POLYTECHNIQUE MONTRÉAL

LE GÉNIE EN PREMIÈRE CLASSE

Département de génie informatique et génie logiciel

INF3995

Projet de conception d'un système informatique

Proposition répondant à l'appel d'offres
no. A2018-INF3995 du département GIGL.

Système audio pour café Internet

Équipe No 1

Team One, Inc.
Emir Khaled Belhaddad
Adam Martin Côté
Anthony Dentinger
Soukaina El-Ghazi
Othman Mounir

Septembre 2018

Table des matières

1. Vue d'ensemble du projet

1.1 But du projet, portée et objectifs (Q2.1 et Q4.1)

Café-Bistro Élévation, Inc. a été fondé il y a de nombreuses années et a vu l'évolution de sa clientèle au cours de ses années de service. Afin de mieux répondre aux critères d'une clientèle plus jeune, l'entreprise souhaite à présent ajouter des installations qui permettront aux clients de jouer leurs propres propres chansons sur le système sonore du café-bistro via une application sur tablette Android. Café-Bistro Élévation, Inc. entrevoit un grand succès pour ce projet d'autant plus que sa clientèle compte de nombreux artistes

Il est attendu que le bien livrable se compose de ladite application Android ainsi qu'un serveur sur le système d'exploitation Arch Linux fonctionnant sur un Zedboard du constructeur Xilinx. L'application sur Arch Linux sera distribuée, par commodité, dans un conteneur Docker.

1.2 Hypothèse et contraintes (Q3.1)

Café-Bistro Élévation, Inc. repose en partie sur les postulats que la clientèle n'est pas informée que l'internet gratuit (WiFi) fait partie des services offerts. Café-Bistro Élévation, Inc. devrait donc voir sa clientèle se rajeunir et augmenter, d'une part en offrant un service sur-mesure qui correspond à la clientèle visée, et, d'autre part, en mettant très justement de l'avant le service d'internet gratuit utilisé dans le cadre de ce projet.

Le projet devra être opérationnel à une date prévue d'avance par Café-Bistro, Inc. et le suivi se fera en deux livrables, décrits plus bas.

Le service proposé reposera sur un serveur Arch Linux situé sur un système sur puce Xilinx Zedboard ainsi qu'une application Android sur tablette qu'utiliseront les clients de Café-Bistro, Inc..

En termes de matériel, le système sur puce visé a les contraintes déterminées par Café-Bistro, Inc. (Hosson, St-Onge et Collin, 2018) :

- deux processeurs ARM Cortex-A9 proprement configurés pour le contexte;
- un accès à une mémoire vive externe de 512 MiB;
- un accès à une mémoire SD externe;
- un accès à l'interface réseau;
- un pilote pour contrôler l'interface HDMI (vidéo);
- un pilote pour contrôler le son en sortie (connecteur audio 3.5mm); et

- un pilote pour contrôler un UART (pour une console en mode administrateur).

Team One, Inc. possède des Zedboard qui correspondent précisément aux spécifications indiquées et pourra donc effectuer son développement dans des conditions quasi-identiques aux conditions réelles.

Le serveur sur Arch Linux sera construit en C++ tel que requis par les spécifications techniques du projet, et nous utiliserons également les technologies suivantes afin de réaliser le projet :

- CMake : Cadriciel qui permettra de lier ensemble les différentes parties du logiciel du serveur Arch Linux.
- Pistache : Librairie qui implémentera le serveur HTTP sous Linux.
- libmad : Librairie qui effectuera le décodage des fichiers MP3 vers un format qui pourra être joué sur le système sonore.
- Alsa : Cadriciel Linux qui permettra de jouer le fichier MP3 décodé sur le système sonore.
- ffmpeg : Programme qui permet la lecture de MP3. Cette solution pourrait remplacer libmad et Alsa, mais pourrait peut-être poser des problèmes de performance.
- id3lib: Librairie permettant le décodage de l'entête MP3, et ainsi l'extraction des informations (artiste, durée, …) de la chanson.
- OpenSSL : Librairie qui déchiffrera les données envoyées par l'administrateur lors de la connexion, la déconnexion et le changement de son mot de passe.
- SQLite : Librairie qui agira comme gestionnaire de la base de données.
- Docker : Cadriciel qui permettra de transformer le serveur Arch Linux en conteneur afin qu'il soit distribué et mis en production très aisément.

L'application Android sera développée à l'aide du langage Kotlin dans le logiciel Android studio. Elle sera testée principalement sur tablette Samsung Galaxy Tab S2 (modèle T710). Nous utiliserons également les technologies suivantes :

- *Koin* : librairie permettant l'injection de dépendance.
- *Gson* : librairie permettant la manipulation de données en format JSON.

1.3 Biens livrables du projet (Q2.3)

Nous proposons de livrer un système complet et fonctionnel répondant aux spécifications du client. Le système sera testé et déployé par notre équipe sur un Zedboard analogue à celui du client. La livraison du produit se fera en 2 jalons, tel que demandé par le Café-Bistro Élévation, Inc. :

- 13 novembre 2018 : livraison d'un système fonctionnel répondant au critères du livrable 1.
- 29 novembre 2018 : livraison de la version finale du produit selon les spécifications du client (Hosson, St-Onge et Collin, 2018).

En plus du système fonctionnel, tout le code source développé au cours du projet sera remis au client. La version du logiciel livrée le 29 novembre sera pleinement fonctionnelle et sera la propriété du client. Aucun support additionnel n'est prévu après cette date.

2. Organisation du projet

2.1 Structure d'organisation

Afin de répondre aux besoins de Café-Bistro Élévation, Inc., Team One, Inc. déploiera les ressources humaines suivantes :

- Emir Khaled Belhaddad : Coordonnateur de projet. M. Belhaddad agira également en tant que personne-ressource au niveau technique ainsi que développeur-analyste, principalement sur le client Android.
- Adam Martin Côté : Développeur-analyste. M. Côté a quelques années d'expériences d'utilisateur d'Arch Linux et agira donc comme personne-ressource sur ce système d'exploitation. M. Côté travaillera principalement sur les serveurs HTTP et HTTPS.
- Anthony Dentinger : Développeur-analyste. M. Dentinger, de par ses années d'expérience du C++, agira comme personne-ressource sur cette technologie et travaillera principalement sur les serveurs HTTP et HTTPS.
- Soukaina El-Ghazi : Développeur-analyste. Mme. El-Ghazi travaillera principalement sur le client Android.
- Othman Mounir : Développeur-analyste. M. Mounir travaillera principalement sur les serveurs HTTP et HTTPS.

Cette structure organisationnelle facilitera le travail et en augmentera l'efficacité dans la mesure où chaque personne travaille sur une portion du projet plus petite, ce qui réduira le nombre d'anomalies qui sont souvent introduites par une compréhension limitée de la portion du projet sur laquelle le développeur-analyste travaille.

Il restera toutefois tout-à-fait envisageable qu'un membre travaillant sur le serveur vienne en aide à l'équipe travaillant sur l'application cliente Android, ou inversement. En particulier, une tâche produite par une équipe pourra être revue par un membre de l'autre équipe afin que les membres d'une équipe soient un minimum au courant de ce que les membres de l'autre équipe font.

2.2 Entente contractuelle

Le projet proposé par Café-Bistro Élévation, Inc. étant clairement délimité et dans la mesure où le répondant construira le projet depuis la base, Team One, Inc. propose au promoteur un contrat à prix ferme aux frais décrits à la section 4.1. Il sera donc entendu, dans les limites légales, que le montant fixé par l'entente contractuelle sera versé lors de l'acceptation du produit logiciel demandé aux délais indiqués.

3. Solution proposée

3.1 Architecture logicielle sur serveur (Q4.2)

Le serveur Arch Linux est au coeur du système visé par le projet, puisque c'est lui qui traitera les demandes de l'application Android (le "client Android") et effectuera réellement la lecture des fichiers MP3 afin de les transmettre sur le système audio.

Team One, Inc. propose la solution décrite à la Fig. 1 afin de répondre aux besoins de Café-Bistro Élévation, Inc..

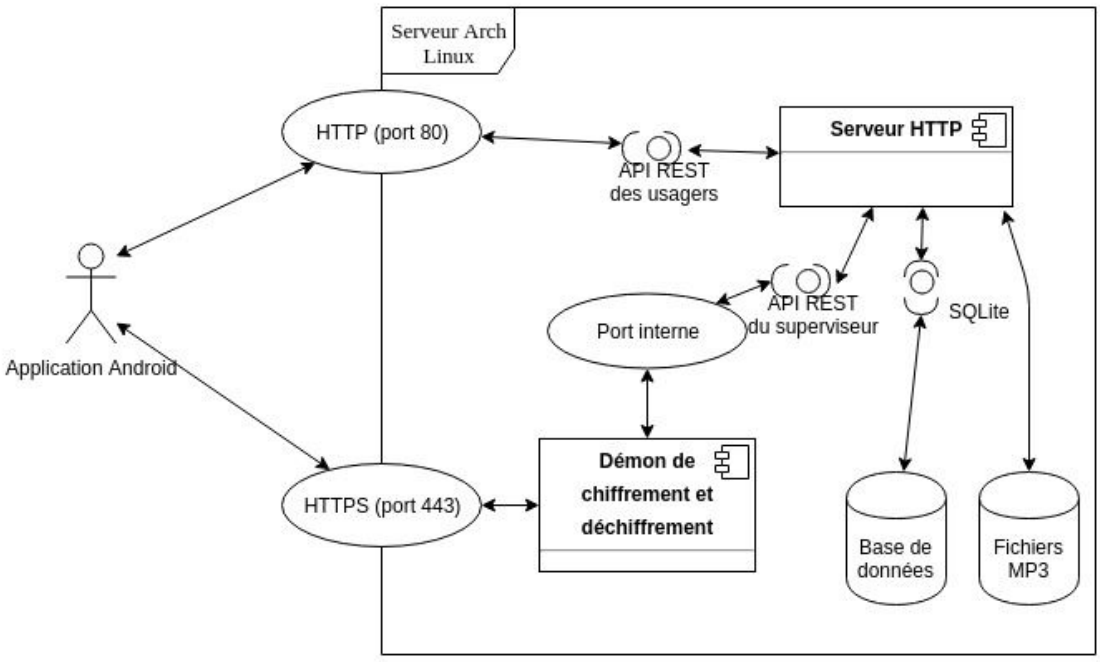

Fig. 1 - Vue à haut niveau de la communication avec le serveur

Les requêtes du client Android se feront, comme spécifié, via deux canaux séparés : un canal HTTP pour les requêtes non sensibles, et un canal HTTPS (HTTP chiffré) pour toutes les requêtes du superviseur. Dans la mesure où la

technologie Pistache ne supporte pas directement le protocole HTTPS, nous développerons donc un programme d'arrière-plan ("démon") qui déchiffrera les messages HTTPS reçus puis les retransmettra au serveur Pistache via un port (*socket*) interne (invisible de l'extérieur du serveur) et via une API REST séparée de celle des usagers normaux. Les réponses du serveur seront ensuite chiffrées à nouveau puis transmises vers le client.

Les demandes de l'application Android qui produisent un effet persistant sur le serveur verront cet effet s'effectuer sur une base de données gérée par SQLite, sauf pour les fichiers MP3 qui seront sauvegardés à même le système de fichiers.

3.2 Architecture logicielle sur tablette (Q4.3)

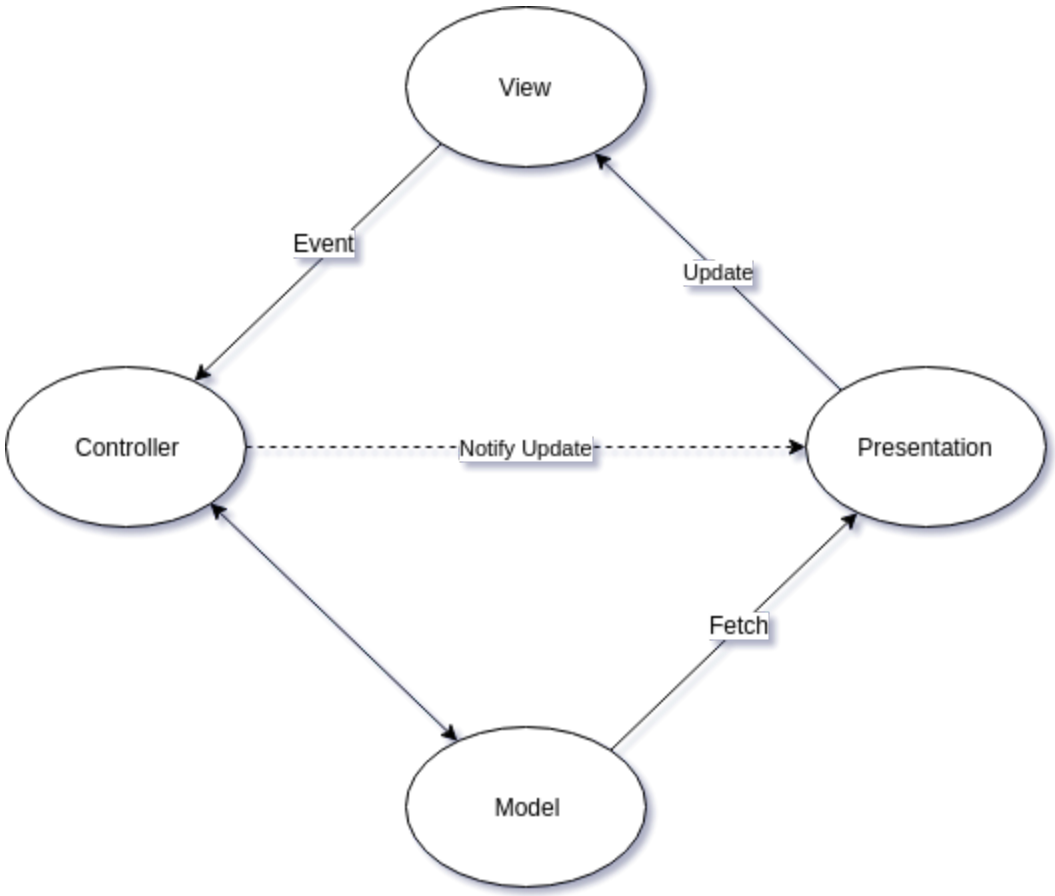

Fig. 2 - Diagramme de très haut niveau de l'architecture *MVCP* utilisée pour le client

L'application client est l'interface avec laquelle les clients vont interagir avec le système. C'est pour cela que Team One, Inc. propose une architecture personnalisée et robuste, basée sur MVC en ajoutant le module présentation de l'architecture MVP afin d'éliminer l'interaction entre la vue et le modèle. Nous

avons choisi cette architecture afin d'avoir une conception claire et efficace grâce à la séparation des données de la vue, de module présentation et de contrôleur. Cette architecture nous permet de faciliter la maintenance de l'application, d'avoir une grande souplesse et moins de couplage.

Voici une description des modules de l'architecture *MVCP* :

- Vue : représente l'interface avec l'utilisateur. Elle a deux tâches principales, la première étant d'envoyer les actions reçues de l'utilisateur au contrôleur, et la deuxième, d'afficher les données qu'elle récupère par le module présentation.

- Modèle : contient les données manipulées par le contrôleur et assure la gestion de ces données en garantissant leur intégrité.

- Contrôleur : s'occupe de la synchronisation du modèle et de la vue. Le contrôleur reçoit tous les évènements de la vue, demande au modèle de faire les modifications nécessaires qui correspondent à l'action reçue, puis envoie une notification au module présentation pour l'aviser que les données ont changé. En cours, ce module prend en charge tous les flux de données.

- Présentation : ce module s'occupe de chercher les données à partir du modèle dès qu'il reçoit une notification du contrôleur puis d'organiser les données à afficher dans la vue.

Voici un exemple de flux de traitement d'une requête à l'application :

La requête est envoyée depuis la vue et analysée par le contrôleur. Par la suite, le contrôleur demande au modèle d'effectuer les traitements nécessaires, envoie au module présentation une notification pour l'aviser que la requête est traitée et qu'il y a des modifications au niveau du modèle. Le module présentation notifié fait ensuite une requête au modèle pour chercher les données modifiées et envoie une requête à la vue pour afficher le résultat du traitement.

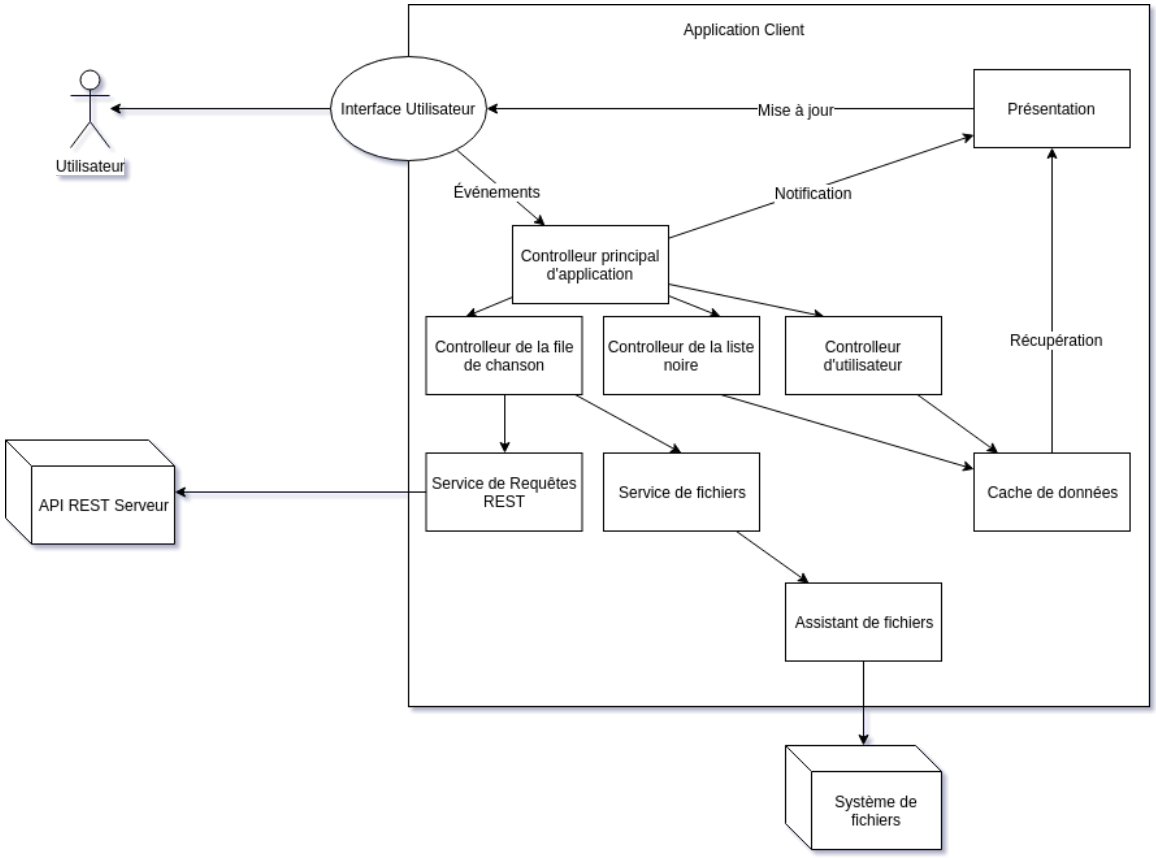

Fig. 3 - Diagramme d'architecture de l'application client Android

Voici un diagramme simplifié de l'architecture de l'application client ainsi qu'une description de chacun de ces modules :

- L'interface utilisateur : Ce module se charge d'afficher les données dans le format désiré et de récupérer les demande de l'utilisateur. Il transmet les interactions de l'utilisateur au contrôleur principal. Comme ce fut stipulé dans l'appel d'offre, le thème d'affichage de l'application sera jaune avec une police de caractère noire. Il y aura une page affichant les prochaines chansons à jouer, les fichiers disponibles, les statistiques ainsi que la liste noire. Comme spécifié dans l'appel d'offre, il y aura 2 modes d'utilisation (utilisateur et superviseur).
- Les Contrôleurs : Les contrôleurs sont les modules qui traitent les flux de données. Ils contiennent la logique principale de l'application. Ils font appellent à des services qui leurs donnent accès aux ressources de l'appareil.
- La Présentation : Cette partie gère la mise à jour et le formatage des différentes vues de l'interface utilisateur. Lorsque les contrôleurs demande une mise à jour, la présentation récupèrent les données de la cache de l'application et le formate pour un affichage rapide sur les vues.

4. Processus de gestion

4.1 Estimations des coûts du projet

Team One, Inc. justifie le coût d'allocation de ses services techniques au projet de Café-Bistro Élévation, Inc. selon le temps estimé de l'effort fourni ainsi que les ressources humaines déployées. Il s'ensuit le calcul suivant :

Poste	**Personne assignée**	**Taux horaire facturé ($)**	**Pondération**
Coordonnateur de projet	Emir Khaled Belhaddad	145.00	20%
Développeur-analyste	Adam Martin Côté	130.00	20%
Développeur-analyste	Anthony Dentinger	130.00	20%
Développeur-analyste	Soukaina El-Ghazi	130.00	20%
Développeur-analyste	Othman Mounir	130.00	20%
Taux horaire résultant		**133.00$**	

L'analyse du projet estime un total de 383.5 heures-personne de travail. Cette donnée résulte de la somme du temps de travail de toutes les tâches décrites à la section 4.2. Le détail exact de l'allocation du temps par tâche peut se retrouver sur (Team One, Inc., 2018). Team One, Inc. soumet donc sa proposition à un montant de 51 005.50$.

4.2 Planification des tâches (Q2.2 et Q11.2)

4.2.1 Planification des tâches - par équipe

Les Fig. 4 et 5 décrivent le planning de réalisation du projet par équipe : lag Fig. 4 décrit le calendrier pour la réalisation du serveur, et la Fig. 5 décrit le calendrier pour la réalisation de l'application cliente Android.

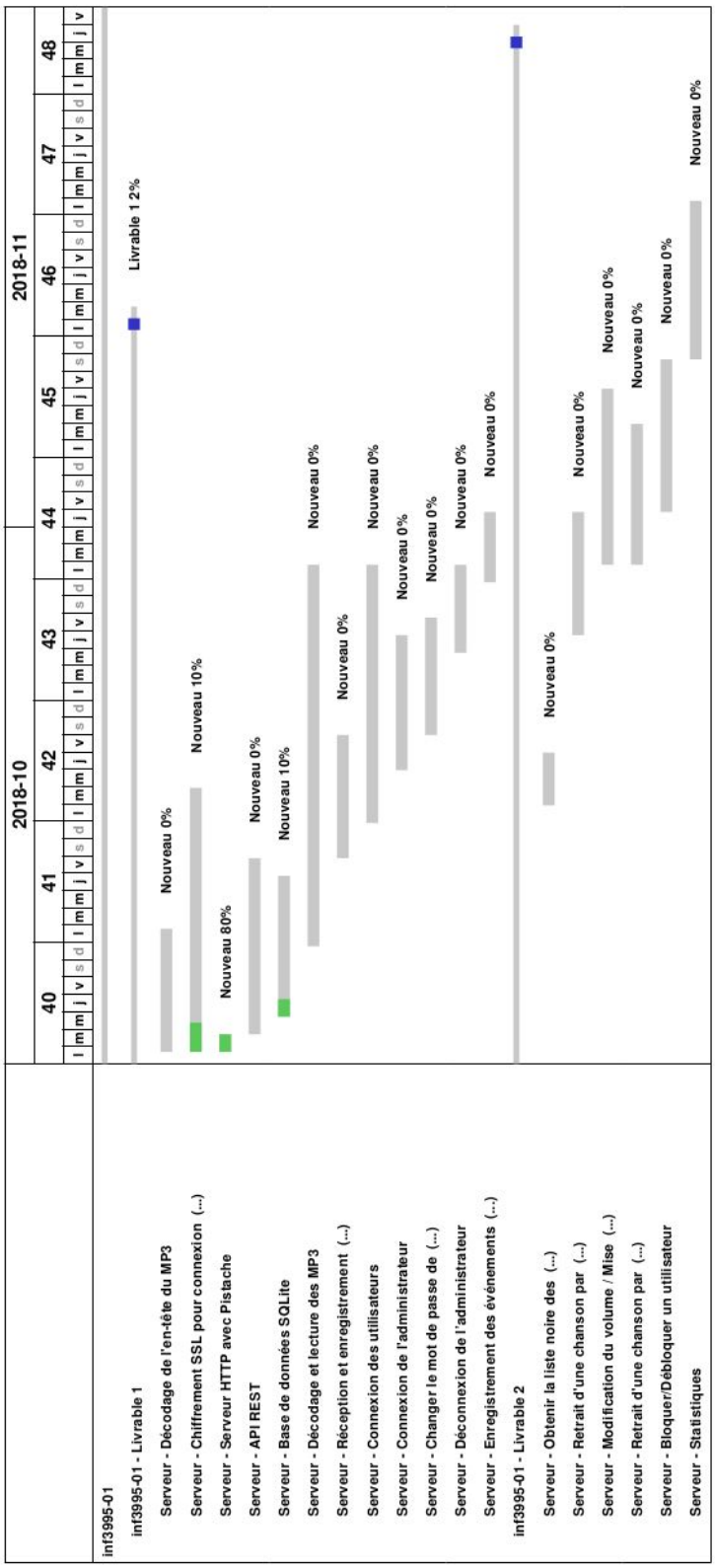

Fig. 4 - Diagramme de Gantt de la progression pour le serveur

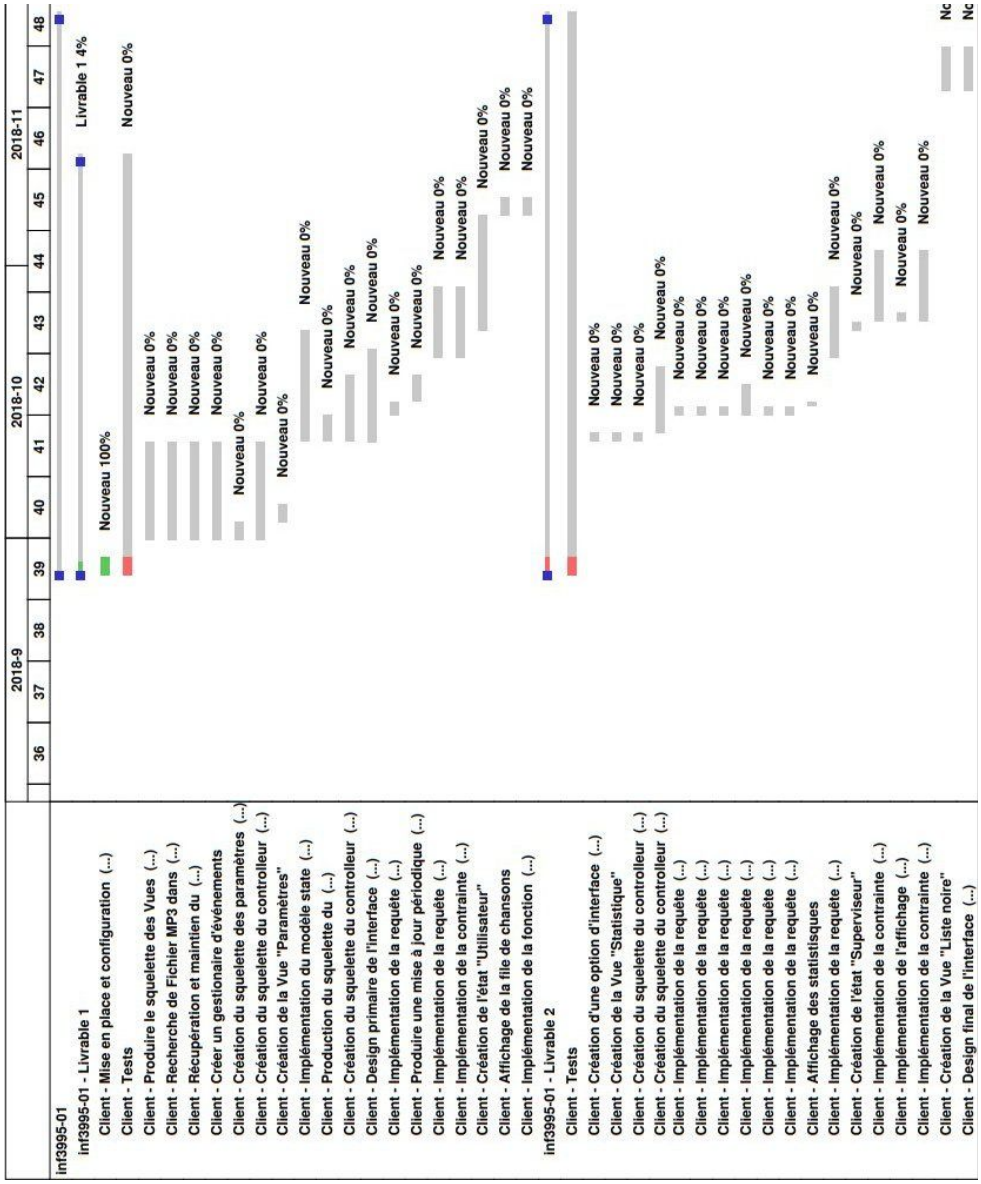

Fig. 5 - Diagramme de Gantt de la progression pour le client

Notons qu'en plus des tâches liées au client et au serveur, nous avons planifier une tâche uniquement pour l'intégration, la vérification et la distribution du produit. Cette tâche sera réalisé par tous les membres de l'équipe.

4.2.2 Planification des tâches - par personne

Les Fig. 6 à 10 décrivent la répartition de ces mêmes tâches, mais par personne.

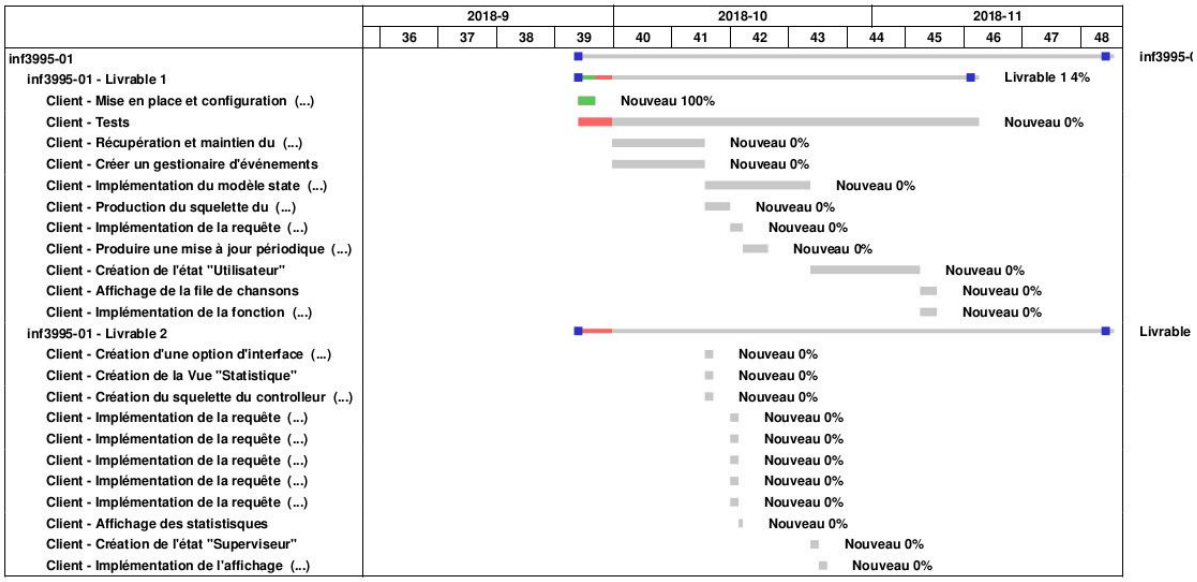

Fig. 6 - Planification des tâches de M. Belhaddad

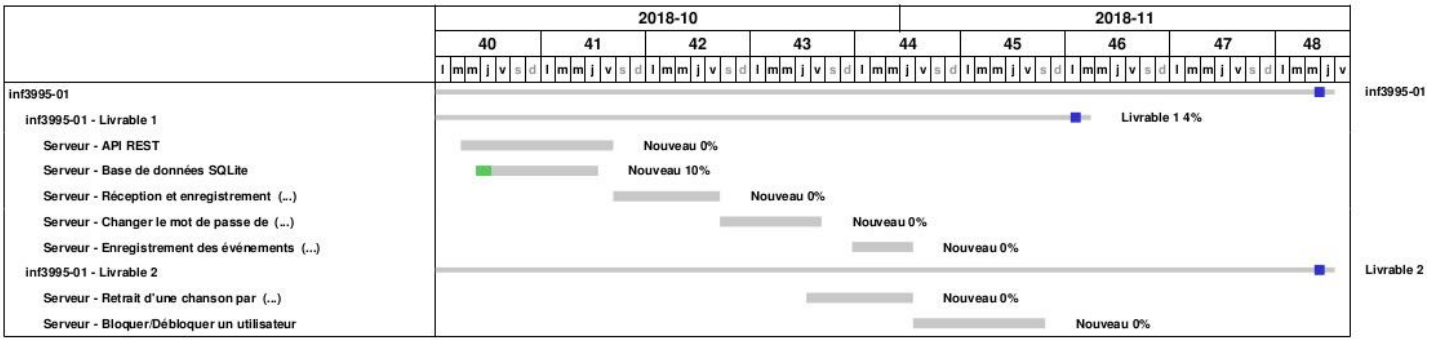

Fig. 7 - Planification des tâches de M. Côté

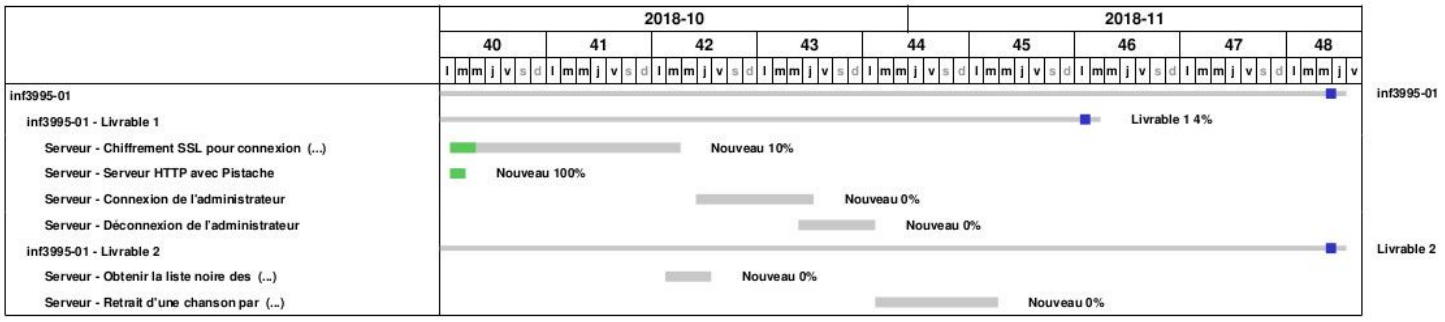

Fig. 8 - Planification des tâches de M. Dentinger

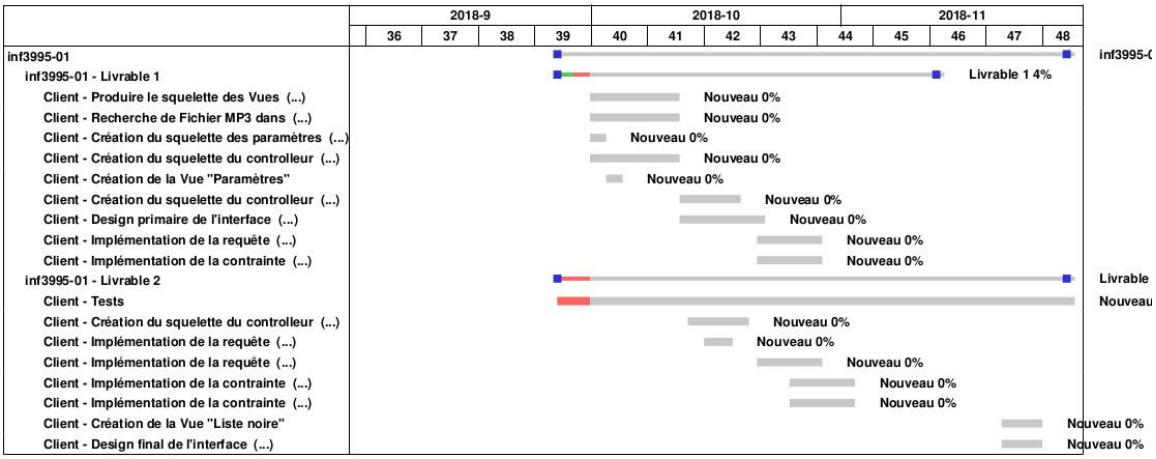

Fig. 9 - Planification des tâches de Mlle. El-Ghazi

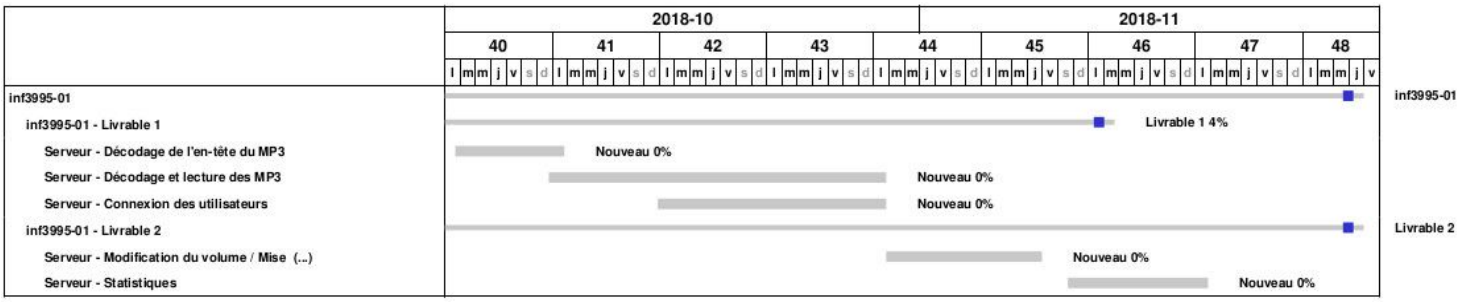

Fig. 10 - Planification des tâches de M. Mounir

4.3 Calendrier de projet (Q3.3)

Le projet de Café-Bistro Élévation, Inc. se déroulera en deux jalons (*milestones*) décrits au tableau I. Notez que la version du livrable 1 sera fonctionnelle, sans toutefois inclure les fonctionnalités avancées d'administration.

Tableau I - Phase de réalisation du projet

Jalon	**Date**	**Contenu**	**Numéro de version**
Jalon 1	Mardi 13 nov. 2018	Fonctionnalités du livrable 1	v0.1
Jalon 2	Jeudi 29 nov. 2018	Fonctionnalités du livrable 2	v1.0

4.4 Ressources humaines du projet

Le projet proposé par Café-Bistro Élévation, Inc. consiste, d'une part, à développer une application Android qui fournira aux clients de Café-Bistro Élévation, Inc. un moyen convivial de transférer des chansons à jouer et à un superviseur d'administrer le système. Ceci requerra deux développeurs-analystes qui travailleront sur les technologies suivantes :

- Le langage de programmation Kotlin pour le coeur de l'application ;
- L'API SDK d'Android pour l'interaction avec le système d'exploitation Android, en particulier pour l'affichage ;
- L'API REST du client pour la communication avec le serveur.

Afin de développer une application maintenable en termes techniques et à s'assurer de la qualité de l'interface graphique, deux développeurs-analystes ayant un minimum de deux années d'expérience d'utilisation de langages de programmation orientés-objet sont nécessaires. Ils devront également montrer une habileté à utiliser de nouvelles technologies s'ils ne sont pas familiers avec les technologies qui seront utilisées par l'application Android.

Ont été assignés sur ladite application Emir Khaled Belhaddad, qui a neuf ans d'expérience en développement orienté-objet, est déjà familier avec l'environnement de développement Android et l'API REST, ainsi que Soukaina El-Ghazi, qui a cinq années d'expérience en développement orienté-objet.

D'autre part, le projet consiste à développer un serveur HTTP/HTTPS qui gérera les demandes des clients du Café-Bistro Élévation, Inc. afin de permettre, notamment, de transmettre, jouer et supprimer les chansons transmises en mode utilisateur, ainsi que, notamment, d'authentifier l'administrateur, bloquer des utilisateurs, interagir avec le volume sonore et supprimer des chansons en

mode administrateur. Ledit serveur HTTP/HTTPS requerra donc notamment l'utilisation des technologies suivantes :

- Le langage de programmation C++ pour le traitement des demandes et le coeur du serveur HTTP/HTTPS ;
- L'utilisation de librairies C++, en particulier pour décoder les fichiers MP3 à jouer et chiffrer les requêtes et réponses de l'administrateur ;
- Le langage de script CMake pour l'assemblage du code C++ en serveur HTTP/HTTPS utilisable sur la carte Zedboard ;
- Le système d'exploitation Arch Linux pour interagir avec le volume sonore et déterminer les causes de problèmes pouvant se présenter durant le développement du projet ;
- La carte FPGA Zedboard de Xilinx, qui sera le support physique du serveur HTTP/HTTPS ;
- L'API REST pour la réception des requêtes de l'application Android.

Le serveur HTTP/HTTPS correspond au coeur du projet soumis par Café-Bistro, Inc.. Trois développeurs-analystes seront nécessaires afin de répondre au cahier des charges. Le serveur HTTP/HTTPS nécessitant un grand nombre de technologies, les développeurs-analystes devront avoir une aisance à s'adapter à des technologies diverses et devraient avoir une connaissance solide du langage C++ et de la programmation orientée-objet de manière générale.

Les trois membres de l'équipe qui travailleront principalement sur le serveur HTTP/HTTPS ont déjà une expérience du langage de programmation C++ et plusieurs années d'expérience d'utilisation de langages orientés-objet. En particulier, Anthony Dentinger a trois années d'expérience en C++ et plus d'une année d'expérience avec CMake. Toute l'équipe a déjà travaillé avec l'API REST pour le développement de serveurs.

Adam Martin-Côté utilise le système d'exploitation Arch Linux depuis plusieurs années et possède une bonne connaissance du fonctionnement interne et de la gestion des configuration sous Linux.

Enfin, les trois membres de Team One, Inc. qui travailleront sur le serveur HTTP/HTTPS ont une facilité à s'adapter à de nouvelles technologies du fait de trois années de formation en génie informatique à Polytechnique Montréal.

5. Suivi de projet et contrôle

5.1 Contrôle de la qualité

En plus du développement proprement dit du projet de Café-Bistro Élévation, Inc., Team One, Inc. soumettra le travail réalisé à un processus

d'assurance qualité durant le cours de son développement et avant la démonstration de chaque jalon. Ce processus consiste en l'exécution des pratiques suivantes :

1. Le développement et l'utilisation régulière de tests automatisés, décrits à la section 5.3 ;
2. L'exécution d'une batterie de tests manuels, décrits à la section 5.3 ;
3. La vérification fonctionnelle de chaque tâche soumise dans le livrable ;
4. La revue par un pair des tâches complétées.

Les pratiques 2 et 3 seront faites lors de tâches d'intégration et précéderont le déploiement en conteneur Docker. Comme décrit à la section 4.2.1, chaque jalon est précédé d'une tâche d'intégration durant laquelle la qualité du produit est assurée.

5.2 Gestion de risque (Q2.6 et 11.3)

L'utilisation d'Arch Linux simplifiera l'installation et le fonctionnement du système d'exploitation sur le Cortex-A9, le processeur se trouvant sur la carte Zedboard qui contiendra le serveur Arch Linux. Arch Linux est toutefois un système d'exploitation avec une politique de mise-à-jour fréquente mais optionnelle. Ceci peut parfois avoir l'effet indésirable de rendre instable les divers logiciels et cadriciels installés sur le système d'exploitation. Team One, Inc. s'assurera de la stabilité du produit, d'une part, en ne faisant pas de mise-à-jour à partir de la phase d'intégration, et d'autre part, en effectuant des sauvegardes de l'état du système d'exploitation.

Le Cortex-A9 présent sur le système sur puce Zedboard étant un processeur relativement simple comparativement à des processeurs pour ordinateurs de bureau classiques, il est possible que certaines tâches, en particulier le décodage des fichiers MP3, prenne plus de temps de calcul (temps CPU) que ce que le processeur nous offre avec des méthodes de base pour décoder le fichier MP3 en temps réel. Si ce cas se présente, nous aurions alors les trois possibilités suivantes :

1. *Réduire la qualité des fichiers MP3 lors de leur réception*. Cette méthode est relativement simple à mettre en place mais impacterait, visiblement ou non, la qualité du son joué.
2. *Décoder les fichiers MP3 d'avance*. Dans la mesure où le Cortex-A9 présente deux coeurs, il serait possible de décoder la prochaine chanson à jouer pendant que la chanson en cours décode et joue. La qualité du son ne serait pas impactée, mis à part le fait qu'il faudrait induire un temps d'attente lors que la première chanson est soumise. Il est possible de

combiner cette solution avec la première solution de manière à réduire ce temps d'attente.

3. *Utiliser un décodeur MP3 matériel*. Cette solution est la plus performante puisqu'elle résoudrait certainement le problème de temps de décodage qualité du son et de, mais elle est également la plus complexe puisqu'elle requiert l'interfaçage d'un module de décodage matériel avec le système d'exploitation via un pilote de périphérique ("*kernel module*", selon la sémantique Linux) et requiert la recompilation du noyau du système d'exploitation. Cette solution aurait également l'effet de rendre le système dépendant de la technologie FPGA de Xilinx, ce qui rendrait la migration du serveur Arch Linux vers un ordinateur personnel plus difficile.

5.3 Tests (Q4.4)

Les tests automatisés sont un outil qui peut à la fois accélérer le développement d'un projet ou le ralentir. Nous avons décidé de ne pas faire des tests systématique sur l'ensemble du projet, mais plutôt d'utiliser des tests ciblé pour les sections critiques ayant un haut potentiel d'erreur et d'effet secondaires.

Nous utiliserons plusieurs approches différentes pour l'élaboration des tests du projet:

- Tests d'intégration en python à haut niveau :

  Quelques tests de sanité pour détecter les anomalies majeures. Nous utiliserons cette approche notamment pour tester l'API rest du serveur.

- Tests unitaires de fonctions critiques :

  Test plus approfondis à l'aide de la librairie de test de *boost* en C++, sur certaines fonctions critiques au projet. Certains membres de l'équipe pourront utiliser ce type de test afin de travailler en TDD (écrire les tests avant le code) à leur discrétion.

- Tests manuels :

  Nous établirons une liste de tests manuels à effectuer de façon systématique avant les remises. Cette catégorie de tests visera principalement l'interface utilisateur.

- Tests de performance (manuels) :

  Le système développé devra respecter certaines contraintes de performance, notamment lors de la lecture de fichier *MP3*, qui devront respecter la durée originale afin d'éviter les déformations. Nous testerons la performance de ceux-ci de façon manuelle.

5.4 Gestion de configuration

Le système de contrôle de version qui sera utilisé dans ce projet est git, nous utiliserons la stratégie GitFlow pour la gestion de branche. Le dépôt central va héberger deux branches principales *master* et *develop*, La branche master est la branche essentielle du projet qui contient la version la plus stable et qui est prête à être déployée. La branche develop centralise toutes les nouvelles fonctionnalités qui seront présentes dans le prochain livrable. Une branche *Feature* sera créée pour chaque fonctionnalité. dès que le développement de la tâche soit terminé, elle sera fusionnée avec la branche *develop.*

La figure ci-dessous présente la stratégie GitFlow de gestion des branches :

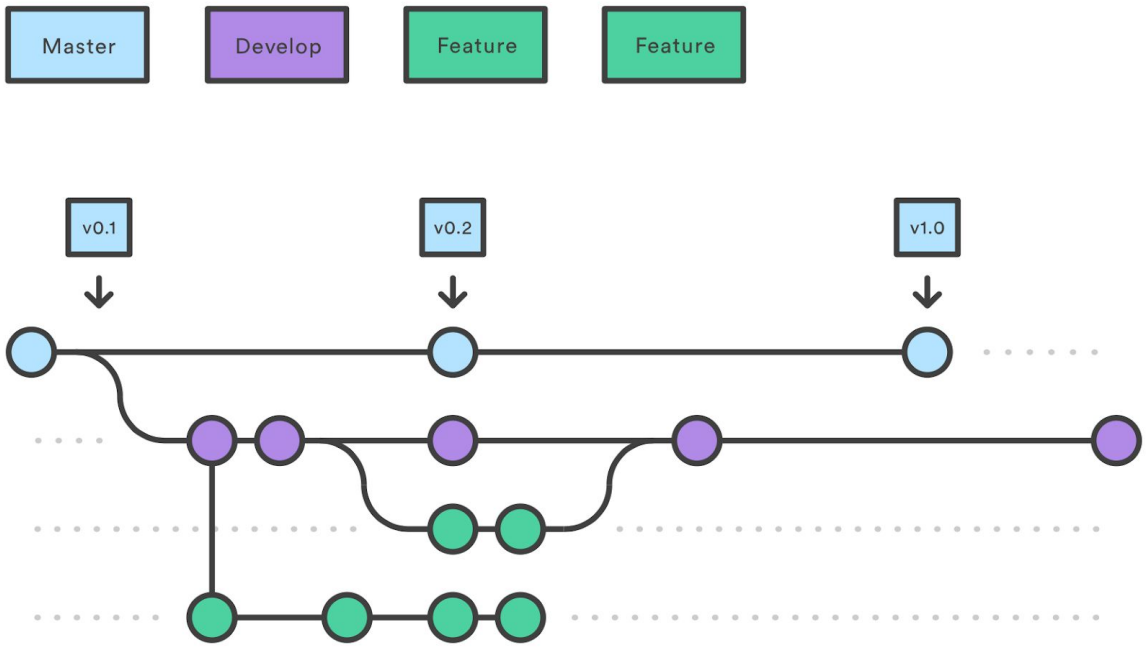

Fig. 11 - Stratégie GitFlow (*Sanchez, 2018)*

En ce qui concerne la rédaction de documentation, nous utiliserons Google Docs pour que les membres de l'équipe peuvent travailler en même temps sur les différents rapports que nous devrons présenter, comme cette réponse d'appel d'offres ainsi que le rapport d'avancement.

Le code source Kotlin de l'application Android sera documenté selon le standard KDoc qui combine le syntaxe JavaDoc et Markdown alors que les parties du code source côté serveur requérant des explications sera documenté selon le format Doxygen.

6. Références (Q3.2)

Hosson, C., St-Onge, P.R. et Collin, J. (2018). *Exigences techniques. Conception d'un système audio pour café Internet*. [EN LIGNE]. Tiré de : https://moodle.polymtl.ca/course/view.php?id=1703

Sanchez. J. (2018). *Gitflow*. [IMAGE]. Tiré de : https://blog.xebia.fr/wp-content/uploads/2018/03/Image.png

Team One, Inc. (2018). *INF3995-01*. [EN LIGNE]. Tiré de : https://redmine.gi.polymtl.ca/projects/inf3995-01