Équipe 203

Chaima Soussi 1925302
Mikael Gilbert 1998202
Léon Le Brun 2020533
Noursen Houidhek 1949635
Nassim Boughedaoui 1946992
Miora Rakoto 2011713

Polygram
Spécifications des requis du système (SRS)

Version 1.1

Historique des révisions

[Dans l'historique de révision, il est nécessaire d'écrire le nom du ou des auteurs ayant travaillé sur chaque version.]

Date	Version	Description	Auteur
2022-01-27	1.0	Rédaction initiale du SRS.	Équipe 203
2022-02-09	1.1	Modifications du niveau d'imbrications des exigences et retrait de certaines exigences.	Chaima Soussi, Mikael Gilbert
2022-02-09	1.2	Modification de la description globale, ajout dans le glossaire, complétion de la partie sécurité (4.6)	Noursen
2022-02-10	1.3	Modification de la section interface matérielle, ajout dans le glossaire, ajout des dépendances.	Miora Rakoto

Table des matières

Spécifications des requis du système (SRS)

1. Introduction

1.1. But

Ce document présente les spécifications du logiciel Polygram. Il décrit en premier lieu les différentes composantes à développer en nous basant sur le document de vision fourni ainsi que les contraintes de conception et les facteurs nécessaires à la description complète des exigences du logiciel à développer. Le SRS décrit dans l'ordre les différentes exigences classées selon les catégories fonctionnelles et non fonctionnelles.

1.2. Définitions, acronymes et abréviations

Glossaire technique:

Android Studio : Environnement de développement pour développer des applications mobiles.

Angular : Cadriciel de développement côté client basé sur TypeScript.

Angular Material : Bibliothèque de composante utilisée par les développeurs d'Angular pour faire des interfaces utilisateurs.

Base de données : Collection d'informations généralement stockées dans un système informatique.

Bogues : Défaut de conception ou de réalisation d'un programme informatique qui se manifeste par des anomalies.

Cadriciel (ang. framework) : Ensemble de bibliothèques, d'outils et de conventions permettant le développement de l'application.

Classe : En programmation orientée objet, une classe décrit les propriétés d'un ou plusieurs objets.

Commit : Dans un système de gestion de versions (ex. : Git), l'action de laisser un message pour décrire les changements effectués.

Dropbox : Service de stockage et de partage de copies de fichiers locaux en ligne.

Electron : Environnement de développement pour des applications multiplateformes de bureau utilisant les technologies du web.

Exécutable : Un fichier contenant un programme.

Hashage: Deuxième couche de protection avant de persister les mots de passe utilisateurs en base de données. Un mot de passe est dit "hashé" lorsqu'il transformé en une version brouillée de lui même.

Heroku: Plateforme permettant de déployer des applications sur le Cloud.

Google Drive : Service de stockage et de partage de fichiers fait par Google.

HTTP/S : Abréviation provenant de l'anglais qui désigne "Hypertext Transfer Protocol". C'est un protocole de communication client-serveur. HTTPS désigne la version sécurisée du protocole.

Infonuagique : Modèle dans lequel le stockage de données et leur traitement sont externalisés sur des serveurs distants accessibles par tout appareil ayant une connexion internet.

Jasmine : Cadriciel pour effectuer des tests pour des applications écrites en JavaScript.

JUnit: Cadriciel pour effectuer des tests pour des applications écrites en Java ou Kotlin.

Kotlin : Langage de programmation orienté objet et fonctionnel à typage statique utilisé notamment pour le développement d'applications Android.

Latence : Temps nécessaire à un paquet de données pour passer de la source à la destination à travers un réseau informatique.

Librairie : Ensemble de fonctions et de routines déjà compilées et/ou prêtes à être utilisées.

Mocha : Cadriciel de test pour des applications en Node.js.

MongoDB: Base de données en NoSQL.

Propriétés de classe : En programmation orientée objet, élément de description d'un objet.

NodeJS : Plateforme logicielle libre qui permet d'exécuter du code JavaScript côté serveur.

NoSQL : Famille de gestion de bases de données qui n'utilise pas la notion de base de données relationnelle.

Serveur : Équipement informatique qui offre des services à un ou plusieurs clients (p. ex. courriel, pages web, stockage dans une base de données).

SocketIO : Bibliothèque pour les applications web permettant une communication bidirectionnelle en temps réel entre les clients et les serveurs.

Story : Publication d'une durée maximale de 24h.

Système de gestion de version : Logiciel qui permet de stocker un ensemble de fichiers (généralement, des fichiers de code) en conservant un historique de toutes les modifications de ces fichiers.

TypeScript : Langage de programmation typée libre et open source développé par Microsoft.

TCP : Abréviation de Transmission Control Protocole, qui désigne le protocole de transmission utilisé dans les réseaux informatiques IP.

Salt: Première couche de protection des mots de passe à sauvegarder en base de données qui consiste à générer des caractères unique et aléatoire qui seront ajoutés au mot de passe avant qu'il ne soit "hashé".

UI (User Interface) : Abréviation provenant de l'anglais qui désigne "interface utilisateur", soit tout dispositif matériel ou logiciel permettant d'interagir avec un produit informatique.

UX (User Experience) : Abréviation provenant de l'anglais qui désigne "expérience utilisateur", soit la qualité du vécu de l'utilisateur dans un environnement numérique (ou physique).

Variable de classe : En programmation orientée objet, variable déclarée avec le mot clé *static.*

Visual Studio Code : Éditeur de code extensible.

WebSocket : Protocole réseau basé sur le protocole TCP.

802.11, 2.4G+5GHz, VHT80: Standard pour le réseau Wi-Fi et son fonctionnement. Les standards de communication Wi-Fi commencent toujours par "802.11" et sont accompagnés par différents caractères tels que "a", "b", "n", ou "ac". 2.4G+5GHz représentent la propriété de la bande passante. VHT80 est la fréquence du signal radio transmis (80-MHz**).**

Glossaire pour Polygram:

Album: Un dessin fait sur Polygram peut être mis dans un album. Un album est un moyen pour l'utilisateur de visualiser différents dessins qui peuvent être les siens comme ils peuvent appartenir à d'autres utilisateurs**.**

Album public: Lorsqu'un dessin est créé dans Polygram il est par défaut placé dans un album public qui est accessible à tous les utilisateurs de l'application.

Album public par défaut: Album accessible à tous les utilisateurs dès la création de leur compte Polygram.

Album privé: Un utilisateur crée un album privé s'il veut restreindre l'accès à ses dessins à seulement un groupe d'utilisateur.

Authentification : Procédure pour vérifier l'identité d'une personne et autoriser l'accès à des ressources.

Avatar : Personnage représentant un utilisateur sur une application.

Badge: Système de reconnaissance basé sur les mentions "j'aime" intégrées au profil utilisateur.

Canaux de discussion: Groupes de clavardage complètement indépendant dont l'utilisateur peut faire partie pour communiquer avec plusieurs personnes en même temps. Les utilisateurs dans un canal de discussion ont un intérêt commun.

Clavardage : Activité par laquelle un internaute tient une conversation écrite et en temps réel avec d'autres internautes.

Client léger : Application exécutable sur une tablette.

Client lourd : Application exécutable compatible avec un ordinateur personnel.

Dessin collaboratif : Dessin fait par plusieurs utilisateurs.

Effet visuels: Rétroaction visuelle que l'utilisateur peut voir sous forme de transition entre les différentes vues de notre application.

Effet sonores: Rétroaction audible par l'utilisateur lors de la majorité de ses opérations d'édition

Identifiant : Code personnel et unique permettant d'accéder à un service informatique.

Menu principal: Page suivant la page de connexion. Le menu principal permet de choisir entre le mode solo ou le mode collaboratif. De plus, il affiche cinq dessins de l'album public par défaut.

Mention "j'aime": Indication par laquelle un utilisateur manifeste son intérêt pour un dessin.

Mode intégré: La boîte de clavardage est intégrée à l'application.

Mode fenêtré : La boîte de clavardage est sur une fenêtre séparée de l'application principale. Ce mode est seulement disponible pour le client lourd.

Mode solo : L'utilisateur dessine seul. Les autres utilisateurs ne peuvent pas dessiner avec lui.

Mode collaboratif : L'utilisateur dessine en temps réel avec d'autres utilisateurs sur le même dessin.

Notification: Un indicateur visuel signale à l'utilisateur qu'un nouveau message lui a été transmis.

Page de connexion: Première page de l'application. Elle demande à l'utilisateur ses informations.

Profil utilisateur : Ensemble de données caractérisant un utilisateur. Chaque profil a une partie privée (courriel, prénom, nom) et une partie publique (pseudonyme et avatar).

Synchronisation simple : Les actions d'éditions de dessins sont propagées aux autres utilisateurs quand l'utilisateur confirme qu'il a fini de dessiner.

Wizz : Animation et son datant de l'époque de MSN qui consiste en la "vibration" visuelle de la fenêtre d'une application de bureau ou mobile.

1.3. Vue d'ensemble du document

La section 2 contient une description générale de l'application, nous précisons la catégorie d'utilisateurs ciblés par l'application. Ensuite, nous détaillons les interfaces usagers, matérielles, logicielles et de communication tout en mentionnant les hypothèses et les contraintes auxquelles nous faisons face . La section 3 a pour objectif d'expliciter chacune de nos exigences fonctionnelles et expliquer ce qui est exactement attendu, les exigences sont regroupées par domaine: **page de connexion, menu principal, Album, Clavardage, profil utilisateur, éditeur de dessin, mode édition, effets visuels et sonores.** Enfin la section 4 est consacrée à la description des exigences non fonctionnelles qui sont : l'utilisabilité, la fiabilité, la performance, la maintenabilité, les contraintes de conception et la sécurité.

2. Description globale

Polygram est une application de bureau pour Windows 10 (client lourd). Elle permet de dessiner en temps réel avec d'autres utilisateurs. L'application sera sous la forme d'un exécutable. La collaboration est possible grâce à un système d'authentification (profil utilisateur) et est facilitée par un système de clavardage. Une version mobile du logiciel (client léger) est également disponible et permet de bénéficier des spécificités d'un appareil Android (p.ex. caméra). Une tablette présentant les mêmes spécifications qu'une Galaxy Tab A avec Android 9.0 est préférable pour l'exécution de la version mobile.

2.1. Caractéristiques des usagers

L'application Polygram s'adresse à un public âgé de 12 et 30 ans. Son utilisation ne nécessite pas de connaissances spécialisées en informatique. Un utilisateur possédant une familiarité avec des applications de dessins similaires, comme MS Paint, pourra utiliser Polygram avec facilité.

2.2. Interfaces

2.2.1. Interfaces usagers

Les interfaces usagers ainsi que l'expérience sont cohérentes entre le client lourd et le client léger, les mêmes chartes de couleurs et manipulations seront possibles et disponibles entre les deux environnements. Afin que les utilisateurs ne soient pas confrontés à des difficultés d'utilisation, la disposition des outils et de la zone de dessin sera comparable aux logiciels de dessins déjà disponibles. Le choix des icônes sera représentatif et conventionnel. Nous allons donner à l'usager la possibilité de créer un compte ou de se connecter par le biais d'une interface de connexion, ensuite il aura accès à son compte qui comporte une interface pour son profil ou il est possible de visualiser ses statistiques et ses paramètres. L'usager aura également accès à une interface d'album et de story afin de visualiser ses dessins et les dessins des autres utilisateurs. Enfin, une interface de dessin permettra à l'usager de commencer un projet **en mode solo** ou en **mode collaboratif.**

2.2.2. Interfaces matérielles

Pour les interfaces matérielles, nous faisons appel aux interfaces standards à tous les ordinateurs, donc clavier et souris pour notre client lourd, écran tactile et appareil photo pour le client léger. Pour nos effets audio, nous faisons appel aux cartes de son des ordinateurs pour le client lourd. Pour les écrans d'ordinateur, nous travaillons avec une résolution de 1920x1080 et pour la tablette on a une résolution de 1920x1200. Pour la partie réseau, étant donné que nous avons un mode collaboratif et un mode clavardage, nous faisons appel à l'entité réseau de la tablette ayant les spécifications de connectivité suivantes : 802.11 2.4G+5GHz, VHT80. Et à une configuration standard de carte réseau pour le client

lourd.

2.2.3. Interfaces logicielles

Comme base de développement de Polygram du côté client lourd, nous réutilisons le projet PolyDessin développé durant le cours LOG2990. Quant au client léger, il sera développé à partir de zéro. Le client lourd est développé avec le cadriciel Angular (Typescript comme langage) et Jasmine pour les tests. Pour le client léger, nous utilisons Kotlin pour le code source et JUnit pour les tests. Afin de générer un exécutable version application de bureau, nous faisons appel au cadriciel Electron. Polygram nécessite également la conception d'un serveur que nous développons en Typescript avec Mocha pour les tests. Le client lourd sera fonctionnel sur des ordinateurs ayant un système d'exploitation Windows 10 64 bits et le client léger sera fonctionnel sur des tablettes Galaxy Tab A 2019 avec Android Pie. Nous utiliserons également un système de base de données MongoDB. Le serveur sera hébergé sur la plateforme infonuagique Heroku. Pour les éditeurs de texte, le client lourd sera manipulé avec Visual Studio Code alors que le client léger sera manipulé avec Android Studio. Nous faisons également appel à plusieurs librairies, dont Angular Material, pour le client lourd.

2.2.4. Interfaces de communication

Le serveur de Polygram est hébergé sur une plateforme infonuagique. Ainsi, une connexion à internet est nécessaire pour l'utilisation de l'application tant au niveau du client lourd que sur celui du client léger.
Nos fonctionnalités vont faire appel à deux sortes de communication avec le serveur, soit par le biais des websockets en utilisant la librairie socket.io soit par des requêtes HTTP en utilisant le REST API. Dans le document protocole de communication nous détaillons ces fonctionnalités ainsi que le contenu des paquets qui circuleront entre les différents modules de notre application.
L'accès à internet peut se faire en utilisant le Wifi.

2.3. Contraintes générales

Le client lourd doit être en mesure de s'exécuter sur un ordinateur tournant avec le système d'exploitation Windows 10 et le client léger doit être en mesure de s'exécuter avec le système Android 9. En termes de mémoire, les clients lourd et léger ainsi que le serveur doivent avoir les spécifications détaillées dans la section 4.3 (Performance) du document. Les interactions avec le serveur doivent être fluides afin de garantir une expérience utilisateur qui se respecte sans effet saccadé. Le serveur doit être en mesure de gérer jusqu'à 16 connexions simultanées (4 utilisateurs x 4 dessins). Notre système doit être en mesure de stocker : les informations relatives à chaque compte utilisateur (description, statistiques, avatar, paramètres), les traits pour les dessins en **mode collaboratif**, les dessins des différents utilisateurs afin qu'ils soient affichés dans les **albums** ou sous forme de **story**, un registre des **albums** créés et le nombre de **mention j'aime** sur chaque dessin.
Pour l'espace de stockage, l'utilisateur devrait avoir 1Go de libre sur son ordinateur pour pouvoir installer polygram avec sa version de bureau, et 500 Mo sur sa tablette pour la version mobile.

2.4. Hypothèses et dépendances

Nous supposons que les utilisateurs de Polygram ont une connexion internet stable avec un débit qui ne mettra pas en péril le fonctionnement de notre application tant sur ordinateur que sur la tablette. Nous supposons également que les utilisateurs de la version de bureau de notre application sont sur Windows 10 et les utilisateurs de la version mobile sont sur Android 9.0.
Pour nos dépendances, nous utilisons les API des services d'hébergement de fichiers en ligne (Dropbox et Google Drive) et notre serveur sera hébergé sur la plateforme Heroku. Donc, vu que nous faisons appel à des API externes et un hébergeur nous ne pouvons pas contrôler si ceux-ci tombent en panne à un moment ce qui rendra quelques fonctionnalités de notre application non-disponibles.

3. Exigences fonctionnelles

3.1. Page de connexion

3.1.1. **[Essentielle]**
Le système doit permettre à l'utilisateur de se connecter avec son identifiant et son mot de passe.

3.1.2. **[Essentielle]**
Le système doit empêcher l'utilisateur de se connecter si l'identifiant ou le mot de passe sont invalides.

3.1.3. **[Essentielle]**
Le système doit informer l'utilisateur avec un message d'erreur clair si l'authentification échoue.

3.1.4. **[Essentielle]**
Le système doit permettre à l'utilisateur de changer son mot de passe s'il l'oublie.

3.1.5. **[Essentielle]**
Le système doit permettre à l'utilisateur de créer un nouveau compte en entrant son nom, prénom, identifiant, mot de passe et avatar.

3.1.6. **[Essentielle]**
Le système doit informer l'utilisateur avec un message d'erreur clair si la création d'un nouveau compte échoue.

3.1.7. **[Essentielle]**
Le système doit, lors de la création d'un compte, permettre à l'utilisateur de choisir son avatar parmi une liste prédéfinie.

3.1.8. **[Essentielle]**
Le client lourd doit permettre à l'utilisateur de créer son propre avatar en téléversant une image de son ordinateur vers l'application.

3.1.9. **[Essentielle]**
Le client léger doit permettre à l'utilisateur de créer son propre avatar en téléversant une photo prise à l'aide de l'appareil photo de la tablette.

3.2. Menu principal

3.2.1. **[Essentielle]**
Le système doit permettre à l'utilisateur de créer un nouveau dessin.

3.2.2. **[Essentielle]**
Le système doit permettre à l'utilisateur de continuer un dessin.

3.2.3. **[Essentielle]**
Le système doit permettre à l'utilisateur d'initier une partie collaborative de dessin.

3.2.4. **[Essentielle]**
Le système doit permettre à l'utilisateur de joindre une partie collaborative déjà existante.

3.2.5. **[Essentielle]**
Le système doit permettre à l'utilisateur d'accéder aux paramètres de son profil.

3.2.6. **[Essentielle]**
Le système doit permettre à l'utilisateur d'accéder au clavardage.

3.2.7**.** **[Essentielle]**
Le système doit permettre à l'utilisateur de se déconnecter.

3.2.8. **[Essentielle]**

Le système doit permettre à l'utilisateur de voir les stories (dessin du jour) des autres utilisateurs.

3.2.9. **[Essentielle]**
Le système doit permettre à l'utilisateur de voir ses propres stories.

3.2.10. **[Essentielle]**
Le système doit permettre à l'utilisateur de supprimer ses propres stories.

3.3. Album(s) de dessins

3.3.1. **[Essentielle]**
Le système doit afficher les albums de dessins.

3.3.2. **[Essentielle]**
Le système doit afficher les dessins contenus dans un album de dessin.

3.3.3. **[Essentielle]**
Le système doit afficher le nom du dessin, le propriétaire, la date de création et le nombre de collaborateurs actifs pour chaque dessin d'un album.

3.3.4. **[Essentielle]**
Le système doit permettre à l'utilisateur de créer un ou plusieurs albums.

3.3.5. **[Essentielle]**
Le système doit permettre à l'utilisateur de rejoindre des albums.

3.3.6. **[Essentielle]**
Le système doit permettre à l'utilisateur de créer son propre dessin dans les albums rejoints.

3.3.7. **[Essentielle]**
Le système doit demander à l'utilisateur les informations pertinentes au moment de la création de dessin (nom du dessin, album)

3.3.8. **[Essentielle]**
Le système doit permettre à l'utilisateur de modifier son propre dessin dans les albums rejoints.

3.3.9. **[Essentielle]**
Le système doit permettre à l'utilisateur de modifier les attributs de ses propres dessins. (nom du dessin, album)

3.3.10. **[Essentielle]**
Le système doit permettre à l'utilisateur de supprimer les attributs de ses propres dessins.

3.3.11. **[Essentielle]**
Le système doit permettre à l'utilisateur de visualiser l'édition de tous les dessins contenus dans les albums rejoints.

3.3.12. **[Essentielle]**
Le système doit permettre à l'utilisateur de supprimer des albums.

3.3.13**.** **[Essentielle]**
Le système doit permettre à l'utilisateur d'accéder automatiquement à un album public (album par défaut) au moment de son inscription à l'application.

3.3.14. **[Essentielle]**
Le système doit permettre à l'utilisateur de créer un album privé avec un nom et une description.

3.3.15. **[Essentielle]**
Le système doit permettre au propriétaire de l'album de modifier les attributs de l'album (nom et description)

3.3.16. **[Essentielle]**
Le système doit permettre au propriétaire de l'album de le supprimer.

3.3.17. **[Essentielle]**
Le système doit supprimer tous les dessins contenus dans un album lors de la suppression de l'album.

3.3.18. **[Essentielle]**
Le système doit permettre à l'utilisateur d'accéder à une liste des albums privés qu'il peut joindre.

3.3.19. **[Essentielle]**
Le système doit permettre à l'utilisateur de faire une demande pour rejoindre n'importe quel album privé dont il n'est pas actuellement membre.

3.3.20. **[Essentielle]**
Le système doit permettre à un membre existant de l'album privé d'approuver cette demande.

3.3.21. **[Essentielle]**
Le système doit permettre à un membre d'un album privé de le quitter à tout moment.

3.3.22. **[Essentielle]**
Le système doit permettre au membre ayant quitté l'album privé de faire une autre demande s'il souhaite le rejoindre à nouveau.

3.3.23. **[Essentielle]**
Le système doit empêcher l'utilisateur de voir les dessins d'un album privé s'il n'est pas membre de cet album.

3.3.24. **[Essentielle]**
Le système doit permettre au membre d'un album privé d'éditer les dessins contenus.

3.3.25. **[Essentielle]**
Le système doit permettre à l'utilisateur de créer un dessin à n'importe quel moment.

3.3.26. **[Essentielle]**
Le système doit permettre à l'utilisateur d'assigner un niveau de visibilité (public ou privé) du dessin au moment de la création.

3.3.27. **[Essentielle]**
Le système doit permettre au propriétaire d'un dessin de modifier le nom du dessin, son niveau de protection, ainsi que l'album dans lequel il se trouve dans le cas d'un dessin privé.

3.3.28. **[Essentielle]**
Le système doit permettre à l'utilisateur de filtrer la liste de dessins à l'aide d'un seul champ de texte regroupant tous les attributs d'utilisateur, de dessins et d'albums.

3.3.29. **[Essentielle]**
Le système doit permettre à l'utilisateur de spécifier si le filtrage est fait sur des albums, des dessins et des utilisateurs.

3.3.30. **[Essentielle]**
Le système doit permettre à l'utilisateur de faire une mention "j'aime" sur les dessins des autres utilisateurs.

3.4. Clavardage

3.4.1. **[Essentielle]**
Le système doit permettre l'accès à tout moment à l'interface de clavardage dès que l'utilisateur est connecté au serveur.

3.4.2. **[Essentielle]**
Le système doit permettre aux collaborateurs d'échanger des messages avant et pendant une édition collaborative.

3.4.3. **[Essentielle]**
Le système doit permettre à l'utilisateur de clavarder dans un mode fenêtré.

3.4.4. **[Souhaitable]**
Le système doit permettre à l'utilisateur de clavarder dans un mode intégré.

3.4.5. **[Souhaitable]**
Le système doit permettre à l'utilisateur d'alterner entre le mode fenêtré et le mode intégré.

3.4.6. **[Essentielle]**
Le système doit informer l'utilisateur lors de la réception d'un nouveau message.

3.4.7**.** **[Essentielle]**
Le système doit permettre à l'utilisateur de créer un ou plusieurs canaux de discussions.

3.4.8. **[Essentielle]**
Le système doit permettre à l'utilisateur de supprimer un ou plusieurs canaux de Discussion.

3.4.9. **[Essentielle]**
Le système doit permettre à l'utilisateur d'accéder à une liste de canaux existants.

3.4.10. **[Essentielle]**
Le système doit permettre à l'utilisateur de joindre un canal existant.

3.4.11. **[Essentielle]**
Le système doit permettre à l'utilisateur de filtrer la liste des canaux par le nom du canal.

3.4.12. **[Essentielle]**
Le système doit permettre à l'utilisateur de participer à plusieurs canaux de discussions en même temps.

3.4.13. **[Essentielle]**

Le système doit permettre à l'utilisateur de passer d'un canal à un autre.

3.4.14. **[Essentielle]**
Le système doit permettre à l'utilisateur de quitter un canal.

3.4.15. **[Essentielle]**
Le système doit permettre la création automatique d'un canal spécifique lors d'une séance de collaboration pour permettre aux collaborateurs de discuter entre eux.

3.4.16. **[Essentielle]**
Le système doit permettre à l'utilisateur de choisir un autre utilisateur et accéder aux paramètres de son profil.

3.4.17. **[Essentielle]**
Le système doit permettre à l'utilisateur de personnaliser sa fenêtre de clavardage.

3.4.17.1 **[Essentielle]**
Le système doit permettre à l'utilisateur de changer l'arrière-plan de sa fenêtre de clavardage.

3.4.17.2 **[Essentielle]**
Le système doit permettre à l'utilisateur de changer la couleur de texte de sa fenêtre de clavardage.

3.4.17.3 **[Essentielle]**
Le système doit permettre à l'utilisateur de changer la police et la taille du texte de sa fenêtre de clavardage.

3.4.18. **[Essentielle]**
Le système doit persister localement les données de personnalisations entre les sessions de l'utilisateur.

3.5. Profil des utilisateurs

3.5.1 **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter une liste de dessins sur le profil de l'utilisateur.

3.5.2. **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter le nombre de mentions "j'aime" sur un dessin.

3.5.3. **[Essentielle]**
Le système doit attribuer un badge différent à l'utilisateur en fonction du nombre de J'aime qu'il a obtenu : Débutant : 5 mentions J'aime, Intermédiaire : 25 mentions J'aime, Expert : 50 mentions J'aime, Artiste : 100 mentions "j'aime".

3.5.4. **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter le badge qui lui a été attribué via la page Profil.

3.5.5. **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter le badge qui a été attribué à un autre utilisateur via la page Profil de l'utilisateur concerné.

3.5.6. **[Essentielle]**

Le système doit permettre à l'utilisateur d'accéder en tout temps au profil d'un autre utilisateur à partir du canal de clavardage, d'une story ou d'un album.

3.5.7. **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter le nombre de dessins auxquels un utilisateur a collaboré.

3.5.8. **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter le nombre de dessins auxquels un utilisateur est un auteur.

3.5.9. **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter le nombre d'équipes collaboratives auxquelles un utilisateur appartient.

3.5.10. **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter le temps moyen de collaboration d'un utilisateur.

3.5.11. **[Essentielle]**
Le système doit permettre à l'utilisateur de consulter le temps total passé par un utilisateur à collaborer.

3.6. Paramètres du compte utilisateur

3.6.1. **[Essentielle]**
Le système doit permettre à l'utilisateur de changer son identifiant via la page Paramètres de compte.

3.6.2. **[Essentielle]**
Le système doit permettre à l'utilisateur de changer son mot de passe via la page Paramètres de compte.

3.6.3. **[Essentielle]**
Le système doit permettre à l'utilisateur de changer son avatar via la page Paramètres de compte.

3.6.4. **[Essentielle]**
Le système doit permettre à l'utilisateur de téléverser son propre avatar sur sa page Paramètres de compte.

3.6.5. **[Essentielle]**
Le système doit permettre à l'utilisateur d'activer les effets sonores via la page Paramètres de compte.

3.6.6. **[Essentielle]**
Le système doit permettre à l'utilisateur d'activer les effets visuels via la page Paramètres de compte.

3.6.7. **[Essentielle]**
Le système doit permettre à l'utilisateur de désactiver les effets sonores via la page Paramètres de compte.

3.6.8. **[Essentielle]**
Le système doit permettre à l'utilisateur de désactiver les effets visuels via la page

Paramètres de compte.

3.6.9. **[Essentielle]**
Le système doit permettre à l'utilisateur de sélectionner un son parmi une liste de sons prédéfinis via la page Paramètres de compte.

3.6.10. **[Essentielle]**
Le système doit permettre à l'utilisateur d'ajouter une description pour son profil.

3.7. Éditeur de dessin

3.7.1. **[Essentielle]**
Le système doit permettre à l'utilisateur de dessiner en main libre (outil Crayon).

3.7.2. **[Essentielle]**
Le système doit permettre à l'utilisateur de dessiner des formes géométriques (outil Forme).

3.7.3. **[Essentielle]**
Le système doit permettre à l'utilisateur de dessiner des rectangles

3.7.4 **[Essentielle]**
Le système doit permettre à l'utilisateur de dessiner des ellipses.

3.7.5. **[Essentielle]**
Le système doit permettre à l'utilisateur de sélectionner une partie de son dessin (outil Sélection).

3.7.6. **[Essentielle]**
Le système doit permettre à l'utilisateur d'effectuer une translation de sa sélection (haut, bas, gauche, droite).

3.7.7. **[Essentielle]**
Le système doit permettre à l'utilisateur de redimensionner sa sélection.

3.7.8.. **[Essentielle]**
Le système doit permettre à l'utilisateur de supprimer sa sélection.

3.7.9. **[Essentielle]**
Le système doit permettre à l'utilisateur de téléverser ses propres dessins sur son Google Drive.

3.7.10. **[Essentielle]**
Le système doit permettre à l'utilisateur de téléverser ses propres dessins sur son compte Dropbox.

3.7.11. **[Essentielle]**
Étampe améliorée : le système doit permettre à l'utilisateur de réutiliser d'anciens dessins des utilisateurs participant à la session de collaboration.

3.7.12.. **[Essentielle]**
Le système doit permettre à l'utilisateur de redimensionner l'étampe.

3.7.13. **[Essentielle]**

Le système doit permettre à l'utilisateur d'effectuer une rotation de l'étampe.

3.7.14. **[Essentielle]**
Le client léger doit permettre à l'utilisateur de publier une Story, c'est-à-dire publier un dessin du jour pour 24 heures sans archivage.

3.7.15. **[Essentielle]**
Le client léger doit permettre à l'utilisateur de prendre une photo avec la caméra de l'appareil.

3.7.16. **[Essentielle]**
Le client léger doit permettre à l'utilisateur de téléverser directement la photo prise avec la caméra dans l'éditeur de dessin.

3.7.17. **[Essentielle]**
Le client léger doit permettre à l'utilisateur de modifier la photo prise avec la caméra.

3.7.18. **[Essentielle]**
Le client léger doit inclure la géolocalisation (ville, pays, heure au format HH:MM) de l'utilisateur lors de la création d'un nouveau dessin.

3.8. Mode d'édition

3.8.1. **[Essentielle]**
Le système doit être en synchronisation simple, c'est-à-dire partager les actions d'édition à leur terme.

3.9. Effets visuels et sonores

3.9.1. **[Essentielle]**
Le système doit jouer une rétroaction sonore à la sélection d'un outil dans la barre d'outils de l'éditeur de dessin.

3.9.2. **[Essentielle]**
Le système doit jouer une rétroaction sonore lorsqu'un utilisateur tape une entrée invalide dans un champ de texte.

3.9.3. **[Essentielle]**
Le système doit générer un "wizz" lorsque l'utilisateur tape une entrée invalide dans un champ de texte.

3.9.4. **[Essentielle]**
Le système doit jouer un son lorsque l'utilisateur publie une story.

3.9.5. **[Essentielle]**
Le système doit jouer une rétroaction sonore lors d'un changement de page dans l'application.

3.9.5. **[Essentielle]**
Le système doit jouer une courte animation de transition lors d'un changement de page dans l'application.

4. Exigences non fonctionnelles

4.1. Utilisabilité

4.1.1. L'utilisateur doit être capable de se créer un compte en moins de 5 minutes.

4.1.2. L'utilisateur doit pouvoir se connecter à son profil en moins de 2 minutes.

4.1.3. L'utilisateur doit être capable de joindre une session de dessin collaborative en moins d'une minute suivant la connexion à son profil.

4.1.4. L'utilisateur doit être capable de créer un dessin en moins de 2 minutes suivant la connexion à son profil.

4.1.5. L'utilisateur doit être capable d'envoyer un message dans le canal de son choix en moins d'une minute suivant la connexion à son profil.

4.1.6. L'utilisateur doit être en mesure de se déconnecter de son profil en moins de 5 secondes, peu importe ce qu'il est en train de faire sur l'application.

4.1.7. Un utilisateur doit être capable d'accéder au profil de l'auteur du dessin qu'il est en train de regarder en un clic.

4.1.8. Un utilisateur doit être capable d'accéder au profil d'un clavardeur en un clic lors du visionnement d'un des messages de celui-ci.

4.1.9. Les options disponibles du menu principal doivent être distinguables les unes des autres.

4.1.9.1 Toutes les options doivent être présentées sur une même vue sans avoir à faire défiler la page.

4.1.9.2 Les boutons d'options doivent être représentés par des icônes permettant à l'utilisateur de facilement les différencier.

4.2. Fiabilité

4.2.1. Le serveur doit être en ligne 98% du temps.

4.2.2. Le temps moyen entre deux pannes doit être de 1 semaine.

4.2.3. Le temps moyen jusqu'à la réparation est d'une journée de travail (8 heures).

4.2.4. Avoir moins de 10 bogues mineurs par semaine.

Bogue mineur : bogue affectant les fonctionnalités de l'application sans compromettre sa disponibilité.

Avoir moins d'un bogue majeur par semaine.
Bogue majeur : bogue qui affecte la disponibilité de l'application.

4.3. Performance

4.3.1. L'application doit permettre à une session de collaboration d'avoir jusqu'à 4 collaborateurs.

4.3.2. Le serveur doit être en mesure de présenter l'action d'un dessinateur à ses collaborateurs sans latence (pas d'effet saccadé).

4.3.3. Le serveur doit être en mesure de supporter au moins 16 sessions de collaboration simultanément (4 utilisateurs x 4 dessins).

4.3.4. Le serveur doit être en mesure de soutenir la connexion d'au moins 16 utilisateurs.

4.3.5. Le système doit être en mesure d'authentifier l'identifiant et le mot de passe en moins de 2 secondes.

4.3.6. Le système doit être capable de connecter un utilisateur à une session de collaboration en moins de 2 secondes.

4.3.7. Le système doit être capable de déconnecter un utilisateur à une session de collaboration en moins de 2 secondes.

4.3.8. Le système doit être capable de communiquer un message d'utilisateur en moins de 2 secondes.

4.3.9. Le système doit être capable d'appliquer un changement d'accessibilité (privé/publique) en moins de 2 secondes.

4.3.10. Le client léger doit être capable d'accéder à la caméra de l'utilisateur en moins de 5 secondes.

4.3.11. La base de données doit avoir un temps de réponse inférieur à 50 ms.

4.3.12. Le client lourd doit nécessiter au plus 1 Go d'espace sur le disque.

4.3.13. Le client lourd doit avoir un taux de rafraîchissement d'au moins 60 images par seconde.

4.3.14. Le client lourd doit consommer au maximum de 500 Mo de mémoire vive.

4.3.15. Le client léger doit nécessiter au plus 500 Mo d'espace sur le disque à la tablette.

4.3.16. Le client léger doit avoir un taux de rafraîchissement d'au moins 30 images par seconde.

4.3.17. Le client léger doit consommer au maximum 300 Mo de mémoire vive.

4.4. Maintenabilité

4.4.1. Les fragments de code complexes doivent être expliqués par des commentaires.

4.4.2. Le code du client léger doit suivre les conventions de nommage de Kotlin :

4.4.2.1. Le nom des paquetages doit être minuscule.

4.4.2.2. Le nom des classes doit commencer par une majuscule et continuer en camelCase.

4.4.2.3. Les constantes doivent être en majuscule et séparées par un *underscore*.

4.4.2.4. Le nom des fonctions doit commencer par une minuscule et continuer en camelCase.

4.4.3. Le code du client lourd doit suivre les conventions de nommage de TypeScript :

4.4.3.1. Le nom de type et les valeurs d'énumérations doivent être en PascalCase.

4.4.3.2. Le nom des fonctions, propriétés et des variables doivent être en camelCase

4.4.3.3. Les constantes doivent être en majuscule et séparées par un *underscore*.

4.4.3.4. Le nom des fonctions doit commencer par une minuscule et continuer en camelCase.

4.4.3.5. Le nom de balises des composants Angular doit être en kebab-case.

4.4.4. Le code du serveur doit suivre les conventions de nommage et les standards de TypeScript :

4.4.4.1. Le nom de type et les valeurs d'énumérations doivent être en PascalCase.

4.4.4.2. Le nom des fonctions, propriétés et des variables doivent être en camelCase

4.4.4.3. Les constantes doivent être en majuscule et séparées par un *underscore*.

4.4.4.4. Le nom des fonctions doit commencer par une minuscule et continuer en camelCase.

4.4.5. Des commentaires doivent être écrits en haut des fonctions et des classes du côté serveur.

4.4.6. Le code du côté serveur, client léger et client lourd doit être écrit en anglais.

4.4.7. La description des "commit" doit être rédigée en anglais.

4.5. Contraintes de conception

4.5.1. Le client léger doit être développé dans le langage de programmation Kotlin.

4.5.2. Le client léger doit être fonctionnel au minimum sous la version Android 9.0 (Pie).

4.5.2. Le client lourd doit être développé dans le langage de programmation TypeScript.

4.5.3. Le client lourd doit être développé avec le cadriciel Angular.

4.5.3. Le client lourd doit être converti en une application de bureau avec Electron.

4.5.4. Le client lourd doit être un exécutable.

4.5.5. Le client lourd doit être fonctionnel sous Windows 10.

4.5.5. Le serveur doit être développé dans le langage de programmation TypeScript.

4.5.6. Le serveur doit être fait à l'aide du cadriciel Node.js.

4.5.7. Le cycle de développement du logiciel sera la méthode Agile (Scrum).

4.5.8. Les données seront entreposées dans une base de données MongoDB

4.5.9 La base de données doit être NoSQL

4.6. Sécurité

4.6.1. Le système doit, lors de la saisie, masquer le mot de passe à la connexion.

4.6.2 Le système fera appel à deux mécanismes d'encryption pour les mots de passe pour plus de sécurité : **hashage et salt**.

4.6.2. Le système doit sécuriser le stockage de données : seul l'administrateur de la base de données aura accès.

4.6.3. La communication entre les clients et le serveur doit être chiffrée à l'aide du protocole HTTPS.