IFT3912 - Développement et maintenance de logiciels

Description du projet - Hiver 2013

Objectifs

L'objectif du projet est de développer en équipe une application web qui permet le partage de vidéos entre les utilisateurs du système (similaire à YouTube, Vimeo et Dailymotion). Ce projet permettra d'appliquer les divers concepts présentés dans le cours dans un contexte de développement d'un logiciel complexe.

Tâche à réaliser

L'application à développer sera principalement constituée d'un serveur HTTP spécialisé. Le système que vous devrez développer offrira une variété de fonctions à ses usagers. Le système supportera deux types d'usagers :

- Usagers anonymes : ne possèdent pas de compte, ou n'ont pas ouvert de session.
- Usagers authentifiés : possèdent un compte et ont préalablement ouvert une session.

Les usagers anonymes devront pouvoir :

- créer un compte incluant une photo et des informations personnelles (nom, prénom, adresse courriel, etc.).
- ouvrir une session, si l'usager possède un compte.
- regarder des vidéos ajoutés par les autres usagers. Chaque vidéo possèdera sa propre page qui affichera les détails du vidéo (titre, description, catégorie, usager, durée, étiquettes, commentaires). Un usager devra pouvoir télécharger un fichier vidéo à partir de cette page.
- consulter la liste des vidéos d'un utilisateur. Chaque utilisateur possèdera sa propre page qui affichera son nom complet, son nom d'usager, ses vidéos ainsi ses listes (voir plus bas).
- lancer la lecture d'une liste de vidéos. La lecture devra débuter par le premier vidéo dans cette liste, puis passer automatiquement au vidéo suivant lorsque le premier vidéo sera terminé, etc.
- consulter la liste des vidéos récemment ajoutés.
- consulter la liste des vidéos les plus souvent visionnés.
- rechercher des vidéos (la recherche utilisera au moins le titre et la description des vidéos pour identifier ceux qui correspondent à la requête).

Les usagers authentifiés devront **en plus** pouvoir :

- modifier les informations de leur compte (excluant le nom d'usager, qui peut être immuable).
- fermer une session.
- ajouter et supprimer des fichiers vidéo (seul le format WebM doit obligatoirement être supporté). Ces fichiers vidéo devront être enregistrés par l'application, et rendus disponibles au autres usagers du système (avec ou sans compte). Chaque vidéo doit posséder un titre (obli-

gatoire), une description (possiblement vide) et une catégorie (obligatoire). Il devra aussi être possible pour l'usager d'ajouter une ou plusieurs étiquettes (*tags*) à ses vidéos. Cliquer sur une des étiquettes devra afficher une liste de tous les vidéos possédant cette étiquette. Lors de l'ajout d'un vidéo, l'application devra aussi extraire une capture d'écran utilisée pour afficher un aperçu du contenu de vidéo dans le système. La description et les étiquettes devront pouvoir être modifiées ultérieurement par l'utilisateur.

- créer, modifier et supprimer des listes de vidéos. Une liste contient une séquence (possiblement vide) de vidéos dans un ordre spécifique déterminé par l'usager. Les vidéos doivent pouvoir être réordonnés dans une liste. Un vidéo peut appartenir à plusieurs listes à la fois. Un usager peut ajouter des vidéos personnels mais aussi des vidéos d'autres usagers à ses propres listes.
- commenter les vidéos disponibles dans le système. Les commentaires devront contenir au minimum du texte, une référence à l'usager ayant écrit le commentaire, ainsi que la date et l'heure de la création du commentaire. Il n'est pas nécessaire de permettre la modification des commentaires, ou l'imbrication des commentaires (ex: répondre à un commentaire).

Conditions de réalisation

Le projet sera réalisé en équipes de 4 ou 5 personnes. Vous avez jusqu'au 8 février (inclusivement) pour former les équipes. Vous devez désigner un membre de l'équipe pour compléter le formulaire d'enregistrement d'équipe (une seule soumission par équipe est nécessaire). Un lien vers ce formulaire sera disponible par StudiUM. Un courriel vous sera retourné avec un numéro d'équipe. Ce numéro devra être utilisé pour identifier votre équipe tout au long du semestre (par exemple lors de la remise des travaux). Lors de la création de vos équipes, vous aurez en plus à fournir un nom d'utilisateur github pour permettre la création d'un dépôt privé pour votre équipe.

L'organisation des équipes est laissée à la discrétion des membres. Il est possible de nommer un chef d'équipe pour la durée du projet, de nommer des chefs temporaires lorsque leur expertise est requise, ou de prendre toutes les décisions en commun (*egoless programming*), par exemple.

Le système devra être **obligatoirement** développé en Java (l'utilisation de l'outil de développement *Eclipse* est recommandée mais pas nécessaire). **L'utilisation d'autres langages de programmation tels que PHP n'est pas permise pour le développement du serveur web**. L'utilisation de JavaScript dans les pages générées est essentielle à la réalisation du projet, et est bien entendu permise. Les pages web générées par le serveur doivent être affichées correctement par une version récente du fureteur Firefox. **Firefox est le seul fureteur qui sera utilisé lors de l'évaluation de votre projet**.

La remise finale devra contenir tous les éléments nécessaires à l'installation et l'exécution du code (aucun élément ne devra être téléchargé pour l'exécution). Le code remis ne devra en aucun cas nécessiter des outils externes comme Eclipse ou NetBeans pour la compilation ou l'exécution. Similairement, l'application ne doit en aucun cas requérir une connexion avec une base de donnée externe ou charger des éléments à partir d'un serveur web (hébergé au DIRO ou ailleurs). Si vous désirez utiliser une base de données, il est nécessaire d'utiliser une base de données qui supporte les bases de donnés locales (*embedded*), comme H2 (http://www.h2database.com) ou Derby (http://db.apache.org/derby/). Vous ne devez **en aucun cas** utiliser un système de bases de données (ou tout autre composant) qui requiert l'installation

d'un serveur de base de données (ex: MySQL). Le code développé doit être portable : il doit pouvoir être exécuté sur toutes les plateformes, et sur de nouvelles machines sans configuration préalable.

Un serveur HTTP vous sera fourni (http://www.eclipse.org/jetty/), ainsi qu'une bibliothèque permettant de manipuler des fichiers vidéo (http://www.xuggle.com/xuggler). Un tutoriel sur ces bibliothèques sera présenté en classe, et vous aurez l'occasion de vous familiariser avec ces outils lors d'une séance de démonstration. L'utilisation de toute autre bibliothèque (à l'exception de JUnit) devra être préalablement approuvée par le professeur. Un squelette de projet sera aussi rendu disponible et devra obligatoirement être respecté pour faciliter l'évaluation des projets lors des remises. De plus, des fichiers vidéo de test seront mis à votre disposition. Si vous le désirez, vous pouvez aussi convertir vos propres vidéos en format WebM à l'aide d'outils tels que Miro Media Converter (http://www.mirovideoconverter.com/).

Un dépôt github privé sera fourni à chaque équipe. L'utilisation de github est requise pour le code du projet. Vous êtes aussi encouragés à utiliser le répertoire github pour vos documents, mais ceci n'est pas obligatoire. De plus, chaque équipe devra gérer ses tâches et son progrès à l'aide de l'outil en ligne Pivotal Tracker (http://www.pivotaltracker.com). Plus de détails seront donnés en classe.

Remises

Le développement du système sera effectué en plusieurs étapes. Chaque étape se terminera par une remise, qui sera ensuite noté et remis avec commentaires (si nécessaire). Plus de détails seront fournis pour chaque remise.

1. Prototype du système (15 février)

Vous devrez créer un prototype statique des pages web qui seront générées par votre système. Les hyperliens devront être fonctionnels, et les pages devront ressembler au produit final. L'apparence des pages devrait être professionnelle et permettre aux utilisateurs d'accéder facilement aux fonctions du système. Vous serez partiellement évalués sur la qualité de l'interface.

2. Plan de développement (22 février)

Un plan de développement devra être rédigé. Le plan comprendra les objectifs fonctionnels et non-fonctionnels du projet, et l'énumération des activités du projet sous forme de *Work Breakdown Structure* (*WBS*), un échéancier et la répartition des tâches entre les membres de l'équipe. Le plan de développement devra **obligatoirement** prévoir une classification des activités selon leur importance:

- Noyau : absolument essentiel au projet. Le noyau doit obligatoirement être complété par chaque équipe **en premier lieu**.
- Complet : Nécessaire pour obtenir un logiciel simple mais fonctionnel. Le système complet est celui que chaque équipe doit pouvoir compléter pour la remise finale.
- Supplémentaire: Les activités supplémentaires sont celles que vous aimeriez compléter si le temps le permet, mais qui ne sont pas essentielles au projet.

Au cours de la phase d'implémentation, vous devrez vous assurez de construire un noyau stable et utilisable en premier lieu. Les autres fonctionnalités seront ensuite progressivement ajoutées à ce noyau. Vous serez évalués sur cet aspect du développement dans une remise ultérieure. Il est très essentiel de compléter le système de base avant d'ajouter des fonctionnalités supplémentaires. Vous devez obligatoirement effectuer la remise finale du projet afin d'obtenir une note pour le projet en entier. Cette remise finale doit être fonctionnelle, même si elle est incomplète.

3. Architecture (1 mars)

Vous devrez décrire l'architecture générale de votre projet. Cette description sera accompagnée d'un squelette du code (classes et méthodes publiques avec commentaires Javadoc). Votre architecture sera évaluée et revue lors d'une courte rencontre au bureau du professeur.

4. Noyau stable (1 avril)

Vous devrez remettre une version stable du noyau avant la remise finale. Ce noyau devra être exécutable et utilisable, bien que plusieurs fonctionnalités seront évidemment absentes. Cette remise sera comparée à votre plan de développement afin d'évaluer le progrès de l'équipe.

5. Rapport de projet final et code (18 avril)

Le code devra être de bonne qualité (facile à lire et modifier, bien organisé, indenté proprement, etc.). Il devra aussi être bien documenté. Par exemple, chacune des classes du système devra contenir un entête indiquant le rôle de cette classe dans le système, ainsi que le nom de l'auteur de la classe. Chacune des méthodes devra aussi contenir de la documentation sous la forme de *Javadoc*, en particulier une description du rôle de la méthode et de chacun de ses paramètres. Vous devrez aussi développer des tests unitaires pour les différents modules de votre application à l'aide de JUnit.

Le rapport final comprendra, en plus du code et des tests, un rapport de synthèse du projet.

6. Présentation du projet en classe (22 avril)

Chaque équipe devra démontrer les fonctionnalités de son système devant les autres équipes en classe. Chaque membre de l'équipe devra présenter sa contribution individuelle.

7. Rencontres par équipe avec le professeur (dates à déterminer)

Suite à la remise finale, chaque équipe devra participer à une courte rencontre finale au bureau du professeur.

Évaluation

Des critères d'évaluation précis seront fournis pour chaque remise. Le projet sera noté comme suit :

10%	Prototype
10%	Plan de projet
10%	Architecture
10%	Noyau stable
50%	Rapport final et code
10%	Présentation

Notez que les notes finales pour le projet seront assignées *individuellement* et non par équipe, et l'évaluation tiendra compte de la contribution de chacun des membres. Il est donc possible pour les membres d'une même équipe d'obtenir des notes différentes pour la section projet du cours.

Retards

Les dates de remise sont **fermes**. Si votre équipe manque une remise, vous recevrez une note de **zéro** pour cette partie du projet. Dans un tel cas, l'équipe devra aussitôt rencontrer le professeur pour expliquer la cause du retard et déterminer une stratégie pour éviter d'éventuels futurs retards.

Plagiat

Les discussions avec les autres étudiants du cours sont encouragées (même avec les membres d'autres équipes), mais le plagiat sous toute forme sera sévèrement puni, conformément aux règlements de l'Université de Montréal (échec du cours et sanctions). Si vous utilisez du code dans votre projet qui n'est pas le vôtre, vous devez clairement indiquer sa provenance.