205

Fais-moi un dessin
Plan de projet

Version 3.1

Historique des révisions

Date	Version	Description	Auteur
2021-02-06	1.0	Première ébauche	Augustin Bouchard
2021-02-08	1.1	Ajout de l'échéancier	Augustin Bouchard
2021-02-10	1.2	Ajout des risques	Félix Dumont
2021-02-18	1.3	Entente contractuelle proposée	Augustin Bouchard
2021-02-19	2.0	Version révisée pour la remise de l'appel d'offre	Augustin Bouchard
2021-04-15	3.0	Première correction de l'appel d'offre	Augustin Bouchard
2021-04-19	3.1	Révision pour la remise du projet	Augustin Bouchard

Table des matières

1. Introduction **4**

2. Énoncé des travaux **4**

2.1. Solution proposée 4

2.2. Hypothèses et contraintes 4

2.3. Biens livrables du projet 4

3. Gestion et suivi de l'avancement **4**

3.1. Gestion des exigences 5

3.2. Contrôle de la qualité 5

3.3. Gestion de risque 5

3.4. Gestion de configuration 8

4. Échéancier du projet **9**

5. Équipe de développement **10**

6. Entente contractuelle proposée **12**

Plan de projet

1. Introduction

Le présent document contient les informations nécessaires pour suivre le développement du projet *Fais-moi un dessin.* La première section porte sur le travail à faire. Quelques solutions, hypothèses et contraintes y seront discutées avant d'énumérer les biens livrables du projet. La deuxième section porte sur le suivi du projet au niveau des exigences, de la qualité, des risques et de la gestion des configurations. La troisième section s'attarde plutôt au suivi temporel du projet tandis que la quatrième porte sur les expertises des membres de l'équipe de développement. Finalement, il sera question de suggérer une entente contractuelle.

2. Énoncé des travaux

2.1. Solution proposée

Le projet *Fais-moi un dessin* sera composé de deux applications, un serveur et une base de données. Le projet consiste à créer une application similaire à skribbl.io permettant aux utilisateurs de jouer ensemble. Les parties de *Fais-moi un dessin* consistent à regarder un joueur dessiner et deviner le mot qu'il essaie de représenter. D'autres fonctionnalités supplémentaires seront implémentées et elles sont disponibles dans le SRS. L'équipe propose le développement d'une application développée avec le framework Angular, conçue pour être un exécutable Windows avec l'outil Electron (client-lourd). Cette application sera une évolution du logiciel *Polydessin* développé lors du projet 2. En parallèle, une application compatible avec Android 9 et plus sera développée sur Android Studio avec le langage de programmation Kotlin. Les deux applications communiqueront entre elles grâce à un serveur en Node.js déployé sur Heroku et une base de données déployée sur Microsoft Azure.

2.2. Hypothèses et contraintes

Le projet se basera sur le client-lourd PolyDessin qui a été développé il y a un an avec Angular. L'équipe assumera que la base possède une structure fiable et testée. Le client-léger, le serveur et la base de données seront développés à partir de zéro. Le format de l'application sera adapté pour une tablette Samsung Galaxy Tab A 2019. Ainsi, on suppose que le client-léger sera testé avec cette dernière et que l'interface n'aura pas à être adaptée pour d'autres formats. On suppose aussi que le client-lourd devra être adapté pour une surface égale, sinon plus grande que celle de la tablette.

Pour porter le projet à terme, l'équipe sera composée de 6 étudiants en ingénierie. Chacun des membres de l'équipe devra être en possession d'un ordinateur assez puissant pour développer les différents logiciels désirés. Chacun des membres de l'équipe devra aussi posséder une connexion internet haute-vitesse acceptable pour faciliter la communication et l'échange d'information. L'équipe devra aussi être en mesure de se rencontrer en personnes toutes les deux semaines durant le projet.

L'équipe devra être en mesure de remettre un prototype de communication le 19 février 2021.

L'équipe devra être en mesure de remettre le produit final le 19 avril 2021.

2.3. Biens livrables du projet

L'équipe devra rédiger un SRS, un plan de projet (ce document), une liste d'exigences, un document d'architecture logicielle et un protocole de communication pour la réponse à l'appel d'offres. Ces documents devront être remis le 19 février 2021.
L'équipe devra par la suite mettre à jour les artéfacts remis, développer un plan de tests et remplir un document de résultats de tests.
Tous les documents devront être remis avec le produit final le 19 avril 2021.

3. Gestion et suivi de l'avancement

3.1. Gestion des exigences

Les exigences essentielles et souhaitables ont été sélectionnées en équipe. Certaines révisions ont été faites à plusieurs reprises en équipe de trois.
Les exigences essentielles seront toutes réalisées. En cas de problème majeur, une rencontre d'équipe aura lieu pour discuter du problème et une solution sera prise par consensus.
Les exigences souhaitables seront sélectionnées lors des réunions hebdomadaires des équipes une fois la majorité des exigences essentielles réalisées. Lors d'un problème avec une exigence, le responsable devra communiquer ce dernier dans un salon virtuel nommé problème. L'équipe sera responsable de consulter ce salon régulièrement et d'aider les personnes bloquées. La formation de paires sera recommandée. Si le problème persiste, l'exigence souhaitable sera amenée à la réunion suivante, ou l'équipe pourra prendre une décision par consensus. Il sera possible à ce moment de consulter le client pour discuter des problèmes et des solutions disponibles. L'équipe pourra aussi changer ses priorités dans le cas des exigences souhaitables.

3.2. Contrôle de la qualité

Le code est toujours développé sur les branches propres aux fonctionnalités ajoutées. Lorsque vient le temps de fusionner la fonctionnalité terminée avec la branche de développement (dev), l'auteur se doit de faire une *merge request*. Cette dernière doit être autorisée par au moins deux collègues. Ces derniers pourront indiquer les corrections à apporter avant d'accepter la requête.
Si jamais une erreur se glisse sur la branche de développement, ou bien s'il faut apporter des changements, la personne déléguée devra ouvrir une nouvelle branche et apporter les correctifs sur cette branche et le même processus de merge request sera nécessaire une fois les correctifs complétés.
Suivant cette logique, les correctifs seront apportés en fin de développement d'une fonctionnalité. Lors d'un problème sur la branche de développement, il sera impératif de résoudre le problème le plus tôt possible puisque cette branche se retrouve au cœur des nouvelles fonctionnalités.
Un plan de test sera rédigé. Toutes les stratégies de test s'y retrouvent. Il sera important de remplir un document de résultats de tests logiciels et de corriger tous les tests qui ont échoué.

3.3. Gestion de risque

La description des risques suit la convention suivante :

- Ampleur : sur une échelle de 1 à 10, 10 étant le risque le plus élevé. Cette analyse est basée sur la probabilité d'occurrence du risque, ainsi que ses impacts.
- Description : une description textuelle du risque ainsi que les problèmes attendus.
- Impact : échelle définissant la portée du risque
    - C – critique (affecte le projet en entier)
    - E – élevé (affecte les fonctionnalités principales du système)
    - M – moyen (devrait être maîtrisable en appliquant une stratégie d'atténuation adéquate)
    - F – faible (l'acceptation du risque est une stratégie envisageable)
- Facteurs : aspects (**métriques**) du système pouvant être compromis.
- Stratégie de gestion : mesures à prendre afin de gérer le risque.

R1 - Déconnexion				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**

6	Mauvaise déconnexion: on utilise un booléen dans une table dans notre BD pour déterminer si un utilisateur est connecté ou non. SI un utilisateur est connecté, il ne peut pas se connecter avec un autre appareil. Ainsi, si l'application se ferme sans envoyer de querie de déconnexion, l'utilisateur ne pourra pas se connecter lors de sa prochaine connexion.	C	Cohérence des données	Gérer tous les moyens possibles d'arrêts et de déconnexion des clients.

R2 - Sécurité du mot de passe				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
3	Les mots de passe ne sont pas cryptés dans la BD, donc une fuite possible d'information personnelle est possible.	M	Cryptage des données	Crypter les mots de passe dans la base de données.

R3 - Compatibilité serveur				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
9	Le serveur est déployé sur heroku et la base de données sur Microsoft Azure. Si les connexions avec le serveur et la base de données ne fonctionnent pas, il faudra changer toute l'architecture et le type de serveur.	C	Connexion avec le serveur et la BD	Faire des tests de connexion sur des requêtes faciles. Faire ces tests avant d'entamer le projet avec Heroku et Microsoft Azure.

R4 - Matières et technologies inconnues				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
7	L'équipe devra apprendre et utiliser de nouvelles technologies. Certaines estimations de temps pourraient être sous-estimées et certaines tâches pourraient s'avérer très ardues.	E	Temps	Il faudra s'assurer de faire les recherches nécessaires avant de se lancer dans une tâche. Si l'empleure de cette dernière semble trop importante, il est possible de partager ses craintes avec l'équipe et de prendre une décision.

R5 - Cohérence entre les clients				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
9	Mauvaise communication d'équipe en lien avec les interfaces utilisateur: L'équipe Android et Web devront développer dans des projets différents, mais ils devront avoir une interface utilisateur et offrir une expérience utilisateur similaire. Dans le cas contraire, un utilisateur pourrait se retrouver perdu dans les applications.	C	Expérience utilisateur	Il faudra être très minutieux dans la communication entre équipe et établir des plan de travail très précis lorsqu'on entame une nouvelle fonctionnalité du projet. Nous utiliserons Figma pour créer des interfaces avant d'entamer le codage.

9	Mauvaise communication d'équipe en lien avec les interfaces utilisateur: L'équipe Android et Web devront développer dans des projets différents, mais ils devront communiquer avec le même serveur, ainsi il est possible qu'ils utilisent des interfaces api différentes et qu'un des deux doivent refactor leur code. Des erreurs de ce genre représentent des grosses pertes de temps.	C	Expérience utilisateur	Il faudra être très minutieux dans la communication entre équipe et établir des plan de travail très précis lorsqu'on entame une nouvelle fonctionnalité du projet.

R6 - Mauvaise gestion du temps				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
7	L'équipe complète doit être à l'ordre et finir le projet pour le 19 avril 2021. Toutes les exigences devront être implémentées. Des problèmes pourraient résulter en l'insatisfaction du client.	C	Temps	Il faudra se donner des buts et des objectifs. Nous sépareront le projet en sprints de deux semaines. Chaque personne sera responsable de réaliser leur portion du sprint.

R7 - Covid-19				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
4	L'équipe doit se rencontrer à plusieurs reprises en présentiel. Il faut limiter les risques de contamination. En cas de maladie, un développeur pourrait se retrouver incapable d'avancer ses parties et ainsi retarder le projet.	E	Ressources humaines	II faudra respecter les normes gouvernementales et s'assurer que tous les membres de l'équipe portent le masque et se lavent les mains.

R8 - Internet				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
1	L'équipe travaille majoritairement à distance. En cas de panne générale d'internet (exemple vidéotron), certains membres pourraient se retrouver bloqués.	F	Ressources humaines pendant la panne	Il faut avoir localement certaines tâches plus faciles. En cas de blocage, chaque personne est responsable de se trouver une tâche alternative.

R9 - Opération dents de sagesse				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
5	Un membre de notre équipe se fait opérer pour l'extraction de ses dents de sagesse durant la semaine avant la remise du prototype. L'opération peut mener à des complications dans les jours suivants l'opération. Un membre de moins affecterait l'efficacité du projet.	M	Ressources humaines	Ce membre devra effectuer ses tâches les plus critiques avant son opération. De plus, il devra bien suivre les indications post-opératoires de son dentiste pour prévenir toute complication et infection.

R10 - Intégration

Ampleur	Description	Impact	Facteurs	Stratégie de gestion
6	L'équipe travaille sur plusieurs branches en parallèle. Lors des fusions des branches, il est possible que les modifications d'un membre de l'équipe ne soient pas compatibles avec celles en fusion. Des conflits, des problèmes pourraient survenir ou pire, quelqu'un pourrait briser le logiciel.	M	Temps	Il faut séparer les tâches intelligemment. Les fonctionnalités développées par chacun ne devraient pas impacter celles d'une autre personne. De plus, chaque personne est responsable de fusionner la branche dev sur sa branche régulièrement. La communication est la clé pour éviter des problèmes d'intégration.

3.4. Gestion de configuration

Tous les problèmes devront être rapportés dès leur détection dans un salon virtuel nommé problèmes. Se référer à la section 3.2 pour les problèmes de développement. Si une résolution de conflit induit un changement nécessaire dans quelconque artéfact ou si une modification d'artéfact est faite dans tout autres conditions, la personne chargée de la modification se doit de mettre à jour la version du document et d'annoncer sa modification à la prochaine rencontre.
Les versions seront gérées de la façon suivante: Version <majeur>.<mineur>
La version majeure devra être incrémentée après toute révision finale d'équipe. À chaque incrémentation majeure, la version mineure doit être remise à zéro. Ce sera la responsabilité de l'équipe de faire une révision à chaque 2 sprints (soit à la fin des sprints 2, 4 et 6).
La version mineure devra être incrémentée après chaque modification du document.
Les artéfacts seront nommés SRS, Plan de projet, Architecture logicielle, Protocole de communication, Plan de tests logiciels et Résultats de tests logiciels.
Les noms des documents resteront tels quels. Ils ne sont pas sujet à changement.

Le code suit un système de gestion de version similaire aux artéfacts. Les versions seront gérées de la façon suivante: Version <majeur>.<mineur>.<correctif>.
La version majeure est incrémentée à chaque remise. Elle implique de remettre les versions mineure et correctif à 0.
La version mineure est incrémentée à chaque fonctionnalité ajoutée. Elle implique de remettre la version de correctif à 0.
La version de correctif est incrémentée à chaque correctif ajouté au logiciel.

Les planifications hebdomadaires suivent la méthode SCRUM. Chaque membre de l'équipe explique sa tâche encore, son développement et ses blocages.Une fois le tour de table terminé, l'équipe peut donner son avis sur les blocages rencontrés ou se séparer en sous-groupes. Avant chaque sprint de 2 semaines, le gestionnaire planifie le sprint en préparant les tâches sur Jira. Lors de la prochaine rencontre, le gestionnaire présente sa planification et ajuste les tâches selon les avancements et les opinions des membres de l'équipe.

La stratégie git sera la suivante:
Le code des fonctionnalités en cours de développement doivent être publié sur GitLab dans une branche nommée <#numéro ticket jira>-<fonctionnalité> ex : 5-architecture-serveur
Les messages de commit Git devront suivre le format suivant [initiales-nom](type): <commentaire>
avec comme types :
dev - développement
deb - debug
ref - refactoring
aut - autre
tes - test
ex : [SA](dev): Modification UI menu principal. Ajout du bouton pour fermer l'application.
Une fois une fonctionnalité terminée, le développeur devra faire une *merge request* sur gitlab. Le reste de l'équipe est responsable de réviser le code et de demander des changements au besoin. Une fois la *merge request* acceptée, le développeur pourra merge sa branche sur dev directement avec gitlab.
Les remises de sprint devront être faites sur la branche dev
La remise finale devront être fait sur la branche master

4. Échéancier du projet

Le projet sera divisé en sprints de 2 semaines. Au moment la réponse à l'appel d'offres, le deuxième sprint sera déjà terminé.

Sprint	Exigences/Artéfacts	Temps	Date
1	• SRS • Liste d'exigences • Nettoyage du code Angular • Migration vers Electron • Recherches sur les architectures possibles • Création de l'application Android • Création de la base de données • Création du serveur	32 heures 32 heures 24 heures 8 heures 32 heures 4 heures 8 heures 8 heures	Du 20 janvier 2021 Au 2 février 2021
Jalon	Premières communications entre les clients fonctionnelles		
2	• Architecture du logiciel • Plan de projet • Protocole de communication • Gestion des connexions • Clavardage - communication entre deux utilisateurs	16 heures 16 heures 16 heures 48 heures 48 heures	Du 3 février 2021 Au 16 février 2021
Jalon	Finir les exigences pour la réponse à l'appel d'offre		
Remise	Réponse à l'appel d'offres		19 février 2021
3	• Clavardage - Intégration • Clavardage - Canaux de discussion • Clavardage - Censure • Profil utilisateur et historique • Outils de dessin Android • Personnalité des joueurs virtuels • Création d'une paire mot-image - Assistée 1	4 heures 40 heures 8 heures 16 heures 16 heures 24 heures 16 heures	Du 17 février 2021 Au 2 mars 2021
Jalon	Finir les fonctionnalités nécessaires aux parties		
4	• Rédaction du plan de tests logiciel • Effets visuels et sonores (Lourd) • Effets visuels et sonores (Léger) • Lobby - Créer, rejoindre et quitter (Serveur) • Lobby - Créer, rejoindre et quitter (Lourd) • Lobby - Créer, rejoindre et quitter (Léger) • Construction d'un jeu - Indices • Construction d'un jeu - Classique • Leaderboard • Pouces	16 heures 8 heures 8 heures 16 heures 16 heures 16 heures 8 heures 48 heures 16 heures 16 heures	Du 3 mars 2021 Au 16 mars 2021
Jalon	Finir la partie classique		

5	• Tutoriel - non interactif (Lourd) • Tutoriel - non interactif (Léger) • Construction d'un jeu - Sprint Solo • Construction d'un jeu - Sprint Coop • Création d'une paire mot-image - Assistée 2	8 heures 8 heures 32 heures 16 heures 24 heures	Du 17 mars 2021 Au 30 mars 2021
Jalon	Finir tous les modes de jeu		
6	• Construction d'un jeu - Battle Royal • Révision des artéfacts • Rédaction des résultats de tests • Corriger les bogues (Serveur) • Corriger les bogues (Lourd) • Corriger les bogues (Léger) • Ajustements UI-UX (Lourd) • Ajustements UI-UX (Léger)	32 heures 32 heures 16 heures 24 heures 16 heures 24 heures 8 heures 8 heures	Du 31 mars 2021 Au 19 avril 2021
Jalon	Finir le projet		
Remise	Révision du projet final	32 heures	19 avril 2021
Total		864 heures	

Chaque sprint implique 4 heures de rencontre par personne, donc 24 heures-personne
Chaque sprint implique 2 heures de gestion de projet par le gestionnaire
Légende:

- Exigences essentielles
- Exigences souhaitables (sujet à changement) (voir ci-dessous pour les choix)

Le tableau suivant représente les exigences souhaitables qui ne seront pas implémentées avec leur estimation de temps.

Exigences souhaitables	Temps
• Tutoriel interactif • Construction d'un jeu - Mode spectateur • Connexion Google • Liste d'amis • Authentification en deux étapes	16 heures 8 heures 16 heures 32 heures 16 heures

5. Équipe de développement

Tous les membres de l'équipe sont des étudiants en génie logiciel à l'école Polytechnique de Montréal. Ayant tous un cursus universitaire similaire, tous les membres de l'équipe devraient être à l'aise avec les technologies suivantes:

- Angular
- HTML, CSS, JavaScript, TypeScript
- Nodejs
- NoSQL, SQL, PostgreSQL

Augustin Bouchard:
Connaissances additionnelles:

- Migration de projet c#
- Automatiser le déploiement logiciel du packaging

Responsabilités:
- Rédaction de documents
- Gestion de projet
- Frontend Android
- Design UI

Félix Dumont:
Connaissance additionnelles:
- Communication entre clients(sockets) en C#
- Gestion base de données

Responsabilités:
- Base de donnée
- Backend Android

Guilhem Dubois
Connaissances additionnelles:
- Angular
- Langage et requêtes SQL

Responsabilités:
- Base de données
- Backend Electron
- Frontend Electron
- Rédaction de diagrammes UML
- Gardien du temps (ramène à l'ordre si on dépasse les temps)

Julien Desalliers:
Connaissance additionnelles:
- Développement android
- Kotlin

Responsabilités:
- Backend Android
- Rédaction de diagrammes UML
- Soutient

Mark Weber-Sadler:
Connaissances additionnelles:
- Angular
- SQL
- Express

Responsabilités:
- Serveur
- Recherchiste, trouve les options possibles et amène des connaissances
- Aide-mémoire (Rappel les objectifs donnés pour la semaine)
- Sceptique (Cherche le questionnement et s'assure que les solutions sont optimales)

Simon Ayotte:
Connaissances additionnelles:
- Automatisation de processus
- Communication entre clients avec WebSockets

Responsabilités:
- Serveur
- Rédaction et expert en communication
- Sceptique (Cherche le questionnement et s'assure que les solutions sont optimales)

6. Entente contractuelle proposée

L'équipe se base sur un budget ventilé en fonction des lots de travail et des taux suivants :

- développeur : 100$/h ;
- gestionnaire de projet : 125$/h.

Le gestionnaire rédige tous les documents, il s'occupe des planifications et il fait partie des rencontres bi-hebdomadaires de l'équipe.
Gestionnaire: 176 heures-personne de rédaction + 36 heures-personne de gestion = 212 heures-personne
Développeurs: 688 heures personne de développement + 120 heures-personne de rencontre = 808 heures-personne

Selon les estimations préparées dans l'échéancier du projet, le total est estimé à 107 300$ + tx.

Étant donné que l'équipe se doit de livrer le produit final le 19 avril, qu'on connait la demande, les spécifications détaillées et les exigences et qu'il y a un suivi minimal des travaux, l'équipe préconise un contrat livraison clé en main.

Polytechnique doit approuver une liste d'exigences. Ces exigences ne pourront pas être changées après l'approbation à moins d'en discuter avec le client.
Le produit final doit être livré pour le 19 avril 2021.
Le prix demandé sera ferme et ne pourra pas être changé après l'entente.