Équipe EOS Marketplace

EOS Marketplace V1
Spécifications des requis du système (SRS)

Version 2.0

Historique des révisions

Date	Version	Description	Auteur
aaaa-mm-jj	x.x	<Détails précis du travail effectué>	<Nom>
2021-09-05	0.1	Rédiger les exigences non-fonctionnelles	Guilhem Dubois
2021-09-08	0.2	Rédiger les contraintes générales et les hypothèses de dépendance	Nicolas Hirab
2021-09-09	1.0	Rédiger les normes applicables et 3.3/3.4	Nicolas Hirab
2021-09-17	1.1	Ajout d'exigences fonctionnelles et de précisions dans les exigences d'utilisabilité	Équipe
2021-09-25	1.2	Mise à jour des exigences	Équipe
2021-09-28	1.3	Correction dans les technologies utilisées	Gabriel Dambreville
2021-10-08	2.0	Mise à jour des exigences	Équipe
2021-12-02	3.0	Correction du SRS avec les commentaires de la correction de mi-session	Arthur Garnier
2021-12-03	3.1	Correction des exigences selon le développement du projet	Équipe

Table des matières

1. Introduction **5**
1.1. But 5
1.2. Définitions, acronymes et abréviations 5
1.3. Vue d'ensemble du document 5

2. Description globale **5**
2.1. Caractéristiques des usagers 5
2.2. Interfaces 5
2.2.1. Interfaces usagers 5
2.2.2. Interfaces matérielles 6
2.2.3. Interfaces logicielles 6
2.2.4. Interfaces de communication 6
2.3. Contraintes générales 6
2.4. Hypothèses et dépendances 6

3. Exigences fonctionnelles **6**
3.1. Interface mobile pour la connexion 6
3.2. Interface mobile pour la création d'une annonce 6
3.3. Carte de Sherbrooke ou Montréal 7
3.4. Compte utilisateur et client 7
3.5. Authentification EOS 7
3.6. SWAP 7
3.7. Intégration d'un processeur de paiement 7
3.8. Intégration paiement par cryptomonnaie 7
3.9. Création du système de référencement automatique 8
3.10. Création de contrats intelligents EOS 8

4. Exigences non-fonctionnelles **8**
4.1. Utilisabilité 8
4.2. Fiabilité 8
4.3. Performance 9
4.4. Maintenabilité 9
4.5. Contraintes de conception 9
4.6. Sécurité 9
4.7. Exigences de la documentation usager en ligne et du système d'assistance 10
4.8. Normes applicables 10

Spécifications des requis du système (SRS)

1. Introduction

1.1. But

Le SRS décrit le comportement externe d'une application. Il décrit aussi les exigences non fonctionnelles, les contraintes de conception, ainsi que les autres facteurs nécessaires à la description complète des exigences du logiciel à développer.

1.2. Définitions, acronymes et abréviations

Anchor Wallet: Un portefeuille numérique open-source axé sur la sécurité et la confidentialité pour tous les réseaux basés sur EOSIO.

Cadastre: Un cadastre est le plan d'un lot inscrit dans un registre dressant l'état de la propriété foncière. Un plan cadastral comporte une vue d'ensemble de cadastres d'une circonscription foncière, d'un territoire foncier ou d'un immeuble détenu en copropriété.

Contrat d'échange: Un contrat d'échange peut représenter une offre ou une demande de bien et de service.

Contrats intelligents Protocoles informatiques qui facilitent, vérifient et exécutent la négociation ou l'exécution d'un contrat.

Crypto-monnaie: Monnaie numérique en usage sur Internet, indépendante des réseaux bancaires et liée à un système de cryptage.

EOSIO / EOS: Plateforme décentralisée basée sur le principe de blockchain, permettant de déployer des applications utilisant des contrats intelligents. EOS peut aussi faire référence à la crypto-monnaie associée à cette plateforme.

Ethereum: Plateforme globale et open-source pour des applications décentralisées permettant la création par les utilisateurs de contrats intelligents.

ETH/Ether: La crypto-monnaie native de la plateforme Ethereum. Parmi les crypto-monnaies, Ether est juste derrière Bitcoin en termes de capitalisation boursière.

MoonPay: Service permettant de payer à partir de monnaie fiduciaire tel que des dollars canadiens qui sont ensuite convertis automatiquement en crypto-monnaie.

Stablecoin: Crypto-monnaie dont la valeur reste stable et qui simule la valeur d'un dollar fiduciaire afin d'éviter le problème de volatilité de plusieurs autres crypto-monnaies.

SWAP: Permet aux utilisateurs d'échanger facilement une crypto-monnaie contre une autre.

Wrapped Bitcoin: Crypto-monnaie ayant la même valeur que les Bitcoin mais pouvant être utilisé dans les transactions sur une blockchain telle que EOS.

1.3. Vue d'ensemble du document

Ce document est subdivisé en 4 parties. La section 2 de ce document est une description générale du logiciel. Dans cette section une description des usagers, interfaces, contraintes générales et hypothèses et dépendances est mise de l'avant. La section 3 de ce document porte sur les exigences fonctionnelles de ce document. La dernière section, la section 4 est une section dédiée aux exigences non-fonctionnelles.

2. Description globale

EOS Marketplace V1 est un projet de la compagnie EOS Nation de la conception d'une interface mobile pour l'octroi de travail. Ce projet de mise en relation d'offres et de la demande est comparable aux compétiteurs "Facebook Marketplace" et "Kijiji.ca". Le projet permettra à l'utilisateur de faire des annonces de biens et services qui seront affichés dans des vues listes et carte. Afin de prendre avantage du réseau de cryptomonnaie EOS, l'authentification à l'application, les ententes entre utilisateurs, les contrats et paiements seront enregistrés sur la blockchain EOS.

2.1. Caractéristiques des usagers

Les utilisateurs principaux de l'application EOS Marketplace ont besoin d'avoir les connaissances de base de la technologie afin de pouvoir s'enregistrer, explorer l'application et effectuer un paiement mobile. Par exemple, au moment de l'enregistrement, on retrouve 2 types d'utilisateurs: les utilisateurs ayant déjà un portefeuille Anchor et les utilisateurs qui n'ont pas de portefeuille Anchor (et qui ne connaissent probablement pas Anchor). Pour ceux qui ne connaissent pas Anchor, la plateforme permet de guider ces utilisateurs pour la création de leur portefeuille, donc ça ne requiert pas de connaissances préalables.

Étant donné qu'un support de la technologie EOS est intégré, un utilisateur cible aura des connaissances de base de cryptomonnaie. Par contre, les paiements argent seront aussi acceptés donc un utilisateur plus ou moins technique devrait pouvoir effectuer des transactions. L'interface sera simple à utiliser, donc les formations logiciel ne devraient pas être nécessaires.

2.2. Interfaces

2.2.1. Interfaces usagers

L'interface usager pour ce projet sera sous forme d'application mobile multi-plateforme et sera donc disponible sur IOS et Android. L'application nécessitera l'intégration de Montréal ou Sherbrook et son cadastre, afin de situer les échanges. L'interface devra permettre les actions suivantes:

- Facile à utiliser, facilité de partage sur les réseaux sociaux.
- connexion avec les identifiants existant (ex : facebook, google) et avec un compte blockchain EOS
- Permettre la création d'une annonce(description, matériel requis, santé sécurité (SST), images)
- Permettre l'ajout de photos des travaux accomplis (avant/après)
- Mettre en relation offre/demande

2.2.2. Interfaces matérielles

Tous les composants côté serveur seront exécutés sur le Cloud à travers Google Cloud. Les protocoles blockchain utiles à l'application seront déclenchés sur la blockchain EOS. L'application fonctionnera sur les mobiles android et IOS (Tablette et cellulaire).

2.2.3. Interfaces logicielles

Une solution d'achat du EOS token sera disponible à travers MoonPay qui pourra être directement intégré à l'application. Pour l'authentification, la connexion par le "EOS Nation login" sera aussi disponible. Pour la carte de Montréal, on utilisera l'API de Open Street Maps. Nous y intégrerons le cadastre en arrière pour qu'on puisse associer une transaction à celui-ci. Pour la signature des transactions sur la blockchain, nous utiliserons l'application Anchor wallet.

2.2.4. Interfaces de communication

L'interface client communiquera avec le serveur à travers internet. Les routes disponibles seront "/post", pour la gestion des annonces sur le serveur, "/auth" pour tout ce qui concerne les utilisateurs, "/cadastre" et "/address" pour les actions concernant la carte interactive, "/chatRooms" et "/chatMessages' pour la fonctionnalité de clavardage de l'application, et finalement, "/contract" la gestion du contrat sur le serveur. Les contrats intelligents utilisent le protocole EOSIO sur la blockchain EOS.

2.3. Contraintes générales

L'application est contrainte à l'utilisation de périphériques traditionnels (clavier, souris, accès à un réseau internet...). De plus, les performances seront limitées par celle de l'infrastructure Google Cloud, tant physique que logiciel. L'application ne pourra qu'être lancée sur des appareils mobiles.

2.4. Hypothèses et dépendances

Le logiciel sera hébergé sur les serveurs de Google Cloud .

3. Exigences fonctionnelles

3.1. Interface mobile pour la connexion

3.1.1 [Essentiel] L'utilisateur doit pouvoir se connecter avec un compte existant.

3.1.2 [Essentiel] L'utilisateur doit pouvoir s'enregistrer avec un compte existant Google ou Facebook.

3.1.2.1 [Essentiel] Après la connexion, l'utilisateur doit entrer le nom de compte de son Wallet EOS.

3.1.2.2 [Essentiel] L'utilisateur doit pouvoir créer un Wallet EOS.

3.1.3 [Essentiel] L'utilisateur doit pouvoir s'enregistrer avec un compte blockchain EOS manuellement.

3.1.3.1 [Essentiel] L'utilisateur doit entrer le nom d'utilisateur de son Wallet EOS.

3.1.3.2 [Essentiel] L'utilisateur doit signer son compte à l'aide d'anchor pour confirmer son appartenance.

~~*3.1.4 [Essentiel] L'utilisateur doit pouvoir s'enregistrer avec un compte EOS Nation.*~~

3.1.5 [Essentiel] L'utilisateur doit pouvoir s'enregistrer avec Anchor Wallet.

3.2. Interface mobile pour une annonce

3.2.1 [Essentiel] L'utilisateur doit pouvoir créer une annonce pour un service.

3.2.1.1 [Essentiel] L'annonce doit permettre d'inclure la description du travail.

3.2.1.2 [Essentiel] L'annonce doit permettre d'inclure le matériel requis pour le travail.

3.2.1.3 [Essentiel] L'annonce doit permettre d'inclure des images.

3.2.1.4 [Essentiel] L'annonce doit permettre d'inclure un prix.

3.2.1.5 [Essentiel] L'utilisateur doit pouvoir modifier le prix au moment de proposer un contrat

3.2.1.6 [Essentiel] L'utilisateur doit pouvoir proposer ou chercher un service.

3.2.1.7 [Essentiel] L'utilisateur doit pouvoir localiser son annonce sur un cadastre de la carte.

~~*3.2.2 [Essentiel] L'utilisateur doit pouvoir créer une annonce pour un bien.*~~

~~*3.2.2.1 [Essentiel] L'annonce doit permettre d'inclure la description du bien.*~~

~~*3.2.2.3 [Essentiel] L'annonce doit permettre d'inclure des images.*~~

~~*3.2.2.4 [Essentiel] L'annonce doit permettre d'inclure un prix.*~~

~~*3.2.2.5 [Essentiel] L'utilisateur doit pouvoir modifier les informations de l'annonce ultérieurement.*~~

3.2.3 [Essentiel] L'utilisateur doit pouvoir voir une annonce.

3.2.3.1 [Essentiel] L'utilisateur doit pouvoir différencier un service recherché et offert.

3.2.3.2 [Essentiel] L'annonce doit avoir une description.

3.2.3.3 [Essentiel] L'annonce doit avoir des images descriptives.

3.2.3.4 [Essentiel] L'annonce doit avoir une miniature.

3.2.3.5 [Essentiel] L'annonce doit afficher le prix.

3.2.3.5.1 [Essentiel] L'annonce doit afficher le prix en dollars canadiens.

3.2.3.5.2 [Essentiel] L'annonce doit afficher le prix en jetons EOS.

3.2.3.6 [Essentiel] L'utilisateur doit voir les informations pour contacter la personne qui a publié l'annonce.

3.3. Carte de Sherbrooke ou Montréal

3.3.1 [Essentiel] L'utilisateur doit pouvoir marquer sur la carte la localisation d'une annonce.

3.3.2 [Essentiel] L'utilisateur doit pouvoir naviguer sur la carte.

3.3.3 [Essentiel] La carte doit afficher les annonces par cadastre.

3.3.4 [Essentiel] L'utilisateur doit pouvoir cliquer sur une annonce pour ouvrir sa page de détails.

3.3.5 [Essentiel] L'utilisateur doit pouvoir filtrer les annonces par catégorie.

3.3.6 [Essentiel] L'utilisateur doit pouvoir distinguer une catégorie de service sur la carte avec des icônes.

3.4. Compte utilisateur

3.4.1 [Essentiel] L'application doit permettre aux utilisateurs d'afficher les informations publiques d'autres utilisateurs.

3.4.1.1 [Essentiel] La page publique d'un utilisateur doit afficher son nom

3.4.1.2 [Essentiel] La page publique d'un utilisateur doit afficher sa date d'inscription.

3.4.1.3 [Essentiel] La page publique d'un utilisateur doit afficher sa description.

3.4.1.4 [Essentiel] La page publique d'un utilisateur doit permettre d'afficher leur référence(s).

3.4.1.5 [Essentiel] La page publique d'un utilisateur doit permettre d'afficher leur certification(s).

3.4.1.6 [Essentiel] La page publique d'un utilisateur doit permettre d'afficher l'évaluation moyenne d'autres utilisateurs.

~~*3.4.1.7 [Souhaitable] La page publique d'un utilisateur doit permettre d'afficher les commentaires d'autres utilisateurs.*~~

3.4.2 [Essentiel] L'application doit afficher les informations privées de l'utilisateur.

3.4.2.1 [Essentiel] La page privée d'un utilisateur est uniquement accessible par lui-même

3.4.2.2 [Essentiel] La page privée d'un utilisateur doit afficher ses informations personnelles (nom, date d'inscription, adresse courriel, numéro de téléphone).

3.4.2.3 [Essentiel] La page privée d'un utilisateur doit permettre d'afficher un contrat en cours.

3.4.2.4 [Essentiel] La page privée d'un utilisateur doit permettre d'afficher un contrat ouvert.

3.4.2.5 [Essentiel] La page privée d'un utilisateur doit permettre d'afficher un contrat terminé.

3.4.2.6 [Essentiel] La page privée d'un utilisateur doit permettre d'afficher un service proposé par lui-même.

3.4.2.7 [Essentiel] La page privée d'un utilisateur doit permettre d'afficher un service recherché par lui-même.

3.4.2.8 [Essentiel] La page privée d'un utilisateur doit permettre d'afficher l'évaluation moyenne d'autres utilisateurs.

3.4.2.9 [Essentiel] La page privée d'un utilisateur doit permettre l'achat de cryptomonnaie.

~~*3.4.2.10 [Souhaitable] La page privée d'un utilisateur doit permettre d'afficher le commentaire d'un autre utilisateur.*~~

3.4.3 [Essentiel] L'utilisateur doit pouvoir accéder au profil d'un autre utilisateur de l'application.

3.4.3.1 [Essentiel] L'utilisateur doit pouvoir rechercher un utilisateur par son nom d'utilisateur.

3.4.3.2 [Essentiel] L'utilisateur doit pouvoir accéder au profil d'un utilisateur à partir de ses contrats d'échange.

3.4.4 [Essentiel] L'utilisateur doit pouvoir accéder à la liste de ses contrats créés.

3.4.4.1 [Essentiel] Les contrats doivent être groupés par status ("open", "active", "done").

3.4.4.2 [Essentiel] L'utilisateur doit pouvoir cliquer sur un contrat afin d'afficher ses détails.

3.5. Accepter une annonce

3.5.1 [Essentiel] L'utilisateur doit pouvoir accepter une annonce.

3.5.2 [Essentiel] L'utilisateur qui a créé l'annonce doit être notifié lorsqu'un utilisateur accepte son annonce.

3.5.3 [Essentiel] Un contrat intelligent doit être créé automatiquement lorsqu'un utilisateur accepte une annonce.

3.5.4 [Essentiel] L'utilisateur proposant le service doit pouvoir ajouter une photo liée au contrat

3.5.5 [Essentiel] L'utilisateur doit pouvoir approuver une transaction.

3.5.6 [Essentiel] Le vendeur doit pouvoir annuler une annonce tant que le vendeur n'a pas déposé de fonds.

3.5.7 [Essentiel] L'acheteur doit pouvoir annuler une annonce tant qu'il n'a pas déposé de fonds.

3.5.8 [Essentiel] L'acheteur doit pouvoir déposer des fonds dans le contrat intelligent.

~~3.6. SWAP~~

~~*3.6.1 [Essentiel] L'utilisateur doit pouvoir acheter des jetons de la plateforme.*~~

~~*3.6.2 [Souhaitable] L'utilisateur doit pouvoir échanger des crypto-monnaies disponibles sur EOS (Stablecoins,*~~

~~*wrapped BTC, ETH sur EOS) contre le jeton EOS.*~~

3.7. Intégration d'un processus de paiement

3.7.1 [Essentiel] L'utilisateur doit pouvoir acheter de la cryptomonnaie à l'aide de la plateforme MoonPay.

3.7.1.1 Lors de l'utilisation de MoonPay au môment du paiement, l'utilisateur doit pouvoir payer en dollars canadiens.

3.7.1.2 Lors de l'utilisation de MoonPay au moment du paiement, les dollars canadiens doivent être automatiquement convertis dans la crypto-monnaie EOS.

3.7.1.3 L'utilisateur doit pouvoir payer par carte de débit.

3.7.1.4 L'utilisateur doit pouvoir payer par carte de débit.

3.7.1.5 L'utilisateur doit pouvoir payer à l'aide de Google Pay

3.7.1.6 L'utilisateur doit pouvoir payer à l'aide d'Apple Pay.

3.7.1.7 La cryptomonnaie EOS acheté doit être directement créditée dans le wallet EOS de l'utilisateur.

3.8. Intégration paiement par cryptomonnaie

3.8.1 [Essentiel] L'utilisateur doit pouvoir payer avec la crypto-monnaie EOS.

~~*3.8.2 [Souhaitable] L'utilisateur doit pouvoir payer par la crypto-monnaie Bitcoin à l'aide de Wrapped Bitcoins (pBTC de pNetwork sur EOS).*~~

~~*3.8.3 [Souhaitable] L'utilisateur doit pouvoir payer par la crypto-monnaie Ethereum (pETH de pNetwork sur EOS).*~~

3.8.4 [Essentiel] Un paiement en crypto-monnaie doit être effectué sur la blockchain EOS afin de diminuer le temps et le coût de la transaction.

3.9. Création du système de référencement automatique

3.9.1 [Essentiel] L'utilisateur doit pouvoir référencer un contrat.

~~*3.9.2 [Essentiel] L'utilisateur doit pouvoir suivre l'état du contrat référé.*~~

~~*3.9.3 [Optionnel] Le système doit pouvoir récompenser la personne qui a référencé le contrat automatiquement lors de la complétion de celui-ci.*~~

3.10. Création de contrats intelligents EOS

3.10.1 [Essentiel] L'utilisateur doit pouvoir créer un contrat employé/employeur intelligent sur EOS.

3.10.2 [Essentiel] Le montant du contrat doit être mis en escrow lors de la création du contrat.

3.10.3 [Essentiel] La transaction doit être validée par les deux parties prenantes du contrat

3.10.4 [Essentiel] Lors de la validation du contrat, la transaction du montant est complétée.

3.10.5 [Essentiel] L'utilisateur doit pouvoir évaluer le service de l'autre utilisateur.

3.11. Intégration d'une messagerie

3.11.1 [Essentiel] L'utilisateur doit pouvoir communiquer avec l'émetteur de contrat.

3.11.2 [Essentiel] L'utilisateur doit pouvoir envoyer un contrat à travers la messagerie

3.11.3 [Souhaitable]L'utilisateur doit recevoir une notification lorsqu'un message est reçu.

3.11.4 [Souhaitable] L'utilisateur doit pouvoir revenir sur l'historique d'une conversation.

3.11.5 [Souhaitable] L'utilisateur doit pouvoir envoyer une image.

4. Exigences non-fonctionnelles

4.1. Utilisabilité

4.1.1. L'interface utilisateur doit être cohérente entre la section de vente et la section d'achat.

4.1.1.1. Les champs entrés lors de l'ajout d'un service doivent avoir la même terminologie que ceux lors de l'achat du produit/service.

4.1.1.2. L'interface doit utiliser les mêmes éléments graphiques entre les deux sections.

4.1.2. L'interface utilisateur doit être facile d'utilisation.

4.1.2.1. Sauf pour la description d'une annonce, l'interface devrait avoir une limite de 50 mots affichés en même temps sur une page.

4.1.2.2. Un bouton doit avoir une dimension d'au moins 0.75cm de largeur et de hauteur afin d'être facilement cliquables sur appareil mobile.

4.1.3. L'interface utilisateur doit suivre les principes important de UI/UX

4.1.3.1. Un bouton important doit être mis en évidence par son style, à l'aide d'une couleur différente.

4.1.3.2. L'application doit avoir moins de 3 types de couleurs

4.1.3.3. L'application doit avoir moins de 5 teintes par couleur.

4.1.3.4. Le contraste entre les éléments doit permettre au texte d'être facilement lisible.

4.1.3.5 Le groupement des éléments entre eux doit permettre de créer un lien entre ces éléments

4.1.4. Ajouter une annonce doit prendre moins de 5 minutes.

4.2. Fiabilité

4.2.1. Le serveur doit être disponible au moins 99% du temps prévu.

4.2.2. Le serveur doit avoir un temps moyen entre les pannes d'au minimum 3 mois.

4.2.3. Le serveur doit avoir un temps moyen jusqu'à la réparation de maximum 2 jours.

4.2.4. En cas d'erreur, le serveur doit redémarrer automatiquement.

4.2.5. Si le serveur est hors ligne, l'application doit afficher un message d'erreur.

4.2.6. En cas d'erreur lors d'une transaction, elle doit être annulée.

4.3. Performance

4.3.1. Chaque action, excepté la confirmation de la transaction, devrait avoir un délai maximal de 3 secondes.

4.3.2. Chaque action, excepté la confirmation de la transaction, devrait avoir un délai moyen de 500 millisecondes.

4.3.3. Lors de l'achat d'un service, la confirmation de la transaction devrait avoir un délai maximal de 8 secondes.

4.3.4. Lors de l'achat d'un service, la confirmation de la transaction devrait avoir un délai moyen de 3 secondes.

4.3.5. L'application doit permettre à une connexion internet de 1 MB/s (8 Mb/s) de naviguer en respectant les délais

estimés.

4.4. Maintenabilité

4.4.1. Le code des fonctionnalités en cours de développement doivent être dans une branche nommée <#numéro ticket jira>-<fonctionnalité> ex : 23-connexion-facebook

4.4.2. Un message de commit Git doit suivre le format suivant [initiales-nom](type): <commentaire>, ex: [GD](dev): Ajout de la connexion avec facebook

Les différents types sont:

dev - développement

deb - débug

ref - refactoring

tes - test

aut - autre

4.4.3. Tout code "pushed" sur la branche "dev" doit avoir été approuvé par au moins une autre personne via un pull request.

4.4.4. Toute fonctionnalité "pushed" sur la branche "dev" doit avoir des tests correspondants qui passent correctement.

4.4.5. Le code doit garder une couverture de tests d'au moins 75%.

4.4.6. Une remise doit se faire à partir de la branche "master".

4.4.7. Le code et ses commentaires explicatifs doivent être en anglais afin de faciliter la reprise du projet.

4.5. Contraintes de conception

4.5.1. Le front-end de l'application doit être développé avec React Native.

4.5.2. Le serveur doit être hébergé sur Google Cloud avec un compte disponible pour toute l'équipe et pour les clients.

4.5.3. La base de données doit être faite à l'aide de MongoDB.

4.5.4. Le système d'authentification doit être fait avec Firebase.

4.6. Sécurité

4.6.1. Le système doit protéger les informations confidentielles de l'utilisateur.

4.6.2. La connexion au serveur doit être encryptée.

4.6.3. Les technologies utilisées par l'application et le serveur doivent êtres mises à jour aux dernières versions disponibles afin de s'assurer d'avoir les correctifs de sécurité les plus récents.

4.7. Exigences de la documentation usager en ligne et du système d'assistance

4.7.1. L'application doit avoir une section "contact" avec des informations suffisantes pour que l'utilisateur puisse reporter un problème.

4.7.2. L'application doit posséder une documentation usager disponible pour un utilisateur sur l'application.

4.8. Normes applicables

4.8.1 Norme ISO/9126 *Cette norme permet d'aborder un grand nombre de préjugés humains bien connus qui peuvent nuire à la livraison et à la perception d'un projet de développement de logiciels.*

4.8.2 Norme IEEE/12207 *Cette norme permet de définir le cycle de vie du logiciel ainsi que les données du cycle de vie.*