

# **Développer avec GitHub Copilot (Mode Agent)**

**De l'auto-complétion à l'agent autonome**

# Le vrai changement avec Copilot Agent

“

Copilot n'est plus un outil réactif.  
C'est un **acteur du développement**.

”

- Tu exprimes l'intention
- Il planifie le travail
- Il implémente le code

👉 On passe du *prompting* à la **collaboration**.

# Les 4 modes Copilot (vue d'ensemble)

- **Ask** : comprendre (questions, explications)
- **Edit** : modifier du code existant
- **Plan** : réfléchir avant de coder
- **Agent** : exécuter une tâche complète



Tous les modes servent un but :

**produire du code cohérent, pas juste du code qui marche**

# Mode Agent : ce qui change

- Vue globale du projet
- Travail sur plusieurs fichiers
- Autonomie basée sur un plan

👉 Fait pour les vraies features.

# **Le workflow Agent (le cœur du sujet)**

Tu expliques → Copilot planifie → Copilot code

# Workflow en 3 étapes

- 1 Décrire le besoin (toi)** — une spécification simple, en langage humain
- 2 Laisser Copilot réfléchir (Mode Plan)** — plan technique + découpage
- 3 Laisser Copilot coder (Agent)** — implémentation selon le plan

# Étape 1 — Écrire de bonnes instructions

Fichier clé : `.github/copilot-instructions.md`

**Question centrale :** Qu'est-ce que je veux construire, et pourquoi ?

Pas de code.

Pas d'architecture détaillée.

Juste l'intention.

# Contenu recommandé

Dans `.github/copilot-instructions.md` :

- Objectif de l'application
- Comportements clés
- Contraintes importantes

👉 Plus c'est clair ici, moins tu corriges après.



# Exemple simple

```
# Application Todo
```

```
## Objectif
```

```
Créer une application web simple pour gérer des tâches.
```

```
## Comportements clés
```

- Ajouter une tâche
- Marquer une tâche comme complétée
- Supprimer une tâche
- Sauvegarde locale des données

```
## Contraintes
```

- React + TypeScript
- Tailwind CSS
- Pas de dépendances externes



Pas besoin d'écrire du code, ni un plan technique.

# Étape 2 — Mode Plan

Prompt :

“ Analyse ces instructions et propose un plan. ”

Copilot :

- structure le projet
- découpe les composants
- ordonne les étapes

👉 On valide avant de coder.

# Étape 3 — Mode Agent

- Implémentation étape par étape
- Création et modification de fichiers
- Respect du plan

Ton rôle : valider et ajuster.

# Rôles

- **Toi** : vision, décisions, validation
  - **Copilot** : plan, découpage, code
- 👉 Tu restes responsable.

# Règle d'or

“

Plus l'intention est claire,  
meilleur sera le code.

”

# Pièges à éviter

- Spec trop détaillée
- Plan non validé
- Contraintes oubliées

👉 Un agent sans cadre = dette technique.

# Démo — Finance dashboard

- Spec → Plan → Code
- FastAPI + Plotly
- UI chatbot
- Données mock



Montrer le process, pas le détail du code.

# À retenir

- Copilot Agent  $\neq$  auto-complétion
- La spec prime sur le code
- Le plan est indispensable
  - ➡ Meilleures instructions
  - ➡ Meilleur code



**Merci 🙏**

**Questions ?**

Slides : <https://r-drumond-cegid.github.io/copilot-example/>

Repo : <https://github.com/r-drumond-cegid/copilot-example>

# Liens utiles

- Demande d'accès à GitHub Copilot chez Cegid :  
<https://devsecopscegid.atlassian.net/servicedesk/customer/portal/1/user/login?destination=portal%2F1%2Fgroup%2F4%2Fcreate%2F43>
- Cegid Design System : <https://cds-website.azurewebsites.net/guidelines/installation>

# Ressources et exemples

- Personnalisation de Copilot dans VS Code :  
<https://code.visualstudio.com/docs/copilot/customization/overview>
- awesome-copilot : [github.com/github/awesome-copilot](https://github.com/github/awesome-copilot)

