Consignes pour le projet Shoot Me Up avec XCL

Bonjour, *T1 2024-2025*

Si vous lisez ce document c'est que vous démarrez un projet et que c'est moi - Xavier Carrel - qui porte la casquette de chef de projet.

Pour que notre collaboration soit le plus efficace possible, je tiens à vous communiquer à travers ce document certains points auxquels je tiens dans la manière de mener un projet.

Méthodologie

Je me reconnais pleinement dans les valeurs énoncées par le manifeste agile, que je me permets donc de rappeler ici :

> - Des personnes et leurs interactions plutôt que des outils et des processus
> - Des logiciels qui fonctionnent plutôt qu'une documentation exhaustive
> - La collaboration avec le client plutôt que la négociation contractuelle
> - L'adaptation au changement plutôt que le suivi d'un plan

Par conséquent, je préconise l'utilisation de méthodes agiles (plus précisément Scrum) pour la gestion de projet.

Pratiques

Analyse fonctionnelle

Je demande à ce que l'analyse fonctionnelle se présente sous la forme d'une liste de User Stories (US) accompagnées d'un nombre significatif de tests d'acceptance.

La formulation des tests d'acceptance est grandement simplifiée lorsqu'ils font référence à des éléments graphiques (maquette, schéma, diagramme,...).

Le sens et la clarté de ces éléments priment sur leur qualité graphique. Je préfère un dessin fait à la main et digitalisé à une maquette réalisée avec Figma si cette dernière prend plus de temps à être réalisée que le premier.

Daily Scrum

Votre projet est individuel ? Vous trouvez peut-être que le Daily Scrum n'a donc pas de sens (on ne fait pas une mêlée avec une seule personne!).
Je vous recommande vivement de conserver cette pratique malgré tout. Prenez les premières minutes de chaque journée de travail pour prendre un peu de recul et faire le point.

- Qu'est-ce que j'ai fait hier ? (l'occasion de vérifier que votre journal de travail est à jour)
- Qu'est-ce que je pense faire aujourd'hui ?
- Est-ce que j'ai un problème qui me freine ou me bloque ? (l'occasion de planifier un moment d'aide avec un tiers)

Outils

Un outil est quelque chose qui nous aide à réaliser un travail plus efficacement. On creuse par exemple plus facilement un trou dans le sol avec une pelle qu'avec ses mains.
Je liste donc ici des outils informatiques qui - j'en suis convaincu - vous aident à réaliser votre projet.

Conformément aux valeurs agiles, les individus et leurs interactions priment sur l'outil. Je suis donc prêt à discuter et à m'adapter si vous souhaitez faire usage d'autres outils que ceux ci-dessous. Attention: il faudra savoir me convaincre que vous serez plus performants avec l'outil que vous proposez.

Gestion de projet: IceScrum

Créez votre projet sur etml.icescrum.com et nommez-moi Product Owner.

Il ne contient qu'une seule release et un seul sprint, qui couvrent tous deux l'entier du trimestre.

Rédigez vos US dans la SandBox.

Vous estimez et planifiez vos stories vous-même.

Vous vous servez du task board du sprint actif pour faire le détail de vos tâches:

- Estimez la durée (en minutes) de la tâche lors de sa création (champ "Remaining Time"). Veillez à ce que les tâches ne soient pas trop grosses (max 120 minutes)
- Mettez un tag décrivant le type de tâche, par exemple: Codage, Analyse, GestionProjet, Réunion, ...
- Inscrivez le temps effectif ("Time Spent") quand vous terminez la tâche
- Quand vous devez interrompre une tâche (fin de journée par exemple), notez le temps passé et ré-estimez le temps restant

Journal de travail et Documentation: IceTools

Si vous utilisez IceScrum, installez et apprenez à vous servir des IceTools grâce à ces tutos.

Note: Le journal de travail généré par TimeSheet a un inconvénient: si vous avez une tâche qui vous prend une heure un jour et une autre le lendemain, cette tâche comptera pour deux heures sur le deuxième jour. Cette erreur d'arrondi est tolérable à mes yeux. Elle restera minime tant que vous ne créez pas des tâches trop grosses.

Contrôle de version: GitHub

- Créez un repo privé dans votre compte et invitez-moi en lecture seule.
- Suivez la convention de nommage des commits conventional commits.
- Appliquez le modèle gitflow pour la gestion des branches, avec les branches `main`, `develop` et les branches de features.
- Tous les fichiers utiles au travail et autres que du code (documentation technique interne ou externe, rapports, exemples, snippets, etc...) se trouvent dans l'arborescence sous `doc`, dont la structure est livrée à votre bon sens.
- Bien que cela soit une pratique controversée, je demande à ce que les fichiers concernant la documentation soient également intégrés dans le repository. Ils doivent se trouver dans un dossier `doc` à la racine du repo.

Aide à la réalisation : une IA ?

Le recours à une Intelligence Artificielle pour vous aider dans la réalisation de votre cahier des charges est autorisé - voire encouragé - pour autant que l'IA reste un **outil** qui vous **aide** et non un système qui fait le travail **à votre place**.
En d'autres termes, vous devez être capable d'expliquer vous-même tout ce qui est produit dans le cadre de votre projet.

Documentation

Conformément aux valeurs agiles, ce que vous réalisez est plus important que la documentation à mes yeux. Il n'est cependant pas possible de s'affranchir de toute documentation.
Je souhaite que la documentation suive une organisation légèrement différente - mais pas en contradiction - du modèle ETML ou TPIVD. Je vous fournis un modèle Word qui contient ces différences.

Une différence notable que je tiens à mentionner ici est la rédaction d'une section sur l'usage - ou pas - de l'IA dans votre projet.

Conclusion

Il ne serait pas cohérent de ma part de vous obliger à mettre en œuvre tout ce qui précède à la lettre, parce que ce serait contraire aux valeurs du manifeste agile citées au début de ce document.
Je reste par conséquent ouvert à discuter et mettre en œuvre des pratiques différentes que vous souhaiteriez me proposer.