Software Crew AI

Méthode : Kanban + MoSCoW **WIP :** 6 TODO / 2 IN PROGRESS / 2 TO TEST (max) **Livraison :** Continue **Outil :** Notion

USER STORIES

Artefact Agile décrivant une fonctionnalité du point de vue de l'utilisateur final, structuré au format "En tant que… Je veux… Afin de… " avec critères d'acceptation testables.

Sarah DEBBAGHI | Elodie TURLIER | Camille VIRASAK

WORKFLOW DE GESTION DES USER STORIES

1 Création → 2 Catégorisation → 3 Priorisation → 4 Validation

En début de projet, une liste de tickets a été créée en relation avec les **besoins prioritaires de nos personas** (externes et internes).
Puis, au fur et à mesure, cette liste a été complétée et enrichie.
Les User Stories sont **automatiquement validées** quand la DoD est complète.

Composition d'une User Story

- Titre
- Priorité (MoSCoW)
- Catégorisation (Epic)
- Statut (colonne Kanban)
- Description : "En tant que… je veux… afin de…"
- Critères d'acceptation : "Étant donné… quand… alors…"

CATÉGORISATION DES USER STORIES (COLONNES KANBAN)

BACKLOG	TO DO	IN PROGRESS	TO TEST	DONE
US **incomplète** (non conforme DoR)	US **conforme DoR** prête à démarrer	US **assignée** en cours de traitement	US en cours de **validation**	US **conforme DoD** livrée
13	0	3	1	38

Période du POC

10 novembre — 14 décembre 2025

3 semaines de développement intensif sur le POC Prompt Engineering + RAG

Méthode

Kanban + MoSCoW

DoD pour définir quand est ce qu'un ticket passe en TODO
DoR pour définir quand est ce qu'un ticket passe en DONE

Definition of Ready (DoR)

- ✓ Avoir un titre Clair
- ✓ Relié à une épique
- ✓ Description respectant le format : *"En tant que… je veux… afin de…"*
- ✓ Critères d'acceptances GERKIN métier ou techniques
- ✓ Dépendances identifiées
- ✓ Priorisation effectuée
- ✓ Équipe a compris + questions posées
- ✓ Estimation faite
- ✓ Livrable clair

Definition of Done (DoD)

- ✓ Critères d'acceptation validés
- ✓ Fichier/URL disponible sur ticket
- ✓ Documentation à jour
- ✓ Objectif atteint (bénéfice vérifié)
- ✓ Livrable présenté et validé

Priorisation MoSCow

MUST	Indispensable au POC
SHOULD	Important mais contournable
COULD	Bonus si temps dispo
WON'T	Hors scope POC

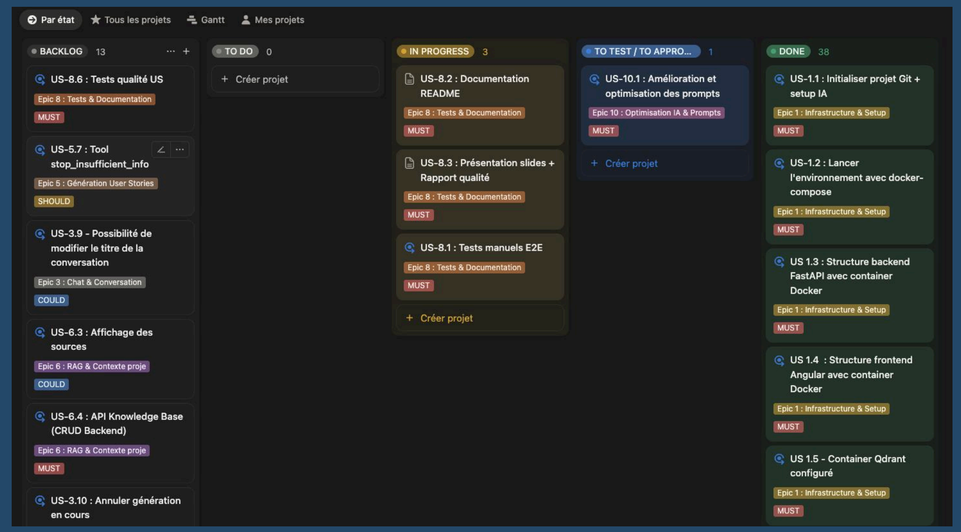
Software Crew AI

USER STORIES

Artefact Agile décrivant une fonctionnalité du point de vue de l'utilisateur final, structuré au format "En tant que… Je veux… Afin de… " avec critères d'acceptation testables.

Sarah DEBBAGHI | Elodie TURLIER | Camille VIRASAK

KANBAN BACKLOG



PROGRESSION PAR EPIC

EPIC	PROGRESSION	PRIORITÉ
1. Infrastructure	8/8	MUST
2. Intégration LLM	1/1	MUST
3. Chat & Conversation	6/12	SHOULD
4. Upload & Parsing	3/4	SHOULD
5. Agent IA (LangGraph)	6/7	MUST
6. RAG & Contexte	2/4	SHOULD
7. Frontend Angular	8/8	MUST
8. Tests & Documentation	3/7	SHOULD
9. Sécurité	0/2	WON'T

Software Crew AI

USER STORIES

Artefact Agile décrivant une fonctionnalité du point de vue de l'utilisateur final, structuré au format "En tant que... Je veux... Afin de..." avec critères d'acceptation testables.

Sarah DEBBAGHI | Elodie TURLIER | Camille VIRASAK

EXEMPLES DE USER STORIES

US-7.1 Structure chat (maquette Figma)

03/12 → 07/12 | Elodie T. | MUST | DONE | 1h

DESCRIPTION

En tant que Product owner
Je veux une interface de chat moderne et épurée
Afin de pouvoir interagir confortablement avec l'assistant IA.

CRITÈRES D'ACCEPTATION (GHERKIN)

Feature: Structure chat En tant que utilisateur Je veux une interface de chat moderne Afin d'interagir confortablement
Background: Given j'ouvre l'application sur http://localhost:4200
Scenario: Layout principal When la page charge Then je vois une sidebar à gauche (280px) And je vois une zone de contenu principale centrée And je vois un header avec le logo "Software Crew AI"
Scenario: Zone de chat When je suis sur une conversation Then je vois la liste des messages (scrollable) And je vois une zone de saisie en bas And la zone de contenu est centrée (max-width 900px)
Scenario: Design system When je regarde l'interface Then les couleurs suivent la palette définie (Cello, Calypso, Tulip Tree) Then la typographie utilise Inter (body) et League Spartan (headings) And les espacements suivent la grille de 4px
Scenario: Responsive desktop Given ma fenêtre fait 1920×1080 When je regarde l'interface Then la sidebar est visible And le chat est confortablement lisible
Scenario: Messages stylisés When je vois des messages Then les messages utilisateur ont un fond San Juan And les messages assistant ont un fond Calypso And les timestamps sont en texte muted

DÉPENDANCES

DEP: US-1.4 Frontend container

EPIC

Epic 7 — Frontend Angular

US-7.3 Affichage notifications chat

03/12 → 07/12 | Elodie T. | MUST | DONE | 1h

DESCRIPTION

En tant que Product owner
Je veux voir des notifications pour les événements système (erreurs, succès, warnings)
Afin de comprendre ce qui se passe dans l'application.

CRITÈRES D'ACCEPTATION (GHERKIN)

Feature: Affichage notifications En tant qu'utilisateur Je veux voir des notifications système Afin de comprendre ce qui se passe
Background: Given l'application est chargée And le ToastService est disponible
Scenario: Toast de succès When une action réussit (ex: upload fichier) Then un toast vert apparaît en haut à droite And il contient le message de succès And il disparaît après 3 secondes
Scenario: Toast d'erreur technique When une erreur technique survient (ex: timeout) Then un toast rouge apparaît And il contient le message d'erreur And il disparaît après 10 secondes (ou clic)
Scenario: Toast de warning When un warning est émis (ex: fichier volumineux) Then un toast orange apparaît And il disparaît après 5 secondes
Scenario: Toast d'information When une info est affichée Then un toast bleu apparaît And il disparaît après 3 secondes
Scenario: Fermeture manuelle Given un toast est affiché When je clique sur le toast ou la croix Then le toast disparaît immédiatement
Scenario: Empilement de toasts Given plusieurs événements surviennent When plusieurs toasts sont déclenchés Then ils s'empilent verticalement And chacun a son propre timer

DÉPENDANCES

DEP: US-7.1 Structure chat

EPIC

Epic 7 — Frontend Angular

US-8.1 Tests manuels E2E

07/12 → | Elodie T. | MUST | IN PROGRESS

DESCRIPTION

En tant que QA / développeur,
Je veux exécuter des tests manuels end-to-end,
Afin de valider le bon fonctionnement de l'application complète.

CRITÈRES D'ACCEPTATION (GHERKIN)

Feature: Tests manuels E2E En tant que QA Je veux exécuter des tests E2E Afin de valider l'application
Scenario: Scénario 1 - Génération d'US simple Given je démarre une nouvelle conversation When j'écris "Je veux une page de login avec email et mot de passe" Then l'agent pose des questions de clarification And après mes réponses, une US est générée And l'US contient titre, description, critères Gherkin
Scenario: Scénario 2 - Upload de fichier Given je suis sur une conversation When j'uploade un fichier PDF de specs And j'écris "Génère une US à partir de ce document" Then l'agent parse le fichier And l'agent génère une US basée sur le contenu
Scenario: Scénario 3 - Knowledge Base Given j'ouvre le drawer KB When j'uploade un document d'architecture Then le document passe en statut "indexing" puis "indexed" And quand je pose une question liée, l'agent cite ce document
Scenario: Scénario 4 - Historique conversations Given j'ai plusieurs conversations When je navigue entre elles via la sidebar Then chaque conversation charge ses messages And je peux supprimer une conversation
Scenario: Scénario 5 - Gestion des erreurs Given le backend est surchargé (simulé) When je tente d'envoyer un message Then un message d'erreur approprié s'affiche And l'application reste utilisable
Lien : Plan de Test Manuel POC

DÉPENDANCES & CONTRAINTES

DEP: Toutes les US fonctionnelles

EPIC

Epic 8 — Tests & Documentation