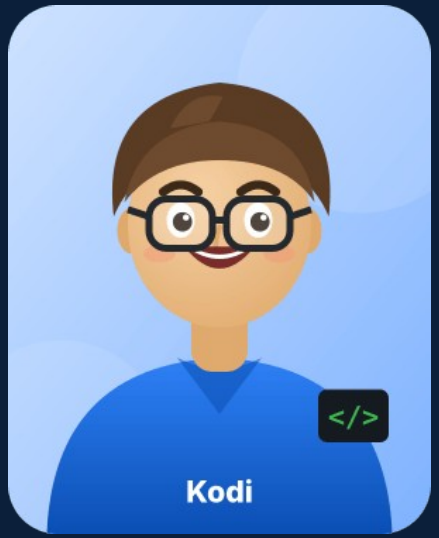
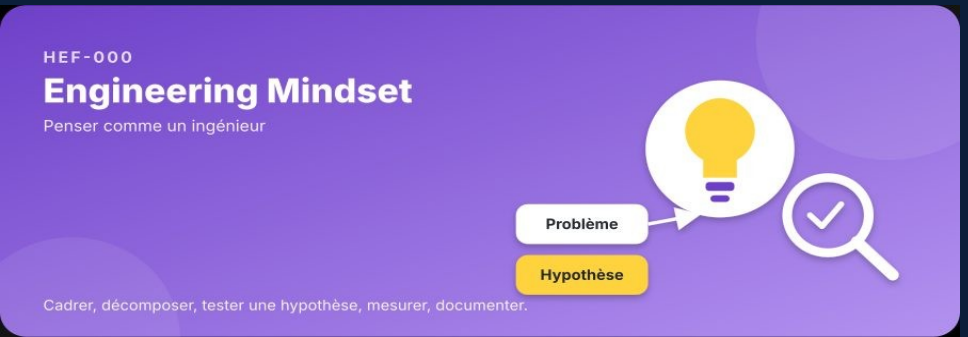
HEF-000 · MISSION 09 / 9

Travailler avec l'IA sans lui laisser le volant

L'IA est un excellent assistant et un mauvais décideur. À toi de rester responsable.



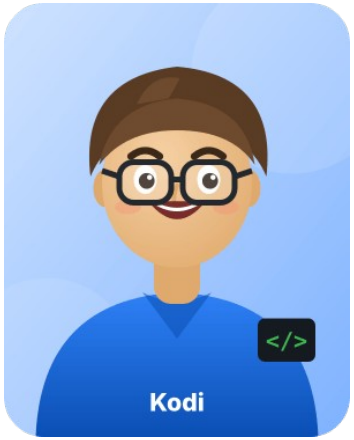


Badge : Pilote augmenté·e

75 min · débutant

L'histoire

Kodi demande à un assistant IA d'écrire une fonction. Le code a l'air parfait. Il le colle, le déploie. Le lendemain, un client reçoit le rapport d'un autre client.



> *« L'IA t'a donné du code plausible, dit Sam. Pas du code **vérifié**. La différence, c'est toi. »*

Dans cette mission, tu apprends à utiliser l'IA comme un outil puissant, avec méthode et responsabilité.

Ta mission

À la fin, je serai capable de…

1. décrire ce que l'IA fait bien et mal
2. rédiger une demande claire (contexte, tâche, contraintes, format)
3. vérifier une réponse de l'IA (tests, sources, relecture)
4. reconnaître une « hallucination »
5. décider ce qu'on peut confier à l'IA et ce qu'on garde

Défi d'échauffement

Défi d'échauffement

Tu demandes à un ami très cultivé mais qui répond toujours avec assurance, même quand il ne sait pas : « Quelle est la capitale du Bhoutan ? » Il répond sans hésiter. Comment sais-tu s'il a raison ? Que fais-tu si tu ne peux pas vérifier ? Écris ta règle personnelle.

Le concept

Ce que fait l'IA générative

Un modèle de langage produit un texte **plausible** à partir de ce qu'il a appris. Il peut :

- rédiger, reformuler, résumer ;
- proposer du code, expliquer une erreur ;
- brainstormer des idées, des tests, des cas limites.

Il peut aussi :

- **inventer** (*hallucination*) : fonctions qui n'existent pas, sources fictives ;
- être **sûr de lui** alors qu'il se trompe ;
- ignorer le contexte réel de ton projet ;
- reproduire des **failles** de sécurité ou des biais.

Le concept

Une bonne demande : CTCF

Lettre	Élément	Exemple
C	Contexte	« Je développe une API en Python pour un site de cours »
T	Tâche	« Écris une fonction qui valide un e-mail »
C	Contraintes	« Sans bibliothèque externe, avec tests »
F	Format	« Réponds avec le code, puis une liste des cas limites »

Vérifier : la règle des trois contrôles

1. **Comprendre** : peux-tu expliquer chaque ligne ? Sinon, ne l'utilise pas.
2. **Tester** : exécute, avec des cas normaux, limites et invalides.
3. **Recouper** : documentation officielle, deuxième source, collègue.

Le concept

Vérifier : la règle des trois contrôles (suite)

```
# Exemple : l'IA propose une fonction. Tu la testes avant de lui faire confiance.
def est_email_valide(texte):
    return "@" in texte and "." in texte.split("@")[-1]

tests_attendus = {
    "ada@exemple.fr": True,
    "ada@": False,
    "@exemple.fr": False,
    "ada@exemple": False,
    "ada@@exemple.fr": False,
}
for entree, attendu in tests_attendus.items():
    resultat = est_email_valide(entree)
    print(entree, resultat, "OK" if resultat == attendu else "ÉCHEC")
```

Les cas qui échouent montrent exactement ce que le code proposé ne couvre pas.

Le concept

Ce qu'on confie, ce qu'on garde

On peut déléguer	On garde
Premier jet de code ou de texte	La décision finale
Idées de tests et de cas limites	La vérification
Explication d'une erreur	La compréhension
Reformulation, traduction	Le jugement sur la qualité

Confidentialité et sécurité

Ne colle jamais dans un outil externe des **mots de passe, clés d'accès, données personnelles ou secrets d'entreprise**. Vérifie les règles de ton organisation.

Le concept

Responsabilité

***Le code que tu livres est le tien**, qu'une IA l'ait écrit ou non. En cas de problème, « c'est l'IA » n'est pas une explication.*

Exemple pas à pas

Cas : tu demandes à une IA une fonction qui calcule la moyenne des notes.

Demande CTCF : « Je développe un outil de suivi de cours en Python (C). Écris `moyenne(notes)` (T) qui renvoie `None` si la liste est vide et lève une erreur si une note est hors de 0..20 (C). Donne le code puis 5 cas de test (F). »

Réponse plausible de l'IA :

```
def moyenne(notes):
    if not notes:
        return None
    for n in notes:
        if n < 0 or n > 20:
            raise ValueError("note hors limites")
    return sum(notes) / len(notes)
```

Exemple pas à pas (suite)

Tes trois contrôles :

1. *Comprendre* : chaque ligne est claire. Question : et si `notes` contient du texte ?
2. *Tester* :

```
def moyenne(notes):
    if not notes:
        return None
    for n in notes:
        if n < 0 or n > 20:
            raise ValueError("note hors limites")
    return sum(notes) / len(notes)

assert moyenne([]) is None
assert moyenne([10, 20]) == 15
try:
    moyenne([25])
```

Exemple pas à pas (suite)

```
    raise AssertionError("aurait dû lever une erreur")
except ValueError:
    pass
try:
    moyenne(["a"])
except TypeError:
    print("TypeError non gérée proprement : à améliorer")
```

3. *Recouper* : on décide si une note peut être décimale, si 0 est valide, etc.

Résultat : tu as trouvé une lacune (types invalides) que l'IA n'avait pas anticipée. Tu corriges ou tu redemandes avec la contrainte.

À toi, avec aide

Voici une demande vague : *« Écris du code pour trier mes données. »*

1. Réécris-la avec la méthode CTCF pour un cas concret (tu choisis les données).
2. Liste trois informations que l'IA aurait dû deviner avec la demande vague.

À toi, avec aide · réponse

Indices progressifs

Indice 1

Précise le langage, le type de données et le critère de tri.

Indice 2

Précise les contraintes : stabilité, taille, bibliothèques autorisées.

Indice 3

Demande aussi des tests.

Réponse

Exemple : *« (C) Je développe un script Python pour HashBox qui traite une liste de 5 000 membres, chaque membre étant un dictionnaire avec `nom` et `missions`. (T) Écris une fonction qui trie les membres par nombre de missions décroissant, puis par nom. (C) Sans bibliothèque externe, ne modifie pas la liste d'origine. (F) Donne le code, puis 4 tests dont un cas de liste vide. »*
Informations à deviner dans la demande vague : le langage, la structure des données, le critère de tri. Une demande plus précise réduit les hallucinations et le travail de vérification.

À toi, en solo

Audit d'une réponse d'IA. Voici un extrait d'explication produit par une IA :

> *« En Python, la fonction list.sortKey(lambda x: x[1]) trie une liste selon le second élément. C'est la méthode la plus rapide, en O(n). »*

1. Repère au moins trois problèmes (fonction inexistante, complexité fausse, etc.).
2. Vérifie dans la documentation Python (ou par un test) la bonne façon de faire.
3. Écris la correction avec un test.

À toi, en solo · réponse

Pistes de correction

Problèmes : la méthode `sortKey` n'existe pas (la bonne est `liste.sort(key=…)` ou `sorted(liste, key=…)`) ; la complexité d'un tri par comparaison est O(n log n), pas O(n) ; « la plus rapide » est une affirmation sans comparaison.

```
liste = [("a", 3), ("b", 1), ("c", 2)]
assert sorted(liste, key=lambda x: x[1]) == [("b", 1), ("c", 2), ("a", 3)]
print("ok")
```

Pièges classiques

Piège	Pourquoi c'est un problème	Comment l'éviter
Copier-coller sans comprendre	Failles et bugs cachés	Expliquer chaque ligne avant de l'utiliser
Croire une réponse assurée	Hallucinations	Vérifier avec tests et sources
Partager des données sensibles	Fuite d'informations	Anonymiser ou ne rien envoyer
Demande vague	Réponse générique	CTCF : contexte, tâche, contraintes, format

Mini-boss

Mini-boss

Charte personnelle d'usage de l'IA (1 page).

1. Trois usages que tu acceptes et pourquoi.
2. Trois usages que tu refuses et pourquoi.
3. Ta règle de vérification (les trois contrôles) appliquée à un cas concret.
4. Ce que tu ne partageras jamais avec un outil externe.
5. Comment tu citeras ou signaleras l'usage de l'IA dans un travail.
6. Un exemple d'erreur d'IA que tu as rencontrée (réelle ou imaginaire) et ce qu'elle t'apprend.

Mini-boss · réponse

Pistes de correction

Une bonne charte est personnelle, concrète (exemples) et assume la responsabilité : « je ne livre rien que je ne puisse expliquer ». Elle distingue l'aide à l'apprentissage (expliquer, proposer des exercices) de la délégation qui court-circuite l'apprentissage (faire à ma place un exercice destiné à m'entraîner).

Quiz éclair

Quiz éclair

1. Une « hallucination » d'IA est :

A. un bug du clavier
B. une affirmation inventée présentée avec assurance
C. une réponse trop courte

2. Avant d'utiliser du code généré, on :

A. le copie directement
B. le garde secret
C. le comprend, le teste et le recoupe

3. Qui est responsable du code livré ?

A. la personne qui le livre

Quiz éclair

Quiz éclair (suite)

B. l'outil d'IA
C. personne

4. 🔁 *Révision mission 08* : Pour retenir, la technique la plus efficace est :

A. la relecture
B. le rappel actif
C. le surlignage

5. 🔁 *Révision mission 07* : Une mesure bien faite répond à :

A. seulement combien
B. seulement quand
C. quoi, comment, sur quoi, par rapport à quoi

Quiz éclair · réponse

Réponses

1. **B**.
2. **C**.
3. **A**.
4. **B**.
5. **C**.

Je réfléchis

1. Dans quel cas l'IA t'a-t-elle fait gagner du temps, et dans quel cas t'a-t-elle induit en erreur ?
2. Quelle règle personnelle vas-tu appliquer dès aujourd'hui ?

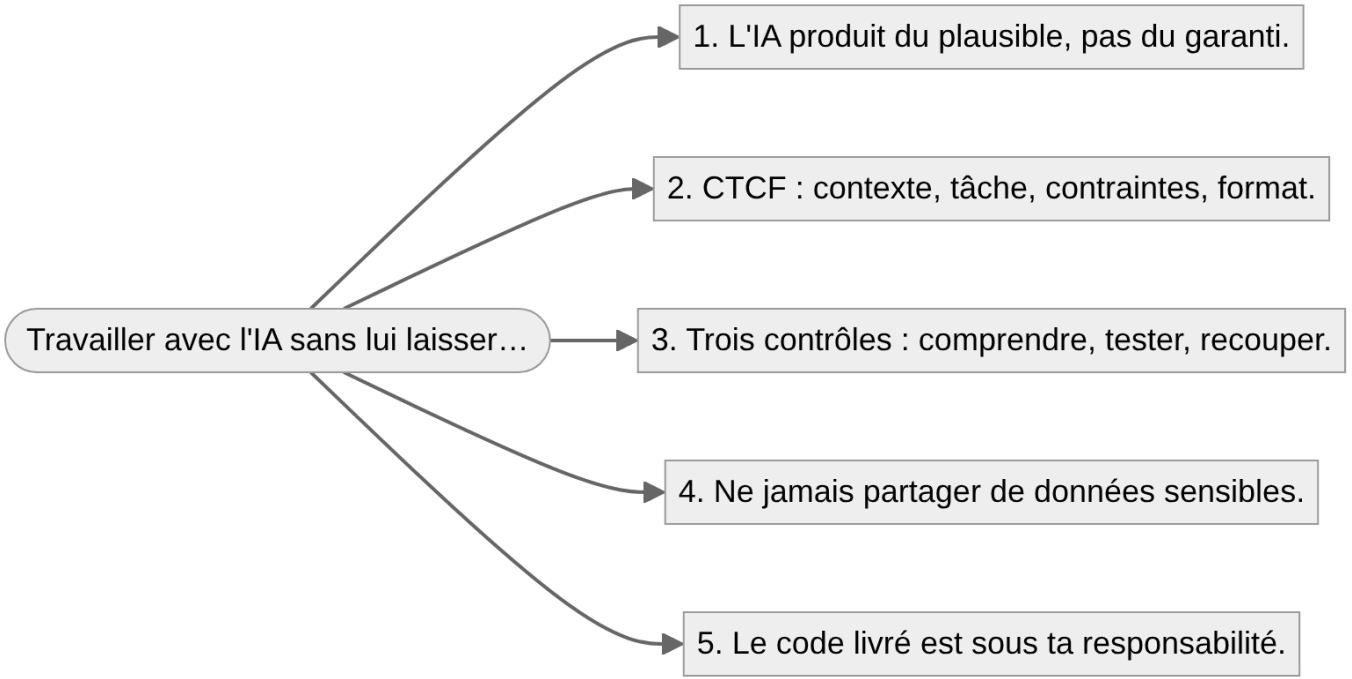
À retenir

À retenir

1. L'IA produit du plausible, pas du garanti.
2. CTCF : contexte, tâche, contraintes, format.
3. Trois contrôles : comprendre, tester, recouper.
4. Ne jamais partager de données sensibles.
5. Le code livré est sous ta responsabilité.

À retenir

Le schéma de la mission



Félicitations

Félicitations

Tu as terminé **HEF-000 — Mentalité d'ingénieur**. Tu sais cadrer, décomposer, raisonner, déboguer, documenter, prouver, apprendre et travailler avec l'IA sans perdre ton esprit critique.
→ Prochain module : **HEF-010 — Algorithmes**

MISSION ACCOMPLIE

+120 XP

Badge débloqué : Pilote augmenté·e

Fin du module HEF-000. Bravo !

