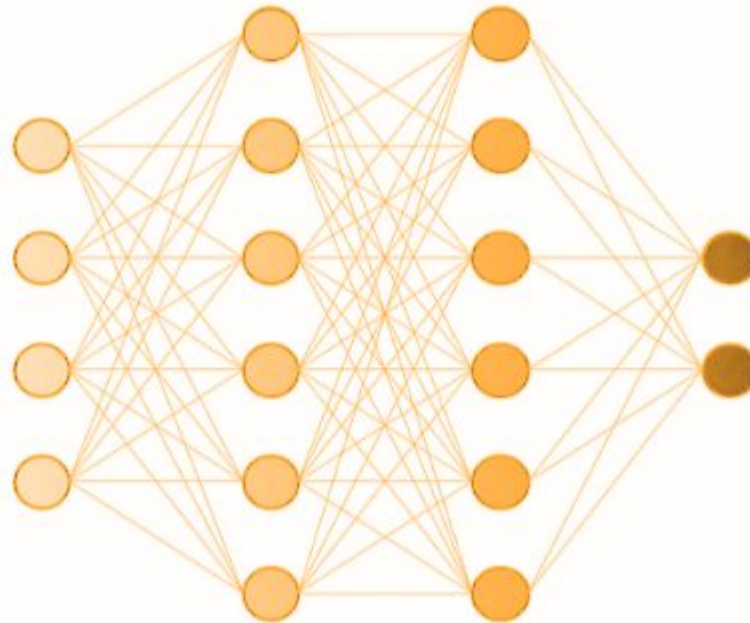


# LLM: qu'est-ce que c'est ?

**LLM** = Large Language Model = **Modèle d'IA** entraîné sur d'**énormes quantités de texte** qui a pour but de **prédire le mot / token suivant** d'une phrase en s'appuyant sur un **contexte**.

**Mike  
is  
quick.  
He  
moves**

*Input*



*Modèle*

Token	Probability
quickly	43%
fast	19%
like	17%
around	11%
...	...

*Output*

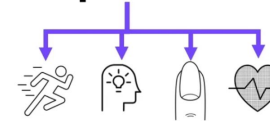
# LLM: comment ça marche ?

Un LLM est un modèle de probabilité, pas une source de vérité.

- Architecture "Transformer"
- Traiter et générer des données en séquence
- Plusieurs étapes sont calculées
  - Valeur des "tokens" (1 mot = 1 token)
  - Ajouter "l'attention" (position des mots)
  - Associer des poids différents aux tokens
  - Révéler une probabilité pour le prochain token

Mike	-0.0723849	-0.237343	0.22738675	-0.1565971	-0.0809315
is	0.01423593	-0.0437407	-0.0392998	-0.1253222	-0.1242475
quick	-0.0041666	-0.1149568	0.25142141	-0.1039098	0.03202019
.	0.04635777	-0.085105	0.13385425	0.04700242	0.02346371
He	-0.0551851	-0.0318704	0.18177843	0.04345733	-0.207683
moves	-0.0551851	-0.0318704	0.18177843	0.04345733	-0.207683

Mike is quick. He moves \_\_\_\_\_



Token	Probability
quickly	43%
fast	19%
like	17%
around	11%
...	...

Mike is quick. He moves quickly



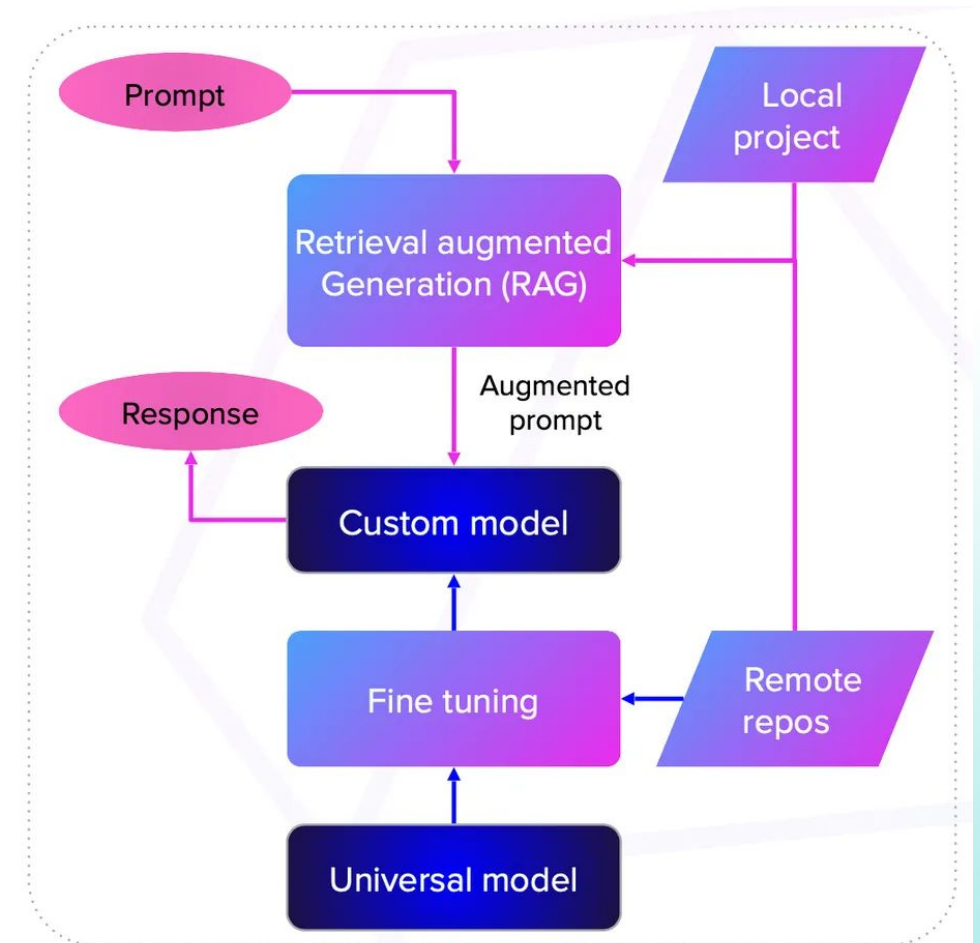
# LLM pour code: comment ça marche ?

## Il connaît mon style de code ?

- Mélange de RAG local et Fine-tuning avec le ou les repository au complet
- Guidelines s'ajoutent au prompt pour définir le style

→ Résultat beaucoup plus puissant qu'un ChatGPT sans contexte (fichiers, architecture, etc) !

Le secret réside dans le contexte, sa taille, la priorisation des éléments et la vitesse de réponse (inférence).



## 2 Panorama des outils & cas concrets





# Panorama : un outil pour chaque besoin

Low-Code

**BOLT**  **replit** 

Dev Assistant

 **CURSOR**  **GitHub Copilot**  **tabnine**

AI Agent

 **Devin**  **CODE BUDDY**  **SWE-agent**  
 **Enigma**

“ The company is no longer interested in professional coders, but rather in non-coders who can learn to use AI tools to create software. ”

Amjad Masad - CEO of Replit



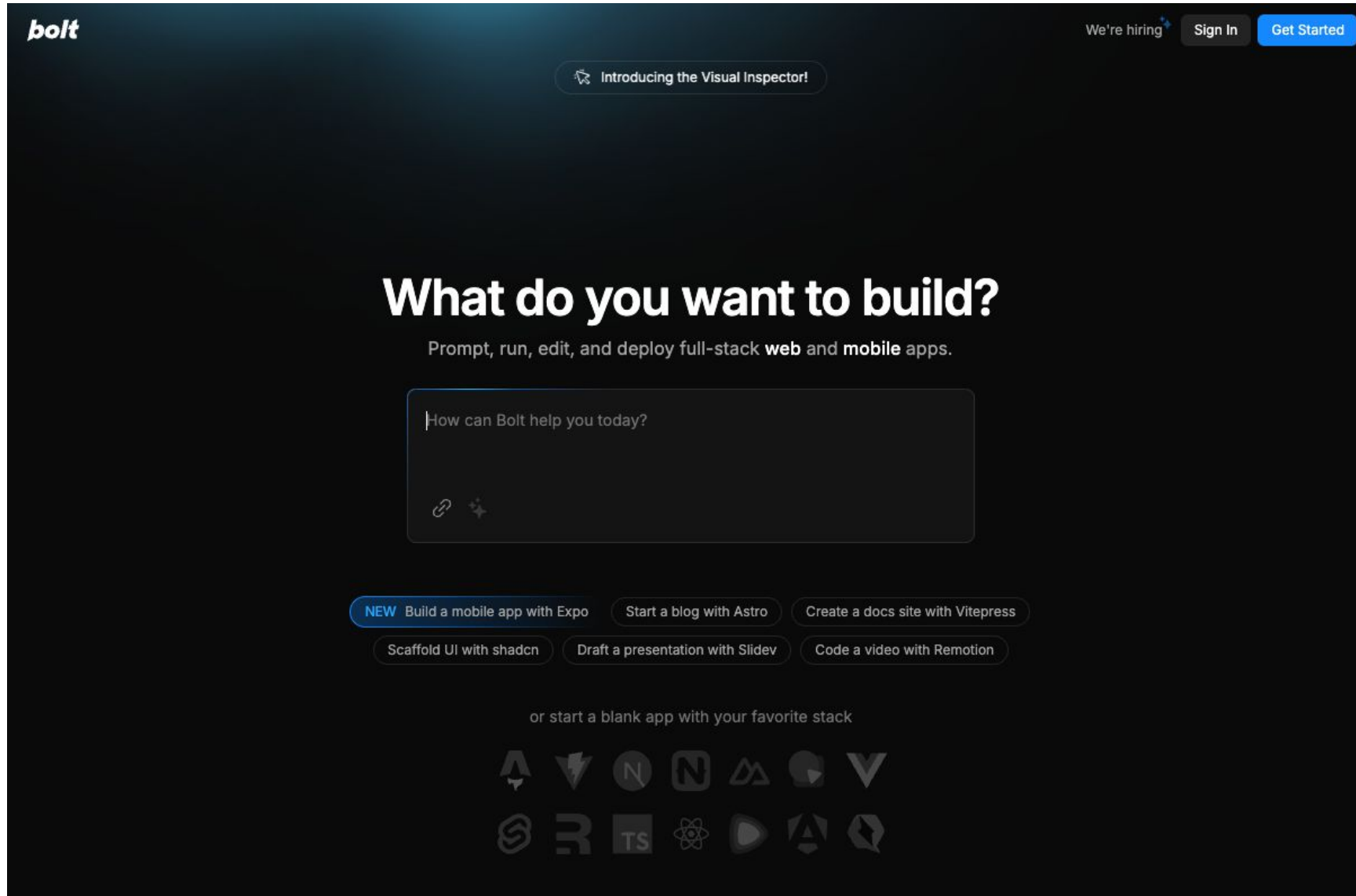
**Low-Code**  
**Démocratiser le**  
**développement**

---



***bolt*.new**

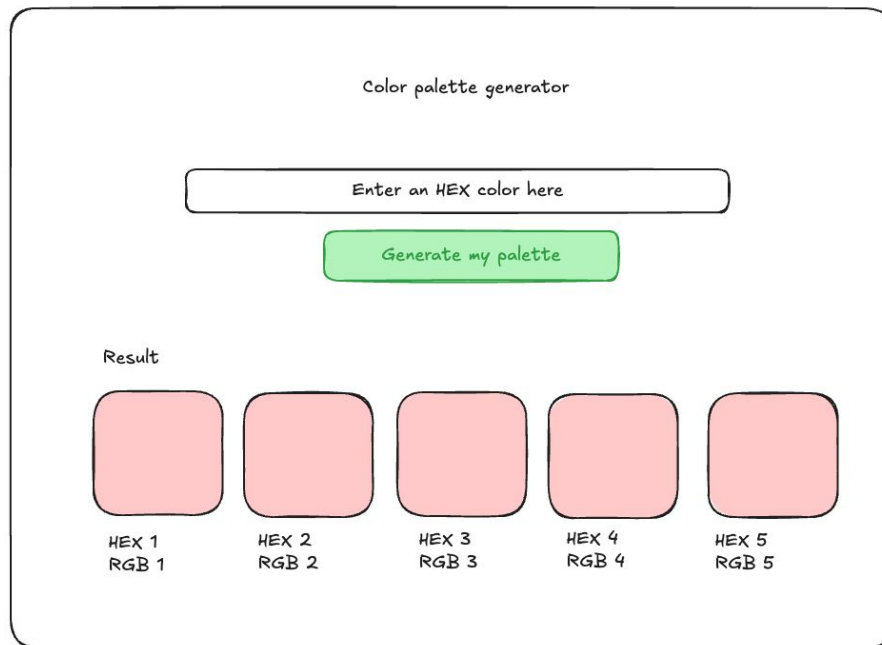
# Bolt.new : du prompt au déploiement en 2 min



# Bolt.new : du prompt au déploiement en 2 min

I want to build a web application where I can put a hex color and displays a five-color palette based on that hex.

Make it beautiful with loader and animation on reveal.



Create a web application that generates a harmonious 5-color palette based on a user-provided hex color code. Requirements:

#### UI/UX:

- Design a clean, modern interface with a prominent hex color input field
- Include a color picker widget for visual color selection
- Display the generated palette in large, equally-sized color swatches
- Show hex codes beneath each color swatch
- Add copy-to-clipboard functionality for each color code

#### Animations:

- Implement a smooth loading animation while generating the palette (duration: 0.8-1.2s)
- Add a subtle reveal animation for the palette swatches (fade-in + scale)
- Include hover effects on color swatches (slight elevation/scale)
- Animate color transitions when a new palette is generated

#### Technical Features:

- Validate hex color input (both with and without # prefix)
- Generate complementary, analogous, and monochromatic variations
- Ensure sufficient contrast between text and background colors
- Make the application responsive for all screen sizes
- Include error handling with user-friendly messages


#### Optional Enhancements:


- Save favorite palettes to local storage
- Export palette in different formats (CSS, SCSS, JSON)
- Display color names alongside hex codes
- Add light/dark theme toggle

# Bolt.new : du prompt au déploiement en 2 min



## What do you want to build?

Prompt, run, edit, and deploy full-stack web and mobile apps.



I want to build a web application where I can put a hex color and displays a five-color palette based on that hex. 

Make it beautiful with loader and animation on reveal.

  Use **Shift + Return** for a new line

### Color Palette Generator Implementation

- Export palette in different formats (CSS, SCSS, JSON)
- Display color names alongside hex codes
- Add light/dark theme toggle

I'll help you create a beautiful and functional color palette generator. Let's implement this with a modern, clean design and all the requested features.

**Color Palette Generator Implementation**

- ✓ Create initial files
- ✓ Install dependencies
- npm install
- ✓ Create src/utils/colorUtils.ts
- ✓ Create src/components/ColorSwatch.tsx
- Update src/App.tsx

```
import React, { useState } from 'react';
import { Palette, AlertCircle } from 'lucide-react';
import { ColorSwatch } from './components/ColorSwatch';
import { generatePalette, validateHexColor } from './utils/colorUtils';

function App() {
  const [inputColor, setInputColor] = useState('#6366F1');
  const [palette, setPalette] = useState<string[]>(<generatePalette('6366F1')>);
  const [error, setError] = useState('');
  const [isLoading, setIsLoading] = useState(false);

  const handleColorChange = (e: React.ChangeEvent<HTMLInputElement>) => {
    const value = e.target.value;
    setInputColor(value);
    setError('');
  };

  const handleSubmit = (e: React.FormEvent) => {
    e.preventDefault();
    let color = inputColor;

    if (!color.startsWith('#')) {
      color = '#' + color;
    }

    if (!validateHexColor(color)) {
      setError('Please enter a valid hex color code');
      return;
    }
  };
}
```

added 275 packages in 3s  
65 packages are looking for funding  
run `npm fund` for details

~/project 3s

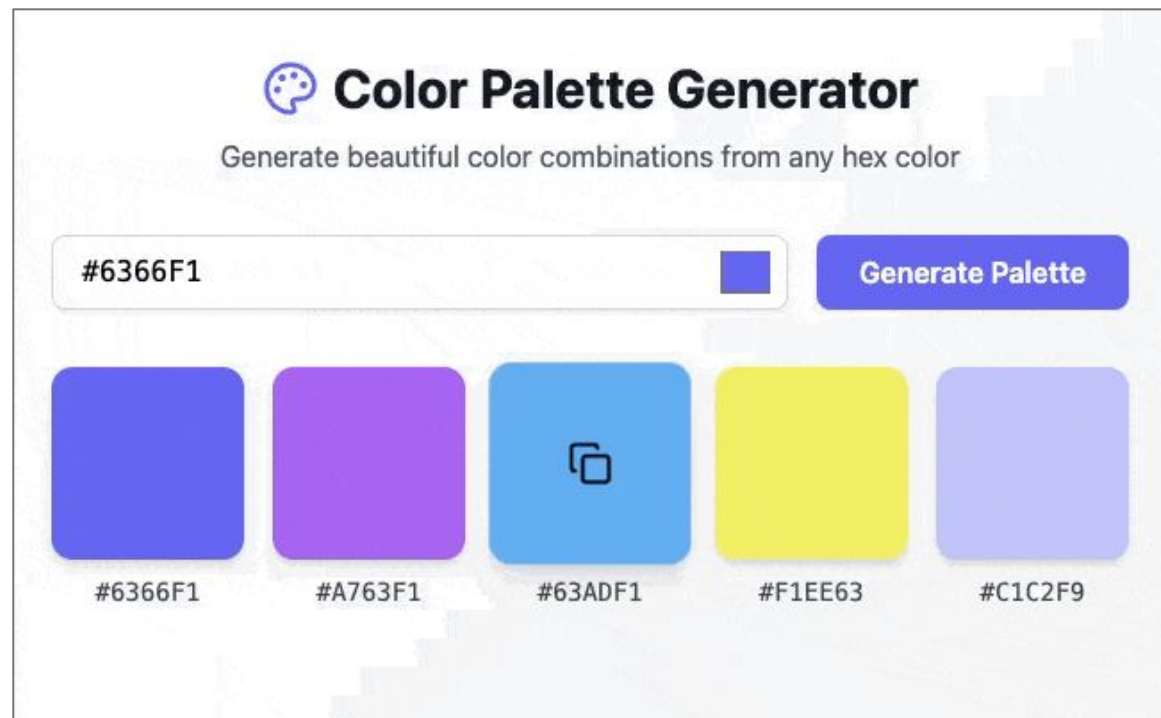


# Bolt.new : du prompt au déploiement en 2 min

- Projet sauvegardé
- Un clic pour déployer sur Netlify

<https://splendid-gelato-f0be0c.netlify.app/>

- Frameworks utilisés
  - React + Tailwind
  - Lucide-react pour les icônes
  - Vite pour bundle
- Code généré
  - 3 fichiers (App, ColorResult, utils)
- Code écrit : 0 ligne



# Bolt.new : du prompt au déploiement en 2 min

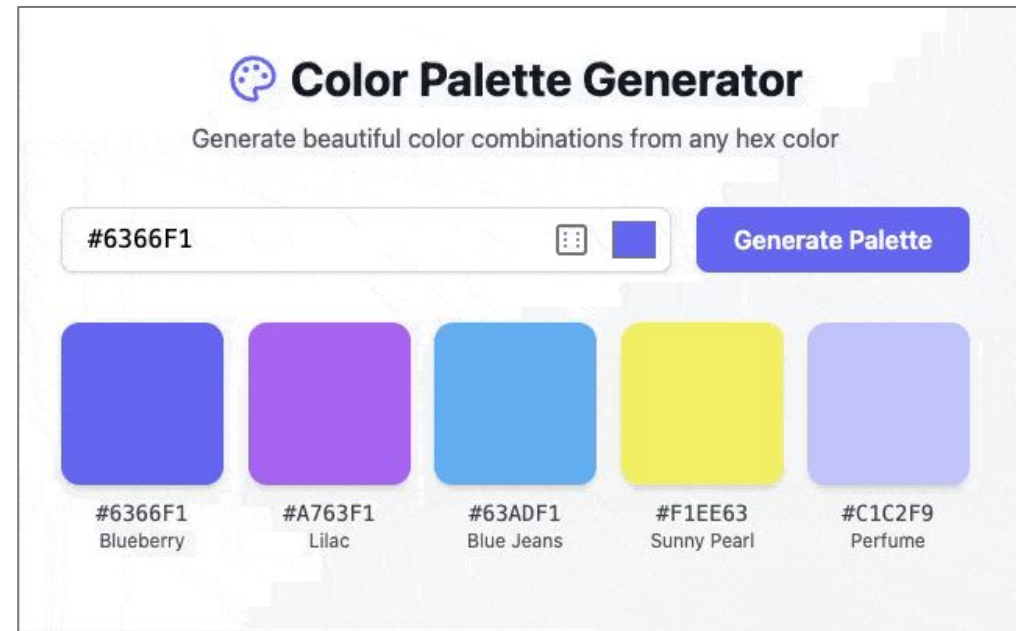
- Allons plus loin ?

```
Call `thecolorapi` to get the name of  
the color and display it below the  
name of each palette result:
```

```
API available at
```

```
https://www.thecolorapi.com/id?hex=0047AB&rgb=0,71,171&hsl=215,100%,34%&cmk=100,58,0,33&format=json and doc at  
https://www.thecolorapi.com/docs
```

```
Also add a random button to select a  
random color
```



- Code mis à jour (1 fichier: App.tsx) - Cas d'erreur géré
- Code écrit : 0 ligne

# Low-Code : rendre la programmation accessible

**Public cible :** Non-développeurs, profils techniques juniors, app simple, ou prototypage.

- ⊕ Editeur simplifié (souvent web)
- ⊕ Suggestions d'IA pour générer des composants ou du code standard
- ⊕ Démarrage ultra-rapide d'une application (front, back, ou bot)
- ⊖ Risques de dépendance à un écosystème propriétaire
- ⊖ Personnalisation avancée parfois limitée

**BOLT**

 **replit**



“ As of Feb 2025, Bolt.new allows you to build iOS and Android apps without writing any code. You can literally go from idea to App Store... ”

Assistant Dev :  
Booster la  
productivité

---



CURSOR

tabnine



# Cursor : un fork de VSCode augmenté

The screenshot displays the Cursor IDE interface. On the left is a file explorer showing a project structure with folders like `src` and `components/color-palette`. The main editor shows the `ColorBox.tsx` file with the following code:

```
1 import React, { FC, useEffect, useState } from "react";
2
3 interface ColorBoxProps {
4   color: string;
5   onClick: (color: string) => void;
6 }
7
8 export const ColorBox: FC<ColorBoxProps> = ({ color, onClick }) => {
9   const [colorName, setColorName] = useState<string>("");
10
11   useEffect(() => {
12     const fetchColorName = async () => {
13       // Remove '#' if present and ensure color is in correct format
14       const hexColor = color.replace("#", "");
15       try {
16         const response = await fetch(
17           `https://www.thecolorapi.com/id?hex=${hexColor}`
18         );
19         const data = await response.json();
20         setColorName(data.name.value);
21       } catch (error) {
22         console.error("Error fetching color name:", error);
23         setColorName("");
24       }
25     };
26
27     fetchColorName();
28   }, [color]);
29
30   return (
31     <div
32       className="color-box"
33       style={{ backgroundColor: color }}
34       onClick={() => onClick(color)}

```

On the right, a chat window is open with the following text:

Call `thecolorapi` to get the name of the color and display it below the name of each palette result:  
API available at <https://www.thecolorapi.com/id?hex=0047AB&rgb=0,71,171&hsl=215,100%,34%&cmky=100,58,0,33&format=json and doc at https://www.thecolorapi.com/docs>

I'll help you modify the ColorBox component to fetch and display the color name using the Color API. We'll need to add state for the color name and make an API call when the component mounts.

Here's the updated code:

```
import React, { FC, useEffect, useState } from "react";
interface ColorBoxProps {
  color: string;
  onClick: (color: string) => void;
}
export const ColorBox: FC<ColorBoxProps> = ({ color, onClick }: ColorBoxProps) => {
  const [colorName, setColorName] = useState<string>("");
  useEffect(() => {
    const fetchColorName = async () => {
      // Remove '#' if present and ensure color is in correct ;
      const hexColor = color.replace("#", "");
      try {
        const response = await fetch(
          `https://www.thecolorapi.com/id?hex=${hexColor}`
        );
        const data = await response.json();
        setColorName(data.name.value);
      } catch (error) {
        console.error("Error fetching color name:", error);
        setColorName("");
      }
    };
    fetchColorName();
  }, [color]);
  return (
    <div
      className="color-box"
      style={{ backgroundColor: color }}
      onClick={() => onClick(color)}

```

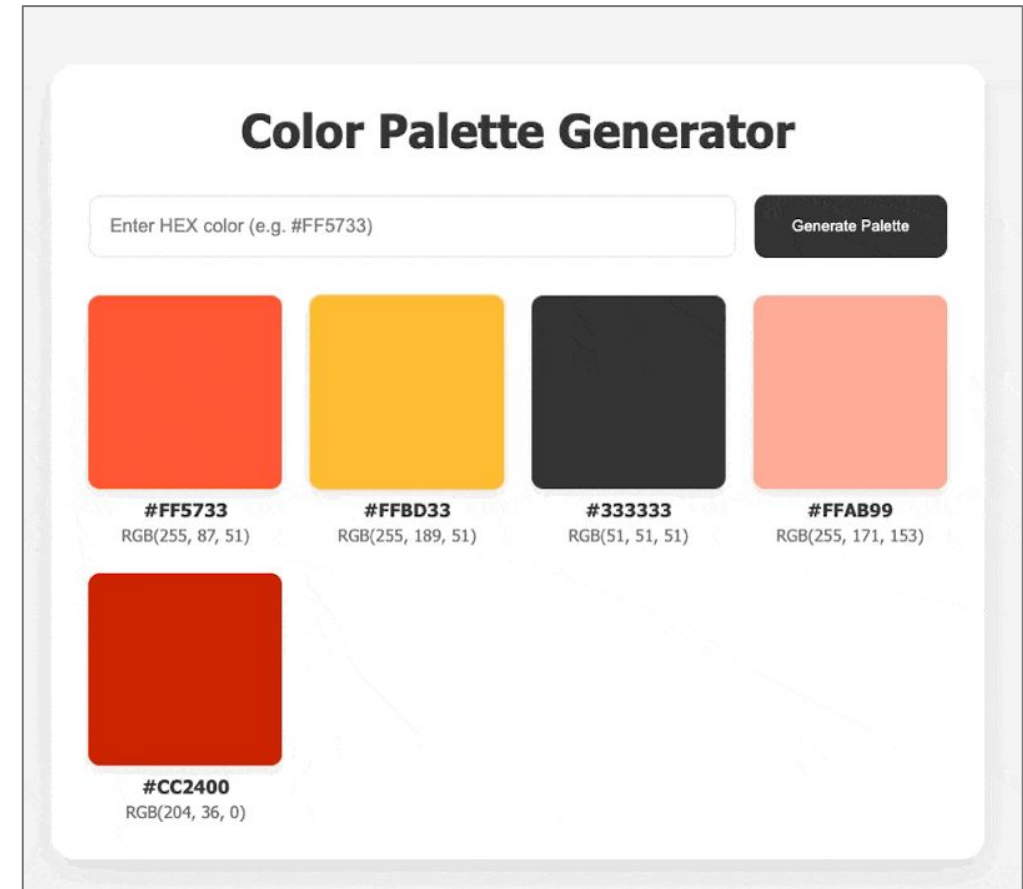
At the bottom, a terminal window shows the command `4:40:36 PM [vite] (client) hmr update /src/components/color-palette/ColorBox.tsx (x2)`.



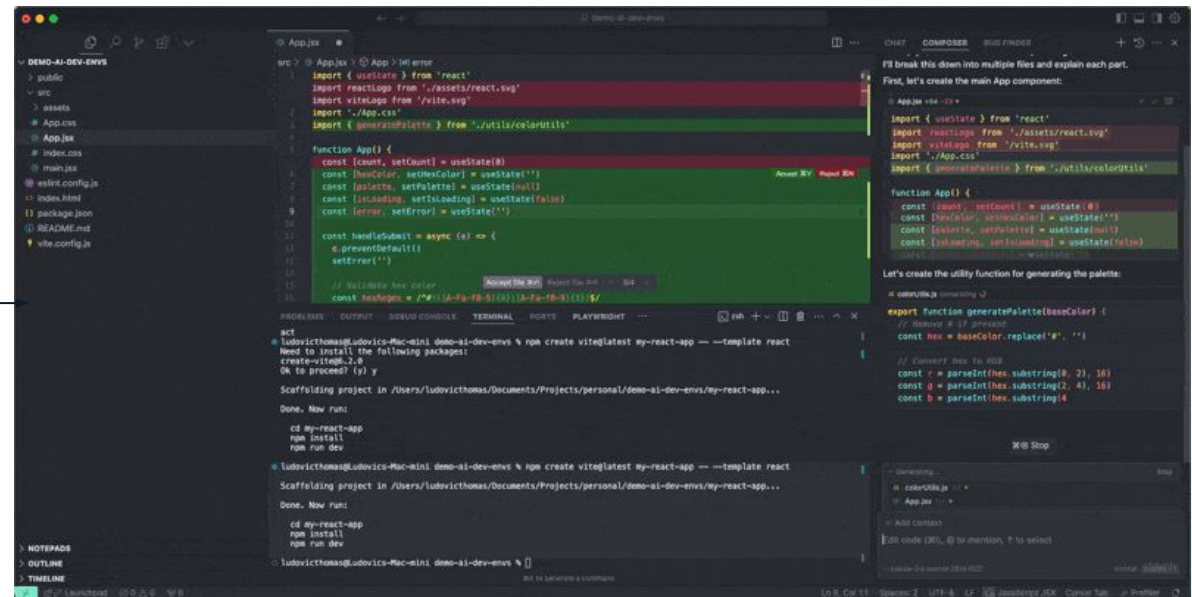
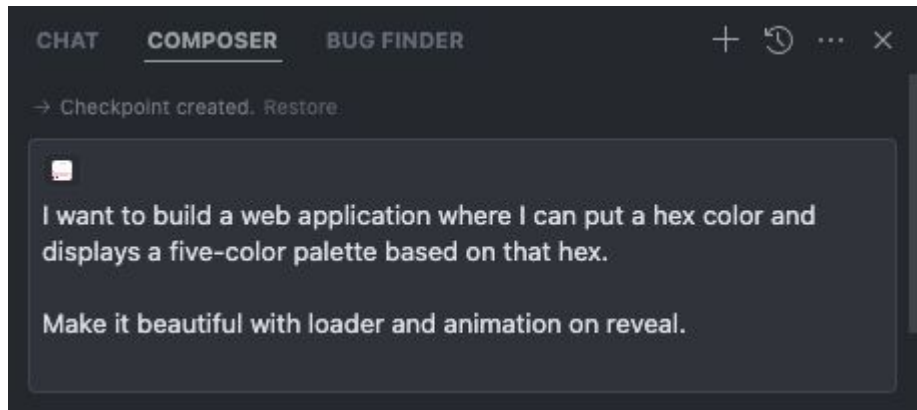


# Cursor : prompt et contexte

- Projet local, pas de déploiement intégré
- Aucun framework utilisé
  - Ouvrir le fichier HTML pour voir le résultat
  - Pas d'icône, rien de fancy, juste ce qui est demandé → Pas de "prompt extension"
- Code généré
  - 3 fichiers (HTML, JS, CSS)
- Code écrit : 0 ligne

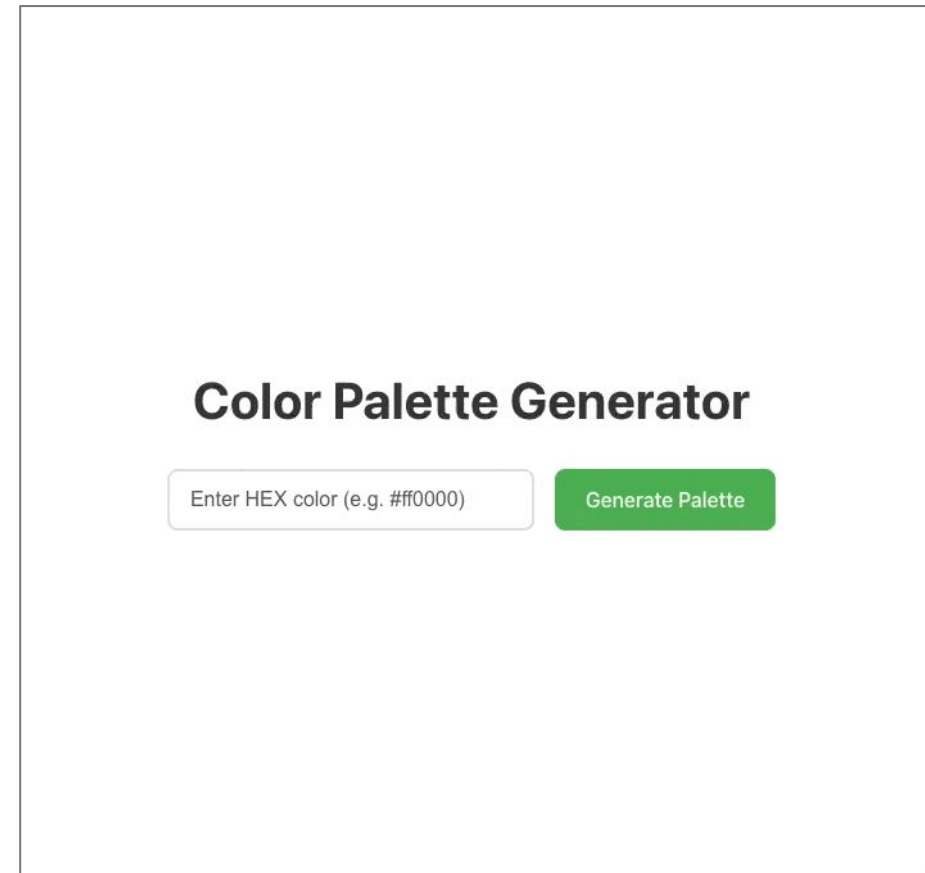
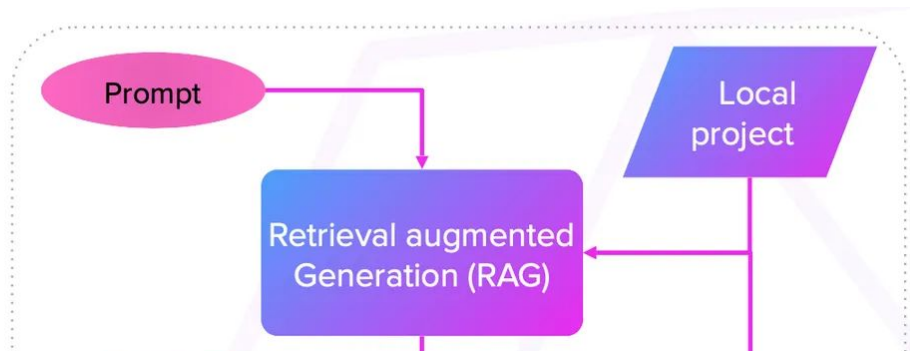


# Cursor : prompt et contexte



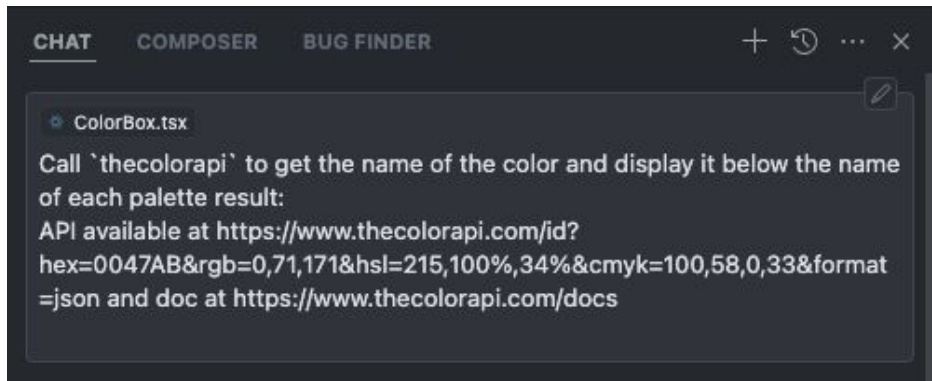
# Cursor : prompt et contexte

- Projet local, pas de déploiement intégré
- Framework local utilisé
  - React
  - Pas d'ajout d'autres dépendances
- Code généré
  - 3 fichiers (App.jsx, colorUtils, CSS)
- Code écrit : 0 ligne



# Cursor : complétion et génération

- Complétion de blocs de code basé sur le commentaire
- Génération de code suite à une demande manuelle à l'assistant



- Génération de tests unitaires

```
// Va fetcher le nom de la couleur
const fetchColorName = async (color: string) => {
  const response = await fetch(`https://www.thecolorapi.com/id?rgb=${color}`);
  const data = await response.json();
  return data.name.value;
}
```

I'll help you modify the ColorBox component to fetch and display the color name using the Color API. We'll need to add state for the color name and make an API call when the component mounts.

Here's the updated code:

```
ColorBox.tsx Apply
import React, { FC, useEffect, useState } from "react";

interface ColorBoxProps {
  color: string;
  onColorClick: (color: string) => void;
}

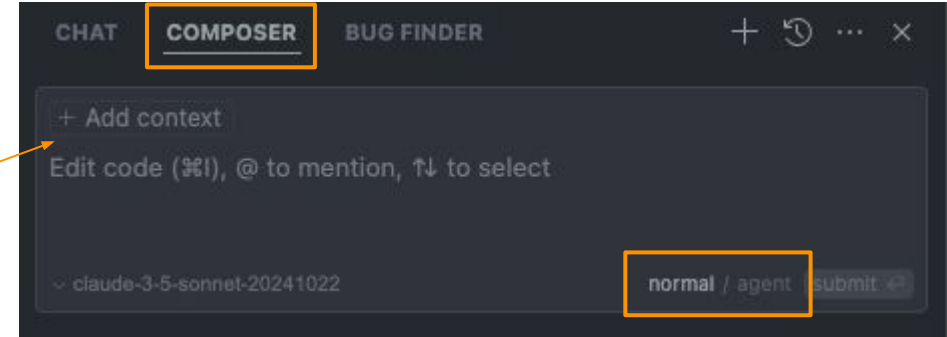
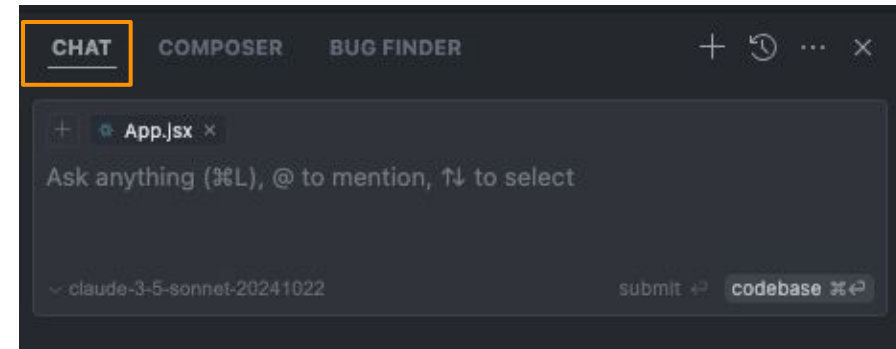
export const ColorBox: FC<ColorBoxProps> = ({ color, onColorClick }) => {
  const [colorName, setColorName] = useState<string>("");

  useEffect(() => {
    const fetchColorName = async () => {
      // Remove '#' if present and ensure color is in correct format
      const hexColor = color.replace("#", "");
      try {
        const response = await fetch(
          `https://www.thecolorapi.com/id?hex=${hexColor}`
        );
        const data = await response.json();
        setColorName(data.name.value);
      } catch (error) {
```



# Cursor : Chat ou Composer ?

- **Chat = Assistant conversationnel**
  - Aucun contexte de base
  - Question libre
- **Composer =**
  - **Normal:** créer, refactoriser avec le projet comme contexte
  - **Agent:** exécute des commandes en local (recherche de fichiers, build, etc.) - Réitère
- **Ajouter du contexte**
  - Fichiers locaux
  - ``@web`` pour aller chercher sur internet
  - ``@git`` pour référencer un commit
  - ``@https://myapi/doc`` pour inclure le site

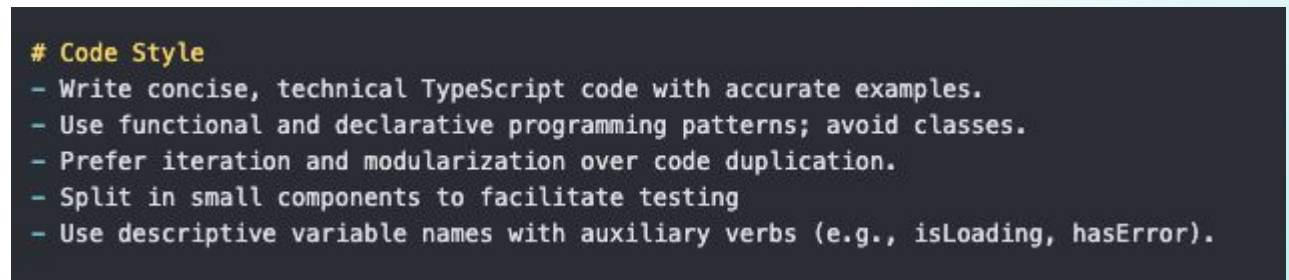
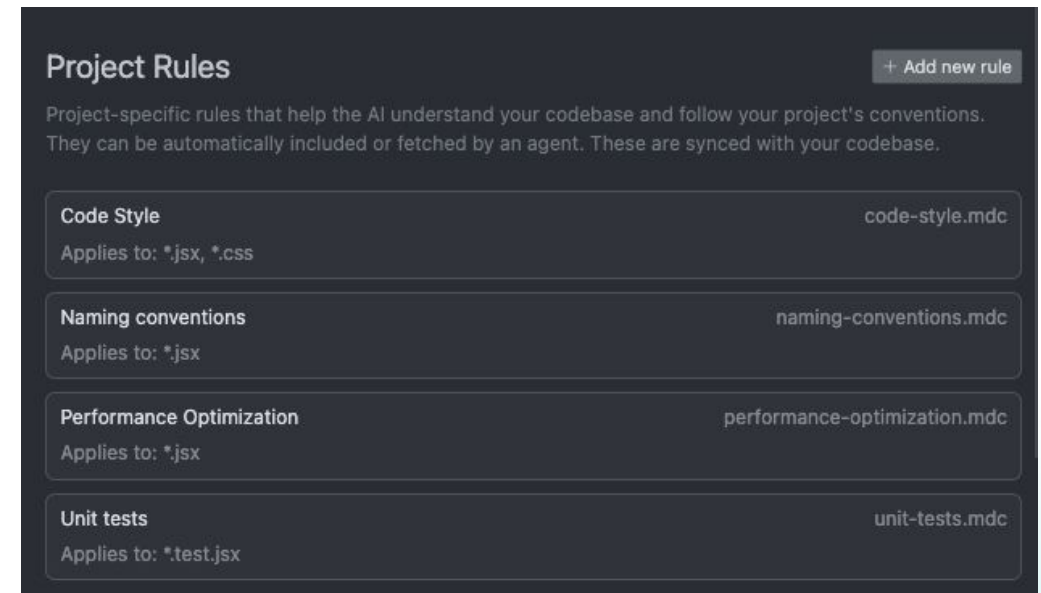
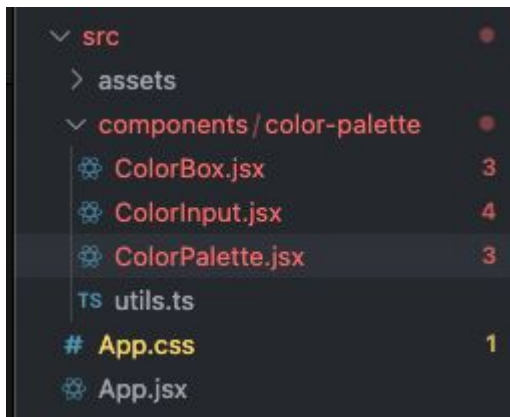


# Cursor : Règles et guidelines ?

## Ajouts des règles

- Générales
- Précises
  - Un fichier par thème
  - Selon les extensions de fichiers
  - Vont être inclus à la demande
  - Cross-références en utilisant `@<file>`

→ Toujours plus de contexte



# Assistant : boost de productivité pour codeurs

**Public cible :** Développeurs juniors ou confirmés souhaitant accélérer l'écriture de code.

- ⊕ Auto-complétion intelligente
- ⊕ Refactoring et suggestions d'optimisation
- ⊕ Génération de tests unitaires et documentation
- ⊕ Intégration IDE existant (VSCode, JetBrains, etc)
- ⊖ Peut engendrer une dépendance excessive
- ⊖ Plus complexe à utiliser



" By 2028, 75% of enterprise software engineers will use AI code assistants up from less than 10% in early 2023. "

Gartner



# Agents IA : Automatisation de tâches

---



# Devin : Un collaborateur IA autonome ?

The image shows a Slack channel interface for '#team-eng' on the left and a terminal window on the right. The Slack channel shows a conversation where Lucy asks Devin to ensure a function has 4 positional arguments. Devin responds that it will check the codebase and update progress. The terminal window shows the 'Set up Next.js repo' process, with Devin's message: 'Absolutely! I'll get started on that right away and keep you updated on my progress.' Below the terminal, there is a code snippet showing a JSX element with an icon prop.

**Slack Channel: #team-eng**

**Lucy 11:55**  
@Devin anywhere generate\_response\_async function is called, can you please ensure that it has 4 positional arguments?

**System 11:56**  
Devin is ready, you can now send messages! To mute Devin, send 'mute'. To unmute, send 'unmute'. Messages starting with '(aside)' will be ignored. You can track the progress [here](#).

**Devin 11:56**  
Certainly, Lucy! I'll go through the codebase and check all instances where the generate\_response\_async function is called. I'll ensure that each call has 4 positional arguments. I'll keep you updated on my progress. Is there anything specific you'd like me to pay attention to while I'm working on this task?

**Terminal: Set up Next.js repo**

**You 5:05 PM**  
Can you start by setting up the Next.js repo, building the project, and running an example?

**Devin 5:05 PM**  
Absolutely! I'll get started on that right away and keep you updated on my progress.

```
ubuntu@ip-10-240-130-118:~$ git clone https://github.com/vercel/next.js.git
Cloning into 'next.js'...
remote: Enumerating objects: 465785, done.
remote: Counting objects: 100% (1887/1887), done.
remote: Compressing objects: 100% (943/943), done.
remote: Total 465785 (delta 1253), reused 1393 (delta 928), pack-reused 463698 (from 1)
Receiving objects: 100% (465785/465785), 1.96 GiB | 20.73 MiB/s, done.
Resolving deltas: 100% (321456/321456), done.
Updating files: 100% (28019/28019), done.
ubuntu@ip-10-240-130-118:~$
```

**Code Snippet:**

```
292 294 href="/knowledge"
293 295 />
294 296 <SidebarAction
295 - icon={<LightbulbFilament className="h-6 w-6 flex-shrink-0" />}
297 + icon={<SealQuestion className="h-6 w-6 flex-shrink-0" />}
```

500\$ / mois

# Agents IA : automatisation multi-étapes

- ⊕ Analyse un repo, boucles de rétroactions, reasoning
- ⊕ Gère des opérations complexes de bout en bout (refactor, upgrade)
- ⊕ Automatisation de tâches fastidieuses
- ⊖ Expérimental
- ⊖ Nécessite des garde-fous, une surveillance
- ⊖ Agent peu réflexif et ne reconnaît pas l'échec (il essaye toujours)



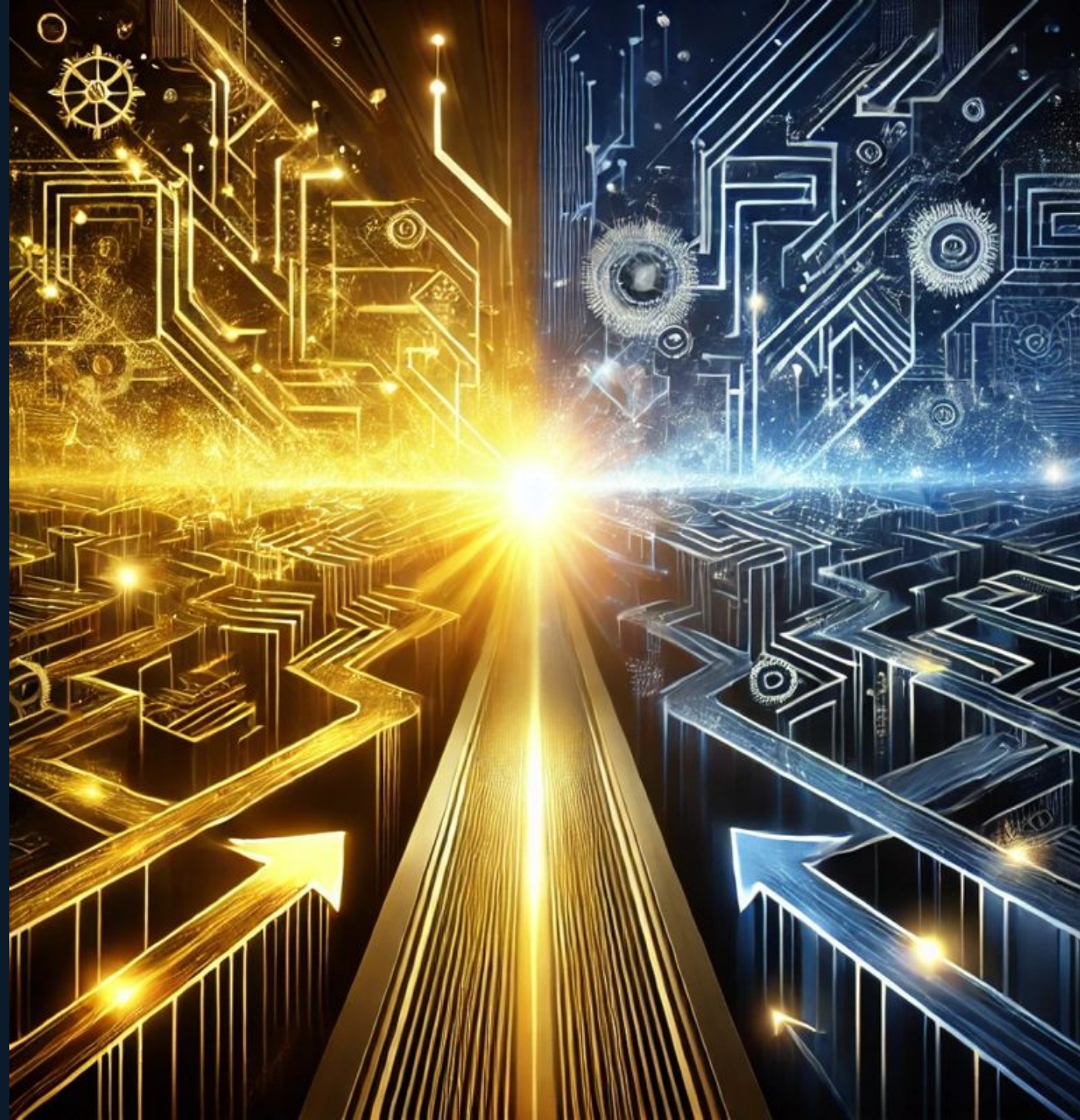


# Agents IA : Au delà de l'écriture de code ?

- **Onboarding agent**
  - setup machine, tour d'horizon des bonnes pratiques, et présentation de l'architecture
- **Code fix agent**
  - de JIRA à la Pull Request
- **Code review agent**
  - utilisation des règles écrites pour vérifier le code avant de soumettre - plus que de l'analyse statique
  - commentaires automatiques sur la Pull Request
- **Test agent**
  - Augmentation de la couverture de code / Recherche d'erreurs
  - Création de scénarios de tests basé sur le code



3 Opportunités  
& défis

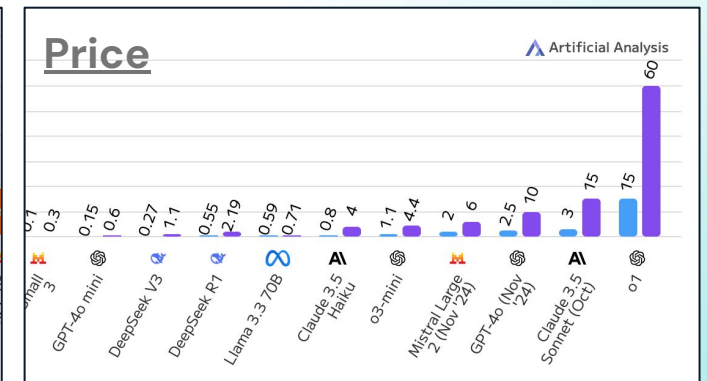
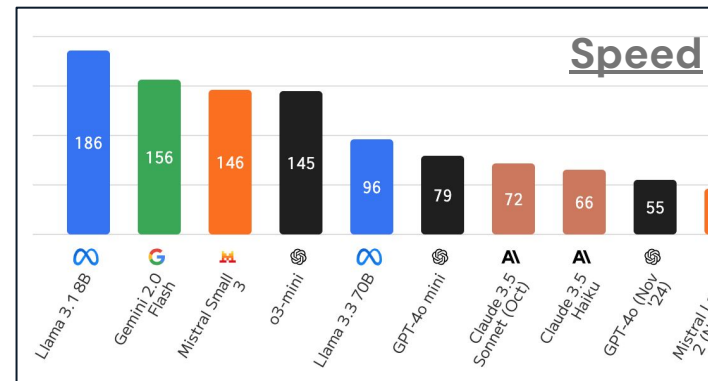
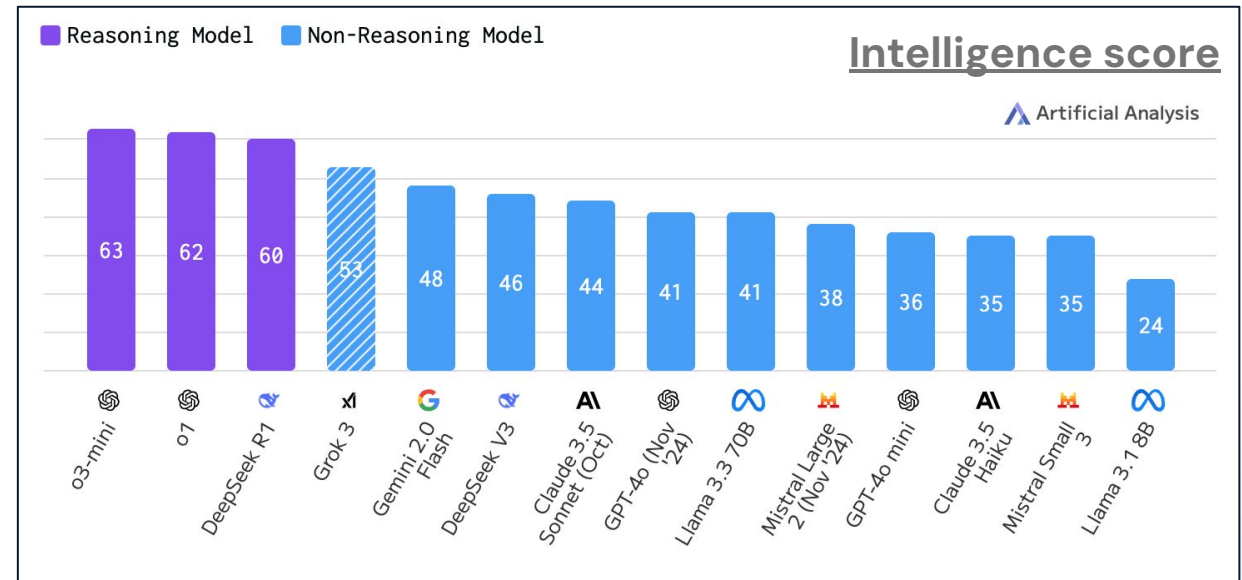


# Opportunités : un modèle pour chaque besoin

## Tous les LLM ne se valent pas !





- Différents objectifs
  - Raisonnement complexe
  - Rapidité & coût réduit
  - Spécialisations
- Choix selon les besoins
  - Refactoring / Architecture
  - Auto-complétion

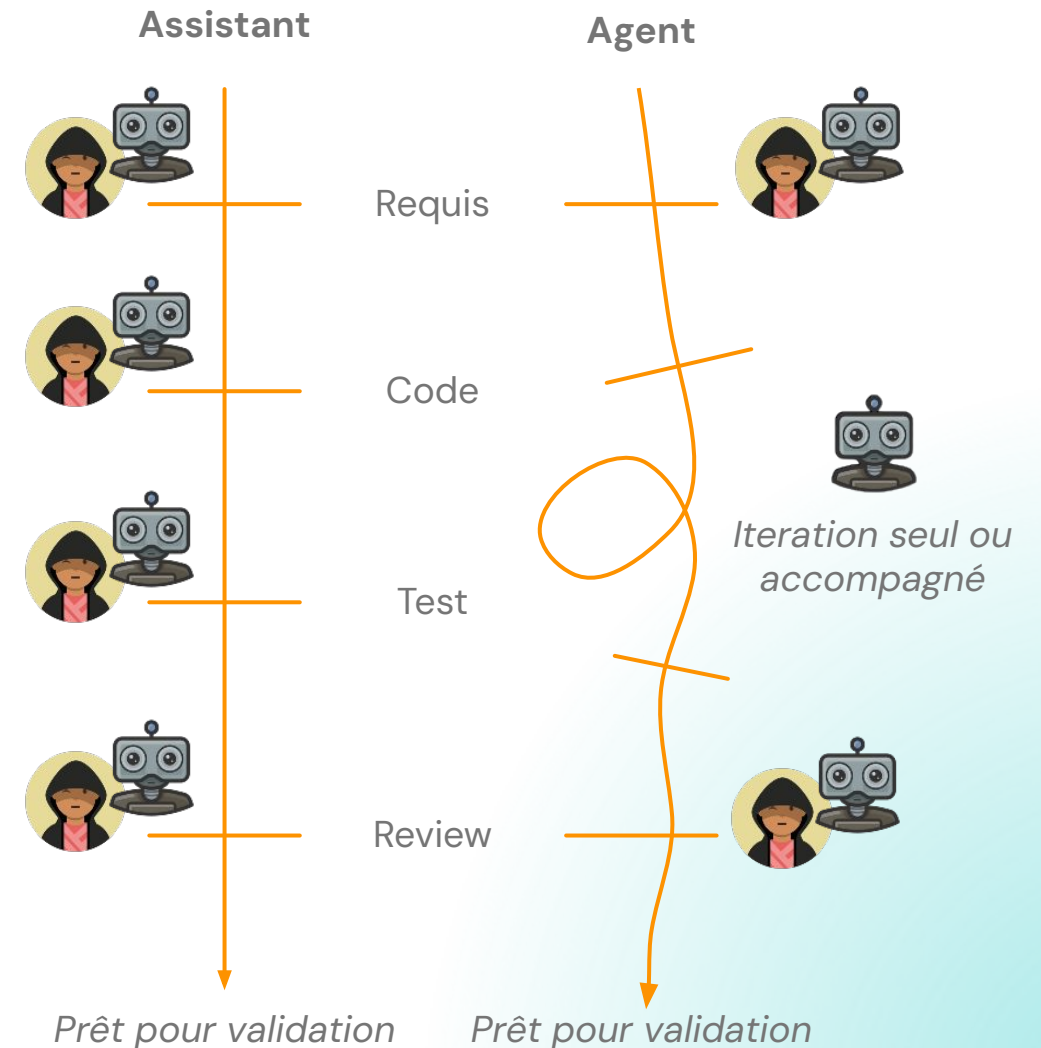
Source: <https://artificialanalysis.ai/>





# Opportunités : bonnes pratiques

-  Validation du code généré
  - Attention aux hallucinations
-  Gardez la main sur la conception
  - Le développeur décide
  - Pas de copier-coller sans comprendre (questionner)
-  Personnaliser avec vos guidelines internes
  - Suggestions alignées
-  Code review humain
  - Sécurité / Failles



# Défis : d'où vient le code, et où va t-il ?

## Origine du code généré ?

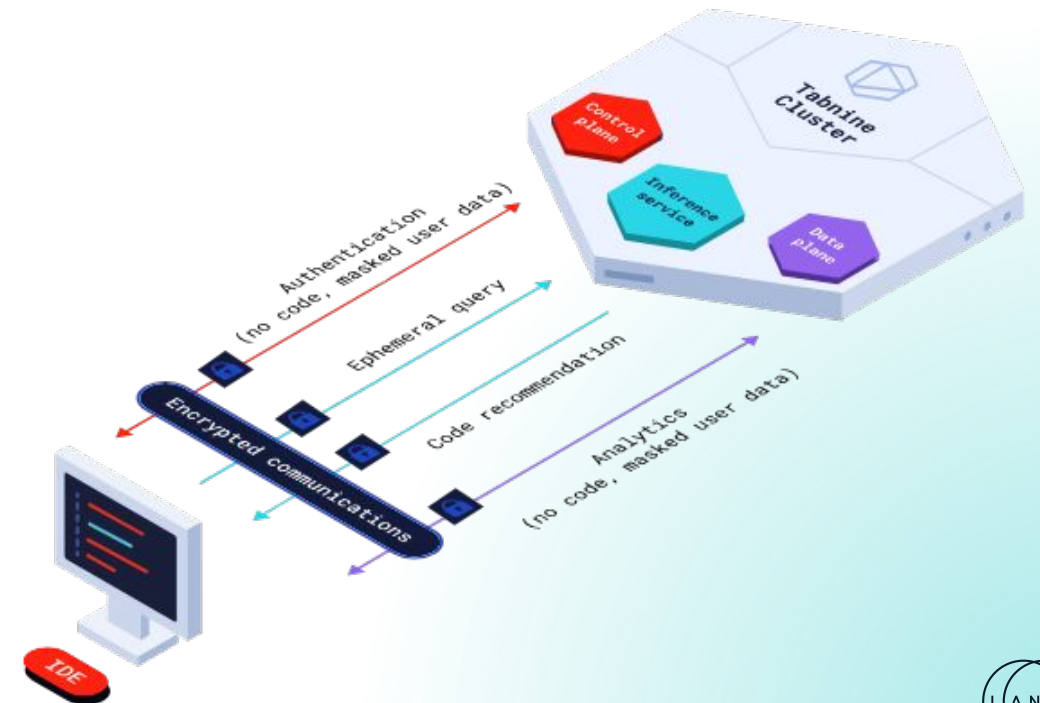
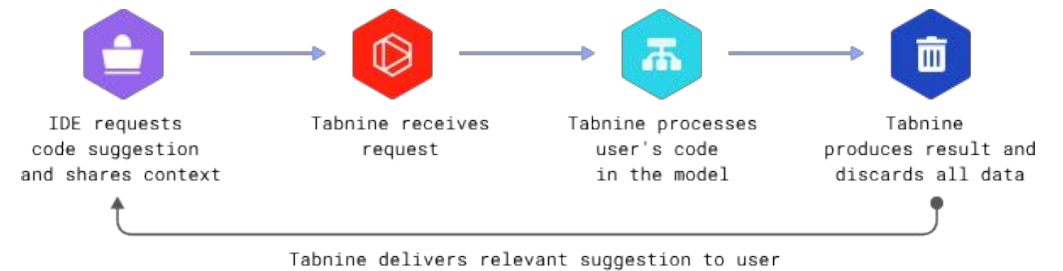
- Modèles entraînés sur des dépôts publics, mais sous quelles licences ?
- Entraînement propriétaire vs. open-source

## Où va notre code quand on utilise l'assistant ?

- Fragments de code envoyés pour complétion
- Ré-entraînement avec le code produit
- Logs et historique pour analyse

→ Dépend de la politique de l'entreprise et de la sensibilité du code

**Privacy-First** : Tabnine Private, Copilot for Business



# Défis : quel avenir pour les développeurs ?

## Évolution du rôle

- Plus de conception et d'architecture
- Collaboration Homme-IA

## Nouvelles compétences

- Prompt Engineering
- Compréhension approfondie du domaine

## Et si je souhaite continuer à coder ?

- Une transition qui va être longue
- L'IA ne remplace pas les développeurs : elle les augmente.
- Adoptez ces outils avec méthode et gardez la main sur la création.





# Merci !

---



Ludovic THOMAS  
Développeur Senior

lthomas@landr.com  
@ludovictthomas

## *Ressources*

- *Blog et sites officiels (Github Copilot, Cursor, Tabnine, Mistral, Anthropic, etc)*
- *Cookbook OpenAI*
- *Podcasts sur l'IA (Underscore\_)*
- *Twitter / X - Comptes officiels compagnies, CTO, Mckay Wrigley)*

