Projet borne de commande - Picard

Pour le développement de l'application de contrôle des bornes de commandes pour Picard, nous utiliserons la méthode Scrum. Scrum est un cadre de travail agile qui permet une gestion itérative et incrémentale des projets, particulièrement adapté au développement logiciel. Ce document décrit la façon dont ce projet sera orchestré, les outils qui seront utilisés, ainsi que les rôles et processus clés qui guideront le travail de l'équipe. Si un point n'est pas clair, n'hésitez pas à contacter le Scrum Master et auteur de ce document, Malo Braunschweig (malo.braunschweig@live.fr).

Organisation de l'équipe Scrum

Dans le cadre de la méthode Scrum, les rôles suivants seront définis :

1. **Scrum Master** : Le Scrum Master veille à la bonne application des principes Scrum, supprime les obstacles qui pourraient entraver l'équipe et s'assure que l'équipe adhère au cadre de Scrum.
2. **Product Owner (PO)** : Le PO représente les intérêts de Picard et est responsable de maximiser la valeur du produit. Il priorise les fonctionnalités et maintient le backlog produit, qui est une liste ordonnée des éléments à développer.
3. **Équipe de développement** : Constituée de développeurs, designers, et testeurs, cette équipe est auto-organisée et responsable de la réalisation des éléments du backlog produit dans chaque sprint.

Outils recommandés

Il est à noter que les outils sont des suggestions, Picard ayant le dernier mot sur les outils utilisés. A cet effet, chaque catégorie de logiciel comporte deux choix recommandés par le Scrum Master.

1. **Outil de gestion de projet : Jira ou Trello** : Pour la gestion du backlog produit et des sprints, un outil comme Jira ou Trello sera utilisé. Cela permettra de suivre l'avancement des tâches, de gérer les priorités, et de visualiser les itérations.

   Conseils de Scrum Master:

***Jira** serait plus approprié si le projet nécessite une gestion complexe avec des fonctionnalités avancées, notamment pour une grande équipe technique.*

Trello** conviendrait mieux si le projet reste simple, avec un besoin de flexibilité et une interface utilisateur facile d'accès pour une équipe diversifiée ou moins technique. - Je recommande **Trello.

2. **Outil de communication : Slack ou Microsoft Teams** : Pour faciliter la communication instantanée au sein de l'équipe, un outil de messagerie comme Slack ou Microsoft Teams sera mis en place.

   Conseils de Scrum Master:

Slack** serait plus adapté si l'équipe recherche une solution de communication simple, intuitive, avec de nombreuses intégrations et une grande flexibilité. - Je recommande **Slack.

***Microsoft Teams** conviendrait mieux si l'équipe utilise déjà l'écosystème Microsoft 365, nécessitant une collaboration poussée sur des documents, et ayant besoin de fonctionnalités robustes pour les réunions et la gestion des fichiers.*

3. **Outil de gestion du code source : GitHub ou GitLab** : Pour le versionnement du code et la gestion des dépôts, GitHub ou GitLab seront utilisés, assurant un con=trôle des versions et une collaboration efficace entre les développeurs.

   Conseils de Scrum Master:

GitLab** serait plus adapté si l'équipe recherche une solution DevOps complète avec des fonctionnalités CI/CD avancées et la possibilité d'auto-hébergement pour un contrôle total sur l'infrastructure. - Je recommande **GitLab.

***GitHub** conviendrait mieux si l'équipe privilégie une plateforme avec une interface conviviale, une large communauté open source, et une intégration native de fonctionnalités d'automatisation via GitHub Actions, en particulier pour des projets publics ou collaboratifs.*

4. **Outil de CI/CD : Jenkins ou GitLab CI** : Pour automatiser les tests et les déploiements, un outil d'intégration et de déploiement continu comme Jenkins ou GitLab CI sera mis en place.

   Conseils de Scrum Master:

Jenkins *serait plus adapté si l'équipe a besoin d'une solution CI/CD très flexible et extensible, capable de s'intégrer avec une large variété d'outils et de technologies, et si elle dispose des compétences nécessaires pour gérer la complexité de son installation et maintenance.*

GitLab CI *conviendrait mieux si l'équipe recherche une solution CI/CD intégrée et facile à utiliser, particulièrement si elle utilise déjà GitLab pour la gestion de code, et souhaite minimiser la maintenance en centralisant les outils sur une seule plateforme. - Je recommande* ***GitLab CI.***

Processus et cérémonies Scrum

Le projet sera découpé en **sprints** de deux semaines. Chaque sprint inclura les cérémonies Scrum suivantes :

1. **Sprint Planning** : Avant le début de chaque sprint, l'équipe se réunira pour planifier le travail à accomplir. Le PO présentera les priorités du backlog produit, et l'équipe sélectionnera les éléments sur lesquels elle travaillera pendant le sprint à venir. La capacité de l'équipe sera prise en compte pour ne pas surcharger le sprint.
2. **Daily Scrum** : Chaque jour ouvré, l'équipe tiendra une réunion de 15 minutes maximum pour discuter de l'avancement du sprint, des obstacles rencontrés, et planifier les actions du jour. Cela garantit une communication continue et la transparence au sein de l'équipe.
3. **Sprint Review** : À la fin de chaque sprint, une revue de sprint sera organisée pour présenter les fonctionnalités développées aux parties prenantes. Le PO validera ou ajustera les éléments livrés en fonction des retours.
4. **Sprint Retrospective** : Après la revue de sprint, l'équipe tiendra une rétrospective pour discuter de ce qui a bien fonctionné, de ce qui pourrait être amélioré, et des actions à entreprendre pour le prochain sprint. Cette réunion favorise l'amélioration continue des processus.

Timeline du projet:

1. **Initialisation du projet** : Mise en place des outils, formation de l'équipe, et collecte des besoins avec le Product Owner. Le backlog produit initial sera créé à ce stade.
2. **Développement des fonctionnalités** : L'équipe travaillera par cycles de sprints pour développer les fonctionnalités. Chaque sprint aboutira à un incrément fonctionnel du produit, potentiellement livrable.
3. **Test et validation** : Les tests unitaires et d'intégration seront réalisés en continu, grâce à la mise en place de l'intégration continue (CI). Une phase de test utilisateur (UAT) sera également prévue avant chaque release majeure.
4. **Déploiement** : Une fois les fonctionnalités validées, elles seront déployées sur l'environnement de production de manière progressive, selon un calendrier établi en collaboration avec Picard.
5. **Maintenance et amélioration continue** : Après la livraison de l'application, une phase de maintenance et d'amélioration continue sera enclenchée, intégrant les retours des utilisateurs finaux.

Estimation du temps nécessaire

L'estimation du temps nécessaire à chaque étape sera faite en collaboration avec l'équipe pendant le Sprint Planning. Pour estimer de manière précise, nous utiliserons la technique des **points de complexité** et les **historiques des sprints précédents** pour affiner les prévisions.

Conclusion

Grâce à l'utilisation de Scrum, le projet de développement de l'application de contrôle des bornes de commandes pour Picard pourra être géré de manière flexible, réactive et transparente. L'équipe bénéficiera d'une organisation claire, d'une communication continue et d'une amélioration continue de ses pratiques, garantissant ainsi un produit final de haute qualité, livré dans les délais prévus.

Pour toute question merci de contacter le Scrulm Master et auteur de ce document à l'adresse suivante: malo.braunscheig@live.fr