Équipe 3

Turb0_P41nt_F1v3_Th0u54nd.exe
Plan de projet

Version 1.5

Historique des révisions

Date	Version	Description	Auteur
2019-08-29	1.0	Première ébauche	Marie-Elaine Bérubé Justine Lambert
2019-09-01	1.1	Ajout de la partie 2 et 5	Justine Lambert
2019-09-03	1.2	Ajout des membres dans la partie 5, début de l'échéancier et contrôle de la qualité	Tous
2019-09-05	1.3	Entête contractuelle, échéancier et partie 3	Marie-Elaine Bérubé Olivier Naud-Dulude
2019-09-08	1.4	Ajout de l'échéancier	Marie-Elaine Bérubé
2019-09-10	1.5	Révision	Justine Lambert

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 5

3. Gestion et suivi de l'avancement 5

3.1. Gestion des exigences 5

3.2. Contrôle de la qualité 5

3.3. Gestion de risque 6

3.4. Gestion de configuration 7

4. Échéancier du projet 8

5. Équipe de développement 10

6. Entente contractuelle proposée 12

Plan de projet

1. Introduction

Ce document présente le plan de projet du logiciel PolyPaint. Il énonce d'abord la solution proposée en tenant compte des hypothèses et des contraintes à considérer pour les livrables du projet. Il décrit ensuite les moyens mis en place pour gérer la qualité et l'avancement du projet, puis il décompose le projet en un échéancier des tâches à réaliser pour les différents livrables avec une estimation du temps requis pour ceux-ci. Les rôles des membres y sont également définis.

2. Énoncé des travaux

2.1. Solution proposée

Nous proposons une solution qui fait évoluer le logiciel PolyPaint afin de le transformer en jeu de type *Fais-moi un dessin*. Ce jeu consiste à faire deviner un mot au joueur en lui dessinant l'objet correspondant. Nous offrons donc d'ajouter plusieurs fonctionnalités, parmi lesquelles on retrouve notamment celle d'un mode multijoueur en réseau pour des parties comptant jusqu'à quatre joueurs. Un clavardage sera également disponible pour permettre aux joueurs de communiquer entre eux. Ensuite, nous proposons d'implémenter des joueurs virtuels pour l'application, qui pourront tracer un dessin et ainsi jouer avec des joueurs réels lorsque le nombre de ceux-ci est impair. Nous souhaitons également permettre à chaque joueur de se connecter au jeu par le biais d'un profil utilisateur, ainsi que proposer un lobby de jeu à partir duquel les joueurs pourront créer une nouvelle partie ou joindre une partie existante. La construction d'un nouveau jeu s'effectuera à partir de données fournies par le joueur. De plus, la solution comportera un tutoriel explicatif afin que chaque joueur puisse comprendre le mode de jeu. Finalement, nous proposons d'offrir la possibilité de jouer au jeu à partir d'une tablette. Notre proposition passe donc par la bonification de l'application pour client lourd, l'ajout d'un serveur et l'ajout d'une application pour client léger (tablette Android).

2.2. Hypothèses et contraintes

Le présent plan de projet repose sur diverses hypothèses. Une première hypothèse est que nous disposons d'une équipe de cinq programmeurs-analystes pour mettre en oeuvre le projet. Ensuite, une autre hypothèse est que cette équipe possède les compétences techniques et non techniques nécessaires à la réalisation du projet. Une dernière hypothèse est finalement que nous disposons du matériel et des logiciels informatiques adéquats pour réaliser la partie technique du projet : ordinateurs personnels, systèmes d'exploitation, tablette, émulateurs de tablette, etc.

Pour ce qui est des contraintes applicables au projet, une première d'entre elles est que l'équipe est constituée de seulement cinq personnes, qui doivent se porter responsables de toutes les facettes du projet (planification, conception, architecture, développement, validation). Ensuite, une contrainte matérielle est que nous disposons d'une seule tablette Android à des fins de tests. La contrainte principale qui s'applique au projet est toutefois une contrainte de temps: le temps de réalisation du projet est limité, d'une part à cause d'un échéancier serré qui prévoit une livraison finale le 28 novembre 2019, et d'autre part à cause de la charge de travail élevée des développeurs à l'extérieur du projet, ce qui peut limiter le temps que l'équipe est en mesure de consacrer au projet.

2.3. Biens livrables du projet

Les biens livrables du projet se décomposent en deux catégories, soit la réponse à l'appel d'offres et le produit final. La réponse à l'appel d'offres comprend les artéfacts suivants: le plan de projet (présent document), le document de spécification des requis du système, une liste d'exigences, le document d'architecture logicielle, le protocole de communication ainsi que les prototypes pour le client lourd et le client léger. La date prévue de livraison de la réponse à l'appel d'offres est le 27 septembre 2019. Le produit final comprend les artéfacts précédents ainsi qu'un plan de tests et les résultats de tests, en plus du code source et de l'exécutable du produit. La date prévue de livraison de ce produit final est le 28 novembre 2019.

3. Gestion et suivi de l'avancement

3.1. Gestion des exigences

En cas de changements aux exigences lors du développement de notre logiciel, nous avons plusieurs méthodes de communication en équipe afin de divulguer ces changements aux autres membres le plus rapidement possible. Par exemple, nous avons créé une conversation sur Slack et sur Facebook Messenger. Nous avons aussi défini nos plages horaires avec nos disponibilités sur un fichier Excel que nous avons déposé dans notre Google Drive. De cette manière, nous pouvons connaître les moments où tous les membres de l'équipe sont disponibles et ainsi organiser des rencontres dans lesquelles nous pourrons apporter des modifications à notre échéancier si nécessaire.

De plus, une partie de nos rencontres consiste à vérifier que l'avancement au niveau des exigences suit bien nos prévisions et notre planification dans Redmine; nous pourrons donc nous accommoder et effectuer les changements nécessaires si ce n'est pas le cas. Ceci nous permettra d'être certains que notre projet se déroule bien et qu'aucun retard grave ne se produit et ainsi pouvoir remettre un travail de qualité pour chacune de nos versions.

3.2. Contrôle de la qualité

Afin de s'assurer de la bonne qualité de notre code, nous allons créer des nouvelles branches pour chacune des fonctionnalités et faire de la revue de code pour chaque partie dans ces branches. Le code d'une personne ne pourra donc pas être intégré à la branche principale tant et aussi longtemps que celui-ci n'a pas été vérifié et approuvé par au moins un autre membre de l'équipe. Afin de mettre en place ce mécanisme, nous allons utiliser les *Pull Requests*, un outil déjà intégré à Github qui permet d'alerter les autres membres de l'équipe lorsque des modifications au code ont été envoyées et sont prêtes à être révisées.

Lorsqu'une anomalie sera détectée par un des membres de l'équipe, celui-ci créera tout de suite une demande sur Redmine qui sera assignée à quelqu'un le plus rapidement possible. La correction apportée par cette personne sera ensuite révisée à l'aide d'une *Pull Request*.

De plus, nous allons tenir des rencontres de type *stand-up* à chaque fois que nous nous rencontrons afin de tenir toute l'équipe informée et de guider chaque membre vers la bonne direction au besoin. De cette manière, nous pourrons être certains que chaque personne travaille sur sa partie avec les technologies et outils adéquats et ainsi nous assurer de livrer un produit de qualité.

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 : 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.

3.3.1 - Changement des requis				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
5	Changement continuel des exigences affectant l'avancement du projet et l'échéancier	M	Échéancier (temps de développement)	Accepter les nouvelles exigences et adapter l'échéancier

3.3.2 - Panne du système				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
6	Certains composants critiques, comme la base de données, tombent en panne	É	Fiabilité / Performance	Changer les technologies utilisées si cela se produit souvent

3.3.3 - Fuite/Perte d'informations				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
6	Des informations plus ou moins importantes (mots de passe, pointages, etc.) sont perdues ou volées	É	Fiabilité / sécurité	Avertir les utilisateurs et faire un *hotfix* le plus rapidement possible

3.3.4 - Problème de mise à l'échelle (*scale*)				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
4	Problème de performance et d'adaptation à un plus grand nombre des joueurs en même temps	M	Performance du logiciel	Solutions pour améliorer la mise à l'échelle: éviter les nombreuses demandes à la BD, etc.

3.3.5 - Problème non détecté				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
3	Une anomalie dans le code n'est pas détectée ou détectée trop tard (après la livraison)	F	Assurance qualité	Corriger le bogue et renforcer les mesures d'assurance qualité (tests)

3.3.6 - Problèmes de sécurité				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
5	Le code contient des vulnérabilités pouvant amener à des risques de sécurité	M	Sécurité	Instaurer des mesures de vérification de la sécurité des nouveaux composants

3.3.7 - Composants non fiables				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
5	Certains composants échouent à effectuer leurs tâches régulièrement	M	Expérience utilisateur	Revue de code afin de déterminer le problème et éviter que ça se reproduise

3.4. Gestion de configuration

Lorsqu'un problème est trouvé par l'un des membres de l'équipe, que ce soit un bogue dans le code, un problème avec la définition d'une tâche, etc., celui-ci doit aviser l'équipe des changements qu'il veut apporter afin de régler ce problème. La communication de ces modifications peut se faire soit par l'une de nos conversations sur Internet (Slack ou Facebook) ou lors d'un de nos *stand up*. S'il s'agit d'un problème dans le code, une demande peut être créée sur Redmine et assignée à un des membres de l'équipe. Celui-ci pourra créer une branche afin de régler l'anomalie avant de créer une *Pull Request*. Dans le cas où il s'agit d'un problème autre que dans le code (dans Redmine, par exemple), l'équipe peut s'entendre sur la bonne modification à apporter.

Pour ce qui a trait au nommage des différents artéfacts, nous avons défini une convention de la sorte pour les noms des demandes dans Redmine : [Nom de la partie du projet] Nom descriptif de la tâche. Par exemple : [Client Lourd] Créer la page de connexion. De cette manière, nous pouvons identifier rapidement quelles tâches appartiennent à quelle partie du projet. Nous allons aussi utiliser la fonctionnalité de Redmine qui permet de définir des demandes liées (enfants) et créer une arborescence afin de faciliter la visualisation des différentes sous-tâches pour une fonctionnalité ou une anomalie. Finalement, nous avons décidé d'utiliser l'anglais comme langue pour nommer nos classes, variables et fonctions dans le code.

4. Échéancier du projet

Ce projet comporte deux livrables: la réponse à l'appel d'offre et la remise du produit final. La date d'échéance de ce premier livrable est le 27 septembre 2019. Nous fonctionnerons avec des sprint hebdomadaires allant du jeudi au mercredi. Les principaux lots de travail pour l'appel d'offre se décomposent comme suit :

		Effort estimé (heures-personnes)
Livrable 1 : APPEL D'OFFRE 3 septembre au 27 septembre 2019	**SPRINTS 0, 1, 2 3 septembre au 27 septembre 2019**	**240**
	Planification des sprints et clarification du projet en équipe	15
	Rédaction du plan de projet	15
	Rédaction du document de SRS pour la remise en LOG3000	20
	Rédaction du document d'architecture logicielle	15
	Rédaction du protocole de communication	15
	Création des maquettes des différentes pages de l'application	10
	[Client lourd] Création de l'interface de la page de connection	15
	[Client lourd] Création de l'interface de la page de messagerie	15
	[Client lourd] Envoi et réception de données avec websockets	20
	[Client léger] Création de l'interface de la page de connection	15
	[Client léger] Création de l'interface de la page de messagerie	15
	[Client léger] Envoi et réception de données	20
	[Serveur] Mise en place des routes de base	15
	[Base de données] Mise en place des modèles de messages, usagers, connexion	10
	Ajustement du document de SRS suite à la correction pour la remise finale	5
	[Client léger] Amélioration de l'expérience utilisateur - connection grâce à un third party et partage de ses scores	10
	[Client lourd] Amélioration de l'expérience utilisateur - connection grâce à un third party et partage de ses scores	10

La date d'échéance du deuxième et dernier livrable est le 28 novembre 2019. Nous fonctionnerons avec des sprint hebdomadaires allant du jeudi au mercredi. Les principaux lots de travail pour la remise du produit final se décomposent comme suit :

		Effort estimé (heures-personnes)
Livrable 2 : PRODUIT FINAL 28 septembre au 28 novembre 2019	**SPRINTS 3, 4, 5** **27 septembre au 16 octobre 2019**	**205**
	Planification des sprints	10
	[Client lourd] Ajustement de l'outil PolyPaint - supprimer les fonctionnalités superflues	25
	[Client lourd] Implémentation du mode de jeu classique	60
	[Client lourd] Interface de construction de jeu	10
	[Client léger] Création de l'outil PolyPaint pour tablette	50
	[Client léger] Implémentation du mode de jeu classique	30
	[Serveur] Implémentation du mécanisme pour transformer une image en SVG	20
	SPRINTS 6, 7 **17 octobre au 30 octobre 2019**	**150**
	Planification des sprints	10
	[Base de données] Gestion et reformatage de la base de données	10
	[Base de données] Gestion de l'historique de clavardage	20
	[Serveur] Implémentation de la liste d'amis	20
	[Serveur] Implémentation du mécanisme pour la recherche automatique d'images et d'indices	40
	[Client léger] Intégration des différents modes de clavardage et canaux multiples	25
	[Client lourd] Intégration des différents modes de clavardage et canaux multiples	25
	SPRINTS 8, 9 **31 octobre au 6 novembre 2019**	**150**
	Planification des sprints	10
	[Client lourd] Création du tutoriel non interactif	10

[Client lourd] Ajustement de l'interface et du UX	5
[Client lourd] Implémentation du mode de jeu sprint coopératif	20
[Client lourd] Implémentation du mode de jeu "un seul trait"	20
[Client léger] Création du tutoriel non-interactif	10
[Client léger] Ajustement de l'interface et du UX	5
[Client léger] Implémentation du mode de jeu sprint coopératif	20
[Client léger] Implémentation du mode de jeu "un seul trait"	20
[Base de données] Création des différentes personnalités pour les joueurs virtuels - formules génériques	10
[Serveur] Gestion et refactoring du serveur	20
SPRINTS 10, 11 7 novembre au 28 novembre 2019	**155**
Planification des sprints	10
[Serveur] Implémentation du mécanisme de reconnaissance de tricherie	35
[Client lourd] Création du tutoriel interactif	10
[Client lourd] Implémentation du mode de jeu "aveugle"	15
[Client lourd] Ajout d'effets visuels et sonores - jusqu'à 3	10
[Client léger] Création du tutoriel interactif	10
[Client léger] Implémentation du mode de jeu "aveugle"	15
[Client léger] Ajout d'effets visuels et sonores - jusqu'à 3	10
Rédaction du plan de tests logiciels	15
Rédaction du document de résultats des tests logiciels	15
Préparation de la présentation orale	10

Cela nous amène à un total de 900 heures-personne.

5. Équipe de développement

Notre équipe est constituée de cinq développeurs qui seront responsables de l'ensemble du cycle de développement logiciel. Ces développeurs possèdent une expérience diversifiée et différentes responsabilités au sein du projet.

Marie-Elaine Bérubé

Marie-Elaine Bérubé est une étudiante en troisième année au baccalauréat en génie logiciel à Polytechnique Montréal. Elle a réalisé un stage à la Ville de Montréal et a de nombreuses implications au sein de l'école, où elle a notamment occupé des postes tels que webmestre. Elle a donc beaucoup d'expérience front-end, Javascript/Typescript et HTML/CSS. Elle a également été chargée de laboratoire du cours INF1005C et a donc de l'expérience en C++. Dans le cadre de ce projet, ses responsabilités seront de travailler sur le client lourd et le client léger, donc plutôt du côté front-end. Elle se chargera entre autres de l'expérience utilisateur.

Gabriel Bottari-Defrance

Gabriel Bottari-Defrance est étudiant en quatrième année au baccalauréat en génie logiciel à Polytechnique Montréal. Il a beaucoup d'expérience en entreprise avec trois stages à son actif, notamment chez Microsoft en tant que développeur full stack C++/C# et XAML, Rakuten en tant que développeur Node.js et React.js, et finalement Google comme développeur Python et Angular.js. Outre ces compétences techniques, il maîtrise le Javascript/Typescript, Flutter/Dart, Java, Golang, MySQL, MongoDB, en plus du déploiement automatisé et Machine Learning. Dans le cadre du projet, ses responsabilités seront de définir les fonctionnalités et routes du serveur selon les exigences, en plus des communications avec la base de données.

Christophe Carreau-Lacasse

Christophe Carreau-Lacasse est un étudiant en quatrième année au baccalauréat en génie logiciel à Polytechnique Montréal. Il est aussi titulaire d'un diplôme en technique de l'informatique au cégep Édouard-Montpetit. Sa technique et ses nombreuses expériences de stage dans des entreprises telles Desjardins, Skytech ou bien Square Enix Montréal, lui ont permis d'acquérir une expertise technique dans plusieurs langages de programmation et cadriciels, tel que C#, Java, C++, Javascript, Typescript, NodeJS, MySQL, MongoDB, T-SQL, et bien plus. Ces expériences lui ont aussi permis de se familiariser avec la méthodologie de développement Agile. Dans ce projet, il se spécialisera dans la communication entre client et serveur, c'est-à-dire qu'il travaillera autant sur le client que sur le serveur et s'assurera que la communication entre les deux fonctionne bien.

Justine Lambert

Justine Lambert est étudiante en troisième année au baccalauréat en génie logiciel à Polytechnique Montréal. Elle a notamment réalisé un stage à la Caisse de dépôt et placement du Québec à titre d'analyste en intelligence d'affaires ainsi qu'un stage chez Desjardins à titre de développeur mobile full stack. Elle travaille actuellement comme développeur front-end chez Desjardins. Ses intérêts résident d'ailleurs principalement dans le développement d'interfaces utilisateurs. Sur le plan technique, Justine maîtrise entre autres les langages suivants : JavaScript/TypeScript, Angular, Flutter/Dart, HTML/CSS, Java et SQL. Dans le cadre du projet, elle interviendra principalement du côté client, autant au niveau du client lourd et que du client léger: interfaces, expérience utilisateur, etc.

Olivier Naud-Dulude

Olivier Naud-Dulude est étudiant en troisième année au baccalauréat en génie logiciel à Polytechnique Montréal. Avec deux stages à son actif chez CanmetENERGY et Genetec, il possède des compétences dans plusieurs langages et

technologies de développement logiciel. Ses principales compétences sont : C#, C++, Java, Angular, React.js, JavaScript/TypeScript, NodeJS, MongoDB, etc. Il tente habituellement d'être full stack dans les projets auxquels il participe, afin d'avoir une connaissance technique globale du projet et d'acquérir le plus de compétences possibles. Dans le cadre du projet, ses responsabilités seront partout au niveau du client lourd et du client léger. Par exemple, il s'occupera d'envoyer des informations vers le serveur, de développer des interfaces graphiques, d'améliorer l'expérience utilisateur, etc.

6. Entente contractuelle proposée

Nous proposons une entente contractuelle de type clé en main à prix ferme. Le produit final, le logiciel Turb0_P41nt_F1v3_Th0u54nd, sera livré dans son entièreté le 28 novembre 2019. Le logiciel sera livré avec toutes les exigences essentielles, en plus d'au moins 50% des exigences souhaitables détaillées dans la liste d'exigences. Nous nous engageons à absorber les dépassements budgétaires s'il s'avérait nécessaire de faire appel à plus de main d'oeuvre pour livrer un logiciel complet à la date d'échéance du contrat. Considérant une charge de travail estimée à 900 heures pour une équipe composée de quatre développeurs et d'un gestionnaire de projet travaillant à un taux de 100$/h et 125$/h respectivement, soit 180 heures par personne, le prix final est de 94 500$. Ce montant devra être acquitté en totalité à la livraison du logiciel.