Architecture d'un projet de développement assisté par IA

1 Utilisateur / Développeur



- Définit le besoin
- Pose ses questions
- Valide les propositions
- Contrôle le résultat

Je décris mon besoin
ou je demande une évolution.
L'agent IA m'accompagne
du début à la fin.

Demande

Résultat

2 Documents du projet (guide et mémoire)

PRD.md
- Ce que le produit doit faire
- Besoins
- Règles métier
- Critères d'acceptation

AGENTS.md
- Comment travailler
- Consignes
- Contraintes
- Qualité / Tests
- Utilisation des outils
- Bonnes pratiques

SUIVI.md
- Avancement
- Décisions
- Anomalies
- Tests
- Résultats

Agent IA principal (CodeX)



- Analyse le besoin
- Planifie le travail
- Appelle des agents et des skills
- Utilise les outils via MCP
- Développe ou modifie le code
- Lance les tests
- Rend compte

3 Découpage fonctionnel

roadmap/
- EPIC (grandes fonctionnalités)
- User Stories (tâches détaillées)
- Priorités
- Suivi de livraison

- EPIC 1 ...
- EPIC 2 ...
- EPIC 3 ...

4 Code source et versions

Git (GitHub ou local)
- Historique des versions
- Branches
- Collaboration
- Retour à un état précédent

main
feature/xxx
feature/yyy

Sous-agents spécialisés (appelés par l'agent principal)



PO / Analyste
- Prépare les EPIC
- Rédige les User Stories



Développeur
- Développe
- Intègre
- Corrige



Testeur
- Vérifie
- Exécute les tests
- Signale les anomalies



Documentateur
- Met à jour le README
- Rédige la doc

Appel d'autres agents (si nécessaire)



Agent Frontend | Agent Backend | Agent Sécurité | Agent DevOps | ...

Skills (méthodes réutilisables)



- Génération de code
- Mise à jour README
- Contrôle qualité
- Création de tests
- Refactorisation
- Analyse de code
- Aide à la documentation
- ...

5 MCP (Connecteur d'outils externes)

Le MCP permet à l'agent d'accéder à des outils externes comme un tuyau d'aspirateur. Il récupère des informations, des fichiers ou des services, mais n'effectue pas le travail à la place de l'agent.



Outils et sources externes
- Google Stitch (maquettes UI)
- Sites web (documentation)
- APIs (Services externes)
- Bases de données (informations)
- Fichiers (ressources locales)

6 Environnement d'exécution (si nécessaire)



Docker
- Environnement isolé et reproductible
- Services (application, API, base)
- Même configuration partout
- Utile pour des projets plus complexes

7 Environnement de développement



VS Code
- Édition du code
- Intégration avec Codex
- Exécution des tests
- Outils Git

8 Application finale



- Code fonctionnel
- Testé
- Documenté
- Prêt à être utilisé

Flux de travail type

Expression du besoin → Analyse et planification (par l'agent) → Appel d'agents et de skills → Développement et tests → Livraison du résultat

Itérations si nécessaire

Structure d'un projet (exemple)

```
MonProjet/
├── PRD.md
├── AGENTS.md
├── README.md
├── SUIVI.md
├── roadmap/          (EPIC et User Stories)
├── .agents/          (configuration des agents)
├── skills/           (procédures réutilisables)
├── src/              (code applicatif)
├── tests/            (tests automatisés)
├── docker/           (configuration Docker)
├── compose.yaml      (si plusieurs services)
└── .git/             (historique Git)
```